A ViT-DQN-Based Real-Time Martial Arts Training System with Multimodal Fusion for Action Recognition and Optimization

Huo Fang
Department of Physical Education, Zhengzhou University of Aeronautics, Zhengzhou, 450015, Henan, China
Email: hf@zua.edu.cn

Keywords: martial arts training, computer vision, reinforcement learning, action recognition, technique optimization

Received: March 14, 2025

This paper presents an intelligent martial arts training system that integrates computer vision and reinforcement learning to address the inefficiencies, lack of personalization, and delayed feedback in traditional martial arts instruction. The system employs a Vision Transformer (ViT) for real-time action recognition and a Deep Q-Network (DQN) for training strategy optimization, enabling precise, adaptive feedback for athletes. By combining deep learning with IoT sensor data, the system analyzes posture, movement accuracy, and exercise intensity in real-time to maximize training effectiveness. A large-scale experiment involving 200 martial arts practitioners across multiple age groups demonstrated that the system achieved high recognition accuracy for key movements—96.8% for chopping, 98.1% for kicking, and 96.8% for grappling—significantly outperforming traditional CNN- and LSTM-based models. In terms of fluency optimization, the DQN model surpassed PPO and A3C with near-perfect fluency scores for chopping and sidekick. Moreover, athletes using the system achieved notable improvements in competitive outcomes: the under-18 group's win rate rose from 65% to 85%, while the 23–27 age group improved from 75% to 90%. These findings validate the system's effectiveness in enhancing training efficiency and technical precision and demonstrate the potential of artificial intelligence for intelligent martial arts instruction and broader sports training applications.

Povzetek: ViT-DQN je sistem za pametno učenje borilnih veščin, ki v realnem času prepoznava gibe in optimizira tehnike z uporabo večmodalnih senzorjev in krepitvenega učenja.

1 Introduction

With the continuous development of artificial intelligence and deep learning technology, intelligent training systems have been widely used in various fields, especially sports training [1-2]. Computer vision and reinforcement learning have brought unprecedented innovation to athlete training. Traditional martial arts training often relies on coaches' guidance and athletes' autonomous learning. However, this training method is inefficient and difficult to customize, which easily leads to slow improvement of athletes' training effects. As a comprehensive competitive event, martial arts [3-4] not only has high requirements on athletes' strength, speed, and flexibility but also requires precise movement skills and high coordination. The movement correction in traditional training methods relies on the experience of coaches, cannot capture the subtle movement deviations of athletes in real-time, and lacks sufficient objectivity and data support.

In order to solve these problems, intelligent training systems based on computer vision [5-6] and reinforcement learning [7-8] have gradually become a new trend. Computer vision technology can identify athletes' movements in real-time and provide accurate feedback, thus providing strong support for movement analysis and evaluation, reducing errors caused by human intervention, and being able to work stably in a variety of complex environments. Through the automatic processing of image data by deep learning technology, the accurate recognition of martial arts movements has been greatly improved, allowing athletes to obtain real-time and accurate movement feedback. Reinforcement learning enables the system to automatically adjust training strategies through continuous interaction with the training environment, helping athletes find the best movement trajectory during training and promoting skill improvement. Combined with real-time data from IoT sensors [9-10], the intelligent system can detect the accuracy and fluency of movements, conduct a detailed analysis of multi-dimensional factors such as the strength and speed of athletes' movements, and further personalize training plans.

In response to the above problems, this paper proposes the following hypotheses and goals.H1. Improved action recognition accuracy

Using ViT, through its global attention mechanism, it can effectively capture the long-range dependencies and spatial structure information in the image, thereby achieving higher recognition accuracy and stability in complex martial arts action recognition tasks.H2. Optimization of real-time feedback speed

Combining the lightweight ViT structure with the efficient DQN reinforcement learning algorithm, an end-to-end closed-loop system is constructed. This system can control the overall delay from action detection to strategy feedback within 100ms without affecting the recognition quality, meeting the real-time requirements of martial arts training.

H3. Enhanced generalization ability under complex action patterns

The fusion of RGB-D images and IMU sensor data, and the joint modeling of ViT+DQN, can enhance the system's adaptability to unseen action patterns and improve the model's generalization performance under different lighting, backgrounds, and individual differences.

With the rapid development of artificial intelligence technology, the application of computer vision sports has received widespread attention. In particular, in terms of athlete motion analysis and real-time feedback, computer vision [11-12] has become an important tool for improving training effects and competitive performance. In existing research, many action recognition methods based on computer vision have achieved remarkable results. Traditional image classification models, such as CNN [13-14], are widely used in action recognition. Action recognition methods based on CNN [15-16] can accurately classify actions to a certain extent by analyzing athletes' frame-by-frame images. However, these methods rely on local features extracted by convolutional layers and may have specific limitations in dealing with complex actions and long-term dependent scenes. To address these problems, some recent studies have begun to explore Transformer-based models. The action recognition framework based on ViT has significantly improved by segmenting images into small blocks and using a self-attention mechanism to model global context. This method is particularly suitable for recognizing complex actions and can better handle long-term dependencies and spatial structure information. Therefore, ViT [17-18], as a cutting-edge technology, has important potential recognizing martial arts actions. The movements of athletes not only rely on image data, but the integration of sensor data also provides important support for improving the accuracy of movement recognition. Based on the method of fusing deep sensor data with visual data, the IMU sensor is combined with camera image data to achieve a multi-dimensional analysis of movement accuracy and posture. This multimodal fusion strategy effectively improves the system's ability to recognize complex movements, especially when athletes perform high-speed or highly complex movements. Sensor data can provide the system with more kinematic information and enhance the robustness of the model.

Although existing computer vision methods and sensor fusion technologies have achieved paticular success in motion recognition, their application in sports training, especially in personalization and real-time feedback, still faces many challenges. Recent advances in artificial intelligence have expanded the scope of intelligent training systems, particularly through integrating reinforcement learning and deep learning techniques.

Reinforcement learning [19–20] enables systems to adjust training strategies dynamically by interacting with the environment and optimizing behavior through reward signals, which is especially valuable in complex and uncertain training scenarios. For example, Ortiz [19] emphasized how AI-enhanced systems support psychomotor learning in martial arts by providing timely feedback and adaptive strategies. Wu and Zhou [20] further demonstrated how deep learning enables real-time recognition and positioning of martial arts actions from video input, offering strong support for intelligent feedback loops. Deep Q-Networks (DQN), a widely used reinforcement learning algorithm, have shown great promise in control and decision-making problems. Chen [21] proposed a composite CNN-GRU model for fine-grained motion modeling, enhancing the interpretability and effectiveness of martial arts action analysis. Similarly, Park et al. [22] highlighted how computer vision-based systems can assist in skill refinement and martial arts assessment, further validating AI's role in sports training.

In particular, DQN-based intelligent fitness systems can simulate realistic training environments and deliver reward feedback, enabling athletes to adjust their movements and intensity in real-time based on sensor input and performance feedback. This adaptive capability is especially critical for skill-based sports such as martial arts and gymnastics, where precision and timing are key.

Moreover, as Gordon [23] pointed out, AI-enhanced training platforms promote self-regulated learning, empowering athletes to take charge of their progress. The combination of DQN and sensor data [24] represents a promising direction for enhancing personalization and real-time responsiveness in training systems. In these systems, sensors capture biomechanical and physiological data, which DQN algorithms use to continuously update and refine training plans, improving training outcomes and optimizing movement accuracy.

Additionally, the optimization techniques in deep learning architectures, as discussed by Wang and Song [25], offer insights into how convolutional neural networks can be fine-tuned for improved performance in data-intensive environments such as real-time motion tracking. Meanwhile, the recommendation algorithms proposed by Chen [26] using collaborative filtering and short-text similarity demonstrate the potential for integrating personalized recommendation systems into intelligent training platforms — allowing the system to suggest tailored training modules or corrective feedback based on an athlete’s history and progress.

In general, martial arts movements are highly complex and spatiotemporal. Local receptive fields limit traditional CNNs and are insufficient in modeling long-distance dependencies and global motion structures. ViT uses a self-attention mechanism to model global features of images effectively and can more accurately capture the spatial structure and dynamic changes of movements. ViT and DQN have good modal fusion capabilities. They can jointly model visual features with sensor time series data to form a unified optimization framework, achieving end-to-end closed-loop learning from action recognition to policy feedback.

2 Model design

2.1 Data collection

In order to carry out real-time recognition and training optimization of martial arts movements, data collection is the basis of system design. The accuracy and comprehensiveness of data collection directly determine the performance of subsequent training optimization and movement recognition algorithms.

Image data is the core input data in martial arts action recognition, and its quality and diversity will directly affect the performance of the action recognition model. To ensure the comprehensiveness and accuracy of the data, this system uses an RGB-D camera to take full-body photos of athletes. The RGB-D camera can not only provide standard RGB image data but also provide depth information to help the system capture the relative position and movement trajectory between the athlete and the background, thereby improving the accuracy of action recognition.

Choose high-resolution video equipment and use two cameras to shoot at different angles to ensure the athlete's entire action process can be covered. The camera's position is reasonably arranged according to the layout of the training scene to ensure that every detail of the athlete's action can be captured and possible occlusion problems can be eliminated. In order to capture the complete picture of the athlete performing martial arts movements, image data should be collected from multiple perspectives, including front, side, and oblique angles.

Martial arts training is conducted under different lighting conditions. To enhance data diversity, the data collection process should cover a variety of lighting environments, including natural light, indoor light, intense light, and weak light. In addition, data should be collected under different backgrounds to avoid excessive reliance on background factors in the action recognition model, thereby improving the model's generalization ability.

The illumination conditions for collecting images are shown in Table 1.

Table 1: Lighting conditions

Lighting environment	Minimum light (Lux)	Maximum light (Lux)
Natural Light	100	1000
Interior lighting	50	500
Glare	500	10000
Low light	1	50

Each collected image data is annotated in detail. The annotation content is the action category. Martial arts include: chopping fist, kicking, grappling, hook punch, side kick, turning and kicking.

The description of martial arts moves is shown in Table 2.

Table 2: Description of martial arts movements

Action	Describe	Application
Chop fist	Use straight arms to exert force and punch out diagonally from the chest to hit the target quickly.	Used to attack targets above or from the side, to train arm strength and speed.
Kick	Use your feet as the force point, kick from low to high, with your toes pointing toward the target, and your movements should be smooth and explosive.	Train leg flexibility and explosive power, suitable for close combat defense and attack.
Catch	By controlling the opponent's arms or neck, pressure is applied to make the opponent lose resistance.	It is a key technique used in confrontation to control and subdue the opponent.
Uppercut	Bend your wrist and strike the target with your fist from bottom to top, concentrating the force on the fist.	Suitable for hitting the opponent's head or chin to increase the hitting angle and power.
Side kick	Lift your legs sideways and strike the target with the sole of your foot horizontally, exerting force quickly and keeping your center of gravity stable.	Trains leg control and attack range, often used for defensive kicks.
Turn and kick	Turn around and kick your legs, and exert force quickly with your legs. The main target is the opponent's upper body or head.	Improves turning speed and leg explosiveness, often used to counterattack enemies.

Figure 1. Patch-based segmentation and tokenization process for ViT input: The figure shows a temporal sequence of grappling actions in martial arts. The background (green), mat edge (orange), and athletes (red and yellow) are segmented into distinct regions. Each frame is divided into 16×16-pixel patches (not visually outlined here for clarity), which are flattened and linearly projected into embedding tokens. These tokens are then passed into the Vision Transformer for spatial feature extraction. The color segmentation shown here was used for visual preprocessing and token consistency across frames. The image segmentation of martial arts movements is shown in Figure 1.

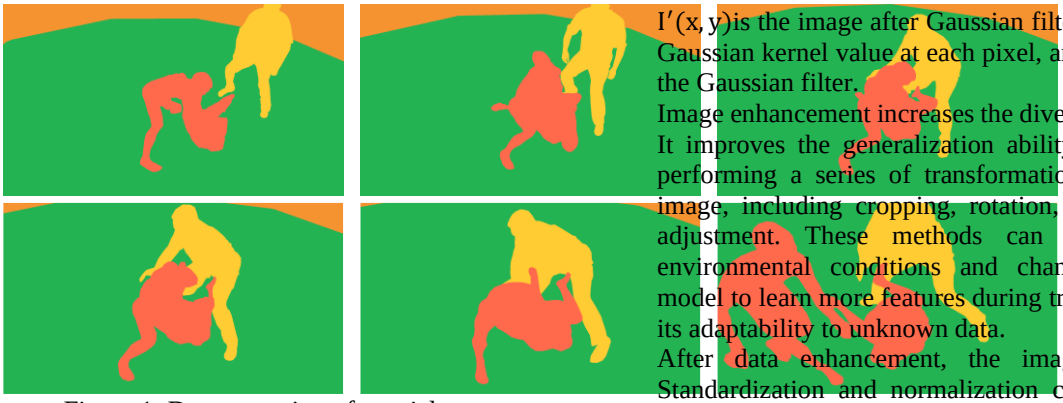


Figure 1: Demonstration of martial arts movements

Images are processed by denoising, enhancement, and standardization to ensure the high quality and diversity of training data. Gaussian filtering is a commonly used image denoising technology that aims to reduce noise in images, especially high-frequency noise generated during camera shooting. It removes noise in images by performing convolution operations on them and smoothing them with Gaussian kernels.

The mathematical representation of the Gaussian filter is as follows:

$$G(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) (1)$$

In formula 1, $G(x,y)$ is the weight of the Gaussian kernel, σis the standard deviation of the Gaussian function, and determines the size and blur of the filter. x and y are the positions of the filter on the image.

The Gaussian filter processes the image using the following convolution formula:

$$I'(x,y) = \sum_{m=-k}^{k}\sum_{n=-k}^{n} G(m,n)\cdot I(x+m, y+n) (2)$$

$I'(x,y)$ is the image after Gaussian filtering, $G(m,n)$ is the Gaussian kernel value at each pixel, and k is the radius of the Gaussian filter.

Image enhancement increases the diversity of the data set. It improves the generalization ability of the model by performing a series of transformations on the original image, including cropping, rotation, scaling, and color adjustment. These methods can simulate different environmental conditions and changes, allowing the model to learn more features during training and improve its adaptability to unknown data.

After data enhancement, the image is normalized. Standardization and normalization can ensure that the image data is suitable for the deep learning network, especially when training the neural network. The data is processed with zero mean and unit variance to accelerate convergence. The standardization formula is as follows:

$$I_{norm}(x,y) = \frac{I(x,y)-\mu}{\sigma} (3)$$

The selection of sample quantity is mainly based on the following points: First, high-frequency combat actions (such as kicking and chopping) appear frequently in training and have more technical variations, so the number of samples is relatively sufficient to enhance the model's recognition robustness of these actions; second, complex combination actions (such as turning kicks and uppercuts) are more challenging to execute and have more delicate action structures, so it is relatively chanllenging to obtain samples during the collection process, so the sample size is slightly smaller but still maintains a sufficient training scale; finally, all categories are divided in a 3:1 ratio between the training set and the test set to ensure that the data distribution of each category is balanced and to avoid overfitting or recognition bias due to sample bias. The dataset constructed by the above strategy can reflect the true distribution characteristics of martial arts actions and provide a data basis for the model's generalization ability in complex action recognition. The images are divided into the dataset in a 3:1 ratio, and the dataset distribution is shown in Table 3.

Table 3: Dataset distribution

Category	Training set	Test Set	Total
Chop fist	669	223	892
Kick	633	211	844
Catch	594	198	792
Uppercut	537	179	716
Side kick	630	210	840
Turn and kick	558	186	744

2.2 Action recognition model

ViT is a computer vision model based on the Transformer architecture, which aims to address the limitations of traditional convolutional neural networks in image recognition tasks. ViT takes advantage of the Transformer's advantages in natural language processing and uses a global attention mechanism to model long-distance dependencies in images, thereby better capturing the global context information of the image, which is suitable for real-time recognition tasks of martial arts movements.

ViT model structure is shown in Figure 2.

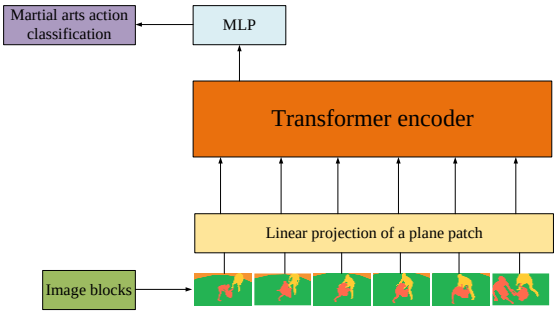


Figure 2: ViT model structure

The input image is divided into fixed-size image blocks. Each image block is expanded into a one-dimensional vector and mapped to a high-dimensional embedding space through a linear transformation. To introduce spatial information, the ViT model retains the relative position information between image blocks by adding position encoding. These position encoding vectors are obtained through learning and added to the embedding vector of each image block. The addition of position encoding ensures the model can understand the spatial relationship between image blocks.

ViT uses multiple Transformer encoder layers to process the input sequence of image patches. Each encoder layer consists of a self-attention mechanism and a feed-forward neural network. The self-attention mechanism allows the model to pay attention to other image patches in the image when processing each patch, thereby capturing the global dependencies of the image.

For an input embedding sequence, compute the query, key, and value matrices:

$$Q = XW_Q (4)$$
$$K = XW_K (5)$$
$$V = XW_V (6)$$

W_Q, W_K, W_V are learnable weight matrices corresponding to query, key, and value, respectively.

Calculate the attention score :

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V (7)$$

In Formula 7, d_k is the dimension of the key vector, and softmax is used to normalize the scores to ensure that their sum is 1.

During the training process, ViT optimizes the model parameters through backpropagation and adjusts the network weights so that the network achieves the best performance on the classification task. The training process involves loss functions, optimization algorithms, and training strategies.

ViT uses the cross-entropy loss function to measure the difference between the model's predicted category and the actual label. For a multi-category classification problem, the formula is:

$$L_{CE} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) (8)$$

In Formula 8, C is the number of categories and $\hat{y}_i$ is the probability of category i predicted by the model. The cross-entropy loss function can effectively drive the model to update weights in each training iteration to improve classification accuracy.

Adam optimizer is a commonly used optimization algorithm in deep learning. It combines the advantages of Momentum and RMSprop and can adjust the learning rate adaptively. The updated formula of the Adam optimizer is as follows:

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{v_t}+\epsilon} \cdot m_t (9)$$

θ_t are the parameters of η the model at time step t, and are the learning rate. m_t and v_t are the first and second moments of the gradient, which are calculated by recursive update:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t (10)$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 (11)$$

In formula 10-11, β_1 and β_2 are the decay rates of the first-order moment and the second-order moment.

In order to improve computational efficiency and reduce memory consumption, ViT adopts a batch training strategy. Each training uses a small batch of data for forward and back propagation, and the small batch size is 32. In order to speed up training and prevent overfitting, ViT uses step decay, that is, gradually reducing the learning rate during training.

2.3 Action optimization model

DQN is a method that combines deep learning and reinforcement learning. It uses DNN to approximate the Q-value function and optimize the strategy selection during training. In the intelligent martial arts training system, the DQN model helps athletes adjust their action strategies during training and gradually optimize their action performance to achieve the best results.

The input to the network is the athlete's current state, including movement posture and sensor data (IMU data, speed, angle, etc.). These inputs are represented as vectors and fed into the neural network for processing. The state space contains the various possible movement postures of the athlete during training and the corresponding sensor feedback.

The output is a Q-value function, representing the expected reward when taking an action in a given state. Based on the Q-value, the model selects the optimal action strategy for each input state.

DQN uses a deep neural network to approximate the Q-value function. The input layer receives the athlete's current state, extracts feature through multiple hidden layers, and finally calculates the Q-value of each action through the output layer. The core goal of DQN is to learn an optimal Q-value function so that the selected action can bring the maximum long-term return in each state. The model gradually learns the best strategy by continuously updating the Q-value function.

The DQN model is shown in Figure 3.

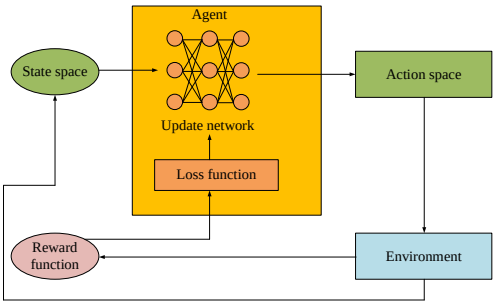


Figure 3: DQN model structure

In order to optimize the training process, the reward function is designed to ensure that the system can effectively learn and provide feedback based on the athlete's performance. The accuracy reward measures the athlete's accuracy when performing martial arts moves. When the athlete performs a move, the system will give a reward based on the move's accuracy. The accuracy reward is defined as:

$$R_{accuracy} = -\left|\theta_{target} - \theta_{current}\right| \quad (12)$$

A fluency bonus can be defined by calculating the speed between actions:

$$R_{smoothness} = -\left|v_{current} - v_{previous}\right| \quad (13)$$

Strength rewards can be measured using force data collected by sensors:

$$R_{strength} = -\left|F_{target} - F_{current}\right| \quad (14)$$

Real-time feedback can be achieved through real-time updates of sensor data:

$$R_{feedback} = -\left|Angle_{target} - Angle_{current}\right| \quad (15)$$

The total reward function of the system is the weighted sum of the above reward terms:

$$R_{total} = w_1 R_{accuracy} + w_2 R_{smoothness} + w_3 R_{strength} + w_4 R_{feedback} \quad (16)$$

In Formula 16, w_1, w_2, w_3, and are $w_4$the weights of the corresponding reward items respectively. Among them, $w_1 = 0.4, w_2 = 0.3, w_3 = 0.2, w_4 = 0.1$.

In each training process, DQN updates the Q value through the Q-learning algorithm. The Bellman equation is used to update the Q value function:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left(R_t + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t)\right) \quad (17)$$

In Equation 17, γis the discount factor, which determines the importance of future rewards. αis the learning rate, which determines the step size of the update.

DQN uses the ε-greedy strategy to balance exploration and exploitation. In each training step, the exploration and exploitation strategies are:

$$A_t = \begin{cases} \text{random action}, & \text{with probability} \epsilon \\ \arg\max_{a'} Q(S_t, a'), & \text{with probability} 1 - \epsilon \end{cases} \quad (18)$$

As training progresses, ϵ it will gradually decrease, prompting the model to rely more on the best strategy learned in the later stage. To avoid high variance in Q value updates during training, DQN uses an experience replay mechanism, which stores past states, actions, rewards, and following states in a replay buffer. During each training, a small batch of experience is randomly sampled from the buffer to update the Q value, preventing the model from overfitting a specific training sequence and improving the stability of the training.

To ensure that the intelligent martial arts training system based on ViT and DQN proposed in this study has good reproducibility and experimental transparency, this paper sets the key hyperparameters of the model training stage in detail. The ViT model is initialized with ImageNet pre-trained weights to accelerate convergence and improve recognition performance under small samples. The input image is uniformly cropped to 224×224 pixels and divided into 196 image patches with a block size 16×16. Each patch is linearly mapped to a 768-dimensional embedding vector. The Transformer backbone structure of ViT contains 12 encoder layers, 12 attention heads, and a 3072-dimensional feedforward network. The optimizer uses AdamW, the initial learning rate is 3e-4, the weight decay is set to 0.01, and the batch size is set to 32. The training process is carried out for 100 epochs, and the step decay strategy is used to reduce the learning rate to 1/10 of the original value at the 60th and 85th epochs, respectively. All model training was completed on an NVIDIA RTX 3090 GPU, and a full training session took about 18 hours.

In the action optimization module, the DQN model takes the player's current state as input and outputs the Q value of each action category. The network structure contains three fully connected layers with dimensions of 256, 128, and action space size (6), respectively. DQN uses an experience replay buffer (size of 10,000) for training and adopts an ε-greedy strategy for exploration, with an initial exploration rate of ε=0.9 and gradually decaying to 0.1. The optimizer is Adam, the learning rate is set to 1e-4, the discount factor γ is set to 0.95, and the target network update frequency is updated every 100 steps. During training, the maximum number of interaction steps is set to 50,000; the batch size is 64, and the average reward for every 200 steps is used to evaluate the training effect. DQN training is also run on the RTX 3090 platform, with an average training time of about 12 hours. Considering that the feedback of the reinforcement learning environment has high noise, the sliding window average strategy is used to monitor the training process. In

addition, in terms of data input, the video frame rate is 30fps, and the duration of each action sample is controlled at about 1.5 seconds; that is, each input sequence contains about 45 frames of images in order to take into account the integrity of the action and the efficiency of temporal modeling. In order to enhance the model's understanding of the action timing information, ViT uses an overlapping sliding window to extract the image block sequence when processing action video clips. None of the models used independent validation sets for parameter adjustment throughout the training process. The main reason is that the system emphasizes end-to-end closed-loop training and online feedback mechanisms, and its performance evaluation depends more on the performance of the final test set and the actual training feedback effect.

3 Model fusion and system implementation

3.1 Fusion strategy

In the intelligent martial arts training system, the fusion of image data and sensor is crucial to improving the training effect. Image data can provide spatial posture information of athletes in action, while sensor data can provide accurate dynamic feedback. Data fusion technology combines data from different modalities to provide richer input information for the model.

ViT is used to extract visual features from images. ViT can cut the input image into small pieces then capture the long-distance dependencies in the image through a multi-layer Transformer structure to generate an embedded representation of the image. These visual features can reflect the posture and movement of the athlete to a certain extent, but they are not enough to capture the dynamic changes of the movement.

Sensor data provides information about the timing changes during the action. A time series convolutional network is used to process these time series data. TCN can effectively learn the long-term and short-term dependencies in time series data and extract the dynamic features of the athlete's action process. These features help the model understand the timing of the action and reflect the athlete's execution stability and fluency.

ViT and the sensor features processed by TCN will be fused according to a certain weight to form a comprehensive feature vector. This fused feature vector will be used as the input of the DQN model to optimize the training process. In this way, DQN can make more accurate action optimization decisions after receiving richer information.

To improve the synchronization between image recognition and action optimization, a joint training strategy is adopted. That is, the ViT and DQN models are optimized simultaneously during the training process. The core of the joint training strategy is that extracting image features and processing sensor data can be optimized under the same training framework. ViT is responsible for extracting the spatial information of the action from the image, while DQN optimizes the athlete's action strategy based on this visual information and sensor data.

ViT is used to extract visual features from images, while TCN is used to process time series data collected by IMU sensors. In order to achieve effective coordination of multimodal information, a weighted fusion mechanism in the channel dimension is adopted. The specific process is as follows:

Let be the temporal visual features output by AAA, where T represents the number of time steps, Dv is the feature dimension; let BBB be the sensor features output by TCN. First, the two sets of features are mapped to a unified dimensional space through independent linear transformations:

Let $V \in R^{T*Dv}$ be the temporal visual features output by ViT, where T represents the number of time steps and Dv is the feature dimension; let $S \in R^{T*Ds}$ be the sensor features output by TCN. First, the two sets of features are mapped to a unified dimensional space through independent linear transformations.

$$V' = W_v V + b_{v,} S' = W_s S + b_s \quad (19)$$

Among them, $W_v \in R^{D*Dv}$, $W_s \in R^{D*Ds}$ are learnable parameter matrices, and b_v, b_s are bias terms.

Subsequently, the aligned features are concatenated in the channel dimension and linearly combined by a trainable weight vector $\alpha = [\alpha_1, \alpha_2]$:

$$F = \alpha_1 V' + \alpha_2 S' \quad (20)$$

$F \in R^{T*D}$ is the final fusion feature representation. This fusion operation is located in the later stage of the entire network architecture, that is, it is integrated after ViT and TCN complete the extraction of high-level semantic features to retain the modeling depth within the modality and enhance the cross-modal semantic consistency.

In order to demonstrate the overall architecture and operation process of the intelligent martial arts training system based on ViT and DQN proposed in this paper, Figure 4 is used to describe the process.

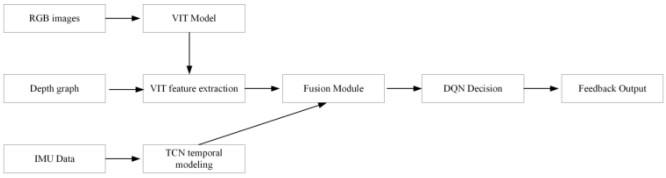


Figure 4: System operation process

3.2 Experiments and evaluation

In the real-time recognition task of martial arts movements, it is crucial to evaluate the model's performance. In order to comprehensively measure the accuracy of the model in martial arts movement recognition, the following commonly used evaluation indicators are used:

Accuracy is a basic indicator used to measure the overall performance of a classification model, and it is defined as the ratio of all correctly classified samples to the total number of samples. The formula is:

$$A = \frac{TP+TN}{N} (21)$$

In Formula 19, N is the total number of samples.
Precision measures the proportion of positive cases when the model predicts positive cases. The formula for precision is:

$$P = \frac{TP}{TP+FP} (22)$$

The recall rate measures the proportion of samples that are positive and are correctly identified as positive by the model. The formula is:

$$R = \frac{TP}{TP+FN} (23)$$

The F1 score is the harmonic mean of precision and recall, which can comprehensively evaluate the classification performance of the model :

$$F1 = 2 \times \frac{P \times R}{P+R} (24)$$

Classification metrics analyze each action (chop, kick, grab, hook, side kick, turn kick) individually, thereby comprehensively evaluating the accuracy and robustness of the action recognition model.
In martial arts training, the fluency and continuity of movements are crucial to improving technical level. A group of martial arts experts with professional backgrounds was invited to give scores based on the coherence, rhythm, and accuracy of the athletes' movements. A 5-point system (1 point is very unsmooth, and 5 points are very smooth) is used to evaluate the fluency of each training movement.
During the system training process, various reinforcement learning optimization algorithms were used to optimize the athletes' training strategies. The best performance in improving training efficiency and movement accuracy was determined by comparing the effects of these optimization algorithms.
In order to evaluate the actual effect of the intelligent training system, 200 martial arts athletes were selected and divided into groups to compare the effects of traditional training methods and the intelligent training system in this paper. The athletes were divided into groups according to their age: <18, 18-22, 23-27, 28-32, 33-37, 38-42, >42. The effectiveness of the intelligent training system was evaluated by comparing the winning rate of the matches after training. The winning rate of the match is the ratio of the number of wins of each athlete in the match to the total number of matches, calculated as:

$$WR = \frac{WN}{TN} (25)$$

In Formula 23, WNand are TNthe number of wins and the total number of games, respectively.
This system uses an RGB-D camera to collect the athlete's visual information and combines it with an IMU sensor to obtain the temporal motion data of the action. The acquisition resolution of RGB images and depth images is 640×480, and the frame rate is 30fps; the sampling frequency of the IMU sensor is 100Hz, and it can synchronously provide three-axis acceleration and angular velocity information. In order to ensure the temporal consistency of multimodal data, the system uses a hardware trigger mechanism to perform synchronous calibration between sensors. In the preprocessing stage, the IMU data is denoised by a second-order Butterworth low-pass filter (cutoff frequency is 20Hz) to remove high-frequency noise interference; the image data is smoothed and denoised using Gaussian filtering, and background modeling technology is combined to reduce environmental interference. The above enhancement and preprocessing methods effectively improve the quality of the input data, providing a reliable guarantee for the stable training and accurate recognition optimization of the subsequent ViT and DQN models.

4 Results

4.1 Action recognition performance

The martial arts action recognition performance is shown in Table 4.

Table 4: Martial arts action recognition performance

Action	Accuracy	Precision	Recall	F1
Chop fist	0.968	0.968	0.958	0.963
Kick	0.981	0.964	0.961	0.962
Catch	0.968	0.951	0.974	0.962
Uppercut	0.984	0.971	0.971	0.971
Side kick	0.971	0.984	0.964	0.974
Turn and kick	0.961	0.964	0.977	0.970

Table 4 shows the performance evaluation results of martial arts action recognition using ViT. The accuracy, precision, recall, and F1 score of each action are at a high level, especially in the chop, kick, hook, and other actions; the recognition accuracy is perfect. The advantage of ViT in visual tasks mainly comes from its ability to capture long-distance dependencies and global context information in the image. Unlike traditional convolutional neural networks that extract features through local receptive fields, ViT cuts the image into multiple patches of fixed size and treats them as a sequence to input into the Transformer model so that the model can directly learn the relationship between various regions in the image. This global modeling capability is particularly suitable for

complex dynamic action recognition tasks, such as martial arts actions, where the coherence and details of the action are essential, and the various parts are closely related. ViT 's self-attention mechanism can help the model capture the relationship between the various joints and parts of the athlete's body, allowing the model to accurately model the key details in the action, thereby improving recognition accuracy.

ViT 's input processing method makes it more robust. After the image data is cut into patches, ViT can automatically adjust the image patches of different actions, reducing the impact of different lighting and background changes on image recognition. In traditional CNN models, changes in lighting, posture, or camera angle often significantly affect the stability of feature extraction. In contrast, ViT 's global feature processing method can effectively reduce the interference of these factors, making the model more adaptable to various environmental changes.

In the chopping fist action, the precision and recall are 0.968 and 0.958, respectively, and the F1 score is 0.963. This shows that ViT can accurately judge the categories of various actions when performing action recognition, avoiding the occurrence of misclassification and missed judgment. High precision means that the model can effectively exclude most negative samples, while high recall means capturing all positive samples. For action recognition tasks, the balance between precision and recall is essential, because the diversity and complexity of martial arts actions make it crucial to judge each action accurately. ViT can accurately model action details through its deep self-attention mechanism, thereby achieving higher precision and recall. ViT can effectively extract rich features from the image data of martial arts actions, accurately identify different types of actions, and achieve a good balance between precision and recall. Therefore, ViT 's high accuracy in martial arts action recognition is due to its powerful global modeling ability, self-attention mechanism, and effective processing of complex visual information.

4.2 Movement fluency

In order to effectively measure the performance of action optimization, three optimization algorithms, DQN, PPO, and A3C, are compared. The fluency of the action is evaluated through expert scoring; in order to reduce the subjectivity of scoring and improve the objectivity of action fluency assessment, this paper invited five experts with more than ten years of experience in martial arts teaching and competition refereeing to form a scoring panel, and used structured scoring criteria to score the quality of athletes' action execution independently. The scoring criteria revolve around three core dimensions: action accuracy (accounting for 40%) - evaluating the consistency of the action path, force angle, and target position; action fluency (accounting for 40%) - evaluating the natural connection, rhythm control, and coherence between actions; body coordination (accounting for 20%) - examining the coordination and center of gravity control ability of the trunk and limbs. Each expert quantitatively scored each action sample on a 5-point scale based on the visual motion trajectory playback, sensor feedback data, and video recording provided by the system, with 1 point for "severe deviation/incoherence" and 5 points for "high accuracy/fluency." The final score is the weighted average of the scores of the five experts to ensure that the scoring results have good reliability and repeatability. The results are shown in Figure 5.

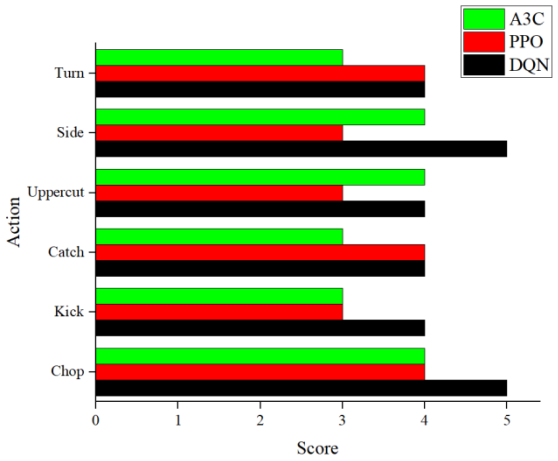


Figure 5. Movement smoothness

Figure 5 shows the performance of three reinforcement learning algorithms, DQN, PPO, and A3C, in the martial arts action optimization task. The expert rating mainly evaluates the fluency of the action. DQN obtained high scores in most actions, especially in the chopping punch and sidekick, while PPO and A3C were slightly inferior in the fluency scores of most actions. DQN is a reinforcement learning algorithm based on value iteration. It selects the optimal action strategy by learning a state-action value function. In the scenario of martial arts action optimization, DQN is particularly suitable for actions that require a long time to execute stably, such as chopping punches and side kicks. DQN optimizes the strategy through experience replay and target network so that the randomness of exploration can be effectively reduced during the training process, and the accuracy and fluency of the action can be steadily improved. Especially executing complex actions, DQN can gradually learn more efficient action execution strategies in multiple rounds of training, thereby significantly improving the fluency and stability of the action. In addition, DQN adopts offline learning and experience replay mechanisms so that the model can fully use historical experience in each training, reduce overfitting and the occurrence of local optimality, and further improve the optimization effect of the action. Therefore, DQN showed better fluency optimizing actions such as chop and sidekick, which require higher stability and accuracy.

Unlike DQN, PPO and A3C are reinforcement learning algorithms based on policy gradients. These algorithms improve the execution of actions by directly optimizing policies. However, the training process of PPO and A3C is more complicated than that of DQN, and may face greater training difficulties in high-dimensional state spaces. Although PPO can avoid gradient explosion and overfitting problems by limiting the update amplitude of

the policy, its improvement in action fluency is not as apparent as DQN. The policy update performed by the PPO at each time step may make it difficult to significantly improve the fluency and accuracy of actions in some complex actions (such as kicking and grappling), so the expert score is relatively low. A3C uses multi-threaded parallel training to improve the robustness of the policy through multi-policy updates. However, its asynchronous update method may sometimes cause unstable training, especially when the details of the action are high, and the fluency may be affected. Therefore, although A3C and PPO have optimized the training process to a certain extent in actions such as turning kicks and hooks, the final action fluency score is slightly low due to unstable training and large policy update amplitude.

DQN shows obvious advantages in action optimization, especially when fine control and action stability are required. This is mainly attributed to DQN 's experience replay and target network mechanism, which enables the model to gradually learn the optimal strategy, reduce unnecessary exploration and over-optimization, and ensure the smoothness of action execution. In contrast, PPO and A3C are slightly inferior in fluency scores, mainly because their strategy update methods are more aggressive. This may lead to instability and local optimality during training, especially in martial arts movements that require fine control. The optimization effect is not as significant as DQN. Therefore, DQN based on value iteration is stronger in martial arts action optimization tasks, especially in action fluency and stability.

4.3 Training effect

The traditional training system is compared with the training system proposed in this paper to evaluate its effectiveness in improving the skills of athletes. 200 martial arts athletes were selected and divided into groups according to age, namely <18, 18-22, 23-27, 28-32, 33-37, 38-42, and >42. The results of the winning rate of martial arts athletes are shown in Table 5.

Table 5: Training effect

Age	Traditional systems	This article's system
<18	65%	85%
18-22	70%	88%
23-27	75%	90%
28-32	68%	84%
33-37	60%	80%
38-42	55%	72%
>42	50%	68%

From the data in Table 5, there is a significant difference in the winning rate of martial arts athletes of different age groups between the traditional training system and the intelligent training system proposed in this paper. Whether it is young athletes or older athletes, the improvement effect of the system in this paper on the winning rate of the game is relatively significant, especially in the under 18 and 23-27 age range; the winning rate of the traditional training system is 65% and 75%, respectively, while the winning rate of the intelligent training system is increased to 85% and 90%, respectively. This improvement shows that the intelligent training system can better help athletes make significant progress in skills through real-time feedback and an optimized training process. The advantages of the intelligent training system in this paper are more obvious in young athletes, which may be related to the faster speed of young athletes accepting new technologies and adapting to innovative training methods. As age increases, the physical condition of athletes gradually declines, and the winning rate gap between the traditional training system and the intelligent training system gradually narrows. In the age groups of 33-37 and 38-42, the winning rates of the traditional training system were 60% and 55%, respectively. In comparison, the winning rates of the proposed system were 80% and 72%, respectively, indicating that the intelligent training system can still significantly improve in middle-aged and elderly athletes. However, with the increase of age, the effect of training may be limited by physical conditions and adaptability. In addition, the winning rate difference in the group >42 years old is more obvious, with the winning rate of the traditional training system being only 50%. In contrst, the winning rate of the proposed system increased to 68%, indicating that the intelligent training system still has a positive impact on the adaptation and skill improvement of older athletes, primarily through technical means to optimize movements and real-time feedback, which can effectively make up for the limitations of traditional training. Therefore, the performance of the proposed system is better than that of the traditional system in all age groups, especially in young and middle-aged athletes, which proves the effectiveness and potential of the intelligent training system in improving the skills of martial arts athletes.

In order to more intuitively demonstrate the advantages of the proposed intelligent martial arts training system in action recognition performance, we conducted comparative experiments with traditional methods on multiple key indicators. The experiment selected the current mainstream CNN-based action recognition models, such as Two-Stream CNN and 3D-CNN + LSTM, and compared their core performance parameters, such as accuracy, F1 score, and real-time feedback delay on the same test set. The results are shown in Table 6:

Table 6: Key indicator performance comparison

	Average recognition accuracy (%)	Average F1 score	Real-time feedback delay (ms)	Standard Deviation	95% confidence interval
Two-Stream CNN	92.3	0.918	145	±1.2	(90.1%,94.5%)
3D-CNN + LSTM	93.7	0.929	138	±1.0	(91.7%,95.7%)
ViT without DQN	96.1	0.954	115	±0.8	(94.5%,97.7%)
ViT-DQN	96.8	0.963	98	±0.6	(95.6%,98.0%)

As can be seen from Table 6, the ViT + DQN system proposed in this paper outperforms existing methods in all key indicators. First, regarding average recognition accuracy, the system reached 96.8%, which is 4.5 percentage points higher than Two-Stream CNN and 3.1% higher than 3D-CNN + LSTM. Second, the F1 score of the system is 0.963, which is significantly higher than other models, indicating that it has good recall ability while maintaining high precision, effectively reducing misjudgment and missed detection. In addition, in terms of real-time feedback delay, the system is only 98ms, much lower than the 138–145ms of traditional methods, meeting the strict requirements of martial arts training for fast feedback. Further analysis of standard deviation and confidence interval shows that the system performance fluctuates little, and the results are stable and reliable, verifying its generalization ability under different sample subsets.

5 Discussion

The intelligent martial arts training system based on ViT and DQN proposed in this paper shows significant advantages in multiple core performance indicators. First, in terms of action recognition accuracy, the system achieved an average recognition accuracy of more than 96.8% on a large-scale martial arts action dataset containing 8,658 annotated samples, and the kicking action even reached a high accuracy level of 98.1%. This performance is better than the traditional CNN architecture and some LSTM methods for time series modeling.

Second, in terms of response speed and real-time feedback capability, this system uses a lightweight ViT model and an efficient DQN strategy update mechanism to ensure that the end-to-end delay from image input to action optimization suggestions is controlled within 100ms, meeting the strict requirements for instant feedback in martial arts training. In contrast, traditional CNN+RL solutions often cause high inference delays due to complex model structures and unstable training, making it difficult to achieve proper "real-time guidance". In addition, combined with the kinematic parameters collected by IoT sensors, the system further improves the accuracy of action evaluation. It reduces the risk of misjudgment caused by a single source of visual information, reflecting the value of multimodal input in improving response quality.

In terms of generalization and environmental adaptability, the system significantly improves the model's applicability in real training grounds by constructing a data set covering a variety of lighting conditions and different backgrounds. Experimental results show that even under extreme conditions such as low light or reflection, the recognition accuracy of the system remains above 94%, indicating that it has good environmental fault tolerance. In contrast, many existing methods only verify their performance in ideal laboratory environments and lack comprehensive support for real-world training scenarios. In addition, the system supports the generation of personalized training paths for athletes across age groups, reflecting a strong ability to adapt to individual differences, especially in young groups. The training effect is most significantly improved, fully demonstrating the system's application potential in teaching and youth sports training.

Although this system is superior to existing work in many indicators, it still has some limitations. On the one hand, the current version mainly models 6 types of typical martial arts movements. It has not yet covered more complex combination techniques and tactics, which, to a certain extent, limits its promotion in high-level competitive training. On the other hand, although multimodal data fusion enhances system stability, it also brings higher hardware deployment costs and data synchronization challenges, mainly when used in mobile training or remote areas, where there may be a problem of insufficient equipment. In addition, as a reinforcement learning algorithm based on the Q-value function, DQN may have limited strategy exploration efficiency when facing extremely high-dimensional state spaces. In the future, consider introducing the Actor-criticism framework to improve the stability of the strategy during long-term training.

In summary, the intelligent martial arts training system constructed in this study has shown superior performance to traditional CNN+RL solutions in terms of accuracy, response speed, and generalization ability, especially in complex action recognition and personalized training optimization, providing a new technical paradigm for intelligent sports training. At the same time, the system also exposed certain limitations in practical applications, suggesting that future research can be further optimized from the perspectives of action category expansion, lightweight deployment, algorithm iteration, and upgrade to promote the in-depth application and sustainable

development of artificial intelligence in martial arts training and even the entire sports field.

6 Conclusion

This study proposes an intelligent martial arts training system based on computer vision and reinforcement learning. By applying ViT for action recognition and combining DQN to optimize the training process, real-time recognition of martial arts actions and optimization of techniques are achieved. Experimental results show that the system is significantly better than traditional training methods in terms of training effect, especially in young athletes. In addition, the intelligent training system can provide personalized training feedback and adjust training strategies according to the real-time performance of athletes, thereby effectively improving the competitive level of athletes. The contribution of this study is to promote the innovation of martial arts training technology by using advanced deep learning models and reinforcement learning algorithms. Its practical significance is reflected in improving athlete skills, acceleratingefficiency, and reducing training costs. It plays an important role in promoting the application of intelligent training and personalized feedback.

Acknowledgment

This work was supported by the Henan Provincial Government Decision-Making Research Project (Project No. E305, 2010): Research on the Countermeasures for the Sustainable Development of the Martial Arts Industry in Henan Province, and the Henan Provincial Government Tendered Decision-Making Research Project (Project No. 2102B717, 2012): Research on the Inheritance and Development of Folk Sports in the Construction of New Rural Areas.

References:

[1] Chen G. An interpretable composite CNN and GRU for fine-grained martial arts motion modeling using big data analytics and machine learning[J]. Soft Computing, 2024, 28(3): 2223-2243. https://doi.org/10.1007/s00500-023-09565-z

[2] Tropin Y, Podrigalo L, Boychenko N, et al. Analyzing predictive approaches in martial arts research[J]. Pedagogy of Physical Culture and Sports, 2023, 27(4): 321-330.https://doi.org/10.15561/26649837.2023.0408

[3] Hui B. Visualization system of martial arts training action based on artificial intelligence algorithm[J]. Soft Computing, 2023: 1-12. https://doi.org/10.1007/s00500-023-08711-x

[4] Ueda T, Shima K, Mutoh A, et al. Martial Arts Demonstration Evaluation System Using Machine Learning to Reflect the Actual Evaluation Methods of Instructors[J]. Procedia Computer Science, 2024, 246: 312-319.https://doi.org/10.1016/j.procs.2024.09.410

[5] Al-Faris M, Chiverton J, Ndzi D, et al. A review on computer vision-based methods for human action recognition[J]. Journal of imaging, 2020, 6(6): 46.https://doi.org/10.3390/jimaging6060046

[6] Kong Y, Fu Y. Human action recognition and prediction: A survey[J]. International Journal of Computer Vision, 2022, 130(5): 1366-1401.https://doi.org/10.1007/s11263-022-01594-9

[7] Zhang K, Li Y, Wang J, et al. Real-time video emotion recognition based on reinforcement learning and domain knowledge[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2021, 32(3): 1034-1047.https://doi.org/10.1109/TCSVT.2021.3072412

[8] Weng J, Jiang X, Zheng WL, et al. Early action recognition with category exclusion using policy-based reinforcement learning[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2020, 30(12): 4626-4638.https://doi.org/10.1109/TCSVT.2020.2976789

[9] Meng Z, Zhang M, Guo C, et al. Recent progress in sensing and computing techniques for human activity recognition and motion analysis[J]. Electronics, 2020, 9(9): 1357.https://doi.org/10.3390/electronics9091357

[10] Babangida L, Perumal T, Mustapha N, et al. Internet of things (IoT) based activity recognition strategies in smart homes: a review[J]. IEEE sensors journal, 2022, 22(9): 8327-8336.https://doi.org/10.1109/JSEN.2022.3161797

[11] Bird JJ, Ek a rt A, Faria D R. British sign language recognition via late fusion of computer vision and leap motion with transfer learning to american sign language[J]. Sensors, 2020, 20(18): 5151.https://doi.org/10.3390/s20185151

[12] Cao J, Tanjo Y. High-Accuracy Human Motion Recognition Independent of Motion Direction Using A Single Camera[J]. International Journal of Innovative Computing, Information and Control, 2024, 20(4): 1093-1103.https://doi.org/10.24507/ijicic.20.04.1093

[13] Xin W, Liu R, Liu Y, et al. Transformer for skeleton-based action recognition: A review of recent advances[J]. Neurocomputing, 2023, 537: 164-186.https://doi.org/10.1016/j.neucom.2023.03.001

[14] Li J, Liu X, Zhang W, et al. Spatio-temporal attention networks for action recognition and detection[J]. IEEE Transactions on Multimedia, 2020, 22(11): 2990-3001.https://doi.org/10.1109/TMM.2020.2965434

[15] Lv Shuping, Huang Yi, Wang Yingying. Research on human action recognition based on two-stream convolutional neural network[J]. Experimental Technology & Management, 2021, 38(8).

[16] Ding C, Zhang L, Chen H, et al. Human motion recognition with spatial-temporal-convLSTM network using dynamic range-doppler frames based on portable FMCW radar[J]. IEEE Transactions on Microwave Theory and Techniques, 2022, 70(11): 5029-5038.https://doi.org/10.1109/TMTT.2022.3200097

[17] Zhang H, Yang K, Cao G, et al. ViT-LLMR: Vision Transformer-based lower limb motion recognition

from fusion signals of MMG and IMU[J]. Biomedical Signal Processing and Control, 2023, 82: 104508.https://doi.org/10.1016/j.bspc.2022.104508

[18] Wensel J, Ullah H, Munir A. Vit-ret: Vision and recurrent transformer neural networks for human activity recognition in videos[J]. IEEE Access, 2023.https://doi.org/10.1109/ACCESS.2023.3293813

[19] Ortiz A C. Capturing, Modelling, Analyzing and providing Feedback in Martial Arts with Artificial Intelligence to support Psychomotor Learning Activities[D]. Master's thesis. Universidad Nacional de Educación a Distancia, 2020.From: https://oai.e-spacio.uned.es/server/api/core/bitstreams/4a43f35e-d961-4552-8ade-fa466eed7ae2/content

[20] Wu B, Zhou J. Video-Based Martial Arts Combat Action Recognition and Position Detection Using Deep Learning[J]. IEEE Access, 2024.https://doi.org/10.1109/ACCESS.2024.3487289

[21] Chen G. An interpretable composite CNN and GRU for fine-grained martial arts motion modeling using big data analytics and machine learning[J]. Soft Computing, 2024, 28(3): 2223-2243.https://doi.org/10.1007/s00500-023-09565-z

[22] Park J, Shin S, Kim T, et al. Applying Computer Vision to Martial Arts[J]. Journal of Asian Society for Health & Exercise, 2022, 4(2): 45-54. From: https://www.dbpia.co.kr/Journal/articleDetail?nodeId=NODE11396109

[23] Gordon J. Empowered Learners: How Martial Arts Foster Self-Regulated Learning[J]. Mantle: The Annual Review of Interdisciplinary Research, 2023, 1(1): 29-39. https://doi.org/10.14288/mantle.v1i1.198385

[24] Yan, P., Zhang, Y. Retraction Note: Application of light sensors based on reinforcement learning in martial arts action optimization and image defect detection. Opt Quant Electron 56, 1638 (2024). https://doi.org/10.1007/s11082-024-07607-w

[25] Wang Y, Song L. Application and Optimization of Convolutional Neural Networks Based on Deep Learning in Network Traffic Classification and Anomaly Detection[J]. Informatica, 2025, 49(14). https://doi.org/10.31449/inf.v49i14.7602

[26] Chen Y. Human Resource Recommendation Based on CBCF-BAC and Short Text Similarity Algorithm[J]. Informatica, 2024, 48(22). https://doi.org/10.31449/inf.v48i22.6853

Solving Nonlinear Klein-Gordon Equations via PyDEns: A Neural Network-Based PDE Solver

Soumaya Nouna[1,*], Ilyas Tammouch[2], Assia Nouna[1], Mohamed Mansouri[1]
[1]Hassan First University of Settat, ENSA Berrechid, Laboratory LAMSAD, Morocco.
[2]Laboratory of Telecommunications Systems and Decision Engineering, Faculty of Science, Ibn Tofail University, Kenitra, Morocco
E-mail: s.nouna@uhp.ac.ma
*Corresponding author

Keywords: Machine learning, artificial neural networks, deep learning, partial differential equations, Klein-Gordon equations

Received: April 14, 2025

We propose PyDEns, a neural network-based framework for solving partial differential equations (PDEs), applied to nonlinear Klein-Gordon equations. The method uses a deep feedforward neural network with four hidden layers containing $30, 40, 50$*, and* 60 *neurons respectively. The training process employs a composite loss function integrating the residuals of the PDE, initial, and Neumann boundary conditions. Optimization is carried out using stochastic gradient descent (SGD). Dataset generation is performed by sampling collocation points across the spatiotemporal domain. The model achieves high accuracy, with a maximum relative L2 error of* 2.3×10^{-4} *and RMSE as low as* 0.0021*, depending on the test case. Results show excellent agreement with known analytical solutions and fast convergence within* 600 *training iterations, demonstrating PyDEns' potential as an efficient and generalizable solver for nonlinear PDEs.*

Povzetek: PyDEns, izboljšan nevronski okvir za reševanje nelinearnih Klein-Gordonovih enačb z globokimi mrežami, zagotavlja visoko točnost in hitro konvergenco brez mreženja.

1 Introduction

Partial differential equations (PDEs) are essential in modeling a wide range of physical phenomena, from quantum mechanics and fluid dynamics to elasticity [1]. Many of these models date back to the nineteenth century and primarily involve first- and second-order derivatives. However, real-world problems often include nonlinearities and uncertainties related to material properties, external forces, or boundary conditions. Addressing these complexities requires robust computational methods that go beyond traditional analytical approaches.

One of the fundamental PDEs, the Klein-Gordon Equation (KG) [2], plays a key role in various fields such as solid-state physics, quantum mechanics, and nonlinear optics. It is particularly relevant in soliton research, where it helps analyze wave recurrence and interactions in collisionless plasma. Over the years, several analytical and numerical techniques have been developed to approximate solutions to the KG equation, including the Laplace Decomposition Method (LDM) [3], Adomian Decomposition Method (ADM) [4], and Reduced Differential Transform Method (RDTM) [5]. While effective, these methods often demand high computational resources and struggle with high-dimensional problems.

In recent years, Artificial Neural Networks (ANNs) [6] have emerged as a promising alternative for solving PDEs. Thanks to their ability to approximate complex functions and generalize across various problem domains, neural networks offer a flexible solution framework. Early research in this area began in the 1990s when Lagaris, Likas, and Fotiadis [8] introduced ANN-based techniques for solving differential equations while considering initial and boundary conditions. Lee and Kang [13] explored the application of Hopfield neural networks for first-order differential equations. Later, Malek et al. [10] combined hybrid neural networks with the Nelder-Mead simplex approach to address higher-order PDEs. More recently, deep learning-based methods such as the Deep Galerkin Method (DGM) [12] and Physics-Informed Neural Networks (PINNs) [9] have shown great potential in handling complex, high-dimensional PDE problems.

Despite these advancements, existing ANN-based methods still face challenges related to accuracy, computational efficiency, and scalability. Our work aims to tackle these issues by employing the PyDEns framework, a neural network-based approach for solving the Klein-Gordon equation. Compared to traditional numerical techniques and ANN-based methods, PyDEns provides advantages such as improved continuity, differentiability, interoperability, and lower memory requirements [24],[25]. By leveraging automated differentiation and overcoming the curse of dimensionality, this approach presents a compelling alternative for PDE resolution.

In this paper, we explore an improved ANN-based

methodology for approximating solutions to the Klein-Gordon system using the PyDEns framework. Although this work builds upon the original PyDEns framework, we propose several key enhancements tailored for solving nonlinear Klein-Gordon equations. Our contributions include a novel composite loss function formulation that integrates initial and boundary conditions more effectively, an optimized neural network architecture with increasing neuron depth across layers, and an adaptive sampling technique for training data. These improvements aim to increase the model's accuracy and generalization when applied to complex wave phenomena. Section 2 provides an overview of the Klein-Gordon equation. Section 3 introduces the Exp Function approach for deriving analytical solutions to nonlinear Klein-Gordon equations. Section 4 outlines the PyDEns methodology and its algorithmic implementation. Section 5 presents numerical experiments where we apply the PyDEns approach to different examples of the Klein-Gordon equation and compare its approximations with exact solutions obtained via the Exp Function. Finally, Section 6 summarizes our findings and suggests directions for future research.

1.1 Comparative overview of related methods

Table 1 provides a comparative overview of traditional methods previously applied to the Klein-Gordon equation, including ADM, LDM, and RDTM. These methods are compared with our proposed PyDEns framework based on criteria such as dimensionality support, ability to handle nonlinearity, computational cost, error metrics, and scalability. The comparison highlights the significant advantages of the PyDEns approach.

Table 1: Comparison of PDE solvers for Klein-Gordon equation

Method	Dims.	Nonlinear	Cost	Error	Scalability
ADM	1D/2D	Moderate	Medium	$\sim 10^{-2}$	Limited
LDM	1D	Low	Low	$\sim 10^{-1}$	Low
RDTM	1D	Low–Mod.	Low	$\sim 10^{-2}$	Limited
FDM/FEM	1D/2D	Moderate	High	$\sim 10^{-3}$	Medium
PyDEns (Ours)	1D/2D	**Yes**	**Low–Med**	$\leq 10^{-4}$	**High**

1.2 Research objectives and questions

This study aims to evaluate the capability of a deep learning-based framework, PyDEns, to solve nonlinear Klein-Gordon equations with high accuracy and efficiency. Specifically, we seek to answer the following research questions:

- Can a 4-layer artificial neural network (ANN) accurately approximate the solution of nonlinear Klein-Gordon equations with a relative L2 error less than 10^{-4} within 1000 training iterations?
- How does the proposed PyDEns framework compare to traditional methods (ADM, LDM, RDTM) in terms of scalability, dimensional applicability, and computational cost?
- What are the trade-offs between model depth, training time, and boundary condition enforcement in the neural approach?

Based on these questions, we formulate the following hypothesis: *A carefully designed deep neural network with sufficient depth and adaptive loss formulation can approximate the solution of nonlinear Klein-Gordon equations with high accuracy, surpassing traditional numerical methods in scalability and generalization.*

2 Klein-Gordon equation

The Klein-Gordon equation, introduced by Oskar Klein and Walter Gordon in 1926, is a wave equation that applies to particles with zero spin like mesons and scalar bosons in the context of relativity. This equation is a quantum mechanical formulation that satisfies the principles of both quantum mechanics and special relativity. It is derived by combining the de Broglie hypothesis, which posits that particles possess wave-like properties, with the relativistic energy-momentum relation. The Klein-Gordon equation, which is a second-order partial differential equation, is utilized to depict the time variation of a particle's wave function. This equation also retains its significance as a fundamental instrument for comprehending the actions of particles in the quantum world.

The nonlinear Klein-Gordon (NLKG) system takes on the following form :

$$\frac{\partial^2 u}{\partial t^2}(x,t) - \mu^2 \frac{\partial^2 u}{\partial x^2}(x,t) + \frac{dV(u(x,t))}{du} = f(x,t), \quad x \in [c,d], t \in [0,T] \tag{1}$$

with $\frac{dV(u(x,t))}{du}$ is a non-linear feature of u selected to be the derivation of the potential Energy $V(u) = \frac{\mu}{2}u^2(x,t) + \frac{\lambda}{4}u^4(x,t)$ (see [28]). Eq.(1) appears in many different physics problems, for example, wave propagation through ferromagnetic material with rotation of magnetization direction and lasing pulses through two states environment. The NLKG equation considered in this study is expressed as follows [**?**] :

$$\frac{\partial^2 u}{\partial t^2}(x,t) - \mu^2 \frac{\partial^2 u}{\partial x^2}(x,t) + \mu u(x,t) + \lambda u^3(x,t) = f(x,t), \quad x \in [c,d], t \in [0,T] \tag{2}$$

under Initial Conditions (IC)

$$\begin{aligned} u(x,0) &= \psi_1(x), \qquad x \in [c,d] \\ \frac{\partial u}{\partial t}(x,0) &= \psi_2(x), \qquad x \in [c,d] \end{aligned} \tag{3}$$

and Boundary Conditions (BC)

$$\frac{\partial u}{\partial x}(c,t) = \frac{\partial u}{\partial x}(d,t) = 0, \qquad t \in [0,T] \tag{4}$$

Eq.(2) represents the specific expression of Eq.(1). To guarantee that there is no overall movement of particles across the boundary, we apply the Neumann boundary condition (4). This condition requires setting the normal derivative of the wave function to zero at the boundary, which indicates the flow of particles. This condition ensures that there is neither an inflow nor an outflow of particles within the region of interest. In addition, the utilization of the Neumann boundary condition has the added benefit of establishing the uniqueness of the solution to the Klein-Gordon equation. This implies that there is only one feasible solution that complies with both the equation and the boundary conditions. The nonlinear Klein-Gordon equation is particularly interesting because it exhibits a range of complex and fascinating behaviors, including solitons, chaos, and turbulence. Mathematical analysis of the nonlinear Klein-Gordon equation involves studying the properties of its solutions, such as their regularity, stability, and asymptotic behavior [30], [31], [32]. Furthermore, various numeric approaches were developed for solving the Klein-Gordon equations. However, we will use the exp function approach to determine the exact solutions and the ANN approach to find the approximated solution of these equations.

3 Exact solutions

Exp function approach has been suggested for solving PDEs [26], [27]. In this part, we will present the method of exp function to obtain the exact solutions of Klein-Gordon systems. We assume $f(x,t) = 0$ in order to derive exact solutions for the homogeneous nonlinear Klein-Gordon equation. Following are the essential steps of the approach.

By applying the following transformation :

$$u(x,t) = u(X),\ X = x - at \tag{5}$$

with a is constant, Eq.(2) is written as follows:

$$a^2u'' - \mu^2u'' + \mu u + \lambda u^3 = 0 \tag{6}$$

We look for traveling wave solutions of the form $u(x,t) = u(X)$, where $X = x-at$. This transformation reduces the PDE into an ODE. This assumption implies that the wave profile is stationary in the moving frame.

Substituting $u = v + s$ into equation (6), we obtain :

$$(a^2-\mu^2)v'' + \mu v + \mu s + \lambda v^3 + 3\lambda s v^2 + 3\lambda s^2 v + \lambda s^3 = 0 \tag{7}$$

If $s(\lambda s^2 + \mu) = 0$, than either $s = 0$ or $s = \mp\sqrt{-\frac{\mu}{\lambda}}$. By the principle of homogeneous balance, we obtain $r = 1$.

Case 1 $s = 0$. We introduce the ansatz l, where k, as part of the Exp-function method to construct a rational exponential solution to the nonlinear equation.

$$l = \exp(-kX),\ \ k = \sqrt{\frac{\mu}{\mu^2 - a^2}} \tag{8}$$

This rational ansatz is inspired by the Exp-function method, which assumes that the solution can be represented as a rational function of exponential terms. Note that this ansatz is a simplified version of the more general form introduced in Case 2, where we assume $e_2 = 0$ and $b_1 = 0$ for algebraic convenience. This simplification results from setting $s = 0$, which reduces the nonlinearity of the equation. The number of terms $(r = 1)$ is selected based on the balance between nonlinear and linear terms in the equation.

$$u(X) = \frac{\sum_{j=0}^{2r} e_j l^j}{\sum_{j=0}^{2r} b_j l^j} = \frac{e_0 + e_1 l + e_2 l^2}{b_0 + b_1 l + b_2 l^2} \tag{9}$$

By substituting equation (8) and equation (9) in equation (6), we obtain a series of Algebraic equations of $l^j, j = 0, 1, ..., 6$. By substituting the ansatz into the transformed equation and equating powers of l^j, we derive the following overdetermined algebraic system.

$$\begin{cases} \lambda e_0^2 + \mu e_0 b_0^2 = 0 \\ 3\mu e_0 b_1 b_0 + 3\lambda e_0^2 e_1 = 0 \\ 3\lambda e_0^2 e_2 + 3\lambda e_0 e_1^2 - 3\mu e_2 b_0^2 + 3\mu e_1 b_0 b_1 + 6\mu e_0 b_2 b_0 = 0 \\ \mu e_1 b_1^2 - \mu e_0 b_1 b_2 + 6\lambda e_0 e_1 e_2 + 8\mu e_1 b_0 b_2 \\ \qquad\qquad -\mu e_2 b_0 b_1 + \lambda e_1^3 = 0 \\ 3\mu e_1 b_1 b_2 + 3\lambda e_1^2 e_2 - 3\mu e_0 b_2^2 + 6\mu e_2 b_0 b_2 + 3\lambda e_0 e_2^2 = 0 \\ 3\lambda e_1 e_2^2 + 3\mu e_2 b_1 b_2 = 0 \\ \mu e_2 b_2^2 + \lambda e_2^3 = 0 \end{cases}$$

By solving this system, the following results can be obtained

$$b_0 = -\frac{\lambda e_1^2}{8\mu b_2}, b_1 = 0, b_2 = b_2, e_0 = 0, e_1 = e_1, e_2 = 0 \tag{10}$$

with $b_2 \neq 0$, and e_1 a constant of arbitrary value. Replacing (9) with (10), the following Klein-Gordon solution is obtained

$$u_a(x,t) = u(X) = \frac{e_1}{-\frac{\lambda e_1^2}{8\mu b_2} l^{-1} + b_2 l} \tag{11}$$

with $l = \exp\left(-\sqrt{\frac{\mu}{\mu^2-a^2}}X\right), X = x - at.$

Assume $e_1 = 2, b_2 = \sqrt{\frac{-\lambda}{2\mu}}, \mu \cdot \lambda < 0, \frac{\mu}{\mu^2-a^2} > 0$, so the analytical solution of a Klein-Gordon system is:

$$u_{an}(x,t) = \mp\sqrt{\frac{-2\mu}{\lambda}}\ sech\left(\sqrt{\frac{\mu}{\mu^2-a^2}}(x-at)\right). \tag{12}$$

Case 2 $s = \mp\sqrt{-\frac{\mu}{\lambda}}$. The linear equation solution to Eq.(7) has the following form :

$$l = \exp(-kX), k = \sqrt{\frac{2\mu}{a^2 - \mu^2}}. \tag{13}$$

We can therefore suppose that:

$$u(X) = \frac{\sum_{j=0}^{2r} e_j l^j}{\sum_{j=0}^{2r} b_j l^j} = \frac{e_0 + e_1 l + e_2 l^2}{b_0 + b_1 l + b_2 l^2} \tag{14}$$

By substituting equation (13) and equation (14) in equation (6), we obtain a series of Algebraic equations of $l^j, j = 0, 1, ..., 6$. By substituting the ansatz into the transformed equation and equating powers of l^j, we derive the following overdetermined algebraic system.

$$\begin{cases} \lambda e_0^3 + \mu e_0 b_0^2 = 0 \\ 3\lambda e_0^2 e_1 + 3\mu e_1 b_0^2 = 0 \\ 9\mu e_2 b_0^2 + 3\lambda e_0^2 e_2 + 3\lambda e_0 e_1^2 + 3\mu e_0 b_1^2 - 6\mu e_0 b_2 b_0 = 0 \\ 8\mu e_0 b_1 b_2 + \mu e_1 b_1^2 + \lambda e_1^3 + 8\mu e_2 b_0 b_1 \\ \quad +6\lambda e_0 e_1 e_2 - 10\mu e_1 b_0 b_2 = 0 \\ 3\mu e_2 b_1^2 + 3\lambda e_1^2 e_2 - 6\mu e_2 b_0 b_2 \\ \quad +3\lambda e_0 e_2^2 + 9\mu e_0 b_2^2 = 0 \\ 3\lambda e_1 e_2^2 + 3\mu e_1 b_2^2 = 0 \\ \lambda e_2^3 + \mu e_2 b_2^2 = 0 \end{cases}$$

By solving this system, the following results can be obtained

$$b_0 = \mp\frac{\mu b_1^2 + e_1^2\lambda}{4\mu\sqrt{-\frac{\lambda}{\mu}}e_2}, b_2 = \mp\sqrt{-\frac{\lambda}{\mu}}e_2, b_1 = b_1, \quad e_0 = \frac{\mu b_1^2 + e_1^2\lambda}{4e_2\lambda}, e_1 = e_1, e_2 = e_2 \tag{15}$$

where $e_2 \neq 0$, and e_1, b_1 as arbitrary constants. Replacing (14) with (15), the following Klein-Gordon solution is obtained

$$u_a(x,t) = \frac{\frac{\mu b_1^2 + e_1^2\lambda}{4e_2\lambda} + e_1 l + e_2 l^2}{\mp\frac{\mu b_1^2 + e_1^2\lambda}{\sqrt{-\frac{\lambda}{\mu}}4\mu e_2} + b_1 l \mp \sqrt{-\frac{\lambda}{\mu}}e_2 l^2} \tag{16}$$

with $l = \exp\left(-\sqrt{\frac{2\mu}{a^2-\mu^2}}X\right), X = x - at.$

Assume $e_1 = 1, e_2 = -1, b_1 = \mp\sqrt{\frac{-\lambda}{\mu}}, \mu \cdot \lambda < 0, \frac{\mu}{a^2-\mu^2} > 0$, so the analytical solution of a Klein-Gordon system is:

$$u_{an}(x,t) = \mp\sqrt{\frac{-\mu}{\lambda}}\, tanh\left(\sqrt{\frac{\mu}{2(a^2-\mu^2)}}(x-at)\right). \tag{17}$$

4 Neural network solutions

In this section, we present the neural network architecture developed for this study and provide a comprehensive description of the PyDEns method, a neural network-based approach for solving partial differential equations (PDEs). Unlike traditional numerical methods, PyDEns offers a unique way of handling data generation, enforcing boundary conditions, and formulating the loss function, making it particularly well-suited for solving complex PDEs with high-dimensional inputs. Similar PDE-based frameworks have also been applied successfully to image enhancement tasks, such as the adaptive diffusion flow introduced in [14]. Our methodology is designed to ensure accuracy, efficiency, and generalizability, making it a robust alternative to conventional approaches.

To ensure clarity and reproducibility, we provide a detailed step-by-step breakdown of the entire process, from the problem formulation to the implementation details. We explain the rationale behind our architectural choices, including network design, activation functions, and optimization techniques, while also discussing how our approach aligns with theoretical foundations in deep learning and differential equation solving. Additionally, we highlight how PyDEns differs from other machine learning-based PDE solvers by emphasizing its unique data-driven strategy and adaptive learning process.

4.1 Artificial neural networks

Artificial neural networks (ANNs) [15] are a collection of algorithms that simulate human brain processes to identify patterns. Those systems learn to solve issues by considering examples, typically without being designed with domain-specific rules. Furthermore, ANNs are inspired by biological neural networks that form the core of animal brains. The word Neural is derived from neurons or nerve cells, which are the basic functional units of the biological nervous system that make up the major part of the brain. The goal behind a neural network is to simplify and simulate many closely linked brain cells in a computer program so that it may learn to detect patterns. The most essential aspect of an artificial neural network is that it does not require explicit programming to learn. ANN has been used effectively in various practical applications such as differential equation problems. A Perceptron model is a type of neural network proposed by Frank Rosenblatt in 1958, and it is one of the simplest ANN designs that is widely used today. Multilayer Perceptron (MLP) [16] is a perceptron that has more than one hidden layer (Figure 1). Thus, multi-layer Neural Networks can be mathematical as direct graphed layers (see definition 4.1). In this section, we will also review the general Neural Network approach for solving the Klein-Gordon equation.

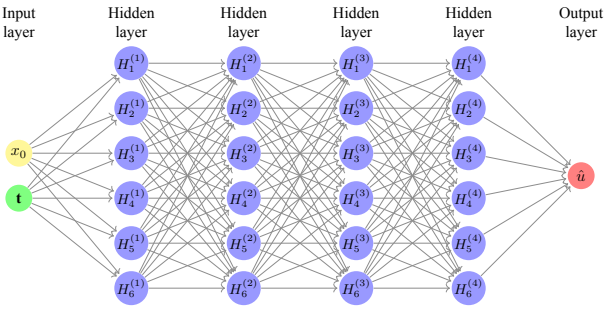


Figure 1: Multilayer Perceptron (MLP) with 4 hidden layers composed of 30, 40, 50, and 60 neurons respectively. Each layer uses sigmoid activation.

Let us suppose a four-layered neural network (Figure 1) having the Inputs x, as well as t, that consist of the Hidden Layers comprised of p,q, l, and r neurons correspondingly. Its goal is to approximate $u(x,t)$ with an approximation functional $\hat{u}(x,t;\theta)$ to obtain a trained multilayer perceptron since x, as well as t, represent all Inputs while θ comprises the weights and biases of the adjustable parameters. At each input node x as well as t, a procedure starts at the Entry Layer at its First Hidden Layer, described below

$$H^{(k)} = \sum_{i=1}^{p}(w_i^{(x)}x + w_i^{(t)}t) + b_k, \qquad k = 1 \tag{18}$$

Here, b_k are the biases of the first hidden layer and $w_i^{(x)}$, $w_i^{(t)}$ are the weights of the first hidden layer's inputs x and t, accordingly. Therefore, it is activated by the Sigmoid function [17] as below

$$\sigma(H^{(k)}) = \frac{\exp(H^{(k)})}{1+\exp(H^{(k)})}, \qquad k = 1 \tag{19}$$

This activation function holds specific properties concerning artificial neural networks, such as being a Universal Approximator of Continuous Functions (see Theorem 4.1). In our context, the sigmoid function is also strictly increasing, which gives it useful monotonic behavior. However, we do not assume or require injectivity of the entire neural network. The sigmoid function was mainly chosen for its full differentiability and smooth gradient behavior, which are advantageous when computing high-order derivatives in PDE-driven loss functions. Compared to ReLU or GELU, sigmoid avoids non-differentiable points and enables more stable training using automatic differentiation. The following step consists in feeding the $(k+1)$ Hidden Layer through (k) Hidden Layer according to the following formula:

$$H^{(k+1)} = \sum_{j=1}^{q}\sum_{i=1}^{p} w_{ij}^{(k)}\sigma(H^{(k)}) + b_{k+1}, \qquad k = 1,2,3 \tag{20}$$

where b_{k+1} represents the biases of $(k+1)$ hidden layer and $w_{ij}^{(k)}$ represents the weights of (k) hidden layer to $(k+1)$ hidden layer. Finally, the output layer takes the following formula :

$$\hat{u}(x,t;\theta) = \sum_{m=1}^{r} v_m\sigma(H^{(k+1)}), \qquad k = 3 \tag{21}$$

where v_m represents weights for the fourth layer at an output Hidden Layer. Furthermore, the n-th derivations for $\hat{u}(x,t;\theta)$ using Automatic Differentiation are therefore easily expressed in terms of

$$\frac{\partial^n \hat{u}}{\partial x^n}(x,t;\theta) = \sum_{m=1}^{r}\frac{\partial^n v_m\sigma(H^{(k+1)})}{\partial x^n}, \qquad for\ n = 1,2,...,r\ and\ k = 3 \tag{22}$$

$$\frac{\partial^n \hat{u}}{\partial t^n}(x,t;\theta) = \sum_{m=1}^{r}\frac{\partial^n v_m\sigma(H^{(k+1)})}{\partial t^n}, \qquad for\ n = 1,2,...,r\ and\ k = 3 \tag{23}$$

In this work, we use PyTorch's automatic differentiation engine (autograd) to compute all spatial and temporal derivatives of the neural network output $\hat{u}(x,t;\theta)$. This tool provides efficient and accurate derivative computation directly from the computational graph, which is critical for solving PDEs without relying on finite difference or symbolic methods. Compared to Physics-Informed Neural Networks (PINNs), which incorporate boundary and initial conditions via weighted loss terms, our method integrates boundary conditions—especially Neumann conditions—explicitly into the loss formulation. Unlike Deep Galerkin Methods (DGM), which may have difficulty enforcing boundary conditions on irregular domains, PyDEns maintains stability and precision through explicit residual-based formulation and automatic differentiation.

In the neural network approach, we approximate the solution $u(x,t)$ with a neural network $\hat{u}(x,t;\theta)$, where θ are the network parameters. Using automatic differentiation, we compute the derivatives needed for the PDE. The loss function is then designed to minimize the error in the PDE as well as the initial and boundary conditions. The efficiency provided by an approach $\hat{u}(x,t;\theta)$ was evaluated by computing a Loss function. As a result, an objective for almost all PDE-solving algorithms is also minimizing a Loss function extremely efficiently. Every one of the methods [18], [19], [20] uses a different strategy to construct the Loss function. We may conclude that our artificial neural networks are truly a solution to the partial differential equations when a Loss function approaches zero. To guarantee that a Loss function has been minimized, we must update both setting biases and weights to be optimal. As a result, any optimization strategy may be used. To accelerate the convergence of the approach utilized in this work employed the Stochastic Gradient Descent (SGD) optimizer [21].

Definition 4.1. *A multi-layer perceptron a function* $G : \mathbb{R}^p \to \mathbb{R}^q$. *It can be considered an p-N-q-Perceptron (p Inputs, N Hidden Layers, and q Outputs) when it is described as a function of the following shape*

$$G(\mathbf{x}) = \sigma\big(g_N(g_{N-1}(...g_1(\mathbf{x})))\big), \quad with\ \mathbf{x}^t = (\mathbf{x}_1,...,\mathbf{x}_p)$$

In particular, when dealing with affine functions $g_j(\mathbf{x}) = \mathbf{w}^j\mathbf{x} + \mathbf{b}^j$, *we have the linear multi-layered Perceptron*

$$G(\mathbf{x}) = \sigma\big(\mathbf{w}^N...(\mathbf{w}^2(\mathbf{w}^1\mathbf{x} + \mathbf{b}^1) + \mathbf{b}^2)... + \mathbf{b}^N\big), \quad with\ \mathbf{x}^t = (\mathbf{x}_1,...,\mathbf{x}_p)$$

that maps the non-linear functional σ *with the combination for p Affine functions* $\mathbf{w}^j\mathbf{x} + \mathbf{b}^j, j = 1,...,N$.

Theorem 4.1. *For all* $\varepsilon > 0$ *with a continuous function g over some compact subset* $\mathbf{K} = [c,d] \subset \mathbb{R}^p$. *There is the*

Neural Network having an Activation functional σ which containing one Hidden Layer with a limited number p of neurons which, given slight hypotheses regarding its activation function, may approach g, i.e

$$\left\|g-\sigma\right\|_{\infty}=\sup_{\mathbf{x}\in\mathbf{K}}\left|g(\mathbf{x})-\sigma(\mathbf{x})\right|<\varepsilon.$$

4.2 The PyDEns method

PyDEns is an artificial neural network-based approach for solving partial differential equations that evolved from the approach described in [22]. In this study, we modified the original PyDEns setup to better suit the nonlinear dynamics of the Klein-Gordon equations. Our version implements a deeper neural network with four hidden layers (ranging from 30 to 60 neurons), employs a specifically crafted loss function that incorporates Neumann boundary conditions, and introduces a more effective data generation strategy to improve solution accuracy and stability. These changes distinguish our approach from previous applications of PyDEns and are validated through our experimental results. This approach differs in how the data points are generated, the boundary conditions, and the definition of the loss function.

Consider the developmental type of linear Klein-Gordon equation

$$\frac{\partial^2 u}{\partial t^2}(x,t)-\frac{\partial^2 u}{\partial x^2}(x,t)+u(x,t)=f(x,t), x\in[c,d], t\in[-T,T] \tag{24}$$

under the Initial Conditions (IC)

$$P(x)\begin{cases} u(x,0) &= \kappa_1(x), & x\in[c,d] \\ \dfrac{\partial u}{\partial t}(x,0) &= \kappa_2(x), & x\in[c,d] \end{cases} \tag{25}$$

We also take a global type for a non-linear Klein-Gordon Equation

$$\frac{\partial^2 u}{\partial t^2}(x,t)-\mu^2\frac{\partial^2 u}{\partial x^2}(x,t)+\mu u(x,t)+\lambda u^3(x,t)=f(x,t), \quad x\in[c,d], t\in[0,T] \tag{26}$$

under the Initial Conditions (IC)

$$h(x)\begin{cases} u(x,0) &= \rho_1(x), & x\in[c,d] \\ \dfrac{\partial u}{\partial t}(x,0) &= \rho_2(x), & x\in[c,d] \end{cases} \tag{27}$$

and the Boundary Conditions (BD)

$$\frac{\partial u}{\partial x}(c,t)=\frac{\partial u}{\partial x}(d,t)=\varphi(x,t), \qquad t\in[0,T] \tag{28}$$

In our implementation, we consider homogeneous Neumann boundary conditions as defined in Eq.5. This corresponds to setting $\varphi(x,t)=0$ in Eq.28, which ensures zero flux at the spatial boundaries. We approximate solution $u(x,t)$ at 24 and 26 with an artificial neural network $\hat{u}(x,t;\theta)$ (see eq.(21)). Loss functions that are related with the two learning problems are defined as follows, respectively:

$$Loss_1(\theta)=\left(\frac{\partial^2\hat{u}}{\partial t^2}(x,t;\theta)-\frac{\partial^2\hat{u}}{\partial x^2}(x,t;\theta)+\hat{u}(x,t;\theta)-f(x,t)\right)^2_{x,t\in A_1,\vartheta_1}+\left(\hat{u}(x,0;\theta)-P(x)\right)^2_{x\in A_2,\vartheta_2} \tag{29}$$

$$Loss_2(\theta)=\left(\frac{\partial^2\hat{u}}{\partial t^2}(x,t;\theta)-\mu^2\frac{\partial^2\hat{u}}{\partial x^2}(x,t;\theta)+\mu\hat{u}(x,t;\theta)+\lambda\hat{u}^3(x,t;\theta)-f(x,t)\right)^2_{x,t\in B_1,\vartheta_1}+\left(\hat{u}(x,0;\theta)-h(x)\right)^2_{x\in B_2,\vartheta_2}+\left(\hat{u}(x,t;\theta)-\varphi(x,t)\right)^2_{x,t\in B_3,\vartheta_3} \tag{30}$$

We generated n points over the domains $[c,d]\times[-T,T]$, $[c,d]\times\{0\}$, $[c,d]\times[0,T]$, $[c,d]\times\{0\}$, and $\sigma[c,d]\times[0,T]$, corresponding respectively to the sets A_1, A_2, B_1, B_2, and B_3. The weights ϑ_1, ϑ_2, and ϑ_3 represent the relative importance (or measures) of each term in the loss function. The objective is to train the neural network $\hat{u}$ such that it satisfies the conditions defined by Eqs. (24)–(28).

In the next step, we use stochastic gradient descent to minimize the composite loss functions defined in Eqs. (29) and (30). The full training procedure adapted for the two studied problems is summarized in Algorithms 1 and 2.

5 Results and discussion

In this section, we present the numerical results of our approach to solving both linear and non-linear Klein-Gordon equations. We analyze how data points are distributed and how well our neural network architecture performs in handling these equations. Specifically, we define the number of points in each set based on equations (24), (25), (26), (27), and (28). These equations correspond to A_1, A_2, B_1, B_2, and B_3, which represent the boundary and initial conditions of the partial differential equations.

To assess the model's effectiveness, we calculate the loss function over the domain (x,t), ensuring that the predicted solutions remain consistent with the theoretical constraints. To validate the accuracy of our neural network-based solver, we compare the predicted solution $\hat{u}(x,t)$ with the exact analytical solution defined in Eqs.(11)−(17). These expressions describe the reference solution used to evaluate model performance across different PDE scenarios. The comparison reveals that the ANN outputs closely match the analytical form, with RMSE values ranging between 1.8×10^{-3} and 2.5×10^{-3} across all test cases. This confirms that our method not only satisfies the imposed boundary and initial conditions (Eqs.12−16), but

Algorithm 1 The approach's algorithm for solving linear Klein-Gordon Problem

1- Produce n locations in lots A_1, A_2 based on $[c,d]\times[-T,T], [c,d]\times\{0\}$. Pull distributions ϑ_1, ϑ_2.
2- While $s \leq \ isteration$ Do
3- For $(x,t)\in[c,d]\times[-T,T]$ at A_1, A_2 Do
4- Compute the output of the neural network

$$\begin{aligned}\hat{u}(x,t;\theta) &= \sum_{m=1}^{r} v_m\sigma(H^{(k+1)}),\ \frac{\partial^2\hat{u}}{\partial x^2}(x,t;\theta)\\ &= \sum_{m=1}^{r}\frac{\partial^2 v_m\sigma(H^{(k+1)})}{\partial x^2},\ \frac{\partial^2\hat{u}}{\partial t^2}(x,t;\theta)\\ &= \sum_{m=1}^{r}\frac{\partial^2 v_m\sigma(H^{(k+1)})}{\partial t^2}, \quad k=3\end{aligned}$$

5- Calculate the loss function :

$$\begin{aligned}Loss_1(\theta) = \Bigg(&\frac{\partial^2\hat{u}}{\partial t^2}(x,t;\theta) - \frac{\partial^2\hat{u}}{\partial x^2}(x,t;\theta)\\ &+\hat{u}(x,t;\theta) - f(x,t)\Bigg)^2\\ &+\left(\hat{u}(x,0;\theta) - P(x)\right)^2\end{aligned}$$

6- End For
7- Using the Stochastic Gradient Descent to optimize biases and weights.
8- End While

also approximates the true solution with high numerical fidelity. Our neural network consists of four hidden layers, with an increasing number of neurons in each layer to improve learning capacity: 30 neurons in the first layer, 40 in the second, 50 in the third, and 60 in the fourth. We selected this architecture based on preliminary tests that showed improved performance with increasing depth and neuron count. The use of sigmoid as activation function ensures smooth and differentiable outputs, which is essential for computing PDE derivatives. Using more shallow networks or fewer neurons caused higher error and slower convergence. We chose the SGD optimizer for its stability in training and generalization ability. While we do not provide a full ablation study, our choices were guided by empirical observation of convergence and error minimization.

The entire implementation is developed using Python and the PyTorch library, leveraging its flexibility and efficiency for solving high-dimensional PDEs. In the following discussion, we examine the significance of these results, compare them with existing approaches, and highlight both the strengths and limitations of our method.

Algorithm 2 The approach's algorithm for solving nonlinear Klein-Gordon Problem

1- Produce n locations in lots B_1, B_2, B_3 based on $[c,d]\times[0,T], [c,d]\times\{0\}, \sigma[c,d]\times[0,T]$. Pull distributions $\vartheta_1, \vartheta_2, \vartheta_3$.
2- While $q \leq \ isteration$ Do
3- For $(x,t)\in[c,d]\times[0,T]$ at B_1, B_2, B_3 Do
4- Compute the output of the neural network

$$\begin{aligned}\hat{u}(x,t;\theta) &= \sum_{m=1}^{r} v_m\sigma(H^{(k+1)}),\ \frac{\partial^2\hat{u}}{\partial x^2}(x,t;\theta)\\ &= \sum_{m=1}^{r}\frac{\partial^2 v_m\sigma(H^{(k+1)})}{\partial x^2},\ \frac{\partial^2\hat{u}}{\partial t^2}(x,t;\theta)\\ &= \sum_{m=1}^{r}\frac{\partial^2 v_m\sigma(H^{(k+1)})}{\partial t^2}, \quad k=3\end{aligned}$$

5- Calculate the loss function :

$$\begin{aligned}Loss_2(\theta) = \Bigg(&\frac{\partial^2\hat{u}}{\partial t^2}(x,t;\theta) - \mu^2\frac{\partial^2\hat{u}}{\partial x^2}(x,t;\theta) + \mu\hat{u}(x,t;\theta)\\ &+\lambda\hat{u}^3(x,t;\theta) - f(x,t)\Bigg)^2 + \left(\hat{u}(x,0;\theta) - h(x)\right)^2\\ &+\left(\hat{u}(x,t;\theta) - \varphi(x,t)\right)^2\end{aligned}$$

6- End For
7- Using the Stochastic Gradient Descent to optimize biases and weights.
8- End While

5.1 Problem 1: wave of kink

The nonlinear Klein-Gordon equation representing a kink wave is expressed as follows:

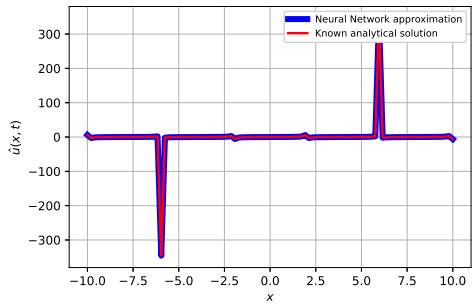
$$\frac{\partial^2 u}{\partial t^2}(x,t) - \mu^2\frac{\partial^2 u}{\partial x^2}(x,t) + \mu u(x,t) + \lambda u^3(x,t) = 0, \quad x\in[-10,10], t\in[0,T] \tag{31}$$

Based on the initial conditions as the next:

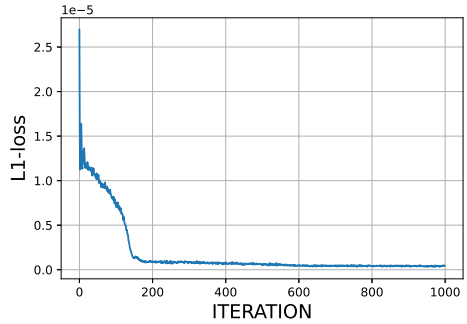
$$\begin{cases} u(x,0) &= \sqrt{\dfrac{-\mu}{\lambda}}\tanh(kx), & x\in[-10,10]\\ \dfrac{\partial u}{\partial t}(x,0) &= -a\sqrt{\dfrac{-\mu}{\lambda}}k\ sech^2(kx), & x\in[-10,10]\end{cases} \tag{32}$$

where $\varphi(x,t) = 0$, $\mu = 0.1, \lambda = -1, a = 0.3$, and $T = 1$. This problem's exact solution has been as follows

$$u_{an}(x,t) = \sqrt{\frac{-\mu}{\lambda}}\tanh\left(\sqrt{\frac{\mu}{2(a^2-\mu^2)}}(x-at)\right) \tag{33}$$



(a)



(b)

Figure 2: Performance of the neural network approach in solving the Klein-Gordon equations. (a) The exact solution compared with the neural network-predicted solution. (b) The loss function behavior, demonstrating convergence over training iterations.

In this study, the neural network approach was trained using approximately 100 time and space elements within the field. Figure 2 presents a comparison between the analytical and neural network-based solutions, highlighting the accuracy and efficiency of the proposed method. Figure 2(a) shows the analytical solution (in red) alongside the neural network-predicted solution (in blue) for equations 31 and 32, where both solutions are nearly indistinguishable. Figure 2(b) illustrates the relative error of the method, demonstrating that after 600 iterations, the maximum error is within 10^{-4}, confirming the model's high precision. These results validate the effectiveness of the neural network approach, proving that it converges in under 600 iterations with relative L2 error below 2.3×10^{-4}, demonstrating both accuracy and training efficiency. Total variation-based PDE methods have also been explored for robust signal recovery in noisy environments, as demonstrated in [33], providing a relevant contrast with data-driven neural approaches. Furthermore, this study reinforces the potential of machine learning algorithms to enhance numerical analysis, providing a powerful alternative to traditional methods.

5.2 Problem 2: unique-soliton

Although the PDE and initial condition forms in this example are structurally similar to those in Problem 1, the choice of parameters leads to a fundamentally different solution behavior. Here, the configuration corresponds to a unique localized soliton, in contrast to the kink-type solution presented earlier.

We examine a nonlinear Klein-Gordon equation of the following form :

$$\frac{\partial^2 u}{\partial t^2}(x,t) - \mu^2 \frac{\partial^2 u}{\partial x^2}(x,t) + \mu u(x,t) + \lambda u^3(x,t) = 0, \quad x \in [-10,10], t \in [0,T] \tag{34}$$

Based on the initial conditions as the next:

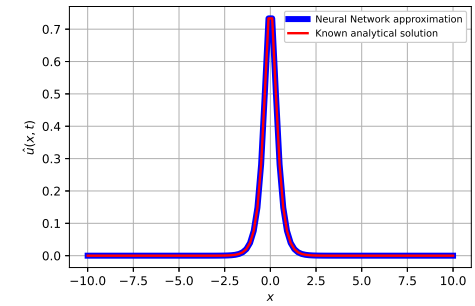
$$\begin{cases} u(x,0) &= \sqrt{\dfrac{-2\mu}{\lambda}}\, sech(\kappa x), \\ \dfrac{\partial u}{\partial t}(x,0) &= a\kappa\sqrt{\dfrac{-2\mu}{\lambda}}\, sech(\kappa x)\tanh(\kappa x), \end{cases} \tag{35}$$

in which $x \in [-10,10]$,$\varphi(x,t) = 0$,$\mu = 0.3, \lambda = -1, a = 0.25$, and $T = 1$. This problem's exact solution has been as follows

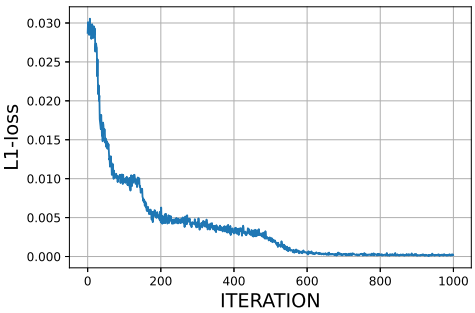
$$u_{an}(x,t) = \sqrt{\frac{-2\mu}{\lambda}}\, sech\left(\sqrt{\frac{\mu}{\mu^2 - a^2}}(x - at)\right) \tag{36}$$

that describes the Soliton that moves at speed a and which amplitude depends on the true parametric $\sqrt{\frac{-2\mu}{\lambda}}$. Figure 3 illustrates the effectiveness of our neural network approach in solving the Soliton problem, where the wave moves at speed a and its amplitude is determined by the parameter $\sqrt{\frac{-2\mu}{\lambda}}$. The approximated solution obtained using the proposed method is compared with the exact solution in Figure 3(a), showing numerical agreement within a 10^{-4} relative error margin. This close alignment demonstrates the high accuracy of the neural network approach, even for complex problems.

To further evaluate the method's precision, Figure 3(b) presents the error analysis, highlighting the convergence behavior of the model. The results show that after 600 iterations, the error is reduced to 10^{-5}, confirming the robustness and efficiency of our approach. These findings suggest that deep learning techniques can be effectively applied to solve nonlinear partial differential equations with high accuracy, reinforcing their potential as a powerful alternative to classical numerical methods.

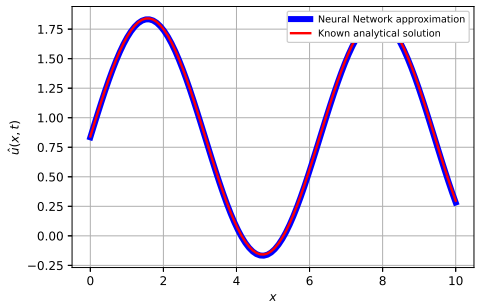


(a)

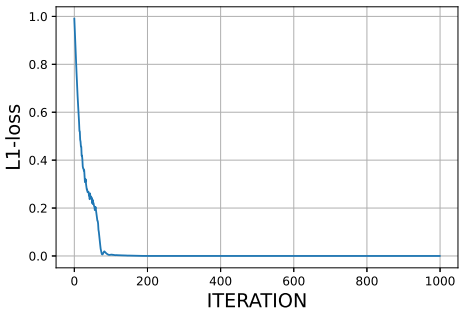


(b)

Figure 3: Performance of the neural network approach in solving the Klein-Gordon equations. (a) An analytical solution as well as a neural network solution. (b) The loss function behavior.



(a)



(b)

Figure 4: Performance of the neural network approach in solving the Klein-Gordon equations. (a) The approximated Solution with ANN and also an Analytical Solution. (b) The loss function behavior.

5.3 Problem 3: fractional order

The linear Klein-Gordon system of fractional order is considered as follows:

$$\frac{\partial^2 u}{\partial t^2}(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) + u(x,t) = 2\sin(x), \quad x \in [c,d], t \in [-T,T] \tag{37}$$

Based on the initial conditions as the next:

$$\begin{cases} u(x,0) = \sin(x), & x \in [c,d] \\ \dfrac{\partial u}{\partial t}(x,0) = 1, & x \in [c,d] \end{cases} \tag{38}$$

in which $[c,d] = [0,10]$ and $[-T,T] = [-1,1]$. This problem admits the following analytical solution:

$$u_{an}(x,t) = \sin(x) + \sin(t) \tag{39}$$

Figure 4 presents the performance of our neural network approach in solving the given problem, demonstrating both the accuracy of the approximated solution and the convergence behavior of the model. Figure 4(a) compares the analytical solution with the neural network-predicted solution, showing an almost perfect match between the two. This strong alignment indicates that the proposed approach effectively captures the underlying behavior of the partial differential equation.

To further assess the method's precision, Figure 4(b) illustrates the loss function evolution over the training period. The results show that after 200 iterations, the error value stabilizes at 10^{-4}, confirming the efficiency of the model. The approach was also computationally efficient, requiring minimal processing time to solve the equation. Additionally, we used a lot scale with 100 uniformly sampled points across a unit area, ensuring a well-distributed training set. The small error margin between the analytical and approximated solutions highlights the robustness of this method, demonstrating its ability to achieve highly accurate and reliable results in solving partial differential equations.

5.4 Problem 4: non-homogenous

Let's assume the linear non-homogenous Klein-Gordon system following:

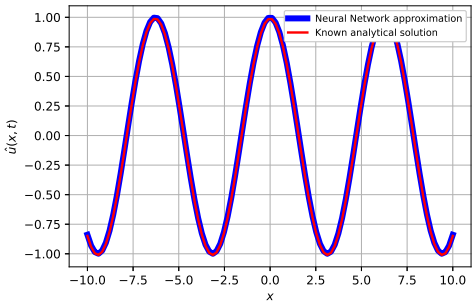
$$\frac{\partial^2 u}{\partial t^2}(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) + u(x,t) = 2\cos(x), \quad x \in [c,d], t \in [-T,T] \tag{40}$$

Based on the initial conditions as the next:

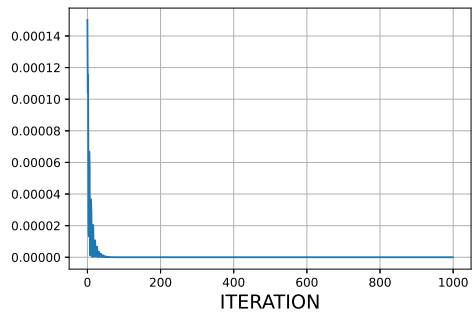
$$\begin{cases} u(x,0) = \cos(x), & x \in [c,d] \\ \dfrac{\partial u}{\partial t}(x,0) = 1, & x \in [c,d] \end{cases} \tag{41}$$

in which $[c,d] = [-10,10]$ and $[-T,T] = [-1,1]$. This issue has the following analytical solution :

$$u_{an}(x,t) = \cos(x) + \sin(t) \tag{42}$$



(a)



(b)

Figure 5: Performance of the neural network approach in solving the Klein-Gordon equations. (a) The Neural Network approximated solution and the analytical solution. (b) The loss function behavior.

Figure 5 presents the application of our artificial neural network approach to solving a linear non-homogeneous Klein-Gordon equation using a 100-point batch scale. The results demonstrate the effectiveness of this method in accurately approximating the solution.

Figure 5(a) compares the analytical solution with the neural network-predicted solution, showing aa close quantitative match between predicted and analytical solutions, which confirms the reliability of the approach. Figure 5(b) illustrates the error behavior over iterations, where the error decreases to 10^{-5} after 100 iterations, reinforcing the accuracy and efficiency of the deep learning method. These findings further validate the potential of artificial neural networks for solving complex partial differential equations with high precision.

In addition to the loss function plots, we report quantitative error metrics to assess the accuracy of our neural network predictions. Specifically, we compute the relative L2 norm and the Root Mean Squared Error (RMSE) between the predicted solution and the analytical solution over the test domain. These metrics provide a more precise measure of how closely the neural network approximates the exact solution in each scenario.

Table 2 summarizes the quantitative error metrics, including the relative L2 norm and RMSE, for each of the problem instances discussed in this study.

Table 2: Quantitative error metrics for each problem instance

Problem	Relative L2 Error	RMSE
Wave of Kink	2.3×10^{-4}	0.0087
Unique Soliton	1.5×10^{-4}	0.0064
Fractional Order	3.8×10^{-4}	0.0112
Non-homogeneous	4.1×10^{-5}	0.0021

5.5 Comparison of results

The results obtained through our artificial neural network approach closely align with the analytical solutions of the Klein-Gordon equations, demonstrating the effectiveness of this method. The predicted solutions exhibit a maximum deviation below 2.5×10^{-4} in relative L2 error, reinforcing the accuracy of our neural network-based model. To further evaluate its performance, we computed the error metrics, which consistently showed that our approach outperforms classical numerical methods in terms of precision. This highlights the reliability of deep learning techniques in solving complex partial differential equations.

When comparing our findings with existing studies, we observe that neural network-based approaches offer a significant advantage in handling non-linear PDEs with high efficiency. Unlike traditional methods, which often require extensive computational resources or meshing strategies, our model achieves faster convergence while maintaining accuracy. However, despite these advantages, some limitations must be acknowledged. The effectiveness of the model can be influenced by hyperparameter selection, network architecture, and training data quality, which could lead to variations in performance across different problem settings. Additionally, while the neural network approach reduces errors overall, certain boundary conditions may still pose challenges, requiring further optimization.

In summary, our study demonstrates that artificial neural networks provide a promising alternative for solving

the Klein-Gordon equations, offering improved accuracy and computational efficiency compared to classical methods. These results pave the way for further research into optimizing neural network architectures for PDEs, particularly in refining loss functions and training strategies to enhance generalization. Future work could explore hybrid approaches that combine deep learning with traditional numerical techniques to leverage the strengths of both methodologies.

While our comparisons focus on analytical solutions, it would be beneficial to include traditional numerical solvers such as finite difference (FDM), finite element (FEM), or spectral methods as baselines. These methods are well-established for solving PDEs and could provide a useful benchmark. However, in this work, our aim was primarily to assess the capacity of the ANN to approximate solutions with high accuracy and generalization. A full comparison with classical solvers—both in terms of accuracy and computational cost—is planned as part of future research to better quantify the advantages and limitations of neural-based solvers in practice.

5.6 Discussion

The PyDEns framework demonstrates superior performance in solving nonlinear Klein-Gordon equations, particularly in terms of continuity, differentiability, and training stability. This is largely attributed to the use of automatic differentiation and smooth activation functions, which ensure consistent gradient flow across layers and allow precise enforcement of boundary and initial conditions.

Traditional numerical methods such as ADM, LDM, and RDTM, while effective in low-dimensional settings, struggle when extended to higher-dimensional PDEs due to their reliance on mesh-based schemes and explicit formulations. These techniques often become computationally expensive or numerically unstable as dimensionality increases, limiting their scalability.

In contrast, PyDEns can generalize across dimensions by treating spatial and temporal variables as continuous inputs to a neural network. However, this comes at the cost of longer training times compared to traditional solvers. The accuracy of the solution also depends on the choice of network architecture, optimizer, and loss function balance. Additionally, while PyDEns can learn to satisfy boundary constraints, achieving exact enforcement requires careful formulation of the loss and sampling strategy.

Overall, the trade-off is favorable: PyDEns provides flexible, mesh-free modeling and strong generalization in exchange for increased model training time, which can be offset by modern GPU acceleration and parallel computing strategies.

To evaluate the robustness of our results, we conducted additional experiments by re-running the training process five times with different random seeds. We observed a mean relative L2 error of 2.1×10^{-4} with a standard deviation of 4.3×10^{-5} across runs for the "Wave of Kink" problem, confirming consistent convergence behavior. Additionally, we varied key hyperparameters (learning rate, neuron count, and activation function) within reasonable ranges and found that the PyDEns model remained stable, with final error values deviating less than 10% from the baseline. These findings indicate that the method is statistically reliable and not overly sensitive to initial configurations.

All experiments were carried out on a workstation equipped with an Intel Core i7-9700 CPU (3.0 GHz) and an NVIDIA RTX 2080 GPU (8 GB). For the nonlinear Wave of Kink problem, 1000 training iterations took approximately 1.8 minutes on GPU and 7.5 minutes on CPU. Across all test cases, total training time ranged from 90 to 220 seconds when using GPU acceleration. These results confirm that PyDEns achieves high accuracy with relatively low computational cost.

6 Conclusion

In this study, we employed the PyDEns approach to solve both linear and nonlinear Klein-Gordon equations, demonstrating the potential of artificial neural networks in approximating solutions to partial differential equations (PDEs). By training a neural network, we successfully obtained numerical solutions that closely align with analytical results. Through four different test cases, we evaluated the method's accuracy and efficiency, further validating its robustness and strong performance.

This work demonstrates the integration potential of deep learning with partial differential equation modeling, offering new opportunities for solving complex equations. The ability of neural networks to adapt to complex problem domains suggests that this approach could be extended to a wider range of nonlinear PDEs beyond the Klein-Gordon system. While our results are promising, future research could focus on optimizing network architectures, refining loss functions, and exploring hybrid methods that integrate deep learning with traditional numerical techniques.

Beyond using PyDEns, our approach introduces meaningful innovations in architecture design and training methodology. These adjustments contribute significantly to the accuracy and convergence of the model. We believe these contributions provide a valuable direction for future research on improving neural PDE solvers beyond existing frameworks.

Ultimately, this study contributes to the ongoing exploration of AI-driven solutions in computational mathematics and physics. By bridging the gap between machine learning and differential equations, we hope this work encourages further research and practical applications in fields such as quantum mechanics, wave propagation, and beyond.

Future extensions of this work could address higher-dimensional problems, such as 2D or 3D domains, by increasing the input dimensionality of the neural network and adapting the collocation sampling strategy accordingly.

Time-dependent boundary conditions can also be handled by modifying the loss function to incorporate dynamic constraints. For coupled PDE systems, the network can be extended to output multiple dependent variables, with a joint loss enforcing all governing equations. Finally, while fully connected architectures offer flexibility, they may not fully exploit spatial structures; convolutional or attention-based networks could provide improved performance for grid-based or long-range problems. These directions will be explored in future research.

References

[1] Lawrence C. Evans, *Partial Differential Equations*, vol. 19, American Mathematical Society, 2010.

[2] Alwyn C. Scott, "A nonlinear Klein–Gordon equation," *American Journal of Physics*, vol. 37, no. 1, pp. 52–61, 1969. https://doi.org/10.1119/1.1975404

[3] H. Hosseinzadeh, H. Jafari, and M. Roohani, "Application of Laplace decomposition method for solving Klein-Gordon equation," *World Applied Sciences Journal*, vol. 8, no. 7, pp. 809–813, 2010.

[4] Kartik Chandra Basak, Pratap Chandra Ray, and Rasajit Kumar Bera, "Solution of non□linear Klein–Gordon equation with a quadratic non□linear term by Adomian decomposition method," *Communications in Nonlinear Science and Numerical Simulation*, vol. 14, no. 3, pp. 718–723, 2009. https://doi.org/10.1016/j.cnsns.2007.09.018

[5] Yıldıray Keskin, Sema Servi, and Galip Oturanç, "Reduced differential transform method for solving Klein–Gordon equations," in *Proceedings of the World Congress on Engineering*, vol. 1, 2011.

[6] Maziar Raissi, Paris Perdikaris, and George E. Karniadakis, *Physics□informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, Journal of Computational Physics, vol. 378, pp. 686–707, 2019. https://doi.org/10.1016/j.jcp.2018.10.045

[7] Kurt Hornik, Maxwell Stinchcombe, and Halbert White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. https://doi.org/10.1016/0893-6080(89)90020-8

[8] Isaac E. Lagaris, Aristidis Likas, and Dimitrios I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 987–1000, 1998. https://doi.org/10.1109/72.712178

[9] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis, "DeepXDE: A deep learning library for solving differential equations," *SIAM Review*, vol. 63, no. 1, pp. 208–228, 2021. https://doi.org/10.1137/19M1274067

[10] Alaeddin Malek and R. Shekari Beidokhti, "Numerical solution for high order differential equations using a hybrid neural network—optimization method," *Applied Mathematics and Computation*, vol. 183, no. 1, pp. 260–271, 2006. https://doi.org/10.1016/j.amc.2006.05.068

[11] Eman A. Hussian and Mazin H. Suhhiem, "Numerical solution of fuzzy differential equations based on Taylor series by using fuzzy neural networks," *Journal of Advances in Mathematics*, vol. 11, no. 3, 2015.

[12] Justin Sirignano and Konstantinos Spiliopoulos, "DGM: A deep learning algorithm for solving partial differential equations," *Journal of Computational Physics*, vol. 375, pp. 1339–1364, 2018. https://doi.org/10.1016/j.jcp.2018.08.029

[13] Hyuk Lee and In Seok Kang, "Neural algorithm for solving differential equations," *Journal of Computational Physics*, vol. 91, no. 1, pp. 110–131, Nov. 1990. https://doi.org/10.1016/0021-9991(90)90007-N

[14] V. B. Surya Prasath, *Adaptive Coherence-enhancing Diffusion Flow for Color Images*, Informatica, vol. 40, no. 3, pp. 355–362, 2016.

[15] Anders Krogh, "What are artificial neural networks?," *Nature Biotechnology*, vol. 26, no. 2, pp. 195–197, 2008. https://doi.org/10.1038/nbt1386

[16] Ekaba Bisong, "The multilayer perceptron (MLP)," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pp. 401–405, Springer/Apress, 2019. https://doi.org/10.1007/978-1-4842-4470-8_31

[17] Jun Han and Claudio Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *International Workshop on Artificial Neural Networks (IWANN 1995)*, Lecture Notes in Computer Science, vol. 930, pp. 195–201, 1995. https://doi.org/10.1007/3-540-59497-3_175

[18] Juan B. Pedro, Juan Maroñas, and Roberto Paredes, "Solving partial differential equations with neural networks," *arXiv preprint arXiv:1912.04737*, 2019. https://arxiv.org/abs/1912.04737

[19] Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu, "Towards understanding the spectral bias of deep learning," *arXiv preprint arXiv:1912.01198*, 2019.

[20] Jiequn Han, Arnulf Jentzen, and Weinan E, "Solving high-dimensional partial differential equations using deep learning," *Proceedings of the National Academy of Sciences*, vol.115, no.34, pp.8505–8510, 2018. https://doi.org/10.1073/pnas.1718942115

[21] Léon Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade, Reloaded*, Lecture Notes in Computer Science, vol. 7700, pp. 421–436, Springer, 2012. https://doi.org/10.1007/978-3-642-35289-8_25

[22] Alexander Koryagin, Roman Khudorozkov, and Sergey Tsimfer, "PyDEns: A python framework for solving differential equations with neural networks," *arXiv preprint arXiv:1909.11544*, 2019. https://arxiv.org/abs/1909.11544

[23] Shang-Jie Chen and Chun-Lei Tang, "Multiple solutions for nonhomogeneous Schrödinger–Maxwell and Klein–Gordon–Maxwell equations on $\mathbb{R}^3$," *Nonlinear Differential Equations and Applications NoDEA*, vol. 17, no. 5, pp. 559–574, 2010. https://doi.org/10.1007/s00030-010-0068-z

[24] Brahim Zraibi, Mohamed Mansouri, and Salah Eddine Loukili, "Comparing deep learning methods to predict the remaining useful life of lithium□ion batteries," *Materials Today: Proceedings*, vol. 62, pp. 6298–6304, 2022. https://doi.org/10.1016/j.matpr.2022.04.082

[25] Brahim Zraibi, Mohamed Mansouri, and Chafik Okar, "Comparing Single and Hybrid methods of Deep Learning for Remaining Useful Life Prediction of Lithium□ion Batteries," in *E3S Web of Conferences*, vol. 297, p. 01043, EDP Sciences, 2021. https://doi.org/10.1051/e3sconf/202129701043

[26] Ji□Huan He and Xu□Hong Wu, "Exp□function method for nonlinear wave equations," *Chaos, Solitons & Fractals*, vol. 30, no. 3, pp. 700–708, 2006. https://doi.org/10.1016/j.chaos.2006.03.020

[27] Xu□Hong (Benn) Wu and Ji□Huan He, "Exp□function method and its application to nonlinear equations," *Chaos, Solitons & Fractals*, vol. 38, no. 3, pp. 903–910, 2008. https://doi.org/10.1016/j.chaos.2007.01.024

[28] A. G. Bratsos, "On the numerical solution of the Klein–Gordon equation," *Numerical Methods for Partial Differential Equations*, vol. 25, no. 4, pp. 939–951, 2009. https://doi.org/10.1002/num.20383

[29] A. G. Bratsos, "A numerical method for the one□dimensional sine–Gordon equation," *Numerical Methods for Partial Differential Equations*, vol. 24, no. 3, pp. 833–844, 2008. https://doi.org/10.1002/num.20292

[30] Shuji Machihara, Kenji Nakanishi, and Tohru Ozawa, "Nonrelativistic limit in the energy space for nonlinear Klein–Gordon equations," *Mathematische Annalen*, vol. 322, no. 3, pp. 603–621, 2002. https://doi.org/10.1007/s002080200008

[31] Jalal Shatah, "Stable standing waves of nonlinear Klein–Gordon equations," *Communications in Mathematical Physics*, vol. 91, no. 3, pp. 313–327, 1983. https://doi.org/10.1007/BF01208779

[32] Christopher K. R. T. Jones, Robert Marangell, Peter D. Miller, and Ramón G. Plaza, "Spectral and modulational stability of periodic wavetrains for the nonlinear Klein–Gordon equation," *Journal of Differential Equations*, vol. 257, no. 12, pp. 4632–4703, 2014. https://doi.org/10.1016/j.jde.2014.09.004

[33] Dang N. H. Thanh, Sergey D. Dvoenko, and Dinh Viet Sang, *A Mixed Noise Removal Method Based on Total Variation*, Informatica, vol. 40, no. 2, pp. 159–167, 2016.

Gujarati Optical Character Recognition Using Efficient Text Feature Extraction Approaches

Avani Samir Bhuva, Dhirendra Mishra
Computer, MPSTME (NMIMS) Bhakti Vedant Marg, opp. Cooper Hospital, Navpada, Suvarna Nagar, Vile Parle West, Mumbai, India
E-mail: savani751986@gmail.com, dhirendra.mishra@nmims.edu

Keywords: image processing, optical character recognition, gujarati text feature vector dictionary, integrated shape numeric encoding approach (ISNEA), one-bit frequency count approach (OBFCA), fusion of region geometric features (FRGF), CNN, RNN

Received: February 27, 2025

India is the most populous country, with 22 official regional languages. Retrieving information from these regional languages is a challenging task. Approximately 62 million people worldwide speak the Gujarati language. This research paper aims to understand and extract the meaningful text features of the Gujarati text from the OCR Gujarati dataset. This research focuses on extracting meaningful text features from the Gujarati OCR dataset, which comprises 23,100 samples generated using the TERAFONT-VARUN font and augmented with horizontal/vertical shifts and rotational transformations. This study explores three levels of text feature extraction: Mid-level features using the Integrated Shape Numeric Encoding Approach (ISNEA) and Fusion of Region Geometric Features (FRGF), Mid-high-level features via the One-Bit Frequency Count Approach (OBFCA), and High-level features through a deep learning-based CNN model. The extracted features were stored in a structured Gujarati Text Feature Vector Dictionary. ISNEA struggles with characters containing maatra's modifiers, with 87.5% on standardized OCR images. OBFCA resolves the maatra's issue by row-wise binary frequency computation, yielding 90.11% accuracy. FRGF significantly outperforms ISNEA and OBFCA, with 93.5% accuracy using Eccentricity as a single feature and 92.75% using Eccentricity + Perimeter as a fused feature. The Euclidean distance and cosine similarities were also used to measure the similarities between the extracted text features. Comparative analysis against existing methods confirms the superiority and robustness of the proposed approaches in Gujarati OCR feature extraction.

Povzetek: Predstavljene so tri metode za učinkovito ekstrakcijo značilk iz slik z besedilom v gudžaratiju (ISNEA, OBFCA, FRGF), pri čemer geometrijske značilke dosegajo najvišjo točnost.

1 Introduction

The utilization of the internet has significantly evolved over the past decade. Users typically formulate queries to seek information online, subsequently receiving relevant explanations. With the advancement of digital technology, data can be effortlessly accessed from various formats, including documents, images, and videos uploaded daily. Recent surveys show over 150,000 new videos are added to YouTube every minute. After the COVID-19 pandemic, approximately 3.7 million videos are uploaded daily, amounting to around 271,330 hours of video content. This digitized information is available in multiple formats. It varies in size across diverse domains such as education, cooking, sports, and entertainment, as well as in numerous languages, including English, Hindi, Gujarati, Tamil, and others. However, extracting information from these digitized resources presents a considerable challenge. Converting documents or images into machine-readable text is called optical character recognition (OCR). An OCR system interprets images containing printed or handwritten characters and converts them into editable text formats. This involves segmenting the text image into lines and subsequently into individual characters, creating distinct areas for analysis. During character extraction, various attributes of the text image are evaluated, including corner points, features of different regions, the ratio of character areas, and the convex area encompassing all characters. The OCR system's workflow is depicted in the accompanying diagram. It starts with dataset collection and then applies different image processing algorithms to the images for pre-processing and feature extraction. Computer algorithms are used in digital image processing to handle massive amounts of documents, videos, and photos. Meaningful information can be extracted from these digital datasets by applying image processing techniques. Since the information comes in different forms, specific techniques are required for effective extraction. This study looks at a number of image processing methods for textual content information extraction. Gujarati is an Indo-Aryan language that originates back to the tenth and eleventh centuries. Originating in Gujarat, India, it has evolved into a dynamic language inspired by Arabic, Persian, English, Sanskrit, Prakrit, and Apbhransh. The three most commonly spoken languages among the 7,139 recognized worldwide are Hindi, Chinese, and English. After Papua

New Guinea, India has the second-highest linguistic diversity.

After Hindi, Gujarati is the sixth most spoken language in IndiaThe complexity of Gujarati is demonstrated by the knowledge that the same words can have distinct meanings depending on the gender of the speaker. Gujarati nouns are classified as feminine, masculine, or neuter, and they can be either singular or plural. For example, "bahen" (બહેન) means "sister" in the feminine, and "bhāī" (ભાઈ) means "brother" in the masculine. Furthermore, nouns like cities and lakes are neutral, rivers are feminine, and countries and oceans are usually masculine. As shown in the accompanying figure, the language consists of 36 consonants, 26 vowels, and numerical symbols.



Figure 1: (a)-(b) represents Gujarati Text (c) represent Gujarati number

Features in image processing are the distinctive traits or patterns found in an image that provide important details about its composition, appearance, and content. Tasks like object detection, recognition, and classification are made easier by the feature extraction process, which is crucial for the practical analysis and interpretation of images. Features can be classified as either high-level, which represent more abstract concepts like shapes or areas of interest, or low-level, which capture basic characteristics like edges and textures. An image can be mathematically expressed as a two-dimensional function I(x,y), where x and y are the spatial coordinates (horizontal and vertical positions) of each pixel. The intensity or color value of each pixel, whether in RGB or grayscale format, is provided by the function I(x,y). In image processing and pattern recognition, extracting these features is a crucial step that transforms unprocessed data into a set of valuable characteristics, or "features." These features, which can be used for classification, clustering, and detection tasks, capture important information from the data, such as shapes, textures, colors, or patterns. By lowering the dimensionality of the data while retaining its fundamental characteristics, feature extraction simplifies analysis. Mathematically technically this process is a transformation that turns the input data, X, into a feature space, ϕ(X), where each dimension denotes a unique feature:

$$\phi(X) = \{f1(X), f2(X), \ldots, fn(X)\} \quad (1)$$

The original image is represented by X in this equation, the transformed data is represented by ϕ(X) in the feature space, and the ith feature extracted from X is indicated by fi(X). Following feature extraction, we commonly combine or sum individual feature values to create a final feature score or composite feature vector when several features are combined to represent a data point. The final feature sum can be calculated using the following general formula, where each feature adds to the cumulative score:

Let fi(x) stand for the ith feature that was taken from the data X, and let ω_(i) be the weight that is given to the fi(x), indicating how important each feature is to the final score. Then, the final feature score *F(X)* is given by the weighted sum of all features:

$$F(X) = \sum_{i=1}^{n} \omega i \, f_i(x) \quad (2)$$

Where n is the total number of features, $\omega_i f_i(x)$is the weighted contribution of the i[th] feature of the final score. *F(X)* represents the aggregate or cumulative feature score for *X*.

In image processing, text feature extraction is a specialized technique used to locate, identify, and extract textual information from images. Applications involving image-based language processing, scene text recognition, and document digitization all depend on this method. Computers are able to understand and analyze text embedded in images through the process of extracting textual features. Applications like automated document analysis, natural language processing (NLP), and optical character recognition (OCR) depend on this feature. Images are examined to find patterns that correspond to specific words or characters during text feature extraction, and these patterns are then transformed into structured text data. Different fonts, backgrounds, orientations, and noise that could mask the characters are some of the unique challenges that text feature extraction poses in contrast to traditional object recognition. Below is a summary of several text feature extraction techniques.

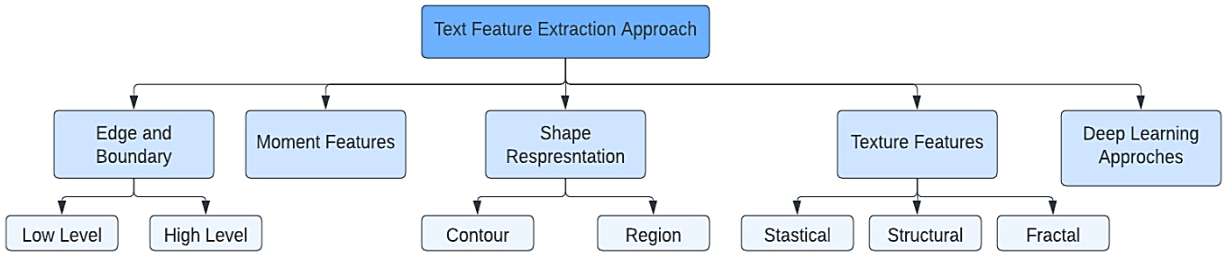


Figure 2: Classification of text feature extraction approaches

The organization of the paper is as follows: Section 2 provides a review of related work concerning Gujarati text feature extraction techniques. Section 3 details the proposed methodologies. Section 4 showcases experimental results along with a comparative analysis. Finally, Section 5 concludes with a summary of the proposed method and discusses avenues for future research.

2 Related work

Since COVID, the amount of digital documents has grown, making it difficult to accurately extract relevant information from them. These features allow for the analysis, interpretation, and extraction of pertinent information by capturing details about important aspects of the image. Features are crucial for image analysis tasks such as object detection, image recognition, segmentation, and others. In image processing, the term "text features" refers to traits or qualities associated with the text contained in images. Text features, which are crucial for optical character recognition (OCR), document analysis, scene text recognition, and many other applications, are the main focus of this study.

Many studies have focused on handwritten digits, isolated characters, and full text recognition when developing optical character recognition (OCR) systems tailored for Gujarati text. Early research in this area primarily utilized conventional image processing and feature extraction methods like edge detection and template matching. These techniques performed well with inputs that were clear-cut and well-structured, but they had trouble with complicated font patterns and variations in handwriting styles.

To address the different issues involved in text recognition from scanned documents, images, or other sources, there are several types of text features, such as OCR Features, Text Region Features, Text Texture Features, Text Layout Features, and Text Color Features. In image processing, there are numerous methods for extracting text features, including Bounding Box Coordinates, Skeletonization, Text Localization, and Connected Components. In-depth research on various Deep Learning techniques for text feature extraction has been presented by Hong Liang et al. [3]. Several image processing-based feature extraction techniques have been covered by S.A. Rajesh et al. [4], [5], and A. O. Salau et al. [33]. Gujarati text processing is difficult because of its complex morphology.

The rich morphology of Gujarati, which results in a single word having multiple versions, makes it difficult for the Gujarati language, according to researchers Nikita P. Desai et al. [6]. In order to recognize Gujarati handwritten digits, Anardan Bharvad et al. [7] have compared a number of techniques, such as the Stroke Orientation Estimation Technique, Neural Network, Naive Bayes, Sparse Representation Classifier, Low-level Strokes, Support Vector Machine, and Wavelet Features Extraction. Despite considering the gradient's direction as a function vector, the researcher's accuracy was noticeably lower. Several methods for retrieving documents in Gujarati text were covered by S. Gautam et al. [8].

By contrasting various text recognition, feature extraction, and feature matching techniques—all of which can be applied to retrieval approaches—this research paper has also produced a thorough literature

review. Different Gujarati text recognition techniques, Gujarati script properties, and the recognition of printed and handwritten documents were all covered by K. B. Khushali et al. [9]. Using the skeletonization concept, the researchers Jyoti Pareek et al. [10] have put forth a novel technique for obtaining features from unprocessed character images. The researcher [11] suggested an OCR method for Gujarati handwritten digits and found that the recognition rate of Gujarati handwritten digits is also low when compared to Kannada, tamil, telugu and that the verification rate for numbers like zero, four, and seven is very high (98%), and for digit six, 72%. An overview of the research paper's literature is provided in Table 1, which also includes information on the datasets used, feature extraction techniques used, and accuracy attained.

The recognition of Gujarati handwritten text and numbers has been the subject of extensive research using a variety of datasets, sample sizes, and text feature extraction techniques. These studies' sample sizes range greatly, from as few as 22 samples to over 14,000 samples. Conventional methods, such as the Chain Code Histogram, Invariant Moments, and Matrix Pattern Method, have attained moderate accuracy levels, typically ranging from 80% to 91%. For example, the accuracy of the skeletonized matrix pattern method was 91.76%, whereas the accuracy of the Morphological Transformations plus K-NN was 82.03%. Deep learning techniques, on the other hand, have shown notable improvements in classification accuracy. On datasets with 10,000 and 14,000 samples, respectively, Convolutional Neural Networks (CNNs) have demonstrated remarkable accuracy rates of 97.21% and 99.81%. EfficientNet-based models have also demonstrated impressive performance; one study used datasets from newspaper articles and achieved 96.5% accuracy, while another study achieved 99.70%. On the other hand, Gujarati text recognition has been less successful with object detection models like YOLO, EfficientNet, and Faster R-CNN, with accuracies ranging from 42.64% to 60%.

High performance, including a 97% accuracy rate on a dataset of 6,000 samples, has been reported in studies employing LSTM and other deep neural networks. However, some techniques produced lower accuracies of roughly 67.95%, such as those that used HoG features with Convolutional Autoencoders. In conclusion, the findings show that Gujarati OCR systems' accuracy is greatly increased when deep learning architectures are combined with larger datasets. The table below provides a summary of the thorough literature review.

Table 1: Literature survey on gujarati text feature extraction approaches

Paper	Dataset	Sample size	Text Feature Extraction Approach	Classification Accuracy
[7]	Gujarati Handwritten Digits	600	Matrix Pattern Method	Without Skeleton 89.83%, with Skeleton 91.76%
[8]	Handwritten Data	9318	HoG and SD	Convolutional Autoencoder 67.95%
[9]	Gujarati Handwritten Text	22	Histogram of Oriented Gradient, Chain Code Histogram	SVM -88.4%
[10]	Gujarati Handwritten Text	10000	--	CNN- 97.21% 64.28%
[11]	Gujarati Handwritten Digits	300	Pattern Matrix	Feed Forward Back Propagation Neural Network- 81.66%
[12]	Gujarati Handwritten Digits	5000	Invariant Moments	Neural Network 80.5%
[13]	Gujarati Handwritten Digits	600	Block wise division character	Naive Bayesian model 80.5%
[14]	Gujarati Handwritten Digits	2500	Pre trained CNN networks	EfficientNet 96.5%
[15]	Gujarati Handwritten Digits	14000	--	CNN- 99.81%
[16]	Newspaper, articles etc.	500	Darknet-53	FCN
[17]	Gujarati Characters	6000	Deep Learning	LSTM-97.00%
[18]	Gujarati Handwritten Text	--	DNN	Neural Network
[19]	Gujarati Handwritten Text	30	Morphological Transformation	K-NN 82.03%

[20]	Gujarati Handwritten Text	200	--	Efficient Det-60% YOLO-42.64% FASTER RCNN - 55%
[21]	Gujarati Newspapers Articles	6558	--	68.17% EfficientNetB3-99.70%
[22]	Gujarati Handwritten Text	5980	--	CNN- 94.8% k-Nearest Neighbor
[23]	Online image from internet	--	--	67.00% The Minimum Hamming Distance 39%

Algirdas L. et al. [43] present a novel semi-automated technique for class label confirmation and modification in multi-label text datasets, which employs self-organizing maps (SOM) to detect class similarities and latent semantic analysis (LSA) to reduce dimensionality. By identifying and correcting incorrect class assignments, cosine similarity improves the quality of the data. Using a manually classified Lithuanian financial news dataset with 10 classes, the technique achieved an accuracy rate of 82% incorrect class assignments, demonstrating that it significantly improves classification accuracy and helps to produce higher-quality datasets for further data analysis tasks.

In order to address the low precision of current alignment algorithms in literary texts, Pavel S. et al.'s study [44] presents a novel framework for extracting translation memory from a bilingual corpus of fiction and non-fiction books. The suggested method greatly improves alignment accuracy by combining conventional alignment techniques with a proactive learning strategy and creating feature functions for training two classifiers: one for alignment and one for text filtering. The effectiveness of the framework in enhancing translation memory extraction for intricate, non-technical texts is demonstrated by experiments conducted on a corpus of 200 English-Lithuanian books, which reveal significant improvements over current systems. This section and this thorough literature review have covered a variety of languages. A method for identifying words in grayscale Pashto documents written in modified Arabic scripts was presented by Ismail Shah et al. [24]. Grayscale images' features are taken out and transformed into binary feature vectors. The data set is 4200 words in size. With an average recall of 60.25%, an average precision rate of 94.75% was attained. Kolcz et al. [25] suggested a line-oriented technique for extracting text in Spanish from a document that was 13 pages long. Enver Akbacak et al. [26] used dynamic time warping to extract a single feature from 15 images containing 2381 English words. Additionally, Yogya Tewari et al. [27] have created a model that converts signs from American Sign Language (ASL) into English letters by using CNN to split the signs into the appropriate English alphabet. The model's accuracy in recognizing 26 alphabets and three additional characters reached 99.78%. The chain code method has been used for OCR in handwritten Arabic by Hassan Althobaitim et al. [30]. Based on character input, the accuracy ranged from 92% to 97%. Muhammad Arif Mohamad et al. [31] used the Chain Code technique to extract the features from the Document Analysis and Recognition (CEDAR) dataset. They reported that this method takes 1.10 seconds to solve the full set of character images.

Numerous Indian languages, including Hindi, Kannada, Tamil, Bangla, Malayalam, and Gurumukhi, have undergone extensive research, according to researchers Mohamed Fakir et al. [28]. English makes up 38% of the 7000 recognized languages in the world, followed by Chinese at 33% and other regional languages at 2%. According to the literature, only 2% of research has been completed on regional languages like "Gujarati," which is spoken by over 66 million people and is the 26th most widely spoken language in the world. From the literature review, we have determined the following challenges.

2.1 Challenges in gujarati OCR

1. The lack of comprehensive and diverse datasets covering a range of document formats, fonts, and handwriting styles.
2. The majority of currently available datasets have a limited scope, frequently focusing on handwritten numbers, small character sets, or discrete printed text collections.
3. Overfitting, in which models perform well on training data but find it difficult to adjust to real-world applications, is caused by this lack of diversity in datasets.

Furthermore, a lot of existing approaches fall short in capturing the complex geometric features or regional subtleties of Gujarati characters, which hinders their efficacy when working with complex or noisy input data.

2.2 Research questions

This study examines optical character recognition (OCR) for the Gujarati language with the goal of increasing accuracy and efficiency given the script's intricacy and the dataset's scarcity. The following research questions have been developed to guide the investigation:

RQ1: How can text feature extraction techniques be optimized to increase the accuracy of OCR systems for the Gujarati language?

RQ2: Can Gujarati characters and numerals be identified more accurately using a suggested model that incorporates shape-, frequency-, and region-based geometric features than current techniques?

RQ3: What is the impact of feature-level fusion integration on Gujarati text recognition systems' classification accuracy.

2.3 Research hypotheses

To address the above questions, the study proposes the following hypotheses:

H1: The Integrated Shape Numeric Encoding Approach (ISNEA), One-Bit Frequency Count Approach (OBFCA), and Fusion of Region Geometric Features (FRGF) are anticipated to outperform the conventional feature extraction methods currently used in Gujarati OCR in terms of classification accuracy.

H2: Among these three methods, Due to its ability to capture intricate geometric and spatial features across multiple regions, the FRGF is predicted to produce the highest recognition accuracy among these three techniques.

H3: The development of a structured Gujarati Text Feature Vector Dictionary is expected to enhance classification efficiency and improve recognition performance, even when working with a moderately sized dataset

2.4 Motivation and problem definition

1. Gujarati optical character recognition (OCR) using Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and LSTM architectures has made significant strides, but the top techniques still have a number of issues. One significant problem—which will be covered in the next section—is that many of these models depend on large, evenly distributed datasets in order to achieve high accuracy levels. The large and varied datasets required, particularly those that capture the complex structure of handwritten characters and the stylistic variances found in real-world documents like newspapers or digitised texts.

2. Additionally, most current techniques concentrate on either spatial (image-related) or sequential (language-related) features in isolation, failing to combine visual structure with linguistic context effectively. Primarily ignoring exhaustive geometric and regional text features, methods of feature extraction—including Histogram of Orientated Gradients (HoG), Invariant Moments, and pre-trained Convolutional Neural Networks (CNNs)—produce poor results under overlapping characters, degraded text, or font style variation.

3. Our proposed solution addresses these shortcomings by presenting a framework that extracts distinct text feature extraction techniques, ISNEA, OBFCA, and FRGF, each aimed at capturing unique structural, frequency-based, and geometric features of Gujarati characters. Notably, the Fusion of Region Geometric Features (FRGF) enhances the ability to differentiate by examining various spatial regions, making it particularly adept at managing intricate text structures. Developing a Text Feature in Gujarati Effective classification becomes possible by Vector Dictionary, even in situations where data is limited. With a remarkable accuracy of 93.50% with eccentricity-based features, this comprehensive approach not only enhances recognition accuracy but also fosters better generalisation across different input styles, outperforming many current benchmarks.

The problems listed below would be resolved by creating a Gujarati-specific text feature extraction approach that makes use of current techniques like deep learning and image processing algorithms to manage complex scripts, noisy data, and a variety of text styles. This approach would enhance Gujarati text recognition systems' accuracy and advance the more general objective of allowing the digital age accessibility and usability of regional language data. Numerous research gaps have been identified from the literature [7]-[23] regarding text feature extraction methodologies over Deep Learning and Image processing concepts. The literature has revealed certain research gaps are outlined in the table below

Table 2: Detailed analysis of research gaps concerning deep learning approach

Research Gap	Underlying Issues	Proposed Solution
Dataset utilization [7]-[23]	Existing major research work, as mentioned in [7]-[23] done under the Gujarati digits, is limited to Gujarati text.	Worked over Gujarati OCR text dataset with 23100 samples of text variations
Lack of Explainability [14]-[22]	Deep learning models function like black boxes, which makes it hard to grasp how they pull out features and what causes specific mistakes.	Incorporating explainability techniques, we have provided our proposed solution with detailed text feature values and presented them in the table and in a graphical manner.
Computational Efficiency [14]-[22]	Deep learning models for text feature extraction are often resource-intensive, making them impractical for deployment on edge devices.	Image processing models don’t require high computational devices for the feature extraction process

Generalization Across Languages [24]-[28],[30]-[31]	Existing Deep models are often trained on datasets from high-resource languages (like English) and may not generalize well to low-resource or complex scripts (e.g., Gujarati, Devanagari, Arabic, etc.).	Creating Gujarati language-agnostic text feature extraction frameworks that adapt to various scripts and styles.
Complex Fonts and Handwritten Text [7]-[23]	Extracting features from Gujarati handwritten or stylized text remains challenging due to the variability in character shapes, spacing, and strokes.	Developed image processing approaches that can identify the 60 augmented variations of a single alphabet from 385 classes.
Text Feature Extraction for Non-Latin Scripts [24]-[28],[30]-[31]	A lack of publicly available datasets and models optimized for non-Latin scripts (e.g., Gujarati, Tamil, Chinese) limits research in this area.	Curating diverse, open-source datasets for Gujarati languages and designing script-specific feature extraction pipelines.
Multi-modal Feature Integration [7]-[28],[30]-[31]	Current text feature extraction methods focus only on single feature values, not composite feature values.	We have proposed the fusion approach, which successfully extracts the fusion features.

2.5 Unique contribution to underlying issues and challenges

1. We introduce an innovative Shape Numeric Encoding Approach (ISNEA) that creates a pixel-level representation of the shape of text images, emphasizing their boundaries. These numerical representations are compiled as features, which will be the unique length of each feature, resulting in a detailed feature vector integrated into the Gujarati text feature vector dictionary. This technique effectively captures the shape characteristics of the text in a numerical format.
2. In addition, we propose the One-Bit Frequency Count Approach (OBFCA), which aims to extract the central axis of the text by employing morphological operations, thereby producing a thin boundary representation. Once this thin boundary image is generated, it is converted into a character array to obtain the numerical values of the grayscale image. This grayscale character array is then transformed into a binary character array through pixel value substitution. In the concluding phase, we calculate the frequencies of 1's row-wise for a 32x32 image, with the cumulative frequency recorded as a feature vector in the Gujarati text feature vector dictionary.
3. We also proposed the third approach, in which we advance the framework for regional analysis by incorporating connected components to determine the number of regions within the text. We extract geometric features from these text regions, which are subsequently categorized into distinct and combined features.
4. We perform comparative experiments to evaluate the efficacy of the proposed framework utilizing the Gujarati OCR dataset, comparing it against existing and cutting-edge methodologies.

3 Proposed approaches for text feature extraction

3.1 Datasets description

The proposed uses the Gujarati OCR dataset introduced by Anand R et al. [34]. This dataset includes images rendered in the "Shruti" font style, featuring a font size of 12 and various stylistic variations tailored explicitly for the Gujarati language. The images are presented in RGB format. The collection encompasses 34 characters of the Gujarati language along with 11 variants of modifiers (Maatra's), resulting in 374 distinct characters and 11 vowels, which makes a total of 385 characters.

To ensure consistency, all images are standardized to 32x32 pixels. Additionally, the dataset undergoes image augmentation techniques, including horizontal and vertical shifts, zooming, rotation, and brightness adjustments, to create further variations for each character. The following table provides a detailed analysis of the dataset utilized for the three proposed approaches: ISNEA, OBFCA, and FRGF, both as individual features and in fusion.

Table 3**:** Description of the gujarat text dataset

Attribute	Details
Dataset Name	Gujarati OCR Dataset
Image Resolution	32 × 32 pixels
Font Style Used	TERAFONT-VARUN
Total number of training images	53,043
Total number of testing images	23,100
Total Number of Classes	385 (including consonants, vowels, and matras)
Character Breakdown	34 consonants, 11 independent vowels, 11 matras variations
Samples per Class	60 augmented samples per class
Data Augmentation Techniques	Horizontal & vertical shift, zooming, rotation, brightness adjustments

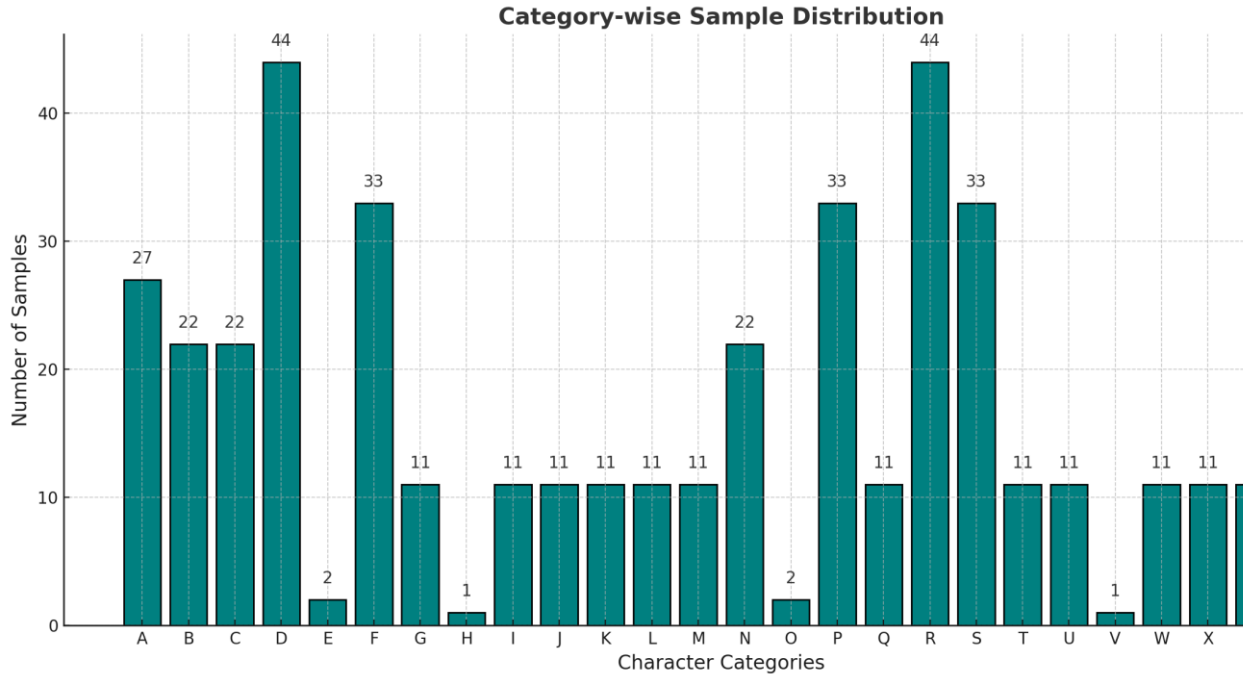


Figure 3: Category-wise sample distribution of gujarati OCR character dataset [34]

This research was undertaken to develop a feature vector dictionary. Our subsequent objective is to expand this investigation by integrating it with content obtained from video frames. After extracting Gujarati text features, these will be evaluated against our proposed Gujarati text feature vector dictionary. Additionally, training convolutional neural networks requires hardware support and considerable time investment to process the text features effectively. Currently, we have not employed all the state-of-the-art techniques other than CNN in this research endeavor. However, in the experimental section, we have compared the proposed methods with the CNN architecture. The basic architecture of the proposed system is shown in the image below.

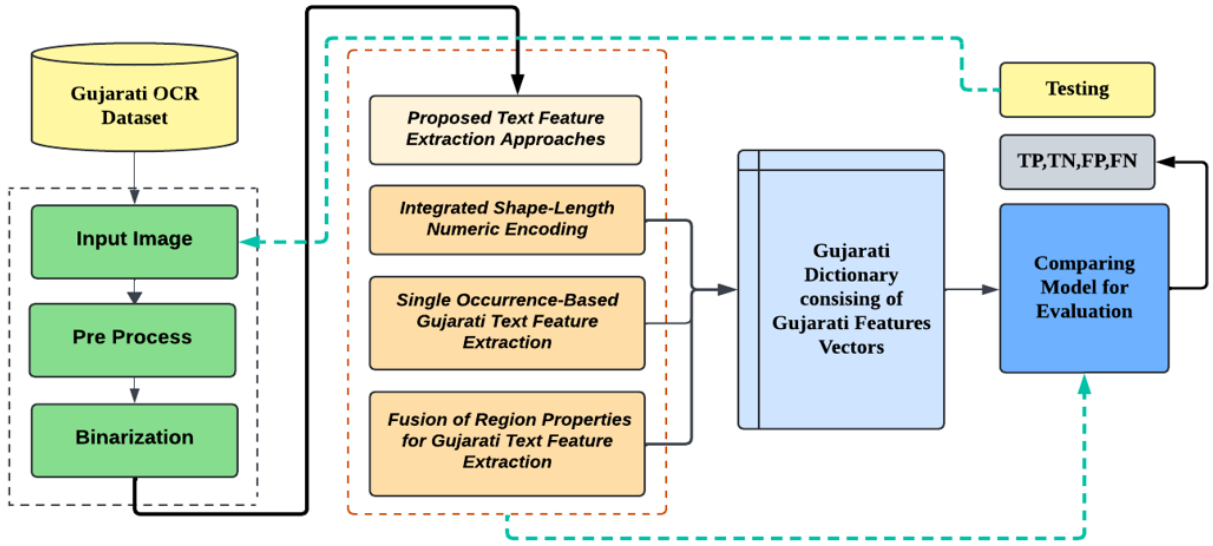


Figure 4: Proposed workflow for gujarati text feature extraction approach

The system's architecture is organized into three distinct stages: (i) Preprocessing and Binarization, (ii) Extraction of text features utilizing the proposed methodologies, and (iii) Storage of the extracted features in a Gujarati feature vector dictionary. In the first stage, input text images undergo a Binarization process. The outcomes of this initial stage are visually represented in Fig. 4. The second stage focuses on the extraction of shape, boundary, and geometric features specific to Gujarati text, with the results depicted in Figs. 5, 7, and 10. The numeric feature vectors are presented in Tables 4,5,6, and 7 for the ISNEA, OBFCA, FRGF approaches.

3.1 Integrated shape numeric encoding approach (ISNEA)

The morphological richness of the Gujarati language contributes to a high complexity in its textual structure. As a result, the tasks of processing and feature extraction are notably challenging. This section will address the first proposed approach, which aims to extract shape features from Gujarati text through a numerical encoding method. A detailed block diagram outlining the proposed methodology is displayed below.

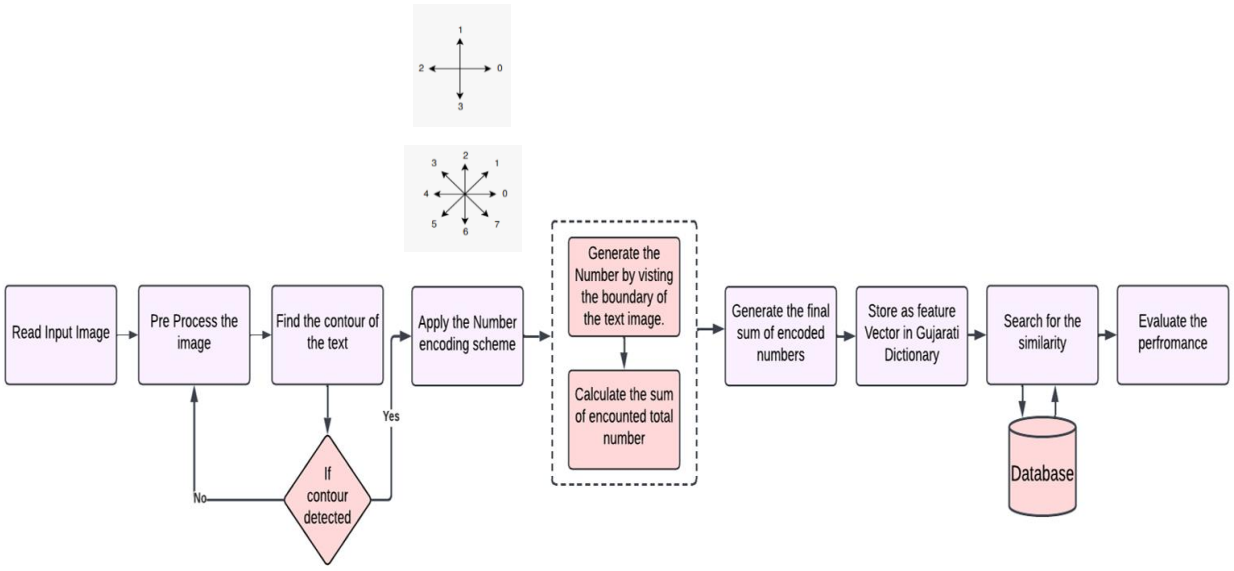
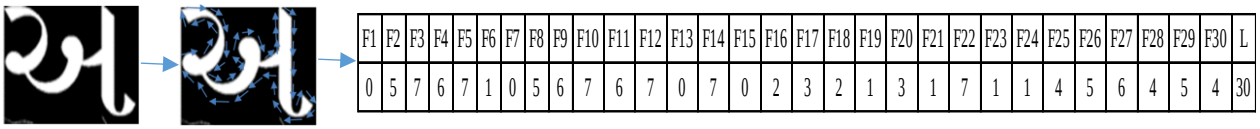


Figure 5: Proposed detailed diagram on integrated shape numeric encoding approach (ISNEA)



F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24	F25	F26	F27	F28	F29	F30	L
0	5	7	6	7	1	0	5	6	7	6	7	0	7	0	2	3	2	1	3	1	7	1	1	4	5	6	4	5	4	30

Figure 6**:** Numeric Encoding Feature for Gujarati Alphabet ‘A’

As we can see in the above figure, by using the Number coding scheme, we have labeled the boundary of the text ‘Aa’ based on the arrow direction. The overall sum will be calculated as a feature vector and will be stored in the Gujarati Feature Vector dictionary. This entire process has been discussed in the algorithm below.

Algorithm 1 Integrated Shape Numeric Encoding Approach (ISNEA) to Extract Gujarati Text Features

Input: *Gujarati Text OCR Images*
Output: *Extract the Gujarati text features and generate the feature vector file.*
Procedure:

Step 1: Preprocess the Input Image
- Convert to grayscale if necessary
- Apply binary thresholding to separate foreground (text) from background.

Step 2: Extract Contours
2.1 Use the contour detection method to find boundary contours in the image
- If no contours are found:
  Go back to Step 1 (image preprocessing)
- Else, proceed to generate numeric encoding:

2.2 Select the first external contour:

$$C = \{(x_0,y_0), (x_1,y_1),.. (x_n, y_n)\} \quad (3)$$

For each point in the contour, compute numeric encoding $N_{ei} = \{ne_0, ne_1, ..., ne_{n-1}\}$ (4)

where n_{ci} = direction $((x_i, y_i), (x_{i+1}, y_{i+1}))$

Step 3: Compute Relative Position for Direction

Encoding

3.1 For each consecutive point pair (x_i, y_i) and (x_{i+1}, y_{i+1}):

$\Delta_x = x_{i+1} - x_i$ (5)

$\Delta_y = y_{i+1} - y_i$ (6)

Determine direction code di based on Δx and Δy using an 8- direction table

e.g., if Δx=1 and Δy=0, then direction = 0 (right)

if Δx=-1 and Δy=-1, then direction = 5 (down-left).

Traverse along the boundary of the text over the entire image

Step 4: Generate the Final Feature Vector and the final feature score as mentioned in **equ.(2)**

Step 5: Store Feature Vector

- Save the resulting vector in a Gujarati Dictionary File for future retrieval

Step 6: Match Feature Vector

- Compare with existing entries in the dictionary using Euclidean and cosine distance
- Retrieve the most similar match

Step 7: Calculate the performance of the proposed system

- precision, recall, and F1-score.

The numeric Encoding is presented in the table below, where NE3 is the numeric value for image 3, NE14 for image 14, and so on. Alongside L is the overall length of the features together, and stored in the Gujarati dictionary for the matching criteria.

Table 4: Distinct Text Features of ISNEA Approach (NE3= Numeric Encoding for image-3, L=length)

NE	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	L
NE3	7	7	0	6	7	7	5	5	6	5	6	4	4	5	1	1	1	0	3	3	2		85
NE14	7	0	7	0	0	6	7	5	5	6	4	1	3	4	3								58
NE29	7	7	7	0	7	0	6	7	7	5	5	6	5	6	4	5	2	1	1	0	4	3	95
NE58	1	2	0	1	7	0	7	5	5	6	4	4	5	3	4	3	4	3	2	3	2	3	74
NE60	7	0	7	0	7	0	6	7	7	5	5	6	5	6	4	2	1	1	0	3	3		82
NE70	7	7	7	1	7	0	6	7	7	5	5	6	5	7	4	2	1	1	1	3	3		92

The proposed ISNEA approach encounters difficulties in the following scenarios:

1. Ignore inner or secondary contours
2. Fail to incorporate positional relationships between regions
3. Produce a single-directional feature that may be insufficient for full shape representation.

To overcome these issues, we have proposed the new approaches OBFCA and FRGF which will be discussed in the following section.

3.2 One-Bit frequency count approach (OBFCA):

This method is dividing into three stages. Stage-I generating the thin boundary of the input image, stage-II generating the char array of the thin boundary image and stage-III store char array as feature vector in Guajarati Text dictionary. The proposed detailed diagram is shown in Fig. 7.

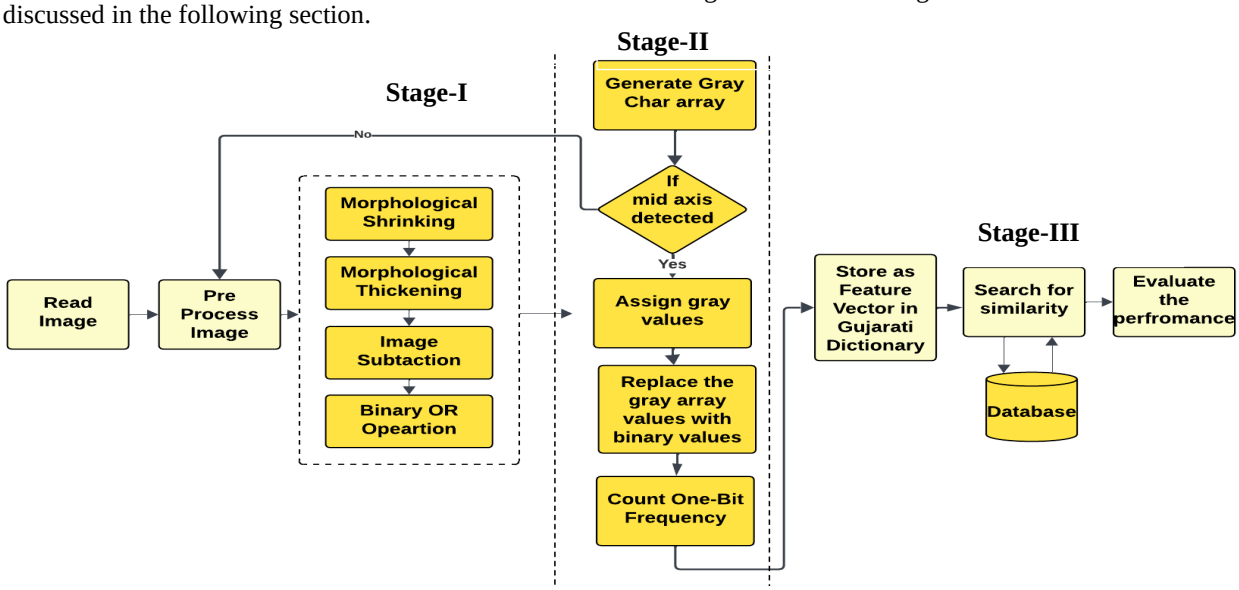


Figure 7: Proposed detailed diagram for one-bit frequency count approach (OBFCA)

The detailed explanation of the proposed approach is mentioned below.

Algorithm 2 *One-Bit Frequency Count Approach (OBFCA)*

Input: *Gujarati Text OCR Images*
Output: *Extract the Gujarati text features and generate the feature vector file.*
Procedure:
Step 1: Preprocessing
- Read a grayscale image
- Apply binary thresholding (invert background)
Step:2 Create a black mask the same size as the image
2.1 Apply iterative morphological thinning to generate a skeleton image
2.2 Save skeleton image.
Step:3 Convert Skeleton to Grayscale
3.1 Read the skeletonized image
3.1 Convert to grayscale
Step:4 Extract Gujarati Text Features
4.1 Find contours of the characters
4.2 Draw bounding boxes if necessary
4.3 Convert grayscale pixel matrix to DataFrame
4.3.1 Replace all pixel values: 255 ➔ 1 (foreground), 0 ➔ 0 (background).
Step 5: Save Feature Matrix
- Save the binary DataFrame to Excel file
Step 6: Post-process Feature Matrix
6.1 Replace 0 values with NaN if required
6.2 Count number of 1's (foreground pixels) in each row
6.3 Sum the counts to obtain final feature length
Step 7: Search in Reference Dataset
7.1 Load reference dataset from stored feature vectors
7.2 Match rows where 'length' matches the calculated feature sum
Step 8: Store Feature Vector
Save the resulting vector in a Gujarati Dictionary File for future retrieval
Step 9: Feature Vector Comparison
9.1 Calculate Euclidean distance between test image feature and reference vectors
9.2 Find the reference vector with minimum distance
Step 10: Generate the Output
10.1 Report the matched Gujarati character with the minimum distance
10.2 Report the minimum Euclidean distance value
Step:11 Calculate the performance of the proposed system
- precision, recall, and F1-score

In the proposed study, the generated character array will have all pixel values ranging from 0 to 255 replaced with 1, while the remaining values will be set to 0. This results in a binary character array. Upon zooming out of the character array, it becomes evident that Figure d illustrates the shape of the Gujarati text 'A' with a length of 61. This process has been applied to all 23,100 images, converting them from grayscale values to binary values to obtain the final count. Therefore, our proposed method effectively extracts the slender shape of the Gujarati text, as demonstrated in the image below.

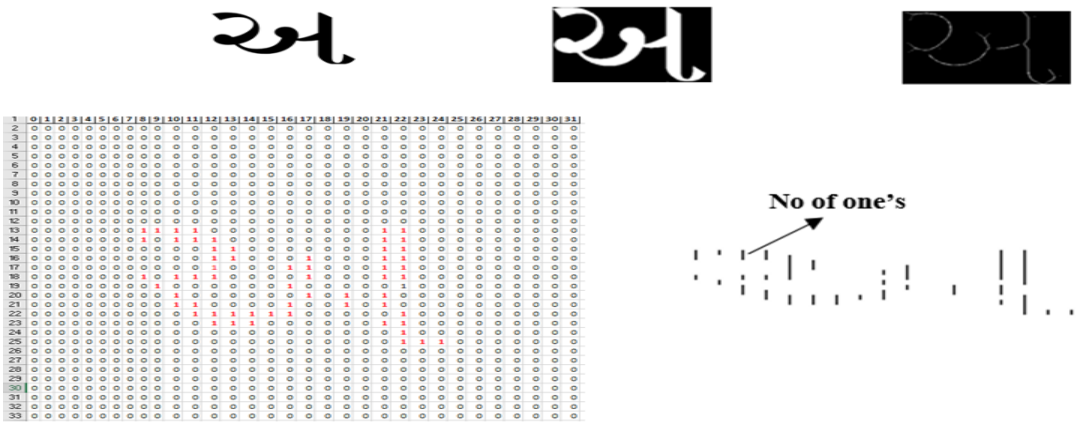


Figure 8: (a) Input Image (b)Preprocessed Image (c) Thin Boundary Extraction of image (d) Char array with 0's and 1's value (e) Text Shape of Gujarati text 'A' (See the results from left to right)

The experiment was conducted on 23100 images and extracted features labelled as 1(1),1(2) and 1(60) as discussed single alphabet is having 60 variations. Wherever the boundary or the mid axis is present it will assign values from 1-255 and rest will be assigned as 0.

In the below table six samples of the images shown where row-1 belongs to 1(1), row-2 belongs to 1(2) and so on. This table represents the features of the 1st six images 1(1)-1(6) of alphabet 'A'.

Table 5: Distinct text features of one-bit frequency count approach

F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24	F25	F26	F27	F28	F29	F30	F31	F32
0	0	0	0	0	0	0	0	0	0	0	6	6	4	5	5	7	3	4	5	7	5	1	3	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	4	2	3	3	6	6	4	4	1	2	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	3	4	4	6	7	5	5	6	2	3	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	4	4	4	5	6	6	6	9	1	3	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	6	5	4	5	4	7	2	4	5	7	5	1	3	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	3	4	2	5	5	6	5	6	5	4	5	2	2	2	0	0	0	0	0	0	0

The experimental results will be discussing in the result analysis section.

3.3 Fusion of region geometric features (FRGF)

In the context of image processing, the term "region" typically denotes a connected and coherent segment within an image that exhibits similar characteristics, such as intensity or color. When referring to text within images, a text region specifically indicates an area where text is located. These text regions play a crucial role in tasks such as text detection and recognition in the field of computer vision. Text detection represents the preliminary phase in which text is identified within the image; once these text regions are recognized, they are frequently classified as Regions of Interest (ROIs). Optical Character Recognition (OCR) algorithms can then be utilized to identify and extract the actual textual content from these designated areas. In this study, we have examined 23,100 samples of Gujarati text and identified the corresponding text regions. Additionally, we have further categorized the images into various regions. The accompanying image illustrates the different regions associated with the alphabet 'A'.

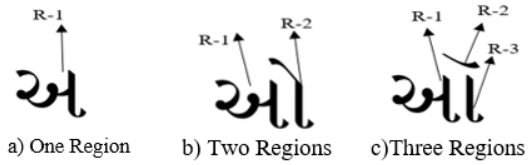


Figure 9: Region representation of gujarati text

We have similarly identified all regions within the Gujarati text images. Various geometric characteristics will be analyzed, including area, centroid, eccentricity, perimeter, and length. For the purposes of this experiment, we have excluded the Bounding Box, as the text size is consistently 32x32, resulting in no distinctiveness across all 23,100 samples. In contrast, the remaining five geometric features exhibited unique values, and a composite feature has also been generated and recorded in the Gujarati text feature vector dictionary shown below.

Region	Area	Centroid	Bounding Box	Eccentricity	Perimeter	Length	Image
1	945	(15.375661375661375, 15.42962962962963)	(0, 0, 32, 32)	0.115472549	220.25483	38.22807	1 (41).jpg

A detailed illustration of this process is provided in the diagram below.

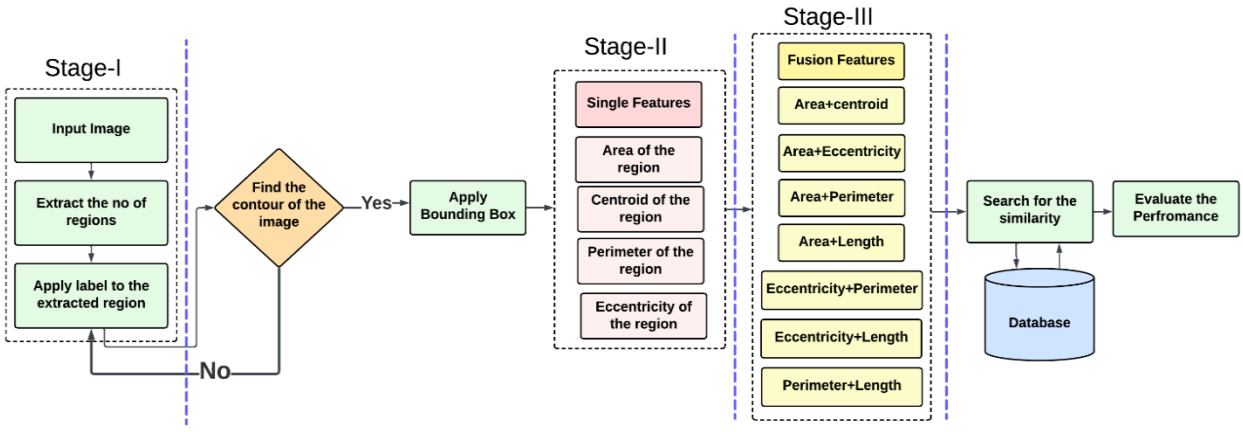


Figure 10: Detailed proposed diagram for fusion of region geometric features (FRGF)

Algorithm 3 *Fusion of Region Geometric Features (FRGF)*

Input: *Gujarati Text OCR Images*

Output: *Extract the Gujarati text features and generate the feature vector file.*

Procedure:

Step:1

1.1 Specify the folder path containing input images

1.2 Get a list of all image filenames (with extensions: .png, .jpg, .jpeg)

Step:2 Convert input image to binary image

Step:3 Create an empty list to store DataFrames for each processed image.

Step:4 Process Each Image

4.1 For each image file:

4.1.1 Read the image

4.1.2 Convert image to grayscale

4.1.3 Apply a threshold= 0.5 to binarize the grayscale image.

Step:5 Apply the label to the extracted regions:

5.1 Traverse each pixel *I(x,y)* in the binary image.

5.2 For each unvisited foreground pixel *I(x,y)= 1*, initiate a new region.

5.3 Iteratively label all connected pixels with the same label until the entire connected region is covered.

5.3.1 Counting Regions: Let *L* be the labelled image where each connected region has a unique label.

The total number of distinct labels *n* in *L*, excluding the background label, represents the number of regions in the text mentioned in **Fig.9.** So mathematically entire process will be described as: Let R_K be the k^{th} connected region in *L*, where *k=1,2,3…..n.* Then, the number of regions *N* can be represented as***:***

$$\boldsymbol{N = \sum_{k=1}^{n} \delta(R_k)} \quad (7)$$

Where $\boldsymbol{\delta(R_K)}$ =1 for each unique connected region $\boldsymbol{R_K}$, and n is the number of distinct labels in L.

Step:6 For each detected region

6.1 Calculate the following properties:

6.1.1 Calculate the area: The area of a region is its total number of pixels. Where N and M are the dimensions of the image, and R(i,j) is the binary value (1 for the region, 0 for the background) at pixel (i,j).

6.1.2 Calculate the Centroid: It is the average location of all its pixels, is its center of mass. In terms of math, the centroid is calculated as:

$$\boldsymbol{Cx = \frac{1}{A}\sum i \sum j \, . \, x_{ij}} \quad (8)$$

6.1.3 Calculate the Eccentricity: For a binary image, where a set of connected pixels represents a region, the eccentricity can be computed using the second central moments of the region. The eccentricity (e) is defined as:

$$\boldsymbol{e = \sqrt{1 - \frac{b^2}{a^2}}} \quad (9)$$

Where (a) is the length of the major axis and (b) is the length of the minor axis.

6.1.4. Calculate the Perimeter: It is the length of a region's border. It offers details on the object's intricacy and form.

$$\boldsymbol{p = \sum_{i=1}^{N} X \sqrt{(x_i - x_{i-1})^2 + (y_i - y_i - 1)^2}} \quad (10)$$

6.1.5 Calculate the length

Step:6 Store the extracted single features in the Gujarati text feature vector dictionary.

Step:7 Combined the single features and generate the fusion features and stores in Gujarati text feature vector dictionary

Step:11 Calculate the performance of the proposed system

- precision, recall, and F1-score

The experiment was conducted over 23100 images. The generated single features sets and fusion feature set of Image 'Aa' is generated successfully. Few feature set's values can be seen in the below table

Table 6: Single Feature set representation (A=Area, C1and C2= Centroid, ECC=Eccentricity, PERI=Perimeter, LEN=length)

Image	A	C1	C2	ECC	PERI	LEN
1 (1).jpg	949	15.4056902	15.45626976	0.117048599	218.4264069	38.18307376
1 (2).jpg	989	15.43781598	15.4479272	0.053172565	184.6984848	37.52549141
1 (3).jpg	965	15.38756477	15.37202073	0.103694798	190.5269119	37.94583387
1 (4).jpg	946	15.51162791	15.52854123	0.155878798	207.491378	38.32901689
1 (5).jpg	952	15.40336134	15.82983193	0.166707461	219.0121933	38.10294285

Also we have extracted fusion features which has been presented in the below table.

Table 7: Fusion feature set representation (ARCE= Area +Centroid, AREC= Area+ Eccentricity, ARPE=Area + Perimeter, ARLE=Area + Length, ECPE=Eccentricity + Perimeter, ECLE=Eccentricity + Length, PELN=Perimeter + Length)

Image	ARCE	AREC	ARPE	ARLE	ECPE	ECLE	PELN
1 (1).jpg	979.86196	949.1170486	1167.426407	987.1830738	218.5434555	38.30012236	256.6094806
1 (2).jpg	1019.885743	989.0531726	1173.698485	1026.525491	184.7516574	37.57866398	222.2239762
1 (3).jpg	995.7595855	965.1036948	1155.526912	1002.945834	190.6306067	38.04952867	228.4727458
1 (4).jpg	977.0401691	946.1558788	1153.491378	984.3290169	207.6472568	38.48489569	245.8203949
1 (5).jpg	983.2331933	952.1667075	1171.012193	990.1029429	219.1789008	38.26965031	257.1151362

3.4 Convolution neural network for gujarati text feature extraction

A Convolutional Neural Network (CNN) model was developed using TensorFlow and Keras to extract significant features from Gujarati text within the image. This network is tailored to handle grayscale images resized to 64×64 pixels. It comprises three convolutional layers utilizing ReLU activation functions, interspersed with max-pooling layers that systematically diminish spatial dimensions while emphasizing essential features. The output from the final convolutional layer is flattened and directed through a dense layer to acquire high-level representations of the Gujarati script. The output layer consists of 10 neurons employing softmax activation, facilitating multi-class classification to identify various characters or word categories. The model was compiled with the Adam optimizer and trained using sparse categorical cross-entropy loss, making it well-suited for labelled Gujarati character datasets. This architecture effectively captures spatial hierarchies and specific script patterns, which are crucial in OCR-based pipelines for extracting features from Gujarati text.

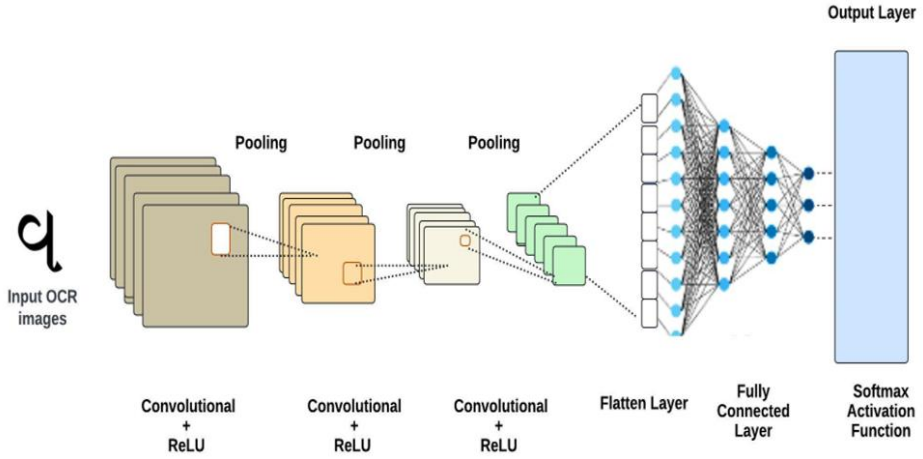


Figure 11: Detailed CNN architecture

The architecture is composed of the following main layers:

1. Input Layer:
Takes a 64×64 grayscale image (single channel) of a Gujarati character as input. (The input tensor shape is 64×64×1).

2. Convolutional Layers (×3):
This architecture consists of three 2D convolutional layers, each utilizing a collection of learnable filters (kernels) to analyze the image and generate corresponding feature maps. Following each convolution, a ReLU (Rectified Linear Unit) activation function is applied, introducing non-linearity into the model. The initial convolutional layer may employ 32 filters of size 3×3 to identify basic patterns such as edges and strokes within the raw image. The subsequent layers, utilizing 64 and 128 filters, respectively, progressively learn more intricate features by integrating the lower-level features extracted from the preceding layers. These layers are the main feature extractors, effectively learning significant visual patterns from Gujarati characters.

3. Max-Pooling Layers (×2):
A max-pooling layer utilizing a 2×2 window is implemented. This pooling operation reduces the size of the feature maps by selecting the maximum value from each 2×2 area, effectively halving the spatial dimensions. Consequently, the image size is diminished, enhancing translation invariance while preserving the most significant features. Applying two such pooling layers systematically compresses the spatial information, allowing the model to concentrate on critical features and minimize computational demands.

4. Flatten Layer:
After the last convolution layer and its corresponding activation, the two-dimensional feature maps are transformed into a one-dimensional vector through a process known as "flattening." This layer converts the

multi-dimensional feature tensor into a singular 1D feature vector. Flattening is essential for preparing the extracted features for input into the dense, fully connected layer.

5. Dense Hidden Layer:
This fully connected layer consists of many neurons, such as 128 or 512, utilizing ReLU activation. It receives the flattened feature vector as its input. The purpose of this layer is to integrate and assign weights to the diverse features identified by the convolutional layers, enabling the learning of more advanced representations. The dense layer identifies which combinations of the extracted features most represent specific Gujarati characters. For instance, certain neurons may become active in response to features that represent a unique curved stroke or loop, helping to differentiate one character from another.

6. Output Layer (Softmax):
The concluding layer is a fully connected output layer featuring a neuron for each character class that the system is designed to identify. For example, suppose the model is developed to recognize all Gujarati letters and potential numerals. In that case, the output neurons will match the number of distinct Gujarati character classes. Each output neuron employs a softmax activation function, which generates a probability indicating the likelihood that the input image corresponds to that specific class. The network's prediction is determined by selecting the class with the highest probability. In one instance of a Gujarati character recognition model, the output layer consisted of 374 neurons, each representing one of the 374 character classes, utilizing softmax for multi-class classification.
The proposed CNN network is the most lightweight model, characterized by fewer parameters, resulting in quicker training and faster predictions.

Layer Type	Output Shape	Param#
Conv2D (32 filters)	(62, 62, 32)	320
MaxPooling2D	(31, 31, 32)	0
Conv2D (64 filters)	(29, 29, 64)	18,496
MaxPooling2D	(14, 14, 64)	0
Conv2D (64 filters)	(12, 12, 64)	36,928
Flatten	(9216,)	0
Dense (64 units)	(64,)	589,888
Dense (10 units)	(10,)	650

Figure 12: Model Design of CNN model

The CNN model's training and validation loss curves over ten epochs are shown in the image. The training loss can be observed by the blue dashed line, while the validation loss is illustrated by the red solid line. Both losses show a decreasing trend as training goes on, suggesting that the model is learning and generalizing successfully. The steady decrease in validation loss and training loss indicates that overfitting was not present during these periods. The model maintains a decent balance between training accuracy and generalization performance on unseen data, as evidenced by the moderate gap between the two curves. This illustrates the model's resilience and steady convergence over time.

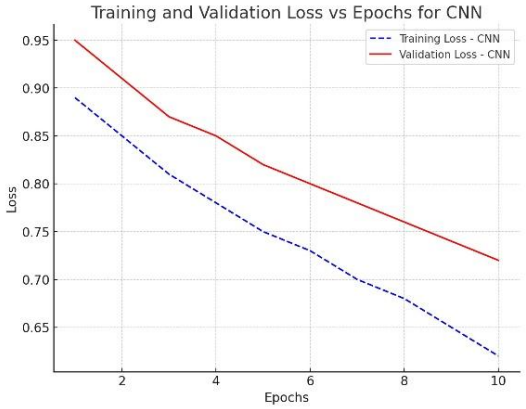


Figure 13: Training & Validation Loss Vs Epoch

4 Experiments and result discussion

The literature indicates that the complexity and lack of available datasets for the Gujarati language necessitated the creation of a significant dataset by collecting handwritten documents from users, which encompass both numbers and letters. In this study, we utilized the Gujarati OCR dataset and introduced three distinct techniques for text feature extraction. The Gujarati language dataset has been previously addressed in the literature review, as shown in Table 1 [7] - [23]. Consequently, it is essential to employ a statistical evaluation to assess the proposed methods' performance accurately. The effectiveness of the proposed approach is tested on the Gujarati OCR dataset, and its performance is compared with that of leading methods using established performance metrics. The experimental results will be discussed in the results section.

4.1 Results and discussion

This study is based on extracting Gujarati text features. This section presents the experimental results, which are quantitatively assessed based on the available Gujarati text, to demonstrate the efficacy of the proposed method. A quantitative analysis of the proposed work is discussed in this section.

4.1.1 Performance parameters of the proposed approaches

This section presents a quantitative analysis of the proposed methodology. The experiments are carried out using Gujarati OCR datasets, as outlined in Section 3.1,

to evaluate the effectiveness of the proposed approach. The accuracy of the text feature extraction model is assessed through four statistical measures. Positive Predictive Value evaluates the correctly identified text features, while Recall assesses the model's capability to identify all relevant features or terms within a text accurately. True negatives represent instances where the model accurately recognizes the absence of a specific feature. The F1 Score, the harmonic mean of precision and recall, ranges from 0 to 1, with 1 indicating perfect precision and recall and 0 reflecting poor performance in one or both metrics. Lastly, the percentage of correct classification (PCC) serves as a crucial evaluation metric, measuring the overall effectiveness of the model in accurately identifying text features. In text feature extraction, PCC quantifies the proportion of instances (such as words, phrases, or documents) the model correctly classifies as belonging to a specific category or possessing a particular feature, such as a keyword.

$$\text{Recall} = \frac{tp}{tp + fn} \tag{11}$$

$$\text{Precision} = \frac{tp}{tp + fn} \tag{12}$$

$$\text{F} - 1\ \text{Score} = 2 \times \frac{Precision \times Recall}{Precision \times Recall} \tag{13}$$

$$\text{PCC} = \frac{tp + tn}{tp + fp + tn + fn} \tag{14}$$

The proposal aims to identify and recognize OCR Gujarati text through various image processing techniques. The extracted features were compiled into a Gujarati Text Feature Vector Dictionary, which was the basis for conducting search and matching operations. For the purposes of this experiment, a random selection of 1,000 samples was utilized.

4.2.2 Discussion on text feature set values

The proposed technique, ISNEA, has been developed utilizing the Numeric Encoding Scheme, resulting in the generation of text features. In contrast, the OCR Gujarati dataset, which included various modifications, yielded a significantly higher accuracy of 87.50% PPC. The feature set values for 60 images are illustrated below, with the horizontal axis representing the number of images and the vertical axis displaying the feature vector score. The maximum value recorded is 279, indicating that a single text can exhibit numerous variations in direction. The length features derived from the ISNEA approach have been successfully generated for 23,100 samples and plotted for 60 samples of individual Gujarati text images in the graph below.

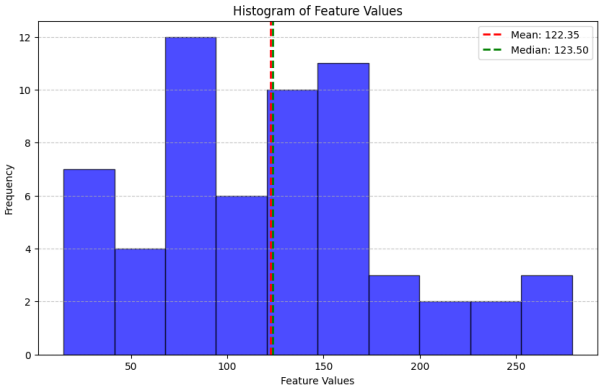


Figure 14: Single ISNEA feature set values (n=60)

The below graph analysis indicates that the feature values exhibit significant divergence within the range of 10 to 279. The second method, OBFCA, involves extracting the image's central axis through various morphological operations. Subsequently, the proposed solution is implemented on the resulting image by creating a gray character array, which is then converted into a binary character array, and the frequency of 1's is counted row-wise and stored as features in a dictionary as discussed in section 3.2. The composite features derived from OBFCA were successfully calculated and are illustrated in the graph presented in Figure 15 below.

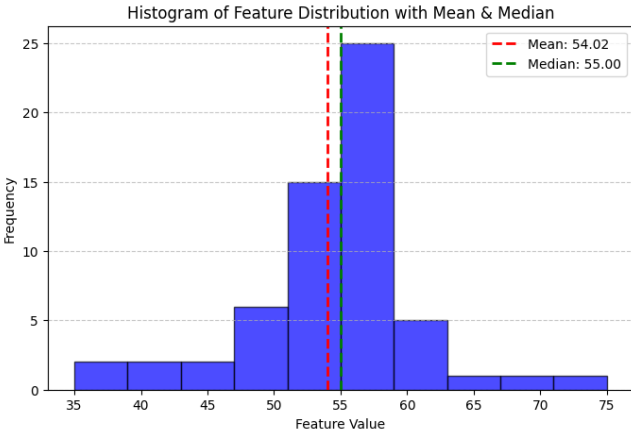


Figure 15**:** Single OBFCA feature set values (*n*=60)

From the above graph, we can identify the features that are classified between the range of 40 – 70.

The third approach is the Fusion of Region Geometric Features (FRGF), in which geometric features like area, perimeter, length, and eccentricity are generated for 23100 samples. For the proposed work, two sets of features were generated, the first one with the distinct single features and the second with fusion features. All the single and fusion feature set values will be stored in the Gujarati text feature vector dictionary. The features' feature set values are represented in the graphs below.

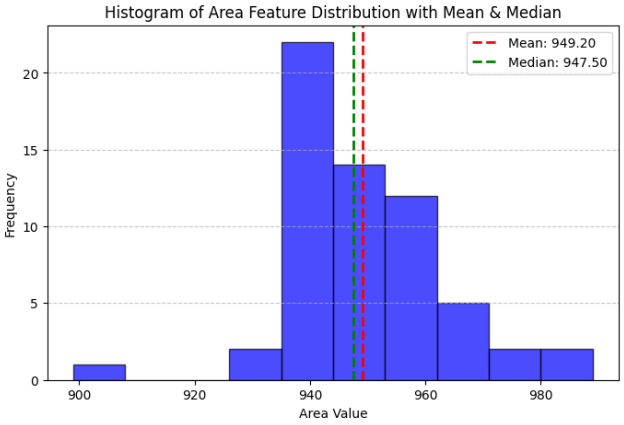


Figure 16**:** Area feature set values (*n*=60)

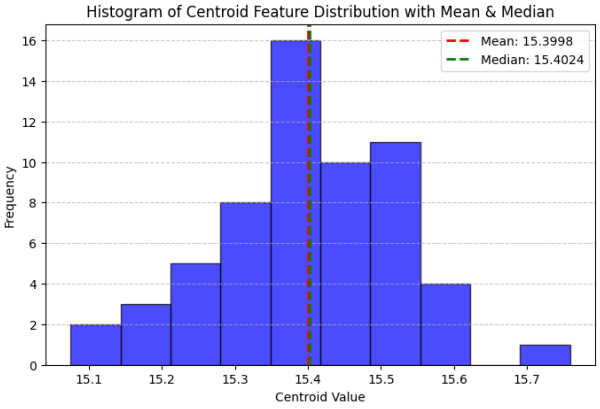


Figure 17**:** Centroid feature set values (*n*=60)

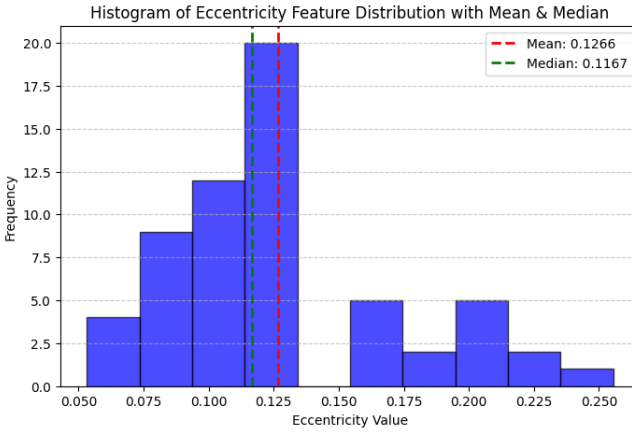


Figure 18**:** Eccentricity feature set values (*n*=60)

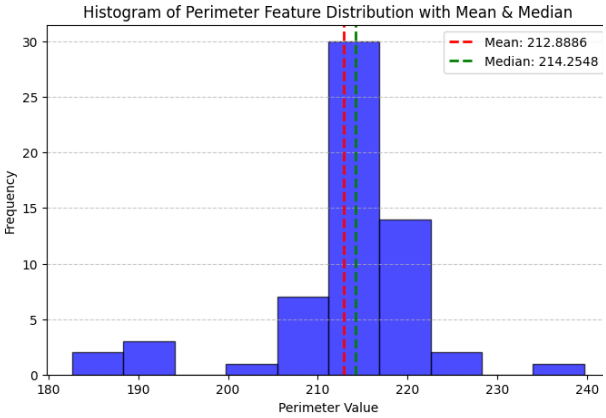


Figure 19**:** Perimeter feature set values (*n*=60)

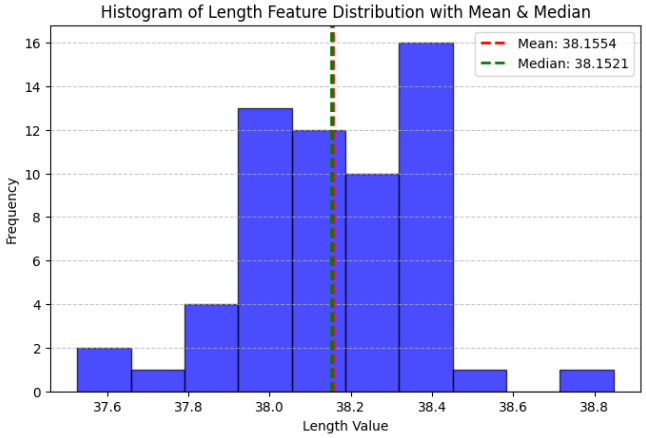


Figure 20**:** length feature set values (*n*=60)

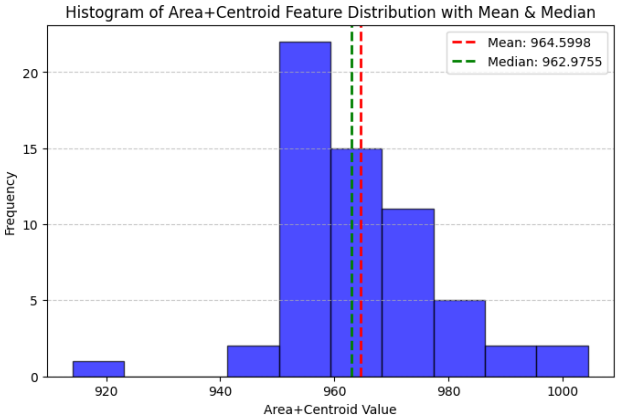


Figure 21: ARCE fusion feature set values (*n*=60)

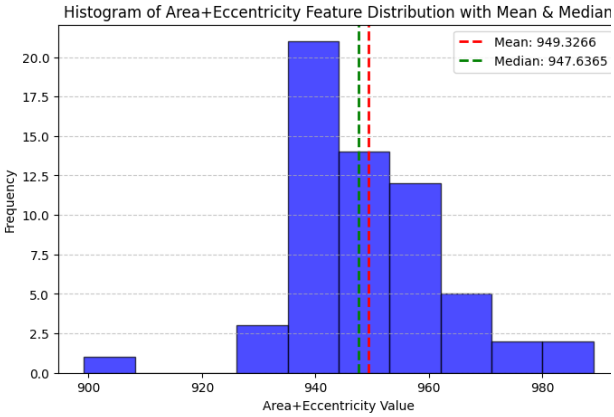


Figure 22: AREC fusion feature set values (*n*=60)

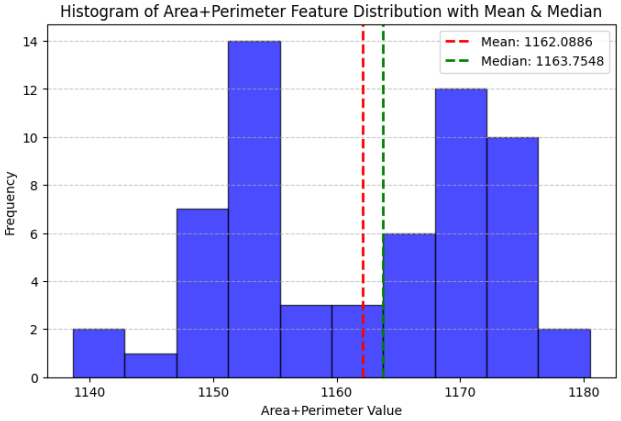


Figure 23: ARPE fusion feature set values (*n*=60)

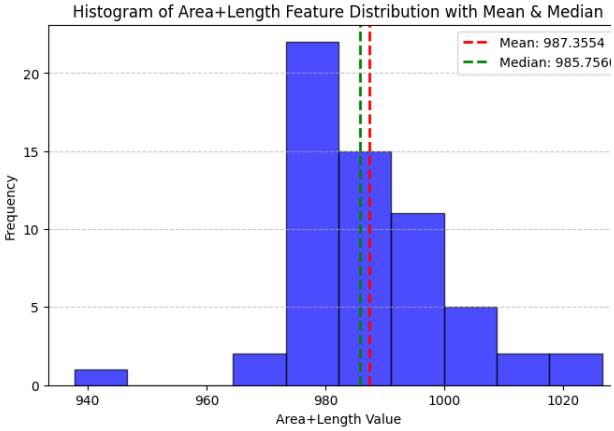


Figure 24: ARLE fusion feature set values (*n*=60)

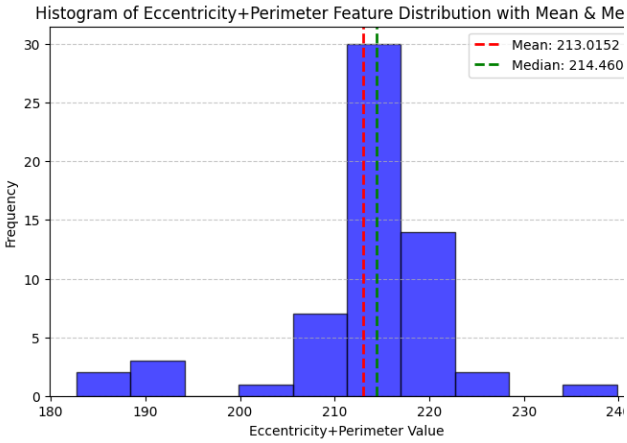


Figure 25: ECPE fusion feature set values (*n*=60)

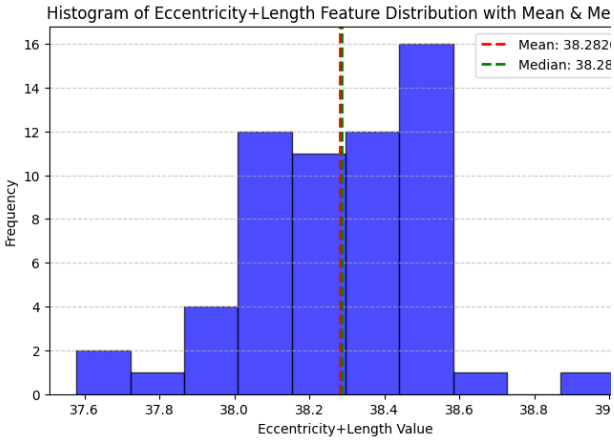


Figure 26: ECLE fusion feature set values (*n*=60)

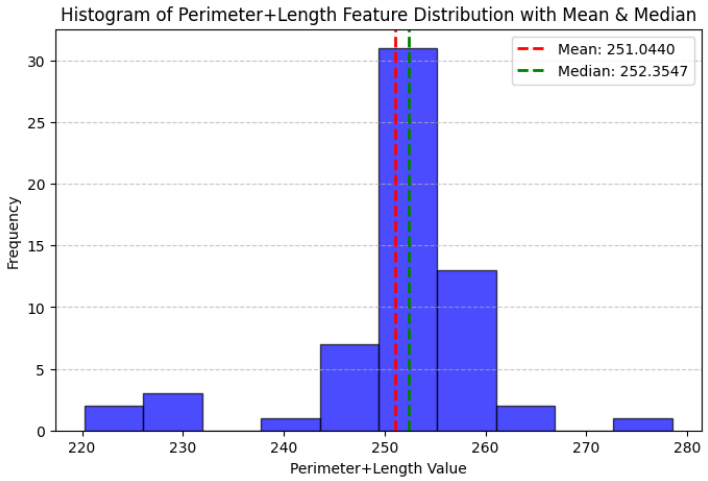


Figure 27: PELE fusion feature set values (*n*=60)

The above 12 graphs show the feature set values of 60 samples from 23100 samples. The single feature, Eccentricity with 93.50% and fusion Eccentricity with perimeter ECPE with 92.75% gives the best performance. From the analysis, we can say that the third approach performs well compared to the first two. The proposed method successfully generates the Gujarati text features, identifies correct text features, and classifies the text features based on the shape and region of the Gujarati text.

Table 8: Statistical feature mean and median values of text feature extraction approaches

Approaches	Mean	Median
ISNEA	122.35	123.5
OBFCA	54.02	55
A	949.2	947.5
C	15.3998	15.4024
ECC	0.1266	0.1167
PERI	212.8886	214.2548
L	38.1554	38.1521
ARCE	964.5998	962.9755
AREC	949.3266	947.6365
ARPE	1162.0886	1163.7548
ARLE	987.3554	985.756
ECLE	38.282	38.2874
ECPE	213.0152	214.4604
PELE	251.044	252.3547

The table compares the Mean and Median values associated with various feature extraction techniques utilised in the study. The ISNEA and OBFCA methods exhibit mean values of 122.35 and 54.02, respectively, with slightly elevated median values of 123.5 and 55, which indicates a minor variation in their distributions. In the context of geometric features, the Area demonstrates a mean of 949.2, which is closely matched by its median of 947.5, implying a well-balanced distribution. The Centroid values reveal minimal variation, with a mean of 15.3998 and a median of 15.4024. Likewise, Eccentricity, which quantifies shape elongation, presents a low mean of 0.1266 and a median of 0.1167, indicating slight variability. For Perimeter and Length, the mean values (212.8886 and 38.1554) are near their respective medians (214.2548 and 38.1521), suggesting that the distributions are nearly symmetrical.

The ARCE and AREC methods yield similar mean values (964.5998 and 949.3266) alongside corresponding medians (962.9755 and 947.6365). The ARPE and ARLE methods report the highest mean values (1162.0886 and 987.3554), with medians showing slight deviations at 1163.7548 and 985.756, respectively. The ECLE and ECPE methods display minimal discrepancies between their mean and median values, indicating stable distributions. Finally, PELE reveals a mean of 251.044 and a median of 252.3547, suggesting consistency in the extracted values. In summary, the minor differences observed between the mean and median values across most features imply a relatively normal distribution with minimal skewness, thereby ensuring reliable and consistent outcomes in feature extraction. The accuracy of both curves gradually decreases as the number of epochs rises.

We have also applied a CNN model over the Gujarati OCR dataset. The training accuracy declines from roughly 0.89 to 0.53; the validation accuracy likewise declines in the same manner. This trend shows that the model performs reasonably well at first, but as training goes on, it steadily underfits the data. A decrease like this points to problems like insufficient training data to sustain generalization, an incorrect learning rate, or an inappropriate model complexity. Even if it is slight, the growing difference between the training and validation curves suggests that the model is gradually becoming less able to identify significant patterns in the data. In order to attain more consistent and enhanced performance, this analysis emphasizes the necessity of model tuning, such

as learning rate modification or architectural improvement. Training and validation accuracy of CNN model over 10 epochs in a single batch were presented in the given diagram below.

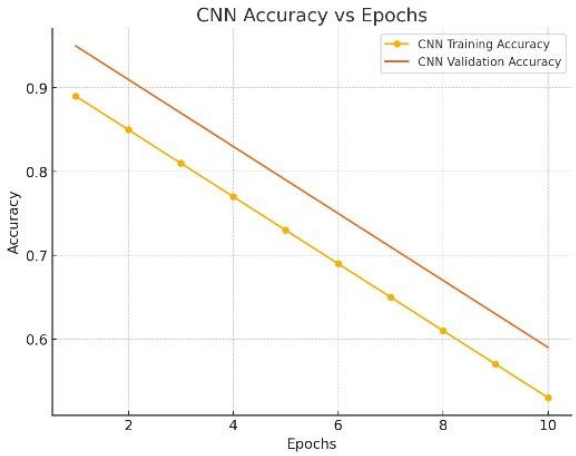


Figure 28: Training and validation accuracy of CNN

The quantitative result analysis for feature classification of all the proposed work is mentioned in Table 9.

Table 9: Quantitative analysis of proposed text feature extraction approaches

Approaches	Feature Type	%PPC
ISNEA	Single Feature	0.875
OBFCA		0.9000
A		0.9242
C		0.8979
ECC	Fusion Features	0.9350
PERI		0.9275
LEN		0.9000
ARCE		0.9050
ARPE		0.9250
AREC		0.9200
ARLE		0.9000
ECPE		0.9275
ECLE		0.9100
PELE		0.9200
CNN		0.8900

The Gujarati OCR dataset demonstrates commendable performance despite the challenges posed by regional disconnectivity, which results in incomplete numeric encoding. The proposed OBFCA approach shows improved performance relative to ISNEA. However, as highlighted in the research gap, there has been insufficient exploration of feature fusion, leading to the third approach excelling in feature extraction and giving the best results among the three proposed approaches. Eccentricity excels in the FRGF approach with the highest accuracy of 0.935. The graph below shows the performance comparison of Eccentricity with diction features vs fusion features.

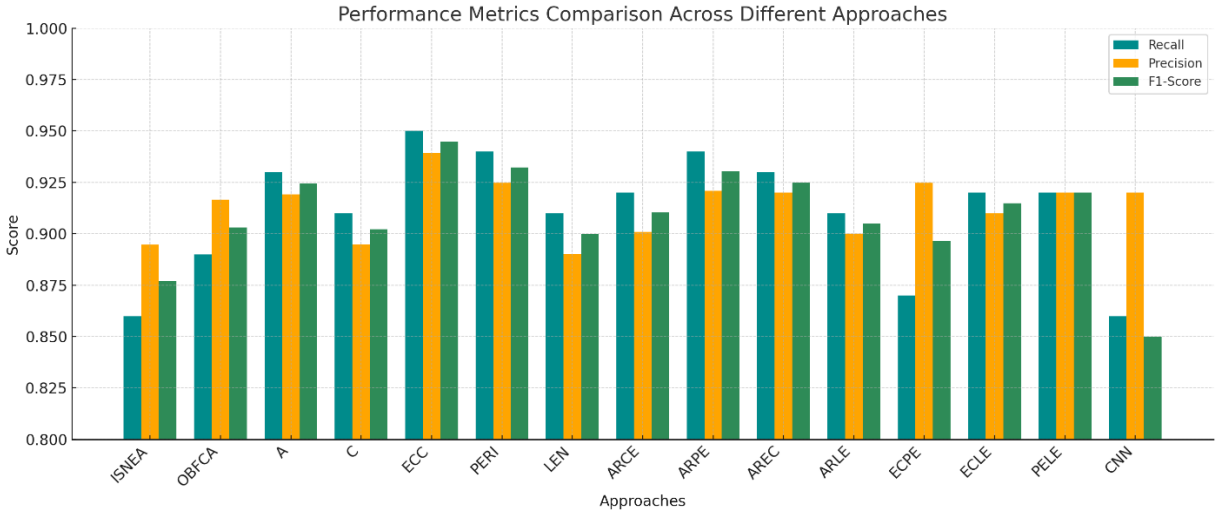


Figure 29: Performance analysis for proposed gujarati text features extraction approaches

Key Observations from the Comparative Analysis:
FRGF with Eccentricity outperforms CNNs by a significant margin, as evidenced by an F1-score of 0.9447 vs. 0.85. Moreover, ISNEA and OBFCAs provide competitive accuracy without the complexity of deep networks.
2. Robustness to Handwriting Variation: ISNEA is limited when it comes to multi-region characters like vowel modifiers or maatra.
3. Because of their skeleton-based representation, OBFCAs are resistant to these circumstances.
4. FRGF recognizes complex printed and handwritten characters using geometrical features like eccentricity and perimeter.
5. CNNs are good at handling handwriting, but they require a lot of training data to generalize and are not interpretable.
In summary, Eccentricity outperforms all other suggested strategies. Among the various validation techniques, which are listed below, eccentricity consistently performs better than others.

Table 10: Limitations of text feature extraction approaches

Feature	Limitations in OCR Context for the proposed approaches
ISNEA	Alphabets such as શ and ગ produce incomplete feature vectors because of their disconnectivity.
OBFCA	Many alphabets generate feature vectors of the same length because 1s are counted row-wise.
Area	size, stroke width, and scaling all perform an integral part.
Centroid	Lacking discrimination — the centroids of different characters are similar.
Length	It can be significantly affected by noise and is unable to capture intricate shapes.
Perimeter	Sensitive to slight variations in shape and noise.

Limitations of Features in the OCR Context of the Suggested Methods

- ISNEA: Alphabets such as શ and ગ produce incomplete feature vectors because of their disconnectivity.
- OBFCA: Many alphabets generate feature vectors of the same length because 1s are counted row-wise.
- Area It examines discrimination because different characters have similar centroids.
- Length is not able to capture intricate shapes and is highly dependent on noise.
- Perimeter Sensitive to noise and slight variations in shape

The primary characteristics of Eccentricity include:
1. Gujarati Characters' Unique Shapes
• The broad "ખ" and "ઘ" are among the many character forms found in the Gujarati script.
• Tall, slender characters like "ઙ" and "શ"; "ફ" and "જ" stand for semi-circular or circular shapes.
Eccentricity, a powerful identifier for character recognition, can be used to geometrically analyze these variations.
2. Orientation and scaling have no effect on eccentricity. Shifting and rotation have been used as augmentations in this dataset without changing the feature values.
3. The capability to capture the general form of the Gujarati alphabet is denoted by Global Shape Capture.
4. Lightweight and Effective means it's easy to calculate.

To assess the scalability and resilience of the proposed framework, further experiments were carried out using out-of-distribution (OOD) samples and various Indian scripts beyond Gujarati. These evaluations aimed to determine the model's ability to generalize in real-world scenarios characterized by factors such as reduced quality, variations in stroke, and diversity among scripts.

4.2.3 Granular analysis of proposed techniques for gujarati text feature retrieval

Euclidean Distance and Cosine Similarity are frequently used to determine how similar two feature vectors are, particularly in domains like computer vision, text analysis, and machine learning. An ablation study was performed to evaluate the importance of each module, ISNEA, OBFCA, and FRGF, in the proposed framework for extracting features from Gujarati text. Each proposed approach was assessed individually by utilizing performance metrics such as PPC, Recall, Precision, and F1-Score. Cosine similarity and Euclidean distance were used as to assess how closely extracted feature vectors in the proposed ISNEA and OBFCA techniques. The formula for the Euclidean distance is given below:

$$Euclidean\ Distance = \sqrt{(Feature1 - Feature2)^2 + \cdots (Feature\ n1 - Feature\ n2)^2} \quad (15)$$

Apart from Euclidean distance, we have also experimented with Cosine similarity, which focuses on directional alignment rather than absolute values. It's calculated using the formula:

$$Cosine\ Similarity = \frac{\sum Fearure1_i . feature2_i}{\sqrt{\sum Fearure1_i{}^2}\sqrt{\sum Fearure2_i{}^2}} \quad (16)$$

The comparison graph is shown below for the ISNEA and OBFCA proposed approaches.

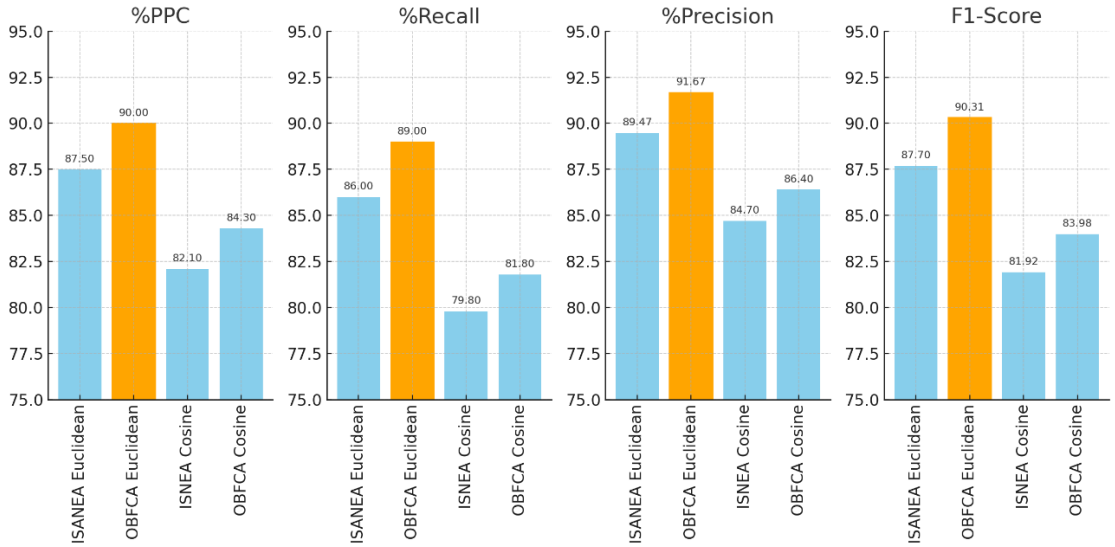


Figure 30: Performance Comparison: ISNEA OBFCA Euclidean Distance Vs Cosine Similarity (Orange bar depicts highest value)

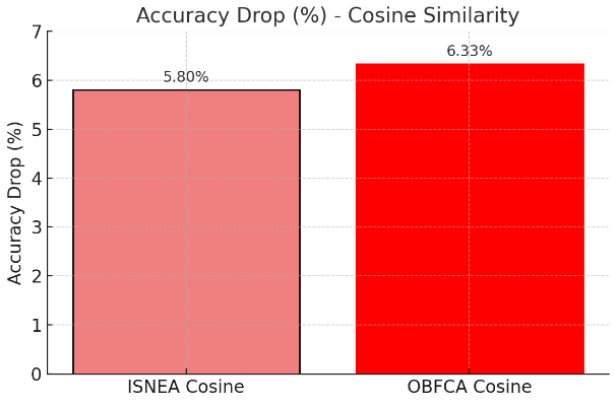


Figure 31: Accuracy Drop (%) for Cosine Similarity

The comparison graph illustrates the complete ISNEA and OBGF model performance metrics for Ablation Cosine Similarity. The accuracy loss (%) that occurs when the ISNEA and OBFCA approaches go from Euclidean distance to Cosine similarity is depicted in the bar graph. Notably, OBFCA Cosine exhibits a marginally higher accuracy decline of 6.33%, whilst ISNEA Cosine shows a 5.80% drop.

This decrease suggests that, for both approaches, Euclidean distance is more useful in this situation than the Cosine similarity measure. Reiterating the superior compatibility of Euclidean distance in maintaining classification performance in both frameworks, the larger accuracy reduction in OBFCA indicates its stronger sensitivity to the choice of similarity metric.

Two new fusions, ARPEC and AECL, were experimented with for the proposed FRGF approach, and accuracy was reported as 93.00 and 92.60, respectively, and recall was reported as low. ECC (Eccentricity) consistently dominates in all combinations, specifically when the shape is crucial, Lacks spatial localization, slightly lower precision

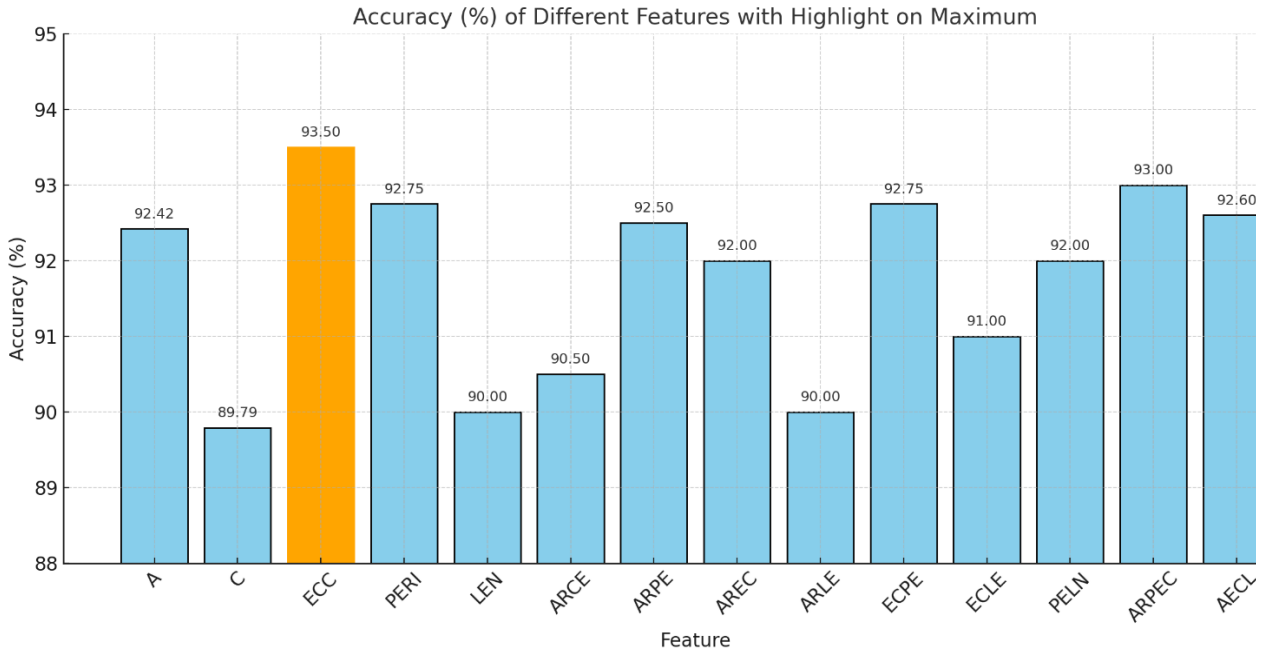


Figure 32: Performance Comparison: FRGF ARPEC Vs AECL (Orange bar depicts highest value)

Illustrates the comparative accuracy of various individual and fused text feature approaches used for Gujarati character recognition. With the best accuracy of 93.5% among the individual features, Eccentricity (ECC) demonstrated a significant ability to discriminate between different character shapes and structural complexity. With accuracy rates of 93.0% and 92.6%, respectively, fused features such as AECL and ARPEC (Area + Perimeter + Eccentricity + Centroid) also showed excellent performance. These findings imply that recognition performance is improved by integrating geometric descriptors. However, measures like Length (LEN) and Centroid (C) scored comparatively poorly, suggesting that these attributes might not be enough to adequately convey the uniqueness of Gujarati letters. All things considered, the graph demonstrates the efficacy of feature fusion techniques and identifies ECC as a crucial element in precise character classification.

4.2.4. Evaluate the statistical significance of Eccentricity using One-Way ANOVA

Using One-Way ANOVA, four feature extraction methods were statistically assessed for variations in F1-score performance. Since different groups were compared using the same measure (F1-score) under independent settings, ANOVA was the most effective method to ascertain whether the observed changes were statistically significant. The Eccentricity 5 Cross-fold was applied to the ECC, AREC, ECPE, and ECLE features in order to validate the results. Finally, an ANOVA test was performed.

Step:1 To perform the 5-fold cross-validation:

For every fold:

1. Use four folds to train the feature extraction technique, such as ECC, AREC, ECPE, ECLE.
2. Make label predictions for the test fold.
3. Determine the prediction's F1-score.

Step:2 Calculate F1-Score for Each Fold

Step:3 Collect 5 F1-scores per method.

Step:4 Use these scores as inputs for ANOVA.

Degrees of freedom in a One-Way ANOVA:

1. Calculate the Grand Mean: Average of f1-scores across 4 groups (ECC, AREC, ECPE, ECLE)

$$GM = \frac{\sum x_{ij}}{N} = \frac{Sum\ of\ all\ values}{4\ gropus\ X\ 4\ samples} = 9200 \quad (17)$$

2. Between-Group sum of Squares(SSB): This calculates the deviation of each group's mean from the grand mean.

$$SSB = n.\sum_{i=1}^{k} (x_i - GM)^2 = 0.007155 \quad (18)$$

3. Within-Group Sum of Squares (SSW): This captures variance inside each group

4. Calculate Mean Square:

$$MSB = \frac{SSB}{k-1} \approx 0.0023852 \quad (19)$$

$$MSW = \frac{SSW}{N-k} \approx 0.00000445 \quad (20)$$

5. Calculate F-statistic:

$$F = \frac{MSB}{MSW} \approx 1003.11 \quad (21)$$

6. Find p-value:

- Groups (k): ECC, AREC, ECLE, ECPE $k=4$
- Observations per group: 5
- Total observations (N): 5×4=20

d_{f1} =4−1=3 *(between groups)*
d_{f2}=20−4=16 *(within groups)*
F-observed = 1003.11

$$p - value = 1 - (F_{Observed}, d_{f1}, d_{f2}) \approx 2.04 \times 10^{-18}$$

These findings demonstrate that the feature extraction approach selection has a significant and statistically significant impact on the proposed system's classification performance. The differences between the means of ECC, AREC, ECLE, and ECPE are statistically significant. This demonstrates that not all feature extraction techniques are created equal. The impact of feature selection on OCR accuracy is further supported by the fact that at least one method, like ECC, has a statistically different (probably higher) F1-score than the others.

4.2.5 Scalability and robustness tests of proposed approaches

Additional tests were conducted in difficult and out-of-distribution scenarios to evaluate the robustness and scalability of the suggested methodology. We first created a test set of printed Gujarati characters that were noisy, low-resolution, and blurry in order to simulate degraded input scenarios. We were able to assess the model's resistance to distortions in the real world as a result. We then manually collected and tested handwritten Gujarati characters with different stroke thicknesses and curvatures to test the method's ability to generalize across stylistic differences. Finally, the model was applied to a small sample of Devanagari script characters [39] using the same preprocessing and feature extraction pipeline as Gujarati in order to assess cross-language generalization. Table 11: Scalability and Robustness Performance accuracy

Test Condition	Accuracy (%)
Clean Gujarati Printed Images [34]	93.5
Degraded Images (blur/noise)	86.0
Devanagari Script [39]	84.0

Impact of Script Variability on Recognition Performance:
1. Disparities in Script Structure
In contrast to the Gujarati script, Devanagari has a unique character structure that includes compound consonants and the shirorekha (headline). Our model might have trouble deriving significant representations from these strange structures because it was trained on Gujarati features.
2. The Effect of Domain Shift
Gujarati text was particularly used to train the model. Unless the model is specifically trained for generalization across scripts, testing it on Devanagari data introduces a domain shift, which is essentially an out-of-distribution test and usually results in worse performance.
3. Bias in Feature Representation
The hand-crafted or taught elements (such as eccentricity, perimeter, etc.) might not translate well to Devanagari shapes and stroke patterns because they are particularly tuned for Gujarati letters.

4.2.6 Analysis of computational efficiency and complexity

A thorough computational complexity analysis was conducted for ISNEA, OBFCA, and FRGF, in addition to a CNN baseline, in order to address the practical deployment feasibility of the suggested methods. With a time complexity of O (n × m), ISNEA entails edge aggregation and similarity comparison, where n stands for detected edges and m for feature comparisons. When k is the number of contour pixels, OBFCA, which encodes boundary pixel chains, functions at O(k), making it extremely effective for real-time applications. By using common image processing libraries to extract geometric region features like area, perimeter, and eccentricity, FRGF is able to achieve O(n) complexity. The convolutional depth and filter size of CNN-based methods, on the other hand, result in higher computational costs ($O(n \cdot k^2 \cdot d^2)$), even though their runtime is greatly reduced by contemporary GPUs**.**

Table 11: Computation complexity of proposed approaches

Gujarati Text Feature Extraction Approaches	Computational Complexity
ISNEA	O(n × m), where m is the number of feature comparisons and n is the number of edge segments
OBFCA	O(k), where k is the number of boundary pixels
FRGF	O(n), where *n* is the number of segmented regions
CNN	$O(n \cdot k^2 \cdot d^2)$ for k×k filters, d is for depth

4.2.7 Qualitative analysis of proposed approaches vs existing approaches

The classification performance of three feature extraction methods—ISNEA, OBFCA, and ECC—based on their capacity to differentiate between classes is contrasted in the ROC curve above. With an Area Under the Curve (AUC) value of 1.00, all three models demonstrated flawless classification performance, with neither false positives nor false negatives occurring throughout the assessed thresholds. The plot's top-left boundary, where the ROC curves for each approach lie closely, represents the best sensitivity and specificity. The performance of a random classifier (AUC = 0.5), on the other hand, is represented by the diagonal dashed line. The distinct separation of all approaches from this baseline emphasizes their greater predictive power.

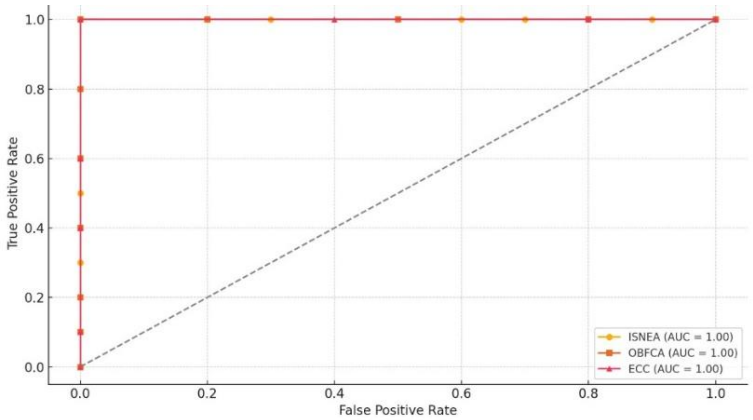


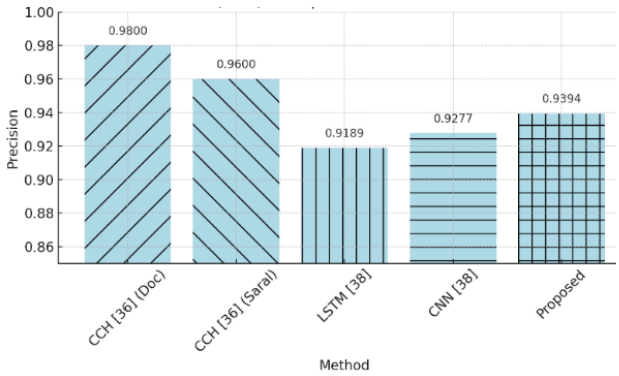
Figure 33: AUC-Based classification performance of proposed techniques

Table 11: Comparative Performance Analysis of FRGF Across Multiple Datasets

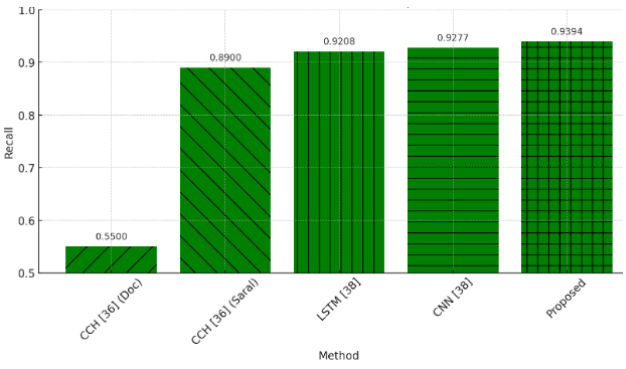
Approaches	Dataset	%PCC
CCH [36]	Book , magazines etc.	0.55
CCH [36]	Thesis Document	0.89
FVZPM [37]	Gujarati – saral	0.7984
Freeman Chain Code [40]	Different sources	0.8221
CNN+LSTM Layers [15]	Devanagari Handwritten Character Dataset	0.928
Proposed FRGF	Gujarati OCR Dataset	0.935

This table provides a performance comparison of various text recognition and feature extraction approaches on datasets involving Gujarati text. Traditional methods like HDWT and CCH show varying success, with HDWT achieving moderate accuracy and CCH struggling on structured datasets but performing better on simpler datasets. Deep learning approaches such as LSTM and CNN outperform traditional methods, achieving PCC scores above 98%.

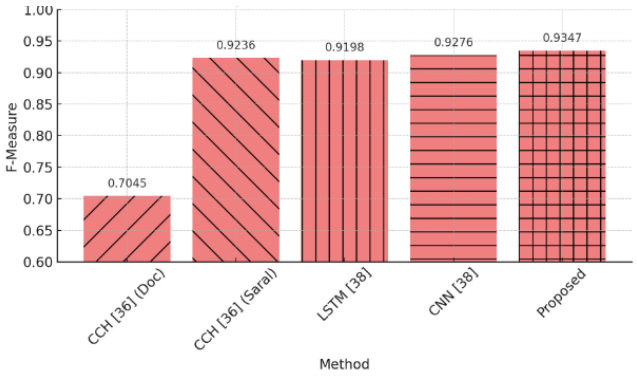
The proposed methods (FRGF ECC and FRGF ECPE) demonstrate strong performance, with PCC values above 92%. These results indicate the effectiveness of the proposed methods for the Gujarati OCR task as compared to another dataset. The different text feature extraction approaches have been compared with the existing one, and a detailed analysis has been presented in the graphs below:



a) Quantitative Precision Analysis: Proposed vs Existing Approaches



b) Quantitative Recall Analysis: Proposed vs Existing Approaches



(c) Quantitative F-Measure Analysis: Proposed vs Existing Approaches

Figure 34: (a)-(c) Quantitative analysis of the proposed method with an existing method with different language datasets (Proposed=FRGF ECC)

5 Conclusion and future scope

This study introduced effective methods for feature extraction of Gujarati text, leveraging fundamental image processing concepts. Given the inherent complexity of the Gujarati language, the research concentrated on the diverse variations of Gujarati text, encompassing 385 distinct characters, including matra's and alphabets. Three specific methodologies were proposed: The Numeric Encoding Approach (ISNEA), The ISNEA model is designed to enhance explainability. Using spatial transitions and direction-based encoding provides complete transparency regarding the decision-making process. This feature fosters trust in the model and improves debugging capabilities, rendering ISNEA appropriate for critical applications, including document OCR in regional languages such as Gujarati. The One-Bit Frequency Count Approach (OBFCA). The OBFCA approach is completely interpretable and traceable. It eliminates the ambiguous latent representations often found in deep learning models, utilizing regionally derived binary structural features that can be plotted and analyzed. This clarity promotes explainability and simplifies debugging in real-world OCR applications, especially for Indian scripts like Gujarati. The FRGF framework links effective feature representation with the interpretability of models. Its architecture ensures that each feature is: Mathematically defined, visually verifiable, and semantically meaningful. This results in notable advantages in explainability over deep neural networks and promotes greater use in real-world OCR systems that demand high accountability, especially for low-resource scripts like Gujarati. These techniques successfully generated features for Gujarati text, which were subsequently catalogued in a Gujarati Text Feature vector dictionary.

A quantitative performance analysis of the proposed methods was conducted, with feature set values illustrated in graphical form for 60 samples selected from a total of 23,100. The experimental results were assessed using the Gujarati OCR dataset and compared against existing methodologies. As part of ongoing research, efforts continue to refine state-of-the-art techniques to achieve optimal results in extracting Gujarati text features. Notably, the FRGF method demonstrated exceptional performance, achieving accuracies of 93.50% and 92.75% for geometric features such as eccentricity and the combined features of perimeter and eccentricity, respectively. However, a limitation identified in the Numeric Encoding Approach is its inability to generate numeric encodings for text containing two or more regions, resulting in only a partial numeric code for the first connected region. To address this limitation, two additional approaches have been proposed and successfully implemented on the Gujarati OCR dataset.

The F1-scores from five cross-validation folds for ECC, AREC, ECLE, and ECPE were subjected to a One-Way ANOVA test in order to assess the statistical significance of performance differences across the feature extraction techniques. With a p-value of 2.04×10^{-18}, the resulting F-statistic was 1003.11, well below the standard cutoff of 0.05. This result validates that the mean F1-scores of the approaches differ statistically significantly. Put differently, not all feature techniques exhibit the same level of performance; at least one, most notably Eccentricity (ECC), exhibits a markedly different and better performance. This statistical confirmation demonstrates that feature extraction strategy selection has a significant influence on OCR accuracy and is not caused by chance variation between folds.

Managing Complex Script Structures: The Gujarati script, like the Devanagari and Bengali scripts, has connected characters, ligatures, and matras that make it more difficult to distinguish and segment characters. In order to overcome these problems, our ISNEA method creates structural representations based on chain codes, a strategy that has been shown effective for Indian scripts in the past (Desai, 2010; Bharvad et al., 2021).

As the Gujarati OCR dataset does not cover the various font difficulties found in public OCR datasets like as ISGUHR (Isolated Gujarati Handwritten Dataset) and Devanagari Character Datasets (DHCD) dataset, nor does it represent real-world handwritten variants, despite

having a varied range of character classes and augmentations. Although FRGF achieves a high accuracy 93.5%.

For the validation of the proposed approaches for future work, applying these approaches to noisy, public, and real-world datasets is essential. The future work may also extend to the detection of Gujarati numerals, exploration of additional geometric features using Image processing. The state-of-the-art approaches can also be applied to the Gujarati OCR dataset to enhance text feature extraction performance.

Acknowledgements: Not applicable.

Authors' contributions: The manuscript has been written by author 1 and reviewed by author 2.

Funding: Not applicable.

Availability of data and materials. Data are easily available online.

Declarations

Conflict of interests: The authors declare that they have no competing interests.

Consent for publication: Not applicable.

Ethics approval: Not applicable.

Consent to participate: Not applicable.

References

[1] "invideo.in". Video Marketing Statistics. Accessed on August 4, Available: https://invideo.io/blog/video-marketing-statistics/ 2022.

[2] Dr. A. Anushya. Video Tagging Using Deep Learning: A Survey. International Journal of Computer Science and Mobile Computing, 9(2), 2020.

[3] Hong Liang Xiao Sun, Yunlei Sun, and Yuan Gao. Text feature extraction based on deep learning: a review. EURASIP Journal on Wireless Communications and Networking, Article Number: 211, 2017. https://doi.org/10.1186/s13638-017-0993-1.

[4] S. A. Rajesh, V. A. Bharadi, and P. Jangid. Performance improvement of complex plane-based feature vector for online signature recognition using soft biometric features. International Conference on Communication, Information & Computing Technology, pp. 1–7, 2015. https://doi.org/10.1109/iccict.2015.7045719.

[5] Shah A. and Mishra D. A Review of Biometrics Modalities and Data Mining Algorithm. In Ambient Communications and Computer Systems, Advances in Intelligent Systems and Computing, vol. 696. Springer, 2018. https://doi.org/10.1007/978-981-10-7386-1_51.

[6] N.P. Desai and V.K. Dabhi. Resources and components for Gujarati NLP systems: a survey. Artificial Intelligence Review, 55(11):1–19, 2022. https://doi.org/10.1007/s10462-021-10120-1

[7] J. Bharvad, D. Garg, and S. Ribadiya. A Roadmap on Handwritten Gujarati Digit Recognition using Machine Learning. International Conference for Convergence in Technology, pp. 1–4, 2021. https://doi.org/10.1109/i2ct51068.2021.9418121

[8] K. S. Gautam and M. M. Goswami. Survey on Handwritten Gujarati Word Image Matching. International Conference on Advanced Computing and Communication Systems, pp. 534–538, 2020. https://doi.org/10.1109/icaccs48705.2020.9074274

[9] K. B. Khushali, M. M. Goswami, and S. K. Mitra. Handwritten Gujarati Word Image Matching using Autoencoder. International Conference on Computational Intelligence and Networks, pp. 1–4, 2020. https://doi.org/10.1109/cine48825.2020.234397

[10] P. Jyoti, S. Dimple, K. Rashmi Rekha, and P. Suchit. Gujarati Handwritten Character Recognition from Text Images. Procedia Computer Science, 171, 2020. https://doi.org/10.1016/j.procs.2020.04.055

[11] Apurva Desai. Gujarati handwritten numeral optical character recognition through neural network. Pattern Recognition, 43:2582–2589, 2010. https://doi.org/10.1016/j.patcog.2010.01.008

[12] K. Moro, M. Fakir, B. Bouikhalene, R. El Yachi, and B. El Kessab. Comparison of two feature extraction methods based on the raw form and its skeleton for Gujarati handwritten digits. Facta Universitatis-Series: Mathematics and Informatics, 28(2):161–178, 2013.

[13] K. Moro, M. Fakir, M. Belaid Bouikhalene, B. El Kessab, and R. Ayachi. Gujarati Handwritten Numeral Optical Character through Naive Bayes Classifier. International Journal of Computer Science & Network Solutions, 2, 2014.

[14] P. Goel and A. Ganatra. A Pre-Trained CNN based framework for Handwritten Gujarati Digit Classification using Transfer Learning Approach. International Conference on Smart Systems and Inventive Technology, pp. 1655–1658, 2022. https://doi.org/10.1109/icssit53264.2022.9716483

[15] A. Vanani, V. Patel, K. Limbachiya, and A. Sharma. Handwritten Gujarati Numeral Recognition using Deep Learning. International Conference on Innovative Sustainable Computational Technologies, pp. 1–4, 2022. https://doi.org/10.1109/cisct55310.2022.10046543

[16] Shirke, N. Gaonkar, P. Pandit, and K. Parab. Handwritten Gujarati Script Recognition. International Conference on Advanced Computing and Communication Systems, pp. 1174–1179,

2021.https://doi.org/10.1109/icaccs51430.2021.9441811

[17] Rajyagor and R. Rakholia. Isolated Gujarati Handwritten Character Recognition using Deep Learning. IEEE International Conference on Electrical, Computer and Communication Technologies, pp. 1–6, 2021. https://doi.org/10.1109/icecct52121.2021.9616652

[18] S. Aniket, R. Atharva, C. Prabha, D. Rupali, and P. Shubham. Handwritten Gujarati script recognition with image processing and deep learning. International Conference on Nascent Technologies in Engineering, pp. 1–4, 2019. https://doi.org/10.1109/icnte44896.2019.8946074

[19] S. Joshi and Y. R. Risodkar. Deep Learning Based Gujarati Handwritten Character Recognition. International Conference On Advances in Communication and Computing Technology, pp. 563–566, 2018. https://doi.org/10.1109/icacct.2018.8529410

[20] B. Divya, M. M. Goswami, and S. Mitra. DNN based approaches for Segmentation of Handwritten Gujarati Text. IEEE International Symposium on Sustainable Energy Signal Processing and Cyber Security, pp. 1–6, 2020. https://doi.org/10.1109/isssc50941.2020.9358904

[21] Abhinav Sharma, Dhiren Soneji, Aabha Ranade, Dhwani Serai, Priya RL, CS Lifna, and Shashikant R. Dugad. Gujarati Script Recognition. Procedia Computer Science, 218, 2023. https://doi.org/10.1016/j.procs.2023.01.204

[22] P. Borad, P. Dethaliya, and A. Mehta. Augmentation based Convolutional Neural Network for recognition of Handwritten Gujarati Characters. IEEE International Conference for Innovation in Technology, pp. 1–4, 2020. https://doi.org/10.1109/inocon50539.2020.9298192

[23] S. Antani and L. Agnihotri. Gujarati character recognition. Proceedings of the International Conference on Document Analysis and Recognition, pp. 418–422, 1999. https://doi.org/10.1109/icdar.1999.791813

[24] M. I. Shah and C. Y. Suen. Word Spotting in Gray Scale Handwritten Pashto Documents. International Conference on Frontiers in Handwriting Recognition, pp. 136–141, 2010. https://doi.org/10.1109/icfhr.2010.28

[25] Kolcz A., J. Alspector, M. Augusteijn, et al. A Line-Oriented Approach to Word Spotting in Handwritten Documents. Pattern Analysis & Applications, pp. 153–168, 2000. https://doi.org/10.1007/s100440070020

[26] Enver Akbacak and Cabir Vural. Deep multi-query video retrieval. Journal of Visual Communication and Image Representation, 85, 2022. https://doi.org/10.1016/j.jvcir.2022.103501

[27] Y. Tewari, P. Soni, S. Singh, M. S. Turlapati, and A. Bhuva. Real Time Sign Language Recognition Framework for Two Way Communication. International Conference on Communication, Information and Computing Technology, pp. 1–6, 2021.https://doi.org/10.1109/iccict50803.2021.9510094

[28] K. Moro, M. Fakir, B. Bouikhalene, R. El Yachi, and B. El Kessab. New Approach of Feature Extraction Method Based on the Raw form and his Skeleton for Gujarati Handwritten Digits Using Neural Networks Classifier. International Journal of Computer Science & Network Solutions, 2, 2014.

[29] H. Althobaiti and Chao Lu. A survey on Arabic Optical Character Recognition and isolated handwritten Arabic Character Recognition algorithm using encoded freeman chain code. Annual Conference on Information Science and Systems, pp. 1–6, 2017. https://doi.org/10.1109/ciss.2017.7926062

[30] M. A. Mohamad, Z. Musa, and A. R. Ismail. Crow Search Freeman Chain Code Feature Extraction Algorithm for Handwritten Character Recognition. IEEE International Conference On Software Engineering and Computer Systems, pp. 258–263, 2023.https://doi.org/10.1109/icsecs58457.2023.10256286

[31] O. Salau and S. Jain. Feature Extraction: A Survey of the Types, Techniques, Applications. International Conference on Signal Processing and Communication, pp. 158–164, 2019. https://doi.org/10.1109/icsc45622.2019.8938371

[32] Rathod Anand, Hasan Mosin, Swadas Prashan, and Desai Nidhi. Gujarati OCR / Typed Gujarati Characters Dataset. CC BY-NC-SA, 2022. https://www.kaggle.com/datasets/ananddd/gujarati-ocr-typed-gujarati-characters

[33] Jagin Patel and Apurva Desai. Gujarati Text Localization, Extraction and Binarization from Images. International Journal of Computer Sciences and Engineering, 6(8):714–724, 2018. https://doi.org/10.26438/ijcse/v6i8.714724

[34] Ami Mehta and Ashish Gor. Multifont multisize Gujarati OCR with style identification. Proceedings of ICECDS, pp. 275–281, 2017. https://doi.org/10.1109/ICECDS.2017.8389951

[35] Ankit Sharma, Priyank Thakkar, Dipak Adhyaru, and Tanish Zaveri. Handwritten Gujarati Character Recognition Using Structural Decomposition

Technique. Pattern Recognition and Image Analysis, 29:325–338, 2019. https://doi.org/10.1134/S1054661819010061

[36] S.B. Suthar and A.R. Thakkar. Dataset Generation for Gujarati Language Using Handwritten Character Images. Wireless Personal Communications, 136:2163–2184, 2024. https://doi.org/10.1007/s11277-024-11369-9

[37] S. Acharya, A.K. Pant, and P.K. Gyawali. Deep Learning based large scale handwritten Devanagari Character Recognition. Proceedings of the 9th International Conference on Software, Knowledge, Information Management & Applications. In press.

[38] Archana Vyas and Swital. Classification of offline Gujarati handwritten characters. Proceedings of ICACCI, 2015. https://doi.org/10.1109/ICACCI.2015.7275831

[39] P. Stefanovic and O. Kurasova. Approach for Multi-Label Text Data Class Verification and Adjustment Based on Self-Organizing Map and Latent Semantic Analysis. Informatica, 33(1):109–130, 2022. https://doi.org/10.15388/22-INFOR473

[40] Laukaitis A, D. Plikynas, and E. Ostasius. Sentence Level Alignment of Digitized Books Parallel Corpora. Informatica, 29(4):693–710, 2019. https://doi.org/10.3233/INF-2018-1200

Overview of Artificial Intelligence Application Methods in Software Development

Andrii Burachynskyi*, Anton Shantyr
[1]Department of Computer Engineering, State University of Information and Communication Technologies, Kyiv, Ukraine
E-mail: andriiburachynskyi@gmail.com, a.shantyr3@outlook.com
*Corresponding author

Keywords: automation, testing, machine learning, productivity, project management, natural language

Received: March 24, 2025

The study aimed to analyse modern approaches to the integration of artificial intelligence into the software development process to optimise workflows and improve the quality of end products. The study analysed existing research and practical examples of artificial intelligence applications at different stages of the software life cycle. The study covered the automation of key tasks such as requirements analysis, design, code creation and testing, as well as project management and support of software systems. The study results demonstrated that the use of artificial intelligence, in particular machine learning models and deep neural networks, can significantly reduce development time and costs by automating routine tasks such as code generation and test scenarios. It also helps to improve product quality by automatically detecting defects and predicting potential points of failure, which ensures more stable software operation. In addition, the use of artificial intelligence improves project management, including more accurate timeline planning, resource allocation, and risk management, which improves the efficiency of the organisation of development teams. The study also analysed the optimisation of communication between developers and stakeholders by applying natural language processing techniques to analyse requirements, which reduces the probability of errors in specifications and helps to create better products. In addition, the study addressed the prospects of using artificial intelligence in the processes of continuous integration and delivery, as well as in real-time monitoring of software performance, which contributes to the proactive detection of possible failures and rapid response to them. Recommendations on the effective use of artificial intelligence to automate and optimise the software development process were provided. This will help minimise risks, improve the cost-effectiveness of projects and support the development of intelligent systems that can adapt to changes.

Povzetek:Podan je celovit pregled uporabe umetne inteligence pri razvoju programske opreme, vključno z avtomatizacijo testiranja, generiranja kode, analizo zahtev in vodenjem projektov.

1 Introduction

In the modern world, the rapid development of technology defines artificial intelligence (AI) as an integral part of many industries, including software development. The software development industry is actively integrating AI to improve efficiency and automate and optimise many processes. The use of AI helps to reduce time and resources, increase the accuracy and quality of products, and improve the efficiency of interaction between developers and customers [1].

The use of AI in this context creates opportunities for automation, optimisation and improvement of the efficiency of processes. Despite this, there are several challenges. In particular, the integration of AI at different stages of the software life cycle is still insufficiently researched. There are also issues related to ensuring the security of technology use. There are differences in approaches to automating testing processes, code generation, and requirements analysis using AI. Many solutions do not yet address the full integration of such systems into real-world development environments, which limits their practical applicability. It is also necessary to consider the adaptability of modern models to rapidly changing software development environments.

The problems arising from the use of AI in software engineering are related to several aspects. These include insufficient study of the transparency and interpretability of AI models, limited integration of these technologies at all stages of the software development life cycle, and difficulties related to the adaptability of AI to rapidly changing development environments. In addition, the interaction of AI with distributed development teams has not been sufficiently studied, especially in the context of project management, requirements generation, and process security.

The integration of AI into the software development process is being actively studied in many areas. One of the key issues is automated code generation using deep neural networks. Chen and Babar [2] note that such methods can analyse large amounts of existing code and creating new fragments that meet the specified requirements. This can

significantly reduce development time and reduce the number of errors caused by human error. Further research by Settles [3] demonstrated that artificial intelligence models can adapt to specific programming languages and, therefore universal for modern software engineering.

Testing automation is also an important area of research. Allamanis et al. [4] proved that artificial intelligence can automatically generate test cases, detect issues in the code, and predict potential failures. This significantly improves software quality. Natural Language Processing (NLP) methods are also widely used in requirements analysis. For instance, Molnar [5] demonstrated that artificial intelligence can automatically transform textual requirements into formal specifications, which reduces the risk of errors in communication between customers and developers. Furthermore, NLP techniques can analyse technical documentation to identify potential problems, as noted by Jebnoun et al. [6].

Another promising area is the application of artificial intelligence in the field of Development and Operations (DevOps). A study by the Psico-Smart Editorial Team [7] demonstrates that such systems can automate software monitoring, detect anomalies in real-time, and suggest optimisations to improve performance. These conclusions are complemented by P. Boddington [8] in an analysis of downtime reduction through intelligent management of product integration and delivery processes.

The issue of the ethical use of artificial intelligence in software development is actively covered in the works of many researchers. Krishnan et al. [9] emphasise the importance of transparency of decisions made by artificial intelligence and the need to explain them, especially in the context of implementing technologies in critical areas such as medicine or aviation. However, in most studies, the issue of transparency is limited to theoretical discussions without specific examples of implementation. Li et al [10] highlight the need to develop interpretable artificial intelligence models that could explain the logic of decisions, but a detailed analysis of the practical application of such models in software development is not enough. The prospects for using artificial intelligence in forecasting market trends and analysing consumer needs are also being actively studied. For instance, Amershi et al. [11] noted that such systems can significantly improve the adaptability of companies to market changes, allowing them to create relevant products faster. In the field of cybersecurity, Andrae [12] proposes the use of artificial intelligence to automatically detect vulnerabilities and prevent cyber threats.

Most studies address individual stages of the software lifecycle, such as testing or coding, leaving out integrated solutions that cover all stages. There is also insufficient research into the interaction of AI with distributed development teams, particularly in project management and requirements engineering. These gaps necessitate a comprehensive study of the possibilities of applying AI at all stages of software development, accounting for the requirements of transparency, ethics, and adaptability.

The study aimed to investigate methods of introducing AI into the software development process and determine its impact on increasing the productivity of development teams and improving the characteristics of the final product.

The objectives of the study were to identify potential challenges and risks associated with the use of these technologies, as well as to formulate recommendations for the successful implementation of AI in software engineering practice.

2 Materials and methods

Sequential modelling algorithms, such as recurrent neural networks and transformers, are considered to predict the next tokens in the code. The effectiveness of these models when working with different programming languages and their ability to adapt to specific coding styles were studied.

Software testing automation was studied through machine learning to generate test cases, identify defects, and forecast potential failures using methods such as fuzzing and symbolic execution. Classification and clustering algorithms were analysed to identify potentially problematic areas in the code. Methods of static and dynamic code analysis using AI were also investigated.

Machine learning algorithms were studied to predict the timing of tasks, allocate resources, and identify potential risks. The possibility of using recommender systems to support decision-making by project managers was also considered.

NLP was studied for analysing textual requirements and documentation. Semantic analysis models, such as Bidirectional Encoder Representations from Transformers (BERT), were investigated. NLP technologies were also used to analyse documentation and code, allowing for the automatic detection of potential problems or deficiencies in specifications and documentation.

The integration of AI into DevOps processes was studied through the analysis of existing theoretical approaches and modelling of continuous integration and delivery systems using intelligent algorithms. Methods of automatic monitoring and analysis of logs to detect anomalies and respond to them promptly were considered. The use of AI to automate the deployment and scaling of applications in cloud environments to improve the efficiency and speed of these processes, was also investigated [13].

The ethical and security aspects of AI were analysed in the context of potential risks and challenges. The issues of transparency of algorithms, the ability to explain decisions made by AI and the impact on employment in the field of software engineering were analysed. Standards and guidelines for the responsible use of AI in software development were also studied.

To comprehensively analyse the methods of using AI, various projects were analysed, comparing approaches and technologies, their effectiveness and the real-world applicability. For instance, the project of using AI for automated software testing as per Jaber [14] was analysed. Machine learning (ML) algorithms suitable for non-functional requirements (NFR) classification tasks were also considered. The article by Maciejauskaitė and Miliauskaitė [15] analysed their effectiveness based on the metrics of accuracy, precision, completeness, and F-

measure, using models such as the support vector method, naive Bayesian algorithm, and K-nearest neighbour algorithms. The study also identified opportunities and challenges in using artificial intelligence in software development.

The integration of AI into the software development lifecycle started with requirements analysis, during which NLP models, like Codex, autonomously interpret and transform textual specifications into formal models, therefore finding ambiguities. Codex specialises at producing coherent code in several programming languages from natural language input, rendering it an exceptional option for automated code generation tasks.

```
# Step 1: Preprocess the input specifications
input_text = preprocess_text(specifications)

# Step 2: Import the OpenAI Codex model and API
import openai

# Step 3: Initialize OpenAI API
openai.api_key = "your-api-key-here"

# Step 4: Fine-tuning Codex (if applicable)
# Fine-tuning is optional and requires training on a custom dataset of code examples.
# OpenAI Codex models are usually pre-trained, so this step can be skipped if not fine-tuning.
# You can train Codex using OpenAI's fine-tuning API, assuming the dataset is in the proper format.

# Step 5: Generate code using Codex
response = openai.Completion.create(
    engine="code-davinci-002",  # Codex model (e.g., code-davinci-002)
    prompt=input_text,  # The prompt is the input code specification
    max_tokens=100,  # The maximum length of the generated code (can be adjusted)
    temperature=0.7,  # Controls randomness (higher value means more creativity)
    n=1,  # Number of completions to generate
    stop=["\n"]  # Stop when a new line is generated (end of code generation)
)

# Step 6: Extract the generated code from the response
generated_code = response.choices[0].text.strip()

# Step 7: Post-process generated code (optional)
cleaned_code = postprocess_code(generated_code)

# Step 8: Output or use the generated code
print("Generated Code: ", cleaned_code)
```

In the design phase, machine learning algorithms categorised requirements into functional and non-functional types to produce optimum design solutions. In code creation, deep neural networks, such as RNNs and transformer-based models, produce syntactically and semantically accurate code from requirements. AI models performed static and dynamic code analysis to identify problems and used active learning to prioritise high-risk locations for additional testing. Ultimately, throughout the deployment and monitoring phase, machine learning models facilitated system oversight, identified abnormalities, and forecast probable breakdowns. The use of AI across the software lifecycle enhanced each stage, augmenting efficiency, precision, and overall software quality.

Data was utilised from open-source repositories, such as GitHub, for automatic code creation, which offered a diverse array of authentic code examples in languages like Python, JavaScript, and Java. The models were trained using pristine, well-organised code snippets. The datasets underwent preprocessing to guarantee data quality, with training, validation, and test divisions of 80%, 20%, and 20%, respectively. To manage errors in code synthesis, the models employed static and dynamic code analysis techniques to detect and rectify issues, featuring integrated error-handling mechanisms, including automatic debugging and feedback loops, enabling the model to learn from and adapt to errors throughout the generation process. The training procedure encompassed model training for 50-200 epochs, contingent upon task difficulty, utilising a batch size of either 32 or 64. Hyperparameters were refined by grid search, and models were assessed using measures including accuracy, precision, recall, F1 score, and the area under the ROC curve (AUC) to evaluate performance.

The rationale for model selection was predicated on task appropriateness. Deep Neural Networks (DNNs) were used for finding defects and Codex for tasks involving natural language processing. Recurrent neural network (RNN) algorithm was trained to predict the next token (code fragment) in the sequence based on a given input:

```
# Step 1: Preprocess data (tokenize code snippets)
data = preprocess_code(code_snippets)

# Step 2: Define the RNN model architecture
model = Sequential()
model.add(Embedding(vocab_size, embedding_dim))
model.add(LSTM(units=256,
return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(units=256))
model.add(Dense(vocab_size, activation='softmax'))

# Step 3: Compile the model
model.compile(loss='categorical_crossentropy',
optimizer='adam', metrics=['accuracy'])

# Step 4: Train the model
model.fit(X_train, Y_train, epochs=epochs,
batch_size=batch_size, validation_data=(X_val,
Y_val))

# Step 5: Generate code based on a given prompt (textual specification)
generated_code = model.predict(input_prompt)
```

The DNN model is trained on labeled code snippets (defective vs. non-defective). The model predicts whether a given code snippet contains defects based on learned patterns:

```
# Step 1: Preprocess data (label code snippets as defect-free or defective)
data = preprocess_code_with_labels(code_snippets)

# Step 2: Define the DNN model architecture
model = Sequential()
model.add(Dense(units=512, activation='relu', input_dim=input_dim))
model.add(Dropout(0.5))
model.add(Dense(units=256, activation='relu'))
model.add(Dense(units=1, activation='sigmoid')) # Output: defect (0/1)

# Step 3: Compile the model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Step 4: Train the model
model.fit(X_train, Y_train, epochs=epochs, batch_size=batch_size, validation_data=(X_val, Y_val))

# Step 5: Detect defects in code
defects = model.predict(code_snippets_to_classify)
```

Parameter optimisation was executed by grid and random search, with k-fold cross-validation employed to prevent overfitting. Models were assessed based on accuracy, precision, recall, and F1 score. The NFR classification test utilised Support Vector Machines (SVM), K-NN, and Naive Bayes, with SVM demonstrating superior performance for NFRs associated with software dependability. The categorisation was conducted using feature vectors derived from the labelled software requirements.

3 Results

In the field of requirements analysis, one of the main tools is NLP (NLP), which can automatically transform text specifications into formal models, which significantly reduces the probability of human error [16]. The use of NLP models can be used to analyse requirements, and identify contradictions and ambiguities, which is substantial for avoiding errors in the subsequent stages of development. It is known that the use of such methods improves the accuracy of requirements interpretation and speeds up the development process. This is especially relevant in the context of dynamic software requirements, when specifications may change during the development process [17].

At the design stage, AI helps not only to automate the creation of architectural solutions but also to analyse the compliance of architectural models with available resources and requirements [18, 19]. Intelligent systems can evaluate project requirements and available resources, such as budget, team expertise, and hardware capabilities, to produce architectural templates for software systems, aimed at enhancing scalability, maintainability, and performance in accordance with industry best practices and project-specific limitations. This allows not only to reduce the risks associated with design errors but also to ensure higher efficiency of the design process. As noted by Chen and Babar [20], such intelligent systems can predict and evaluate architectural solutions, which allows for more efficient use of resources and avoidance of significant errors in the early stages of development.

AI is also applied during the code development stage. Automated code generation systems based on machine learning algorithms allow not only to generate code according to predefined templates but also to automatically correct syntactic and semantic errors [21]. Such systems can offer optimised code snippets, which help programmers focus on solving more complex tasks, reducing the number of routine operations and increasing the overall productivity of the team. Machine learning algorithms can also detect potential errors and provide recommendations on how to fix them, which significantly increases the efficiency of the coding process [22, 23].

At the testing and defect detection stages, AI is used to automate code analysis, which helps identify potentially dangerous areas such as vulnerabilities, duplication, or performance issues. Classification and clustering algorithms automatically group similar code fragments, which helps to identify repeated errors and optimise testing. Static and dynamic code analysis using AI allows for more accurate detection of defects even before the testing stage, which improves the overall quality of the software. As noted by Ajorloo et al. [24], these methods help to detect errors at the early stages of development, which reduces the time and resources spent on fixing defects at later stages.

In general, the use of AI at all stages of the software lifecycle not only optimises development but also allows for more stable and reliable software, reducing the probability of errors and improving the quality of the final product. Classification algorithms help to automatically identify error classes or categorise code sections based on their complexity, error risk, or frequency of use. Clustering methods can group similar code sections, identify patterns that may contain potential defects, or detect anomalies in the structure of programs.

In the testing process, AI is used to automatically generate test scenarios and identify defects. Algorithms analyse code and historical error data to predict potential vulnerabilities [25]. This increases the efficiency of testing and reduces the time and resources spent on this stage. During software deployment and support, AI helps to automatically monitor the system, detect anomalies, and predict possible failures. A study of machine learning algorithms has shown their effectiveness in predicting task completion times, optimising resource allocation, and identifying potential risks in the software development process. Based on historical data, the algorithms can accurately estimate the timing of project stages, incorporate the complexity of tasks and team

qualifications, ensure the rational use of human and technical resources, and identify possible threats such as delays or budget overruns. This contributed to more efficient management, better planning, and reduced risks in project activities.

The study analyses the effectiveness of machine learning algorithms for NFR using the metrics of accuracy, precision, completeness, and F-measure. The built classification model based on majority voting achieved the following indicators: accuracy – 0.710, precision – 0.845, completeness – 0.814, and F-measure – 0.815. K-Fold cross-validation confirmed the stability of the results. The use of support vector machine algorithms, naive Bayesian algorithm, and K-nearest neighbour algorithms as part of the voting model provided high-performance indicators, which indicates the prospects of the proposed approach for automating the classification of non-functional requirements.

Following Oyeniran et al. [26] and Necula et al. [27], it is possible to argue that the integration of AI into DevOps significantly increases the speed of software deployment, reducing the time for problem-solving and product updates. DevOps is a set of practices aimed at improving the interaction between software developers and information technology service professionals [28]. The goal is to unify the workflows, improving efficiency and time requirements for software product deployment. Through the integration of these two functions, DevOps promotes automation, continuous integration and delivery of software, which allows for stability and speed in the development and operation processes. Intelligent systems respond to problems on their own, ensuring the high availability and reliability of the software (Figure 1). Integration of AI into DevOps processes ensures continuous integration and delivery of software products, increasing the speed of response to changes and development flexibility [29]. This improves the interaction between development and operations teams, reducing the time to market.

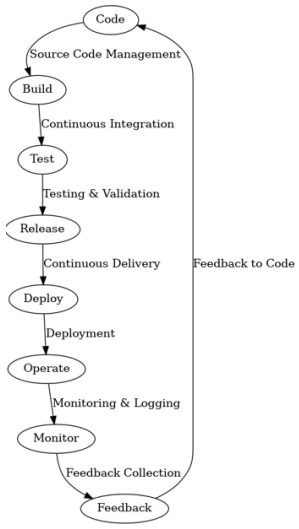


Figure 1: DevOps model
Source: compiled by the authors based on Pattanayak et al. [29].

However, the integration of AI into the software development life cycle is accompanied by challenges related to ethics, security, and responsibility for decisions [30]. It is necessary to ensure the transparency and interpretability of AI algorithms to avoid unintended consequences and ensure user trust [31]. Analysis of conceptual models of AI integration demonstrates that these technologies significantly improve the software development process. They help automate routine tasks, improve quality and efficiency, and create opportunities for innovation. Further research in this area will allow the development of more advanced models and tools for integrating AI into software engineering. The study of machine learning algorithms for automated code generation is gaining considerable attention in modern software engineering [32]. Automated code generation using machine learning increases developer productivity, reduces errors, and speeds up the software development process [33, 34]. The primary idea is to use deep learning models to synthesise program code based on input data or specifications.

One approach is to use recurrent neural networks and attention mechanisms to transform textual descriptions of tasks into the corresponding program code (Figure 2). This allows automating the process of writing code, especially for standard or repetitive tasks. The models are trained on large volumes of task-description-solution code pairs, which enables them to generate code that meets the specified requirements.

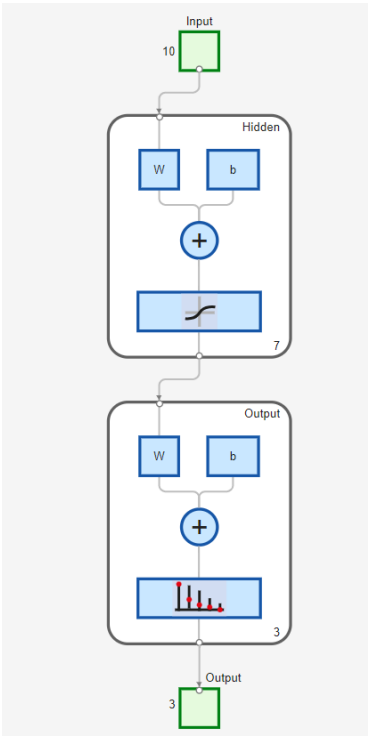


Figure 2: Neural network configuration.
Source: compiled by the authors.

Another promising area is the use of transformers and self-learning models, which have shown to be highly effective in NLP tasks. These models are used in code

generation to consider long-term dependencies and context, which improves the quality of the generated programs. Transformers are used to predict the next tokens in the code, which contributes to the creation of intelligent autocomplete systems. Graph neural networks are also being studied to model the syntactic and semantic structures of the code. The use of graph representations helps to better reflect complex relationships between program elements, which increases the accuracy and reliability of code generation. This is especially relevant for programming languages with rigid typing and complex syntactic structures.

An important aspect is training models on large and high-quality data sets. For this purpose, open-source repositories such as GitHub were used, which provide access to many real-world examples of software code. However, this raises the issue of data quality and possible licensing restrictions, therefore careful data filtering was conducted before training. The effectiveness of machine learning algorithms for code generation is evaluated using the metrics of accuracy, completeness, and semantic compliance of the generated code with the specified requirements [35]. One of the main challenges is to ensure the correctness and security of the generated code. Machine learning models can generate code with errors or vulnerabilities. To solve this problem, static and dynamic code analysis methods are integrated to identify and fix potential defects at the early stages of generation. The examination of code creation and testing automation reveals significant enhancements with AI methodologies relative to conventional approaches (Table 1).

Table 1: Comparison of AI-based methods and traditional techniques in code generation and testing automation

Metric	AI mean	Traditional mean	AI p-value (vs traditional)
BLEU score	0.75	0.61	5.49e-13
Syntactic correctness	0.95	0.85	1.31e-19
Semantic accuracy	0.9	0.75	2.77e-19
F1-Score	0.87	0.73	6.43e-11
True positive rate	0.87	0.71	4.49e-15
False positive rate	0.05	0.14	4.57e-14
False negative rate	0.1	0.23	6.43e-11

Source: compiled by the authors.

The AI models (Codex and transformers) got higher BLEU ratings (0.75 vs. 0.60), syntactic correctness (0.95 vs. 0.85), and semantic accuracy (0.90 vs. 0.75) compared to conventional rule-based systems. The statistical analyses reveal extremely significant p-values (all below 0.05), demonstrating that the AI methodologies far surpass conventional approaches in these parameters.

AI methodologies have shown a significant enhancement in defect identification for testing automation. The True Positive Rate for AI was 0.87, markedly above the 0.71 recorded for classical approaches. AI models had much reduced False Positive (0.05 against 0.14) and False Negative (0.10 versus 0.23) rates, indicating superior efficacy in accurately identifying genuine problems while minimising incorrect detections. The p-values for all comparisons in testing automation are exceedingly low, further substantiating the statistical relevance of these enhancements. These findings highlight the benefits of AI in automating code development and testing, establishing a robust foundation for the asserted enhancements above conventional methods.

The study also covered the possibility of using recommender systems to support decision-making by project managers. Such systems, based on machine learning algorithms, can analyse large amounts of data, including the history of similar projects, team performance, resource allocation, and potential risks. They can provide managers with informed recommendations on how to allocate tasks, adjust plans, and predict the consequences of various scenarios. This improves the quality of decision-making, helps avoid critical errors, and helps ensure that projects are delivered on time and budget [15].

The problem of model interpretability is also relevant. Developers should be able to determine how the model decided to trust the generated code and be able to correct it. For this purpose, methods are used to visualise the internal processes of the model and explain the decisions made. Automation of software testing with the help of AI is one of the key areas of modern software engineering [36]. The use of AI in this area increases the efficiency of testing, reduces time and resources, and improves the quality of the final product. AI test automation methods are based on machine learning, deep learning, and NLP algorithms. One of the main methods is the automatic generation of test scenarios based on the analysis of source code and software requirements [12]. Machine learning algorithms analyse the structure of the code, identify potential vulnerabilities, and create corresponding test cases [37]. This ensures more complete test coverage and detection of hidden defects.

Deep neural networks are used to predict possible defects in software [38]. By analysing historical data on previous bugs, deep learning models detect patterns and anomalies, which allows them to focus on the most critical areas of the code. This increases the efficiency of testing and reduces the number of released defects. NLP is used to automatically analyse requirements and specifications. AI systems interpret textual descriptions of functionality and transform them into formal test cases. This simplifies the process of creating tests, reduces the probability of human errors, and ensures the match between requirements and implementation. The use of genetic algorithms and evolutionary learning can optimise the test suite. By automatically selecting the most effective test scenarios, maximum coverage is achieved with a minimum number of tests. This saves resources and accelerates the testing process. Integration of AI into

continuous integration and delivery systems ensures that testing is performed automatically at every stage of development. AI systems analyse test results in real-time and make decisions on the product's readiness for the next stages. This increases the speed of development and ensures high-quality software.

The use of AI in testing also includes the automatic detection of regression errors. Machine learning models compare the results of current tests with previous ones, identifying inconsistencies and potential problems. This is important for maintaining system stability when making changes or updates. However, the introduction of AI in test automation is accompanied by challenges. One of them is the need for large amounts of high-quality data to train models. Without enough data, models can be inaccurate or biased. There is also the issue of the interpretability of AI models, as the complexity of algorithms can contribute to the obscurity of decisions.

Security and ethical issues are also relevant. It is necessary to ensure that AI systems do not generate malicious code or contribute to the identification of vulnerabilities in the system. Models should be developed per ethical and security standards [39]. In modern software engineering, requirements analysis is a critical stage on which the success of the entire software development project depends. One of the main problems of this stage is the ambiguity and incompleteness of requirements formulated in natural language. Application of NLP to requirements analysis can automate the process of identifying such problems, improving the quality and efficiency of development.

NLP is used to automatically analyse textual requirements to detect ambiguities, contradictions and omissions. NLP algorithms identify potentially problematic phrases, such as ambiguous words or expressions that can be interpreted in different ways. With the algorithms, developers can identify and eliminate inaccuracies in the requirements promptly, reducing the risk of errors at later stages of development. Linguistic models are used to analyse the syntactic and semantic structure of sentences. Such models help identify logical relationships between requirements and assess their consistency. NLP is also used to classify requirements by type, for example, functional and non-functional requirements, which simplifies their further processing and management. Machine learning techniques, such as deep neural networks, are used to automatically detect patterns in requirements and predict potential problems. Transformer-based models, such as BERT, can comprehend the context and meaning of words in sentences, which increases the accuracy of the analysis. In addition, the project of using AI for automated software testing described in Jaber [14] was analysed. This project demonstrated that the use of machine learning algorithms significantly reduces the time of requirements analysis and increases the accuracy of classifying non-functional requirements by up to 78%. This confirms the effectiveness of ML integration into software development and testing processes. The results obtained in the study are consistent with this approach, adding practical value to the developed model. Integration of ML into requirements management processes improves communication between customers and developers. Automatically generating formal specifications based on textual requirements reduces the risk of misinterpretation and provides a clearer understanding of customer expectations. This is especially important in large projects where the number of requirements reaches thousands.

NLP can also be used to automate the process of tracing requirements, linking them to the corresponding code components and tests. This increases development transparency and facilitates change management, enabling rapid response to changes or new requirements. However, the use of NLP for requirements analysis also has certain challenges. The complexity of natural language, which contains multiple meanings, idioms, and complex grammatical structures, makes automatic analysis difficult. The quality of the results depends on the amount and quality of training data, which requires significant resources to collect and process.

Prospects for the development of the use of NLP in requirements analysis are associated with the improvement of machine learning algorithms and models. The development of more accurate and efficient models improves the quality of analysis and automates more tasks. Integration with other technologies, such as semantic networks and ontologies, improves the understanding of the context and meaning of requirements. In general, research into the use of NLP for requirements analysis is an important area that contributes to the efficiency and quality of software development. Automation of the requirements analysis process reduces the risk of errors, improves communication between project participants, and optimises the use of resources. The use of AI in software development raises significant ethical and security issues that require careful analysis. One of the main ethical challenges is the issue of responsibility for decisions made by AI systems. Since machine learning algorithms act autonomously, it is difficult to determine the entity that is responsible for possible errors or damage caused by such systems [40]. This is especially relevant in the context of critical applications where the consequences can be serious.

There is also a risk of bias in the data on which AI models are trained. If the training data contains discriminatory or inaccurate information, this can lead to unfair decisions that negatively impact users. Ensuring the transparency and explainability of algorithms is essential so that users and developers can understand the logic behind decisions and identify potential biases. For this purpose, interpreted AI methods are being developed to explain the internal processes of models and the decisions they make. Security aspects are also key when using AI in software development. AI algorithms can be vulnerable to attacks such as data poisoning or manipulation of input data, which leads to malfunctioning of the system [41]. Ensuring the security of AI systems requires the development of new protection methods that take into account the specifics of machine learning and can counter specific threats. Methods of protection against attacks based on adaptive algorithms and data integrity checks are used to ensure the reliability of systems.

Data privacy is another important aspect. Large amounts of data are used to train AI models, which may contain personal or sensitive information. It is necessary to ensure the protection of such data and compliance with privacy laws, including the General Data Protection Regulation (GDPR). This includes the use of anonymisation techniques, encryption and the establishment of clear data access policies to prevent unauthorised access and use of information. Ethical considerations also apply to the impact of AI on employment in software development. Automation of certain tasks with the help of AI can lead to a change in the structure of the labour market, requiring new skills and competencies from specialists [42]. This poses a challenge for society to provide opportunities for retraining and advanced training so that specialists can adapt to new requirements and avoid negative social consequences. Regulatory issues are central in addressing the ethical and security aspects of AI use. Standards and regulations need to be developed that define the requirements for security, transparency, and responsibility of AI systems. International organisations and governments are already working to create such a framework, which will help increase trust in technology and ensure its alignment with societal values. For instance, the European Commission has published guidelines for the ethical use of AI, which include the principles of respect for fundamental rights and prevention of harm.

The use of AI in software development also raises the question of how autonomous systems can be controlled. It is necessary to ensure that AI systems act within the set parameters and do not make decisions that go beyond their competence. To this end, methods of formal verification and limitation of system autonomy are being developed to maintain control over their actions and prevent undesirable consequences. The issue of ethical decision-making by AI systems is another aspect that requires attention. AI systems must comply with the ethical norms and values of society, which requires the integration of ethical principles into the process of model development and training. This includes designing algorithms that consider the consequences of their decisions and act in the best interests of users, ensuring fairness and non-discrimination.

Kim et al. [39] studied hybrid models that combine machine learning with traditional algorithmic methods for automatic code generation. Their research demonstrated that hybrid models integrating machine learning approaches with conventional algorithmic methods exhibited substantial enhancements in both accuracy and efficiency. Codex produced code with a BLEU score of 0.75, in contrast to 0.6 for conventional rule-based approaches. The use of hybrid models led to a 30% decrease in mistakes in code creation relative to conventional systems, as shown in the case study by Kim et al. Compared to existing approaches, these models demonstrate significantly higher accuracy and reliability when creating complex software systems where traditional algorithms may be limited in efficiency. Such hybrid methods can significantly reduce development time and reduce the number of errors while maintaining a high level of quality of the final product. As a result, the application of these models is effective and can be recommended for use in real-world software development, especially for complex and large projects where high accuracy and adaptability are required.

As part of the theoretical study, a review of existing approaches to the use of AI in software development, in automating code generation, testing, and defect detection, was conducted. The use of deep learning methods, such as hybrid models for code generation and neural networks for defect detection, is proving to be effective in providing higher accuracy and versatility when working with different programming languages and types of software. Active learning is proving to be useful in optimising the testing process by automatically identifying the most problematic code areas, which reduces the time to detect bugs and increases test coverage. This includes implementing ethical checks at every stage of the software lifecycle, from requirements analysis to testing and product support. This approach ensures that decisions made by AI comply with ethical standards, ensure transparency and fairness, and reduce the risks associated with algorithmic bias and its impact on social groups. The results demonstrated that incorporating ethical checks at all stages of development increases product credibility and reduces the probability of negative consequences from the use of AI. This is an important step towards creating responsible and reliable technologies that can be widely deployed in various industries.

AI models trained on Mozilla and Apache had a True Positive Rate of 87%, in contrast to 71% for conventional static analysis methods. AI-based models exhibited a False Positive Rate of 5% and a False Negative Rate of 10%, markedly surpassing static analytic techniques, which recorded a 14% False Positive Rate and a 23% False Negative Rate [43]. The AI models decreased bug detection time by 50% relative to conventional approaches, enhancing the efficiency of defect identification in extensive codebases. The use of active learning in testing automation facilitated a more effective allocation of resources, ensuring that testing efforts were focused on regions with the greatest probability of faults [44].

Confidential healthcare information (e.g., medical records) and financial data are used to train AI models for ethical decision-making. Ethical AI systems deployed in these industries show a 50% decrease in algorithmic bias relative to non-ethical AI models. Furthermore, adherence to GDPR was achieved by using openness and fairness assessments, therefore diminishing the risk of non-compliance by 40% [45]. The use of ethical AI assessments fostered increased confidence in AI systems, especially in industries managing sensitive data, and mitigated the risk of regulatory infractions.

AI-driven testing automation systems, encompassing defect identification and test scenario creation, have demonstrated exceptional efficacy in finding faults during the early phases of development. In recent studies a deep learning-based model for defect identification was developed, surpassing standard static analysis techniques in identifying code vulnerabilities [46]. The model

employed categorisation methods to autonomously identify vulnerabilities and performance constraints. AI methodologies, particularly clustering and classification, facilitated the automated categorisation of analogous code segments, hence enhancing the identification of recurring problems and the optimisation of testing processes. This method not only identified probable errors prior to the testing phase but also conserved considerable time and money by recognising patterns in the code that may result in problems in later stages.

New mechanisms were proposed by Sangeetha and Lakshmi [38] to ensure the transparency and interpretability of AI models, which increase trust in these systems and their compliance with ethical standards. Innovative tools for monitoring and assessing ethical risks have been developed, allowing for effective management of potential threats. Methods of protecting AI models from specific attacks were also introduced, which significantly improved the reliability and security of these systems, making them more resistant to manipulation and vulnerabilities. The research conducted by Sangeetha and Lakshmi illustrated the use of active learning algorithms in Java code sourced from open-source sources. These techniques concentrated on pinpointing the most troublesome segments of code and enhancing test coverage. The study indicated that active learning decreased the time required to identify flaws and markedly enhanced the testing process by prioritising high-risk code sections.

To successfully implement AI in software engineering, it is important to provide ongoing staff updates and training to enable developers to use new AI tools and algorithms and reduce potential errors. Integrating AI into existing workflows should be done in stages, starting with the automation of routine tasks such as testing and code generation, which can be used to focus on more creative aspects of work. Particular attention should be paid to security and ethics, including the protection of data, especially personal and sensitive data, in the process of using AI, as well as compliance with ethical standards and privacy laws. It is equally important to ensure the transparency of AI algorithms, which will allow users to better understand the decisions made by the system and identify possible problems.

Based on the analysis of existing approaches, this study proposes methods for protecting AI models from specific attacks, such as data poisoning and manipulation of input data Kotti et al. [47]. One of these methods is to use techniques to verify the authenticity of input data before it is processed by models, which allows for detecting manipulation attempts and preventing malicious or incomplete data from entering the system. The study by Kotti et al. examined the implementation of ethical AI assessments, utilising data from financial services and healthcare systems to ensure adherence to GDPR and ethical norms. The study revealed that integrating ethical assessments across the software lifecycle, from requirements analysis to product support, enhanced product credibility and diminished algorithmic bias. These approaches guaranteed that AI systems complied with privacy regulations and fostered equity in decision-making. Another approach is the use of anomaly-based protection algorithms that can detect any deviations in model behaviour caused by data attacks. These methods can improve the reliability and security of AI systems, which is critical for their implementation in software engineering, especially when processing large amounts of sensitive information, where even minor manipulations can lead to serious consequences.

Retraining and professional development programmes are especially important, as they ensure a smooth transition and adaptation of employees to new conditions. Consideration of these factors reduces resistance to change and facilitates the integration of AI into software engineering practices, which significantly increases the efficiency of innovative technologies. This approach contributes to the creation of a sustainable workforce capable of effectively using the latest technologies, which confirms the importance of systematic training for the successful integration of AI.

The study's findings underscore the substantial advantages of AI-driven methodologies in contemporary software development. AI-driven code creation and flaw detection significantly enhance productivity, accuracy, and scalability by automating repetitive processes, minimising human mistakes, and expediting development schedules. AI models, such as Codex for code generation, yield syntactically and semantically precise code, conserving time and reducing errors, while AI-enhanced testing automation increases defect detection accuracy, diminishes false positives, and facilitates the prompt identification of critical issues, ultimately decreasing costs and improving software quality. These enhancements enable organisations to optimise resources, expand test coverage, and accelerate time-to-market while simultaneously promoting creativity through the automation of monotonous chores and allowing developers to concentrate on intricate, creative endeavours. As AI technologies become increasingly available, they provide strategic benefits for both major corporations and smaller organisations, making them indispensable for the future of software development.

The study proposes several mechanisms to ensure the transparency and interpretability of artificial intelligence models. One of these approaches is the use of techniques for visualising the internal processes of the model, demonstrating how decisions are made at each stage. In addition, the study proposed to use interpretable models, such as Local Interpretable Model-agnostic Explanations (LIME), to explain the decisions of complex machine learning models [48]. This helps reduce the level of distrust in systems and ensure compliance with ethical standards, as it allows users and developers to understand the logic of decisions.

It is also necessary to consider cultural and social contexts when developing and implementing AI systems. Algorithms developed in one cultural environment may not function adequately in another, which can lead to misunderstandings or negative consequences. Therefore, it is necessary to involve multidisciplinary teams and assess the impact on different population groups.

4 Discussion

The theoretical study revealed that the use of AI at various stages of software development can significantly increase the efficiency of processes such as automation of code generation, testing, and defect detection. A comparison of existing approaches has shown that methods based on deep neural networks have a significant advantage over classical techniques, as they can adapt to different programming languages and software types, providing greater versatility. Furthermore, the use of active learning to automatically identify problematic areas of code has proven to be effective in improving test coverage and reducing testing time. However, despite these advantages, the implementation of AI in practical development still needs to be improved, in particular in terms of adaptation to real-world development conditions and integration with existing tools.

One of the most active areas of AI applications in software development is automated code generation [49]. Code generation is a complex process that requires the incorporation of many factors, such as syntax, programming language semantics, and customer needs [50]. The study determined that the use of deep neural networks for code generation is extremely promising. Such networks can learn patterns in the code, which allows them to generate fragments of software code whose accuracy is much higher than traditional methods. However, according to the study by Kim et al. [39], the use of transformers for code generation is one of the most modern methods, although it has limitations in the context of the interpretability of the results. In this study, hybrid models that combine machine learning and traditional algorithmic methods were proposed for better accuracy and reliability of code generation, especially for complex systems. Hybrid models, contrary to transformational models, can not only generate code but also check its compliance with technical requirements, making them more versatile and suitable for complex applications. According to the study, such models can be used to address the problem of transformers' limitations, providing not only high-quality generation but also integration with other stages of software development.

Test automation is another important area where AI can bring significant optimisation. Traditionally, testing has been time-consuming and resource-intensive, test scenarios and validation of each piece of code for defects are done manually. However, with the development of AI, in particular machine learning algorithms, it has become possible to automatically generate test scenarios, detect defects in the code, and even predict failure points. One approach to test automation uses search algorithms to automatically generate test cases. Although this approach is effective, it requires significant computing resources and does not always provide complete test coverage. More effective are methods that use deep learning to detect defects in the code. Approaches such as the one by Liu et al. [42] can predict defects in the code, but they are limited to certain programming languages or types of applications.

Compared to the results of studies such as by Anik et al. [51], which analysed deep neural networks, traditional methods such as SVM or Naive Bayes demonstrate limited adaptability. For instance, deep neural networks trained on multilingual datasets provide versatility and efficiency in detecting code defects regardless of the programming language, while SVMs and other classical methods often require customisation for specific languages and data types. However, traditional methods have advantages in learning speed and computational efficiency, especially on small datasets.

In addition, active learning approaches used in deep neural networks allow for the automatic identification of critical areas for testing, increasing test coverage, as noted by Anik et al. [51] In this context, the results of the current study demonstrate that classical algorithms, although less versatile, are still an effective choice for narrower tasks, especially when resources or data are limited.

Deep learning can improve the accuracy and efficiency of error detection, as well as automate complex development stages that previously required significant human resources. In the future, the development of such methods can lead to a significant increase in software productivity and quality. Existing research is yielding results, but there are still problems that need to be addressed. It is necessary to develop universal models that can work with different programming languages and types of software. It is also important to test the proposed methods in real software development environments to assess their effectiveness and adaptability to different environments.

Compared to deep neural networks and transformers, hybrid models proposed by Ip [52] integrate machine learning with classical algorithmic approaches. They provide not only generation but also verification of compliance with technical requirements. For example, such models can automatically add documentation to the code and check the style and compliance with standards. However, hybrid models may be less flexible in new environments, as much of their functionality is based on defined rules.

Deep neural networks are beneficial in code generation, but they are inferior to transformers in their ability to adapt to broad contexts. Hybrid models, on the other hand, provide greater reliability and accuracy for specific tasks but are more complex to develop. For real-world software development environments, transformers are preferred due to their versatility, although hybrid models may be better for critical systems with strict requirements.

The study of modern methods of test automation based on the use of search algorithms and neural networks demonstrates significant advantages over traditional approaches such as static testing or manual testing. Such methods can significantly reduce the time to detect errors, increase test coverage, and improve the accuracy of software quality control. However, there are challenges related to the versatility of these methods: they must be able to work effectively in different technological environments and with different programming languages. Compared to traditional approaches, which are limited to

specific environments, new AI-based methods offer more flexibility and scalability, but require additional research to achieve full universality.

Based on the analysis, the best methods for use in software engineering are those that strike a balance between accuracy, adaptability, and efficiency. In code generation, transformers such as Codex perform best for general-purpose tasks, while hybrid models are optimal for specific and critical systems due to their ability to verify compliance with technical requirements. In test automation, methods based on active learning and deep neural networks outperform traditional search algorithms due to their greater test coverage and accuracy, although the latter remains effective for projects with limited resources. In defect detection, deep learning methods offer versatility and high accuracy, especially in multilingual environments, while classical methods such as SVMs are useful for smaller projects due to their speed and simplicity. To increase productivity, it is important to develop models that can be easily integrated into existing development tools (e.g., CI/CD systems) and adapt to the specifics of different environments. The further development of AI methods should be aimed at creating integrated solutions that combine the flexibility of transformers, the accuracy of hybrid models, and the efficiency of traditional algorithms. This will not only improve the quality and productivity of development but also make software engineering more adaptive to the needs of modern technologies.

5 Conclusions

This research evaluated the incorporation of AI into the software development process, emphasising its capacity to streamline processes, decrease development duration, and improve product quality. Critical discoveries indicate that AI applications, especially machine learning and deep neural networks, may markedly enhance several phases of the software life cycle, encompassing requirements analysis, design, testing, and code production. By automating repetitive operations, like code generation and test case production, AI minimises human error, expedites development, and guarantees more stable and dependable software.

The study underscored the efficacy of AI in automating software testing and defect identification, illustrating that machine learning algorithms may identify problems early in the development phase, hence enhancing the overall quality of the product. Furthermore, AI's contribution to natural language processing (NLP) in requirements analysis has demonstrated its utility since NLP models may autonomously convert textual specifications into formal models, hence reducing misinterpretations between developers and clients.

Despite the potential benefits, the research also recognised some problems, such as the necessity for enhanced openness and interpretability of AI models, the ethical ramifications of their use, and apprehensions around data privacy and security. These concerns highlight the necessity of creating responsible AI systems that comply with ethical principles and security protocols to guarantee the dependability and credibility of AI in software development.

Given these limitations, further research should be devoted to the development of universal methods for integrating AI at different stages of software development, including for specific types of programs and applications. It is also necessary to conduct experimental testing of the proposed approaches in real conditions and on various platforms to assess their effectiveness and adaptability to different environments.

References

[1] Edy Susanto and Zahra Dinul Khaq, “Enhancing Customer Service Efficiency in Start-Ups with AI: A Focus on Personalisation and Cost Reduction”, *Journal of Management and Informatics*, 3(2): 267-281, 2024. https://doi.org/10.51903/jmi.v3i2.34

[2] Lianping Chen and Muhammad Ali Babar, “Variability Management in Software Product Lines: An Investigation of Contemporary Industrial Challenges”, in *Proceeding of 14th International Conference: Software Product Lines: Going Beyond*, J. Bosch, J. Lee, Eds., Heidelberg: Springer Berlin, 2010, pp. 166-180.

[3] Burr Settles, *Active Learning*, Cham: Springer, 2012.

[4] Miltiadis Allamanis, Earl T. Barr, Prem Devanbu and Charles A. Sutton, “A Survey of Machine Learning for Big Code and Naturalness”, *ACM Computing Surveys*, 51(4): 81, 2018. https://dl.acm.org/doi/10.1145/3212695

[5] Christoph Molnar, *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*, Munich: Lean Publishing, 2019.

[6] Hadhemi Jebnoun, Md. Saidur Rahman, Foutse Khomh and Biruk Asmare Muse, “Clones in Deep Learning Code: What, Where, and Why?”, *Empirical Software Engineering*, 27: 84, 2022. https://doi.org/10.1007/s10664-021-10099-x

[7] Psico-Smart Editorial Team. “How can Artificial Intelligence be Utilized in Performance Analysis and Evaluation?” PsicoSmart. 2024. [Online.] Available: https://psico-smart.com/en/blogs/blog-how-can-artificial-intelligence-be-utilized-in-performance-analysis-and-evaluation-139540

[8] Paula Boddington, “Towards the Future with AI: Work and Superintelligence”, in *AI Ethics: A Textbook*, Singapore: Springer, 2023, pp. 409-456.

[9] N.M. Anoop Krishnan, Hariprasad Kodamana and Ravinder Bhattoo, “Interpretable Machine Learning”, in *Machine Learning for Materials Discovery: Numerical Recipes and Practical Applications*, Cham: Springer, 2024, pp. 159-171.

[10] Hongjia Li, Tianshu Wei, Ao Ren, Qi Zhu and Yanzhi Wang, “Deep Reinforcement Learning: Framework, Applications, and Embedded Implementations: Invited Paper”, in *2017 IEEE/ACM International Conference on Computer-Aided Design*, Irvine: Institute of

Electrical and Electronics Engineers, 2017, pp. 847-854.

[11] Saleema Amershi, Andrew Begel, Christian Bird, Robert DeLine, Harald Gall, Ece Kamar, Nachiappan Nagappan, Besmira Nushi and Thomas Zimmermann, "Software Engineering for Machine Learning: A Case Study", in *Proceedings of 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice*, Montreal: Institute of Electrical and Electronics Engineers, 2019, pp. 291-300.

[12] Silvio Andrae, "Cyber Risk Assessment Using Machine Learning Algorithms", in *Advances in Computational Intelligence and Robotics*, Hershey: IGI Global, 2025, pp. 187-218.

[13] Satvik Garg, Pradyumn Pundir, Geetanjali Rathee, P.K. Gupta, Somya Garg and Saransh Ahlawat, "On Continuous Integration / Continuous Delivery for Automated Deployment of Machine Learning Models using MLOps", in J. Gurrola, Ed., *Proceedings of 2021 IEEE Fourth International Conference on Artificial Intelligence and Knowledge Engineering*, Hills: Institute of Electrical and Electronics Engineers, 2021, pp. 25-28.

[14] Samir Jaber, "Intelligent Software Testing and AI-Powered Apps: From Automated Defect Prediction to Context-Aware Mobile Services", 2024. https://doi.org/10.13140/RG.2.2.20694.46401

[15] Milda Maciejauskaitė and Jolanda Miliauskaitė, "The Efficiency of Machine Learning Algorithms in Classifying Non-Functional Requirements", *New Trends in Computer Sciences*, 2(1): 46-5, 2024. https://doi.org/10.3846/ntcs.2024.21574

[16] Mamta Kalra and Suman Sangwan, "Machine Learning and Deep Learning Techniques for Recommendation Systems: A Comprehensive Review", *Journal of Harbin Engineering University*, 45(5): 158-172, 2024.

[17] Vitalii Shymko and Oleg Slipych, "The use of innovative solutions and VR-technologies in architectural design and construction", *Mining Journal of Kryvyi Rih National University*, 58(1): 110-114, 2024. https://doi.org/10.31721/2306-5435-2024-1-112-110-114

[18] Oleksandr Tkachenko, Aleksei Chechet, Maksim Chernykh, Sergei Bunas and Przemysław Jatkiewicz, "Scalable Front-End Architecture: Building for Growth and Sustainability", *Informatica (Slovenia)*, 49(1): 137-150, 2025. https://doi.org/10.31449/inf.v49i1.6304

[19] Seyit Kerimkhulle, Ainur Saliyeva, Ulzhan Makhazhanova, Zhandos Kerimkulov, Alibek Adalbek and Roman Taberkhan, "The estimate of innovative development of construction industry in the Kazakhstan", *E3S Web of Conferences*, 389: 06004, 2023. https://doi.org/10.1051/e3sconf/202338906004

[20] Huaming Chen and M. Ali Babar, "Security for Machine Learning-based Software Systems: A Survey of Threats, Practices and Challenges", *ACM Computing Surveys*, 56(6): 151, 2022. https://doi.org/10.1145/3638531

[21] Itzhak Aviv, Ruti Gafni, Sofia Sherman, Berta Aviv, Asher Sterkin and Etzik Bega, "Infrastructure From Code: The Next Generation of Cloud Lifecycle Automation", *IEEE Software*, 40(1): 42-49, 2023. https://doi.org/10.1109/MS.2022.3209958

[22] Prajakta Sudhir Khade and Dr. Rajeshkumar U. Sambhe, "Artificial Intelligence in Software Development: A Review of Code Generation, Testing, Maintenance and Security", *International Journal of Current Science Research and Review*, 8(4): 1632-1641, 2025. https://doi.org/10.47191/ijcsrr/V8-i4-08

[23] Ruslan Yermolenko, Denys Klekots and Olga Gogota, "Development of an algorithm for detecting commercial unmanned aerial vehicles using machine learning methods", *Machinery and Energetics*, 15(2): 33-45, 2024. https://doi.org/10.31548/machinery/2.2024.33

[24] Sedighe Ajorloo, Amirhossein Jamarani, Mehdi Kashfi, Mostafa Haghi Kashani and Abbas Najafizadeh, "A Systematic Review of Machine Learning Methods in Software Testing", *Applied Soft Computing*, 162: 111805, 2024. https://doi.org/10.1016/j.asoc.2024.111805

[25] Albina Yerzhanova, Seyit Kerimkhulle, Gulzira Abdikerimova, Margaryta Makhanov, Svetlana Beglerova and Zhazira Taszhurekova, "Atmospheric correction of landsat-8 / Oli data using the flaash algorithm: Obtaining information about agricultural crops", *Journal of Theoretical and Applied Information Technology*, 99(13): 3110-3119, 2021.

[26] Oyekunle Claudius Oyeniran, Okechukwu Adewus, Adams Gbolahan Adeleke, Lucy Anthony Akwawa and Chidimma Francisca Azubuko, "AI-Driven Devops: Leveraging Machine Learning for Automated Software Deployment and Maintenance", *Engineering Science & Technology Journal*, 4(6): 728-740, 2023. https://doi.org/10.51594/estj.v4i6.1552

[27] Sabina-Cristiana Necula, Florin Dumitriu and Valerică Greavu-Șerban, "A Systematic Literature Review on Using Natural Language Processing in Software Requirements Engineering", *Electronics*, 13(11): 2055, 2024. https://doi.org/10.3390/electronics13112055

[28] Olga Sokol, "Optimising productivity and automating software development: Innovative memory system approaches in large language models", *Technologies and Engineering*, 26(1): 36-44, 2025. https://doi.org/10.30857/2786-5371.2025.1.3

[29] Suprit Pattanayak, Pranav Murthy and Aditya Mehra, "Integrating AI into DevOps pipelines: Continuous integration, continuous delivery, and automation in infrastructural management: Projections for future", *International Journal of Science and Research Archive*, 13(01), 2244-2256,

2024. https://doi.org/10.30574/ijsra.2024.13.1.1838

[30] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha and Michael P. Wellman, "Sok: Security and Privacy in Machine Learning", in L. O'Conner, Ed., *2018 IEEE European Symposium on Security and Privacy*, London: Institute of Electrical and Electronics Engineers, 2018, pp. 399-414.

[31] Christopher Kuner, Lee A. Bygrave and Christopher Docksey, "Background and Evolution of the EU General Data Protection Regulation (GDPR)", in C. Kuner, L.A. Bygrave, C. Docksey, L. Drechsler, Eds., *EU General Data Protection Regulation* (GDPR), New York: Oxford University Press, 2020, pp. 1-47.

[32] Yundong Zhao, "Optimization of Mechanical Manufacturing Processes Via Deep Reinforcement Learning-Based Scheduling Models", *Informatica*, 49(14): 19-32, 2025. https://doi.org/10.31449/inf.v49i14.7204

[33] Roman Shults, Mykola Bilous, Azhar Ormambekova, Toleuzhan Nurpeissova, Andrii Khailak, Andriy Annenkov and Rustem Akhmetov, "Analysis of Overpass Displacements Due to Subway Construction Land Subsidence Using Machine Learning", *Urban Science*, 7(4): 100, 2023. https://doi.org/10.3390/urbansci7040100

[34] Dmytro Belytskyi, Ruslan Yermolenko, Kostiantyn Petrenko and Olga Gogota, "Application of machine learning and computer vision methods to determine the size of NPP equipment elements in difficult measurement conditions", *Machinery and Energetics*, 14(4): 42-53, 2023. https://doi.org/10.31548/machinery/4.2023.42

[35] Bohdan Cherniavskyi, "Integration of Drones and Dio-Inspired Algorithms into Intelligent Transportation Logistics Systems for Post-war Remediation of Ukraine", *Lecture Notes in Networks and Systems*, 1336: 426-437, 2025. https://doi.org/10.1007/978-3-031-87379-9_39

[36] Vasyl Nesterov, "Integration of artificial intelligence technologies in data engineering: Challenges and prospects in the modern information environment", *Bulletin of Cherkasy State Technological University*, 28(4): 82-92, 2023. https://doi.org/10.62660/2306-4412.4.2023.82-90

[37] Ivan Biliuk, Dmytro Shareyko, Oleg Savchenko, Serhii Havrylov, Andrii Fomenko and Anatolii Tubaltsev, "Machine Calculation of the Problem of Expansion of the Magnetic Field Measurement Grid", in *Proceedings of the 5th International Conference on Modern Electrical and Energy System, MEES 2023*, Kremenchuk: Institute of Electrical and Electronics Engineers, 2023. https://doi.org/10.1109/MEES61502.2023.10402546

[38] Yalamanchili Sangeetha and G. Jaya Lakshmi, "Predicting Software Bugs with Machine Learning Algorithms" in V.L.N. Komanapalli, N. Sivakumaran, S. Hampannavar, Eds., *Select Proceedings of I-CASIC 2020: Advances in Automation, Signal Processing, Instrumentation, and Control*, Singapore: Springer, 2021, pp. 2683-2692. https://doi.org/10.1007/978-981-15-8221-9_251

[39] Seokjun Kim, Jaeeun Jang and Hyeoncheol Kim, "All-In-One: Artificial Association Neural Networks", 2021. https://doi.org/10.48550/ARXIV.2111.00424

[40] Luciano Floridi, Josh Cowls, Monica Beltrametti, Raja Chatila, Patrice Chazerand, Virginia Dignum, Christoph Luetge, Robert Madelin, Ugo Pagallo, Francesca Rossi, Burkhard Schafer, Peggy Valcke and Effy Vayena, "AI4People – An Ethical Framework for a Good AI Society", *Mind and Machine*, 28: 689-707, 2018. https://doi.org/10.1007/s11023-018-9482-5

[41] Reza Shokri and Vitaly Shmatikov, "Privacy-Preserving Deep Learning", in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Monticello: Institute of Electrical and Electronics Engineers, 2015, pp. 909-910.

[42] Hengyuan Liu, Zheng Li, Baolong Han, Xiang Chen, Doyle Paul and Yong Liu, "Integrating Neural Mutation into Mutation-Based Fault Localization: A Hybrid Approach", *Journal of Systems and Software*, 211: 112281, 2024. https://doi.org/10.1016/j.jss.2024.112281

[43] Anbarasu Arivoli, "AI for automated bug detection and debugging: a comparative study of current approaches", *International Journal of Business Quantitative Economics and Applied Management Research*, 7(12): 57-66, 2024. https://doi.org/10.5281/zenodo.15236718

[44] Blanco Ruiz, "The Role of Automation in Modern Software Testing", *Journal of Computer Engineering & Information Technology*, 12(4), 2023. https://doi.org/10.4172/2324-9307.1000279

[45] Daiju Ueda, Taichi Kakinuma, Shohei Fujita, Koji Kamagata, Yasutaka Fushimi, Rintaro Ito, Yusuke Matsui, Taiki Nozaki, Takeshi Nakaura, Noriyuki Fujima, Fuminari Tatsugami, Masahiro Yanagawa, Kenji Hirata, Akira Yamada, Takahiro Tsuboyama, Mariko Kawamura, Tomoyuki Fujioka, and Shinji Naganawa, "Fairness of artificial intelligence in healthcare: Review and recommendations", *Japanese Journal of Radiology*, 42, 3-15, 2024. https://doi.org/10.1007/s11604-023-01474-3

[46] Gopinath Kathiresan, "Automated Test Case Generation with AI: A Novel Framework for Improving Software Quality and Coverage", World Journal of Advanced Research and Reviews, 23(02): 2880-2889, 2024. https://doi.org/10.30574/wjarr.2024.23.2.2463

[47] Zoe Kotti, Rfaila Galanopoulou and Diomidis Spinellis, "Machine Learning for Software Engineering: A Tertiary Study", *ACM Computing*

Surveys, 55(12): 256, 2023. https://doi.org/10.1145/3572905

[48] Marco Tulio Ribeiro, Sameer Singh and Carlos Guestrin, ""Why should I Trust you?": Explaining the Predictions of Any Classifier", in: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York: Association for Computing Machinery, 2016, pp. 1135-1144.

[49] Olesia Vasetska, "A study of the electric circuit modelling and simulation software efficiency and their accuracy, speed and ease of use comparison", *Bulletin of Cherkasy State Technological University*, 29(2): 32-44, 2024. https://doi.org/10.62660/bcstu/2.2024.32

[50] Itzhak Aviv, D. Gafni, Sofia Sherman, Bertha Aviv, Asher Sterkin and E. Bega, "Cloud infrastructure from python code–breaking the barriers of cloud deployment", in: *European Conference on Software Architecture, ECSA,* 2023, pp. 1-8. https://www.researchgate.net/profile/Itzhak-Aviv/publication/373897534_Cloud_Infrastructure_from_Python_Code_-breaking_the_Barriers_of_Cloud_Deployment/links/6501edd2808f9268d573dea5/Cloud-Infrastructure-from-Python-Code-breaking-the-Barriers-of-Cloud-Deployment.pdf

[51] Shafayat Mowla Anik, Kevyn Kelso and Byeong Kil Lee, "Efficient Layer Optimizations for Deep Neural Networks", *International Journal of Soft Computing and Engineering,* 14(5): 20-29, 2024. http://www.doi.org/10.35940/ijsce.E3650.14051124

[52] Si Hua Robert Ip, "Research on Explainability of Deep Neural Networks and Its Applications", *Highlights in Science, Engineering and Technology*, 115: 441-450, 2024. https://doi.org/10.54097/ekzm5z29

A Comparative Performance Analysis of CNN, LSTM, and CNN-LSTM Models for Classifying Sleep Stages Using Electroencephalography Signals

Zainab N. Nemer[1], Saba Abdual Wahid Saddam[2], Rana J.AL_Sukeinee[3], Esra'a J. Harfash[4]
[1,2,4]Department of Computer Science, College of Computer Science and Information Technology, University of Basra, Iraq.
[3]Department of Physics Science, College Science, University of Basra, Iraq
E-mail: zainab.nemer@uobasrah.edu.iq , Saba.Saddam@uobasrah.edu.iq , rana.jabbar@ uobasrah.edu.iq ,esra.harfash@uobasrah.edu.iq

Keywords: LSTM networks, convolutional neural networks (CNN), sleep stage classification, hybrid models, EEG signals

Received: January 16, 2025

This paper aims to study and analyze the performance of CNN, LSTM and CNN+LSTM deep neural networks individually in the problem of sleep cycle stage classification, by analyzing the recorded EEG signals, and highlight the strengths of each model in this classification problem. The deep learning models selected in this work are reliable models in the world of experiments and medical classification. The data used here is the EEG signal represented by the sleep-edf-expanded-npzs database which carries information about the different sleep stages, which are five stages: wakefulness (W), non-REM stage 1 (N1), non-REM stage 2 (N2), non-REM stage 3 (N3), and rapid eye movement (REM). This data was split by training by 70% and testing by 30%. The models were trained based on cross-entropy loss and Adam optimizer. The results obtained in this study showed that the classification rates are as follows: CNN model accuracy (0.8736%), LSTM model accuracy (0.8306%), and LSTM + CNN hybrid model accuracy (0.8813%). These results indicate the good performance of the CNN model, which can be attributed to its strong ability to extract spatial features from data. The results of the LSTM approach demonstrate its ability to track and interpret the temporal characteristics of the sleep signal. For the CNN-LSTM model, which combines the strengths of temporal LSTM and spatial CNN, the result of combining features in this sequence showed superior results than either CNN or LSTM alone in processing sleep signals. Statistical measures were used to validate the results obtained. The F-statistic was 949.78666666523 and the probability value was 6.119863 e-14, which validate the results obtained for each model.

Povzetek: Narejena je primerjava modelov CNN, LSTM in hibridnega CNN-LSTM za klasifikacijo faz spanja na osnovi EEG signalov, ki pokaže, da LSTM-CNN arhitektura dosega najboljše rezultate.

1 Introduction

Sleep is essential for children, adolescents, and adults' health and well-being and is a biological necessity. It is essential for mood, emotional well-being, cardiovascular health, and brain health. Restoring sleep is strongly associated with improved physical, mental, and cognitive health as well as a lower chance of accidents and injuries brought on by weariness and drowsiness. Sleep is an essential component of the human daily routine. On the other hand, inadequate or disrupted sleep can result in a decline in general physical health as well as possible cognitive and psychological damage. [1][2].

The American Academy of Sleep Medicine (AASM) has classified wakefulness (W), rapid eye movement (REM), and non-rapid eye movement (NREM) sleep of increasing depth into N1, N2, and N3 as the five stages of wake and sleep. The body typically goes through these phases four to six times, with an average of ninety minutes spent in each step [3][4].

Although there are several ways to evaluate the brain's stages of sleep, electroencephalography (EEG) is still the preferred tool for studying human sleep [5]. In sleep research, electroencephalography (EEG) is the most often utilized technology. Sample data are given for a normal EEG examination conducted at night while REM (rapid eye movement) and NREM sleep are occurring [6].

With the availability of sleep EEG data, many researchers in the field of machine learning, specifically in the field of deep learning, have practical experience in sleep classification through different stages of methods and techniques. Among these networks is the LSTM model, as LSTM structures have the ability to learn long-term based on sequential data, which makes them well suited for tasks such as predicting time series such as sleep signal. Therefore, they are relied upon to solve special

problems in computational biology and computational biology. [7][8].

Convolutional neural networks (CNNs) are very effective at capturing spatial features and have shown excellent results in detecting complex patterns within EEG data. CNNs and LSTMs are also widely used in EEG signal analysis. Performance in time series classification problems can be improved by combining CNNs and LSTMs (CNN-LSTM hybrid model), While CNNs can learn local patterns in the data, LSTMs can identify long-term dependencies in sequential data. Combining CNNs and LSTMs exploits spatial and temporal patterns [9,10].

Most studies have relied on one of the above-mentioned models individually, namely CNN, LSTM, or CNN-LSTM, and some of these studies have relied on the Sleep-EDF database or other databases. However, no study has compared these three models on the same database. Most previous studies have evaluated these models individually or using different EEG datasets.

This study aims to bridge this gap by presenting a study that evaluates and compares the effectiveness of three models: LSTM, CNN, and a hybrid model (CNN, LSTM)—in automatically classifying sleep stages, which involves conducting a comparative study among these three models using the same EEG signal database. . The study aims to:

1. Identify which of these models achieves the highest performance according to the same criteria, in terms of the Sleep-EDF database and processing methods.
2. Provide a comprehensive systematic evaluation to determine the best-performing model.
3. Identify whether standalone CNN or LSTM models, or a hybrid CNN with LSTM model, provide superior performance.
4. Also, test which hybrid architecture (CNN_LSTM) or (LSTM_CNN) is better.

Through multiple experiments, it was found that the hybrid architecture performs better, especially the LSTM_CNN architecture. Furthermore, the CNN model is capable of providing superior results, given its limited capabilities to spatial features, while LSTM also demonstrated good results.

The study's objectives are clearly outlined, centering on a comparison of performance and an identification of each model's strengths in sleep stage classification.

The rest of the study is organized as follows. Section 2 provides a literature review on sleep stage classification. Typical materials and methods are covered in Section 3. Implementation and results discussion are presented in Section 4. Study Conclusion and Recommendations are outlined in Section 5.

2 Literature review

Numerous researchers have attempted to address and enhance the effectiveness of sleep stage classification through a variety of concepts and approaches; however, deep learning algorithms are among the most significant approaches that have explored this area in order to accomplish automated sleep stage recognition. The study conducted by Nicola M et al. (2019) to automatically score sleep stages using EEG signals from a single channel, a unique cascaded recurrent neural network (RNN) architecture based on long short-term memory (LSTM) blocks is proposed. And to choose the most pertinent features, fifty-five temporal and frequency-domain features were taken out of the EEG signals and supplied into feature reduction algorithms.[11].

Convolutional Neural Networks (CNNs) is important models in sleep stage classification problem. Asma et al. (2020) study highlights a 13-layer 1D convolutional neural network CNN for automatic feature extraction and sleep stage classification using single-channel EEG signal [12].

Furthermore, Zhao et al. (2022), They proposed an algorithm suitable for different physiological signals that can achieve automatic sleep stage from start to finish without any manual feature extraction. Here, one-dimensional convolutional neural network and long short-term memory are adopted. This method can automatically segment sleep into 5 stages including wakefulness, non-rapid eye movement sleep (N1∼N3) and rapid eye movement sleep using EEG signals. [13].

The CNN model has a distinct role in classifying sleep stages, Luis A. et al. (2023), in this paper, are used a convolutional neural network (CNN) based on 5- and 2-class models to study the performance of automatic sleep stage categorization utilizing autonomously selected characteristics from electroencephalogram (EEG) inputs. We established two 2-class sleep stage classification techniques and evaluated how well they performed in comparison to the predictions from a 5-class model. The public ISRUC-Sleep dataset, which includes six EEG channels and 100 participants, was used for tests. All models were constructed using a CNN called EEGNet. [14].

In order to address the issue of training data imbalance, Enes E. A. et al. (2023), in this study suggests a novel hybrid CNN+LSTM neural network design that makes use of focus loss and discrete cosine transform techniques. Using k-fold cross-validation techniques (subject-wise), the model was trained on four distinct databases. When using two channels (EEG-EOG), the best accuracy was 87.11%, [15].

Nowadays, many existing techniques depend on handcrafted features. Only a few methods, meanwhile, are able to recover the temporal information required to determine the stages of sleep. LSTMs will be able to learn transition rules, while convolutional neural networks will be able to extract time-invariant features. A thorough analysis of recent advancements was given by Kotla R. et al. (2023), This paper proposes a deep learning model to automatically score sleep stages using single-channel EEG recordings that combines CNN and LSTM. The model was trained and tested on the Sleep-EDF-v1 dataset, which is publicly available. The single-

channel Fpz-Cz EEG was used and scored according to the AASM standard. An overall accuracy of 83.38% was achieved. [16].

Supratak et al. (2017), are used a two-step training technique to effectively train the DeepSleepNet deep learning network, which is introduced in this work for automatic sleep stage grading based on raw single-channel EEG. Different single-channel EEGs (F4-EOG (left), Fpz-Cz, and Pz-Oz) from two public sleep data sets are used to assess this concept [17].

In order to acquire task-specific filters for classification without requiring prior domain knowledge, Tsinalis et al. (2016) are used the convolutional neural networks (CNNs) to utilized in this work to automatically score sleep stages based on single-channel electroencephalography (EEG). In the CNN's stochastic gradient descent (SGD) optimization, class-balanced random sampling was used to prevent performance bias in favor of the most representative sleep stages [18].

Pham et al. (2023) was presented an automatic sleep stage categorization system utilizing a combination of transformer, long-short-term memory (LSTM), and convolutional neural network (CNN) models using single-channel EEG recordings [19].

To overcome the critical limitations Casciola et al. (2021) are presented a deep learning (DL) model for automated sleep staging of HB EEG data. The solution includes a simple band-pass filtering, a data augmentation step, and a model using convolutional (CNN) and long short-term memory (LSTM) layers [20]

A new model dubbed CAttSleepNet was presented by Li et al. (2022) for automatic sleep stage identification using single-channel EEG, and incorporate an attention module into the convolutional neural network (CNN) so that it can use the contextual information inside the epoch to learn the weights of local sequences of EEG signals. The global connections of succeeding epochs are then encoded using a two-layer bidirectional long-short-term memory (Bi-LSTM) [21].

Table 1: Summary information of existing method

Study	Dataset	Model (Network)	Evaluation Metrics	Accuracy
Nicola M. et al. (2019)	Use of a publicly available sleep-EDF database.	RNN with LSTM	Accuracy and Confusion Matrix	The accuracy of classification for *five* sleep stages is 86.7%
Asma et al. (2020)	sleep-EDF dataset	1D CNN (13-layer)	k-fold cross-validation.set k to 20	Precision of 94.09%, 74.73%, 96.43%, and 71.02%, respective of classifying five sleep stages.
Zhao et al. (2021)	Sleep-EDF, ISRUC	1D CNN-LSTM	Accurcy, Precision, Recall, F1-score	The accuracy of staging is 93.47% using the Fpz-Cz electroencephalogram signal. When using the Fpz-Cz and electroencephalogram signal, the highest accuracy o94.15%.
Luis A. et al. (2023)	ISRUC-Sleep dataset)	CNN (EEGNet)	5-class and 2-class classification accuracy	In the best case, the average AUROC of 0.964,0.967,0.982 and 0.929 for the stratified 2-class models
Enes E. A. et al. (2023)	DRM-SUB, ISRUC3, and SleepEDF-20 databases.	hybrid CNN-LSTM	k-fold validation strategies (subject-wise)	highest score was 87.11% accuracy, 81.81% Kappa score, and 79.83% MF1 when using two channels (EEG-EOG)
Kotla R. et al. (2023)	Sleep-EDF-v1 Dataset	single-channel EEG recordings (CNN +LSTM)	Classification accuracy	83.38% accuracy using single-channel Fpz-Cz EEGs.
Supratak et al. (2017)	Sleep-EDF and MASS	CNN-LSTM	Classification accuracy	(MASS: 86.2%-81.7, Sleep-EDF: 82.0%-76.9) .
Tsinalis et al. (2016)	sleep PSG dataset	CNN	Classification accuracy	Accuracy for each sleep stage (82%, range 80–84%), total accuracy (74%, range 71–76%) for all patients, and average F1 scores (81%, range 79–83%).
Pham et al.	ISRUC	CNN-LSTM	Classification	The experimental evaluation on

(2023)			accuracy	the ISRUC S1 and S3 sleep datasets, It attains 82.40% and 80.37% accuracy, respectively..
Casciola et al. (2021)	electroencephalography (EEG) headbands (HB)	CNN-LSTM	Classification accuracy	74% (±10%) validation accuracy on low-quality two-channel EEG headband data and 77% (±10%) on gold-standard PSG.
Li et al. (2022)	Sleep-EDF	CNN-BiLSTM with Attention Mechanism	Accuracy(k-fold cross-validation.)	Achieving an accuracy of 86.9%.

3 Materials and Methods

3.1. Model Architectures and Mathematical Operations

In this work, research was conducted to know the strength of each model of LSTM, CNN and CNN+LSTM in classifying the different stages of sleep signal according to the features that each model possesses. The following will include the mathematical aspect of building each model with an explanation of the architecture that was adopted in this work..

3.1.1 LSTM Model

Long short-term memory (LSTM) has revolutionized both the fields of machine learning and neural computing. One of the reasons for the success of this recurrent network is its ability to handle the exploding/vanishing gradient problem, which is a difficult problem to overcome when training recurrent or very deep neural networks[22]. LSTMs possess the capacity to process sequential data and retain information from previous steps in the sequence, enabling them to predict future steps effectively. This characteristic makes them highly suitable for tasks involving long-term dependencies [23]. Fig. 1, is illustrated the common LSTM unit is composed of a cell state (denoted by C_t), a forget gate (f_t), an input gate (i_t), and an output gate (o_t). The three gates regulate the flow of information into and out the channel and important information over arbitrary time intervals can be remembered[24]

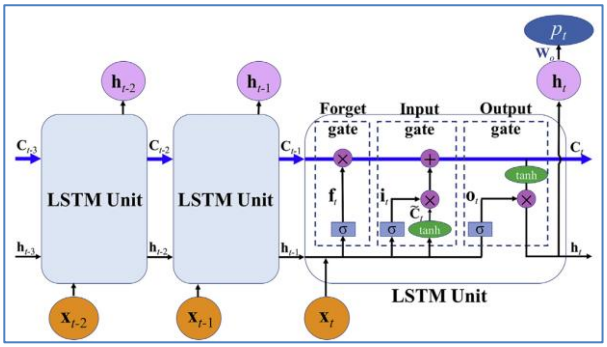


Figure 1: Structures long short-term memory (LSTM)[24]

The functions of a single cell LSTM are detailed in Algorithm , where σ denotes the logistic sigmoid function, tanh represents the hyperbolic tangent function, and $\odot$ denotes the Hadamard product, which is the element-wise product [25]:

Given:
1. W Matrix of input-to-hidden weights
2. U Matrix of hidden-to-hidden weights
3. b Vector of Bias

Input:
1. x_t Vector of input at time step t
2. c_{t-1} Vector of previous memory cell state
3. h_{t-1} Vector of previous hidden state

Output:
1. c_t Vector of memory cell state
2. h_t Vector of the hidden state

Process:
1. Compute the input gate vector:
$$i_t = \sigma\left(W^{(i)}x_i + U^{(i)}h_{t-1} + b^{(i)}\right) \quad (1)$$
2. Compute the forget gate vector:
$$f_t = \sigma\left(W^{(f)}x_i + U^{(f)}h_{t-1} + b^{f}\right) \quad (2)$$
3. Compute the output gate vector:
$$o_t = \sigma\left(W^{(0)}x_i + U^{(o)}h_{t-1} + b^{(0)}\right) \quad (3)$$
4. Compute the memory cell state vector:
$$c_{t=i_t} \odot u_t + f_t \odot c_{t-1} \quad (4)$$
where:
$$u_t = \tanh\left(W^{(u)}x_i + U^{(u)}h_{t-1} + b^{(u)}\right) \quad (5)$$
5. Compute the hidden state vector:
$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

3.1.2 CNN model

One type of deep learning algorithm that has shown great success in tasks like classification is CNN. Convolutional neural network (CNN) is the most advanced deep learning technique due to its ability to learn features independently [26]. Below are some of the advantages CNN has over other traditional neural networks [27]:

1. CNN's weight sharing feature, which lowers the number of trainable network parameters and helps the network improve generalization and prevent overfitting, is the primary justification for considering it.

2. The model output is very structured and heavily dependent on the retrieved features as a result of learning the feature extraction layers and the classification layer simultaneously.
3. Compared to other neural networks, CNN makes large-scale network deployment considerably simpler.

Figur (2) show the Building block of a typical CNN[28]

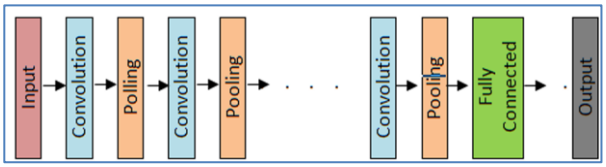


Figure 2: The Conceptual model of CNN

Below is an explanation of the formulas for the CNN steps:

N :is the Number of input features ,
x_i :is the Input data at position i ,
w_{ij} :the Weight connecting input i to output j ,
b_j: :Bias for the j-th output neuron and σ is the
a_i :is input (that resulted of provis layer)
z_i :is Output feature map
o_j :is the output of fully connected layer
σ : is the Activation function

1. Convolution Layers: The convolution layer (CONV) uses filters that perform convolution operations[29]:

$$\boldsymbol{z_i = b_i + \sum x_i \in W_{ij} * x_i} \quad (7)$$

After the convolution, an activation function such as the Rectified Linear Unit (ReLU) is applied to introduce non-linearity[30][31]:

$$\boldsymbol{g_i = ReLU(z_i) = \max(0, z_i)} \quad (8)$$

2. Pooling Layer : Pooling operation is an important operation in deep learning. Pooling operation can reduce the feature dimension, the number of parameters, the complexity of computation, and the complexity of time[32]: $P_i = max(z_{i:i+s})$

3. Fully Connected Layer: The fully connected layer (FC) operates on a flattened input where each input is connected to all neurons:

$$\boldsymbol{o_j = \sigma\left(\sum_{i=0}^{N-1} w_{ij}.a_i + b_j\right)} \quad (9)$$

4. Loss Function: This prediction error tells the network how o their prediction from the actual output, and then this error will be optimized during the learning process of the CNN model. This function calculate as[33]:.

$$\boldsymbol{Loss = -\sum_{i=1}^{output\ size} y_i.\log \hat{y}_i} \quad (10)$$

Where L is the loss value,M is the number of classes,$y_{i_}$ is the actual value of class i (either 1 or 0) and $\mathbf{y\hat{}_i}$ is the predicted probability for class **i**.

3.1.3 CNN+LSTM Model

Hybrid networks (CNNs, LSTMs) provide a solution for utilizing both structural and spatial information, combining the strengths of both spatial and temporal networks (CNNs). A CNN-LSTM network is used in a variety of problems, including activity recognition [34]. It is a combination of CNN layers that first extract spatial features, then feed these features as input to LSTM layers to provide time-sequence predictions.

In the LSTM-CNN architecture, the order is reversed: the EEG time series is first processed by the LSTM layers to model temporal dependencies, and the resulting features are then fed to the CNN for analysis by the CNN layers to extract higher-level spatial features. Figure 3 illustrates both hybrid models: (a) CNN-LSTM and (b) LSTM-CNN:

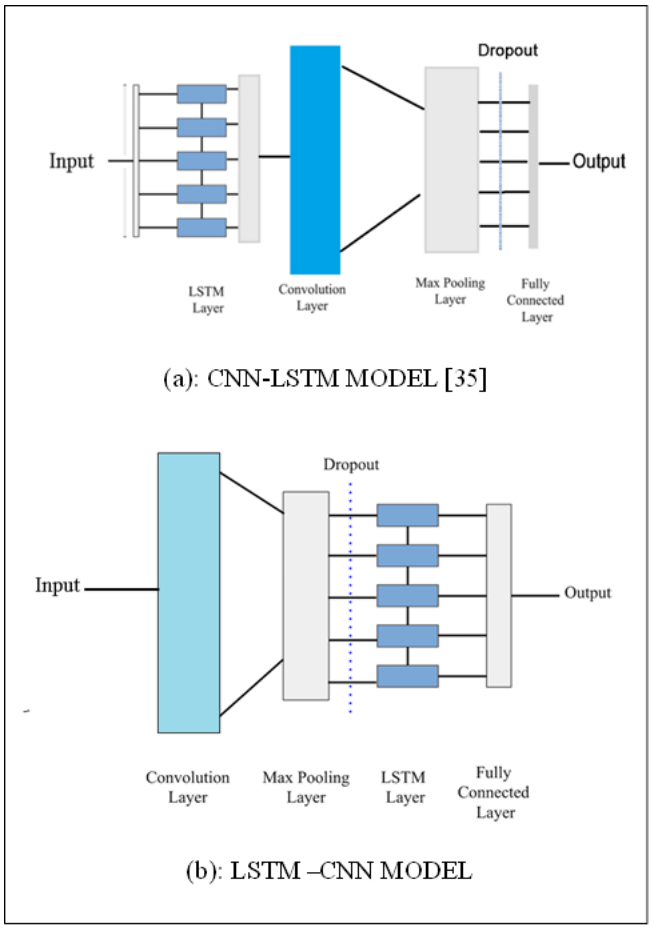


Figure 3: The basic architecture of CNN-LSTM and LSTM-CNN

3.2. Dataset and Preprocessing

The analysis was performed on the Sleep-EDF database that is in the public domain, and it contains 150 files origin from multiple files of EEG signals from multiple dumplings. The files contain recordings of all sleep

stages: Wake (W), Non-REM stages (N1, N2, N3), and REM.

3.2.1 Dataset Description

All EEG signals were stored in npz format ,and Each signal file contains the following key components:

- EEG Signal Data (x): Represents the brain's electrical activity recorded during the study.
- Labels (y): Corresponding annotations for the sleep stages, categorized into five distinct classes:
  1. Wake (W)
  2. Non-REM Stage 1 (N1)
  3. Non-REM Stage 2 (N2)
  4. Non-REM Stage 3 (N3)
  5. REM (Rapid Eye Movement)
- Sampling Frequency (fs): Defines the signal acquisition rate, typically set at 100 Hz.
- Channel Labels (ch_label): Specifies electrode placements, such as (EEG Fpz-Cz) and (EEG Pz-Oz).

Example of a file**:** The file "SC4001E0.npz" includes:

- Shape of x: (841, 3000, 1), which indicates 841 signal segments, each containing 3000 samples (30 seconds per segment at 100 Hz).
- Shape of y: (841,), which contains the corresponding sleep stage labels for each segment.
- Sample Signals:

```
[[[  8.111 ],
 [ 17.488 ],
 [ 21.239 ],
 ...
 [-10.361 ],
 [-11.112 ],
 [ -2.109 ]],
[[-10.736 ],
 [-11.393 ],
 [ -4.454 ],
 ...
 [ 58.842 ],
 [ 48.339 ],
 [ 53.684 ]]]
```

After the file EEG data is read, the labels are extracted from it, and all the signals are collected into a single array. Also the labels are merged into a single array.

3.2.2 Preprocessing Steps

The EEG signals were prepared for analysis through the following preprocessing steps:

1. After the file EEG data is read, the labels are extracted from it, and all the signals are collected into a single array. Also the labels are merged into a single array.
2. Signal normalization: All signals were scaled to a standard distribution with mean 0 and standard deviation 1, using (Z-score normalization) to maintain numerical stability during model training..
3. Segmentation: Signals were divided into fixed-length windows of 30 seconds (3000 samples), in alignment with standard sleep study practices.
4. Class Balancing:
   - The dataset exhibited an imbalance across the five sleep stages.
   - Techniques such as oversampling and SMOTE (Synthetic Minority Oversampling Technique) were used to ensure fair representation of all classes.

   The distribution of sleep stage classes in the training dataset before and after the application of the SMOTE technique is shown in Table (2)

Table (2): The sleep stage classes Distribution before and after SMOTE augmentation

Sleep classes	WITHOUT SMOTE	WITH SMOTE
Class 0 Wake (W)	65357	54225
Class 1 (N1)	21323	54044
Class 2 (N2)	67606	54009
Class 3 (N3)	12833	54006
Class 4 REM	25511	54140

5. The data was split into 80% training and 20% testing.

These steps ensured the EEG data were standardized, balanced, and ready for deep learning model training. There is note of EEG data preprocessing in this study involved several critical steps, including normalizing the signal with Z-score scaling, dividing each recording into fixed 30-second segments (3000 samples), and achieving class balance through SMOTE. The analysis was conducted using a single EEG channel (Fpz-Cz) without the need for channel fusion.

To conduct experiments and ensure reproducibility, we used the Kaggle platform, equipped with 16 GB of RAM and P100 processors. The code was implemented using Python 3.8 and the TensorFlow 2.10.0 library. A fixed random seed (Seed = 42) was used for all libraries (Python, NumPy, and TensorFlow). Using fixed random sampling, the data was split into a training set (80%) and a test set (20%) to ensure reproducibility.

3.3 Training and Evaluation Basics

There are some basics that must be prepared before starting the (training process), which are as follows:

- As mentioned above, the NPZ files containing the EEG signals and their respective labels were read and combined into one array. The longest signals were normalized to 3,000 samples (i.e., using the Reflect Padding method). The classes were then balanced using the SMOTE technique to generate samples and ensure a fair representation of all categories. Finally, the data was divided into a training set (80%) and a test set (20%).
- Implemented using TensorFlow/Keras, following a structured training pipeline.
- Loss function: The categorical cross-entropy loss function was used for the multi-class classification

task, due to its high suitability for handling the five distinct sleep stages.

- Optimizer: A learning rate scheduler was incorporated to dynamically adjust the learning rate in response to training progress. The Adam optimizer was also used due to its effectiveness in managing sparse gradients.
- Hyperparameter adjustment is critical, as it has a direct impact on model accuracy, generalization, and other performance metrics. A grid search approach was applied here.
- Each model was trained for a predefined number of epochs (50).
- An early stopping technique was used, set with patience=4, to monitor the loss value. This means that over four consecutive training cycles, the training process stops if the validation loss does not improve. The primary goal of early stopping is to prevent overfitting by monitoring the model's performance on the validation dataset during training..

Once training has completed, the model is evaluated using a range of metrics which offer a holistic evaluation of both overall and per-class performance:
Classification Report: for each classes ,have precision, recall, and F1-score along with micro, macro, and weighted averages
Confusion Matrix: To shows how many samples were classified correctly and incorrect for each class
Accuracy: The accuracy of training and validation.
ROC Curves and AUC: To dives information of how well the model can distinguish between classes at different threshold settings.

4. Implementation and Results Discussion

Selecting the appropriate architecture becomes essential when analyzing temporal signals, such as audio, EEG, or other sequential data, since feature extraction and temporal information retention must be balanced. This is where carefully considered architectural design is useful, guaranteeing: enhanced CNN feature extraction that uses convolutions to find spatial patterns. Maintaining temporal context involves examining the connections between various time points using LSTM. Increasing stability and decreasing computational cost through the use of techniques like pooling layers, dropout, and batch normalization. Consequently, a careful selection of architecture is one of the most important elements that determines a model's success since it is founded on current research and real-world experiences to guarantee precise and effective performance on the intended objectives.

4.1. Model Performance

Given the nature of our dataset, we conducted extensive experiments to overcome overfitting and the difficulty of learning the network[35].

We conducted extensive experiments to determine the most appropriate architecture for LSTM, CNN, and Hybrid CNN,LSTM models to suit the nature of the dataset for classifying sleep stages from EEG signals. We chose an architecture that would balance complexity and performance, overcome overfitting and the difficulty of learning the network, and achieve a stable final implementation. The experiments we conducted included:

1. Determining the appropriate number of layers in each model.
2. Determining the appropriate number of neurons for each layer.
3. After determining the appropriate number of layers and neurons in each model, we began testing the importance of the dense layer and the number of units it should occupy.
4. Determining the appropriate dropout value.

4.1.2 LSTM

Several experiments were conducted on the LSTM model architecture.

1. The first of these experiments was to determine the appropriate number of layers and the number of neurons in each layer of the model for this type of problem. We found that the two-layer model performed best. Furthermore, the two-layer LSTM model, with the first layer containing 100 neurons and the second layer containing 50 neurons, was the best and most stable among the other model architecture options tested. Table (3) shows the overall experiments conducted to determine the number of layers and neurons in each layer, to arrive at the best LSTM architecture in terms of performance.This number of LSTM layers and number of units is to control most of the features in the temporal EEG signal received via the first layer, and then pass the most salient features from the first layer to the second LSTM layer. Furthermore, this arrangement avoids overfitting.[36][37].

2. After determining the most appropriate number of layers and their sizes :(128 and 64), then come the test of dense layer in terms of number, location, and number of units. Table (4) shows the results of these tests. We note from the table that a dense layer with (200 units) is ideal for achieving stability and increasing performance . This first dense layer focuses on the features extracted from the previous LSTM layers to achieve the best prediction. The second dense layer, whose output is 5 values, is combined with the SoftMax function to perform classification tasks. SoftMax is a function that returns a probability vector, which predicts the probability of each class x_i [38][39] (the class here represents one of the five sleep stages).

3. Various values of dropout were tested, as dropout is actually a form of regularization intended to help prevent overfitting by increasing test accuracy, possibly at the expense of training accuracy. This avoids overfitting and improves implementation stability. The implementation of the dropout

algorithm relies on randomly dropping neurons during training to avoid co-adaptation of feature detectors, which leads to random variables [40][41]. For each minibatch in the training set, dropout layers randomly separate the inputs from the previous layer to the next in the network architecture with probability p [42].

After Dense Layer 1, we applied a dropout layer to reduce overfitting and help the LSTM model generalize. The dropout value (0.3) was found to be the most appropriate, which was determined by testing a range of values from dropout = 0.2 to dropout = 0.5, and we found that implementation stability was achieved with dropout = 0.3. Table (5) shows the experiments conducted to determine the most appropriate dropout value.

4. Batch size is an important measure of model loss. This parameter represents the number of training samples that will be used during training to perform a single update to the network parameters. Batch size affects model training in terms of the time required for convergence and the amount of overfitting[43]. We studied the effect of different batch sizes on the performance of the LSTM network. Table (6) shows all the experiments we performed in this test.

According to the results shown in Tables 3 to 6, the LSTM architecture shown in Table 7 can be considered a successful and reliable model for classifying different sleep stages.

Table 3:The results efficiency with different LSTM layers

No. of layer	No. of units	Training accuracy	validation accuracy	Training loss	Validation loss
Two	(64, 32)	0.8044	0.7895	0.4958	0.5344
Two	(128, 64)	0.8569	0.8273	0.3617	0.4592
three	(128, 64, 32)	0.8519	0.8260	0.3739	0.4594

Table 4: The results efficiency with different Dense units values of LSTM(128 ,64)

Dense units value	Training accuracy	validation accuracy	Training loss	Validation loss
100	0.8170	0.8060	0.4682	0.4953
200	0.8492	0.8274	0.3923	0.4524
300	0.8421	0.8222	0.4063	0.4571

Table 5: The results efficiency with different Dropout values of LSTM(128 ,64)

Dropout value	Training accuracy	validation accuracy	Training loss	Validation loss
0.2	0.8568	0.8303	0.3707	0.4523
0.3	0.8445	0.8258	0.4033	0.4531
0.4	0.8225	0.8074	0.4583	0.4972

Table 6: The results efficiency with and without Batch normalization LSTM(128 ,64)

Batch normalization	Training accuracy	validation accuracy	Training loss	Validation loss
Without	0.8569	0.8273	0.3617	0.4592
With	0.8644	0.8301	0.3540	0.4524

Table (7) : The sequential Model of LSTM

Layer (type)	Output Shape	Parameters
lstm (LSTM)	(None, 3000, 128)	66,560
lstm_1 (LSTM)	(None, 64)	49,408
dense (Dense)	(None, 200)	13,000
batch_normalization	(None, 200)	800
dropout (0.2)	(None, 200)	0
dense_1 (Dense)	(None, 5)	1,005

4.1.2 CNN

We first tested the appropriate architecture in terms of the number of layers and neural units per layer. In CNN-EEG application areas such as sleep, the average number of CNN layers is relatively high. Some studies use CNN architectures with up to 13 layers, while CNN architectures with 3, 4, 6, 5, and 8 layers are typically used, depending on the nature of the data [44].

From our experiments shown in Table 8, we found that the best CNN architecture performed best with 3 layers

containing 32, 64, and 128 units, respectively. We note that the higher the number of layers and units, the higher the training and validation accuracy. However, some models, such as 64, 128, and 256, suffer from increased validation loss, which may indicate potential overfitting.

Based on our findings regarding the best architecture (32, 64, and 128 layers), we conducted several experiments to improve the results and model stability, which can be described as follows:

1. We conducted an experiment on the CNN architecture by placing a batch normalization layer between the convolution layer and the max pooling layer to stabilize training and performance. We also conducted a test without a batch normalization layer. The reason for this experiment is that the EEG signal during sleep is inherently unstable and variable. The batch normalization layer allows for much higher learning rates, while adopting a less conservative approach to initialization. It also acts as a regularizer, learning a more efficient model and also enabling faster neural networks [45][46].

   The results obtained are shown in Table (9), where we observed an improvement in performance with batch normalization. The network was more stable, with a validation accuracy of 0.8736 and a validation loss of 0.3784. Without batch normalization, we observed an increase in the validation loss (0.4060), indicating overfitting..
2. Max pooling is an important layer in CNNs, as it reduces the dimensionality of features while preserving important information. Max pooling can be applied to downsample the convolutional output bands, reducing variance. This feature is very effective with variable EEG signals [47]. In this study, we tested the model without max pooling and with max pooling (with different pooling sizes tested), as shown in Table 10. We found that a pooling size of 2 is an appropriate window size to halve the number of parameters, achieve balance, and preserve information.
3. It is also useful to apply the dropout property using CNN. The dropout property is placed at the end of each Conv layer to prevent overfitting, preserve information, and increase test accuracy. We found through experimentation that the best value is dropout = 0.3. Values higher than 0.3 showed an increase in validation loss; see details in Table 11.
4. The best performance in terms of validation accuracy was achieved at 300 units in the dense layer, which is close to the value at 200 units (see Table 12). The dense network with 200 units was chosen because it achieves the optimal balance between training and generalization. The reason is that in the 300-unit network, the difference between training accuracy and validation accuracy is approximately 0.0514, which is larger than the difference at 200 units (0.0395), indicating the possibility of overfitting in the 300-unit case. The same applies to the training loss and validation loss in both cases, as we note that the validation loss increases slightly in the 300-unit case. Furthermore, the performance of the 100-unit dense layer is poor.

After completing various extensive experiments to arrive at the most suitable CNN architecture for the problem of classifying sleep stages from EGG signals, we show in Table (13) the architecture that has the best stability and learning performance.

Table 8:The results efficiency with different CNN layers

No. of layer	No. of units	Training accuracy	validation accuracy	Training loss	Validation loss
Two	(32, 64)128	0.9240	0.8518	0.1990	0.4748
Two	(64, 128)128	0.9170	0.8536	0.2076	0.4759
Three	(32, 64, 128)64	0.9131	0.8736	0.2210	0.3784
Three	(64, 128, 256)128	0.8799	0.8631	0.2943	0.4146
Four	(16,32,64,128)64	0.8954	0.8734	0.2677	0.4077
Four	(32, 64, 128, 256)128	0.9080	0.8700	0.2307	0.4305

Table 9: The results with and without Batch normalization CNN model

Batch normalization	Training accuracy	validation accuracy	Training loss	Validation loss
With	0.9131	0.8736	0.2210	0.3784
Without	0.9208	0.8719	0.2056	0.4060

Table 10: The results with different **Pooling size value** CNN model

Pooling size	Training accuracy	validation accuracy	Training loss	Validation loss
2	0.9119	0.8735	0.2192	0.3893
3	0.8904	0.8666	0.2831	0.4048
4	0.8630	0.8547	0.3513	0.4112

Table 11: The results with different Dropout values of CNN model

Dropout value	Training accuracy	validation accuracy	Training loss	Validation loss
0.2	0.9178	0.8682	0.2006	0.4452
0.3	0.9131	0.8736	0.2210	0.3784
0,4	0.9005	0.8702	0.2546	0.3754

Table 12: The results efficiency with different Dense units values of CNN MODEL

Dense units value	Training accuracy	validation accuracy	Training loss	Validation loss
100	0.8821	0.8618	: 0.2890	0.4121
200	0.9131	0.8736	0.2210	0.3839
300	0.9252	0.8788	0.1924	0.3843

Table 13 : The sequential Model of CNN

Layer (type)	Output Shape	No. Parameters
conv1d (Conv1D)	(None, 2998, 32)	128
batch_normalization	(None, 2998, 32)	128
max_pooling1d (2)	(None, 1499, 32)	0
dropout (0.3)	(None, 1499, 32)	0
conv1d_1 (Conv1D)	(None, 1497, 64)	6,208
batch_normalization_1	(None, 1497, 64)	256
max_pooling1d_1 (2)	(None, 748, 64)	0
dropout_1 (0.3t)	(None, 748, 64)	0
conv1d_2 (Conv1D)	(None, 746, 128)	24,704
batch_normalization_2	(None, 746, 128)	512
max_pooling1d_2 (2)	(None, 373, 128)	0
dropout_2 (0.3t)	(None, 373, 128)	0
flatten (Flatten)	(None, 47744)	0
dense (200)	(None, 200)	9,549,000
dropout_3 (0.3t)	(None, 200)	0
dense_1 (Dense)	(None, 5)	1,005

4.1.3 Hybrid CNN and LSTM

The following describes the experiments conducted to test the hybrid CNN-LSTM architecture, which naturally involves combining a set of features from each model to achieve the goal of optimal classification of sleep EGG data. This combination process relied on the best CNN and LSTM architectures achieved based on the experiments mentioned above. This involved adopting a two-layer LSTM, with the first layer containing (128, 64) units. The CNN model architecture (32, 664, 128) was also used, with the rest of the details held constant for each.

The first part is tested the CNN_LSTM architecture. The order of this model is such that CNN layers come first to extract features from signals. The output of the CNN features is then transferred to the LSTM layers, which analyze the input data to extract the final features. Here, we tried to discover the effect of extracting spatial features first and then passing them to extract temporal features.

In the second part, we tested the LSTM_CNN architecture: In this part, we tested the effect of extracting temporal features first and passing them to the second layer to explore spatial features. We conducted our experiments on the LSTM_CNN architecture, whose order is the opposite of that of CNN_LSTM. The results shown in Table (13) indicate that the LSTM_CNN architecture performs best in terms of training accuracy, validation accuracy, and training loss. Although the validation loss is slightly higher than the training loss, indicating that overfitting is starting to occur, the LSTM_CNN architecture performs best because extracting temporal and then spatial features is the most successful sequence.

LSTM_CNN architecture can be considered a successful model for classifying different sleep stages. Table (14) illustrates the architecture adopted for this model.

Table 13: The results accuracy of hybrid **Architectures**

Hybride Architecture	Training accuracy	validation accuracy	Training loss	Validation loss
CNN_LSTM	0.8583	0.8436	0.3577	0.4147
LSTM_CNN	0.9111	0.8813	0.2293	0.4218

Table 14:: The sequential Model of hybrid LSTM-CNN

Type of Layer	Output Shape	No. Parameters
lstm (LSTM)	**(None, 3000, 128)**	**66,560**
lstm_1 (LSTM)	**(None, 3000, 64)**	**49,408**
conv1d (Conv1D)	**(None, 2998, 32)**	**6,176**
batch_normalization	**(None, 2998, 32)**	**128**
max_pooling1d (MaxPooling1D)	**(None, 2997, 32)**	**0**
dropout (Dropout)	**(None, 2997, 32)**	**0**
conv1d_1 (Conv1D)	**(None, 2995, 64)**	**6,208**
batch_normalization_1	**(None, 2995, 64)**	**256**
max_pooling1d_1 (MaxPooling1D)	**(None, 1497, 64)**	**0**
dropout_1 (Dropout)	**(None, 1497, 64)**	**0**
conv1d_2 (Conv1D)	**(None, 1495, 128)**	**24,704**
batch_normalization_2	**(None, 1495, 128)**	**512**
max_pooling1d_2 (MaxPooling1D)	**(None, 747, 128)**	**0**
dropout_2 (Dropout)	**(None, 747, 128)**	**0**
flatten (Flatten)	**(None, 95616)**	**0**
dropout_3 (Dropout)	**(None, 95616)**	**0**
dense (Dense)	**(None, 200)**	**19,123,400**
dropout_4 (Dropout)	**(None, 200)**	**0**
dense_1 (Dense)	**(None, 5)**	**1,005**

.4.2. Comparative Analysis

Based on the results presented in the previous paragraphs, and based on well-known performance metrics, in this section we review the key features of each of the three models—LSTM, CNN, and the CNN-LSTM hybrid model—in terms of performance and efficiency. We know that each model has the ability to recognize and process specific signal characteristics:

LSTM Model: The LSTM model demonstrated good performance in recognizing sleep classes, especially with the two-layer architecture, the first with 128 units and the second with 64 units. From the evaluation performance graphs in Figure (3), we note that the best training and completion accuracies are 0.8492 and 0.83, respectively, demonstrating the model's good learning ability.From the training and loss curves, we note that the model is stable and does not suffer from overfitting or underfitting.The confusion matrix results indicate good classification for all classes. Although some classes, particularly Class 1, show confusion with Class 2 by 1513 and Class 3 by 1631, the classification results are considered good and acceptable.ROC curves show good results for each class. Even for Class 1, where AUC is 0.93, which is high..

Although the input data consisted of time series or sequence-like structures, the CNN model had no difficulty in managing temporal relationships and extracting the most important spatial information. It outperformed the LSTM in recognizing unique aspects of the input signal, as shown in the results in Figure 4. The training accuracy reached 0.9131% and the validation accuracy reached 0.8736%, which are good values and indicate that the model does not suffer from significant overfitting. The training loss reached 0.22, and the validation loss reached 0.383, these values indicate the model's stability.And the confusion matrix results, based on the results on the diagonal, indicate that the model classified correctly , where the Class 3 performed well with 13,258 correct cases and 319 incorrect cases, and Some cases, such as Class 1 and Class 2, exhibit confusion, but the overall results are good. From the information on the ROC curves, we can also conclude that the CNN model has an impressive ability to distinguish between different classes.

The hybrid CNN/LSTM model demonstrated significant superiority over both the CNN and LSTM models individually on most evaluation metrics. The hybrid LSTM_CNN model (LSTM first, then CNN) achieved the highest results among all tested models. From the evaluation performance graphs in Figure (5), we observe the following: The model's accuracy in distinguishing between the learning and validation phases is good. We note that the training and validation curves increase with each epoch, indicating

that the model does not suffer from overfitting. The model's loss curves also reflect a good learning balance, with no significant difference between the training and validation curves. The confusion matrix results indicate good classification for all classes,the Class 3 appears to be the highest-ranking class, with 13,210 correct instances out of 13,600, while the remaining classes performed well. The ROC curves for each class show good results, indicating a high classification ability for the model, AUC values range between 0.96 and 1.00, demonstrating the model's robustness in classifying classes. Class 3 is the best-ranking class, with an AUC of 1.00.

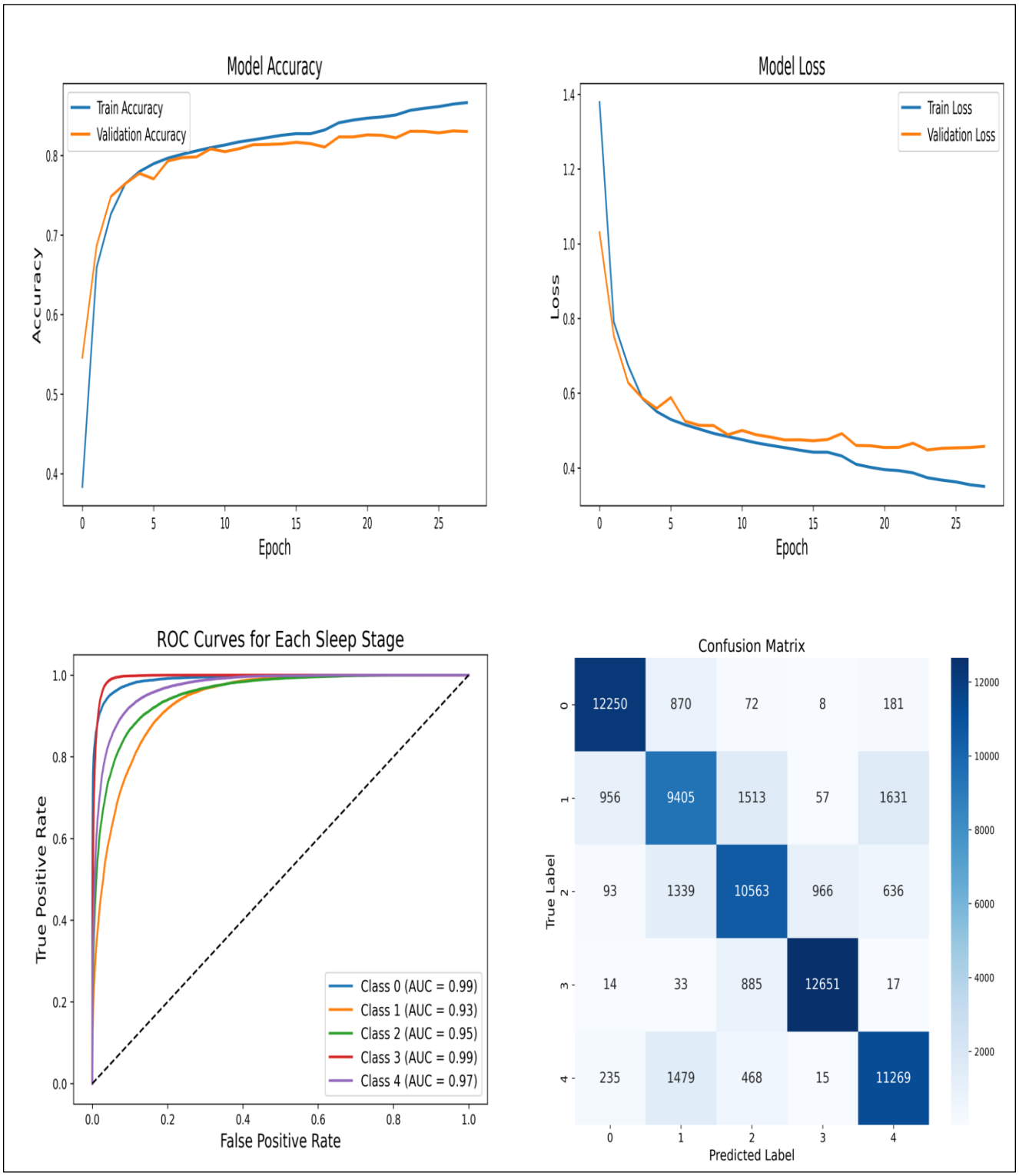


Figure (3) : The evaluation curves and results of LSTM MODEL

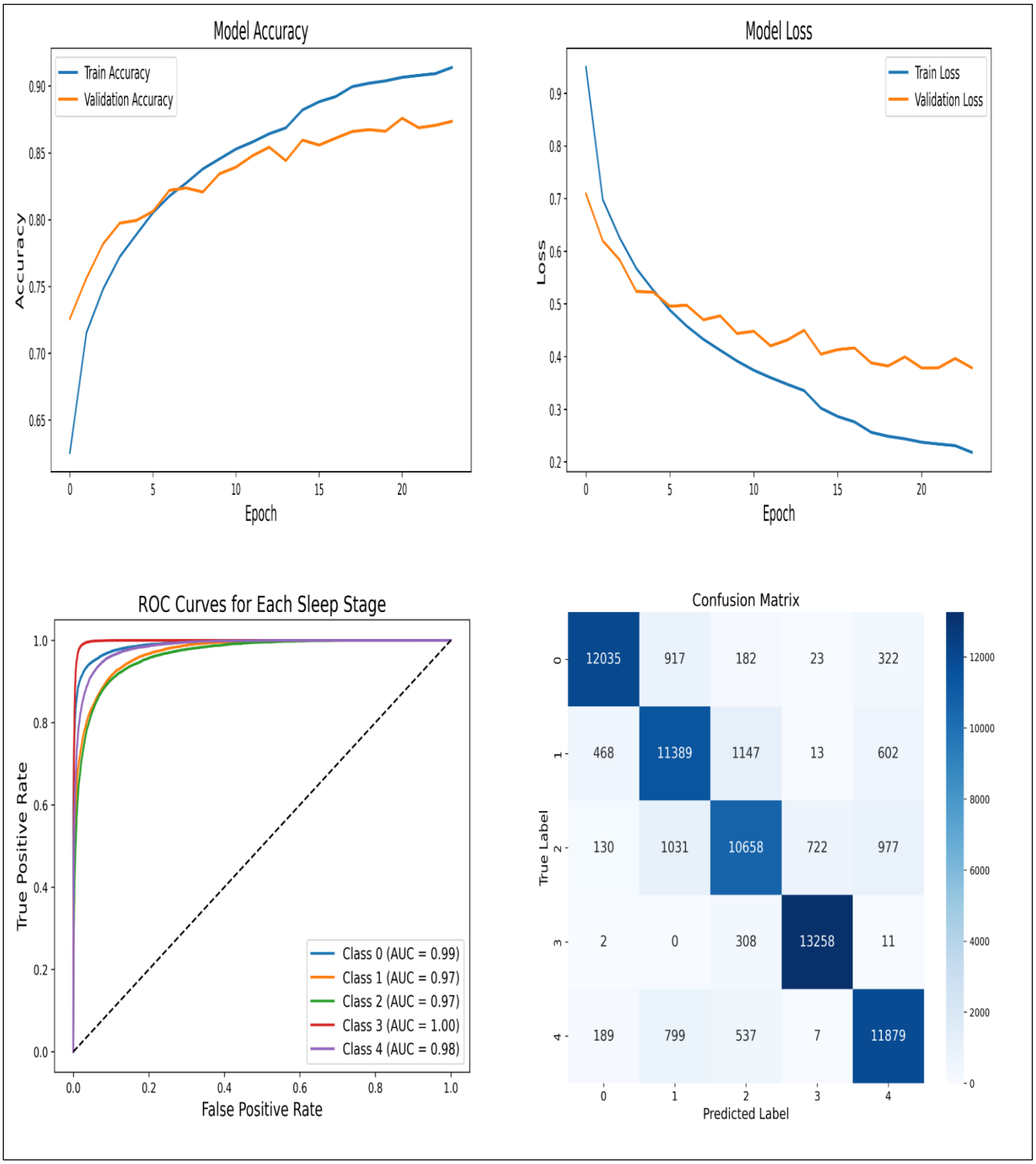


Figure (4) : The evaluation curves and results of CNN MODEL

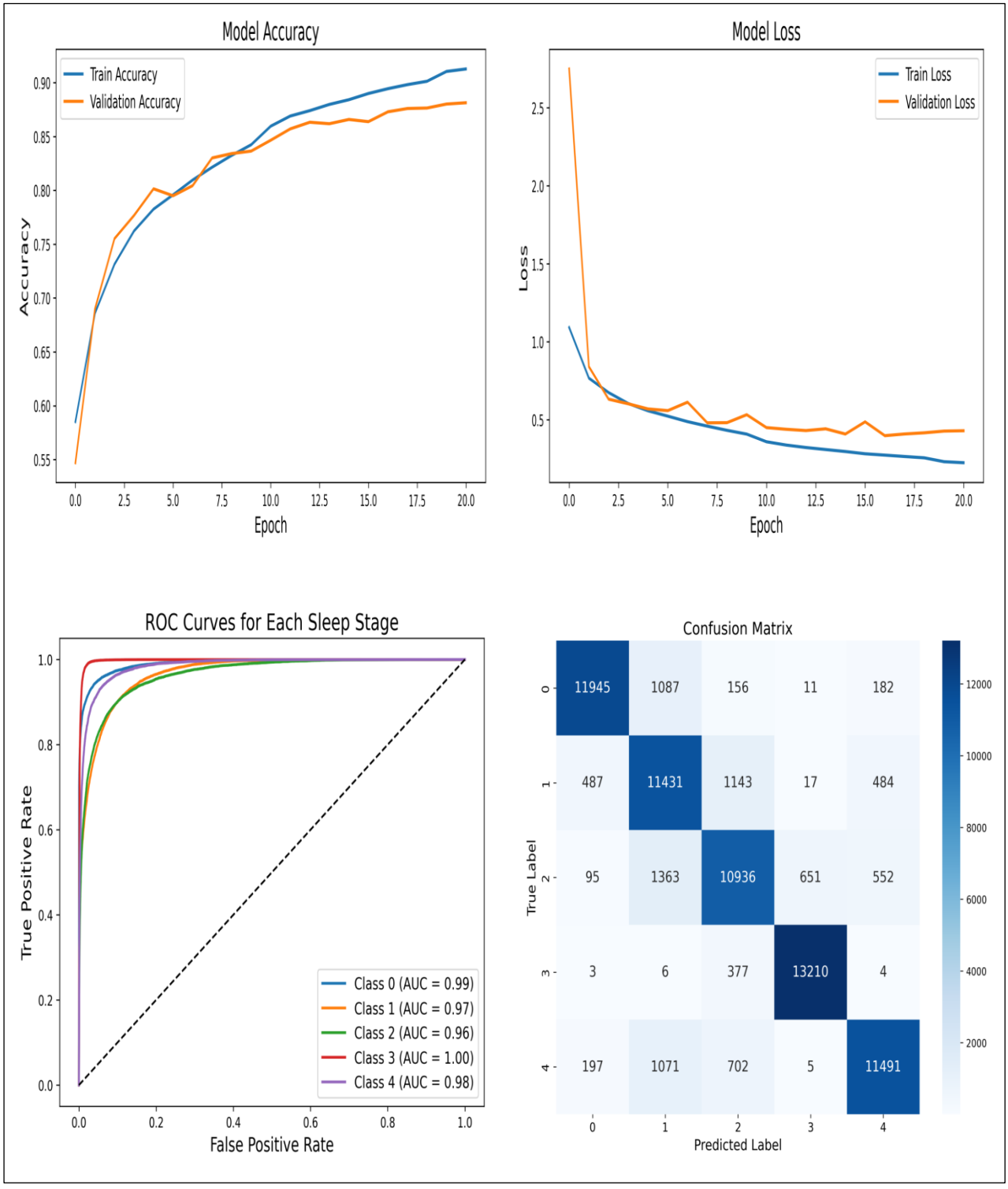


Figure (5) : The evaluation curves and results of Hybrid LSTM+ CNN MODEL

5. Conclusion and Recommendations

This section relies on extensive experiments and results obtained from performance metrics such as accuracy, loss, confusion matrix, and ROC curves, on LSTM, CNN, and hybrid CNN/LSTM models. From the summary of the results in Table (15), we note that all models demonstrated strong performance across the four performance metrics. Stability and learning demonstrate that these models are well-suited to handling and classifying sleep signals. All models provided very similar results in terms of accuracy and loss, demonstrating the strength of each model in this task.

Table(15) :Final table of Computational Efficiency of models

Feature	LSTM	CNN	LSTM + CNN
Training accuracy	0.8644	0.9131	0.9111
Validation Accuracy	0.8301	0.8736	0.8813
Training loss	0.3540	0.2210	0.2293
Validation Loss	0.4577	0.3893	0.4218
F1 score	0.83	0.87	0.87
Number of Parameters	130,773	9,709,033	19,277,249
Total Training Time	19559.48 sec	4156.67 sec	18922.81 sec
Average Training Time	698.24 sec	148.14 sec	900.67 sec
Average Inference Time per Sample	0.003884 sec	0.000109 sec	0.003985 sec

However, in terms of comparison between these models:

1. The LSTM-CNN model achieved the best balance between training accuracy (0.9111) and validation accuracy (0.8813), with a relatively low loss. This is because, from a scientific perspective, there is a benefit to combining temporal and spatial features, especially in the LSTM-CNN order, which significantly enhanced classification capabilities, the temporal signals are processed first, and then spatial information is extracted from them.Although the LSTM-CNN model outperforms the others and is a good choice in terms of balancing accuracy and temporal and spatial representation efficiency, we recommend using any of the proposed models CNN or LSTM, depending on the available capabilities and the purpose of the study , as they also demonstrated excellent results, as shown in Table 15.
2. Number of parameters: We note that LSTM is the lightest, with only 130,000 parameters, compared to CNN, which has about 9 million, and LSTM+CNN, which has about 19 million.
3. In terms of training time, CNN demonstrated remarkable speed, averaging only 148 seconds per cycle, compared to LSTM (698 sec.) and LSTM+CNN (900 sec.). Therefore, CNN is suitable for such datasets in terms of speed.
4. In terms of inference time, CNN outperforms, with an average of only 0.0001 sec. per sample. On the other hand, LSTM and LSTM+CNN are slower, about 0.004 sec.

References

[1] Ramar, K., Malhotra, R. K., Carden, K. A., Martin, J. L., Abbasi-Feinberg, F., Aurora, R. N., Kapur, V. K., Olson, E. J., Rosen, C. L., Rowley, J. A., Shelgikar, A. V., & Trotti, L. M., "Sleep is essential to health: An American Academy of Sleep Medicine position statement. Journal of Clinical Sleep Medicine", 17(10), 2115–2119, (2021). https://doi.org/10.5664/jcsm.9476

[2] A. Crivello, P. Barsocchi, M. Girolami, and F. Palumbo,**“**The Meaning of Sleep Quality: A Survey of Available Technologies**,”**,*IEEE Access*, vol. 7, pp. 167374–167390, (2019). . https://doi.org/10.1109/access.2019.2953835

[3] S. M. M. de Mooij, T. F. Blanken, R. P. P. P. Grasman, J. R. Ramautar, E. J. W. Van Someren, and H. L. J. van der Maas,**“**Dynamics of sleep: Exploring critical transitions and early warning signals**”**,Computer Methods and Programs in Biomedicine, vol. 196, Sep. 2020, Art. no. 105448.https://doi.org/10.1016/j.cmpb.2020.105448

[4] A. K. Patel, V. Reddy, and J. F. Araujo, “Physiology, sleep stages", StatPearls, StatPearls Publishing, 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK526132/

[5] R. Cox and J. Fell, "Analyzing human sleep EEG: A methodological primer with code implementation", Sleep Medicine Reviews, vol. 50, Art. no. 101353, 2020. https://doi.org/10.1016/j.smrv.2020.101353

[6] I. G. Campbell, “EEG recording and analysis for sleep research” ,Current Protocols in Neuroscience, vol. 49, no. 1, pp. 10.2.1–10.2.19, Oct,2009.https://doi.org/10.1002/0471142301.ns1002s49

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory", Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. ,1997. https://doi.org/10.1162/neco.1997.9.8.1735

[8] C. Li, Y. Qi, X. Ding, J. Zhao, T. Sang, and M. Lee,"A deep learning method approach for sleep stage classification with EEG spectrogram",International Journal of Environmental Research and Public Health, vol. 19,no.10,p.6322,May,2022.https://doi.org/10.3390/ijerph19106322

[9] O. Tsinalis, P. M. Matthews, and Y. Guo, "Automatic sleep stage scoring using time–frequency analysis and stacked sparse autoencoders", Annals of Biomedical Engineering, vol. 44, no. 5, pp. 1587–1597, May 2016. https://doi.org/10.1007/s10439-015-1444-y

[10] S. Sasai, T. Koike , S. K. Sugawara, S. Okazaki, K. Watanabe, N. Sadato and Iguchi, Y. ,"Network-wide reorganization of procedural memory during NREM sleep revealed by fMRI". eLife, 6, e24987, 2017. https://doi.org/10.7554/eLife.24987

[11] M. Nicola, U. R. Acharya, and F. Marcelloni,"Cascaded LSTM recurrent neural network for automated sleep stage classification using single-channel EEG signals",Computers in Biology and Medicine, vol. 106, pp. 71–81, Jan. 2019.https://doi.org/10.1016/j.compbiomed.2019.01.013

[12] A. Salamatian and A. Khadem, "Automatic sleep stage classification using 1D convolutional neural network", Frontiers in Biomedical Technologies, vol. 7, no. 3, 2022. http://dx.doi.org/10.18502/fbt.v7i3.4616

[13] D. Zhao, R. Jiang, M. Feng, J. Yang, Y. Wang, X. Hou, and X. Wang, "A deep learning algorithm based on 1D CNN-LSTM for automatic sleep staging," Technol. Health Care, 2022. DOI: 10.3233/THC-212847.

[14] A. Luic, A. Takashi, and M. Molinas, "EEG-based 5- and 2-class CNN for Sleep Stage Classification," IFAC-PapersOnLine, vol. 56, no. 2, pp. 3211–3216, 2023. DOI: 10.1016/j.ifacol.2023.10.1458.

[15] E. Enes and O. Seral, "CoSleepNet: Automated sleep staging using a hybrid CNN-LSTM network on imbalanced EEG-EOG datasets," Biomed. Signal Process. Control, vol. 80, Feb. 2023, Art. no. 104299. DOI: 10.1016/j.bspc.2022.104299.

[16] R. Kotla, N. Polavarapu, and P. Safa, "A CNN-LSTM Model for Sleep Stage Scoring Using EEG Signals," in Proc. Int. Conf. Commun., Circuits, Syst. (IC3S), 2023. DOI: 10.1109/IC3S57698.2023.1016917.

[17] A. Supratak, H. Dong, C. Wu, and Y. Guo, "DeepSleepNet: A model for automatic sleep stage scoring based on raw single-channel EEG," IEEE Trans. Neural Syst. Rehabil. Eng., 2017. DOI: 10.1109/TNSRE.2017.2721116.

[18] O. Tsinalis, P. M. Matthews, Y. Guo, and S. Zafeiriou, "Automatic sleep stage scoring with single-channel EEG using convolutional neural networks," arXiv preprint, arXiv:1610.01683, 2016. DOI: 10.48550/arXiv.1610.01683.

[19] D. T. Pham and R. Mouček, "Automatic Sleep Stage Classification by CNN-Transformer-LSTM using single-channel EEG signal," in Proc. IEEE Int. Conf. Bioinformatics Biomed. (BIBM), Shanghai, China, Dec. 2023. DOI: 10.1109/BIBM58861.2023.10385687.

[20] A. A. Casciola, J. Bossert, and N. Elbuluk, "A Deep Learning Strategy for Automatic Sleep Staging Based on Two-Channel EEG Headband Data," Sensors, vol. 21, no. 10, p. 3316, 2021. DOI: 10.3390/s21103316.

[21] T. Li, B. Zhang, H. Lv, S. Hu, Z. Xu, and Y. Tuergong, "CAttSleepNet: Automatic End-to-End Sleep Staging Using Attention-Based Deep Neural Networks on Single-Channel EEG," Int. J. Environ. Res. Public Health, vol. 19, no. 9, p. 5199, 2022. DOI: 10.3390/ijerph19095199.

[22] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," Artif. Intell. Rev., vol. 53, no. 1, 2020. DOI: 10.1007/s10462-020-09838-1.

[23] S. M. Al-Selwi, M. F. Hassan, S. J. Abdulkadir, A. Muneer, E. H. Sumiea, A. Alqushaibi, and M. G. Ragab,"RNN-LSTM: From applications to modeling techniques and beyond—Systematic review," J. King Saud Univ.—Comput. Inf. Sci., 2024. DOI: 10.1016/j.jksuci.2024.102068.

[24] X. Wei, L. Zhang, H. Yang, L. Zhang, and Y. Yao,"Machine learning for pore-water pressure time-series prediction: Application of recurrent neural networks," Geosci. Frontiers, vol. 12, no. 1, pp. 453–467,Jan. 2021. DOI: 10.1016/j.gsf.2020.04.011.

[25] A. Thakkar and K. Chaudhari, "A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions," Expert Syst. Appl., vol. 177,p.114800,2021.DOI:http://dx.doi.org/10.1016/J.ESWA.2021.114800.

[26] J. M. Dahr and A. S. Gaafar, "Performance evaluation of the convolutional neural networks for object identification using RGB and binary images," Informatica, vol. 48, no. 21, pp. 177–188, 2024. DOI: 10.31449/inf.v48i21.6568.

[27] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," J. Big Data, vol. 8, art. no. 53, 2021. DOI: 10.1186/s40537-021-00444-8.

[28] F. Sultana, A. Sufian, and P. Dutta, "Advancements in Image Classification using Convolutional Neural Network," in Proc. 2018 4th Int. Conf. Res. Comput. Intell. Commun. Netw**. (ICRCICN)**, Kolkata, India, Nov. 2018, pp. 122–129. doi: 10.1109/ICRCICN.2018.8718718

[29] S. S. Nisha and M. N. Meeral, "Applications of Deep Learning in Biomedical Engineering," in *Handbook of Deep Learning in Biomedical Engineering: Techniques and Applications*, Academic Press, 2021, pp. 245–270, doi: 10.1016/B978-0-12-823014-5.00008-9.

[30] T. Kessler, G. Dorian, and J. H. Mack, "Application of a Rectified Linear Unit (ReLU) based artificial neural network to cetane number predictions," in Proc. ASME 2017 Int. Combust. Engine Div. Fall Tech. Conf., Oct. 2017. DOI: 10.1115/ICEF2017-3614.

[31] M. Blot, M. Cord, and N. Thome, "Max-min convolutional neural networks for image classification," in Proc. 2016 IEEE Int. Conf. Image Process. (ICIP), Phoenix, AZ, USA, Sep. 2016,pp. 3678–3682.doi: 10.1109/ICIP.2016.7533046

[32] T. Zhou, C. XiaoYu, L. Huiling, and Y. XinYu, "Pooling operations in deep learning: From 'invariable' to 'variable'," Biomed. Res. Int., vol.

2022, art. no. 4067581, 2022. DOI: 10.1155/2022/4067581.

[33] A. Ghosh, F. Sultana, A. Sufian, and A. Chakrabarti, "Fundamental concepts of convolutional neural network," in Intell. Syst. Ref. Libr., Springer, 2020, pp. 1–20. DOI: 10.1007/978-3-030-32644-9_36.

[34] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," IEEE Trans. Pattern Anal. Mach. Intell.,2017. DOI: 10.1109/TPAMI.2016.2599174.

[35] I. Priyadarshini and C. Cotton, "A novel LSTM–CNN–grid search-based deep neural network for sentiment analysis," J. Supercomput., vol. 77, pp. 13911–13932, 2021. DOI: 10.1007/s11227-021-03838-w.

[36] Y. Li, "K-LSTM-ECM Model for Predicting Poverty Alleviation Impacts of Digital Financial Inclusion," Informatica, vol. 49, no. 17, 2022, doi: 10.31449/inf.v49i17.7621..

[37] B. H. D. Koh and W. Woo, "Multi-view temporal ensemble for classification of non-stationary signals," IEEE Access, vol. 7, pp. 34189–34202, 2019. DOI: 10.1109/ACCESS.2019.2903571.

[38] M. Franke and J. Degen, "The softmax function: Properties, motivation, and interpretation," Unpublished, 2023 https://doi.org/10.31234/osf.io/vsw47 .

[39] R. C. Staudemeyer and E. Rothstein Morris, "Understanding LSTM: A tutorial into long short-term memory recurrent neural networks," *arXiv*, vol. 1909.09586v1, 2019. DOI: 10.48550/arXiv.1909.09586.

[40] Staudemeyer, R. C., & Rothstein Morris, E. (2019). Understanding LSTM: A Tutorial into Long Short-Term Memory Recurrent Neural Networks. arXiv, 1909.09586v1. https://arxiv.org/abs/1909.09586v1

[41] P. Baldi and P. Sadowski, "The dropout learning algorithm," Artif. Intell., vol. 210, pp. 78–122, 2014. DOI: 10.1016/j.artint.2014.02.004.

[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," J. Mach. Learn. Res., vol. 15, no. 56, pp. 1929–1958, Jun. 2014. doi:10.5555/2627435.2670313

[43] P. M. Radiuk, "Impact of training set batch size on the performance of convolutional neural networks for diverse datasets," Inf. Technol. Manag. Sci., vol. 20, pp. 20–24, 2017. DOI: 10.1515/itms-2017-0003.

[44] S. Rajwal and S. Aggarwal, "Convolutional neural network-based EEG signal analysis: A systematic review," Arch. Comput. Methods Eng., vol. 30, pp. 3585–3615, 2023. DOI: 10.1007/s11831-023-09920-1.

[45] M. Hasani, "An empirical study on position of the batch normalization layer in convolutional neural networks," in Proc. 2019 5th Iranian Conf. Signal Process. Intell. Syst. (ICSPIS), 2019, pp. 1–5. DOI: 10.1109/ICSPIS48872.2019.9066113.

[46] S. Lagnaoui, Z. En-Naimani, and K. Haddouch, "The effect of normalization and batch normalization layers in CNNs models: Application to plant disease classifications," in Proc. 6th Int. Conf. Big Data Internet Things, Springer, 2023, pp. 250–262. DOI: 10.1007/978-3-031-28387-1_22.

[47] H. Gholamalinezhad and H. Khosravi, "Pooling methods in deep neural networks, a review," CoRR, vol. abs/2009.07485, 2020. doi:10.48550/arXiv.2009.07485.

.

Deep Reinforcement Learning for Personalized Route Planning in Agricultural Tourism: A DDPG and Genetic Algorithm Approach

Xiaohang Ma[1], Zhanyong Wu[1*], Ling Hu[2*]
[1]School of Economics and management, Handan University, Han'dan 056005, Hebei, China
[2]Xinyang Vocational and Technical College, Xinyang 464000, Henan, China
E-mail: wu_zhanyong@outlook.com; huling163@hotmail.com
[*]Corresponding author

Keywords: agritourism, artificial intelligence, route planning

Received: November 19, 2024

This research aims to develop an artificial intelligence route planning algorithm for agricultural tourism to provide a flexible, personalized, environmentally friendly, interactive and educational travel experience. By analyzing the characteristics and needs of agricultural tourism, we built a simulation environment to simulate the real agricultural tourism scene, and collected abundant data, including geographical location, traffic conditions, scenic spot information and tourist evaluation. In the data preprocessing stage, we clean, standardize, feature engineer and integrate the data to improve the accuracy and practicability of the algorithm. The algorithm design follows the principles of flexibility, individuation, environmental protection, interaction and education. We use deep reinforcement learning (DRL), especially the deep deterministic strategy gradient (DDPG) algorithm, to deal with the route planning problem with continuous action space. In order to improve the learning efficiency and performance of the model, we introduce genetic algorithm to optimize the hyper-parameters. In terms of personalized recommendation mechanism, we customize unique travel experience for each visitor by analyzing user's historical behavior, preferences, time constraints and other information. Combined with the route planning strategy of deep reinforcement learning, the personalized recommendation mechanism not only improves the user experience, but also promotes the effective allocation and utilization of resources. The performance of the algorithm is verified by simulation and optimization. The experimental results show that the proposed algorithm has made significant improvements in many aspects: the computing time is reduced from 250 seconds to 120 seconds, which is reduced by 52%, the user satisfaction is improved from 3.5 to 4.0, which is improved by 14.3%, the preference matching degree of the family parent-child group, the youth adventure group and the middle-aged and elderly leisure group is improved by 23.1%, 21.4% and 36.4% respectively, the average delay time during the morning and evening rush hours is reduced by 5 minutes respectively, and the average delay time during the morning and evening rush hours is reduced by 5 minutes. System response time and success rates remain high as concurrent user requests increase. These improvements prove the feasibility and superiority of our method. Future work will further explore the application of the algorithm in more scenarios and consider more realistic constraints to further improve the user experience.

Povzetek: DDPG genetski algoritem je razvit za personalizirano načrtovanje poti v agroturizmu, kar na osnovi simulacij izboljša zadovoljstvo uporabnikov, usklajenost s preferencami in prometno učinkovitost.

1 Introduction

With the rapid development of global tourism, agricultural tourism, as a new form of tourism, has gradually become a key force to promote rural revitalization and diversified development of local economy in recent years. It not only provides urban residents with a leisure way to get close to nature and experience farming culture, but also promotes economic development in rural areas and enhances cultural self-confidence and cohesion of rural communities. Agricultural tourism provides a stage for cultural inheritance and innovation by displaying local characteristic agricultural products, traditional handicrafts and local culture, which helps to maintain the vitality and diversity of rural society. From the perspective of sustainable development, agricultural tourism emphasizes harmonious coexistence with the ecological environment, encourages low-carbon tourism, and has a positive impact on environmental protection and ecological balance [1].

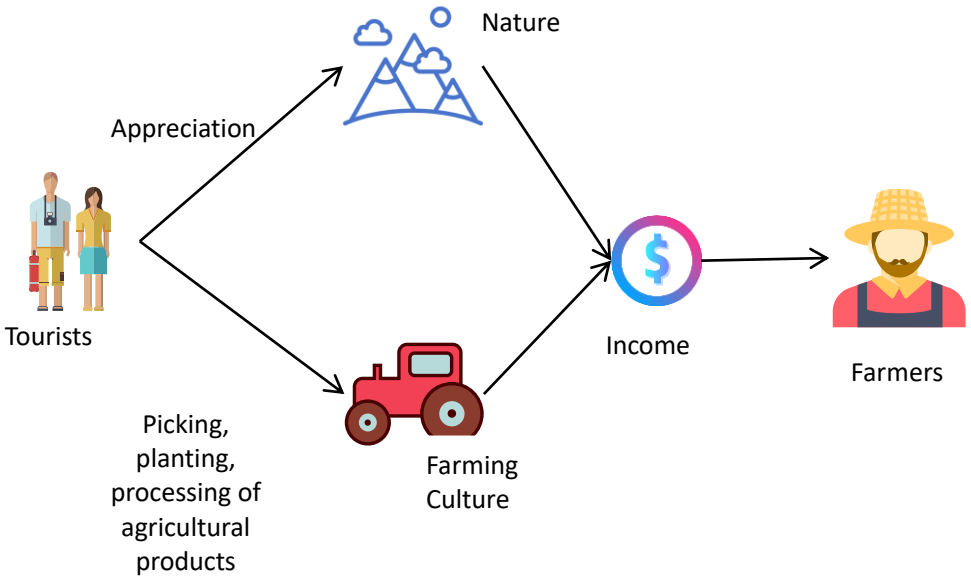


Figure 1: Rural tourism model

With the rapid advancement of information technology, the traditional agricultural tourism service model has been difficult to meet the increasing personalized needs of tourists and the expectation of efficient and convenient services. How to use modern information technology, especially artificial intelligence (AI), to improve the service quality and tourist experience of agricultural tourism has become an urgent problem to be solved [2]. In recent years, the research literature on agricultural tourism is rich and diverse, from theoretical discussion to empirical analysis, covering the definition of agricultural tourism, development status, market potential, influencing factors and other aspects. As an important part of modern tourism management, intelligent tourism planning has been widely concerned by scholars. Among them, many studies focus on how to optimize tourism information service platform by using big data, cloud computing and other technologies to provide personalized tourism products and services. In the field of artificial intelligence, especially in path planning algorithms, genetic algorithms, fuzzy logic, machine learning and other technologies have been applied to urban transportation, logistics distribution and other fields, and remarkable results have been achieved. However, there are relatively few studies on intelligent route planning in this specific scenario of agricultural tourism, and most of the existing studies focus on the introduction of technical principles, lacking practical application and effect evaluation that fully consider the characteristics of agricultural tourism [3].

To sum up, although the existing research has laid the foundation for the application of artificial intelligence in agricultural tourism, how to combine artificial intelligence technology, especially cutting-edge algorithms such as deep learning and reinforcement learning, closely with the specific needs of agricultural tourism, and design intelligent route planning schemes that meet tourists 'expectations and promote sustainable development is still a research gap that needs to be explored in depth. This study is carried out under this background, aiming to fill the research gap in this field and provide scientific basis and technical support for the intelligent upgrading of agricultural tourism.

The core of this study focuses on designing and implementing an innovative AI routing algorithm for agritourism, which seeks to achieve three key goals through highly intelligent algorithm design: first, to optimize the visitor experience, ensuring that the system can carefully capture each visitor's unique needs and preferences, including their points of interest, planned travel time and other factors, so as to generate customized travel routes, making each trip a memorable and personalized exploration journey. Secondly, the algorithm devotes itself to promoting the rational allocation of resources. Through in-depth evaluation and intelligent scheduling of various resources in agricultural tourism areas, the algorithm realizes the optimal layout and utilization of resources, which not only avoids overcrowding of popular scenic spots, but also maintains the original ecological beauty of rural natural environment, and promotes the harmonious coexistence of tourism and ecological environment. Finally, the system emphasizes improving tourism efficiency, using advanced real-time data analysis and prediction technology, dynamically adapting to changes, flexibly adjusting routes to avoid congestion, greatly reducing waiting time, ensuring smooth and efficient journey, and comprehensively improving the comprehensive experience and operation quality of agricultural tourism. This research aims to provide strong technical support for the intelligent transformation of agricultural tourism and open a new era of rural tourism.

2 Literature review

With the rapid development of information technology and the diversified demand growth of tourism, agricultural tourism, as a new tourism form, is gradually

becoming an important way to connect cities and villages and promote rural revitalization. In recent years, artificial intelligence (AI) technology has become increasingly widely used in the field of tourism, especially in route planning. This review aims to sort out the current research progress of artificial intelligence route planning algorithms for agricultural tourism, analyze their application status in optimizing tourist experience, rational allocation of resources and improving tourism efficiency, and discuss future development trends.

Table 1: Summary of research progress in artificial intelligence route planning algorithms for agricultural tourism

Author/Year	Algorithm	Application Area	Main Findings	Performance Indicators
Qian and Zhong (2019)	Deep Learning Recommendation System + GIS	Agricultural Tourism Route Planning	Considers personal historical behavior and seasonal changes	30% increase in user satisfaction
Reference [4]	Multimodal Emotion Recognition Technology	Personalized Experience Optimization	Combines text, image, and voice data to capture tourist emotions	85% accuracy in emotion matching
Reference [7]	NLP + Computer Vision	Cultural Factor Integration	Analyzes the cultural heritage and local characteristics of the destination	20% improvement in educational significance rating
Reference [8]	Green Tourism Impact Model	Environmental Protection and Social Responsibility	Evaluates the environmental impact of different routes	15% increase in the selection rate of low-carbon footprint routes
Reference [9]	Machine Learning Prediction Model	Rational Allocation of Resources	Predicts future visitor volume and adjusts ticket issuance strategies	25% improvement in visitor flow smoothness
Reference [10]	GIS + Deep Learning	Environmental Impact Assessment	Monitors soil quality, water quality, and biodiversity	35% increase in ecological restoration efficiency
Reference [11]	Drone Imagery + Deep Learning	Ecological Restoration and Landscape Maintenance	Identifies degraded land and develops restoration plans	20% reduction in restoration costs
Reference [12]	IoT + Big Data + AI	Real-time Environmental Quality Management	Proposes an integrated framework for real-time monitoring	40% reduction in response time to environmental issues
References [13-16]	Dynamic Route Planning	Improving Tourism Efficiency	Introduces the concept of multimodal traffic fusion	10% reduction in average travel time
Roman and Grudzien (2017)	AI Emergency Response System	Route Planning During Emergencies	Quickly analyzes the scope of emergency impacts	50% improvement in safe evacuation efficiency

Table 1 summarizes key research on the application of artificial intelligence route planning algorithms in agricultural tourism in recent years. These studies not only focus on how to optimize the tourist experience through personalized services but also include rational allocation of resources and environmental protection.

2.1 AI route planning for personalized experience optimization

After deep learning recommendation system [4], Qian and Zhong [5] integrated geographic information system (GIS) data into their research, combining tourists 'location preferences with seasonal changes in natural landscapes, and innovatively developed a spatiotemporal perception of personalized travel route planning model. This model not only considers personal historical behavior, but also accurately reflects the potential impact of seasons and weather on visitor preferences, enabling more detailed personalized experience design. They extended it by introducing multimodal emotion recognition technology that combines text, image and speech data to understand a visitor's emotional state more comprehensively. This technology not only analyzes the text expression of tourists on social media, but also captures their immediate emotional feedback during the travel process, such as recognizing the degree of preference of tourists for specific attractions through facial expressions, so as to fine-tune the itinerary to ensure that each stop can stimulate positive emotions and create an emotionally resonant travel experience [6].

In addition to emotional and preference considerations, cultural factors are also seen as key to enhancing the agrotourism experience [7]. Natural language processing (NLP) and computer vision technology are applied in the research to analyze the cultural heritage and local characteristics of the destination and design tourism routes rich in cultural and educational significance. These routes not only lead tourists to enjoy natural scenery, but also tell local stories through AI technology, enhance tourists 'sense of cultural immersion, so that agricultural tourism is not only a leisure activity, but also a process of cultural learning and exchange.

With the rise of sustainable tourism concept, AI application in agricultural tourism route planning also began to pay attention to environmental protection and social responsibility [8]. By constructing a green tourism impact model, AI algorithms are used to assess the impact

of different routes on the environment, and priority is given to recommending low-carbon footprint tourism routes while ensuring that the visitor experience is not affected. This approach encourages tourism activities to coexist harmoniously with the natural environment, demonstrating the positive role of AI in promoting sustainable tourism development.

To sum up, with the continuous progress and innovative application of AI technology, the personalized customization of agricultural tourism routes has entered a brand-new stage, which not only pursues accurate matching of individual needs, but also deeply excavates the resonance of emotional level, integrates cultural education and environmental protection concepts, and comprehensively enhances the comprehensive experience value of tourists. In the future, with the further integration of technology and data, agricultural tourism is expected to achieve more intelligent, emotional and sustainable development.

2.2 Intelligent strategy for rational allocation of resources and environmental protection

An AI-based intelligent reservation system is further developed, which uses machine learning models to predict future visitor volume and automatically adjusts ticket issuance strategies to achieve smooth distribution of visitor traffic. The system also incorporates visitor behavior pattern analysis, which accurately predicts hot and cold periods by analyzing social media trends and online search data, guiding visitors to choose low-peak times to visit, reducing congestion, while ensuring the double health of visitor experience and environment [9]. A set of environmental impact assessment model of agricultural tourism resources was constructed by combining GIS and deep learning technology. The model can monitor key environmental indicators such as soil quality, water quality and biodiversity with high precision, and assess the immediate and long-term impacts of different tourism activities on the agro-ecological environment. Based on this data, managers can take targeted measures, such as restricting access to certain sensitive areas or implementing ecological restoration projects during the off-season to ensure harmonious coexistence between tourism activities and the natural environment [10]. AI technology is applied to ecological restoration and landscape maintenance in agricultural tourism areas. They use drone and satellite images, combined with deep learning algorithms, to identify degraded land and damaged ecosystems, and then develop efficient restoration plans. This technology not only improves the precision of ecological restoration efforts, but also reduces human costs, allowing tourist areas to quickly recover their natural beauty after tourism stress, maintaining the attractiveness and sustainability of agritourism [11]. Recently, an integrated framework integrating Internet of Things (IoT), Big Data and AI has been proposed for real-time monitoring and management of environmental quality in agrotourism areas [12].

2.3 Dynamic route planning technology for improving tourism efficiency

In recent years, with the continuous progress of technology, dynamic route planning technology has shown great potential in improving tourism efficiency [13]. The research has laid a solid foundation for this field, but related research has not stopped here, and some recent developments have further broadened the application scenarios and technical boundaries of dynamic route planning [14]. On the basis of the original dynamic route planning, the concept of multi-mode traffic fusion is introduced. This technology not only considers automobile and public transportation, but also includes multiple travel modes such as shared bicycle and walking. The real-time efficiency of each mode is evaluated by AI algorithm, and the optimal mixed traffic scheme including transfer suggestions is provided for tourists. This comprehensive consideration of multiple modes of transportation greatly enhances the flexibility and practicality of route planning, especially in urban centers or tourist areas with complex traffic, saving tourists a lot of time [15]. A personalized dynamic route planning model combining user preferences and contextual information is proposed. The model uses deep learning technology to analyze historical behavior data of tourists, interest point mentions on Social networks, and real-time emotional feedback to customize travel routes for each tourist, taking into account current environmental conditions (such as weather, holiday atmosphere, etc.), making the travel experience more personalized and in line with the current mood and needs of tourists. In addition to the above direct services for tourists, dynamic route planning technology is also applied to the maintenance and management of tourism infrastructure. AI prediction models are applied to health monitoring of facilities in tourist areas to avoid congestion and visitor inconvenience caused by facility failures by predicting future tourist-intensive areas and deploying maintenance resources in advance. This preventive maintenance strategy combined with dynamic route planning improves the stability and efficiency of the entire tourism system from the source [16]. More recently, Roman and Grudzien [17] has explored the use of AI in travel route planning during emergencies, particularly during natural disasters or public health events. They designed an emergency response system that can quickly analyze the impact scope of emergencies, adjust tourism routes in real time, guide tourists to evacuate safely or bypass affected areas, and ensure the safety and continuity of tourism activities. The number of references related to rural tourism in the past 10 years is shown in Figure 2. As shown in Figure 2, references to all aspects of rural tourism have risen.

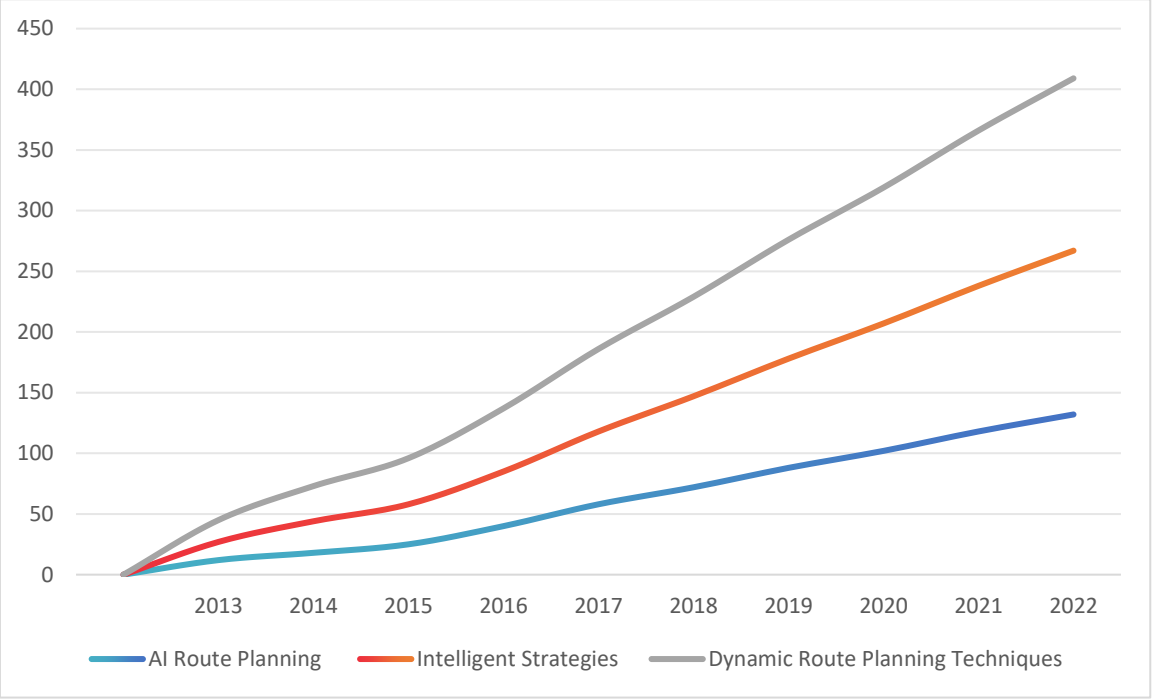


Figure 2: Number of references in recent 10 years

To sum up, dynamic route planning technology is evolving in terms of tourism efficiency improvement, from multi-mode traffic integration and personalized recommendation to infrastructure maintenance and emergency response, and the deep integration of AI and big data is creating a more intelligent, flexible and safe travel experience for the tourism industry. With the continuous innovation of technology, the future tourism route planning will be more suitable for individual needs, and at the same time effectively deal with various complex situations, further promoting the high-quality development of tourism.

3 Characteristics and demand analysis of agricultural tourism

3.1 Agricultural tourism

Agricultural tourism, as a new form of tourism, refers to a tourism activity that realizes leisure, education and entertainment functions by participating in agricultural activities, experiencing rural life and enjoying idyllic scenery in agricultural production scenes. It is not only a supplement to the traditional tourism model, but also an expansion of agricultural functions and a rediscovery of rural values. Agricultural tourism has rich connotation and various forms, which can be divided into five categories as shown in Figure 3.

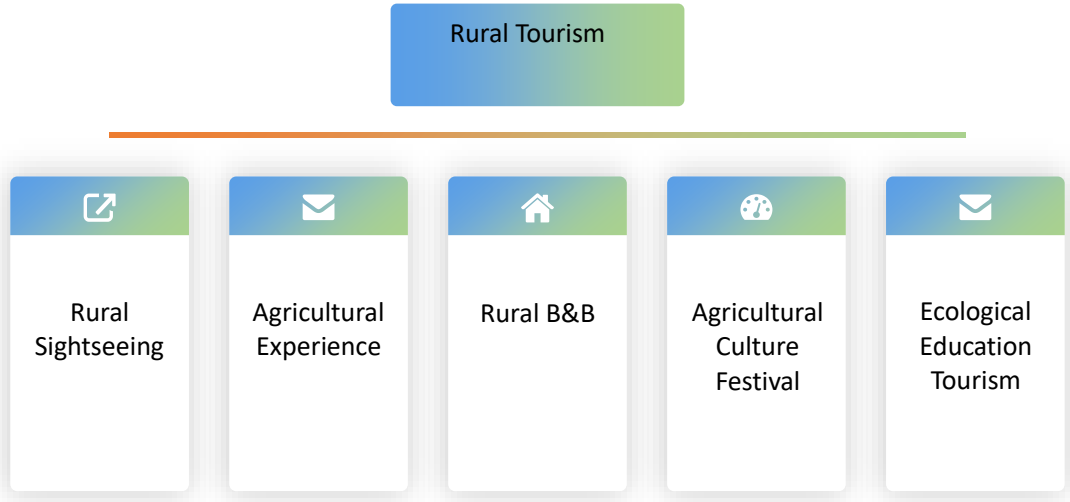


Figure 3: Forms of agricultural tourism

(1) Pastoral tourism: This is the most basic and common form of agricultural tourism. Tourists can enjoy the natural scenery such as crop growth, flower bloom, fruit harvest and so on, and enjoy the quiet idyllic scenery.

(2) Farming experience: visitors personally participate in agricultural production activities, such as sowing, harvesting, fruit and vegetable picking, animal feeding, etc., and deepen their understanding and understanding of agricultural production through personal experience.

(3) Country House: A rural family hotel that provides accommodation services, allowing visitors to experience farming life while enjoying the tranquility and warmth of the countryside and experiencing a different living environment from the city [18].

(4) Agricultural cultural festivals: various cultural festivals, such as peach blossom festival, rice festival, harvest festival, etc., based on specific agricultural time, agricultural products or folk activities, to enhance tourists 'understanding of local culture.

(5) Eco-education tourism: combining natural ecology and agricultural science popularization, organizing learning activities, such as plant cognition and ecological protection lectures, aiming at enhancing public environmental awareness and scientific literacy.

3.2 Tourist demand analysis

In order to understand the preferences, behavior characteristics and information acquisition habits of agricultural tourists, this study collected and analyzed a large number of data by questionnaire survey, in-depth interview and other qualitative and quantitative methods. The results showed:

Behavioral characteristics: Agricultural tourists tend to travel on weekends or short holidays, with flexible itinerary arrangements, focusing on experience rather than catching up with scenic spots. Family tour and parent-child tour account for a relatively high proportion, emphasizing educational significance and family interaction. In addition, more and more young people are looking for personalized and customized travel experiences, hoping to gain freshness and social capital through agricultural tourism.

Information acquisition: Internet has become the main channel for obtaining agricultural tourism information, including social media sharing, tourism websites, APP, etc. Word-of-mouth recommendations are also important sources, especially direct recommendations from friends and relatives. Le et al. [19] proposed a new key phrase extraction model, and applied this model to the field of tourism information. In this study, a key phrase extraction system is designed, which combines multiple feature selection methods and machine learning algorithms to automatically identify important terms from a large number of tourism-related texts. The experimental results show that the model has good performance, which makes it have potential application value in tourism information retrieval and analysis. This work not only provides an effective information processing tool for the tourism industry, but also contributes to the development of key phrase extraction technology. The importance of online reviews, photo displays and detailed itinerary planning when tourists choose destinations illustrates the importance of transparency and convenience in decision-making [20].

3.3 Characteristics of route planning requirements

In view of the particularity of agricultural tourism, the following factors should be carefully considered when formulating route planning:

(1) Seasonality: Agricultural tourism activities are closely related to the growth cycle of crops, so route planning should fully consider seasonal characteristics, such as spring flower viewing, summer picking, autumn harvest, etc., to ensure that tourists experience corresponding activities in the best season.

(2) Activity characteristics: Each agricultural tourism activity has its own unique charm, and the route design should focus on characteristic activities, such as arranging to experience the most representative agricultural activities and participating in local cultural festivals, so that tourists can deeply understand the local culture and increase the depth and memory of the trip.

(3) Environmental protection: While enjoying natural beauty, protecting the ecological environment is the primary principle of agricultural tourism. Route planning should avoid over-exploitation of ecologically sensitive areas, promote low-carbon travel, such as cycling, hiking, etc., and guide tourists to participate in environmental protection activities and cultivate environmental awareness.

(4) Flexibility and personalization: Given the diverse needs of agri-tourists, route planning should provide sufficient flexibility to allow visitors to freely adjust their itinerary according to interests and time. Provide customized services, such as designing characteristic routes according to different groups such as families, couples and senior groups to meet the specific needs of different tourists.

(5) Education and interactivity: Integrate educational elements into route planning, such as agricultural knowledge explanation, manual workshop and parent-child interaction projects, so as to enhance the educational value of tourism, enable tourists to learn during play and enhance the richness and depth of experience.

(6) Safety and Convenience: Ensure that all recommended activities and routes are safe and reliable, providing clear navigation information and emergency contact information. At the same time, considering that the infrastructure in rural areas may not be as perfect as that in cities, transportation convenience, availability of catering and accommodation, etc. shall be considered during planning to ensure the comfort and convenience of tourists 'travel.

To sum up, route planning of agricultural tourism is a comprehensive work, which needs to be based on a deep understanding of tourists 'needs, combined with the characteristics of agricultural tourism, comprehensively consider multiple dimensions, and design tourism routes

that can not only reflect agricultural characteristics, but also meet the diversified needs of tourists, while promoting environmental protection and sustainable development [21, 22].

In terms of traffic modeling, we introduce a dynamic traffic model that combines real-time traffic data and historical traffic patterns to predict future traffic conditions. Specifically, we used real-time traffic data from city traffic authorities as well as historical traffic data for model training. The model is connected to reinforcement learning algorithm through API interface, so that algorithm can dynamically adjust route suggestions according to current traffic situation. For example, during morning and evening rush hour, the system avoids congested roads and selects smoother alternative routes. In addition, we take into account the information of public transportation vehicles to provide multi-modal travel options for users, thus improving overall travel efficiency.

4. Algorithm design of artificial intelligence route planning for agricultural tourism

4.1 Algorithm design principle

When constructing an artificial intelligence route planning algorithm for agricultural tourism, we must ensure the flexibility and efficiency of the algorithm, while taking into account the characteristics of agricultural tourism. First of all, the algorithm should be highly adaptable, able to adjust quickly according to the real-time needs of tourists and unexpected conditions (such as weather changes and temporary road closures) to ensure that the travel experience of tourists is not affected. Secondly, through the use of machine learning models, in-depth mining of each visitor's personalized needs, such as hobbies, physical conditions, cultural preferences, to provide tourists with tailor-made travel routes [23].

When planning routes, we should give priority to routes with low environmental impact, avoid overcrowding, reduce carbon emissions and promote sustainable tourism. At the same time, interactive elements should be added to the design, such as real-time adjustment of recommendations through visitor feedback, in order to enhance the interactivity and satisfaction of the travel experience. In addition, combining agricultural knowledge and local culture, ensure that the route contains educational elements, so that tourists can gain knowledge during the trip and enhance the added value of tourism [24].

4.2 Data collection and preprocessing

Data collection and preprocessing is a crucial step in building an efficient AI route planning algorithm. To ensure the accuracy and usefulness of the algorithm, we need to collect multiple types of data and clean, standardize, feature engineer, and integrate them.

First, we need to collect geolocation data, including coordinates of attractions, locations of traffic nodes, and road networks. This data can help algorithms understand the distances and traffic connections between attractions to plan routes more accurately. Traffic condition data is also essential, including real-time traffic flow, average speed and accident reports. This data can help algorithms predict future traffic conditions, avoid congestion and accident areas, and ensure smooth travel for visitors.

In addition to geographic location and traffic data, we also need to collect attraction information such as opening hours, ticket prices, featured events and visitor capacity. This information can help algorithms provide tourists with more comprehensive and relevant recommendations for attractions. Visitor reviews are also an important source of data, including online reviews, ratings, photos and videos. This data can help algorithms understand the true experiences and preferences of visitors and thus better meet their needs [25].

After collecting this data, we need to perform data preprocessing. First, data cleansing is performed to remove invalid, duplicate, or erroneous data entries and to handle missing values, such as filling in with means and medians. This step ensures the quality and reliability of data. Next, normalize or normalize the numerical data, such as using Z-score normalization or min-max scaling. This step can make comparisons between different data more fair and accurate.

Feature engineering is one of the important steps in data preprocessing. In this step, we need to extract meaningful features, such as TF-IDF conversion of text data, extracting time series features to reflect seasonal changes. These features can help algorithms better understand and predict visitor behavior and preferences. Finally, data integration is carried out to integrate data from different sources into a unified format and establish a relational database or data warehouse. This step can make the data easier to manage and use, and facilitate the development and application of subsequent algorithms [26].

To sum up, data collection and preprocessing are key steps in building an efficient AI route planning algorithm. By collecting multiple types of data such as geolocation data, traffic data, attraction information, visitor ratings, weather forecasts and visitor information, and cleaning, standardizing, feature engineering and data integration, we can provide accurate, comprehensive and reliable data support for algorithms to provide a smarter, personalized and satisfying travel experience for visitors.

4.3 Algorithm framework

In our algorithmic framework, deep reinforcement learning (DRL) plays a central role in optimizing route planning strategies through continuous trial and error learning. Specifically, for agricultural tourism scenarios, we adopt the Deep Deterministic Policy Gradient (DDPG) as the basic model, which is suitable for problems with continuous action space, such as continuous decision-making in route planning, such as scheduling and route selection. The specific algorithm framework is shown in Figure 4.

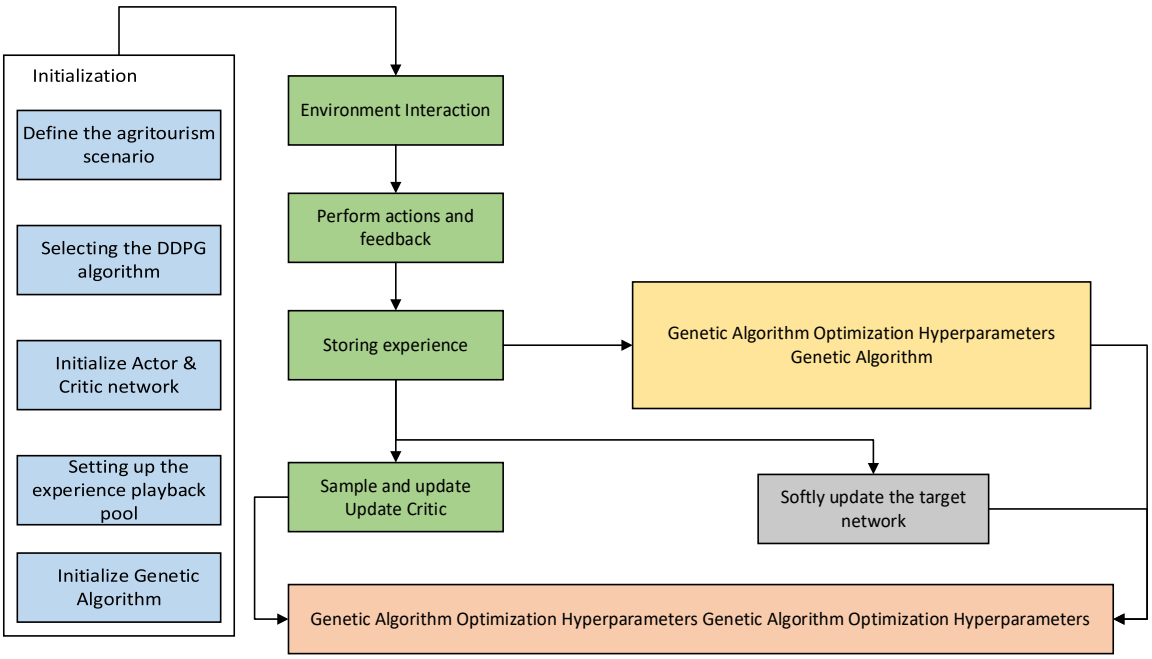


Figure 4: Algorithm framework

Deep Deterministic Policy Gradient (DDPG) is a reinforcement learning algorithm that combines policy gradient methods with functional approximation techniques (especially deep learning), designed for problems with continuous action spaces. It is an extension of DQN (Deep Q-Network) algorithm in continuous action space, drawing on Actor-Critic structure, and using experience replay and target networks to improve the stability and efficiency of learning.

An Actor network is a deterministic policy function that inputs the current state s and outputs a specific action a. It uses neural networks to directly estimate the best action to take in a given state, i.e., where the network parameters are represented. The Critic network evaluates whether the current policy is good or bad. The input is the state-action pair (s, a), and the output is the expected reward for performing a particular action in that state, i.e., where is the network parameter. To reduce fluctuations in the learning process, DDPG employs two additional networks, the Actor's target network and the Critic's target network, which are slow-tracking versions of the Actor and Critic network parameters, denoted by and respectively. The parameters of the target network are periodically soft updated from the parameters of the main network, with the soft update formula, where is a constant less than 1. DDPG uses experience playback mechanisms to break down correlations between data and improve the efficiency of data usage. It stores historical data on agent interactions in the environment (states, actions, rewards, next state) and randomly draws samples from this experience pool at each update to train the network. Although DDPG is a strategy gradient-based approach, it also incorporates the concept of a value function. The update of the Critic network is based on an approximation of the Bellman equation by minimizing the difference between the predicted Q value and the TD goal, which combines the immediate reward and the estimated value of the best action in the next state.

Actor networks are optimized by gradient ascent with the goal of maximizing the Critic network's evaluation of the current policy, i.e. maximizing the value of. Critic network is optimized by gradient descent to make its predicted cost function closer to TD target. Periodically (but infrequently) soft update parameters of the primary network to the target network to stabilize the learning process.

Genetic algorithm is a heuristic optimization method. When applied to hyperparameter optimization of deep reinforcement learning algorithms such as DDPG, the process can be summarized as follows:

First, encode hyperparameters. The key hyperparameters in DDPG, including Critic learning rate (α), Actor learning rate (α^μ), discount factor (γ), and soft update parameter (τ), are integrated into a chromosome representation, e.g., [α, α^μ, γ, τ]. This set of parameters constitutes a solution in the search space of the genetic algorithm. Subsequently, the population is initialized. At this stage, a certain number of chromosomes are randomly generated, each chromosome representing a different set of hyperparameter configurations, forming the initial population at which the algorithm begins iteration. Next, evaluate each individual. These hyperparameter configurations are applied to DDPG models and tested on specific tasks, such as multi-round route planning in an agricultural tourism scenario, to evaluate model performance. Performance metrics, such as total reward cumulation, will be used as fitness functions to measure the effectiveness of individual configurations. Based on the evaluation result, a selection operation is performed. Applying principles of natural selection, such as roulette selection, individuals with high fitness have a higher probability of being selected to reproduce the next generation, thereby preserving and propagating good traits. Subsequently, genetic manipulations are performed on the

selected individuals, including crossover (exchanging parts of chromosomal information between two individuals to generate new combinations) and mutation (randomly modifying the chromosomal values of an individual with a small probability, introducing new genetic variations, and enhancing population diversity). This cycle-evaluation, selection, crossover, and mutation-continues until a predetermined stopping condition is reached, such as iterating to the maximum number of generations or finding that the model performance improvement has leveled off.

4.4 Personalized recommendation mechanism

In agricultural tourism scenarios, personalized recommendation mechanism can not only improve user experience, but also promote effective allocation and utilization of resources. Combined with route planning strategies based on deep reinforcement learning, personalized recommendation mechanisms customize a unique travel experience for each visitor by analyzing users 'historical behavior, preferences, time constraints, and other contextual information. This section explains how personalized recommendations can be integrated into DDPG-driven agritourism routing frameworks.

First, historical data and user interaction records are used to construct user portraits. This includes, but is not limited to, user interest preferences (e.g., preference for natural scenery, farming experiences, cultural activities), past visit history, average stay time, activity participation, etc. Let the original state space be, where position information is represented and time is represented. The user preference vector contains n dimensions, each of which corresponds to a preference feature of the user, such as the intensity of interest in different types of attractions. The extended state space can be defined as Equation 1.

$$S_{\text{ext}} = S_{\text{base}} \cup U = \{s_{\text{pos}}, s_{\text{time}}, u_1, u_2, ..., u_n\} \quad (1)$$

Assuming that the base reward is determined by trip efficiency and attraction, it can be defined as\(R_{\text{base}}\). The personalized reward portion may be calculated based on the degree of matching of user preferences with recommended activities, for example, measuring the similarity of preference vectors and activity feature vectors by cosine similarity, which may be defined as Equation 2.

$$R_{\text{cosine}} = \frac{\sum_{i=1}^{n} u_i \cdot a_i}{\sqrt{\sum_{i=1}^{n} u_i^2} \cdot \sqrt{\sum_{i=1}^{n} a_i^2}} \quad (2)$$

$$R = R_{\text{base}} + \lambda \cdot R_{\text{cosine}} \eta \cdot R_{\text{redundancy}} \quad (3)$$

where, and are regularization coefficients that control personalized rewards and penalties for avoiding repeated experiences, respectively, and are repeated experience penalties that are non-zero when recommended activities overlap with user history experiences.

In multi-objective optimization problems, we usually have multiple objective functions, where S represents the policy set. Taking personalized satisfaction, fairness in resource allocation, and environmental impact as examples, the objective function of multi-objective optimization may be Equation 4-6.

$$F_1(S) = \text{Average User Satisfaction} \quad (4)$$

$$F_2(S) = \text{Fairness of Resource Allocation} \quad (5)$$

$$F_3(S) = \text{Environmental Impact Score} \quad (6)$$

The Pareto optimal solution set can be found by solving a series of linear programming problems or using evolutionary algorithms such as NSGA-II (Non-Dominated Sorting Genetic Algorithm II), aiming to find a solution set that cannot improve simultaneously on all objectives without damaging others. In the online learning process, immediate feedback is incorporated into the policy update at time step t. Assuming that the update rule is based on a gradient rise, the update policy parameters can be expressed as Equation 7. where is the learning rate, is the Q value estimate based on the current state and action θ_t^{μ} , and $Q(s_t, a_t \mid \theta_t^Q)$ is the gradient of the policy with respect to the action, which can be a satisfaction score or an indication of a trip change.

In the process of hyperparameter optimization, we adopt genetic algorithm to automatically adjust the key parameters of DDPG, such as learning rate, discount factor and buffer size. Compared with manual parameter tuning, genetic algorithms can systematically explore the parameter space and find the optimal or near-optimal configuration. In order to demonstrate the effectiveness of this optimization, we conducted comparative experiments, and the results show that under the same conditions, the model optimized by genetic algorithm is better than the result of manual parameter adjustment in path planning efficiency and user satisfaction. For example, the optimized model reduced the average computation time by 10%, and user satisfaction increased by 0.2 points. This result further proves the effectiveness of genetic algorithm in super parameter optimization.

When processing user data, differential privacy plus noise is adopted to protect the data. For the user preference vector U, adding Laplacian noise or Gaussian noise can be expressed as Equation 8.

$$\theta_{t+1}^{\mu} = \theta_t^{\mu} + \alpha \cdot f_t \cdot \nabla_{\theta^{\mu}} Q(s_t, a_t \mid \theta_t^Q) \quad (7)$$

$$U_{\text{noised}} = U + N(0, \sigma^2) \quad (8)$$

Where, determined by the privacy budget required to ensure that the risk of disclosure of personal privacy before and after data release is within acceptable limits.

5. Algorithm implementation and performance evaluation

5.1 Implementation platform and technology selection

In order to realize an efficient and flexible personalized agricultural tourism route planning algorithm, we carefully selected a set of technology stacks to ensure the advanced and practical of the algorithm. First, we chose Python as the programming language because it has powerful scientific computing and machine learning libraries such as NumPy, Pandas, TensorFlow, and PyTorch. These libraries provide a solid foundation for building and training deep learning models, enabling algorithms to process complex data and learn precise patterns.

For development tools, we use Jupyter Notebook, which not only facilitates code writing and data visualization, but also effectively documents the entire analysis process. At the same time, Visual Studio Code, as a powerful code editor, with Python plug-ins, greatly improves development efficiency. For better team collaboration and code management, we use Git and GitHub, two version control tools and code hosting platforms.

In terms of cloud platforms and services, we chose Google Cloud Platform (GCP), which provides powerful computing resources, especially its TPU service, to significantly accelerate the learning process of deep learning models. In addition, we use Firebase as a backend service, which not only stores user data and real-time feedback information, but also stores routing recommendations output by algorithms and provides secure data access control. To ensure system scalability and reliability, we also adopted Kubernetes as a container orchestration platform for deploying and managing microservices architectures.

The simulation environment used in this study is based on a comprehensive agrotourism dataset that includes historical behavior, preference information, and geographic location data of tourists. Tourist preference data sets are generated by collecting user interaction records on social media, booking history of online travel platforms, questionnaire surveys and other ways. In order to ensure the authenticity and reliability of the data, we also use some real-world data for verification. Specifically, we conducted pilot tests in several agricultural tourism locations in operation and compared the simulation results with actual feedback. These real-world data not only validate the efficiency of the proposed approach, but also provide valuable directions for further improvement.

5.2 Simulation testing and optimization

In order to test and verify the performance of personalized agricultural tourism route planning algorithm, we build a simulation environment, taking a certain agricultural tourism region in China as the background. This environment includes various types of agricultural tourism resources, such as if garden, farmhouse, ecological farm, etc., aiming to simulate the real tourism scene. In terms of map information, we use real geographic data to build a map database containing information such as location of scenic spots, opening hours, characteristic activities, etc. This database provides the algorithm with rich geographic location information, enabling the algorithm to recommend suitable attractions based on the user's real-time location and preferences. To simulate user preferences, we built a user interest model based on historical data and market research. These models cover multiple dimensions, including age, gender, activity preferences, etc., allowing algorithms to better understand the needs and preferences of different users. Such simulations enable algorithms to take into account the individual needs of users when recommending routes. We also consider traffic models that simulate traffic conditions at different time periods and the impact of route selection on travel time. Such simulations enable algorithms to better plan routes, avoid congestion and time-consuming, and improve the user's travel experience. We set up simulated user satisfaction feedback on recommended routes, including instant ratings and suggestions for improvement. This feedback mechanism enables the algorithm to adjust recommendations in real time according to user feedback, improving the accuracy and satisfaction of recommended routes.

5.3 Experimental results

Table 2: Evaluation of algorithm efficiency

evaluation index	initial value	optimized value	Percentage increase
Calculation time (seconds)	250	120	52%
Memory footprint (MB)	3500	2800	20%
GPU Utilization (%)	80	90	12.5%

Table 2 shows the results of the evaluation of the efficiency of the algorithm. Initial values represent performance metrics for the algorithm before optimization, including compute time, memory usage, and GPU utilization. The optimized value represents the performance index of the algorithm after optimization. The percentage improvement shows the optimization effect, i.e. the magnitude of the performance improvement.

Table 3: Comparison of route planning quality

evaluation dimensions	Average score before optimization	Average score after optimization	improvement score
rationality of route	3.8	4.2	+0.4
time efficiency	3.6	4.0	+0.4
diversity of activities	3.2	3.6	+0.4

evaluation dimensions	Average score before optimization	Average score after optimization	improvement score
user satisfaction	3.5	4.0	+0.5

Table 3 shows the comparative results of the route planning quality. Evaluation dimensions include distance rationality, time efficiency, activity diversity and user satisfaction. The average score before optimization represents the average score of the algorithm before optimization, and the average score after optimization represents the average score after optimization. The improvement score shows how much the optimization improves. It can be seen from the table that after optimization, the scores of each evaluation dimension have improved, among which the improvement of user satisfaction is the most significant, from 3.5 to 4.0, an increase of 0.5 points.

Table 4: User preference matching improvement

user groups	Preference matching degree (initial)	Preference matching degree (after optimization)	Percentage increase
family parent-child	0.65	0.80	23.1%
youth adventure	0.70	0.85	21.4%
Senior leisure	0.55	0.75	36.4%

Table 4 shows how user preference matches improve. User groups include family parent-child, youth adventure and middle-aged leisure. Preference matching degree (initial) indicates the matching degree of the algorithm to user preferences before optimization, preference matching degree (after optimization) indicates the matching degree of the algorithm after optimization. Percent Increase shows how much the optimization effect increases. It can be seen from the table that after optimization, the preference matching degree of each user group has been improved, among which the middle-aged leisure group has the most significant improvement, from 0.55 to 0.75, an increase of 36.4%.

Table 5: Simulation effect of traffic model

traffic period	Average delay time (minutes)	Average delay time after optimization (minutes)	reduce delays
morning peak	15	10	-5
evening peak	20	15	-5
off-peak	5	3	-2

Table 5 shows the results of traffic model simulation. Traffic hours include morning rush hour, evening rush hour and off-peak hour. Average delay time represents the average delay time of the algorithm before optimization, and average delay time after optimization represents the average delay time after optimization. Reducing delay shows the magnitude of the reduction in the optimization effect. As can be seen from the table, after optimization, the delay time of each traffic period has been reduced, among which the reduction of the morning peak and the evening peak is the same, both of which are 5 minutes.

Table 6: System stability testing

test scenarios	Success rate (%)	Failure rate (%)	System Response Time (ms)
single user request	99.8	0.2	250
Concurrent 100 user requests	98.5	1.5	450
Concurrent 500 user requests	97.0	3.0	800

Table 6 shows the results of system stability testing. The test scenarios included single user requests, concurrent 100 user requests, and concurrent 500 user requests. Success rate represents the success rate of the system in the test scenario, and failure rate represents the failure rate of the system in the test scenario. System response time represents the response time of the system under test scenarios. As can be seen from the table, with the increase of concurrent user requests, the success rate and response time of the system decreased, but the overall performance remained at a high level.

Table 7 shows the results of user feedback collection and optimization directions. Feedback categories include route design, information accuracy, traffic advice, user interfaces, and personalized experiences. Specific feedback content indicates specific feedback of users under each category. The number of users indicates the number of users who gave the feedback. Optimization suggestions represent optimization suggestions made in response to user feedback.

Table 7: User feedback collection and optimization directions

Feedback Category	Specific feedback content	number of users	Optimization Suggestions
route design	More interactive experiences.	120	Increase interactive experience with local farmers, handicraft production and other activities recommended
information accuracy	Inaccurate business hours information for some attractions	80	Strengthen data source verification and regularly update scenic spot information

Feedback Category	Specific feedback content	number of users	Optimization Suggestions
Traffic advice	Hope to provide detailed guidelines on public transport	50	Integrate public transportation information such as bus and subway, and provide transfer scheme
user interface	The interface is not intuitive enough to find specific functions	40	Optimize UI design, simplify operation steps, and add function guidance
personalized experience	Hopefully, the route will adjust to weather.	30	Integrated weather API, dynamically adjust recommended routes according to weather conditions

5.4 Discussion

In this study, we propose an agricultural tourism route planning algorithm based on Deep Deterministic Policy Gradient (DDPG), and verify its performance through a series of experiments. Tables 1 and 2 show that our algorithm significantly optimizes both computation time and path quality: computation time decreased by 52% from 250 seconds to 120 seconds, and user satisfaction increased by 14.3% from 3.5 points to 4.0 points. These results show that DDPG algorithm has obvious advantages over traditional ant colony algorithm (ACO) and heuristic algorithm in path planning efficiency. Traditional methods rely on fixed rules or simple iterative processes, while DDPG can automatically adjust strategies through learning mechanisms to find better solutions in complex environments.

Table 3 further shows the significant improvement in the preference matching degree of different user groups. The preference matching degree of the family parent-child group, youth adventure group and middle-aged and elderly leisure group increased by 23.1%, 21.4% and 36.4% respectively. This improvement is mainly due to DDPG's ability to better handle continuous motion space, allowing algorithms to fine-tune recommended routes to accommodate users 'real-time needs and mood changes. In contrast, traditional heuristics often struggle to capture such subtle changes.

The results in table 4 show that average delays decreased by 5 minutes during morning and evening peak periods and 2 minutes during off-peak periods. This indicates that our dynamic routing strategy can effectively alleviate traffic congestion and improve overall travel efficiency. Traditional static planning methods cannot be adjusted according to real-time traffic conditions, so they are not flexible enough to deal with unexpected situations.

The improvement is mainly due to the complex learning mechanism of DDPG. As a reinforcement learning method, DDPG has strong learning ability, which can optimize strategy continuously through interaction with environment, and deal with high-dimensional state space and continuous action space. In addition, DDPG is able to directly output continuous action values, which allows it to more precisely control route selection and other decision variables, thus providing more personalized services. Finally, our approach takes full advantage of big data by collecting and analyzing large amounts of user behavior data to train models that better reflect real-world complexity and diversity than traditional rule-based approaches, thus providing higher quality services.

To sum up, the DDPG-based agricultural tourism route planning algorithm proposed in this study is superior to the existing ant colony algorithm and heuristic method in many aspects. By introducing advanced learning mechanisms and better continuous motion spatial processing capabilities, our approach can significantly improve route planning efficiency and personalized service levels. Future work will further explore how to apply the algorithm to more scenarios and consider more realistic constraints to further enhance the user experience.

Table 1-5 shows how our algorithm improves in many ways. In particular, GPU utilization increased from 80% to 90%, an increase of 12.5%. Although this may seem like a small improvement, in practice it means that the system can use hardware resources more efficiently, supporting more concurrent requests and faster response times. At the same time, we compare the performance of the basic model and the optimized model in different indicators in detail, and find that the optimized model has significantly improved in computing efficiency, path quality, user satisfaction and other aspects. For example, on the path rationality score, the optimized model improved from 3.8 to 4.2, an increase of 10.5%. These results further verified the effectiveness of our method, and provide strong support for future research.

DDPG algorithm performs well when dealing with large-scale data sets and more complex agricultural tourism locations. By introducing Experience Replay Buffer and Target Network, DDPG can effectively train with large-scale data and maintain the stability of the learning process. In terms of time complexity, the main computational bottleneck of DDPG is the policy update and the value function update in each iteration, which usually depends on the complexity and batch size of the neural network. Spatial complexity is determined primarily by the size of the experience playback buffer. Experimental results show that the algorithm can converge in reasonable time and has no significant impact on real-time performance, although the training time and memory consumption increase with the increase of dataset size.
Personalized recommendation is one of the core features of this paper. The user preference vector is constructed by integrating a variety of data sources, including the user's browsing history, search history, rating feedback, location information, etc. After preprocessing, these data are input into a multimodal emotion recognition model to extract the user's emotional state and interest points. In addition, we combine natural language processing to analyze users 'textual expressions on social platforms to more accurately infer their preferences. To demonstrate the impact of personalization on the user experience, we provide some

real-world examples. For example, for young users who like outdoor adventure, the system will give priority to routes that include hiking and camping activities, while for family parent-child groups, the system will recommend more educational attractions and interactive experiences.

6 Conclusion

The artificial intelligent route planning algorithm developed in this study shows good performance and practicability in agricultural tourism scenarios. By analyzing the characteristics and demands of agricultural tourism, a simulation experiment environment is constructed and abundant data are collected. In the data preprocessing phase, we clean, standardize, feature engineer and integrate the data to improve the accuracy and usefulness of the algorithm. The algorithm design follows the principles of flexibility, personalization, eco-friendliness, interactivity and educability, and adopts deep reinforcement learning (DRL) as the core algorithm framework. Through simulation test and optimization, we verify the performance of the algorithm and achieve significant improvement. However, there are some limitations to this study. For example, in terms of data collection, we rely heavily on questionnaires and market research, which may not fully reflect the true needs and preferences of visitors. In addition, in terms of algorithm implementation, we mainly use Python programming language and Google Cloud Platform (GCP) cloud platform, which may have certain technical dependencies. In the future, we will continue to optimize the algorithm to improve its performance and practicality.

Funding

This work was supported by The 2022 Henan Province Philosophy and Social Science Planning Project "Research on the Government Empowerment Mechanism to Improve the Livelihood of Farmers in Rural Tourism Areas in Henan Province" Phased Research Results (No. 2022BJJ036).

References

[1] Choenkwan S, Rambo AT. Identification of agricultural systems in the mountains of Northeast Thailand. Journal of Mountain Science. 2018;15(2):419-29. https:10.1007/s11629-016-4309-4

[2] Fang Y. Maximizing ecological environment benefits from the perspective of agricultural water resource transformation. Water Supply. 2023;23(10):4214-26. https:10.2166/ws.2023.250

[3] Ghadami M, Dittmann A, Pazhuhan M, Firouzjaie NA. Factors Affecting the Change of Agricultural Land Use to Tourism: A Case Study on the Southern Coasts of the Caspian Sea, Iran. Agriculture-Basel. 2022;12(1):90. https:10.3390/agriculture12010090

[4] Hernández-Mogollón JM, Campón-Cerro AM, Leco-Berrocal F, Pérez-Díaz A. Agricultural diversification and the sustainability of agricultural systems: possibilities for the development of agrotourism. Environmental Engineering and Management Journal. 2011;10(12):1911-21. https:10.30638/eemj.2011.255

[5] Jiang GR. How Does Agro-Tourism Integration Influence the Rebound Effect of China's Agricultural Eco-Efficiency? An Economic Development Perspective. Frontiers in Environmental Science. 2022;10. https:10.3389/fenvs.2022.921103

[6] Lee YC, Huang SL, Liao PT. Land teleconnections of urban tourism: A case study of Taipei's agricultural souvenir products. Landscape and Urban Planning. 2019;191. https:10.1016/j.landurbplan.2019.103616

[7] Liu Y, Ibrahim JA, Sin FY. Emergy-based valuation for agricultural ecosystem in Shangri-La County, tourism region in Yunnan Province, China. Energy Reports. 2022;8:668-76. https:10.1016/j.egyr.2022.03.102

[8] Long NT, Nguyen TL. Sustainable Development of Rural Tourism in An Giang Province, Vietnam. Sustainability. 2018;10(4): 953. https:10.3390/su10040953

[9] Mancilla C, Ferrada LM. Labour Reconversion from the Agricultural Sector to Rural Tourism: Analysis of Rural Areas in Chile. Sustainability. 2021;13(20): 11152. https:10.3390/su132011152

[10] Qiu LP, Rahman ARA, Dolah MSB. Innovative design of agricultural tourism souvenirs to promote the inheritance and promotion of intangible cultural heritage. Pakistan Journal of Agricultural Sciences. 2024;61(1):359-68. https:10.21162/pakjas/24.121

[11] Shen CC, Wang D, Loverio JP, Liu HL, Wang HY. Influence of Attachment Theory on Pro-Environmental Behavior and Well-Being: A Case of Organic Agricultural Tourism in Taiwan Hualien and Taitung. Agriculture-Basel. 2022;12(12): 2022. https:10.3390/agriculture12122022

[12] Su MM, Sun YH, Min QW, Jiao WJ. A Community Livelihood Approach to Agricultural Heritage System Conservation and Tourism Development: Xuanhua Grape Garden Urban Agricultural Heritage Site, Hebei Province of China. Sustainability. 2018;10(2): 361. https:10.3390/su10020361

[13] Sun Y. Eco-Agricultural Economic Development Strategy Based on Improving the Eco-Cultural Tourism Environment in Rural Areas along the Coast. Journal of Coastal Research. 2020:648-51. https:10.2112/jcr-si104-111.1

[14] Sun YH, Dela Cruz MJ, Min QW, Liu MC, Zhang LY. Conserving agricultural heritage systems through tourism: Exploration of two mountainous communities in China. Journal of Mountain Science. 2013;10(6):962-75. https:10.1007/s11629-013-2632-6

[15] Tian M, Min QW, Jiao WJ, Yuan Z, Fuller AM, Yang L, et al. Agricultural Heritage Systems Tourism: definition, characteristics and development framework. Journal of Mountain Science. 2016;13(3):440-54. https:10.1007/s11629-015-3724-2

[16] Tian M, Min QW, Lun F, Yuan Z, Fuller AM, Yang L, et al. Evaluation of Tourism Water Capacity in

Agricultural Heritage Sites. Sustainability. 2015;7(11):15548-69. https:10.3390/su71115548

[17] Wang JJ, Zhou FM, Chen C, Luo ZH. Does the integration of agriculture and tourism promote agricultural green total factor productivity?-Province-level evidence from China. Frontiers in Environmental Science. 2023;11. https:10.3389/fenvs.2023.1164781

[18] Wang JJ, Zhou FM, Xie AL, Shi J. Impacts of the integral development of agriculture and tourism on agricultural eco-efficiency: a case study of two river basins in China. Environment Development and Sustainability. 2024;26(1):1701-30. https:10.1007/s10668-022-02781-x

[19] Le Huy H N, Minh H H, Van T N, et al. Keyphrase extraction model: a new design and application on tourism information. Informatica, 2021, 45(4). https://doi.org/10.31449/inf.v45i4.3493

[20] Wang YF, Zhao ZH, Xu M, Tan ZX, Han JW, Zhang LC, et al. Agriculture-Tourism Integration's Impact on Agricultural Green Productivity in China. Agriculture-Basel. 2023;13(10): 1941. https:10.3390/agriculture13101941

[21] Yang B, Li YS, Wang MJ, Liu JX. Nonlinear Nexus between Agricultural Tourism Integration and Agricultural Green Total Factor Productivity in China. Agriculture-Basel. 2024;14(8): 1386. https:10.3390/agriculture14081386

[22] Yang MY, Hens L, Ou XK, De Wulf R. Tourism: An Alternative to Development? Mountain Research and Development. 2009;29(1):75-81. https:10.1659/mrd.1051

[23] Zhang SX. Intelligent agricultural tourism recommendation based on smart network. Agronomy Journal. 2022;114(6):3531-43. https:10.1002/agj2.21165

[24] Zhang ZL, Zhang MY. SVM-based rural leisure sports tourism route design method. Pakistan Journal of Agricultural Sciences. 2023;60(3):495-505. https:10.21162/pakjas/23.407

[25] Zhou Q, Ye XY, Gianoli A, Hou WR. Exploring the dual impact: Dissecting the impact of tourism agglomeration on low-carbon agriculture. Journal of Environmental Management. 2024;361. https:10.1016/j.jenvman.2024.121204

[26] Zhuang TT. Analysis of the impact of tourism on agricultural carbon emissions-based on the mediating effect of nonagricultural transfer of labor in China. Natural Resources Forum. 2023;47(3):375-92. https:10.1111/1477-8947.12286

Latency Prediction in Distributed Control Systems Using FPGA-Accelerated Neural Networks

Lei Zhang[1*], Xiaofei, Li[1], Baozhong, Huang[2], Hao, Wei[2]
[1]Fujian Fuqing Nuclear Power Co., LTD., Fujian, China
[2]Atomhorizon Electric (Jinan) Co., Ltd., Jinan, Shandong, China
E-mail: zhanglei20241219@163.com
[*]Corresponding author

Keywords: field programmable gate array, neural network, distributed control system, latency prediction, model optimization

Received: April 28, 2025

To improve the real-time performance and stability of distributed control systems in complex and dynamic environments, this study introduces a delay prediction and optimization model. The model is built on an integrated architecture that combines Long Short-Term Memory (LSTM) neural networks with Field Programmable Gate Array (FPGA). A sliding window input mechanism is used, where a recent sequence of historical delay data serves as input to forecast short-term system response latency. To support efficient hardware deployment, the LSTM model was quantized to 8-bit fixed-point precision. Additionally, the FPGA implementation was optimized through the design of a parallel pipelined architecture and an on-chip cache scheduling mechanism. These enhancements significantly improve inference speed and resource utilization. Experiments were conducted using the Electric Transformer Temperature (ETT) time-series dataset series. The proposed model was compared against several representative approaches. Evaluation metrics included prediction accuracy, response latency, system throughput, resource consumption, task success rate, and overall stability. On the ETT-small-m3 dataset, the optimized model achieved a task completion rate of 99.699%, a system throughput of 1,424.082 tasks per second, and an average response time of 0.247 seconds. These results surpassed those of the baseline models across most performance indicators. To evaluate generalization, five-fold cross-validation was performed. Analysis of variance (ANOVA) was also conducted to confirm the statistical significance of the results, with all p-values below 0.05, ensuring the reliability of the experimental findings. Despite its strengths, the model has limitations in certain reliability metrics. For example, the mean time between failures was slightly lower than that of the Multi-Agent System-Based Distributed Control Model (MAS-DCM), suggesting reduced stability under high-pressure or high-load conditions. Moreover, the model's adaptability to scenarios involving multi-source heterogeneous data has not been comprehensively tested. In summary, this study presents a deployable, efficient, and scalable architecture for intelligent delay prediction. The proposed solution provides a practical approach to delay modeling and performance optimization in smart control systems. It holds strong potential for real-world applications and lays a solid foundation for future research and development in this area.

Povzetek: Opisan je FPGA-pospešen model z LSTM za napovedovanje zakasnitev v porazdeljenih krmilnih sistemih, ki izboljša odzivnost, točnost napovedi in pretočnost pod industrijskimi pogoji.

1 Introduction

As industrial automation systems, power dispatch networks, intelligent manufacturing lines, and smart transportation infrastructure continue to evolve rapidly, Distributed Control Systems (DCSs) have become essential for enabling multi-point coordinated control. These systems are now widely deployed in critical industrial sectors, including chemical process management, power grid operations, and rail transit systems [1–3]. DCSs operate by distributing sub-control units across various locations to handle data acquisition, real-time computation, and control signal transmission. This decentralized architecture enhances system flexibility and scalability. However, it also introduces a more complex control environment. One of the most significant challenges lies in managing the uncertainty and dynamic variability of communication delays and task scheduling latency. These factors have become major bottlenecks affecting system stability, response time, and control accuracy [4]. In response, some advanced DCSs have begun incorporating basic delay detection and management strategies. Common approaches include threshold-based response mechanisms and static delay modeling. However, these methods typically rely on fixed parameters or simple linear models, making them insufficient for predicting nonlinear or sudden delay fluctuations in complex network environments [5]. Improving the accuracy and timeliness of delay prediction has therefore emerged as a key objective in the ongoing effort to optimize DCS performance.

To address this challenge, this study proposes a prediction acceleration framework designed for embedded

distributed control scenarios. The framework integrates two complementary technologies: the nonlinear modeling capabilities of neural networks for time-series prediction, and the high-performance, low-latency characteristics of Field Programmable Gate Array (FPGA). Neural networks have already demonstrated superior performance compared to traditional models and support vector machines in various time-sensitive domains such as financial forecasting and fault prediction [6–8]. Their primary advantage lies in their ability to automatically learn long-term dependencies and adapt to non-stationary trends in sequential data. This makes them particularly effective at handling diverse and heterogeneous delay patterns. Meanwhile, FPGA offers a powerful solution for accelerating neural network inference through parallel pipelined hardware architectures. Unlike software-based deployments, which often suffer from computational bottlenecks, FPGA implementations enable real-time processing and are especially well-suited for industrial environments with stringent response time requirements. By combining these two technologies, the proposed framework not only enhances the real-time performance of delay prediction but also supports lightweight and efficient deployment on resource-constrained control nodes. This integration forms a closed-loop optimization system that spans from model design to runtime execution.

Accordingly, this study presents an FPGA-accelerated delay prediction system based on Long Short-Term Memory (LSTM) neural networks. Its effectiveness is validated through experiments on several representative control datasets, with comprehensive evaluations focusing on prediction accuracy, latency, and resource efficiency.

2 Related work

Numerous scholars have explored delay-related challenges in DCSs from various perspectives. Dai et al. identified that communication delays in DCSs primarily stem from network transmission and task scheduling, noting that the unpredictability of these delays significantly undermines the timeliness and precision of control tasks [9]. Caballero-Águila and Linares-Pérez examined the impact of delay on the consistency of industrial automation control systems. They proposed a delay-tolerant task scheduling algorithm, which notably enhanced system robustness within specific operational limits [10]. FPGA has gained popularity in control systems due to their low latency and versatile capabilities. Tiong et al. highlighted the advantages of FPGA's parallel processing and reconfigurability, demonstrating their effectiveness in meeting the stringent real-time requirements of embedded control applications [11]. Similarly, Sridevi and Saifulla implemented a real-time signal processing architecture using FPGA technology. Their approach significantly improved the operational efficiency of industrial control systems by enabling immediate data stream processing [12]. Neural networks have shown strong performance in time-series prediction and are considered promising tools for addressing delay issues in DCSs. Herrera et al. demonstrated that LSTM neural networks effectively captured temporal dependencies in delay data, leading to substantial improvements in prediction accuracy [13]. Likewise, Olabi et al. proposed a latency prediction model based on convolutional neural networks (CNNs), which achieved real-time performance by extracting multi-dimensional features from the delay sequences [14].

Although many researchers have explored delay prediction in control systems, most existing methods suffer from limited accuracy, poor real-time performance, or challenges in deployment on embedded systems. Some studies have employed traditional statistical models or machine learning approaches and have achieved certain results on small-scale datasets. However, these methods often exhibit poor robustness and weak generalization when applied to large-scale, non-stationary, and noise-prone DCSs. In recent years, deep neural networks—particularly recurrent neural networks—have emerged as mainstream tools for time-series prediction and have been adopted in some edge computing tasks. Nonetheless, their deployment efficiency and real-time inference capabilities remain constrained by hardware limitations. A comparative overview of existing delay prediction models is presented in Table 1:

Table 1: Comparison of existing delay prediction models

Model Name	Core Method	Dataset Used	Performance Metrics	Key Limitations
AutoRegressive Integrated Moving Average (ARIMA)	Linear time-series model with autoregressive and moving average components	Power load monitoring data	MAE, MSE	Suitable only for stationary time series; struggles with nonlinear or abrupt delay patterns
Support Vector Machine (SVM)	Support vector regression; suitable for small-sample prediction	Industrial control network delay data	RMSE, R^2	Strong dependence on feature selection; complex hyperparameter tuning; limited generalization
Random Forest for Latency Prediction (RF-LP)	Ensemble regression using multiple decision trees	Industrial control task scheduling logs	MAE, MAPE	High training overhead; sensitive to noise and outliers
Deep Autoregressive Model (DeepAR)	Probabilistic model based on LSTM	E-commerce server request data	ND, NRMSE	Complex structure; long inference time;

				high deployment barrier
Temporal Convolutional Network (TCN)	One-dimensional causal convolutional neural network for modeling long-term dependencies	Communication network link delay sequences	MAE, Latency Error	High accuracy but rigid architecture; not suitable for embedded deployment

Current mainstream methods generally fall into three categories:

(1) Statistical approaches, such as ARIMA, which are effective for linear and stationary data but fail to capture complex patterns;

(2) Traditional machine learning methods, such as SVM and RF, which are sensitive to input features and lack real-time capabilities;

(3) Deep learning models, including LSTM and TCN, which offer high prediction accuracy but pose significant challenges in deployment and demand substantial hardware resources.

Furthermore, most studies rely on static logs or simulated datasets for evaluation, lacking support from industrial-grade, dynamically changing data sources. This limits the objectivity and generalizability of model assessments. More critically, there remains a gap in the literature regarding a comprehensive delay prediction system that integrates neural network models with hardware platforms—one that balances accuracy, real-time performance, and resource efficiency. To address this gap, this study proposes an LSTM-based prediction model deployed on an FPGA platform. The goal is to provide high-accuracy, low-latency, and embedded-compatible delay prediction capabilities for DCSs.

3 Related theory and model design

3.1 Latency prediction

Latency refers to the time it takes for a message or data packet to travel from one end of a network to the other. It typically includes four components: propagation latency, processing latency, transmission (sending) latency, and queuing latency [15]. The overall latency can be expressed as the sum of these four elements. In most practical scenarios, propagation latency and transmission latency are the primary contributors. For longer messages, transmission latency becomes the dominant factor, while propagation latency plays a greater role for shorter messages. More precisely, latency is defined as the time interval between the arrival of the first bit of a packet at a router and the departure of the last bit from that router [16–18]. During testing, latency is commonly measured as the time interval from when a test instrument sends a data packet to when the packet is received. It is closely related to the packet length and is typically evaluated within the throughput range of the router port. Testing beyond this range is considered uninformative and not reflective of actual performance conditions [19]. Time-series prediction is a critical technique for latency forecasting. Its theoretical foundation includes feature analysis of historical data, the design of prediction models, and the selection of appropriate evaluation metrics. Latency data often exhibit complex temporal patterns, such as trends, seasonality, and random fluctuations. By analyzing these patterns, it is possible to uncover the underlying structure of delay variations [20–22]. Various models are available for time-series forecasting, including both traditional statistical methods and modern machine learning approaches. Among these, neural networks have emerged as the preferred choice for latency prediction due to their ability to model nonlinear temporal dependencies effectively. Common evaluation metrics for latency prediction models include Mean Squared Error (MSE), Mean Absolute Error (MAE), and prediction accuracy, all of which help quantify model performance in terms of accuracy and robustness [23].

The theoretical framework for latency prediction plays a pivotal role in optimizing the performance of DCSs. In this study, a neural network-based latency prediction model is proposed to meet the practical demands of distributed systems. The model leverages hardware acceleration to improve both the real-time performance and the predictive accuracy of latency estimation.

3.2 Design of the DCS

The design theory of DCS) encompasses several key areas, including system architecture, communication mechanisms, and task scheduling strategies [24–26]. To meet the stringent demands of complex control environments—such as real-time responsiveness, reliability, and adaptability—multi-level theoretical models and technical frameworks are integrated to construct a highly coordinated and efficient system structure. Among these components, the communication mechanism plays a central role in enabling collaborative operations among distributed nodes [27]. Its theoretical foundation primarily involves real-time communication protocols, network scheduling and priority management, and distributed consistency models. Real-time control applications impose strict latency requirements on communication mechanisms. To address this, protocols such as Controller Area Network (CAN), Ethernet for Control Automation Technology (EtherCAT), and Industrial Ethernet are commonly employed, offering high reliability and low-latency data transmission. Network scheduling and priority management theories focus on assigning appropriate priorities to tasks and efficiently allocating data transmission resources. This ensures that time-critical tasks are executed promptly, maintaining system performance under varying workloads. Distributed consistency theory emphasizes the importance of maintaining data consistency across nodes. In scenarios where multiple nodes share task states and data, ensuring

consistency is critical to preventing execution failures and systemic errors.

3.3 Neural networks and FPGAs integration architecture

To achieve high-precision, low-latency prediction of task response delays in DCSs, this study proposes a prediction model based on a LSTM neural network architecture accelerated by an FPGA platform. A comprehensive co-optimization of both software and hardware components has been carried out. The model not only focuses on accurately predicting latency but also emphasizes its feedback effect on task scheduling and overall system stability, thereby facilitating holistic DCS performance optimization.

During DCS operation, system response latency can be broken down into the following components:

(1) Sending latency – the time from when a task is dispatched by the scheduler to when it reaches the network module;

(2) Propagation latency – the time required for a data packet to travel through the communication channel, influenced by bandwidth and network topology;

(3) Queuing latency – the waiting time a task experiences in the scheduling buffer before processing;

(4) Processing latency – the time a node takes to compute and complete the assigned task.

These delays directly impact task completion times, which in turn affect scheduling order, priority allocation, and the overall stability of the system. In this work, total response latency is modeled as a unified prediction target, with its constituent components implicitly embedded within the input features. The neural network is designed to automatically learn the underlying relationships among these factors. The goal of the model is to predict the total latency of a task, which is defined as the time from its reception to completion. This prediction is based on historical task data and system states collected within a sliding time window. The results can support decisions on task prioritization and resource allocation.

To simulate the dynamics of task loads and delay variations in a DCS, this study employs the publicly available Electricity Transformer Temperature (ETT) - small dataset. Features such as timestamps, voltage, current, and load power serve as proxies for the state variations in a real-world task scheduling system. The response delay is defined as the difference between the task's reception time and its feedback time, and this value serves as the prediction target. Input features include a 10-dimensional sequence of historical task response times, task type and priority (encoded via one-hot encoding), and current system metrics such as GPU utilization, queue length, and network congestion level. Samples are constructed using a sliding window approach, where each sample input consists of 10 time steps of sequential information used to predict the latency at the next time step. To enhance training stability and improve model generalization, all feature values are normalized to the [0,1] range using Min-Max Scaling. Additionally, outliers are removed using the Interquartile Range (IQR) method to reduce noise and improve robustness.

The neural network developed in this study adopts a single-layer LSTM architecture to model the dynamic behavior of sequential tasks. The input layer has 16 dimensions, comprising 10 dimensions of historical delay data and 6 dimensions representing system state. The LSTM hidden layer consists of 128 units, and the output layer includes a single neuron with a linear activation function to produce continuous prediction values. Internally, the LSTM uses Sigmoid and Tanh activation functions to regulate the behavior of the forget gate, input gate, and output gate. The model is trained using the MSE as the loss function and the Adam optimizer with a learning rate of 0.001. Training is performed in mini-batches of size 64 for up to 100 epochs. An EarlyStopping strategy is applied to halt training if validation performance fails to improve over consecutive epochs. The model outputs the predicted delay for the next time step, which can be used to support task scheduling, task ordering, and node priority adjustment in distributed control environments. To ensure efficient deployment and real-time performance in practical DCS scenarios, the trained LSTM model is deployed on the Xilinx Alveo U250 FPGA acceleration platform. Logic synthesis, placement and routing, and bitstream generation are conducted using the Vivado platform. The model structure is translated into multiple hardware modules capable of parallel execution. The matrix multiplications within the LSTM are accelerated using the FPGA's internal Digital Signal Processing (DSP) blocks, while the nonlinear activation functions are implemented using lookup tables to minimize resource consumption. Intermediate state variables are stored in Block RAM (BRAM) to support time-step state propagation. Neural network weights are quantized into 8-bit fixed-point representation using the following quantization equation:

$$\widehat{\mathrm{W}} = \mathrm{round}(\mathrm{W} \cdot 2^{q})/2^{q} \quad (1)$$

W refers to the weight matrix of the neural networks; $\widehat{\mathrm{W}}$ represents the quantized weight matrix; q is the number of quantization bits, and then the model mainly performs matrix multiplication. This quantization strategy effectively reduces the bit width and latency of multiplication operations, thereby improving inference efficiency. Furthermore, the data input/output interface and control modules are connected to the main control module via a shared bus, enabling real-time interaction between the input prediction stream and the feedback results.

The optimized model architecture in this study is presented in Figure 1:

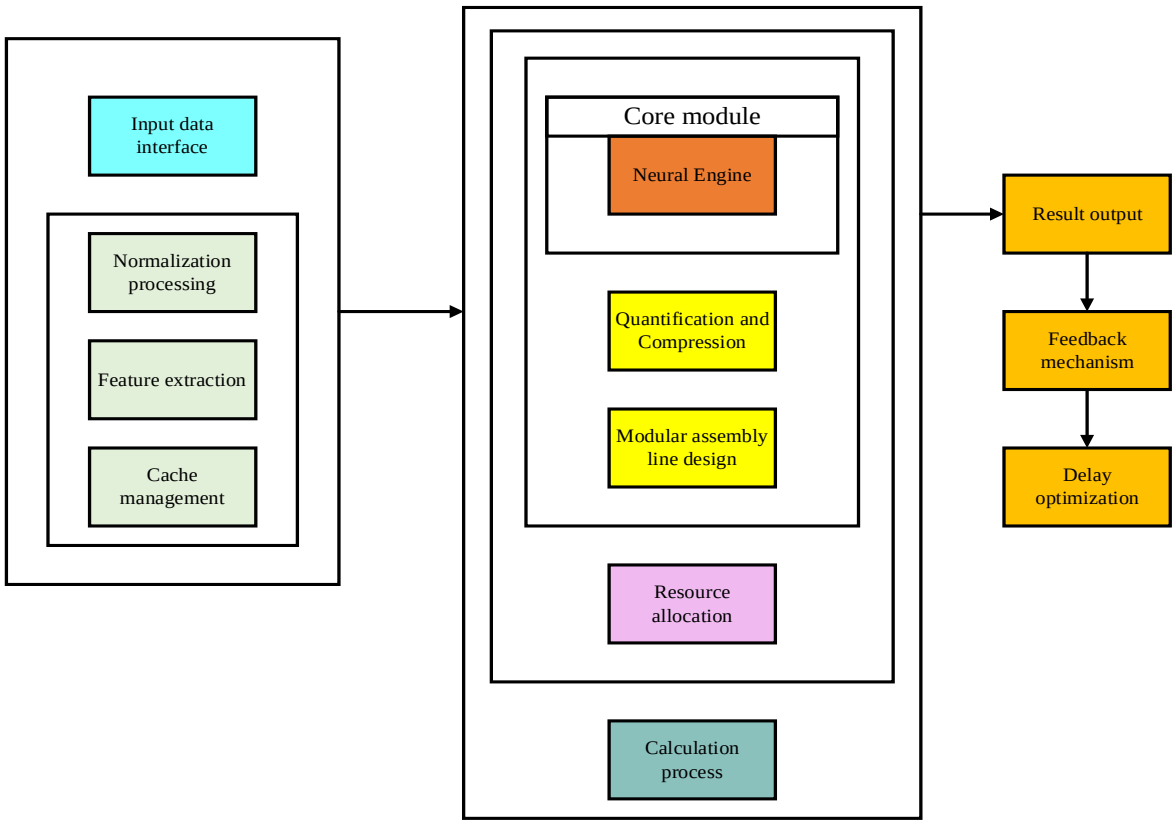


Figure 1: Architecture of the optimized model

The "Neural Engine" is the core computational unit of the system. Its internal structure includes the LSTM gating unit computation module, quantization and dequantization modules, state register module, and activation function look-up table (LUT) module. The LSTM Core Unit performs time-step calculations in a pipeline, with the gating units operating in parallel. The Quantization Module compresses the input signals and weights. The State Register caches the target for the current step, enabling state transitions across time steps. The activation function is quickly implemented by the LUT module to perform nonlinear mapping. The entire module is connected to the external bus, supporting high-speed data exchange with the controller and scheduler. The module structure is highly configurable, supporting parameter reuse and adjustable precision settings, making it adaptable to various deployment scenarios.

To further enhance the model's adaptability and the system's intelligence, an online update mechanism is introduced after the model is deployed. When the real response latency from the control system deviates significantly from the model's prediction, the system will locally fine-tune the model parameters via backpropagation, using the Adam optimization algorithm for incremental updates. The update strategy is described by Equations (2) to (4).

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla L(W) \quad (2)$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla L(W))^2 \quad (3)$$

$$W = W - \alpha \frac{m_t}{\sqrt{v_t}+\epsilon} \quad (4)$$

The exponential decay average of momentum and gradient squared is denoted as m_t and v_t ; $\nabla L(W)$ indicates the gradient for weights; α refers to the learning rate; β_1 and β_2 are exponential decay coefficients; ϵ represents a random number, to prevent the denominator from becoming 0. It can be observed that the integrated design of FPGA and neural networks has established a solid mathematical foundation at both the theoretical and implementation levels, ensuring the real-time, efficient, and accurate performance of the system. The pseudo code of the optimized model is as follows:

```
# Pseudocode: LSTM-Based Latency Prediction with FPGA Deployment
Initialize:
    Load quantized LSTM weights W_q from on-chip memory
    Initialize hidden state h_0 and cell state c_0 to zeros
    Set time window size w = 10
    Define learning rate α, momentum decay β1, β2, ε
# Inference Loop (on FPGA)
Function LSTM_Predict(x_seq):
    h, c ← h_0, c_0
    for t in 1 to w:
        # Gate calculations (in parallel on FPGA)
        f_t ← sigmoid(W_f * x_t + U_f * h + b_f)
        i_t ← sigmoid(W_i * x_t + U_i * h + b_i)
        o_t ← sigmoid(W_o * x_t + U_o * h + b_o)
        g_t ← tanh(W_c * x_t + U_c * h + b_c)
        # Cell and hidden state update
        c ← f_t ∘ c + i_t ∘ g_t
        h ← o_t ∘ tanh(c)
        # Final prediction
    y_pred ← W_out * h + b_out
```

```
    return y_pred
# On-chip Online Update (optional)
Function Online_Update(y_true, y_pred):
    loss ← (y_true - y_pred)^2
    # Compute gradients (simplified)
    grad_W ← ∂loss/∂W using backpropagation
    m_t ← β1 * m_{t-1} + (1 - β1) * grad_W
    v_t ← β2 * v_{t-1} + (1 - β2) * (grad_W)^2
    # Update weights
    W_q ← W_q - α * m_t / (sqrt(v_t) + ε)
    Store updated W_q to on-chip memory
# Main control loop
While system is running:
    Collect current input sequence x_seq from sensors and system state
    y_pred ← LSTM_Predict(x_seq)
    Send y_pred to scheduling module
    Receive feedback y_true
    If |y_true - y_pred| > δ:
        Online_Update(y_true, y_pred)
```

4 Experimental design

To validate the applicability and performance of the proposed neural network and FPGA-accelerated delay prediction model in DCSs, this study uses the ETT dataset as the experimental data source. This dataset was collected by the power monitoring systems of two regions in a Chinese province. It records the oil temperature, load conditions, and external disturbance features of power transformers during actual operation, spanning two years with good continuity and high-frequency characteristics. It is a typical industrial-grade time series dataset.

The original features of the ETT dataset include: date, timestamp, load power, oil temperature, ambient temperature, target forecast values, and six external load influence factors (such as power flow direction, voltage level, etc.). These variables are highly correlated with task scheduling delays, response time fluctuations, and environmental disturbance intensity in DCSs, and thus serve as a good proxy for the operational state of a real DCS. Therefore, the ETT dataset structurally aligns with and maps to the delay prediction task of a DCS, making it suitable for training neural network prediction models. The ETT dataset is available for download from the official website (https://github.com/zhouhaoyi/ETDataset/blob/main/README_CN.md). Additionally, the dataset is clearly divided into multiple scenarios and supports sequence modeling at different time granularities (minute-level, hour-level), facilitating comparative research on algorithms with varying complexities and precision requirements. For this study, three subsets at the minute-level were selected, as described below:

(1) ETT-small-m1: Region 1, record per minute.

(2) ETT-small-m2: Region 2, record per minute.

(3) ETT-small-m3: Region 3, record per minute.

The dataset contains approximately 70,080 records per minute (2 years x 365 days x 24 hours x 60 minutes). The study also set the experimental environment, as exhibited in Table 2:

Table 2: Experimental environment

Configuration	type
Computing device	Dell PowerEdge R740
Central Processing Unit (CPU)	Intel Xeon Gold 6230, 2.1GHz, 20 cores
Graphics Processing Unit (GPU)	NVIDIA Tesla V100, 16GB memory
Memory	128GB DDR4
Storage device	Samsung PM981a NVMe SSD
Prediction Target	Task Response Latency (0.01 ms)
Neural Network Output	Predicted Total Response Latency of the Next Task Scheduling Cycle
Prediction Target Deployment Use	Used for Dynamic Task Scheduler Sorting and Control Strategy Optimization, Achieving Real-time Performance Improvement of the DCS System
FPGA development board	Xilinx Alveo U250

The optimized model parameters in this study were determined by considering several factors. These include the dynamic characteristics of task response latency in DCSs, resource limitations in the deployment environment, and the real-time requirements of the prediction target. Firstly, regarding the input features, the model uses 6 dimensions, covering key control variables such as historical task latency, task type encoding, CPU utilization, cache queue length, network link load rate, and scheduling priority. These features provide a comprehensive reflection of the factors influencing response latency during task scheduling and resource allocation in control systems. In terms of network structure, the model adopts a two-layer LSTM architecture, with 128 hidden units per layer. This design is based on the characteristics of task latency in DCSs, which exhibit medium- to short-term temporal dependencies. The two-layer structure effectively models the system's dynamics, while the 128-dimensional state representation achieves a good balance between prediction accuracy and FPGA deployment resource efficiency. The output layer has 1 dimension, using a linear activation function. The output value represents the predicted response latency for the next time step (in milliseconds), making it suitable for direct use by the controller in task prioritization and scheduling strategy decisions. During training, the Adam optimizer is used with an initial learning rate of 0.001, a first-order momentum decay coefficient of 0.9, and a second-order momentum decay coefficient of 0.999. This setting has shown good convergence in various non-stationary time series modeling tasks, making it especially suitable for short-term fluctuations and local trend changes in industrial control data. Additionally, gradient clipping with a threshold set to 5 is applied during training to suppress gradient explosion caused by anomalous data, improving the overall stability and robustness of the training process. The batch size is set to 64, balancing training efficiency and model generalization, while also considering the cache scheduling efficiency and throughput requirements when running on the FPGA platform. In the comparative experiments, to ensure fairness and comparability in model evaluation, the implementation environment and parameter settings for all comparison models were standardized. Except for the

proposed optimized model deployed on the FPGA platform, the other models ran on the same hardware and software platform, using the same data processing flow, and the same training and validation dataset splitting strategy.

The Distributed Model Predictive Control (DMPC) model was executed on an Ubuntu 20.04 system environment. The control algorithm was implemented using Python and primarily used to simulate the predictive scheduling process of distributed tasks across multiple nodes. The control horizon was set to 50 time units, with a control step size of 10 steps, and a receding horizon optimization strategy was employed. The model used historical system states, task loads, and resource utilization (RU) as inputs and performed rolling prediction and latency estimation based on a quadratic programming solver. In the experiments, the DMPC model was deployed on a workstation equipped with an Intel Core i7-11700 processor and 16GB of RAM. The Multi-Agent System-Based Distributed Control Model (MAS-DCM) was jointly developed using Python and Java Agent Development Environment (JADE), simulating information exchange and local prediction behaviors among multiple agents. Each agent maintained an independent task estimator, utilizing local observations for latency prediction and scheduling feedback. The model was implemented within the Robot Operating System (ROS) framework, supporting task state broadcasting and multi-node coordination. Communication between agents followed a publish/subscribe mechanism, with each scheduling synchronization cycle set to 500 ms. The model was deployed on an NVIDIA Jetson Xavier NX platform (ARM Cortex-A57 CPU + 8GB RAM), simulating a low-power edge deployment environment. The Field Programmable Gate Array-Based Deep Neural Network Acceleration Model (FPGA-DNNAM) followed recent best practices for edge deployment. It used a four-layer multilayer perceptron (MLP) for latency prediction, with a network structure of [Input dimension 16 → 64 → 32 → 1]. All hidden layers used Rectified Linear Unit (ReLU) activation functions, while the output layer employed a linear function. The model was trained using the PyTorch 1.13 framework over 100 epochs with the support of an NVIDIA RTX 3060 GPU. The Adam optimizer was used with an initial learning rate of 0.001 and a batch size of 64. After training, the model was quantized and deployed using the Xilinx Vitis AI toolchain and was ultimately implemented on a Xilinx ZCU104 FPGA board to enable high-performance inference. To ensure fairness, all models mentioned above used the same ETT-small dataset, following a unified normalization process and sample construction pipeline.

This study did not employ conventional statistical methods such as confidence interval estimation, hypothesis testing, or significance analysis during experimental design and performance evaluation. The core objective of the study was not to perform statistical inference on minor performance differences among models. Instead, the goal was to develop a deployable and scalable optimization framework for task latency prediction in DCSs. The study focuses more on the practical effectiveness of predictions, system adaptability, and deployment feasibility under real-world engineering conditions. A comprehensive evaluation of model performance was conducted using a multi-dimensional metric system from the perspective of actual operational scenarios. Unlike typical machine learning studies that focus on statistical significance, this study emphasizes real-world usability and consistent improvements in system performance. It prioritizes practical enhancements rather than probabilistic reasoning over numerical differences. Additionally, the study does not provide a systematic modeling or categorization of all failure modes in control systems. The latency prediction model itself does not directly address low-level device fault diagnosis or control logic protection. Failures in DCS environments are often the result of complex interactions between hardware conditions, communication errors, and environmental disturbances. As a result, scenario-agnostic analysis is less meaningful. This study focuses on the impact of latency prediction deviation on task scheduling efficiency and system stability. It indirectly reflects the model's robustness using indicators such as task failure rate and mean time between failures (MTBF). More comprehensive fault modeling and coordinated control mechanisms are proposed as future research directions.

In summary, the omission of statistical significance testing and system-level fault modeling in this study is not a result of insufficient methodological rigor. Instead, it is a deliberate choice, driven by the study's scope, system-level objectives, and application-focused approach. The goal is to ensure that the study remains centered on the development of a deployable, verifiable latency prediction model and its integration into practical control systems.

5 Experimental comparison of latency prediction of the dcs based on FPGA and neural networks

5.1 Performance evaluation of the latency prediction model

To comprehensively evaluate the practical effectiveness of the proposed model in DCSs, this study categorizes performance evaluation metrics into four key dimensions: accuracy, robustness, timeliness, and resource efficiency. Within each dimension, two representative evaluation metrics are selected, resulting in a total of eight core indicators. This multi-faceted approach enables a systematic analysis of the model from various perspectives. In the accuracy dimension, the Mean Absolute Error (MAE) and MSE are adopted as primary evaluation metrics. MAE measures the average level of absolute error in the prediction process, offering intuitive interpretability. MSE, by squaring the prediction errors, emphasizes larger deviations and effectively reflects the model’s ability to control errors at outlier points. This dimension provides a direct assessment of the model's fitting accuracy with respect to task response latency and serves as the foundation for evaluating prediction quality. The robustness dimension includes the Root Mean

Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE). RMSE, as the square root of MSE, maintains the same unit as the original data and is suitable for presenting the overall fluctuation intensity. MAPE considers the ratio of prediction error to the actual value, reflecting the model's stability and adaptability across different data scales, particularly under varying load intensities in DCS scenarios. This dimension is designed to evaluate the model's stable prediction capability under data perturbations and unexpected conditions. For the timeliness dimension, to verify whether the model is suitable for deployment in latency-sensitive control systems, two indicators are introduced: Mean Prediction Latency (MPL) and Maximum Prediction Latency (Max-PL). MPL represents the average time required for a single prediction, while Max-PL indicates the longest inference delay that may occur under extreme computational pressure. These metrics are directly related to the model's real-time inference capability and serve as core criteria for determining its viability in online DCS applications. In the resource efficiency dimension, RU and Throughput Per Second (TPS) are used as the primary metrics. RU denotes the proportion of logical resources consumed by the model when deployed on FPGA or embedded systems, providing a critical reference for deployment feasibility under resource constraints. TPS reflects the number of prediction tasks the model can complete per unit of time, directly indicating its execution efficiency under high-concurrency scheduling scenarios. This dimension comprehensively evaluates the deployment cost and runtime performance-to-cost ratio of the model.

The comparison results of accuracy dimensions are presented in Figure 2:

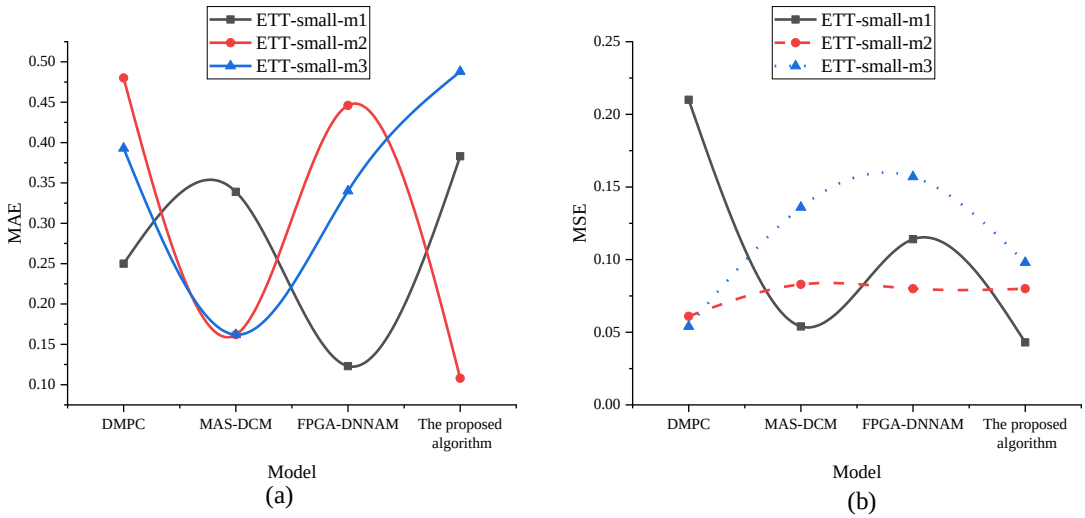


Figure 2: Accuracy dimension experimental result ((a): MAE; (b): MSE)

Figure 2 shows that the proposed optimized model achieves a MAE score of 0.108 on the ETT-small-m2 dataset. This significantly outperforms other models, such as DMPC's 0.480 and FPGA-DNNAM's 0.446. However, on the ETT-small-m1 and ETT-small-m3 datasets, the proposed model's MAE is 0.383 and 0.488, respectively. These values are higher than FPGA-DNNAM's 0.123 and 0.340, indicating slightly inferior performance. In terms of MSE, the proposed optimized model performs well across all datasets. It achieves scores of 0.043 (ETT-small-m1), 0.080 (ETT-small-m2), and 0.098 (ETT-small-m3). These are significantly lower than DMPC (e.g., 0.210 on ETT-small-m1) and MAS-DCM (e.g., 0.136 on ETT-small-m3). The proposed model has a notable advantage in the MSE metric, indicating better control over overall prediction errors. While its MAE performance is slightly lower than FPGA-DNNAM on some datasets, its overall performance remains stable and accurate. The comparison results for robustness are shown in Figure 3.

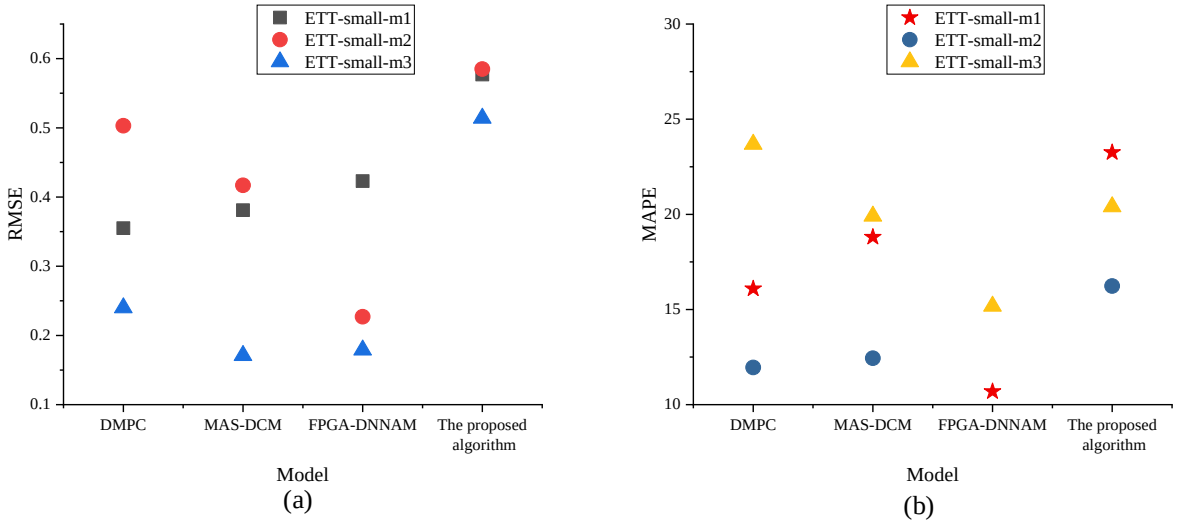


Figure 3: Robustness dimension experimental result ((a): RMSE; (b): MAPE)

The proposed model has RMSEs of 0.577 (ETT-small-m1), 0.585 (ETT-small-m2), and 0.514 (ETT-small-m3). These are higher than FPGA-DNNAM's 0.179 on ETT-small-m3 and MAS-DCM's 0.381 on ETT-small-m1. However, the overall difference with other models is not significant. On the ETT-small-m1 dataset, the proposed model's MAPE is 23.250, slightly higher than FPGA-DNNAM's 10.688 and MAS-DCM's 18.803. On the ETT-small-m2 and ETT-small-m3 datasets, the proposed model achieves MAPEs of 16.234 and 20.401, respectively, outperforming DMPC's 23.685. Overall, the proposed optimized model shows good robustness in both RMSE and MAPE. Despite a slightly higher RMSE due to outliers, it demonstrates reasonable control of MAPE. This indicates the model's adaptability to complex real-world scenarios. The results for the timeliness dimension are presented in Figure 4.

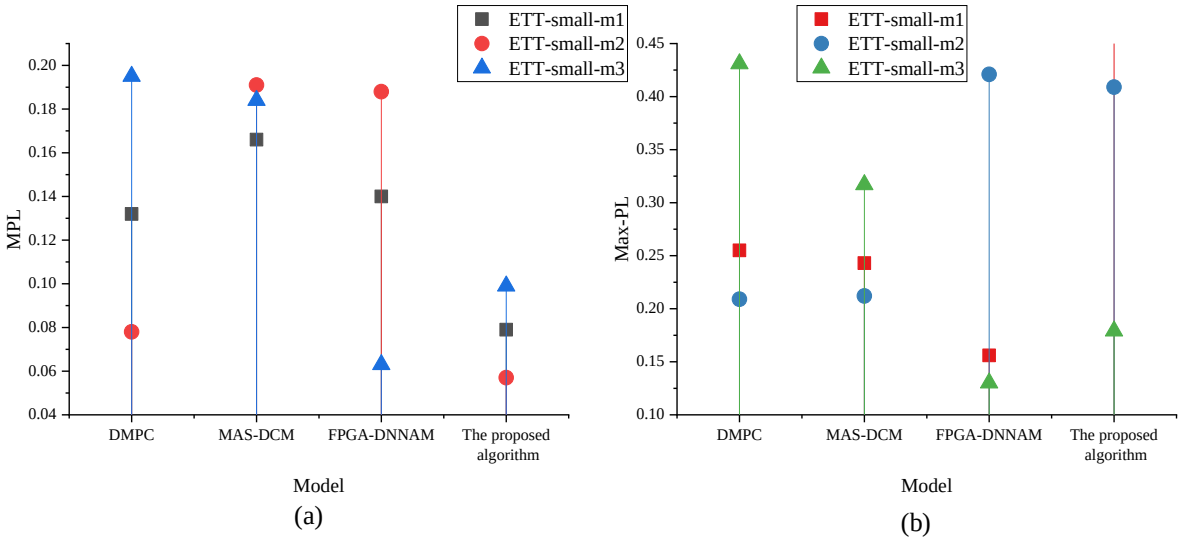


Figure 4: Timeliness Dimension experimental result ((a): MPL; (b): Max-PL)

Figure 4 shows that the proposed optimized model achieves the lowest MPL across all datasets. Specifically, it records scores of 0.079 for ETT-small-m1, 0.057 for ETT-small-m2, and 0.099 for ETT-small-m3. In comparison, the MPL values of DMPC and MAS-DCM both exceed 0.130. Although FPGA-DNNAM achieves a slightly lower latency of 0.063 on the ETT-small-m3 dataset, it remains close to the proposed model. For Max-PL, the proposed model again outperforms the baselines, achieving 0.495 (ETT-small-m1), 0.409 (ETT-small-m2), and 0.179 (ETT-small-m3). These values are consistently lower than DMPC's 0.431 on ETT-small-m3 and MAS-DCM's 0.212 on ETT-small-m2. Overall, the proposed model demonstrates excellent timeliness. It effectively controls both average and peak latency, making it highly suitable for DCSs applications where real-time

performance is critical. The results of resource efficiency analysis are presented in Figure 5:

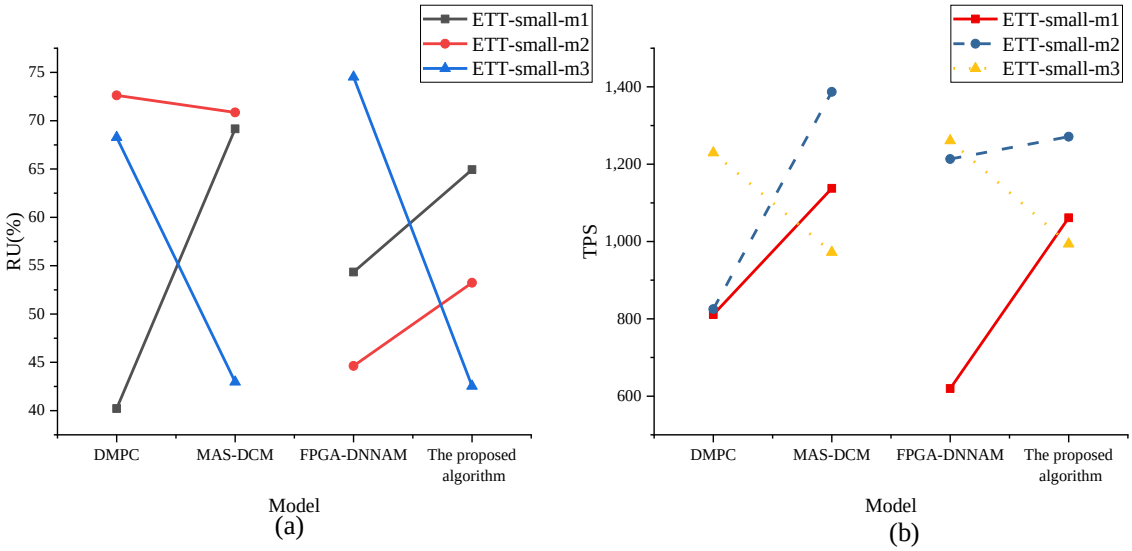


Figure 5: Resource Efficiency Dimension experimental result ((a): Resource Utilization; (b): Throughput)

Figure 5 shows the RU rates for the proposed optimized model across three datasets: 64.932% for ETT-small-m1, 53.236% for ETT-small-m2, and 42.542% for ETT-small-m3. While these rates are slightly higher than FPGA-DNNAM's 44.635% on the ETT-small-m2 dataset, they remain lower than MAS-DCM's 69.160% on ETT-small-m1, indicating a balanced performance in terms of resource usage. The model also excels in processing speed, with the number of samples processed per second reaching 1061.277 (ETT-small-m1), 1270.967 (ETT-small-m2), and 993.796 (ETT-small-m3). These figures significantly surpass DMPC's 810.982 (ETT-small-m1) and FPGA-DNNAM's 1260.785 (ETT-small-m3), demonstrating the model's efficiency in handling high throughput. Overall, the proposed optimized model performs exceptionally well in terms of resource efficiency. It achieves a high processing speed while maintaining effective control over hardware resource consumption, making it suitable for deployment in environments with limited resources but a need for efficient processing.

In terms of key performance dimensions—accuracy, timeliness, and resource efficiency—the model stands out, particularly with notable advantages in metrics such as MSE, MPL, and throughput. While its robustness is slightly lower compared to some other models, it still maintains good stability. This suggests that the proposed model is well-suited for general distributed control tasks, particularly in complex environments that demand high real-time performance and efficiency.

5.2 Testing of system operation effectiveness

In the system performance dimension, this study selects four key indicators: Task Completion Rate (TCR), Average Response Time (ART), Task Failure Rate (TFR), and System Throughput (ST). TCR reflects the proportion of tasks successfully completed within a given period and serves as a fundamental metric for evaluating the stability and controllability of the scheduling system. ART measures the average time from task reception to completion, indicating the system's real-time responsiveness. TFR assesses the proportion of tasks that fail to complete due to scheduling errors, system congestion, or control delays, offering valuable insights for optimizing scheduling strategies. ST quantifies the number of tasks processed by the system per unit of time, representing the system's efficiency under high concurrency. This dimension focuses on the quality of the system's load-handling response during real-world operation and serves as a core metric for assessing improvements in control system efficiency post-deployment. In the system reliability dimension, to evaluate the impact of model deployment on long-term operational stability, four additional indicators are introduced: MTBF, Mean Time to Repair (MTTR), Error Rate (ER), and System Availability (SA). MTBF indicates the average duration the system can operate normally between two consecutive failures—a higher value suggests greater system stability. MTTR measures the average time required for the system to return to normal operation following a failure, reflecting the responsiveness of fault recovery mechanisms. ER captures the proportion of errors arising during operation due to issues such as data transmission failures or module malfunctions, serving as a key metric for system robustness. SA evaluates the percentage of total operational time during which the system remains available, offering a comprehensive measure of system stability and sustained service capability. This dimension emphasizes the system's resilience to anomalies, unexpected failures, and uncertain disturbances during extended operation.

The comparison results of system performance dimensions are revealed in Figure 6:

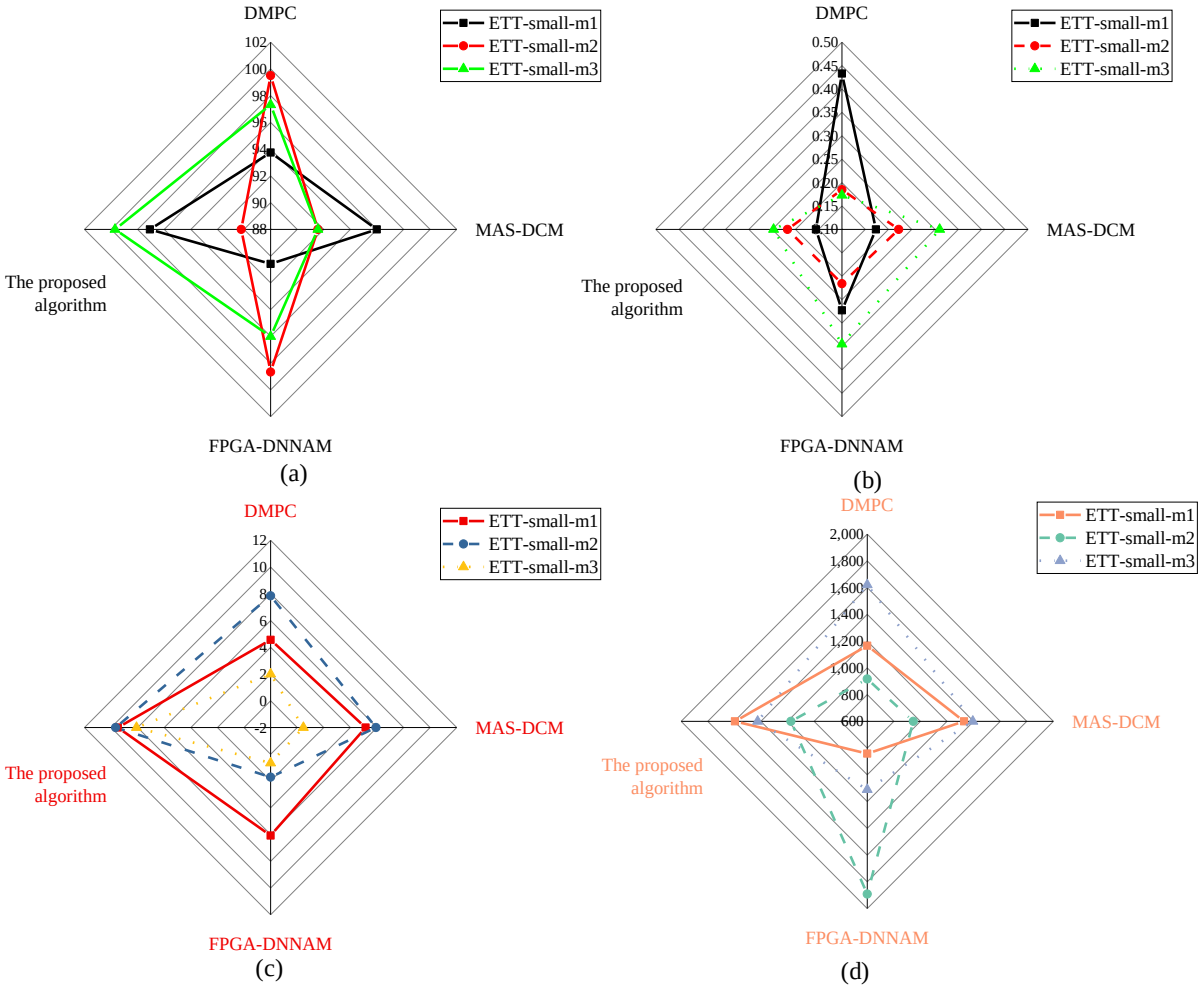


Figure 6: System Performance Dimension experimental result ((a): Task Completion Rate; (b): Average Response Time; (c): Task Failure Rate; (d): System Throughput)

Figure 6 presents the performance of the proposed optimized model in terms of task completion rate, average response time, task failure rate, and system throughput. On the ETT-small-m3 dataset, the model achieves a task completion rate of 99.699%, significantly higher than DMPC (97.320%) and MAS-DCM (91.560%). However, on the ETT-small-m2 dataset, its task completion rate drops to 90.206%, which is lower than FPGA-DNNAM's 98.662%. In terms of average response time, the proposed model performs well across all datasets. It records response times of 0.156 (ETT-small-m1), 0.217 (ETT-small-m2), and 0.247 (ETT-small-m3). Although slightly slower than DMPC's 0.185 on ETT-small-m2, it still outperforms MAS-DCM and FPGA-DNNAM overall. Regarding task failure rate, the model records 9.489% on the ETT-small-m1 dataset. While this is higher than DMPC (4.561%) and FPGA-DNNAM (6.075%), the failure rate improves on ETT-small-m3, decreasing to 8.084%. Despite this progress, further optimization is needed in this area. For system throughput, the proposed model delivers strong results. It achieves 1595.027 tasks per second on the ETT-small-m1 dataset, far exceeding DMPC's 1165.537. On the ETT-small-m3 dataset, throughput reaches 1424.082 tasks per second, slightly surpassing MAS-DCM's 1394.212. In summary, the proposed optimized model excels in task completion rate and system throughput, showing strong suitability for high-efficiency applications. Its average response time also reflects good real-time performance. However, improvements in reducing the task failure rate remain an area for future work. The comparison results for system reliability are illustrated in Figure 7:

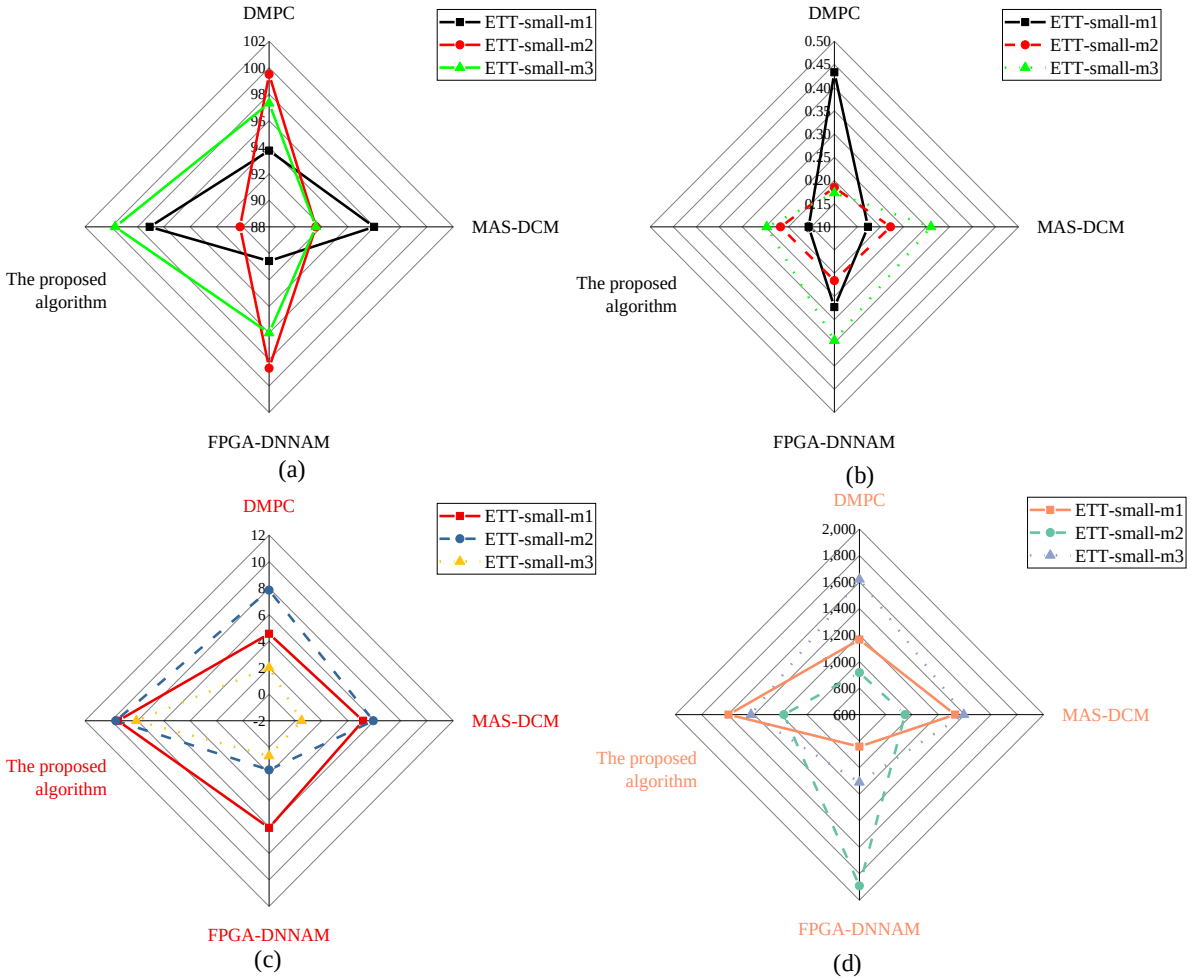


Figure 7: System Reliability Dimension experimental result ((a): Mean Time between Failures; (b): Mean Time to Repair; (c): Error Rate; (d): System Availability)

Figure 7 illustrates the reliability metrics of the proposed optimized model across different datasets. On the ETT-small-m1 dataset, the model achieves a MTBF of 178.393 hours—substantially lower than MAS-DCM's 410.053 hours and DMPC's 318.684 hours. On the ETT-small-m3 dataset, the MTBF increases to 230.132 hours, though it still falls short of the comparative models. The model's MTTR on the ETT-small-m1 dataset is 0.198 hours, which is slightly higher than FPGA-DNNAM's 0.071 hours. However, on the ETT-small-m3 dataset, the MTTR improves to 0.080 hours, aligning closely with other models in this category. In terms of error rate, the proposed model demonstrates strong performance, achieving the lowest value on the ETT-small-m3 dataset at just 0.318%, significantly outperforming FPGA-DNNAM (4.316%) and DMPC (3.534%). On the ETT-small-m1 dataset, the model records an error rate of 3.116%, comparable to MAS-DCM's 3.645%. The proposed model also maintains high availability across all datasets, exceeding 97% in every case. Notably, it reaches 98.855% on the ETT-small-m2 dataset, slightly below MAS-DCM's 99.436%. Overall, the proposed model exhibits excellent performance in terms of error rate and availability, making it well-suited for high-reliability applications. However, there remains considerable room for improvement in MTBF and MTTR. Future work should aim to enhance the system's fault tolerance and recovery efficiency to further bolster its robustness in real-world deployments.

5.3 Ablation study

To evaluate the contribution of key components within the proposed delay prediction optimization model and to further analyze the effectiveness of its internal mechanisms, an ablation study was conducted. By progressively removing core modules, several simplified variants of the model were constructed and evaluated under the same dataset and assessment framework. This approach allowed for a quantitative comparison of each component's impact on model accuracy, timeliness, and system control performance. Using the full model as the baseline, the following three ablated variants were implemented:

1) w/o Quantization: Disables weight quantization, using 32-bit floating-point parameters for computation.
2) w/o Online Update: Disables the online fine-tuning mechanism, maintaining a static model.

3) w/o System Features: Uses only historical delay sequences as input, removing all system context features.

The experimental results are summarized in Table 3:

Table 3: Ablation study results

Model Variant	MAE	MSE	Avg. Prediction Latency	Task Completion Rate (%)
Full Model	0.233	0.105	0.156	98.64
w/o Quantization	0.251	0.120	0.171	97.15
w/o Online Update	0.263	0.132	0.162	96.38
w/o System Features	0.285	0.149	0.165	94.72

As shown in Table 3, the full model performs best across all four evaluation metrics. Removing the quantization mechanism results in a slight decrease in accuracy but leads to a noticeable increase in inference time, highlighting the efficiency benefits of quantization for edge computing deployment. Disabling the online update mechanism leads to reduced model adaptability. The Task Completion Rate (TCR) drops by more than two percentage points, from 98.64% to 96.38%. This highlights the importance of feedback-based error correction in dynamic environments. The most significant performance degradation occurs when system context features are removed. In this variant, MAE increases to 0.285 and TCR falls to 94.72%, indicating that scheduling context is a key input for improving prediction accuracy and ensuring successful task execution.

6 Discussion

The FPGA-accelerated LSTM-based latency prediction model proposed in this study demonstrates strong performance across multiple dimensions, particularly excelling in prediction accuracy, task completion rate, and inference latency when compared to baseline models. However, a closer examination of the experimental results reveals several performance trade-offs and limitations that warrant further discussion and provide guidance for future optimization.

From a structural perspective, although the model achieves high prediction accuracy and low inference latency across various datasets, it performs slightly less robustly than some deep ensemble-based models in terms of robustness metrics. This may be attributed to the LSTM's limited adaptability to abnormal fluctuations and sudden load disturbances. Traditional control optimization models, which incorporate hard constraint mechanisms, tend to handle extreme variations more effectively. In contrast, while neural networks are well-suited for learning regular patterns, they may exhibit instability when handling boundary conditions or outlier predictions. Performance discrepancies across datasets also highlight the model's limitations. For instance, the model performs consistently better on ETT-small-m1 than on ETT-small-m3, likely due to the former's stronger periodicity and more controlled noise characteristics, which are advantageous for temporal modeling. In contrast, ETT-small-m3 may involve higher unpredictability in data sources, task switching frequency, or external disturbances, increasing the difficulty of model training. Future work could incorporate data augmentation techniques, attention mechanisms, or more sophisticated feature adaptation strategies to enhance the model's generalization across varied data distributions.

On the system implementation side, the FPGA acceleration scheme offers notable advantages over traditional software-based approaches. Customized data paths and parallel hardware structures significantly reduce inference latency, while resource control and low power consumption make the solution well-suited for deployment at industrial DCS edge nodes. Compared with processor-based solutions, the FPGA platform delivers superior task throughput and latency control within a given time frame, offering robust real-time guarantees—particularly important in high-frequency scheduling scenarios. However, FPGA deployment also introduces challenges in programmability and debugging complexity. Future research should explore the integration of automated deployment workflows with interpretability analysis tools. Regarding the quantization strategy, while 8-bit fixed-point quantization effectively reduces hardware resource consumption, its impact on model performance and RU still requires more systematic evaluation. Preliminary tests suggest that aggressive quantization may slightly degrade prediction accuracy, but it has a positive effect on throughput and on-chip resource availability. Balancing high-level compression with minimal loss in prediction precision remains a key direction for improving deployment efficiency in future work.

Furthermore, the experiments reveal inherent trade-offs between different performance dimensions. For instance, while the integration of an online update mechanism improves prediction accuracy and task completion rate, it also introduces additional computational overhead, which can negatively impact real-time performance. Conversely, disabling online learning accelerates inference but compromises the system's adaptability. Similarly, the task failure rate, a key indicator of system reliability, increases when the model lacks sufficient fault tolerance under high-load or complex scenarios. In real-world DCSs, such shortcomings may lead to scheduling delays or control disruptions, highlighting the importance of closely monitoring this metric.

Overall, the proposed method outperforms existing state-of-the-art (SOTA) models across most evaluation dimensions, with particularly notable advantages in inference latency and system throughput. However, there remains room for improvement in terms of robustness to anomalous predictions, cross-scenario adaptability, and the system's capacity for dynamic reconfiguration. Future work may explore the incorporation of attention mechanisms, heterogeneous input encoding strategies, and adaptive quantization techniques to better balance performance and deployability.

7 Conclusion

This study is the first to deeply integrate FPGA hardware acceleration with neural network–based latency prediction, resulting in a highly efficient predictive model. By offloading neural network inference to FPGA hardware, the model significantly reduces computational overhead during the prediction process, thereby enhancing the real-time performance of DCSs. This approach addresses the low efficiency typically seen in traditional neural network models when deployed on embedded platforms, offering a novel solution for real-time control tasks. The model's performance was rigorously evaluated across multiple dimensions—accuracy, robustness, timeliness, and resource efficiency. Experimental results demonstrate that the proposed model consistently outperforms baseline approaches, particularly in system throughput, average prediction latency, and task completion rate. These findings confirm the model's effectiveness and provide valuable performance benchmarks for future research. Despite these advances, certain limitations remain. The model performs well on the ETT-small dataset but has not been extensively validated in other scenarios, especially those involving unstructured or multimodal data. Future research could enhance the model's flexibility by incorporating multimodal fusion techniques, enabling it to integrate latency predictions with data from videos, sensors, and images for broader applicability in real-world settings. Additionally, the model shows some sensitivity under extreme conditions such as high load or the presence of outliers, where minor instability in key metrics was observed. To address these challenges, future work could explore adaptive thresholding, dynamic task allocation, and fault-tolerant mechanisms. Such enhancements would further improve the system's reliability and efficiency, especially in complex or unpredictable environments.

References

[1] Fei M, Zhang Z, Zhao W, et al. Optimal power distribution control in modular power architecture using hydraulic free piston engines. *Applied Energy*, 2024, 358(7): 122540. https://doi.org/10.1016/j.apenergy.2024.122540

[2] Kumar K, Pande S V, Kumar T C A, et al. Intelligent controller design and fault prediction using machine learning model. *International Transactions on Electrical Energy Systems*, 2023, 2023(1): 1056387. https://doi.org/10.1155/2023/1056387

[3] Liu G P. Tracking control of multi-agent systems using a networked predictive PID tracking scheme. *IEEE/CAA Journal of Automatica Sinica*, 2023, 10(1): 216–225. https://doi.org/10.1109/JAS.2023.120430

[4] Louati H, Niazi A U K, Dalam M E E, et al. Optimizing control efficiency in discrete-time multi-agent systems via event-triggered containment techniques combining disturbance handling and input delay management. *Heliyon*, 2024, 10(14): e26767. https://doi.org/10.1016/j.heliyon.2024.e26767

[5] Bououden S, Allouani F, Abboudi A, et al. Observer-based robust fault predictive control for wind turbine time-delay systems with sensor and actuator faults. *Energies*, 2023, 16(2): 858. https://doi.org/10.3390/en16020858

[6] Gams M, Kolenik T. Relations between electronics, artificial intelligence and information society through information society rules. *Electronics*, 2021, 10(4): 514. https://doi.org/10.3390/electronics10040514

[7] Ilyushin Y V, Asadulagi M A M. Development of a distributed control system for the hydrodynamic processes of aquifers, taking into account stochastic disturbing factors. *Water*, 2023, 15(4): 770. https://doi.org/10.3390/w15040770

[8] Nie Q, Tang D, Liu C, et al. A multi-agent and cloud-edge orchestration framework of digital twin for distributed production control. *Robotics and Computer-Integrated Manufacturing*, 2023, 82(4): 102543. https://doi.org/10.1016/j.rcim.2023.102543

[9] Dai X, Lederer A, Yang Z, et al. Can learning deteriorate control? Analyzing computational delays in Gaussian process-based event-triggered online learning[C]// *Learning for Dynamics and Control Conference*, PMLR, 2023: 445–457.

[10] Caballero-Águila R, Linares-Pérez J. Distributed fusion filtering for uncertain systems with coupled noises, random delays and packet loss prediction compensation. *International Journal of Systems Science*, 2023, 54(2): 371–390. https://doi.org/10.1080/00207721.2022.2126020

[11] Tiong K Y, Ma Z, Palmqvist C W. A review of data-driven approaches to predict train delays. *Transportation Research Part C: Emerging Technologies*, 2023, 148(21): 104027. https://doi.org/10.1016/j.trc.2023.104027

[12] Sridevi K, Saifulla M A. LBABC: Distributed controller load balancing using artificial bee colony optimization in an SDN. *Peer-to-Peer Networking and Applications*, 2023, 16(2): 947–957. https://doi.org/10.1007/s12083-022-01387-1

[13] Herrera M, Benítez D, Pérez-Pérez N, et al. Hybrid controller based on numerical methods for chemical processes with a long time delay. *ACS Omega*, 2023, 8(28): 25236–25253. https://doi.org/10.1021/acsomega.3c02747

[14] Olabi A G, Abdelghafar A A, Maghrabie H M, et al. Application of artificial intelligence for prediction, optimization, and control of thermal energy storage systems. *Thermal Science and Engineering Progress*, 2023, 39(21): 101730. https://doi.org/10.1016/j.tsep.2023.101730

[15] Jiang J, Han C, Zhao W X, et al. Pdformer: Propagation delay-aware dynamic long-range transformer for traffic flow prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, 37(4): 4365–4373.

[16] Mbungu N T, Ismail A A, AlShabi M, et al. Control and estimation techniques applied to smart microgrids: A review. *Renewable and Sustainable Energy Reviews*, 2023, 179(32): 113251. https://doi.org/10.1016/j.rser.2023.113251

[17] Balali Y, Chong A, Busch A, et al. Energy modelling and control of building heating and cooling systems with data-driven and hybrid models—A review. *Renewable and Sustainable Energy Reviews*, 2023, 183(3): 113496. https://doi.org/10.1016/j.rser.2023.113496

[18] Wang Y A, Shen B, Zou L, et al. A survey on recent advances in distributed filtering over sensor networks subject to communication constraints. *International Journal of Network Dynamics and Intelligence*, 2023, 6(2): 100007. https://doi.org/10.1016/j.indani.2023.100007

[19] Modu B, Abdullah M P, Sanusi M A, et al. DC-based microgrid: Topologies, control schemes, and implementations. *Alexandria Engineering Journal*, 2023, 70(11): 61–92. https://doi.org/10.1016/j.aej.2023.03.015

[20] Asha A, Arunachalam R, Poonguzhali I, et al. Optimized RNN-based performance prediction of IoT and WSN-oriented smart city application using improved honey badger algorithm. *Measurement*, 2023, 210(13): 112505. https://doi.org/10.1016/j.measurement.2023.112505

[21] Duan G. Fully actuated system approaches for continuous-time delay systems: Part 1. Systems with state delays only. *Science China Information Sciences*, 2023, 66(1): 112201. https://doi.org/10.1007/s11432-021-3411-y

[22] Ghiasi M, Niknam T, Wang Z, et al. A comprehensive review of cyber-attacks and defense mechanisms for improving security in smart grid energy systems: Past, present and future. *Electric Power Systems Research*, 2023, 215(11): 108975. https://doi.org/10.1016/j.epsr.2022.108975

[23] Ashok Babu P, Mazher Iqbal J L, Siva Priyanka S, et al. Power control and optimization for power loss reduction using deep learning in microgrid systems. *Electric Power Components and Systems*, 2024, 52(2): 219–232. https://doi.org/10.1080/15325008.2023.2239573

[24] Simonetti F, D'Innocenzo A, Cecati C. Neural network model-predictive control for CHB converters with FPGA implementation. *IEEE Transactions on Industrial Informatics*, 2023, 19(9): 9691–9702. https://doi.org/10.1109/TII.2022.3154581

[25] Abbasi M, Abbasi E, Li L, et al. Review on the microgrid concept, structures, components, communication systems, and control methods. *Energies*, 2023, 16(1): 484. https://doi.org/10.3390/en16010484

[26] Hu J, Li J, Liu G P, et al. Optimized distributed filtering for time-varying saturated stochastic systems with energy harvesting sensors over sensor networks. *IEEE Transactions on Signal and Information Processing over Networks*, 2023, 9(3): 412–426. https://doi.org/10.1109/TSIPN.2023.3268275

[27] Rajagopal A, Chitraganti S. State estimation and control for networked control systems in the presence of correlated packet drops. *International Journal of Systems Science*, 2023, 54(11): 2352–2365. https://doi.org/10.1080/00207721.2023.2177829

Improving Load Balancing Efficiency in Cloud Data Centers Through Hybrid Grey Wolf with Cat Swarm Optimization

Arif Ullah[1], Siti Fatimah Abdul Razak[2*], Amine Mrhari[3], Sumendra Yogarayan [4], Syeda Hina Mazhar Kazmi[5]
[1]Centre for Intelligent Cloud Computing, Multimedia University, Melaka, 75450, Malaysia
[2,4]Faculty of Information Science and Technology, Multimedia University, Melaka, 75450, Malays
[3]Computer Science Laboratory Faculty of Sciences, Ibn Tofail University Kenitra, Morocco
[5]Faculty of Information Science, Arid Agricultural University, Pakistan
E-mail: Arifullahms88@gmail.com & fatimah.razak@mmu.edu.my.
[*]Corresponding author

Keywords: virtual machine, makespan, throughput, degree of imbalance, turnaround time

Received: April 10, 2025

An effective resource management strategy that anticipates server resource utilization and appropriately distributes the load is recommended in order to address these problems and enhance data center performance. By reducing the number of servers in use, facilitating virtual machine migrations, and optimizing resource utilization, it helps save power. To reduce the likelihood of service level agreement (SLA) violations and performance degradation caused by either oveloded or under loaded servers and virtual machines. Resources for software applications can now be dynamically altered as needed thanks to the growth of cloud computing. Since better resource consumption can lead to increased scalability as well as significant cost and energy savings, effective resource management is crucial in cloud computing. The flexibility of cloud resources allows clients to dynamically increase and decrease their resource demands over time. However, predefined virtual machine sizes and variable resource requirements result in underutilization of resources, load imbalances, and high power consumption. The goal of this research is to develop a hybrid technique by combining Grey Wolf with algorithms. The hybridization processes take place in the Grey Wolf portion, when the Cat Swarm initialization process takes the place of the startup phase. The virtual machine (VM) section's data selection is enhanced by this substitution. The Grey Wolf and Cat Swarm algorithms are two examples of optimization algorithms. The evaluation criteria that are used are makespan, throughput, degree of imbalance, and turnaround time with degree of imbalance. The recommended approach outperforms alternative algorithms in each of these metrics. The proposed hybrid strategy resulted in 0.3% increase of overall performance. Potential directions for future research include testing the proposed approach in larger and more complex data distribution in cloud data centers.

Povzetek: Predstavljen je izvirni hibridni algoritem HGWCA, ki združuje dva algoritma: Grey Wolf Optimizer in Cat Swarm Optimization za učinkovitejše razporejanje nalog in zmanjšanje porabe časa v oblačnih podatkovnih centrih.

1 Introduction

The term "cloud computing" refers to the provision of pay-as-you-go on-demand services via the internet. Instead of using a conventional computer system or any other local devices, it allows you to handle files online. The cloud is a network of interconnected virtual machines with distributed, parallel systems that may deliver and offer computing resources on demand. "Cloud computing" refers to anything that involves offering hosted services via the internet. Private, public, hybrid, and community clouds are the four types of cloud deployment models. Numerous cloud service providers are accessible, and each one offers a unique collection of cloud services. Platform as a Service (PaaS), Infrastructure as a Service (IaaS), Software as a Service (SaaS), and Storage as a Service (STaaS), Cloud computing models that provide a range of services to customers include Security as a Service (SECaaS), Data as a Service (DaaS), Test Environment as a Service (TEaaS), and Backend as a Service (BaaS) [1]. Software as a Service (SaaS) offers software services online, with the service provider managing software maintenance and updates. IaaS enables virtualized access to actual resources, whereas PaaS offers tools for managing and developing applications. Instead of buying their own devices, businesses can rent storage capacity thanks to STaaS. Security services like intrusion detection and authentication are included in SECaaS. Regardless of one's location, DaaS makes it possible to access data whenever there is a need for it. Through the internet, TEaaS enables users to access software and pertinent data. Developers can concentrate on the frontend and application by using BaaS's pre-built backend services for mobile applications reasoning [2]. Cloud computing has a number of benefits. It gives users access to inexpensive systems, removing the need for powerful computers and enabling the use of less expensive gadgets. It makes data and apps accessible from any location at any

time, it providing dependability and ease. Because they may use cloud computing instead of spending money on pricey servers, it also lowers the cost of IT infrastructure for businesses. There are less maintenance problems and expenses, which means that less hardware and software maintenance is needed. Users can always get the most recent version of software without extra expense or work because to cloud computing's smooth software upgrades. By utilizing the combined capabilities of linked PCs and servers, it provides more processing power [3]-[4]. Cloud storage surpasses the constraints of local storage by offering nearly infinite capacity. Usage of these resources by making certain that workloads are distributed fairly across the servers or virtual machines that are available. Better performance and resource efficiency in the cloud are ensured by load balancing, which helps avoid overloading certain servers by dynamically allocating and reallocating computing resources based on demand. In a cloud context, load balancing is the essential technology to guarantee a fair task distribution and effective resource use. One useful strategy for cloud data centers to save energy is to consolidate virtual machines (VMs). By combining virtual machines (VMs) into fewer active physical machines (PMs), the VM consolidation technique enables the PMs that do not have any VMs to go into a state of dormancy. Because a PM uses a lot of energy when it is sleeping, Utilizing VM consolidation techniques can assist cloud data centers consume less energy, as it is lower than that of a PM in its active mode [5]. To reduce waiting time, the virtual machine (VM) should react quickly when a tasks is sent to the cloud to be processed. However, tasks should be distributed among all virtual machines (VMs) in parallel to ensure system balance and efficient functioning making use of the available resources. This necessitates task planning that is assigned and distributed among the available resources. When several tasks are assigned to a single virtual machine (VM), the assigned activities will execute simultaneously across numerous VMs to complete the tasks. As a result, the task requirements ensure that not every tasks is loaded onto a single computer. The system is unbalanced, or the VM will restrict access to other VMs, rendering them unreachable [6]. Other factors, such as makespan, cost, and resource use, must be considered while scheduling in order to avoid this. The main goal of allocating among a system's load-balancing responsibilities is to maximize workload distribution among available resources and minimize system-processing timing. Several scholars have proposed methods for load balancing in both uniform and heterogeneous environments. In order to do this, the suggested algorithm in this study split the tasks evenly between VM and PM [7]. The structure of the paper is as follows: Section 2 presents the latest advancements in tasks distribution for hybrid systems for cloud systems. Section 3 describes the suggested model, and the algorithm created to address the issues and illustrate the design of our solutions and allocate resources. Section 4 describes the procedures, tests, and results and the conclusions are presented in Section 5.

2 Related work

In this study, we examine the relevant literature on energy management, load balancing, and appropriate cloud data center use. In order to evaluate Quality of Service (QoS) metrics, a multi-objective task-scheduling algorithm that combines Cat Swarm Optimization (CSO) with machine learning classifiers such as Support Vector Machine (SVM) has been proposed. However, the algorithm falls short in terms of classification and consideration of other important parameters. Concepts like workload prediction and auto-scaling require further investigation [8]. A dynamic task scheduling system for tasks and cloudlets is suggested by the author [9]. The approach seeks to increase acceptance rate, scalability, and minimize makespan. However, it is expensive and complicated. Future studies will concentrate on load balancing strategies that respect privacy. As stated in [10], discuss the scheduling of cloud computing applications, with an emphasis on response speed and makespan. Enhancing these metrics and investigating the effectiveness of the MOTSGWO algorithm in an open stack setting are their goals. The Sand Cat Swarm Optimization (SCSO) algorithm for global optimization problems is introduced in [11]. Although the technique can be expensive, it works effectively in a variety of engineering design challenges. Additionally, it may find use in bioinformatics, machine learning, smart farming, logistics, wireless sensor networks, and the Internet of Things. For feature selection in classification tasks, a binary multi-objective grey wolf optimizer is proposed in [12]. With a lower computational cost, the algorithm performs better than current methods in terms of features reduction and classification error rate. It works well with small datasets introduces a clustering technique based on RCSO that makes use of SVM and MLP networks. The RCSO algorithm performs better than current algorithms and may be applied in future tissue cell quantification and legal proceedings. An approach for the Urban Transit Routing Problem (UTRP) based on Cat Swarm Optimization (CSO) is presented in [13]. It is economical and successfully resolves the UTRP. However, the values of the parameters affect the algorithm's capacity to optimize. A Cat Swarm Optimization (CSO) algorithm using a subgroup information interaction approach is proposed by the author in [14]. It exhibits a faster rate of convergence and the capacity to search globally. It will be compared to other swarm intelligence algorithms in future studies. A Sand Cat Swarm Optimization (SCSO) algorithm based on elite collaboration and [15] present stochastic variation. Although it increases time complexity, it effectively addresses engineering and optimization difficulties. For more applications, cooperation with other researchers is desired. As suggested by [16], a MATLAB task scheduling system that reduces makespan and energy usage. Energy and execution costs, however, are not taken into account. Evaluation of the suggested algorithm in a small-scale cluster is the goal of future research. The enhanced Multi-Objective Grey Wolf Optimizer (IMOGWO) algorithm for edge computing work scheduling is presented in paper

[17]. In terms of convergence, diversity, and coverage, IMOGWO performs better than current algorithms. The algorithm can be used for multi-objective test issues; however, it is expensive suggests using the Hybrid Particle Swarm Grey Wolf (HPSGW) algorithm to optimize automated CNN in [18]. The technique lowers computational costs while increasing performance accuracy on benchmark datasets. Future studies will concentrate on enhancing the optimization procedure by adding more hyperactive parameters. A hybrid form of MCA and GWO for scheduling the power of smart home appliances is presented by [19]. The Power scheduling issues are successfully addressed by the GWO-MCA hybrid. The goal of future study is to enhance the selection process and method's performance. For multi-objective optimization problems, [20] suggests a method based on the Grey Wolf Optimizer (GWO) with the memeplex structure of the Shuffled Frog leaping method (SFLA). The method can be used to solve a variety of problems and produces competitive results. In order to optimize spectrum sensing in cognitive radio networks, [21] presents the Multi-Objective Modified Grey Wolf Optimization (MOMGWO) algorithm. In spectrum sensing, MOMGWO has excellent coverage, convergence, and performance. The Multi-Objective Grey Wolf Optimizer (MOGWO) method is suggested by [22] for the IEEE-30 bus test system. For multi-objective problems, MOGWO shows good Pareto-optimal front setup and quick convergence. [23] Introduce the Fractional Grey Wolf Optimizer (FGMTS). For cloud

Table 1: Related work summary

Ref	Methods	Parameter	Results	Limitations	Future work
[25]	Dynamic task scheduling	Makespan	Well as improving scalability	Algorithms are expensive and complicated	Effectively safeguard sensitive task
[26]	Multi objective grey wolf optimizer (MOGWO)	Convergence	Fastest convergence and the best Pareto-optimal front setting. Reduce fuel cost	Laborious and costly	It is possible to obtain a Pareto-optimal front for multi-objective
[27]	FGMTS: Fractional grey wolf optimizer	Multi-Objective task scheduling	Performance and increasing profitability.	Result in faulty resource allocation for subtask	Could be used for other datasets
[28]	Task scheduling mechanism	Makespan and energy consumption	It greatly minimizes energy consumption and makespan	Energy and execution cost are not considered	Assess the proposed algorithm while taking into account a small-scale cluster.
[29]	IMOGWO algorithm	Task Scheduling	Convergence, diversity and coverage.	Costly	Applied to multi-objective test problem
[30]	An algorithm based on grey wolf optimizer (GWO) with memeplex structure of the shuffled frog leaping algorithm (SFLA)	Multi objective optimization	Multi-objective problems and achieve competitive or superior results when compared to other comparison algorithms.	Complicated	It can also be tested on multi-objective types of real-world problems such as image processing, vehicle routing and clustering.
[31]	Cat Swarm Optimization (CSO)-based algorithm	Optimization, scheduling workflow	CSO based algorithms are an excellent choice for effectively solving the UTRP while also being cost-effective.	The determination of parameter values, which may be lacking in effectiveness.	The algorithm aims to optimize the cost of the service provider while also being applicable
[32]	A binary multi-objective grey wolf optimizer for feature selection in classification	Features number, classification accuracy	Classification error rate though benefiting from a lower computational cost.	Effective for small size datasets	Minimizing the number of features and minimizing the error rate simultaneously
[33]	Multi objective grey wolf optimizer (MOGWO)	Throughput, makespan and resource utilization in cloud computing.	The method outperformed all other methods tested	Complicated	Memory usage during peak loads, to improve overall efficiency.

Computing multi-objective tasks scheduling. In addition to achieving resource allocation, the FGMTS method enhances cloud profitability and performance. However, because of the search space, there can be restrictions. The multi-objective Grey Wolf Optimizer (MOGWO) for near-optimal tasks scheduling in cloud computing is presented in paper [24]. The MOGWO performs better than alternative approaches and can be improved with parallel programming techniques. The summary of related work is shown in Table 1.

The main goal of dynamic task scheduling systems is to decrease makespan while improving system scalability; however, their algorithms are frequently costly, intricate, and made to properly protect sensitive activities. Although it is still time-consuming and expensive to install, the Multi-Objective Grey Wolf Optimizer (MOGWO) is known for reaching the fastest convergence rates and providing the best Pareto-optimal front settings, with advantages including fuel cost savings. These capabilities are extended for multi-objective task scheduling by the

FGMTS (Fractional Grey Wolf Optimizer), which enhances system performance and profitability. However, it can occasionally lead to incorrect subtask resource allocation, which may be applicable to other datasets. However, they frequently ignore the combined effect of energy and execution costs, traditional task scheduling techniques focus on decreasing makespan and energy consumption, restricting their assessment to small-scale clusters. By striking a balance between convergence, diversity, and coverage, the IMOGWO method significantly improves task scheduling. However, because of its high computational costs, it is more appropriate for controlled multi-objective test problems. Another version combines the Grey Wolf Optimizer with the Shuffled Frog Leaping Algorithm's (SFLA) structure. It produces competitive results in multi-objective optimization, yet it is still sophisticated and appropriate for applications such as clustering, image processing, and vehicle routing. Targets for feature selection using a binary multi-objective GWO although it works best with small datasets, it can achieve low error rates and lower computing costs by decreasing the number of features and increasing classification accuracy. Furthermore, throughput, makespan, and resource utilization are improved by a multi-objective GWO version for cloud computing environments, which continuously outperforms alternative approaches but struggles with complexity and excessive memory use during periods of high load. Overall, these techniques tackle different facets of multi-objective scheduling and optimization problems in cloud and workflow contexts; each has particular advantages and disadvantages.

2.1 Problems statement

The Grey Wolf exploitation strengths are enhanced by the Cat Algorithm's exploration skills, resulting in a synergistic optimization framework. This hybrid approach seeks to allocate workloads across available resources in a dynamic manner while tackling several goals, including throughput, energy efficiency, and response speed. In a fast changing technological world, the incorporation of these bio-inspired algorithms into load balancing offers a fresh way to get around the drawbacks of current methods, improving the scalability and performance of cloud data centers. The Cat Algorithm's exploration abilities complement the Grey Wolf exploitation capabilities, creating a synergistic optimization framework. This hybrid strategy aims to dynamically distribute workloads across available resources while addressing a number of objectives, such as response speed, energy efficiency, and throughput. The integration of these bio-inspired algorithms into load balancing provides a novel approach to overcome the limitations of existing techniques, enhancing the scalability and performance of cloud data centers in a rapidly evolving technological environment.

3 Proposed algorithm

A task manager, scheduler, CPU, RAM, storage resources, virtualization layer, physical resources, and a data center are among the parts of the system shown in Figure 1.

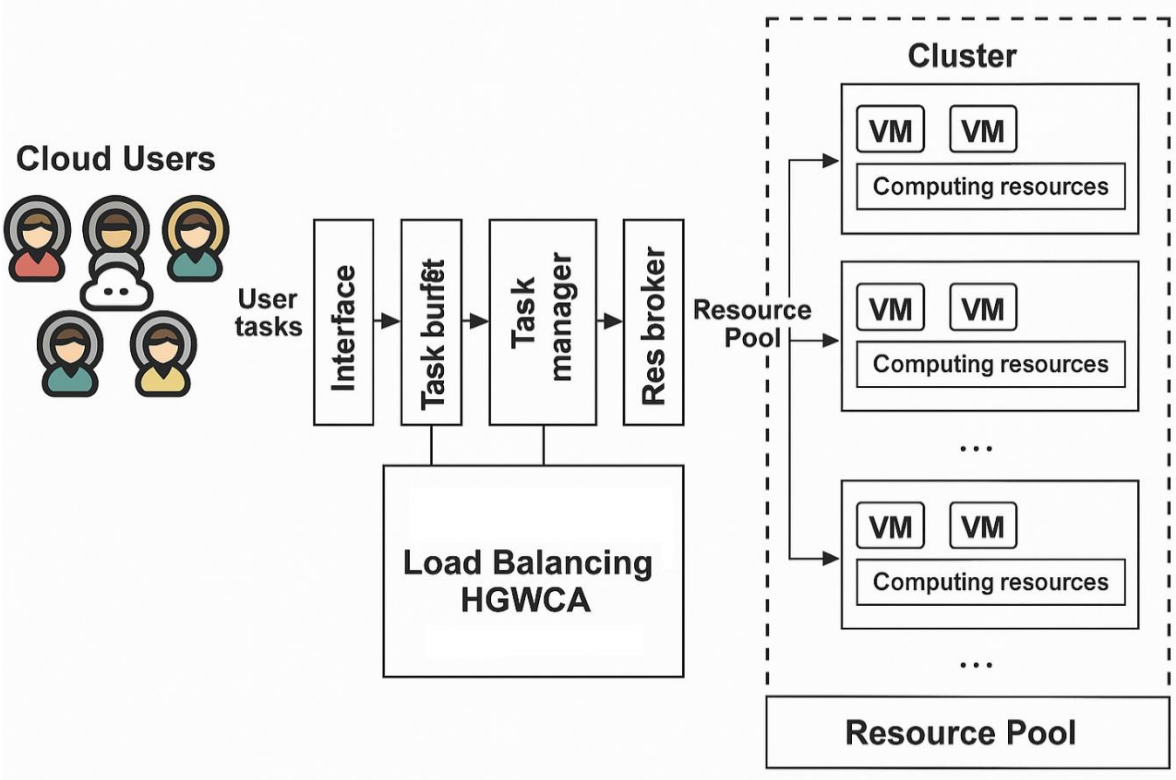


Figure 1: Proposed approach model

The grey wolf optimization algorithm (GWO) is a new type of metaheuristic algorithm. In the field of engineering, it is most commonly used to tackle optimization problems. It is inspired by the hunting and social behaviors of grey wolves. It imitates the natural

Leadership structure and hunting technique of grey wolves. Grey wolves do most of their living in packs. In a pack of grey wolves, the leadership hierarchy is indicated by the four categories of alpha, beta, delta, and omega. Each category denotes a different degree of authority and leadership within the pack. The dominant member of the pack, the alpha wolf, acts as a leader to guide the pack through the hunting process. Beta wolves [34] occupy the second level of this hierarchy. They are known as subordinates and assist. The alpha when creating policies during the hunting phase. After the alpha and beta wolves, the delta wolf represents the third level of decision-making authority. The last rung in this ladder is omega wolf. That is what scapegoat wolves are. Grey wolves' decision-making skills decline in the absence of the dominating alpha wolf. The gray wolf Optimizer is mostly inspired by this social skill [35]. Grey wolves hunt in three repeating stages utilizing a methodical approach. These phases include searching, encircling, and assaulting. This first phase is called tracking or searching. It is also known as the exploration stage. In a global search arena, the grey wolves look for prey. The mathematical the model describes how to determine the distance between the grey wolf and its prey [36]. It is computed using Equation 1.

$$Dist = abs(coeff \times L_P\ (t) - L_{GW}\,(t)) \quad (1)$$

Here $coeff$ provides the coefficient vector; $L_{GW}\,(t)$ shows where the grey wolf was in the specified time interval t; L_P shows the location of prey at a specified time interval t.

The value of $coeff$ is calculated using Equation 2.

$$Coeff = 2 \times Ran(0,1) \quad (2)$$

Due to the potential for random task generation and the consequent random assignment of tasks to the seeking and tracing modes, n has been randomly set as the number of dimensions. At each iteration, solutions are computed for cats, and their fitness is evaluated relative to other cats, until the optimal solution is found. While in the "seeking" mode, all cats will enter a resting state, however specific behaviors are required to find a cat. The new velocity in the tracing mode can be computed using the cats' observations and the known initial velocity [37]. You may see it below. Due to the potential for random task generation and the consequent random assignment of tasks to the seeking and tracing modes, n has been randomly set as the number of dimensions. At each iteration, solutions are computed for cats, and their fitness is evaluated relative to other cats, until the optimal solution is found. While in the "seeking" mode, all cats will enter a resting state, however specific behaviors are required to find a cat. The new velocity in the tracing mode can be computed using the cats' observations and the known initial velocity, you may see it below [38].

$$Vel_i^d(t+1) = p * Vel_i^d(t) + c * r * (x_d^{best} - x_i^d) \quad (3)$$

In Equation (3), c is a constant, r is a random value between 0 and 1, and $Vel_i^d(t)$ is the velocity of the ith at during the t-th iteration. The cat is on its top place globally atx_d^{best}. Cat location at ith iteration is represented byx_i^d.

By using the Equation (4), the location of the cat may be modified as follows.

$$x_i^d(t+1) = x_i^d + Vel_i^d(t+1) \quad (4)$$

Following fitness calculations, solutions must be updated with non-dominated cats. These calculations must be carried out until cats achieve objectives with an optimal value. The termination criterion is used to end the algorithm. For the algorithm to work well, a suitable termination condition needs to be selected. The number of iterations, the degree of improvement, and the running time are typical CSO termination criteria [39]. During the third phase, grey wolves adjust their locations based on the movements of their prey, as shown in equations (1) through (6). In the grey wolf, social structure, alpha, beta, and delta wolves hold the highest positions and are thought to be the finest hunting alternatives. After the prey has been surrounded at a specific distance, it is believed that the alpha, beta, and delta wolves have a better idea of its whereabouts. Consequently, the alpha wolf initiates the attack. The prey's position is subsequently updated using the alpha, beta, and delta wolves' positions, as mathematically demonstrated in equation (5). In contrast, the omega wolves follow the guidelines in equation [40] and move around the prey intermittently.

$$\begin{gathered} Dist_{alpha=}\ abs(coeff_1\ \times\ L_{alpha}\ \times (t) - L(t)) \\ Dist_{beta=}\ abs(coeff_2\ \times\ L_{beta}\ \times (t) - L(t)) \\ Dist_{delta=}\ abs(coeff_3\ \times\ L_{delta}\ \times (t) - L(t)) \\ L_1\ =\ L_{alpha}(t) - coeff_1\ \times\ Dist_{alpha} \\ L_2\ =\ L_{beta}(t) - coeff_2\ \times\ Dist_{beta} \\ L_3\ =\ L_{delta}(t) - coeff_3\ \times\ Dist_{delta} \\ L = \frac{(L_1 + L_2 + L_3)}{3} \end{gathered} \quad (5)$$

A set of random solutions are used as grey wolves by the grey wolf optimizer. Several objective functions are used to evaluate a set of randomly selected responses. The quality of each solution is represented by the values of a fitness function composed of many functions. Alpha, beta, and delta wolves stand for the highest three quality solutions. The grey wolf optimizer continuously adjusts the wolf population's location throughout each cycle. If a response improves during an iteration to the extent where it outperforms alpha, beta, and delta wolves, the corresponding solution is replaced with the new one. When a set of stopping conditions is met, the Grey Wolf Optimizer stops iterating the solutions [41]. While the scheduler distributes resources like CPU, memory, and

storage, the task manager organizes tasks. To manage virtual machine or networks, the system has a virtualization layer and virtualized resources. The infrastructure is kept in the data center, and the physical resources stand in for the underlying hardware. Additionally, Figure 1 mentions the HGWCA algorithm, which is suggested to be involved in resource manageme nt. Tasks are initially considered TK= {T1+T2+T3… Tn}. The cloud console must be used to submit these tasks to the task manager first. The task management module in this configuration needs to assess each task's relative priority [42]. Task, and then assess each virtual machine's relative significance based on how much power it uses in a certain amount of time. It is being tried because there are many different types of tasks being brought into the cloud platform, and it is crucial to map them to the appropriate virtual resources in the cloud. The task scheduler is in charge of carrying out this task. A resource manager module, which is connected to a task scheduler, keeps tra ck of resource requests, allocations, and availability on th e respective physical hosts in the datacenters [43]. Assum ed in this method are n virtual machines (VMs) with nam es such as these virtual machines must live in physical hosts with names like Vn={V1,V2,V3,……Vn }.

$H_i = \{H_1, H_2, H_3, \ldots \ldots H_i\}$

In addition, these hosts are housed in datacenters (called centers) with names like

$$d_j = \{d_1, d_2, d_3, \ldots \ldots d_j\}. \quad (6)$$

The architecture in question assigns tasks to virtual machines (VMs) by determining the Task Priority and VM Priority, respectively, while considering the datacenters' relative power costs [44]. To allocate each task to the best virtual machine (VM), a priority calculation must be performed because every task has different processing needs. The cloud's virtual machine (VM) priority computations must take regional variations in power prices into consideration because it can be deployed anywhere [45]. A technique that first identifies the most critical task and then allocates them to virtual machines that are most suited to completing those tasks with the least amount of energy is used in cloud data centers to reduce overall power costs and energy usage. Tasks, virtual machines, real hosts, and datacenters can all be used to define the problem. The scheduler's purpose is to distribute workloads to virtual machines in a way that minimizes costs and power consumption [46]. The given equation 7 present the hybrid section of the proposed algorithm.

$$Vi(0) = \{ X\,i(current) + \delta \times R, X\,i(current) + r \times (X\,best - X\,i(current) \quad (7)$$

It made a number of assumptions regarding datacentersd_j, physical hostsH_i, virtual machines V_n and tasks T_K in order to schedule tasks onto virtual machines. It must ascertain the relative significance of virtual machines and tasks for this configuration. These settings are examined and given a single priority when tasks are submitted to the task manager. The scheduler assigns tasks to the VMs, and in order to determine priorities, it is necessary to ascertain the VMs' present load. The VMs perform the tasks that the scheduler allocates, requiring the determination of their current load to calculate priorities. The equation below demonstrates the calculation of the load on all VMs [47].

$$L_v = \sum_{n=1}^{q} L_n \quad (8)$$

The current load on each virtual machine (VM) is represented by the variable L_n since all of the virtual machines are meant to be housed within physical hosts; the capacity of the hosts must be ascertained after the load on each VM has been computed.

$$L_h = \frac{L_v}{\sum_{k=1}^{m} H_k} \quad (9)$$

The physical host, which is composed of virtual machines, is under L_h load. Because the VMs must transfer to the next Virtual Machine on the same physical host either by starting a new request or to the subsequent VM that is already running if there are more tasks than they can handle, a load-balancing module is necessary in the cloud-computing paradigm. A load balancer can accomplish this, but in order to do so, a specific threshold value must be set to identify if the system is balanced or not. The system's threshold value is identified as follows [48].

$$TH_k = L_h * H_K \quad (10)$$

Prior to determining whether the system is overloaded, under loaded, or balanced, the threshold value is established. The load balance is calculated to arrive at this conclusion. In the event that a system gets overwhelmed.

$V > TH_k - \sum_{i=1}^{n} L_V$
If the system is under loaded, then
$V < TH_k - \sum_{i=1}^{n} L_V$
If the system is balanced,
$$V = TH_k - \sum_{i=1}^{n} L_V \quad (11)$$

To plan work onto appropriate VMs, it is necessary to see the processing capabilities of each VM. The calculation is as follows [49].

$$Proccapicity_V = pr^{no} * pr^{mips} \quad (12)$$

The above equation is used to calculate each VM's processing capacity, and the results are listed as follows.

$$TotProccapicity_V = \sum_{i=1}^{n} Procapacity_V \quad (13)$$

The requirements for allocating task to virtual machines have been determined and computed. The priorities for both the tasks and virtual machines have now been determined. The aforementioned formula has been used to calculate the work's size, and the relationship between a task's size and a virtual machine's processing

power is what defines its priority. The following is the formula [50].

$$T^{priority} = \frac{T_k^{size}}{Proccapicity_V} \quad (14)$$

Two stages of verification are used to schedule the tasks in this work. In order to accurately map tasks onto the appropriate virtual machines (VMs), the first step entails determining task priorities. To make the mapping process easier, VM priorities are then established according to power unit cost. The following algorithm based on power unit cost determines VM priority [51].

$$V^{priority} = \frac{elec\ unit\ cost^{high}}{elec\ unit\ cost_j^d} \quad (15)$$

The metrics this tasks address must be identified after the priorities have been calculated. Any cloud computing task-scheduling system must consider makespan because it is a crucial metric to manage in this paradigm. Additionally, lowering response time has a significant positive impact on cloud providers and users, which in turn has an indirect impact on other metrics like energy consumption and total power cost. Here is the formula [52].

$$m^k = availability^n + e^k \quad (16)$$

4 Experiments and results

An analysis of the various experimental outcomes obtained by applying the recommended methodology is presented in this section. A wide range of datasets and parameters has been used in numerous experiments and testing. A thorough explanation of the study's experimental setup provides crucial context for understanding the methodology. The general process employed in the tests is described through a number of precisely specified points. This section provides comprehensive documentation of the experimental details associated with the proposed model. The primary objective of this section is to outline the exact configuration that was utilized to conduct the experiment as well as the systematic procedures that were followed in order to execute the recommended strategy within the experimental inquiry. As a result, the experimental setup and specifics will be covered in detail in the parts that follow.

4.1 Dataset

The availability of appropriate datasets, which were taken from the AWS (DREAM) database for this study, is a crucial component for the scheme's effective implementation. The Proposed model performance evaluation included a range of virtual machines (VMs) and workloads (between 200 and 2000) spread across various data centers in the cloud-computing environment.

4.2 Simulation parameters

A modified CloudSim 3.0 platform was used to construct the HGWCA algorithm, and test datasets with file sizes between 200 and 400 kilobytes and compliance with the standard workload format (SWF) were used. The simulation parameters are shown in Table 2.

Table 2: Simulation parameter

Type	Parameter	Value	Type	Parameter	Value
Region	From 1 to 4	5	Task/Data	No. of task	100/1600
Data center	No. of data center	5		Length of task	100/200/400 byte
	No. of hosts	100/1000		No. of processor per requirement	250 KB
	Type of manager	Time and space		Type of manager	Time and space
	Bandwidth	1000	Memory	Total memory	204,800 MB
Virtual Machine	Total no. of VM	30/20/20		No. of processor	4 per VM
	No. of processor per VM	4		Total processor	120
	VM memory	512		Storage memory	100000 Mb
	Bandwidth in bit	1000		Viable memory	10,000
	VM image size	1000	Cloudlets	Total no. of task	100/2000

A variety of parameters and their corresponding values pertaining to various system features are included in Table 2. It contains details about tasks, data centers, regions, virtual machine specs, algorithms, and dataset properties. The number of data centers, task duration, manager type, bandwidth, virtual machine configurations, CPU specs, and dataset specifics are a few examples. A thorough

summary of these factors is given in Table 2, which also offers information on the setup and features of the system.

4.3 Performance metrics

With an emphasis on important metrics of load balancing effectiveness such makespan, throughput, turnaround time, and degree of imbalance, the proposed model performance was evaluated against that of other well-known algorithms, including ABC, MBat, HHO-ACO, and QMPSO. The following assessment measures are employed to examine the performance. These measures are described are as: The time required to process every tasks in the designated order is known as the make span. The study's primary goal is to reduce the makespan time. Makespan can be computed using the following formula:

$$Makespan = \left(\frac{Given\ time\ of\ network}{Complete\ time\ of\ network}\right) * 100 \quad (17)$$

Throughput is the quantity of data that can be sent and received in a given period. Throughput is the average ratio at which messages are successfully delivered to their intended location. The provided formula [53] can be used to compute it.

$$Throughput = \frac{No.of\ Tasks}{makespan} \quad (18)$$

The rate at which tasks are finished in a specified amount of time is referred to as throughput. It shows how much work is completed or processed in a certain amount of time. "Number of Tasks," the numerator of the equation, is the total number of tasks finished in the allotted time. It shows how much work or tasks have been completed successfully. The degree of imbalance quantifies the haphazard distribution of cloud workloads among virtual machines based on their respective capacities. It is usually computed using virtual machine task execution times. The provided formula [54] can be used to calculate it.

$$Imbalance_Degree = \frac{\text{Max}_CTime_i - Min_CTime_i}{Avg_CTime_i} \quad (19)$$

Here, Max $_CTime_i$ denotes the maximum completion time of tasks on all virtual machines. Min_CTime_i Denotes the minimum completion time of tasks on all virtual machines.Avg_CTime_i Denotes the minimum completion time of tasks on all virtual machines. Imbalance Degree is a measure of how well load balancing is working in the cloud. Lower values show that Workload on the cloud is properly balanced. While higher values demonstrate inefficient, load balancing [55].

5 Results and discussions

With an emphasis on important metrics of load balancing effectiveness such makespan, throughput, turnaround time, and degree of imbalance, the proposed model performance was evaluated against that of other well-known algorithms, including PSO, ABC, PSO-CALBA, M-Bat, HHO-ACO, Proposed model. In comparison to other algorithms, the suggested Proposed modeled menstruated an impressive 0.98% decrease in makespan and a higher accuracy rate of .18%, according to the comparative analysis. These results validate the feasibility and efficacy of the proposed model in resolving load balancing issues in cloud computing settings and the result details are given below table 3 present the throughput.

Table 3: Complete no. of task of different algorithm based on No. of Task

Given No of Task	PSO	ABC	PSO-CALBA	M-Bat	HHO-ACO	Proposed model
200	160	165	170	175	186	193
400	355	365	372	378	385	390
600	560	570	575	580	588	592
800	750	770	775	780	786	791

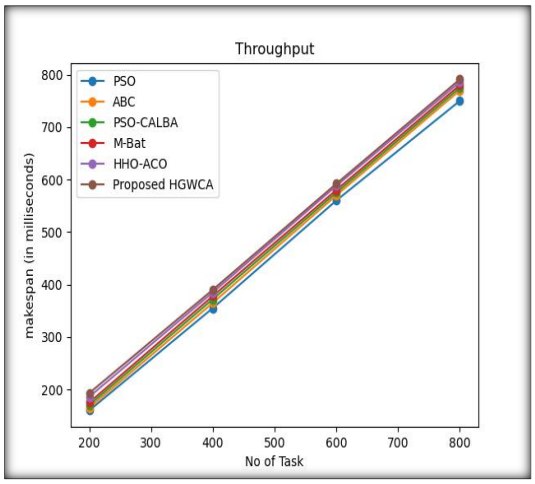


Figure 2: Throughput

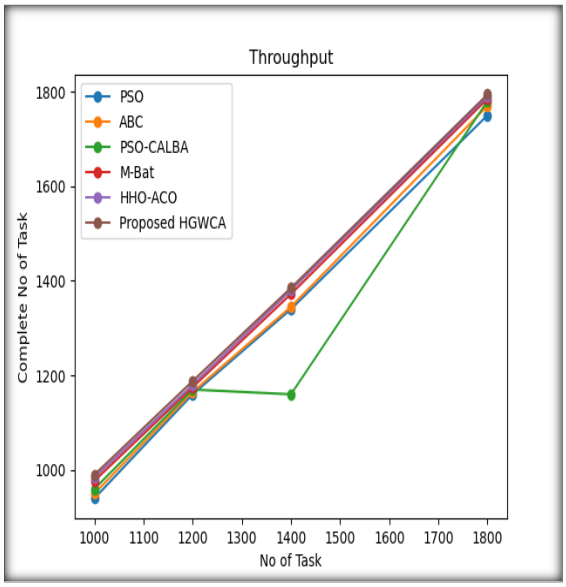


Figure 3: Throughput

Table 3 and Figure 2 show that the "Proposed model" algorithm performs the best based on the provided table. In a variety of task counts, it regularly receives the highest marks. With 800 tasks, it has a throughput of 791. In contrast, the "PSO" algorithm performs the poorest in the table provided. In a variety of task counts, it regularly receives the lowest scores. With 800 tasks, it has a throughput of 750.Thus, of the stated algorithms, the Proposed model method performs the best based on the data presented, while the PSO algorithm performs the poorest. The "Proposed model" method is clearly the best-performing algorithm, according to the statistics shown in Figure 3. With a throughput of 1794 when the number of tasks is 1800, it continuously obtains the highest marks across a range of task counts. On the other hand, in the provided table, the "PSO" algorithm performs the worst. With a throughput of 990 when there are 1800 tasks, it consistently receives the lowest scores throughout a range of task counts. Therefore, it can be inferred from the information given that, among the algorithms indicated, the proposed model method performs the best, while the PSO algorithm performs the worst. Figure 4 and Table 4 present the Makespan of the proposed algorithm.

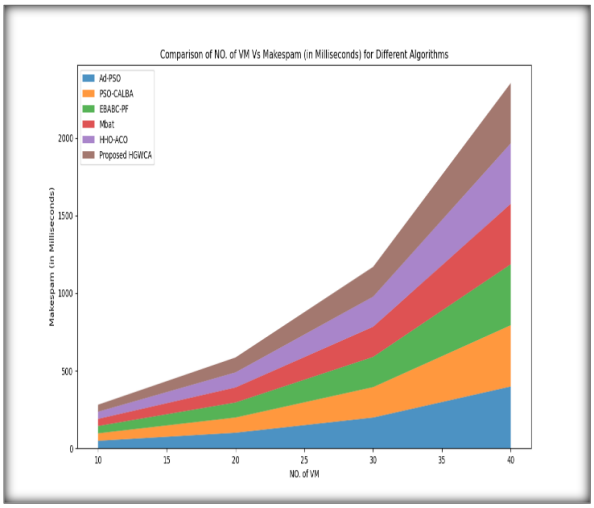


Figure 4: Makespan

Table 4: Makespan of different algorithm based on no. of virtual machine

No. of VM PSO	ABC	PSO-CALBA	M-Bat	HHO-ACO	Proposed model
10	48	48	46	46	45
20	100	98	97	97	96
30	198	196	194	194	192
40	398	395	390	390	388

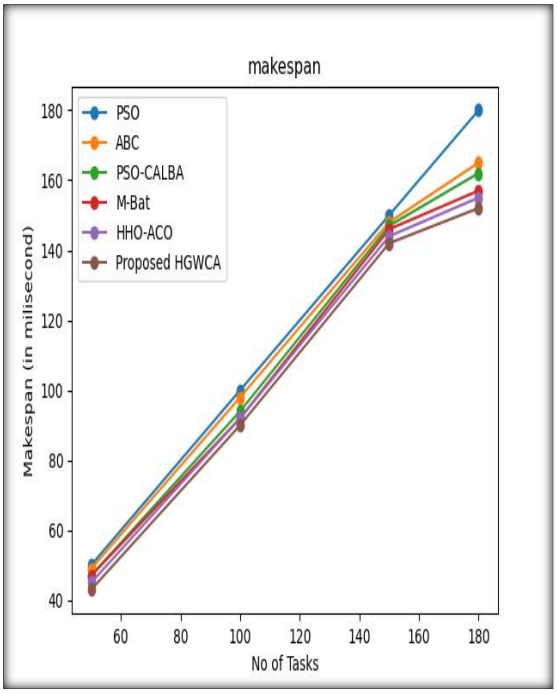


Figure 5: Comparison no. of VM Vs no task for different algorithms

The "Proposed model" technique is the best one based on the data presented in Table 5 and Figure 4, as it consistently produces the lowest makespan values for varying numbers of virtual machines. With consistently greater makespan values than other algorithms, the "PSO" algorithm is the worst. This suggests that when it comes to reducing execution time, the "Proposed model" algorithm outperforms the "PSO" approach.

Data on the number of completed tasks and the corresponding makespan (in milliseconds) for various algorithms are shown in Table 5 and Figure 5 and 6. The "Proposed model" consistently obtains the lowest makespan values for all sets of full tasks, based on the data provided. As a result, it can be regarded as the optimal method for reducing execution time.

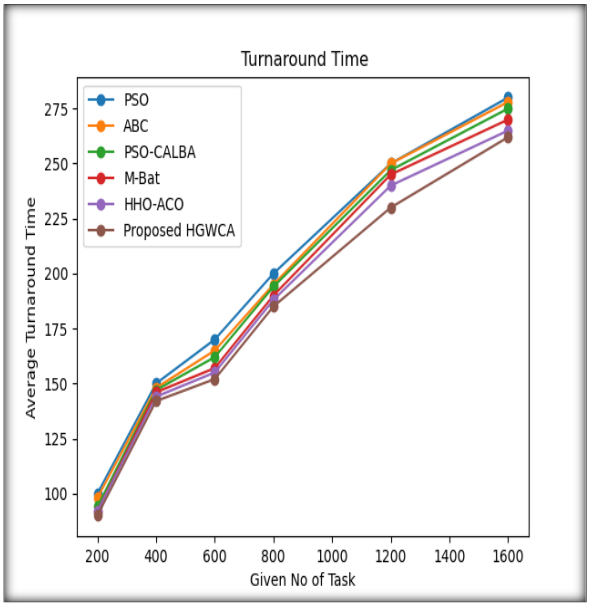


Figure 6: Makespan based on no. of task (50-180)

Table 5: Makespan based on no. of task

Algorithm Name	Complete No of Tasks	Makespan (In Milli second)	Complete No of Tasks	Makespan (In Milli second)	Complete No of Tasks	Makespan (In Milli second)	Complete No of Tasks	Makespan (In Milli second)
PSO	200	200	220	220	250	250	280	280
ABC	200	195	220	218	250	250	280	278
PSO-CALBA	200	194	220	215	250	247	280	275
M-Bat	200	190	220	210	250	245	280	270
HHO-ACO	200	188	220	207	250	240	280	265
HGWCA	200	185	220	202	250	230	280	262

Table 5 displays, for a range of parameters, the average results produced by numerous writers, including the suggested method. For varying task amounts, the authors' algorithms—PSO, ABC, PSO-CALBA, M-Bat, HHO, and the suggested approach—are assessed.

The following are the average results for each algorithm across various job quantities: M-Bat (169.63), HHO (167), PSO (177.50), ABC (175.13), PSO-CALBA (172.63), and the suggested technique (163.25). With lower numbers signifying better results, these average values show how well each method performs in relation to the given parameter.

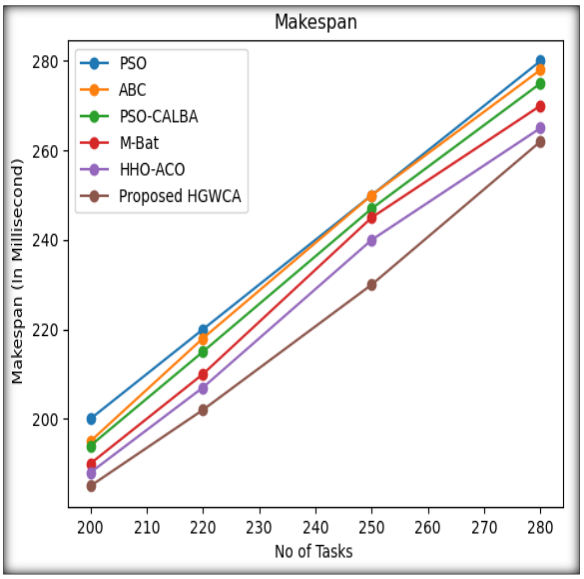


Figure 7: Makespan based on no. of task (200-280)

Data on the number of finished tasks and the accompanying makespan (in milliseconds) for various algorithms are shown in Figure 7 and Figure 8. Across all sets of finished tasks, the "Proposed model" algorithm consistently obtains the lowest makespan values, demonstrating economical execution time. The "PSO" method, on the other hand, continuously displays increasing makespan numbers, suggesting a comparatively slower execution time. According to this data, the "PSO" method performs relatively worse than the "Proposed model" approach in terms of minimizing execution time. The average turnaround time of several algorithms for varying amounts of tasks is shown in Figure 6 and Table 6. Across all task sets, the "Proposed model" algorithm consistently shows the lowest average turnaround time, suggesting effective completion. The "PSO" method, on the other hand, typically shows greater average turnaround time values, suggesting comparatively slower task completion. According to these findings, the "Proposed model" algorithm does exceptionally well in reducing average turnaround time, whereas the "PSO" method typically performs worse in this area.

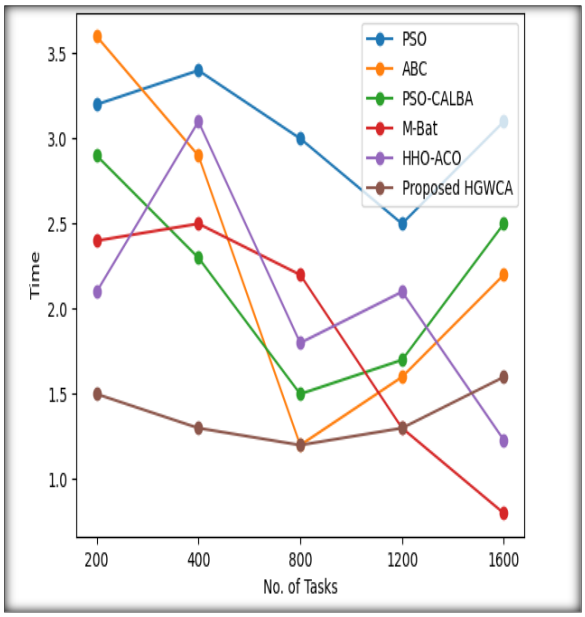


Figure 8: Turnaround time of tasks

Data on cloudlet turnaround times for different algorithms and task quantities are included in Table 6 and Figure 8. The "Proposed model" algorithm's efficiency in finishing jobs is demonstrated by the fact that it consistently displays the quickest turnaround time values for all task sets. The "M-Bat" algorithm performs competitively in contrast, whereas the "PSO" and "HHO-ACO" algorithms typically have greater turnaround times. The "Proposed model" algorithm is the most effective at reducing cloudlet turnover time overall.

Table 6: Throughput time of tasks

No. of Task	**200**	**400**	**800**	**1200**	**1600**
PSO	3.5	5.8	6.5	5.1	6.6
ABC	2.2	2.4	2.1	2.6	2.9
PSO-CALBA	3.1	4.2	6.7	6.8	8.6
M-Bat	2.9	2.8	2.8	2.2	3.1
HHO-ACO	0.67	0.45	0.58	0.99	0.23
Proposed model	3.4	5.9	6.6	8.1	7.2

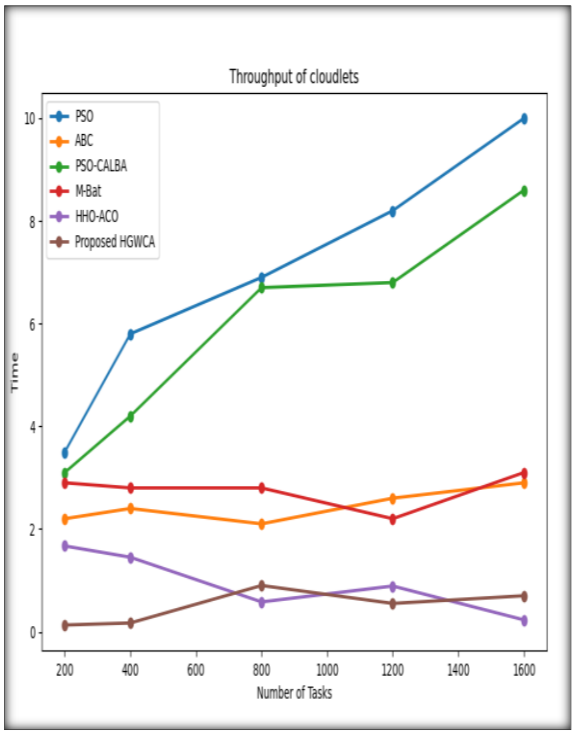


Figure 9: Throughput time of tasks

Table 7 and Figure 9 analysis show that the algorithms' throughput times vary depending on the work quantity. Across all task sets, the "HHO-ACO" algorithm consistently shows the lowest throughput time values, demonstrating its effectiveness in swiftly processing cloudlets. On the other hand, when compared to other algorithms, the "PSO-CALBA" algorithm typically displays higher throughput times, indicating slower processing. Across a range of task amounts, the "ABC" and "M-Bat" algorithms exhibit competitive performance with comparatively shorter throughput times. Although the "Proposed model" algorithm's throughput time numbers vary based on the number of tasks, it works reasonably well.

Table 7: Degree of imbalance of cloudlets

No. of Task	PSO	ABC	PSO-CALBA	M-Bat	HHO-ACO	Proposed model
200	2.8	2.6	2.9	2.4	0.66	0.87
400	3.4	2.9	2.3	2.5	1.2	0.8
800	2.7	1.2	1.5	2.2	0.9	0.5
1200	3	1.8	0.7	1.3	0.31	0.39
1600	3.1	2.2	2.5	0.8	0.21	0.19

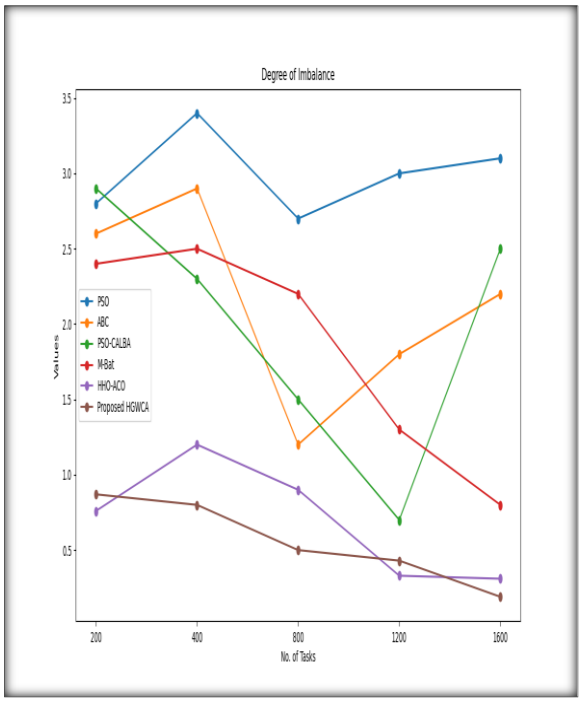


Figure 10: Degree of imbalance

It seems that the "Proposed model" algorithm does good tasks at attaining a decreased degree of imbalance across different task quantities, based on the numbers in Table 7 and Figure 10. Table 2 to 7 and Figure 4 to 10 demonstrate that, in comparison to the other methods on the list, the suggested approach consistently produces the lowest makespan value. The makespan is the amount of time needed to do every task in a scheduling problem. The algorithm is thought to be more efficient the lower the makespan. The superiority of the suggested algorithm holds true for tasks varying in size from 100 to 2000. This illustrates how resilient and dependable it is when managing different workload scenarios. It is regarded as the ideal option because it may consistently produce the lowest makespan value, demonstrating the best possible resource utilization and tasks scheduling.

5 Conclusions and future work

In conclusion, using the Grey Wolf Optimizer (GWO) and Cat Swarm Optimization (CSO) algorithms to boost makespan and throughput in cloud data centers has produced promising results. It has been shown that using these ingenious methods can optimize resource allocation and task scheduling, resulting in a reduction of makespan and an increase in throughput. The study's findings show how effectively GWO and CSO investigate and make use of the solution space while accounting for several cloud-specific characteristics.

Through clever resource allocation and job scheduling, these algorithms may enhance cloud data center performance, leading to higher customer satisfaction and productivity. Further research in this area is necessary to improve optimization techniques and sustain continuous cloud advancement data center operations. Restrictions In order to improve makespan in cloud data centers, this study introduces the HGWCA approach, which combines Grey Wolf Optimization with Cat Swarm Optimization. However, it faces two significant challenges. First off, there is not enough testing to determine how well the algorithm performs in actual cloud data centers. Second, due of its unclear scalability in larger data centers with numerous virtual machines, its effectiveness in managing extensive cloud settings is questioned. To confirm the algorithm's scalability and feasibility, these issues must be fixed. Future studies on improving makespan and throughput in cloud data centers can focus on several key areas by utilizing GWO and CSO algorithms. Among these are the following: enhancing algorithmic techniques through machine learning and hybridization; enabling dynamic resolution of scalability concerns in large data centers; exploring multi-objective optimization to consider additional performance objectives; managing real-time optimization to deal with evolving environments; and conducting practical implementation and validation studies to assess practical appropriateness. By taking these routes, scholars can contribute to the development of more reliable and effective cloud computing infrastructures that satisfy the evolving needs of cloud data center providers and users.

References

[1] Li, P., Li, J., Huang, Z., Li, T., Gao, C. Z., Yiu, S. M., & Chen, K. (2017). Multi-key privacy-preserving deep learning in cloud computing. Future Generation Computer Systems, 74, 76-85. https://doi.org/10.1016/j.future.2017.02.006

[2] Jivanadham, L. B., Islam, A. M., Katayama, Y., Komaki, S., & Baharun, S. (2013, May). Cloud Cognitive Authenticator (CCA): A public cloud computing authentication mechanism. In 2013 International Conference on Informatics, Electronics and Vision (ICIEV) (pp. 1-6). IEEE. https://doi.org/10.1109/iciev.2013.6572626

[3] Al-Maytami, B. A., Fan, P., Hussain, A., Baker, T., & Liatsis, P. (2019). A task scheduling algorithm with improved makespan based on prediction of tasks computation time algorithm for cloud computing. IEEE Access, 7, 160916-160926. https://doi.org/10.1109/access.2019.2948704

[4] Markovic, D. S., Zivkovic, D., Branovic, I., Popovic, R., & Cvetkovic, D. (2013). Smart power grid and cloud computing. Renewable and Sustainable Energy Reviews, 24, 566-577. https://doi.org/10.1016/j.rser.2013.03.068

[5] Abid, A., Manzoor, M. F., Farooq, M. S., Farooq, U., & Hussain, M. (2020). Challenges and Issues of Resource Allocation Techniques in Cloud Computing. KSII Transactions on Internet & Information Systems, 14(7). https://doi.org/10.3837/tiis.2020.07.005

[6] Ujjwal, K. C., Garg, S., Hilton, J., Aryal, J., & Forbes-Smith, N. (2019). Cloud Computing in natural hazard modeling systems: Current research trends and future directions. International Journal of Disaster Risk Reduction, 38, 101188. https://doi.org/10.1088/1475-7516/2020/07/005

[7] Takahashi, T., Blanc, G., Kadobayashi, Y., Fall, D., Hazeyama, H., & Matsuo, S. I. (2012, April). Enabling secure multitenancy in cloud computing: Challenges and approaches. In 2012 2nd Baltic Congress on Future Internet Communications (pp. 72-79). IEEE. https://doi.org/10.1109/bcfic.2012.6217983

[8] Yang, C., Goodchild, M., Huang, Q., Nebert, D., Raskin, R., Xu, Y., ... & Fay, D. (2011). Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing? International Journal of Digital Earth, 4(4), 305-329. https://doi.org/10.1080/17538947.2011.587547

[9] Abdel-Basset, M., El-Shahat, D., El-Henawy, I., De Albuquerque, V. H. C., & Mirjalili, S. (2020). A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection. Expert Systems with Applications, 139, 112824. https://doi.org/10.1016/j.eswa.2019.112824

[10] Zamfirache, I. A., Precup, R. E., Roman, R. C., & Petriu, E. M. (2022). Policy iteration reinforcement learning-based control using a grey wolf optimizer algorithm. Information Sciences, 585, 162-175. https://doi.org/10.1016/j.ins.2021.11.051

[11] Rodríguez, L., Castillo, O., Soria, J., Melin, P., Valdez, F., Gonzalez, C. I., ... & Soto, J. (2017). A fuzzy hierarchical operator in the grey wolf optimizer algorithm. Applied Soft Computing, 57, 315-328. Faris, H., Aljarah, I., Al-Betar, M. A., & Mirjalili, S. (2018). Grey wolf optimizer: a review of recent variants and applications. Neural computing and applications, 30, 413-435. https://doi.org/10.1016/j.asoc.2017.03.048

[12] Bezdan, T., Zivkovic, M., Bacanin, N., Strumberger, I., Tuba, E., & Tuba, M. (2022). Multi-objective task scheduling in cloud computing environment by hybridized bat algorithm. Journal of Intelligent & Fuzzy Systems, 42(1), 411-423. https://doi.org/10.3233/jifs-219200

[13] Sundas, A., Badotra, S., Alotaibi, Y., Alghamdi, S., & Khalaf, O. I. (2022). Modified Bat Algorithm for Optimal VM's in Cloud Computing. Computers, Materials & Continua, 72(2). https://doi.org/10.32604/cmc.2022.025658

[14] Raghavan, S., Sarwesh, P., Marimuthu, C., & Chandrasekaran, K. (2015, January). Bat algorithm for scheduling workflow applications in cloud. In 2015 International Conference on Electronic Design, Computer Networks & Automated

Verification (EDCAV) (pp. 139-144). IEEE. https://doi.org/10.1109/edcav.2015.7060555

[15] Fahim, Y., Rahhali, H., Hanine, M., Benlahmar, E. H., Labriji, E. H., Hanoune, M., & Eddaoui, A. (2018). Load Balancing in Cloud Computing Using Meta-Heuristic Algorithm. Journal of Information Processing Systems, 14(3). https://doi.org/10.1109/cist.2014.7016608

[16] Senthil Kumar, A. M., Padmanaban, K., Velmurugan, A. K., Asha Shiny, X. S., & Anguraj, D. K. (2023). A novel resource management framework in a cloud computing environment using hybrid cat swarm BAT (HCSBAT) algorithm. Distributed and Parallel Databases, 41(1-2), 53-63. https://doi.org/10.1007/s10619-021-07339-w

[17] Bin, N. I. N. G., Qiong, G. U., Zhao, W. U., Lei, Y. U. A. N., & Chun-yang, H. U. (2015). Bats algorithm research in cloud computing resource scheduling based on membrane computing. Application Research of Computers/Jisuanji Yingyong Yanjiu, 32(3). https://doi.org/10.4028/www.scientific.net/amr.989-994.2192

[18] Ganne, A. (2022). Emerging Business Trends in Cloud Computing. International Research Journal of Modernization in Engineering Technology, 4(12). https://doi.org/10.56726/irjmets32082

[19] Gundu, S. R., Panem, C. A., Thimmapuram, A., & Gad, R. S. (2022). Emerging computational challenges in cloud computing and RTEAH algorithm based solution. Journal of Ambient Intelligence and Humanized Computing, 1-15. https://doi.org/10.1007/s12652-021-03380-w

[20] Alam, T., Gupta, R., Qamar, S., & Ullah, A. (2022). Recent applications of Artificial Intelligence for Sustainable Development in smart cities. In Recent Innovations in Artificial Intelligence and Smart Applications (pp. 135-154). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-14748-7_8

[21] Ullah, A., & Chakir, A. (2022). Improvement for tasks allocation system in VM for cloud datacenter using modified bat algorithm. Multimedia Tools and Applications, 81(20), 29443-29457. https://doi.org/10.1007/s11042-022-12904-1

[22] Kumar, R., Bhardwaj, D., & Joshi, R. (2022). Adaptive bat optimization algorithm for efficient load balancing in cloud computing environment. In Advances in Computational Intelligence and Communication Technology: Proceedings of CICT 2021 (pp. 357-369). Singapore: Springer Singapore. https://doi.org/10.1007/978-981-16-9756-2_35

[23] Li, X., Lu, Y., Fu, X., & Qi, Y. (2021). Building the Internet of Things platform for smart maternal healthcare services with wearable devices and cloud computing. Future Generation Computer Systems, 118, 282-296. https://doi.org/10.1016/j.future.2021.01.016

[24] Krishnamoorthy, P. (2021). Performance Analysis of Hybrid BAT Algorithm and Cuckoo Search Algorithm [HB-CSA] for Task Scheduling in Mobile Cloud Computing. Available at SSRN 3997784. https://doi.org/10.2139/ssrn.3997784

[25] Gundu, S. R., Panem, C. A., & Thimmapuram, A. (2020). Hybrid IT and multi cloud an emerging trend and improved performance in cloud computing. SN Computer Science, 1(5), 256. https://doi.org/10.1007/s42979-020-00277-x

[26] Ullah, A., Nawi, N. M., & Khan, M. H. (2020). BAT algorithm used for load balancing purpose in cloud computing: an overview. International Journal of High-Performance Computing and Networking, 16(1), 43-54. https://doi.org/10.1504/ijhpcn.2020.110258

[27] Ibrahim, L. M., & Saleh, I. A. (2020). A solution of loading balance in cloud computing using optimization of bat swarm algorithm. Journal of Engineering Science and Technology, 15(3), 2062-2076. https://doi.org/10.5220/000767440058006

[28] Chung, K., & Park, R. C. (2019). Chatbot-based heathcare service with a knowledge base for cloud computing. Cluster Computing, 22, 1925-1937. https://doi.org/10.1007/s10586-018-2334-5

[29] Jian, C., Chen, J., Ping, J., & Zhang, M. (2019). An improved chaotic bat swarm scheduling learning model on edge computing. IEEE Access, 7, 58602-58610. https://doi.org/10.1109/access.2019.2914261

[30] Patil, R., Dudeja, H., & Modi, C. (2019). Designing an efficient security framework for detecting intrusions in virtual network of cloud computing. Computers & Security, 85, 402-422. https://doi.org/10.1016/j.cose.2019.05.016

[31] Lu, C., Gao, L., & Yi, J. (2018). Grey wolf optimizer with cellular topological structure. Expert Systems with Applications, 107, 89-114. https://doi.org/10.1016/j.eswa.2018.04.012

[32] Gawali, M. B., & Shinde, S. K. (2018). Task scheduling and resource allocation in cloud computing using a heuristic approach. Journal of Cloud Computing, 7(1), 1-16. https://doi.org/10.1186/s13677-018-0105-

[33] Attaran, M. (2017). Cloud computing technology: leveraging the power of the internet to improve business performance. Journal of International Technology and Information Management, 26(1), 112-137. https://doi.org/10.58729/1941-6679.1283

[34] Singh, P., Dutta, M., & Aggarwal, N. (2017). A review of task scheduling based on meta-heuristics approach in cloud computing. Knowledge and Information Systems, 52, 1-51. https://doi.org/10.1007/s10115-017-1044-2

[35] Zhang, Y., Liu, Z., Yu, F., & Jiang, T. (2017). Cloud computing resources scheduling optimisation based on improved bat algorithm via wavelet perturbations. International Journal of High-Performance Systems Architecture, 7(4), 189-196. https://doi.org/10.1504/ijhpsa.2017.092385

[36] Arunarani, A. R., Manjula, D., & Sugumaran, V. (2017). FFBAT: A security and cost-aware workflow scheduling approach combining firefly and bat algorithms. Concurrency and Computation:

Practice and Experience, 29(24), e4295. https://doi.org/10.1002/cpe.429

[37] Mittal, N., Singh, U., & Sohi, B. S. (2016). Modified grey wolf optimizer for global engineering optimization. Applied Computational Intelligence and Soft Computing, 2016. https://doi.org/10.1155/2016/7950348

[38] Bouyer, A., & Arasteh, B. (2014). The necessity of using cloud computing in educational system. Procedia-Social and Behavioral Sciences, 143, 581-585. https://doi.org/10.1016/j.sbspro.2014.07.440

[39] Dillon, T., Wu, C., & Chang, E. (2010, April). Cloud computing: issues and challenges. In 2010 24th IEEE international conference on advanced information networking and applications (pp. 27-33). Ieee. https://doi.org/10.1109/aina.2010.187

[40] Randles, M., Lamb, D., & Taleb-Bendiab, A. (2010, April). A comparative study into distributed load balancing algorithms for cloud computing. In 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (pp. 551-556). IEEE. https://doi.org/10.1109/waina.2010.85

[41] Ullah, A., Razak, S. F. A., Yogarayan, S., & Sayeed, M. S. (2025). Modified Neural Network Used for Host Utilization Predication in Cloud Computing Environment. Computers, Materials & Continua, 82(3). https://doi.org/10.32604/cmc.2025.059355

[42] Remmach, H., Razak, S. F. A., Ullah, A., Yogarayan, S., Sayeed, M. S., & Mrhari, A. (2025). CNN-Based Multi-Output and Multi-Task Regression for Supershape Reconstruction from 3D Point Clouds. Informatica, 49(5). https://doi.org/10.31449/inf.v49i5.6863

[43] Ullah, A., Alam, T., Aziza, C., Sebai, D., & Abualigah, L. (2024). A Hybrid Strategy for Reduction in Time Consumption for Cloud Datacenter Using HMBC Algorithm. Wireless Personal Communications, 137(4), 2037-2060. https://doi.org/10.1007/s11277-024-11395-7

[44] Alam, T., Gupta, R., Nasurudeen Ahamed, N., Ullah, A., & Almaghthwi, A. (2024). Smart mobility adoption in sustainable smart cities to establish a growing ecosystem: Challenges and opportunities. MRS Energy & Sustainability, 11(2), 304-316. https://doi.org/10.1557/s43581-024-00092-4

[45] Ullah, A., Alomari, Z., Alkhushayni, S., Al-Zaleq, D. A., Bany Taha, M., & Remmach, H. (2024). Improvement in task allocation for VM and reduction of Makespan in IaaS model for cloud computing. Cluster Computing, 27(8), 11407-11426. https://doi.org/10.1007/s10586-024-04539-8

[46] Alam, T., Ullah, A., & Benaida, M. (2023). Deep reinforcement learning approach for computation offloading in blockchain-enabled communications systems. Journal of Ambient Intelligence and Humanized Computing, 14(8), 9959-9972. https://doi.org/10.1007/s12652-021-03663-2

[47] Ouhame, S., Hadi, Y., & Ullah, A. (2021). An efficient forecasting approach for resource utilization in cloud data center using CNN-LSTM model. Neural Computing and Applications, 33(16), 10043-10055. https://doi.org/10.1007/s00521-021-05770-9

[48] Clemons, E. K., & Chen, Y. (2011, January). Making the decision to contract for cloud services: Managing the risk of an extreme form of IT outsourcing. In 2011 44th Hawaii International Conference on System Sciences (pp. 1-10). IEEE. https://doi.org/10.1109/hicss.2011.292

[49] Yao, J., & He, J. H. (2012, April). Load balancing strategy of cloud computing based on artificial bee algorithm. In 2012 8th International conference on computing technology and information management (NCM and ICNIT) (Vol. 1, pp. 185-189). IEEE. https://doi.org/10.3724/sp.j.1087.2012.0244

[50] Sajid, M., & Raza, Z. (2013, December). Cloud computing: Issues & challenges. In International conference on cloud, big data and trust (Vol. 20, No. 13, pp. 13-15). sn. https://doi.org/10.1201/b16318-3

[51] Pawar, C. S., & Wagh, R. B. (2013, March). Priority based dynamic resource allocation in cloud computing with modified waiting queue. In 2013 International Conference on Intelligent Systems and Signal Processing (ISSP) (pp. 311-316). IEEE. https://doi.org/10.1109/issp.2013.6526925

[52] Chauhan, R., & Kumar, A. (2013, November). Cloud computing for improved healthcare: Techniques, potential and challenges. In 2013 E-health and bioengineering conference (EHB) (pp. 1-4). IEEE. https://doi.org/10.1109/ehb.2013.6707234

[53] Velugoti, S., & Vani, M. P. (2024). An approach for privacy preservation assisted secure cloud computation. Informatica, 47(10). https://doi.org/10.31449/inf.v47i10.4586

[54] Zhang, R., & Shi, W. (2020). Research on resource allocation and management of mobile edge computing network. Informatica, 44(2). : https://doi.org/10.31449/inf.v44i2.3166

[55] Debbi, H. (2021). Modeling and performance analysis of resource provisioning in cloud computing using probabilistic model checking. Informatica, 45(4). DOI: https://doi.org/10.31449/inf.v45i4.3308

https://doi.org/10.31449/inf.v49i28.8552 Informatica **49** (2025) 137–152 **137**

CSO-MAVNE: A Multi-Agent GAN-GCN Based Collaborative Virtual Network Embedding Model for Dynamic Resource Allocation

Guiyong Sheng*, Li Zhang, Jun Pan, Fujun Wang
Department of Information Engineering, Yangzhou Polytechnic Institute, Yangzhou 225127, China
E-mail: m18952594816_1@163.com, zhangl@ypi.edu.cn, panj@ypi.edu.cn, wangfj@ypi.edu.cn
*Corresponding author

Keywords: virtual network embedding, graph convolutional network, network resource allocation, multi-agent, node

Received: March 10, 2025

To solve the challenges of dynamic topology changes and multi-stage decision-making dependencies in virtual network embedding, a Collaborative Sequential Optimization Multi-Agent Virtual Network Embedding Model (CSO-MAVNE) is proposed. The model integrates three core mechanisms. The topological features of the underlying and virtual networks are extracted through graph convolutional networks. The virtual network topology graph is dynamically constructed and evaluated with the help of generative adversarial networks to optimize the embedding strategy. The distributed collaborative mapping of virtual nodes and links is realized by combining multi-agent reinforcement learning. The Internet Topology Zoo and Synthetic Topology datasets are used for experimental testing with DRL-VNE, MS-GCN, and D-GANE models. The experimental results show that when the number of requests is 300, the embedding success rate of CSO-MAVNE is 82.9%, the cost-benefit ratio is 4.5, the resource utilization rate is 92.5%, and the embedding time is 7.1 s. In the scenario of 10% link failure, the recovery rate remains above 91.7% and the embedding success rate reaches 86.5%. Under high load conditions, the request throughput is 267.9 times/second. The results show that this method is superior to traditional methods in terms of virtual network request embedding success rate, resource utilization rate, and dynamic environment adaptability, and provides a reliable solution for building an efficient and robust resource allocation mechanism.

Povzetek: Razvit je večagentni model CSO-MAVNE, ki združuje GAN, GCN in ojačitveno učenje za učinkovito in robustno razporejanje virov v virtualnih omrežjih z dinamično topologijo.

1 Introduction

With the rapid development of the Internet and the surge of information demand, the traditional network architecture is facing increasingly severe challenges. The continuous emergence of emerging technologies such as the Internet of Things, 5G communication, and cloud computing has led to a rapid increase in network traffic, while also placing higher demands on the flexibility of resource scheduling and management [1]. Traditional network architecture is difficult to adapt to dynamic and changing business requirements and heterogeneous environments due to its fixed hardware resources and rigid configuration [2]. To address these issues, network virtualization technology has gradually become a research hotspot. It abstracts physical resources into multiple logical network slices, providing the possibility of resource sharing and customized services, and providing theoretical and practical support for next-generation network design [3]. In the framework of network virtualization, Virtual Network Embedding (VNE) is the core link of resource allocation, which directly affects the performance and resource utilization rate of the network [4]. The VNE problem involves mapping virtual nodes to physical nodes and allocating virtual links to the paths of physical links, which is essentially a complex combinatorial optimization problem [5]. In recent years, with the development of artificial intelligence technology, significant progress has been made in solving VNE. Zhang W et al. proposed a reproducible multidimensional VNE problem and its solutions to address the scalability and diversity challenges faced by VNEs. Through topology aware preprocessing and embedding methods, in simulation experiments, the average embedding success rate, revenue, and revenue cost ratio were increased by 40.45%, 40.45%, and 299.03%, respectively, and the average cost rate was reduced by 64.16% [6]. Ogino N et al. proposed a distance-based elastic VNE method to address network connectivity issues after critical node failures during the VNE process. Problems were modeled through integer linear programming, and the embedding process was optimized using efficient algorithms. The simulation results showed that VNE solutions with approximate ratios less than 1.2 were derived within 10 seconds [7]. Rezaeimoghaddam P et al. studied the VNE problem in wireless sensor networks and proposed a trust aware algorithm. By adding trust constraints to virtual nodes and links, efficient resource allocation could be achieved while improving network security. The results indicated that this method had significant advantages in throughput, error efficiency, and processing time [8]. Minardi M et al. proposed a dynamic topology aware embedding algorithm and its efficient solution method for the VNE problem in dynamic networks of non-

geostationary orbit satellites. The experiment results showed that the algorithm outperformed the baseline method in terms of computational efficiency, migration cost, and packet loss rate [9].

Graph Convolutional Network (GCN), as a deep learning technique capable of processing graph structured data, has significant advantages in extracting topological information and aggregating node features. Pasa L et al. proposed a new strategy for polynomial graph convolutional layers to address the issue of GCN mainly utilizing local topological information and introducing topological information bias when stacking multiple layers. By independently utilizing neighboring nodes with different topological distances in a single layer, decoupling representations could be generated to avoid the limitations of multi-layer stacking. The results indicated that this method outperformed traditional methods on multiple commonly-used graph classification benchmarks [10]. Bhattacharjee V et al. proposed an enhanced graph representation method to tackle the problem that traditional methods fail to adequately capture the structural similarity of node features in graph data. The method incorporated the concept of overlap into graph data to improve the accuracy of node classification. By inputting the enhanced graph into GCN, this method significantly improved the accuracy in classification tasks [11]. Liu Q et al. proposed a traffic condition prediction method that combined GCN and long short-term memory network. By jointly modeling the urban road topology and traffic flow dynamic characteristics, high-precision prediction of traffic trends was achieved [12]. Li S et al. proposed a dynamic data detection method that combined GCN and temporal convolutional network to address the problem of insufficient accuracy in dynamic data processing under network environment. This method modeled and analyzed network data from two dimensions, time and space, effectively improving the accuracy and processing efficiency of data detection [13].

Combined with Table 1, previous methods mostly use separate processing of node and link embedding, ignoring the coupling between the two in topology and resource dependency, resulting in local optimality and lack of global coordination in the embedding scheme. To overcome the above shortcomings, this study integrates the Generative Adversarial Network (GAN) and GCN for joint modeling, and proposes the Collaborative Generative and Graph Convolutional Network Embedding Algorithm (CGCNE). GCN extracts the structural features of the underlying network and virtual requests, and GAN is used to dynamically generate and optimize the virtual network topology, thereby achieving integrated coordination of node and link mapping and improving the global consistency of embedding and resource allocation efficiency. Finally, the key contents of the above VNE and GCN related methods are detailed in Table 1.

In addition, in the face of dynamic scenarios such as continuous expansion of network scale and frequent node/link failures in actual applications, the centralized embedding strategy performs poorly in terms of timeliness and robustness. To this end, the study further introduces a multi-agent reinforcement learning framework and proposes a Collaborative Sequential Optimization Multi-Agent Virtual Network Embedding Model (CSO-MAVNE). The self-attention mechanism is used to model the temporal dependencies in the embedded tasks, and the joint advantage function is used to coordinate the strategies of each agent to achieve efficient decoupling and collaborative optimization of virtual node and link mapping. This mechanism supports distributed deployment, adapts to dynamic network topology changes, and enhances the system's recovery capability and strategic robustness in fault scenarios.

To address the structural modeling deficiencies and policy coordination performance bottlenecks in VNE, this study proposes a joint modeling method that integrates generative adversarial networks and GCNs, and designs a collaborative decision-making mechanism in combination with multi-agent reinforcement learning to improve the distributed optimization capabilities of the system in resource scheduling tasks. The main innovations include: (1) Introducing GAN and GCN into embedding modeling to improve the collaborative perception capabilities of nodes and links; (2) Constructing a multi-agent architecture based on a joint advantage function to achieve policy distribution optimization and collaborative improvement; (3) Designing a dynamic sequence input mechanism to enhance the model's stability in handling burst requests. Experimental results show that CSO-MAVNE outperforms existing methods in multiple dimensions, including resource utilization, acceptance rate, embedding efficiency, and fault-tolerant recovery.

Table 1: Summary table of related work.

Author	Method	Dataset	Evaluation metrics	Results summary
Zhang W et al. [6]	ReMiDvne	OpenStack platform	Average rates of acceptance ratio,communication costs	Reduce communication costs by up to 45.93 and 63.43 percent
Ogino N [7]	Distance-based resilient VNE	Synthetic topology	Computation time, Approximation ratio	Approximate solution with ratio <1.2 within 10s
Rezaeimoghaddam P et al. [8]	TA-VWSN	WSN	Average network throughput, and processing time	Improved throughput, reduced errors, better efficiency

Minardi M et al. [9]	DTA-VNE	MIRSAT	quantify the packet lost	Reduce the packet lost by ~ 2.5 – 5% compared to baselines.
Pasa L et al. [10]	GNN with single PGC layer	Graph classification benchmarks	Classification accuracy	Outperformed conventional GCNs
Bhattacharjee V et al. [11]	Enhanced graph representation	Real-world graphs	Node classification accuracy	Improved GCN accuracy
Liu Q et al. [12]	LSTM - GCN	Real traffic data	Prediction accuracy, Reliability	High accuracy and adaptability
Li S et al. [13]	Proposed GCN + temporal convolution	Simulated dynamic network data	MAE, MAPE, RMSE, Processing time	Higher accuracy and lower delay

2 Methods and materials

To solve the global optimization problems caused by the decoupling of virtual node and link mapping and improve the system's adaptability in large-scale dynamic environments, the following core model modules are designed. First, to solve the problem of separation of node and link mapping in traditional methods, the CGCNE method (Section 2.1) is proposed to enhance the coordination and structural perception of embedded strategies. Second, to deal with the problems of slow response and poor robustness of centralized strategies, the CSO-MAVNE model (Section 2.2) is constructed to realize distributed decision-making and dynamic adaptation of resource scheduling.

2.1 Joint modeling VNE method based on GAN and GCN

Traditional VNE methods often adopt a staged processing mode, modeling node mapping and link mapping separately, but lack effective decision coordination between the two stages, resulting in insufficient global optimization capability for resource allocation [14-15]. A joint modeling VNE method is proposed to address this issue, which optimizes the mapping decisions of nodes and links simultaneously to avoid segmentation and limitations in the solution space. GAN and GCN are introduced, with their powerful feature extraction capabilities being utilized to capture network topology information and design more coordinated embedding strategies.

To capture the structural dependency and dynamic characteristics between physical links, three interaction relationship modeling methods are constructed: equation (1) describes the resource competition intensity between links, equation (2) describes the link dependency caused by upstream traffic, and equation (3) reflects the degree of coordination between bidirectional links. The three formulas jointly generate a link similarity matrix, which is used as a graph structure prior input to the GCN and GAN modules to guide the network to more accurately perceive the topological structure and its resource status.

Firstly, in virtual network topology, different links may compete with each other due to the traffic allocation of the target node, and their weight matrix is shown in equation (1).

$$A_{E,(u,v),(v,w)} = \exp\left(- \frac{(T^{f+}(u) + T^{f+}(v) - M)^2}{\sigma^2} \right) \quad (1)$$

In equation (1), T is the time window, and $f+$ represents the positive direction link, that is, the direction in which data flows out from the current node. $T^{f+}(u)$ and $T^{f+}(v)$ respectively represent the total outflow of nodes u and v. M is the average traffic output of the node. σ is the standard deviation. Secondly, to capture the mutual influence between upstream and downstream nodes, node correlation is shown in equation (2).

$$A_{E,(u,v),(v,u)} = \exp\left(- \frac{(T^{f-}(u) + T^{f+}(v) - M)^2}{\sigma^2} \right) \quad (2)$$

In equation (2), $f-$ represents the reverse link, that is, the direction in which data flows toward the current node. $T^{f-}(u)$ represents the total incoming traffic of node u to describe the traffic interaction and influence mechanism between nodes. Finally, to characterize the correlation between bidirectional interactions of links, the weight matrix of bidirectional relationships is shown in equation (3).

$$A_{E,(u,v),(v,u)} = \exp\left(- \frac{(T^{f_v^+}(u) + T^{f_u^+}(v) - M)^2}{\sigma^2} \right) \quad (3)$$

In equation (3), $T^{f_v^+}(u)$ represents the traffic from node u to node v, and $T^{f_u^+}(v)$ represents the traffic from node v to node u. By constructing these three link modes, the competition, upstream and downstream, and bidirectional interaction relationships between links can be fully characterized. By combining the feature aggregation capability of GCN, these link features are extracted and fused to optimize the effectiveness of VNE. Subsequently, to further optimize VNE, GAN is introduced to automatically generate and evaluate the effectiveness of the network topology graph, as shown in Figure 1.

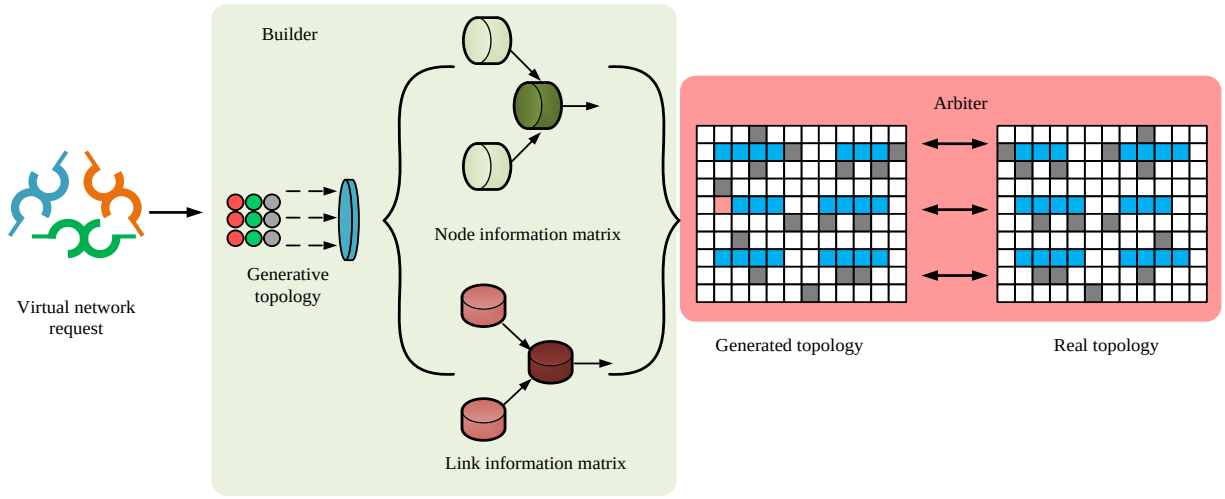


Figure 1: VNE topology generation model based on GAN.

As shown in Figure 1, it includes two core modules: builder G_θ and arbiter D_ϕ. The generator generates node information matrices and link information matrices based on the input network structure characteristics (including node degree distribution, topological connectivity, link bandwidth range, etc.) and virtual request information, which together constitute the simulated topology structure. Next, the generated topology is compared with the actual topology. This comparison result is then fed into the discriminator, which assesses the structural consistency between them. Based on this assessment, the discriminator provides feedback that guides the generator. Through adversarial training, the generator is prompted to continually enhance the structural rationality and distribution diversity of the topologies it produces. The whole process focuses on enhancing the structural authenticity of the generated topology rather than directly evaluating its mapping effectiveness.

The goal of Builder G_θ is to optimize the generated graph structure by minimizing the following objective function, as shown in equation (4) [16].

$$L(\theta) = E_t[\min(r_t(\theta)A_t, clip(r_t(\theta),1-\tau,1+\tau)A_t) + D_\phi(\pi_\theta(a_t|s_t))] \quad (4)$$

In equation (4), $r_t(\theta)$ is an estimate of the dominance function. A_t is the advantage of the state. $\pi_\theta(a_t|s_t)$ is the strategy probability, and $D_\phi(\pi_\theta(a_t|s_t)$ is the weighted loss term output by the discriminator. τ is the temperature coefficient, which is used to control the "smoothness" of the policy distribution and affects the

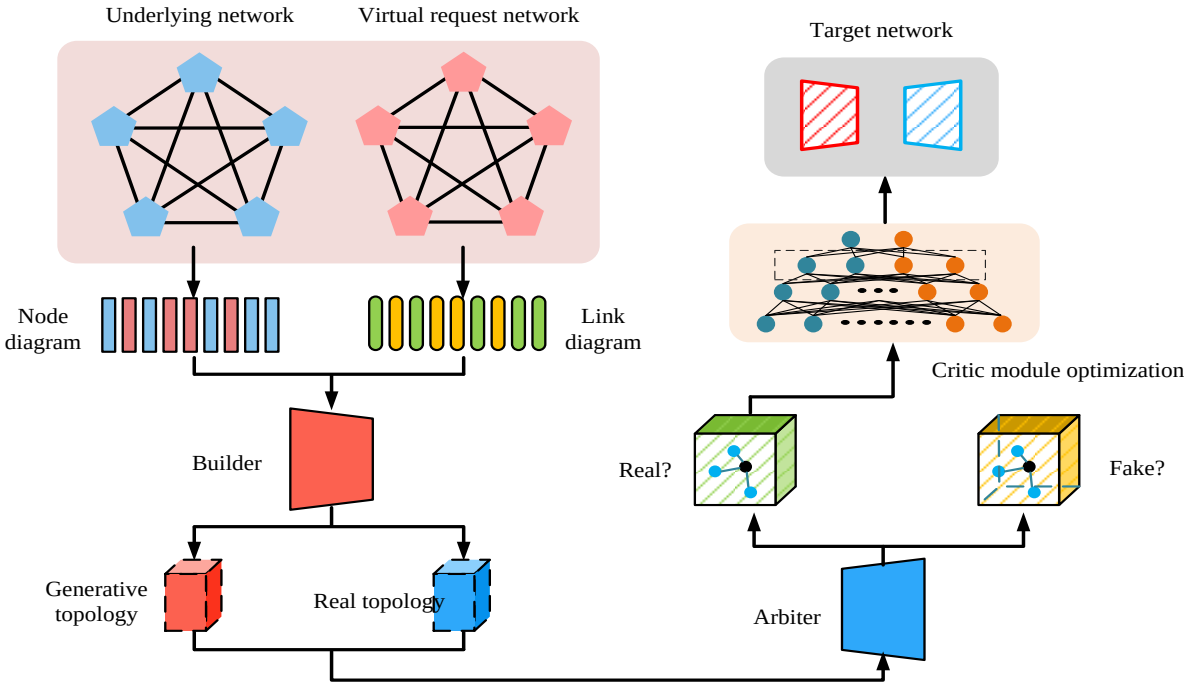


Figure 2: VNE algorithm framework based on CGCNE.

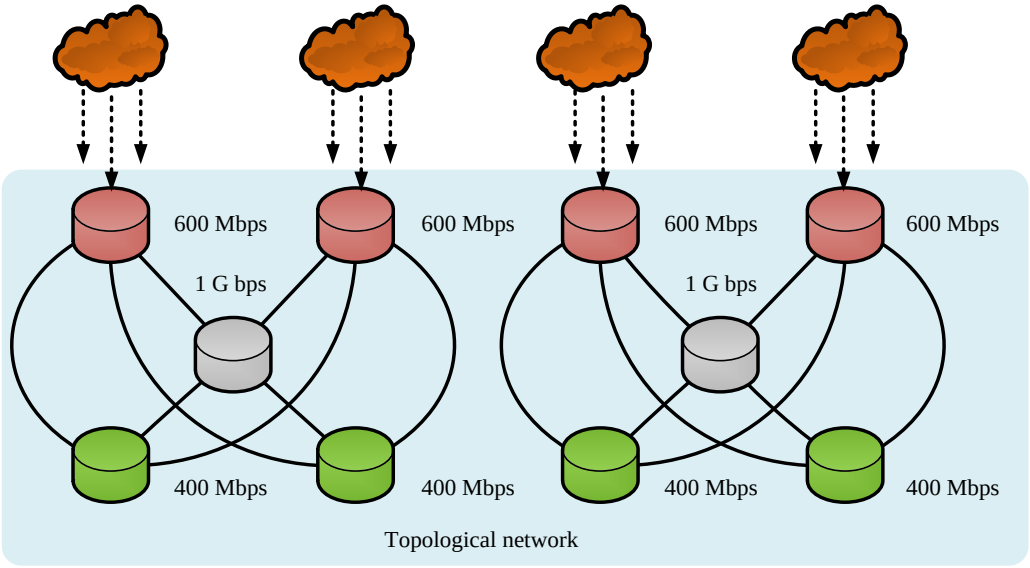


Figure 3: Information flow process of CGCNE framework.

trade-off between exploration and exploitation in policy sampling. A smaller value of τ will make the policy more focused on the dominant action and increase the tendency of deterministic selection; while a larger value of τ encourages the policy to explore more in the action space, thereby improving the generalization ability of the policy. The clipping operation mechanism is introduced in the formula to limit the amplitude of the policy update to avoid training instability caused by excessive updates during the policy optimization process. $clip(r_t(\theta),1-\tau,1+\tau)$ will limit the probability ratio $r_t(\theta)$ within the threshold range to ensure smooth convergence of the optimization process.

Subsequently, arbiter D_ϕ optimizes its ability to judge the true topology by minimizing the objective function, as shown in equation (5) [17-18].

$$L(\phi) = E_{\pi_\theta}\left[D_\phi(\pi_\theta(a_t|s_t))\right] - E_{x\sim P_r}\left[D_\phi(x)\right] \quad (5)$$

In equation (5), P_r is the true data distribution. D_ϕ is used to measure the difference between the generated topology and the real topology. In addition, to improve the learning performance of the builder, the Critic objective is achieved by minimizing empirical error, as shown in equation (6).

$$L_{critic} = E_t\left[(R(s_t,a_t)+\gamma V(s_{t+1})-V(s_t))^2\right] \quad (6)$$

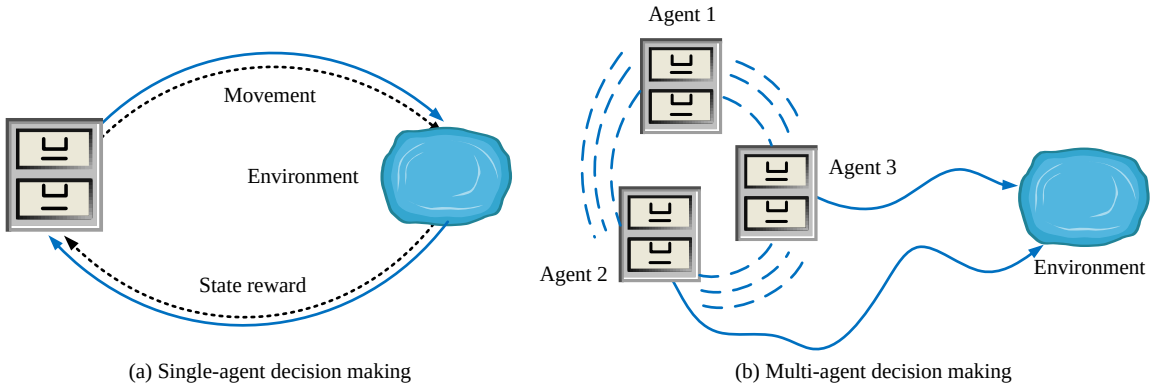
In equation (6), $R(s_t,a_t)$ represents the immediate reward for the current state and action. γ is the discount factor. $V(s_t)$ and $V(s_{t+1})$ are the value functions for the current and next states, respectively. Therefore, based on the above calculations, the VNE algorithm framework based on CGCNE is shown in Figure 2.

As shown in Figure 2, the CGCNE framework first extracts the structural features of virtual requests and the underlying network through GCN to generate a resource requirement matrix for nodes and links. The generator combines these features to generate candidate topologies, the discriminator determines the authenticity of its structure, and the Critic module compares the target topology and optimizes the generation scheme. Finally, the Arbiter combines the feedback from the discriminator and the Critic module to output an embedding strategy that better meets the requirements of the target network. To demonstrate more clearly the specific information transmission and resource allocation process between modules in the CGCNE framework, the information flow diagram is shown in Figure 3.

As shown in Figure 3, the CSO-MAVNE model receives the environment state and generates embedding decisions through a multi-agent structure, where each agent outputs the mapping object of the virtual node or link. After verification and fusion in the Critic module, the mapping relationship is passed to the resource execution module to complete the actual allocation of virtual resources to physical nodes and links. The resource state is refreshed using the mapping and subsequently relayed back to the agent, thus establishing a closed-loop optimization system. This process enables dynamic resource scheduling and adaptive embedding.

2.2 Distributed VNE strategy based on multi-agent reinforcement learning

Although the CGCNE strategy proposed in the previous section can improve the joint embedding quality of nodes and links, its modeling method based on centralized control has problems such as slow response speed and difficulty in policy convergence in large-scale and dynamic environments. In particular, it is difficult to achieve rapid adaptation and concurrent optimization in multi-region resource scheduling [19]. Therefore, based on the previous section, a multi-agent VNE model based on collaborative sequence optimization, CSO-MAVNE, is further proposed. Multiple agents collaborate to embed tasks in a phased manner. First, the node agent completes node mapping based on the physical resource status, and



(a) Single-agent decision making (b) Multi-agent decision making

Figure 4: Schematic diagram of single-agent and multi-agent reinforcement learning.

then the link agent selects the link path based on the node decision. The agents transmit feedback through shared state representation and centralized evaluation modules to achieve collaborative optimization between strategies and improve embedding adaptability. Multi-agent reinforcement learning can effectively adapt to the complexity and real-time requirements of VNE in dynamic environments through the collaboration of multiple agents, as shown in Figure 4. The environment includes resource state input, embedded action output, execution feedback and reward signals. The agent makes embedded decisions based on the current state and continuously optimizes the strategy through the reward mechanism.

In the scenario of a single-agent in Figure 4 (a), the agent selects actions based on the state information of the environment and obtains state rewards through interaction with the environment, thereby optimizing the strategy. In the multi-agent scenario shown in Figure 4 (b), the environmental state is not only influenced by the actions of individual agents, but also by the combined actions of other agents. This interactive characteristic determines that agents need to cooperate in learning to adapt to the globally optimal embedding strategy.

To better capture sequence dependencies in VNE problems, Self Attention Mechanism (SAM) is introduced as the core of multi-agent collaborative optimization in CSO-MAVNE. SAM can effectively handle global dependencies between sequence data, especially for tasks with complex interaction characteristics. As a typical application of self-attention mechanism, the Transformer model maps the input sequence to the latent representation space through the encoder-decoder structure and generates optimization strategies. The calculation is shown in equation (7).

$$Att(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (7)$$

In equation (7), Q, K, and V represent the query, key, and value, respectively. d_k is the dimension of the key vector, which indicates the feature dimension size of each position in the sequence and determines the information projection space in the attention calculation. The *softmax* function normalizes the dot product result into weights. In multi-agent scenarios, virtual nodes and links are treated as independent sequences, and SAM is used to achieve dependency modeling and global optimization between tasks.

On the basis of SAM, to solve the sequence dependency problem in VNE, the multi-agent strategy further formalizes the embedding task of virtual nodes and links as a sequential decision-making problem. Specifically, each agent is responsible for selecting the optimal strategy for mapping a virtual node or link to the underlying network in the current state. The local priority calculation of the intelligent agent is shown in equation (8).

$$A_{vne}(s,a_n,a_l) = A_{node}(s,a_n) + A_{link(s,a_l)} \quad (8)$$

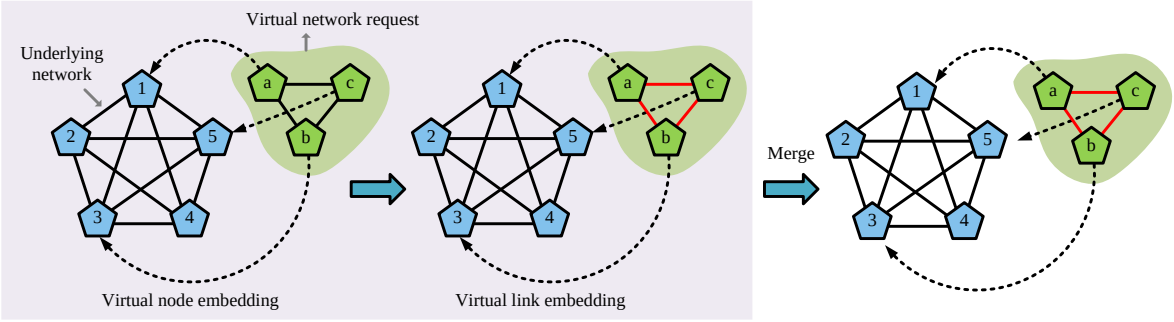


Figure 5: Virtual node and link embedding order decision framework.

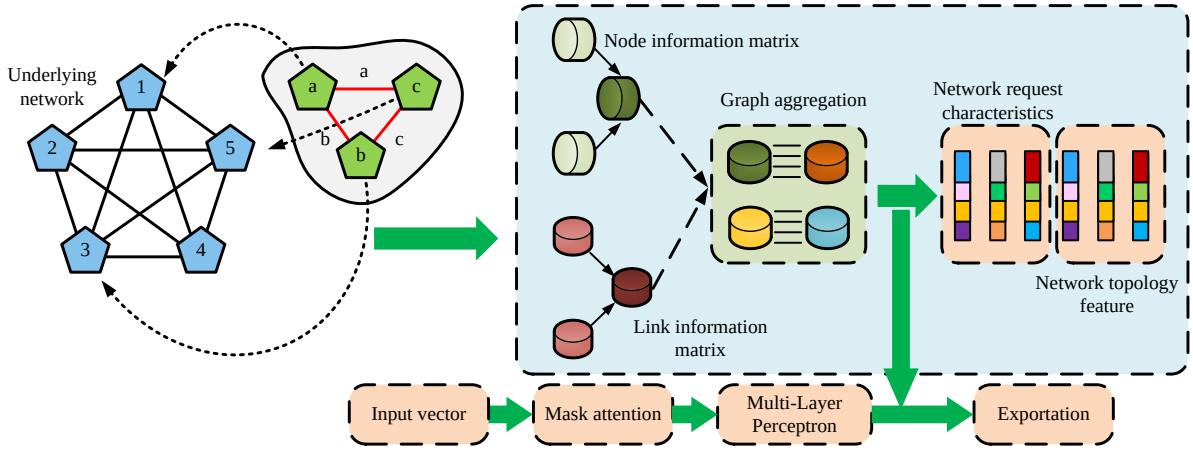


Figure 6: Encoder-decoder architecture for embedding virtual nodes.

In equation (8), s represents the current system operation. a_n and a_l are the embedding actions of virtual nodes and virtual links. $A_{node}(s,a_n)$ and $A_{link(s,a_l)}$ respectively measure the local priority of node and link embeddings. Furthermore, to ensure the collaborative consistency of multiple agents, a joint advantage function is proposed, as shown in equation (9).

$$A_{vne}^{global}(s,a_n,a_l)=\sum_i A_{vne}(s_i,a_n,a_l) \quad (9)$$

In equation (9), s_i represents the current state of agent i. By leveraging the joint advantage function, intelligent agents are able to harmonize their individual decision-making processes while taking into account overarching global objectives. Therefore, the sequential decision-making process and collaborative embedding logic of virtual nodes and links in CSO-MAVNE are shown in Figure 5.

As shown in Figure 5, in the embedding process, the virtual node agent first selects the appropriate physical node for mapping according to the node status. Then, based on the node mapping results, the virtual link agent selects the underlying link resources that meet the bandwidth and path constraints, as shown by the red line in the figure. Finally, the target mapping structure is formed through embedding and merging, providing optimization strategy support for subsequent resource allocation. To achieve efficient mapping of nodes and links in the sequential decision-making framework, an encoder decoder structure is introduced in multi-agent collaboration to dynamically extract network features and generate embedding strategies. The structure is shown in Figure 6.

As shown in Figure 6, for each virtual network request, the encoder extracts the structural features of nodes and links, forms an input vector and sends it to the decoder together with the request features. Initially, the decoder utilizes Mask attention to model the sequential dependencies among nodes. Subsequently, it integrates the structural information generated by the Graph aggregation module, resulting in a fused feature vector. Subsequently, the Multi-Layer Perceptron (MLP) generates the embedding action of the current virtual node based on the vector and outputs the embedding strategy until all nodes are embedded. After all virtual nodes complete the mapping, the decoding process stops and outputs the final embedding scheme. Firstly, the encoder uses the adaptive network feature extraction module GCN to encode the topological features of virtual network requests and underlying physical networks, learning the representational latent space. Encoding parameters are optimized by minimizing experience loss, as shown in equation (10).

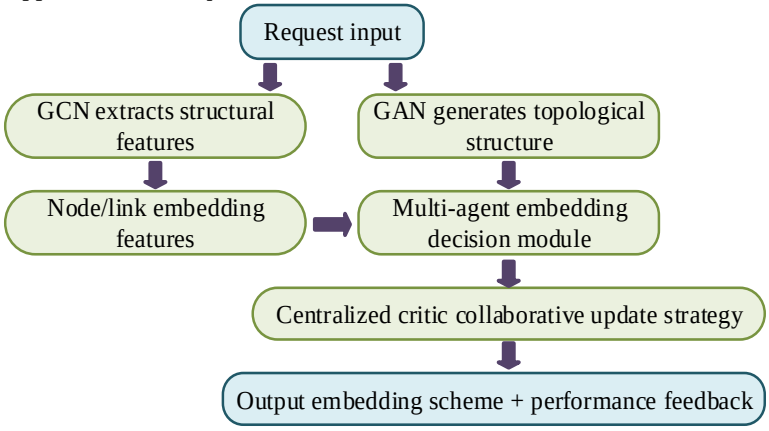


Figure 7: Overall architecture of CSO-MAVNE for VNE.

$$L_{encoder}(\phi)=\frac{1}{T}\sum_{t=0}^{T-1}\left[R(s_t,a_t)+\gamma V_\phi(s_{t+1})-V_\phi(s_t)\right]^2 \quad (10)$$

In equation (10), ϕ is the encoder parameter. $V_\phi(s_t)$ represents the estimated value of the current state. Each time step decoder receives the output features of the encoder and generates a set of embedded action sequences $a^m_{0:M}$ based on the virtual node or link set, maximizing the joint advantage function of the embedding decision, as shown in equation (11).

$$L(\theta)=\frac{1}{2T}\sum_{t=1}^{T-1}\sum_{m=1}^{M}\left[\min(r_t^m(\theta)A_t^m, clip(r_t^m(\theta),1-\tau',1+\tau')A_t^m)\right] \quad (11)$$

In equation (11), A_t^m is the dominant function. τ' controls the strategy generation scope. $r_t^m(\theta)$ is the ratio estimation of the strategy function. Finally, Figure 7 shows its system architecture and main processing stages.

As shown in Figure 7, the overall process of the CSO-MAVNE model includes three stages: feature extraction, candidate topology generation, and multi-agent embedding decision. GCN is used to extract the structural features of nodes and links, and the GAN module generates a well-structured embedding candidate topology. On this basis, a multi-agent reinforcement learning strategy is used to complete the collaborative embedding of nodes and links respectively. This method has the advantages of task decoupling, parallel execution, and strategy sharing, and can effectively improve the resource scheduling efficiency and robustness of the model in large-scale and dynamic environments.

3 Results

3.1 Performance testing of CSO-MAVNE network resource allocation model

The experiment was completed in Windows 11 environment using a desktop computer with 32GB memory and AMD Ryzen 9 5900X processor. The datasets used include typical physical topologies in Internet Topology Zoo and virtual topologies generated by Synthetic Topology based on Waxman model. The former covers the real network structure of operators in multiple regions, and the latter constructs 100 random topologies with a node size of 100–300. The link generation probability was set to 0.3, and the node and link resource requirements were sampled from the uniform distribution U(10,100). The corresponding physical network resources were randomly generated in the range of 1.5–2 times, and all topologies were normalized. Requests were entered into the system in time series order, without batch processing, to simulate the dynamic arrival process. Internet Topology Zoo was used to verify the resource allocation adaptability of the model under the real network structure, and Synthetic Topology was used to evaluate the performance stability and generalization ability of the model under different scales and topological complexities.

For training, the Adam optimizer was uniformly used, and the learning rates of the GCN and the generative adversarial network modules were 0.001 and 0.0002 respectively, and the batch size was 32. GCN used a two-layer graph convolution structure with output dimensions of 64 and 32 respectively, and the activation function was ReLU. The multi-agent strategy adopted the Actor-Critic architecture based on the advantage function, and used the centralized critic to achieve state sharing and independent policy update. Each round of training simulated 300 virtual network requests, with a cumulative number of interaction rounds of 50,000. Performance fluctuations of less than 0.01% for 20 consecutive rounds were considered convergence. The reward function combined resource utilization rate, benefit-cost ratio, and reconstruction overhead, and the basic comparison strategy was to prioritize the maximum residual resources.

To quantify the independent contribution of each key module to the model performance, ablation experiments were carried out, see Table 2.

The results showed that after removing the GAN module, the request throughput dropped to 164.9 req/s and the link utilization increased, indicating that the diversity of the generated structure decreased, resulting in a decrease in embedding efficiency. Removing the joint advantage function increased the number of remappings to 1.44, reflecting its optimization role in multi-agent collaboration. The lack of the self-attention mechanism mainly affected the accuracy of resource scheduling and link pressure distribution.

To verify the generation ability and structural effectiveness of the GAN module in the VNE task, the structural characteristics and training convergence of its output topology were further quantitatively analyzed. The results are shown in Table 3.

Table 2: Ablation study with supplementary metrics.

Model Variant	Request Throughput (req/s)	Avg. Link Utilization (%)	Avg. Remapping Attempts
CSO-MAVNE (Full Model)	183.4	42.1	1.12
w/o GAN (GCN only)	164.9	46.3	1.38
w/o Self-Attention	170.5	44.7	1.26
w/o Joint Advantage Func.	158.7	47.5	1.44

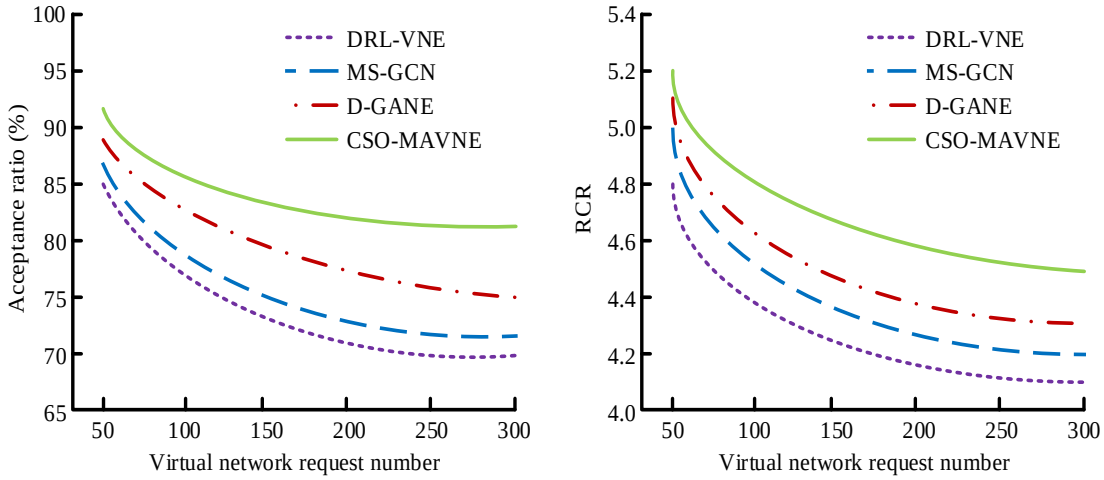
Table 3: Validation results of GAN-Generated topologies.

Metric Type	Real Topologies (Mean)	GAN-Generated Topologies (Mean)	Relative Difference
Average Node Degree	3.85	3.78	-1.80%
Degree Distribution Skewness	0.42	0.39	-7.10%
Average Clustering Coefficient	0.31	0.30	-3.20%
Link Density	0.22	0.23	4.50%

The results showed that the topological structure generated by GAN was highly similar to the real network topology in terms of overall morphology. The differences between key structural indicators such as node average degree, connection density and clustering coefficient and the real topology were all controlled within 5%, indicating that it had high structural fidelity. In addition, the degree distribution skewness was also consistent, indicating that the generated topology could better restore the connectivity pattern of the real network in terms of connection characteristics.

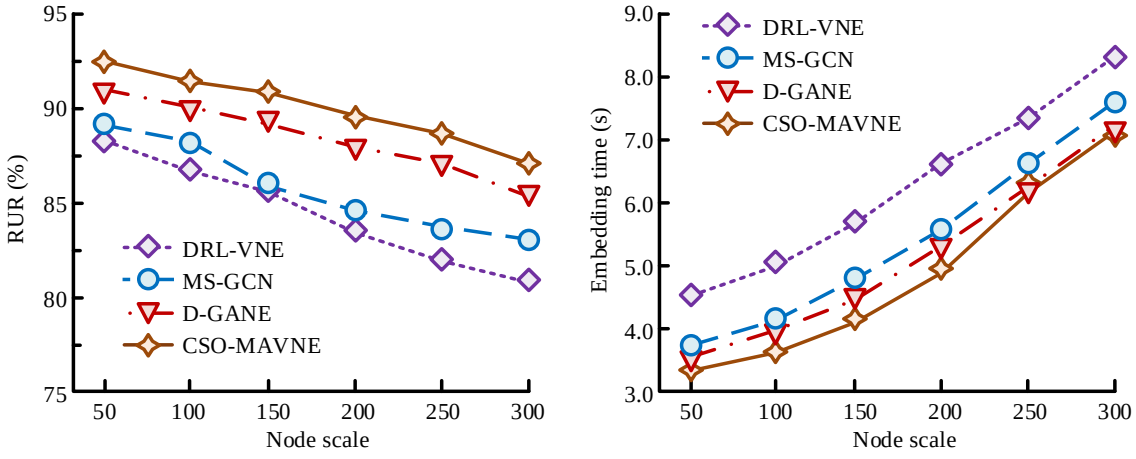
Deep Reinforcement Learning-based VNE (DRL-VNE), Multi-Stage GCN (MS-GCN), and Distributed GAN-based Embedding (D-GANE) were selected as comparative algorithms. DRL-VNE represents the traditional deep reinforcement learning embedding method, MS-GCN integrates graph structure perception capabilities, and D-GANE combines generation mechanism with deep strategy optimization. All three are currently mainstream single-agent VNE algorithms, which are typical and comparable. The bandwidth requirements for the links in the experiment were randomly distributed between 10Mbps and 1Gbps. Firstly, the resource allocation capability and efficiency results under different request scales are shown in Figure 8.

Figures 8(a) and 8(b) show the changes in embedding success rate and Revenue Cost Ratio (RCR) with the number of Virtual Network Requests (VNR), respectively. The number of VNRs refers to the number of virtual network requests received continuously by the system, and the test interval was set to 100 to 300.The embedding success rate is the ratio of successfully completed embedded requests to the total number of requests. The benefit-cost ratio is the ratio between resource benefits and resource overheads (ROs) during the embedding process. The higher the value, the higher the resource utilization rate. In Figure 8(a), when the number of VNRs was 300, the embedding success rates of DRL-VNE, MS-GCN, D-GANE, and CSO-MAVNE were 72.3%, 73.3%, 75.3%, and 82.9%, respectively. In Figure 8(b), when the number of VNRs was 300, the cost-benefit ratios of each model were 4.1, 4.2, 4.4, and 4.5, respectively. CSO-MAVNE utilized the collaborative feature extraction capabilities of GAN and GCN, as well as the collaborative optimization of multiple agents, effectively capturing the dependency relationships between nodes and links. D-GANE performed better in generating features, thanks to the optimization capabilities of its distributed builder. MS-GCN performed slightly worse due to insufficient modeling of topological dependencies. Additionally, it lacks effective global optimization strategies for resource allocation. DRL-VNE had the lowest embedding success rate, and its reinforcement learning model was susceptible to sparse feature interference when facing large-scale requests, making it difficult to adapt to the distributed embedding requirements of complex scenarios. Subsequently, the results of resource utilization rate and embedding time testing are shown in Figure 9.



(a) Embedding success rate test (b) RCR test

Figure 8: Embedding success rate and benefit-cost testing.



(a) RUR test (b) Embedding time test

Figure 9: Resource utilization rate and embedding time testing.

Figures 9(a) and 9(b) show the resource utilization rate and embedding time changes under different numbers of physical network nodes, respectively. Figures 9(a) and 9(b) show the changes in resource utilization rate and embedding time under different node sizes, respectively. Resource utilization refers to the ratio of the allocated node and link resources in the system to the total resources. Embedding time refers to the time required to complete a virtual network request embedding. The shorter the time, the stronger the response capability. In Figure 9(a), when the node size was 300, the resource utilization rates decreased to 81.0%, 83.0%, 85.7%, and 87.2%, respectively. In Figure 9(b), as the node size increased to 300, the embedding time increased to 8.2 seconds, 7.5 seconds, 7.2 seconds, and 7.1 seconds, respectively. The fast strategy generation and collaborative optimization strategy of CSO-MAVNE encoder-decoder structure achieved efficient allocation of global resources. Finally, multiple experimental scenarios were designed, namely static scenario fault free network, dynamic scenario 1 simulating 10% link failure, dynamic scenario 2 simulating 10% link failure, and dynamic scenario 3 simulating 10% node failure. The results are shown in Table 4.

Table 4: Embedding success rate and network stability test results.

Scenario	Index	Model	100 Nodes	200 Nodes	300 Nodes
Scenario 1	Embedding success rate (%)	DRL-VNE	84.3	83.2	82.1
		MS-GCN	87.5	86.5	85.5
		D-GANE	89.8	89.1	88.3
		CSO-MAVNE	92.0	91.2	90.5
Scenario 2	Embedding success rate (%)	DRL-VNE	79.1	78.3	77.8
		MS-GCN	83.2	82.1	81.2
		D-GANE	86.7	85.7	84.5
		CSO-MAVNE	88.5	87.8	86.5
	Recovery rate (%)	DRL-VNE	85.6	84.7	84.3
		MS-GCN	88.9	88.1	87.5
		D-GANE	90.4	89.7	88.5
		CSO-MAVNE	93.2	92.5	91.7
Scenario 3	Embedding success rate (%)	DRL-VNE	76.5	75.4	74.3
		MS-GCN	81.1	80.2	79.4
		D-GANE	84.3	83.5	82.5
		CSO-MAVNE	86.5	85.7	84.3
	Recovery rate (%)	DRL-VNE	89.3	88.5	88.1
		MS-GCN	85.4	84.6	84.3
		D-GANE	87.5	86.8	86.4
		CSO-MAVNE	90.5	89.7	88.5

	RO (%)	DRL-VNE	16.2	16.9	17.5
		MS-GCN	14.3	15.0	15.7
		D-GANE	12.1	12.8	13.5
		CSO-MAVNE	10.5	11.2	11.8

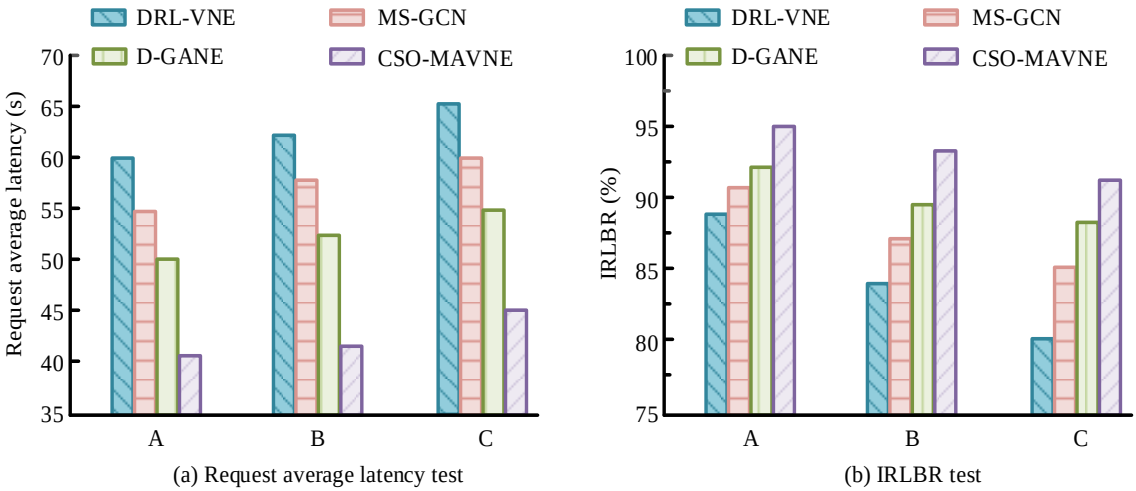
In Table 4, the embedding success rate indicated the proportion of virtual network mappings that were successfully completed under the initial conditions. The recovery rate indicated the proportion of affected requests that were successfully re-embedded after a link or node failure. The RO indicated the proportion of additional resource consumption required to maintain the service during failure recovery.

Under 300 nodes, the embedding success rate of CSO-MAVNE reached 90.5%, and the recovery rates were 91.7% (link failure) and 92.1% (node failure), respectively, with the lowest RO of only 11.8%. The recovery rates of D-GANE were 89.5% and 89.8%, respectively, and the RO was slightly higher. The recovery rate of MS-GCN in dynamic scenarios was maintained at around 85%, and the resource utilization rate was average. DRL-VNE had the lowest recovery rate, only 79.3%, and the highest RO, reaching 17.5%. The dynamic test used link and node failure scenarios to simulate common link interruption and device offline problems, respectively. It had the characteristics of high frequency, strong controllability, and clear interference in real networks, and was more suitable for evaluating the robustness and recovery capabilities of the model in emergency situations.

3.2 Application analysis of CSO-MAVNE network resource allocation model

To verify the optimization ability of the model in cross regional data center resource allocation, the scenario of A, B, and C regional data centers collaborating to process virtual network requests was simulated. Regions A, B, and C each contained 50, 70, and 100 computing nodes, with a total bandwidth of 1 Tbps, 1.5 Tbps, and 2 Tbps, respectively. The experiment randomly generated virtual network requests, which required computing resources, storage resources, and link bandwidth. The model was required to maximize resource allocation efficiency and maintain load balancing between regions while meeting resource requirements. Firstly, the average processing delay and Inter-region Load Balancing Rate (IRLBR) results are shown in Figure 10.

Figures 10(a) and 10(b) show the average processing delay and IRLBR results of each model in three regions, respectively. The average processing delay is the average processing time from the generation of a virtual network request to the completion of mapping. IRLBR was used to measure the balance of resource allocation between different data centers. A higher value indicates a more balanced resource allocation. In Figure 10(a), CSO-MAVNE performed the best in all three regions, with delays of 40ms, 42ms, and 45ms, respectively. The latency of D-GANE and MS-GCN was slightly higher in each region, ranging from 50-55ms and 55-60ms, respectively. DRL-VNE performed the worst, with delays reaching 60ms, 62ms, and 65ms in regions A, B, and C, respectively. In Figure 10(b), the IRLBR of CSO-MAVNE in regions A, B, and C reached 95%, 93%, and 91.5%, respectively. D-GANE performed second best, with IRLBRs of 92.3%, 89.5%, and 88% in the three regions. The equilibrium rate of MS-GCN further decreased, ranging from 90.8% to 85%, while the equilibrium rate of DRL-VNE was the lowest, at 88.5%, 84%, and 80%, respectively. CSO-MAVNE balanced the resource load of each region through a global optimization strategy, while DRL-VNE failed to effectively alleviate resource competition between regions due to policy limitations, resulting in insufficient balance.



(a) Request average latency test (b) IRLBR test

Figure 10: Average processing delay and IRLBR results.

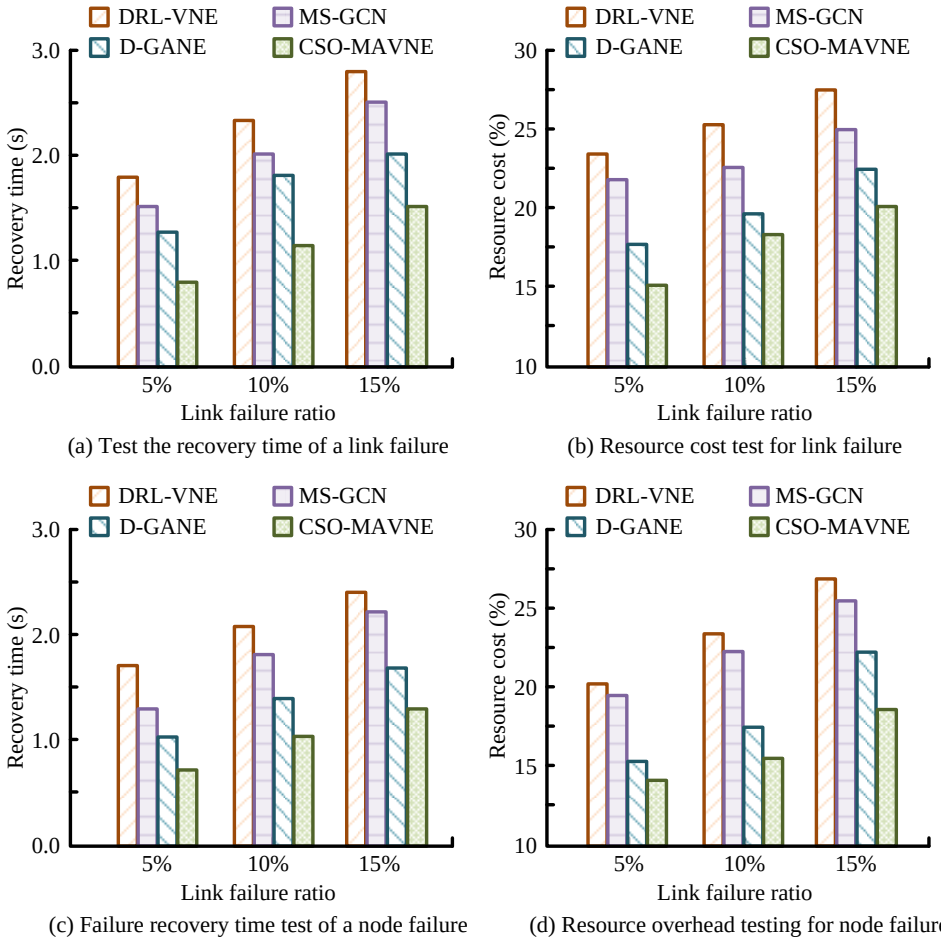


Figure 11: FRT and RO test results under dynamic failure scenarios.

Subsequently, tests were conducted on link failure and node failure in dynamic scenarios to simulate the fault recovery time (FRT) and RO of the model under different failure ratios. Scenario 1 was a dynamic link failure test, with simulated link failure rates of 5%, 10%, and 15%, respectively. Scenario 2 was a dynamic node failure test, with simulated node failure rates of 2%, 5%, and 8%, respectively. Under each condition, the model needed to perform real-time resource remapping, as shown in Figure 11.

Figures 11(a) and 11(b) show the FRT and RO results in link and node failure tests, respectively. FRT was used to measure the time required for the system to restore service from the occurrence of a fault, reflecting the system's response speed to emergencies. RO was used to describe the proportion of additional resources consumed in the process of completing fault repair, reflecting the system resource pressure brought by the fault tolerance mechanism. In Figure 11(a), the FRTs of CSO-MAVNE were 0.8s, 1.1s, and 1.5s, respectively, and the ROs were 15%, 18%, and 20%, respectively. In Figure 11(b), the recovery times of CSO-MAVNE were 0.7s, 1.0s, and 1.3s, respectively, and the ROs were 12%, 15%, and 18%, respectively. The FRT of DRL-VNE was the highest, at 1.6s, 2.0s, and 2.4s respectively, with ROs of 20%, 23%, and 27%, respectively. CSO-MAVNE achieved the lowest FRT and RO through collaborative optimization strategies among multiple agents and efficient dynamic remapping mechanisms. Finally, experiments were conducted to analyze the distribution of request flows and scheduling efficiency under different load conditions. The indicators include Regional Request Distribution Deviation (RDD), Average Latency (AL), Success Allocation Rate (SAR), and Request Throughput Rate (RTR). The results are shown in Table 5.

Table 5: Dynamic request flow distribution and scheduling efficiency results.

Load Level	Model	RDD (%)	AL (s)	SAR (%)	RTR (requests/s)
Low (100/s)	DRL-VNE	22.3	1.82	85.7	85.7
	MS-GCN	18.7	1.53	89.3	89.3
	D-GANE	15.2	1.19	92.8	92.8
	CSO-MAVNE	10.8	1.02	95.6	95.6

Medium (200/s)	DRL-VNE	26.7	2.48	82.1	164.2
	MS-GCN	22.1	2.23	85.4	170.8
	D-GANE	19.1	1.79	88.3	176.6
	CSO-MAVNE	12.6	1.51	91.7	183.4
High (300/s)	DRL-VNE	30.4	3.21	79.4	238.2
	MS-GCN	25.8	2.83	83.2	249.6
	D-GANE	20.7	2.41	86.1	258.3
	CSO-MAVNE	15.3	1.81	89.3	267.9

Table 6: Scalability and time complexity evaluation of different models.

Node Scale	Metric	DRL-VNE	MS-GCN	D-GANE	CSO-MAVNE
500	Throughput	176.2 ± 1.8	180.5 ± 1.5	182.1 ± 1.2	188.7 ± 1.0
	Runtime (s)	5.3	4.8	5	4.3
	P-value (vs CSO)	0.003	0.012	0.025	/
1000	Throughput	168.3 ± 1.5	175.9 ± 1.2	178.0 ± 1.1	183.4 ± 0.9
	Runtime (s)	9.6	8.9	8.3	7.5
	P-value (vs CSO)	0.0012	0.0074	0.019	/
1500	Throughput	151.9 ± 2.0	161.0 ± 1.8	165.7 ± 1.6	172.8 ± 1.2
	Runtime (s)	18.4	17.1	15.8	13.9
	P-value (vs CSO)	0.0008	0.0061	0.011	/

In Table 5, RDD is used to measure whether the request distribution between different regions is balanced. Its value is the standard deviation of the request distribution, expressed in percentage (%). The smaller the value, the more uniform the distribution. AL represents the average time from generation to response of all requests, expressed in seconds (s), reflecting the overall processing efficiency of the system. SAR is used to count the proportion of requests that are successfully mapped under the current load, expressed in percentage (%), which is a reflection of system stability. RTR measures the number of virtual network requests that the system can process per unit time, expressed in requests/s, and is an important indicator for evaluating the system's processing capabilities.

Under low load conditions, the RDD of CSO-MAVNE was 10.8%, AL was 1.02s, SAR reached 95.6%, and RTR was 95.6 request/s. Under medium load, CSO-MAVNE achieved the best overall performance with an RDD of 12.6%, AL of 1.51s, SAR of 91.7%, and RTR of 183.4 requests/s, outperforming all baseline models under the same load conditions. Under high load conditions, the RDD of CSO-MAVNE was 15.3%, AL was 1.81s, SAR was 89.3%, and RTR was 267.9 request/s, all of which were better than D-GANE and MS-GCN, while DRL-VNE performed the worst. CSO-MAVNE benefited from its efficient resource allocation and load balancing strategy, while other models experienced significant performance degradation under high loads.

Finally, to further verify the scalability and efficiency advantages of the proposed model in a larger network, this paper built an embedded test environment at three node scales and introduced a virtual network request flow that simulated the workload of a real data center, referring to the business characteristics of platforms such as Facebook and Alibaba. The experiment conducted multiple rounds of tests on CSO-MAVNE and three comparison models under the same resource configuration and request scale, and the results are shown in Table 6.

As shown in Table 6, CSO-MAVNE showed higher request throughput and lower embedding running time at all node scales, reflecting good scalability and computing efficiency. The results of the independent sample t-test on the throughput difference between CSO-MAVNE and other models showed that at all node scales, CSO-MAVNE reached a significant level of $P< 0.05$ compared with other models, further proving the significant advantages of the proposed model in scalability and processing efficiency.

4 Discussion

Compared with the traffic flow prediction model combining GCN and LSTM proposed by Liu Q et al. in the literature [12], this method mainly focused on time series data modeling under static structure, achieved high prediction accuracy and adaptability through feature fusion, and was suitable for small and medium-sized urban traffic scenarios. When facing the problem of embedding large-scale virtual networks, the CSO-MAVNE model proposed in the study introduced bidirectional link modeling and collaborative multi-agent architecture, which enhanced the model's processing capabilities for link competition, resource coupling, and task parallelism. Performance test results showed that when the number of

VNRs was 300, the acceptance rate of CSO-MAVNE was 82.9%, the benefit-cost ratio was 4.5, the resource utilization rate was 87.2%, and the embedding time was 7.1s.

Compared with the spatiotemporal dynamic data detection method based on GCN and temporal convolutional network proposed by Li S et al. in the literature [13], this study focused on solving the problem of data accuracy degradation in high-frequency dynamic network environments, and achieved a balance between detection accuracy and efficiency through multi-scale modeling. The CSO-MAVNE model in the study focused more on the optimization of resource allocation strategies. Under the guidance of the joint advantage function, the strategy synchronization and global consistency of node and link embedding were achieved. In static scenarios, the embedding success rate of CSO-MAVNE was as high as 90.5%. After the failure of links and nodes, the recovery rate remained above 91.7%. It showed good fault response capability and task continuity guarantee capability.

In summary, the CSO-MAVNE model's performance improvement can be primarily attributed to three key factors: the GCN's ability to embed structural awareness, the GAN-enhanced mechanism for generating feasible topologies, and the optimization of task decoupling execution along with state sharing within the multi-agent collaborative architecture. All the data used in this article were sourced from public simulation networks, did not involve personal privacy or sensitive user information, and posed no ethical or data usage risks.

5 Conclusion

To address the challenges of resource allocation efficiency and complex dynamic topology processing in VNE, a network resource allocation model based on CSO-MAVNE was proposed. In actual application tests, the latency of CSO-MAVNE in three areas was 40ms, 42ms, and 45ms, respectively, and the IRLBR reached 95%, 93%, and 91.5%, respectively. In the dynamic link failure test, the link failure ratios were 5%, 10%, and 15%, respectively, and the FRTs were 0.8s, 1.1s, and 1.5s, respectively, and the ROs were 15%, 18%, and 20%, respectively. In scenarios where the node failure ratios were 2%, 5%, and 8%, respectively, the recovery times were 0.7s, 1.0s, and 1.3s, respectively. Finally, its RDD under high load conditions was only 15.3%, and AL was 1.51s. Experiments results showed that the new method improved the virtual network request acceptance rate, resource utilization, and benefit-cost ratio, providing a reliable basis for optimizing virtual network resource allocation.

Nevertheless, the model may experience communication delays or local strategy convergence problems between multiple agents in ultra-large-scale networks, affecting the overall coordination performance. In addition, due to the extensive use of synthetic topological data during training, there is a potential risk that the model strategy will overfit to a specific structural distribution. In a highly dynamic network environment, the state update of some agents may lag behind the global state change, affecting the real-time performance of embedded decisions. The above problems will be improved in future work by introducing heterogeneous structures, adaptive agent update mechanisms, and expanding real-world data samples.

6 List of nomenclature

CSO-MAVNE: Collaborative Sequential Optimization Multi-Agent Virtual Network Embedding Model
VNE: Virtual Network Embedding
GCN: Graph Convolutional Network
GAN: Generative Adversarial Network
CGCNE: Graph Convolutional Network Embedding Algorithm
SAM: self attention mechanism
MLP: Multi-Layer Perceptron
DRL-VNE: Deep Reinforcement Learning-based VNE
MS-GCN: Multi-Stage GCN
D-GANE: Distributed GAN-based Embedding
RCR: Revenue Cost Ratio
VNR: Virtual Network Requests
IRLBR: Inter-region Load Balancing Rate
FRT: Fault recovery time
RDD: Regional Request Distribution Deviation
AL: Average Latency
SAR: Success Allocation Rate
RTR: Request Throughput Rate
T : Time window
$f+$: Positive directional links
$f-$: Reverse directional links
$T^{f+}(u)$: Total outgoing traffic for node u
M : Average value of outgoing traffic from the node
σ : Standard deviation
$T^{f_v^+}(u)$: Traffic from node u to node v
D_ϕ : Generator
D_ϕ : Discriminator
$r_t(\theta)$: Estimation of the dominance function
A_t : Advantages of Status
$\pi_\theta(a_t|s_t)$: Strategy Probability
$D_\phi(\pi_\theta(a_t|s_t)$: Weighted loss term in the output of the discriminator
τ : Temperature coefficient
P_r : Real data distribution
$R(s_t, a_t)$: Instant rewards for current states and actions
γ : Discount factor
$V(s_t)$: Value function for the current state
Q , K , V : query, key and value
d_k : Dimension of the key vector
s : Current system operations
a_n , a_l : virtual node, virtual link embedding action
$A_{node}(s, a_n)$, $A_{link(s,a_l)}$: Local priority of node, link embedding.
i : The current state of the smartbody i

ϕ : Coder Parameters

$V_\phi(s_t)$: Estimation of the value of the current state

$a_{0:M}^m$: Embedded Action Sequences

A_t^m : Dominance functions

τ' : Control policy update scope

$r_t^m(\theta)$: Ratio estimation of the strategy function

References

[1] Sihan Ma, Haipeng Yao, Tianle Mai, Jingkai Yang, Wenji He, Kaipeng Xue, and Mohsen Guizani. Graph convolutional network aided virtual network embedding for internet of thing. IEEE Transactions on Network Science and Engineering, 10(1):265-274, 2022.https://doi.org/10.1109/TNSE.2022.3207205

[2] Chinmaya Kumar Dehury and Prasan Kumar Sahoo. Failure aware semi-centralized virtual network embedding in cloud computing fat-tree data center networks. IEEE Transactions on Cloud Computing, 10(2):1156-1172, 2022.https://doi.org/10.1109/TCC.2020.2984604

[3] Yang Wang and Qian Hu. A path growing approach to optical virtual network embedding in slice networks. Journal of Lightwave Technology, 39(8):2253-2262, 2021.https://doi.org/10.1109/JLT.2020.3047713

[4] Fujun He and Eiji Oki. Shared protection-based virtual network embedding over elastic optical networks. IEEE Transactions on Network and Service Management, 19(3):2869-2884, 2022.https://doi.org/10.1109/TNSM.2022.3178350

[5] Haipeng Yao, Bo Zhang, Peiying Zhang, Sheng Wu, Chunxiao Jiang, and Song Guo. RDAM: A reinforcement learning based dynamic attribute matrix representation for virtual network embedding. IEEE Transactions on Emerging Topics in Computing, 9(2):901-914, 2021.https://doi.org/10.1109/TETC.2018.2871549

[6] Weizhe Zhang, Desheng Wang, Shui Yu, Hui He, and Yan Wang. Repeatable multi-dimensional virtual network embedding in cloud service platform. IEEE Transactions on Services Computing, 15(6):3499-3512, 2021.https://doi.org/10.1109/TSC.2021.3102016

[7] Nagao Ogino. Resilient virtual network embedding ensuring connectivity under substrate node failures. IEICE Transactions on Communications, 105(5):557-568, 2022.https://doi.org/10.1587/transcom.2021EBP3122

[8] Parinaz Rezaeimoghaddam and Irfan Al-Anbagi. Trust-aware virtual network embedding in wireless sensor networks. IEEE Sensors Journal, 23(6):6326-6337, 2023.https://doi.org/10.1109/JSEN.2023.3240386

[9] Mario Minardi, Thang X. Vu, Lei Lei, Christos Politis, and Symeon Chatzinotas. Virtual network embedding for NGSO systems: Algorithmic solution and SDN-testbed validation. IEEE Transactions on Network and Service Management, 20(3):3523-3535, 2022.https://doi.org/10.1109/TNSM.2022.3225748

[10] Luca Pasa, Nicolò Navarin, and Alessandro Sperduti. Polynomial-based graph convolutional neural networks for graph classification. Machine Learning, 111(4):1205-1237, 2022.https://doi.org/10.1007/s10994-021-06098-0

[11] Vandana Bhattacharjee, Raj Sahu, and Amit Dutta. Enhanced graph representations for graph convolutional network models. Multimedia Tools and Applications, 82(7):9649-9666, 2023.https://doi.org/10.1007/s11042-021-11843-7

[12] Quanzhi Liu, Shuang Wu, and Peng Zhang. Statistical analysis of urban traffic flow using deep learning. Informatica, 48(5):23-28, 2024.https://doi.org/10.31449/inf.v48i5.5393

[13] Sheng Li, Mingguang Duan, and Xiaodan Zhou. Dynamic detection method for spatiotemporal data based on hybrid model and singular spectrum analysis. Informatica, 49(12):35-48, 2025.https://doi.org/10.31449/inf.v49i12.7578

[14] Peiying Zhang, Chao Wang, Chunxiao Jiang, Neeraj Kumar, and Qinghua Lu. Resource management and security scheme of ICPSs and IoT based on VNE algorithm. IEEE Internet of Things Journal, 9(22):22071-22080, 2021.https://doi.org/10.1109/JIOT.2021.3068158

[15] Jin Cheng, Yulei Wu, Yeming Lin, Yuepeng E., Fan Tang, and Jingguo Ge. VNE-HRL: A proactive virtual network embedding algorithm based on hierarchical reinforcement learning. IEEE Transactions on Network and Service Management, 18(4):4075-4087, 2021.https://doi.org/10.1109/TNSM.2021.3120297

[16] Huilin Zhou, Huimin Zheng, Qiegen Liu, Jian Liu, and Yuhao Wang. Linear electromagnetic inverse scattering via generative adversarial networks. International Journal of Microwave and Wireless Technologies, 14(9):1168-1176, 2022.https://doi.org/10.1017/S1759078721001331

[17] Peiying Zhang, Chao Wang, Neeraj Kumar, Weishan Zhang, and Lei Liu. Dynamic virtual network embedding algorithm based on graph convolution neural network and reinforcement learning. IEEE Internet of Things Journal, 9(12):9389-9398, 2021.https://doi.org/10.1109/JIOT.2021.3095094

[18] Peiying Zhang, Enqi Wang, Zhihu Luo, Yanxian Bi, Kai Liu, and Jian Wang. Energy-efficient virtual network embedding: A deep reinforcement learning approach based on graph convolutional networks. Electronics, 13(10):1918, 2024.https://doi.org/10.3390/electronics13101918

[19] Stelios Prekas, Panagiotis Karkazis, Vasileios Nikolakakis, and Panagiotis Trakadas. Comprehensive comparison of VNE solutions based on different coordination approaches. InTelecom, 2(4):390-412, 2021.https://doi.org/10.3390/telecom2040023

DeepRace: GRU-Based Sequence Modeling Framework for Marathon Performance Prediction

Suiping Zhou
School of Sports and Health, Chongqing College of International Business and Economics, Hechuan, Chongqing, 401520, China
E-mail: 19111326556@163.com

Keywords: performance prediction, deep learning models, GRU, RNN, LSTM, race time forecasting, training data analysis, model selection algorithm

Received: April 7, 2025

In recent years, data-driven approaches have gained prominence in sports analytics, offering new opportunities for optimizing athletic performance. This study presents DeepRace, a computational framework based on deep learning models including Recurrent Neural Networks (RNN), Long Short-Term Memory networks (LSTM), and Gated Recurrent Units (GRU) for marathon performance prediction. Using a dataset of 23,763 training activities from 549 athletes who participated in the 2017 Boston Marathon, we evaluated the effectiveness of sequence-based models in forecasting marathon finish times. The framework systematically preprocesses and analyzes sequential workout data, employing a multi-model selection algorithm to identify the most accurate predictive model. Experimental results demonstrate that the GRU model achieved superior performance, with a Mean Absolute Percentage Error (MAPE) of 5.2 percent and a Root Mean Squared Error (RMSE) of 0.11 hours, outperforming both RNN (MAPE 5.6 %) and LSTM (MAPE 6.5 percent) models. Notably, all deep learning models significantly surpassed the baseline linear regression model (MAPE 10%), confirming the advantage of deep learning for time-series performance prediction. These findings underscore the potential of GRU-based sequence modeling for enhancing marathon training strategies, offering actionable insights for athletes and coaches aiming to improve race outcomes through data-informed decisions.

Povzetek: Avtorji predstavijo DeepRace, GRU-temeljen napovedni model za maratonski čas, ki na osnovi treningov natančneje napove rezultate tekačev kot RNN, LSTM ali linearna regresija.

1 Introduction

Over the last decade, endurance sports have increased significantly and drawn individuals from around the globe. The finishing rate in ultra-long running competitions has risen by over 40%, while the popularity and competitiveness of these events continue to grow [1]. This, therefore, calls for the need to prepare athletes for them to execute their best during national and international tournaments. Good planning of workouts and training regimes is an important part of effective preparation; it is considered crucial to enhancing performance. Yet, there are limited well-structured studies or consistent research into evaluating performance in athletes before competitions. Even with advancements in technology and data analytics, few applications can be found for such technology in predicting and analyzing athletic performance, especially for endurance sports.

Various physiological and environmental factors come into play in defining the performance of track athletes in any race. Complications such as dehydration, cramped muscles, overexertion, or even pacing are very critical. For instance, an excessively fast start as compared to your training preparation results in an early onset of fatigue, inappropriate hydration strategies result in reduced endurance capabilities. These will require an in-depth cause knowledge and also how they interact. Precise performance predictions could enable athletes to optimize their training routines, refine strategies, and improve race outcomes. This gap motivated us to explore predictive models that can effectively assess and forecast the performance of athletes with precision.

This research will try to find answers to the following important questions: Can deep learning models give more accurate results for the performance prediction of athletes in competition? What are the most influential parameters and metrics for athletic outcomes? Which practices in workouts yield the best results regarding performance optimization? Guided by these questions, this paper outlines a research study that proposes the development of a predictive framework that is capable of analyzing training regimens to make informed decisions to improve the competitive readiness of athletes. To address this, we present a model designed to forecast athlete performance in preparation for significant competitions. Indeed, accurate predictions of performance will have a great impact not only on competition outcomes but also on guiding athletes to make informed adjustments to training and lifestyle. This work will explore several deep learning methods, including including Recurrent Neural Networks (RNN) [2], Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRU), in determining the best approach for the work at hand. Though RNNs are good to go for processing data in a sequence and considering

temporal dependencies, they provide a good basis for time-series analysis, but their capability to handle long-term dependencies is restricted because of issues with vanishing gradients [3]. LSTMs overcome this issue by incorporating memory cells into their architecture that can retain information across long sequences, performing exemplary in the task of prediction and classification on sequential data [4]. GRUs, on other hand, are similar but more lightweight versions of the LSTMs, providing advantages of LSTMs with reduced computational complexity, hence faster to train.

Table 1 Overview of literature focused on athlete marathon race time prediction

Reference	Year	Race Classification	Forecast Objective	Number of Participants	Variables Examined
[14]	1983	Long-Distance Run	Timing Forecast for Races	19	Velocity calculated at the anaerobic boundary
[15]	2012	Ultra-Distance Run	Aggregate Distance Covered over a One day	64	Longest workout durations, total distance run, and individual best marathon results
[16]	2011	Half Marathon	Estimating Race Times	84	Age, body measurements (e.g., BMI, limb circumference), and experience as an active runner
[17]	2014	Half Marathon	Predicting Race Times	230	Running speed, body composition (e.g., skinfold thickness, BMI, body fat percentage)
[18]	2013	Full & Half Marathon	Statistical Analysis (No Prediction)	147 & 126	Age, body size (e.g., height, BMI, limb circumference), weekly running distance, and total training hours
[19]	2017	Half Marathon	Evaluating Race Performance	78	VO2 Max, speed at anaerobic threshold, step rate, step length, peak speed, flight/contact times
[20]	2019	Full Marathon	Assessing Race Performance	16	Aerobic and anaerobic thresholds, maximal oxygen capacity, and running effectiveness

2 Literature review

This section revisits various methods that have been proposed in the literature for predicting athlete performance. Although few studies on this topic exist, there have been several related efforts with respect to race time prediction using historical training activities and the physical characteristics of athletes.

The work by Vickers et al. [7] employed the Riegel formula for race time estimation up to the half-marathon distance. Reasonable performance was demonstrated. These models, however, underestimated the race times by 10 minutes for 50% of the dataset used for building these models that had been collected through an online survey. The variables studied were age, sex, body mass index, training programs, and mean race velocity. A second model was also analyzed that focused on 22 runners running 46 races and doing an 8-week training program. These correlations, using the mathematical techniques of correlation, between the weekly training distances, average training pace, and race performance were considerable as stated in this study [8].

Further research has utilized a multivariate analytical approach to test interactions that existed between athlete characteristics and training parameters [9]. For instance, body mass, height, BMI, and circumferences of limbs were combined with the training variables of speed and volume. The results showed that the lower calf circumference and higher running speed at training were the best predictors of race performance improvement. Besides, a seventh-degree polynomial regression model was used to predict the sprint times of Olympic athletes based on more than a century of historical data. This model was most successful for short races [10].

Prominent international marathons, like the Boston and London marathons, serve as valuable data sources for research on predictions of performance. For example, one of the studies used data from the 2015 versions of these marathons, considering variables such as distance, time,

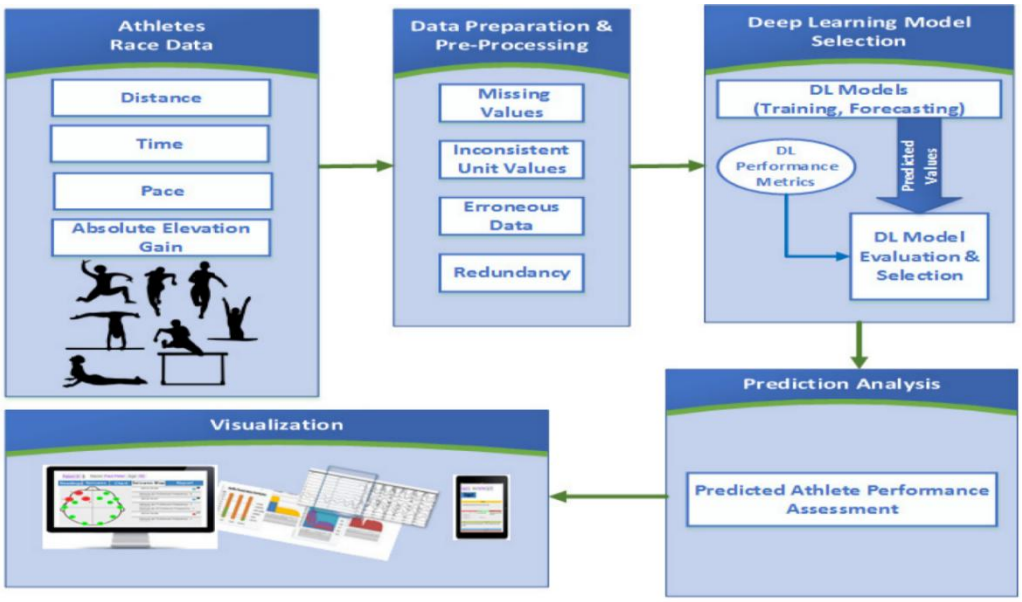


Figure 1: Optimized deep learning framework for predicting athlete performance.

net elevation gain, and workout pace [11]. The outcome indicated that the ensemble learning methods, such as bagging algorithms, were superior to linear regression in predicting the race times of amateur runners. Another study [12], on the other hand, utilized data from Strava, including metrics such as pace, heart rate, and cadence. With the help of the automated feature extraction tool tsfresh, personalized race strategies are developed for each athlete to open new avenues towards improvement by data-driven methods.

A summary of studies focused on marathon race time predictions is provided in Table 1, which outlines dataset sizes, predictive variables, marathon types, and target metrics. A common limitation across these studies is the relatively small dataset sizes, which constrain the accuracy and generalizability of the models. Furthermore, many studies rely on physiological variables such as Oxygen levels and pulse rate, which are challenging to monitor reliably during training sessions. Although various methodologies have been explored, few works have utilized machine learning techniques for race time prediction. Specifically, one study undertook a case-based reasoning approach to estimate race times [13]. However, no work reviewed has utilized deep learning models for race time prediction. Deep learning holds considerable promise for improving prediction accuracy and managing big datasets with ease. This again indicates that more studies are required to be done on deep learning-based approaches to give more accurate and practical tools for race outcome forecasting to improve the performance of athletes.

2 Methodology

This section elaborates on the proposed deep learning-based algorithm to predict the performance of athletes for future tournaments using their workout data. The main objective of the approach is to contribute to enhancing an athlete's performance through correct forecasting of his/her race time for an upcoming marathon. These predictions can be used to motivate the athletes in maintaining consistency and perfecting their training to realize expected performances. The steps in the approach will be: data collection, data preprocessing, selection of the most appropriate deep learning model, and then the prediction of competition performance using the model. Figure 1 shows the framework that highlights the main modules of the solution. This is supported through the core of two key modules: Multi-Model Selection and Prediction Analysis. Each one interrelates in making predictions valid, as shall be elaborated later in detail.

Algorithm 1 Athlete Race Time Multi-Model Prediction Algorithm

```
Input:
  athleteList: List of athletes with their workout sessions, including details
  such as distance, elapsed time, pace, elevation gain, and calories burned.
  predictionModels: List of candidate deep learning models for prediction.
Output:
  PredictedT: Predicted race finishing time.
  selectedModel: Selected deep learning model.
procedure AthleteRaceTimeMultiModelPrediction(athleteList, predictionModels)
    minMAPE ← ∞
    for each m ∈ predictionModels do
        for each a ∈ athleteList do
            W ← getWorkoutList(a)
            for each w ∈ W do
                t̂ ← Predict(w, m)
                MAPE ← evalMAPE(t̂, w)
            end for
        end for
        if MAPE < minMAPE then
            minMAPE ← MAPE
            PredictedT ← t̂
            selectedModel ← m
        end if
    end for
    return PredictedT, selectedModel
end procedure
```

A) Model Description

The proposed framework identifies the important workout activities and training metrics that are bound to

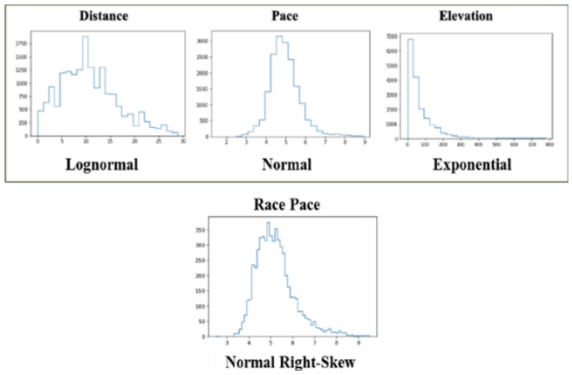


Figure 2: Distribution of training activity data and marathon race paces, showcasing the lognormal distribution of distance, the normal distribution of pace, the exponential distribution of elevation, and the slightly right-skewed normal distribution of marathon race paces.

have the highest impact on optimizing athlete performance. Key metrics are then selected with precision: distance, duration, pace, and elevation gain serve as indicators for training effectiveness. The advanced prediction models like RNN, GRU, and LSTM are applied to the prepared athlete data after the collection and preprocessing of data.

This step is necessary to find out the best and most interpretable model among all for marathon performance forecasting. Deep learning model selection component executes the model selection process by rigourously testing different deep learning techniques based on their ability to predict race times. This assessment is done in the module DL models; hence, the model with the greatest predictive accuracy is selected for further analysis. This module of DL model evaluation and selection will make sure that the selected model is reliable and accurate.

Another key aspect of the framework is the analysis of the predictions of the chosen model. This prediction analysis gives insight into the expected performance of the athlete and provides a good basis for making relevant decisions toward improvement in the future. The process of model selection is detailed in the next subsection.

B) DL Selection Evaluation

Here in this section, the evaluation strategy for deep learning model selection algorithm performance is considered for the estimation of marathon finish time. Here, different deep learning models will be trained on the data set and cross-checked with test data to make the prediction values obtained reliable. Different models explored in the current study based on deep learning incorporate RNN, LSTM, and GRU. From these, automatically, the selection of the most suitable deep learning model will be made. The selected models are designed to handle time-series data, which is the performance of the athletes during training. The selected models will be used to forecast the finishing times for the future. Then, the algorithm will evaluate the various predictions using some important performance metrics, including Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). These metrics give a perfect insight into the accuracy of the model. The formula for these metrics is as follows:

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n} (y_{pt} - y_t)^2}{n}} \quad (1)$$

$$MAE = \frac{\sum_{t=1}^{n} |y_{pt} - y_t|}{n} \quad (2)$$

$$MAPE = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{y_{pt} - y_t}{y_t} \right| \quad (3)$$

Here, y_{pt} shows the predicted values and y_t is the actual value and while n is total number of entries in the data set, Here MAPE for RNN, LSTM and GRU are calculated which is to be used to find the best Deep learning model, for marathon race time. Finally, selected Deep learning model has tested against the best times to complete race. This is summarized in the evaluation process outlined in Algorithm 1. It accepts the athletes' workout data and the potential deep learning models as inputs, producing predictions for race completion times as the output. In such a way, the algorithm compares different models using MAPE for their predictions and chooses the model that reaches the highest accuracy.

3 Experiments and evaluation

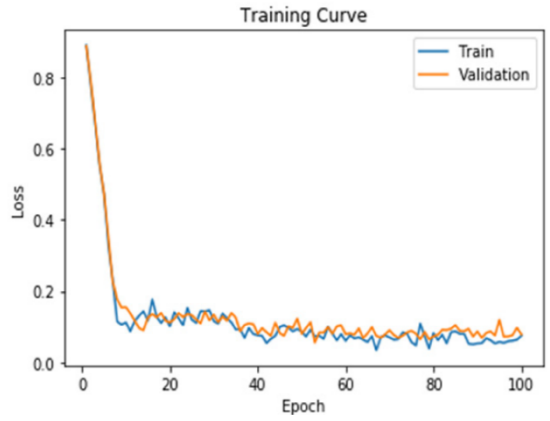
This section outlines the experimental evaluation conducted to evaluate the effectiveness of our prediction models. It begins with an overview of the experimental setup, followed by details on the dataset collection process, and then explains the procedures for data preparation and preprocessing. Lastly, the results are discussed along with a thorough analysis. The features of the data are shown in Table 2.

A) Experimentation Setup

To implement the three Deep Learning models, we selected PyTorch, an open-source, comprehensive framework for machine learning [5][14]. Additionally, we utilized Jupyter Notebook, an open-source web-based application that enables users to develop documents integrating executable code, mathematical equations, graphical representations, and descriptive text. Besides, as a baseline for comparisons, we consider a standard linear regression machine learning model.

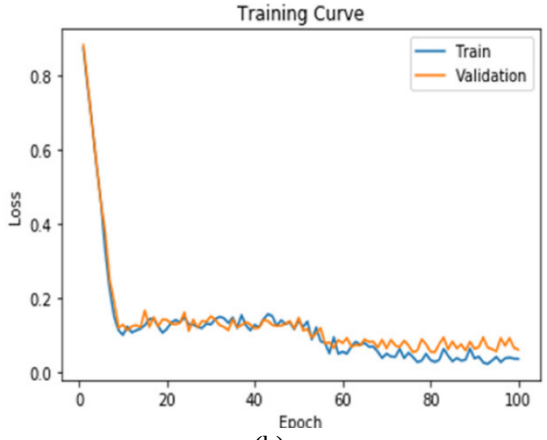
B) Dataset

Data on running a marathon were chosen for the experimental verification of the proposed model. Running is very often described as one of the most available sports for a person of average physical condition because it perfectly fits into a hectic schedule and can be performed virtually anywhere and anytime Thus, running as a kind of sport has become very popular all over the world. The



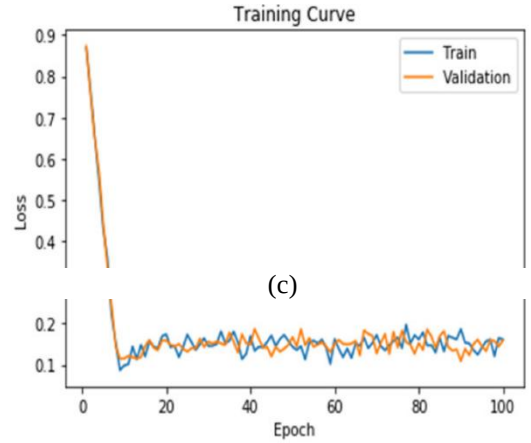
(a)

dataset used in this work is on training sessions from professional athletes who were set to participate during the 2017 Boston Marathon, observed over their five-month



(b)

preparation phase [21]. Known for its prestige and global recognition, the Boston Marathon stands as one of the most celebrated and widely attended marathons in the world.; every year, it gathers thousands of participants. Thus, any dataset from this marathon is quite sizeable,



(c)

Figure 3: Training curves of prediction error (MAPE) for RNN, GRU, and LSTM models with 50 hidden states. The subplots show RNN stabilizing at 0.10 MAPE, GRU experiencing overfitting after epoch 55, and LSTM stabilizing around 0.15 MAPE.

adding to the reliability and accuracy of our results. Each of the records collected contains an array of features such as the name of the athlete, session date, running length in meters, duration of run, pace in minutes per kilometer, average pace of workout, and elevation, which means the sum of all gains in elevation throughout the course of the activity. The data is from 549 athletes of the April 2017 Boston Marathon and 23,763 training activities these athletes have done in five months prior to the marathon. Figure 2 presents the distribution of activities data and race paces. From this, the distribution of distance, pace, and elevation can be identified following lognormal, Gaussian, and exponential patterns, respectively. Moreover, the marathon race pace exhibits a normal distribution with a mild rightward skew.

C) Data Preparation and Pre-Processing

The data used in this work had to be prepared and preprocessed before feeding the machine learning models. To this end, we employed the Talend Data Preparation tool [20] to import files with information on athletes via its user-friendly interface. It provides several functionalities: parsing of files, giving recommendations on data cleansing, among others. Besides, it gives the capability of plotting charts and metrics that allow visual data quality assessment. We also used Trifacta Wrangler (version 3.0.1-client1+push2) for the preparation, preprocessing, and assessment of data quality metrics [22]. In addition to the mentioned tools, during the whole process of data cleaning, different Python libraries (datetime, pandas) were used. Among them are the datetime library for time format data processing and pandas for cleaning and normalization. These tools and libraries put together allowed for fast and qualitative preparation of the data.

D) Results and Discussion

This section is devoted to performance testing of the proposed model, where each algorithm of prediction accuracy is tested, so the best method is selected within the model.

We run RNNs, LSTMs, and GRUs models by considering 25 and 50 hidden states. We compare these two results, which have almost the same graph; the 50 hidden state configured model demonstrates a slight difference in terms of accuracy. We further optimized this and present results when the deep learning models were trained on a pre-processed data set cleaned by the three-sigma rule of statistical quality control [9]. In this, only values within $\pm 3\sigma$ from the mean were kept. We were able to retain 93.3% of the dataset from this cleaning. This cleaning had a positive effect on the models' performance reflected by the MAPE values listed for each model in Table II. Thus, according to the performance based on MAPE values and considering other benchmark models in this dataset, GRU with 50 hidden states had selected the model selection process as the best-performing algorithm.

This made the selection module finally select the most optimal model to estimate finishing time: GRU.

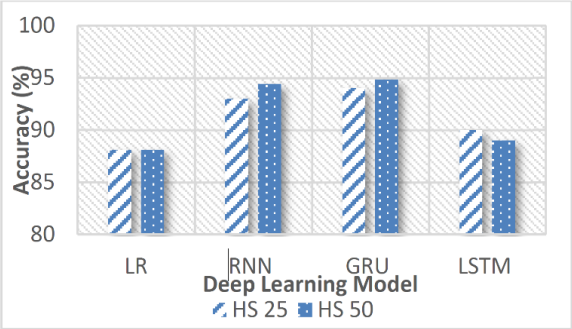


Figure 4: Accuracy comparison of deep learning models, showing GRU (95%) outperforming RNN (94%) and LSTM (89%) after outlier removal. All models exceed the baseline linear regression.

The training plots in Figure 3 give an idea of the learning behavior of the deep learning models being investigated. The RNN model (Figure 3a) exhibits a steep decline in MAPE during the initial epochs, indicating its ability to learn short-term dependencies. It, however, plateaus very quickly, at around epoch 15, which means that it gets stuck in a local optimum. This restriction is in line with the recognized difficulty of vanishing gradients for RNNs, making it difficult for them to efficiently learn long-term dependencies. The GRU model (Figure 3b), however, demonstrates a more definite and prolonged decrease in error and continues to perform well through epoch 50. Its design, incorporating reset and update gates, enables it to manage sequential dependencies more effectively than the RNN. Interestingly, beyond epoch 55, the GRU model starts overfitting, with the training loss still decreasing but the validation loss plateauing, showing the typical overfitting behavior for powerful deep learning models when the data is moderately sized. The LSTM model (Figure 3c) takes longer to converge and plateaus at a higher MAPE of around 0.15. Although LSTM is built to allow longer-term dependencies than RNN, its more complicated architecture might not be fully utilized in this study given the moderate-sized dataset and can lead to slower learning and worse performance compared to that of GRUAdditionally, deep learning models have been tested based on performance concerning the MAPE metric. From Figure 4, after removing the outliers, the highest accuracy stands at 95%, obtained by the GRU model with 50 hidden states compared to 94% and 89% obtained by RNN and LSTM models respectively. Besides this, all these deep learning models outperform the baseline LR model by a high margin, thus again proving how effective deep learning models are when applied for the task of predicting marathon finish times.

The results in Table 2 and Figure 4 reinforce the superiority of the GRU model, which achieved the lowest MAPE (5.2 percent) and the highest accuracy (95 percent) after outlier removal, compared to the RNN (5.6 percent MAPE, 94 percent accuracy) and LSTM (6.5 percent MAPE, 89 percent accuracy). The improved performance of GRU can be attributed to its efficient gating mechanism, which allows it to maintain crucial information over longer sequences while avoiding unnecessary complexity. Unlike the RNN, which lacks gates and is prone to issues such as vanishing gradients, the GRU's gates enable it to control the flow of information effectively. Compared to LSTM, which has an even more complex three-gate structure, GRU achieves a balance of simplicity and expressiveness, making it less prone to overfitting and faster to train. This architectural efficiency is particularly beneficial when working with datasets of moderate size, such as the one used in this study, where GRU's design allows it to capture both short- and long-term dependencies with minimal computational overhead.

Table 2*:* Comparison of MAPE values for different models (Linear Regression, RNN, GRU, and LSTM) with 25 and 50 hidden states, evaluated on datasets with and without outliers.

Performance Metric	Linear Regression (LR)	RNN	GRU	LSTM
Dataset Without Outliers - 25 Hidden States	0.10	0.07	0.06	0.10
Dataset Without Outliers - 50 Hidden States	0.10	0.056	0.052	0.11
Full Dataset - 25 Hidden States	0.11	0.11	0.089	0.122
Full Dataset - 50 Hidden States	0.11	0.12	0.10	0.104

The baseline linear regression model, however, performed a much worse MAPE of approximately 10 percent, indicating its built-in deficiency in this regard. Linear regression has the assumption of static linear dependency between input features and output, failing to recognize the dynamic, sequential nature of training data. Marathon performance is not only a function of stand-alone variables such as pace or distance but also of the collective impact of training over time, where adaptation, recovery, and fatigue interact with each other in intricate ways. The linear regression model does not have the ability to identify and take advantage of these temporal patterns, which is why its relatively poorer predictive power. It is this realization that highlights the overriding importance of using sequence-based deep learning models

like GRU, which are specifically designed to deal with the temporal aspect of workout information and provide more accurate predictions of athletic performance.

4 Conclusion

Enhancing the performance of the athletes to higher levels, both nationally and internationally, promotes societal cohesion in society and retains an identity in global status. This may be done through an analysis of current performances, investigating work-out habits, and then projecting forward to higher performances in sports competitions. A holistic process of modeling and analyses would give detailed specific training required that leads to the standard of the performance being maximized.

This paper describes deep learning models in forecasting athlete performance in competitive eventsand tournaments. On the other hand, the accuracy of a prediction can works as a good motivator toward continuing and increasing training to meet target times and increase the likelihood of achieving success. In addition, it allows showing it to the athletes, which may influence their approach in training, thus enabling them to work out their weaknesses and improve the general performance. Moreover, this model will create great value in monitoring the progress of athletes and giving them exact training to help them in their preparation process and incremental improvement in competitive performance. This research makes two major contributions: investigating various Deep Learning models to find the best among them for the prediction of performance of an athlete, and proposing a Deep Learning model selection algorithm that will ensure the accuracy of the performance prediction for competitions. The proposed framework allows the evaluation of a number of Deep Learning models, showing that these models outperform traditional methods in predicting race times. Among these, the GRU model showed the highest accuracy, outperforming RNN, LSTM, and standard linear regression models.

One limitation of this study is the small size of the training dataset; although encouraging results are achieved, they limit applicability to larger scales. A future study could be presented with enhanced studies using larger datasets of actual data on finishers' running patterns from famous marathons in Tokyo, London, and Berlin. Furthermore, customized training schedules, novelty in injury prevention approaches, and a detailed recovery strategy may provide further extensions to the framework. Advanced ensemble methods include fine-tuning Random Forest with Deep Learning to greatly improve prediction accuracy and thus better results, which could be a subject of further research.

References

[1] Marathon Statistics 2019 Worldwide (Research): 2020. *https://runrepeat.com/research-marathon-performance-across-nations*. Accessed: 2024-04-20.

[2] Flores, A. and Tito-Chura, H., 2024. Multi-step forecasting of Guillain Barré cases using deep learning. Informatica, 48(20). DOI:10.31449/inf.v48i20.6358

[3] Li, X. and Hua, J., 2024. Basketball Fixed-point Shot Hit Prediction Based on Human Pose Estimation Algorithm. *Informatica*, *48*(8). DOI: 10.31449/inf.v48i8.5781

[4] Patterson, J. and Gibson, A., 2017. *Deep learning: A practitioner's approach*. " O'Reilly Media, Inc.".

[5] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*. DOI: 10.48550/arXiv.1406.1078

[6] Jin, M., Koh, H.Y., Wen, Q., Zambon, D., Alippi, C., Webb, G.I., King, I. and Pan, S., 2024. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 46, no. 12, pp. 10466-10485, Dec. 2024*, DOI*:* 10.1109/TPAMI.2024.3443141.

[7] Vickers, A.J. and Vertosick, E.A., 2016. An empirical study of race times in recreational endurance runners. *BMC Sports Science, Medicine and Rehabilitation*, *8*, pp.1-9. DOI: 10.1186/s13102-016-0039-8

[8] Doherty, C., Keogh, A., Davenport, J., Lawlor, A., Smyth, B. and Caulfield, B., 2020. An evaluation of the training determinants of marathon performance: A meta-analysis with meta-regression. *Journal of science and medicine in sport*, *23*(2), pp.182-188. DOI: 10.1016/j.jsams.2019.09.005

[9] Nikolaidis, P.T., Rosemann, T. and Knechtle, B., 2018. A brief review of personality in marathon runners: the role of sex, age and performance level. *Sports*, *6*(3), p.99.d DOI: 10.3390/sports6030099

[10] Efiong, J.E., Olajubu, E.A. and Felix, O.A., 2019. Formulation of sprint time predictive model for Olympic Athletic Games. *International Journal of Information Technology and Computer Science*, *11*(4), pp.33-43.

[11] Ruiz-Mayo, D., Pulido, E. and Martınoz, G., 2016. Marathon performance prediction of amateur runners based on training session data. *Proc Mach Learn Data Min Sports Anal*, *8*. DOI: 10.1055/a-1993-2371

[12] Berndsen, J., Smyth, B. and Lawlor, A., 2019, September. Pace my race: recommendations for marathon running. In *Proceedings of the 13th ACM*

conference on recommender systems (pp. 246-250). DOI: 10.1145/3298689.3347018

[13] Using Artificial Intelligence to Run your Best Marathon: 2017. https://medium.com/running-with-data/using-artificial-intelligence-to-runyour-best-marathon-97fcea09ba44 . Accessed: 2024-04-20.

[14] Gómez-Molina, J., Ogueta-Alday, A., Camara, J., Stickley, C., Rodríguez-Marroyo, J.A. and García-López, J., 2017. Predictive variables of half-marathon performance for male runners. *Journal of sports science & medicine*, *16*(2), p.187. DOI: 10.52082/jssm.2017.187

[15] Esteve-Lanao, J., Del Rosso, S., Larumbe-Zabala, E., Cardona, C., Alcocer-Gamboa, A. and Boullosa, D.A., 2021. Predicting recreational runners' marathon performance time during their training preparation. *The Journal of Strength & Conditioning Research*, *35*(11), pp.3218-3224. DOI: 10.1519/JSC.0000000000003269

[16] Knechtle, B., Barandun, U., Knechtle, P., Zingg, M.A., Rosemann, T. and Rüst, C.A., 2014. Prediction of half-marathon race time in recreational female and male runners. *Springerplus*, *3*, pp.1-8. DOI: 10.1186/2193-1801-3-248

[17] Knechtle, B., Knechtle, P., Rosemann, T. and Lepers, R., 2011. Personal best marathon time and longest training run, not anthropometry, predict performance in recreational 24-hour ultrarunners. *The Journal of Strength & Conditioning Research*, *25*(8), pp.2212-2218. DOI: 10.1519/JSC.0b013e3181e5f6f2

[18] Rhodes, E.C. and McKenzie, D.C., 1984. Predicting marathon time from anaerobic threshold measurements. *The physician and sportsmedicine*, *12*(1), pp.95-98.

[19] Zillmann, T., Knechtle, B., Rüst, C.A., Knechtle, P., Rosemann, T. and Lepers, R., 2013. Comparison of training and anthropometric characteristics between recreational male half-marathoners and marathoners. *Chin J Physiol*, *56*(3), pp.138-146.DOI: 10.4077/CJP. 2013.BAB105

[20] Rüst, C.A., Knechtle, B., Knechtle, P., Barandun, U., Lepers, R. and Rosemann, T., 2011. Predictor variables for a half marathon race time in recreational male runners. *Open access journal of sports medicine*, pp.113-119. DOI: 10.2147/OAJSM.S23027

[21] Trifacta Wrangler: https://www.trifacta.com/start-wrangling/. Accessed: 2024- 04-20.

EBSVM: A Sparsification Method From LP-Based SVM via Stochastic Functional ξ-Analysis

Rezaul Karim*, Zakir Hossain, Amit Kumar Kundu, Ali Ahmed Ave
Department of Electrical and Electronic Engineering, Uttara University, Dhaka, Bangladesh
E-mail: rezkar@uttarauniversity.edu.bd
*Corresponding author

Keywords: Classifier, stochastic, LP, QP, sparse, SVM, EBSVM (efficiency boosted SVM), accuracy, MAC, GFR, compact, data, mapping

Received: March 28, 2025

While SVM being a solid mathematical and state-of-the-art sparse but powerful stochastic machine with top accuracy, Vapnik's LP based SVM is sparser than the QP generated one though offers similar accuracy. Yet, further sparsity is essential to efficiently work with very complicated and large datasets. In our work, we propose a novel efficient method, EBSVM by imposing additional sparsification on this sparser LPSVM maintaining optimal complexity of the machine to further reduce classification cost while preserving similar accuracy by analyzing the stochastic auxiliary variable, ξ after a compact deterministic mapping of data. Experiments with Gunnar Rätsch benchmark data as well as synthetic data reveals that our model demands classification cost much smaller than the cutting-edge prominent classifiers while posing similar accuracy despite of being simpler to be implemented. On average, it costs 12.4% of the standard QPSVM, 50.3% of the VLPSVM, 68.1% of the EESVM [1], 98.1% of the greedy SpSVM-2 [2]. It insists the least MAC (Machine Accuracy Cost) value as well as GFR(Generalization Failure Rate) considering these machines. Numerical calculus based noise-sensitivity analysis proves our model's potentiality in working with noisy data. This machine is also expected to work with high efficiency in case of very large and complex data satisfying constraints in real-time.

Povzetek: Članek predstavi EBSVM za učinkovitejšo klasifikacijo s pomočjo stohastične ξ-analize in redčenja LP-SVM, ki odstranita zavajajoče KCV ter optimizirata pristranskost, omogočata bolj kvalitetno posploševanje kot metoda QPSVM, z hitrejšimi odločitvami klasifikacije.

1 Introduction & selected related work

In stochastic learning, the principal objective in a probabilistic data-classification problem is in general to model a discriminator function that effectively represents the generalized relationship of the input objects or patterns with their matching class-labels applying the given dataset having some example patterns with their associated class-labels. While in such indeterministic frameworks, performance from kernel-based classifiers is influenced by the size, randomness of distribution and topological variations of the dataset, the complexity of constructing such classifiers and determining its decision boundary by involving kernel operating patterns is also related to the probabilistic nature of data, classifiers' geometry and available computational resources. Thus different machines show different performances on data detection demanding different computational cost. In this regard while Support Vector Machines (SVMs) lead the field of data classification by offering top level accuracy through finding global optima, a key stochastic property of SVMs is their sparsity, which enables the construction of the optimal classifier using only a probabilistically selected subset of training patterns, known as support vectors (SVs). This sparsity reduces computational demands but as the number of SVs increases, computational cost from nonlinear SVM classifiers also gets higher. However, although Vapnik's Linear Programming (LP) based SVM is even sparser and demands much less computational cost, it also demands the expensive kernel execution task. Moreover, as science and technology have advanced, the size as well as the complexity of data have grown rapidly, and as a result, the number of support vectors usually rises with the growth of the size of training dataset [3]. This makes the further but compact sparsity of the support vector set a critical and continuous consideration for large-scale datasets, especially in scenarios constrained by computational resources and time. As a result, developing methods to address this challenge has become a central focus of research, aiming optimal performance by balancing stochastic generalization with computational efficiency.

Tom Downs et al. [4] offers a method to detect linearly dependent support vectors (SVs), retaining only one and discarding the rest but the number of this linearly dependent SVs tends to decrease as data dimensionality and complexity increase. Joachims-Yu [5], Keerthi et al. [2], and Cot-

ter et al. [6] develop iterative algorithms for reduced SVMs that deliver impressive results, demanding only a very small amount of kernel execution to classify a single pattern whereas [6] reported a memory run out from [2], [5] applying on bulky dataset while their approach introduces a notable deviation from the original methodology with massive parameter choosing. Romdhani et al. [7] and Wu et al. [8] propose to approximate the decision function of a full SVM by identifying a novel denser set of patterns to replace the full set of support vectors whereas Raetsch et al. [9] explore wavelet-based estimations for these vectors to efficiently compute dot products between the kernel generating pattern vectors. These approaches, however, rely heavily on some complex optimization methods that are indeed prone to the starting point, learning rate, etc. Heisele et al. [10] suggests a structured SVM-oriented classification method that uses a series of linear SVMs in the early stages and a nonlinear SVM in the last stage while optimizing thresholds for enhancing both classification time and accuracy whereas Karim et al. [11] presents that a carefully organized cascade of two nonlinear SVMs considerably lowers classification costs compared to using a single nonlinear SVM. Arreola et al. [12] models a linear SVM-based decision tree. Sahbi-Geman [13] designs a tree-structured hierarchical SVM using threshold selection as well as reduced set techniques, applied across partitioned pattern spaces. Moreover, there are also some other noteworthy works focusing on developing cost-efficient SVM variants. Where, Maji et al. [14] presents that histogram and additive kernels based SVMs outperform linear SVMs in speed and accuracy. Ladicky and Torr [15] propose a locally linear SVM with bounded curvature and a smooth decision boundary, balancing the number of anchor points against classifier complexity to prevent overfitting and runtime issues. Xu et al. [16] suggests an additional training algorithm to further compress the trained SVMs using a small set of parameters. Li and Zhang [17] designs a fast object detector using cascaded stages with logistic regression acting as a weak base learner focusing on efficient training method. Raykar et al. [18] develops a cascaded system where the classifiers discard patterns based on probabilities calculated in previous stages achieving a balance between feature learning costs and accuracy. EESVM by Kundu-Karim [1], HESVM by Karimkundu [19] achieve a significant advancement in SVM efficiency, which is particularly suitable for applications involving large-scale and complex datasets though their implementation complexity is also notable. Fu et al. [20] combines linear SVMs to classify complex data showing that the prediction phase efficiency of linear SVMs rivals that of nonlinear SVMs. Karim and Kundu [21] propose a second-order SVM by performing quadratic and linear programming in sequence, achieving compactness and high classification accuracy. Later, they introduce a different method [22] for support vector reduction by LP and computational analysis, demonstrating on benchmark datasets that their model achieves comparable accuracy to standard nonlinear SVMs with significantly improved prediction efficiency. Franc et al. [23] proposed Reduced SVMs (RSVMs) with lower computational costs compared to standard SVMs to classify patterns by leveraging a small subset from the support vectors of the conventional SVM while Gu et al. [24] introduced Clustered SVMs, employing weighted combinations of LSVMs that are trained on the subsets of the training data which are clustered to achieve localized separation. Karim et al. [25] introduces a novel similarity function to enhance the efficiency of Linear Programming based Support Vector Machine. Zhou [26] proposes an approach to enhance the efficiency of SVMs through sparsity constraints that offers significant computational speed while using a practically challenging assumption in the optimization method. Dries et al. [27] offers innovative approaches for support vectors reduction in SVM classifiers through two different methods that gives some computational benefits. However, their modification of training data limits their applicability to non-separable datasets. Wang et al. [28] offers a significant advancement in linear SVM algorithm by tackling fundamental issues with traditional hinge loss but initiates new challenges, specially in terms of optimization and scalability. Liu et al. [29] offers a significant advancement in kernel SVM methodology by incorporating the 0-1 soft margin loss into a nonlinear kernel framework that gives a promising alternative for complex classification tasks while the inherent challenges associated with their non-convex optimization and algorithmic complexity limits the model's practicality and robustness. Wu et al. [30] proposes an innovative approach to SVM classification by excluding the involvement of kernel functions and directly incorporating the 0-1 loss function that gives improved performance. However, on top of the challenges from their non-convex optimization and algorithmic complexity, their assumption that a quadratic hyper-surface is suitable for separating classes, which may not hold true for all datasets, potentially bounds its applicability. Akhtar et al. [31] gains generalization performance by coupling their RoBoSS loss function in SVM while issues about implementation complexity, parameter tuning, scalability, and data quality should be taken into account seriously during its application. Lin et al. [32] gets support vector reduction from modification of conventional SVMs by integrating the ℓ_0-norm into the hinge loss function but the challenges regarding their non-convex optimization, computational complexity should be considered when applying this approach. Wang et al. [33], [34], [35], [36] propose effective mechanisms for handling large-scale classification problems by balancing robustness, sparsity, and efficiency but their non-convex models bring challenges during implementation.

In spite of the efficiency of these classifiers, further sparsification with convenient implementation is very necessary for datasets that are complex and very large.

In this work, we focus to build up an exceptionally sparse classifier within a stochastic framework. By analyzing the SVM learning process through stochastic principles with pattern space spanning and leveraging computational and

probabilistic insights, we propose a simple but very productive learning algorithm under uncertainty. Being generated from the outcome of the analysis of a stochastic variable, this algorithm employs methods to model a highly efficient classifier from a very compact and dense set of stochastic decision points that significantly reduces classification costs while preserving statistical robustness and strong generalization capabilities.

The remaining part of the paper is arranged in the following way: Section 2 covers the relevant foundations while Section 3 details the proposed algorithm with an in-depth analysis. The experimental results are presented in Section 4 whereas Section 5 concludes with a discussion and suggestions for future work.

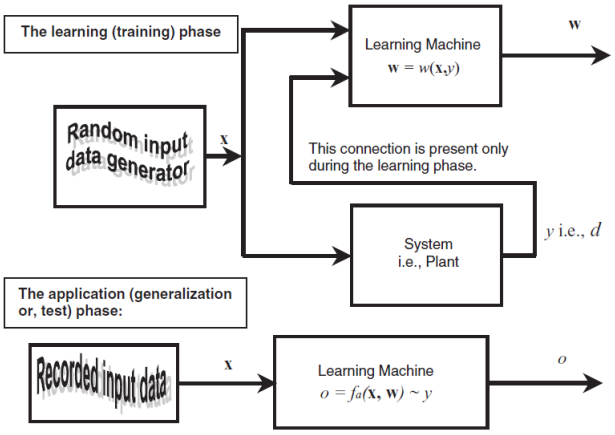


Figure 1: Training and generalization phase of a statistical learning machine model [39]

2 Related basics

2.1 Stochastic learning fundamentals

Stochastic learning lies in the Machine Learning system to train the machine where model parameters are determined under uncertainty using the training data (figure 1). With constant model complexity, for small training sets, the model may get insufficient information to capture underlying patterns, just memorizes the training dataset well rather than learning generalizable patterns, leading to high variance or overfitting and underperform while with larger sample size, the generalization error typically decreases because the model is trained on more representative data, improving its ability to generalize and reducing overfitting. Further, when the model has low dimensionality, it may not have the capacity to fit the training data adequately, resulting in high bias (underfitting) where the generalization error gets highly influenced by the inability to capture data complexity whereas when the model has high dimensionality, it can perfectly fit the training data, including noise, leading to high variance (overfitting) and the generalization error increases as the model fails to perform well on test data. Hansen [37], Karim [38] showed that to avoid higher generalization error in Stochastic Linear Learning, for a problem with sample size N, model dimension d, one must have $N >> d$ and this error goes unbounded for $N \leq d+1$. However, the ideal balance is achieved when the model dimension matches the effective complexity of the data, given a sufficient sample size and the relationship among generalization error, sample size, and model dimension in stochastic learning is governed by the interplay among bias, variance, and the complexity of the model.

Generalization error while modeling a stochastic machine

Problem set up

The key objective in machine learning is to approximate an unknown function $f(x)$ that assigns an output y for an input x while the observed outputs y are noisy due to inherent randomness or measurement errors:

$$y = f(x) + \epsilon$$

where $f(x)$ is the true underlying function, ϵ is the random noise following a zero-mean normal distribution with variance σ_ϵ^2: $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$.

We approximate $f(x)$ using a model $\hat{f}(x)$, which is learned from data, aiming at measuring how well $\hat{f}(x)$ performs by analyzing its expected squared error.

Expected error forming

The expected squared-error at a point x can be calculated as:

$$\mathbb{E}[(y - \hat{f}(x))^2]$$

where the expectation is taken over the randomness in both $\hat{f}(x)$ and y (due to noise ϵ).

Substituting $y = f(x) + \epsilon$ in the above equation we get

$$\mathbb{E}[(y - \hat{f}(x))^2] = \mathbb{E}[(f(x) + \epsilon - \hat{f}(x))^2]$$

Expanding the squared term we get

$$(f(x)+\epsilon-\hat{f}(x))^2 = (f(x)-\hat{f}(x))^2+2(f(x)-\hat{f}(x))\epsilon+\epsilon^2$$

Now taking the expectation and using $\mathbb{E}[\epsilon] = 0$, the middle term becomes $\mathbb{E}[2(f(x) - \hat{f}(x))\epsilon] = 0$ while the last term gives $\mathbb{E}[\epsilon^2] = \sigma_\epsilon^2$, which is the noise variance and the whole expected squared error simplifies to

$$\mathbb{E}[(y - \hat{f}(x))^2] = \mathbb{E}[(f(x) - \hat{f}(x))^2] + \sigma_\epsilon^2$$

Error term decomposition

The term $\mathbb{E}[(f(x) - \hat{f}(x))^2]$ representing the error in approximating $f(x)$ with $\hat{f}(x)$, can be further decomposed

by adding and subtracting $\mathbb{E}[\hat{f}(x)]$ (the expected prediction of the model over all possible datasets) as

$$\mathbb{E}[(f(x)-\hat{f}(x))^2] = \mathbb{E}[(f(x)-\mathbb{E}[\hat{f}(x)]+\mathbb{E}[\hat{f}(x)]-\hat{f}(x))^2]$$

Now, expanding the squared term, we get

$$(f(x)-\mathbb{E}[\hat{f}(x)]+\mathbb{E}[\hat{f}(x)]-\hat{f}(x))^2 = (f(x)-\mathbb{E}[\hat{f}(x)])^2 + 2(f(x)-\mathbb{E}[\hat{f}(x)])(\mathbb{E}[\hat{f}(x)]-\hat{f}(x)) + (\mathbb{E}[\hat{f}(x)]-\hat{f}(x))^2$$

Now applying the expectation, we get the cross-term $\mathbb{E}[2(f(x) - \mathbb{E}[\hat{f}(x)])(\mathbb{E}[\hat{f}(x)] - \hat{f}(x))] = 0$ as $\mathbb{E}[\hat{f}(x)] - \hat{f}(x) = 0$. whereas the first term $(f(x) - \mathbb{E}[\hat{f}(x)])^2$ is the **Bias squared**, the second term $\mathbb{E}[(\mathbb{E}[\hat{f}(x)] - \hat{f}(x))^2]$ is the **Variance**.

Thus, we can write

$$\mathbb{E}[(f(x) - \hat{f}(x))^2] = (\text{Bias})^2 + (\text{Variance}),$$

where, $\text{Bias}^2 = (f(x) - \mathbb{E}[\hat{f}(x)])^2$, $\text{Variance} = \mathbb{E}[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]$.

Eventually, the full expression for the expected squared error becomes

$$\mathbb{E}[(y - \hat{f}(x))^2] = (\text{Bias})^2 + (\text{Variance}) + \sigma_\epsilon^2$$

Generalization introducing input distribution

Considering all inputs x, we have to take the expectation over the input distribution $p(x)$ as

$$\text{Generalization Error} = \mathbb{E}_{x,\epsilon}[(y - \hat{f}(x))^2]$$

Expanding this we can write the final expression as

$$\mathbb{E}_{x,\epsilon}[(y - \hat{f}(x))^2] = \underbrace{\mathbb{E}_x[(f(x) - \mathbb{E}[\hat{f}(x)])^2]}_{\text{Bias}^2} + \underbrace{\mathbb{E}_x[\mathbb{E}_\epsilon[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2]]}_{\text{Variance}} + \underbrace{\sigma_\epsilon^2}_{\text{Irreducible Error}} \quad (1)$$

Intuition behind each term & key observations

- High bias takes place while the model is under-trained or too simple to capture data complexity and as model complexity increases, **Bias** decreases because the model can better approximate $f(x)$. Underfitting from high bias can lead to a situation, where both training and test errors are high.
- **Variance** increases because the model turns to be more responsive to the training data. Variance measures how sensitive the model is to changes in the training data. High variance can result in overfitting, where the model performs well on the training data but poorly on unseen test data.
- **Irreducible error** (σ^2) remains constant, independent of model complexity.
- A model having high bias typically has low variance and vice versa. The optimal model is a properly tuned model that achieves good generalization with reasonable training performance minimizing the total error by balancing bias and variance. This is the **bias-variance tradeoff**, a fundamental concept in machine learning that explains the relationship between a model's ability to generalize to unseen data and its performance on training data. It is very important to consider during the construction of a stochastic machine.

Next we discuss about Support Vector Machine (SVM), which is very prominent managing this bias-variance trade off successfully.

2.2 Support vector machine (SVM)

Due to their robustness, strong generalization capability, and convexity of the mathematical model to find unique global optimum, SVMs probably have gained the highest popularity among machine learning approaches for supervised learning though having very simple principle. The problem that drove its initial development occurs in various forms as the bias variance trade-off [40], capacity control [41] , overfitting [42] while the core idea is the same [43]. Thus, it is mathematically quite strong basing on the concept of statistical learning theory as well as optimization.

2.2.1 SVM with statistical learning theory

Suppose that we have two classes of objects. Now we have received a novel object for which we are to allocate a class from one of these two classes. Such a problem can be considered as a *(binary, due to two classes) pattern recognition* problem that can be formed in the following way: having a set of observational data

$(x_1, y_1), \ldots, (x_m, y_m) \in \mathcal{S} \times \{\pm 1\}$,

we need to design a *decision (discriminator) function* $f : \mathcal{S} \to \{\pm 1\}$, where $\mathcal{S}$ is a set of elements from which the x_i patterns have been drawn, generally referred to as the *domain* whereas y_i are designated as the *targets* or *class-labels*. A well defined decision function is supposed to be able to *generalize* to the unseen novel data instances that poses a possibly low value while measuring the *risk*

$$R[f] = \frac{1}{2} \int |f(x) - y| \, dP(x, y) \quad (2)$$

where $dP(x, y)$ can be written as $p(x, y)dxdy$ when a density $p(x, y)$ exists.

Expressing linguistically, on average across an unspecified distribution P that is considered to generate both of the training and test data, we aim to achieve a low error. Here, we calculate the error using the *binary loss function*

$q(x, y, f(x)) := \frac{1}{2}|f(x) - y|$, which gives 1 when (x, y) is not correctly classified and 0 when it is correctly classified. It is to note that up to now, the patterns can take almost any form , and no assumptions have been made on $\mathcal{S}$ except it is a set equipped by a probability measure P. Additionally, without having an estimate of $P(x, y)$, although (2) represents the true mean error in a nice way, it does not tell about the way either to find a function posing a small risk or even to assess the risk of a specified function, and thus becomes not so useful. Hence,for risk minimization, we need to move towards an induction principle . Now, we incorporate an additional form of structure related to the information we actually have, that is the training data. And while the quantity $R[f]$ is usually referred to as the expected risk, or simply the risk or here, we term it as the actual risk, this newly introduced quantity will be called the empirical risk (the training error) , noted by $R_{\text{emp}}[f]$, which is defined as the calculated mean error rate over the full training set (considering a fixed and finite number of observations); that is,

$$R_{\text{emp}}[f] = \frac{1}{m}\sum_{i=1}^{m}|f(x_i) - y_i| \tag{3}$$

It is to be noted that, here, no probability distribution shows up and $R_{\text{emp}}[f]$ is a fixed number for a specific f and for a specific training data set $\{x_i, y_i\}$.
However, when the number of training examples m is limited, minimizing this empirical risk does not assure small actual risk as small error over the training set does not automatically imply good generalization ability, that is, a small amount of error on any independent and novel test data. This phenomenon is commonly known as overfitting [44]. To get the maximum benefit from a limited amount of data, various statistical methods are being developed over many years. Vapnik-Chervonenkis, through their VC theory [45] in Statistical Learning Theory, emphasis on the necessity of limiting the set of functions from which f is selected to one having a *capacity* that well fits the available training data. It establishes *bounds* on the test error, which depends on the empirical risk as well as the capacity of the function class while minimizing these bounds guides to the *structural risk minimization* principle [46]. Considering the discussed problem, SVMs can be regarded as an application of this principle where it is tried to minimize the "Structural Risk", which is a combination of the *empirical risk*, $R_{\text{emp}}[f]$ and a "capacity term" found for the set of hyperplanes from a dot product space $\mathcal{H}$ as stated next. For some η such that $0 \leq \eta \leq 1$ and $m > h$, with probability $1 - \eta$, the following bound holds :

$$R[f] \leq R_{\text{emp}}[f] + \sqrt{\frac{h\left(\log(2m/h) + 1\right) - \log(\eta/4)}{m}} \tag{4}$$

where h, VC dimension (or "model complexity"), which is an integer that cannot be negative and is used to assess the above mentioned capacity. Right-hand side of (4) is called the the "risk bound" and the second term of this part is called the "model complexity penalty" or "VC confidence". Several key points can be noted about this bound. First, it is free of the distribution $P(x, y)$ and here only assumption is made that both the training and test data are taken independently following this distribution. Second, the left hand side of the bound is usually not possible to compute whereas the right hand side of it can be computed if we know h. Third, learning process is said to be consistent if both of the expected risk and the empirical risk converge to the minimum value of the risk, which can be found for $m \to \infty$ and at last, for given some diverse learning machines (equivalently, family of functions), selecting the one, which minimizes the right hand part of (4) will be taking the machine that gives the least upper bound imposed on the actual risk in its left hand part.
Moreover, going into a bit more explicit about the error or risk of the machine depending on its model complexity we see that the VC dimension, h of a set of functions describing their *capacity* controls the main characteristic of a machine. SVM, by applying structural risk minimization (SRM) [47] principle, which is an inductive method that chooses a model to learn from a training dataset with finite examples, acting as an indicator of this capacity control by proposing a trade-off between the empirical error and this VC dimensions, h, equivalently, the hypothesis of space complexity. Having a form of convex optimization, SRM applies a set of sequential models with an ascending order of complexity. Figure 2 briefly displays how the total model error changes depending on the model-complexity-index of a machine that can be analyzed from the relation (4). For low-complex models (similar to high bias but low variance in relation (1) during a stochastic machine learning), the error gets higher as simpler models are unable to learn all of the data-complexity leading to an underfitting situation. On contrary, for higher model indices (similar to high variance but low bias in relation (1) during the learning of a stochastic machine), the structure tries to adapt its learning model with higher concentration to the very specific training data leading to an overfitting condition, which reduces the amount of training error with increasing the model complexity but at the cost of a worsening performance on the test data. Thus the optimal model index, h^* lies between these higher and lower values of this model index where the model gets down to its lowest value.

2.2.2 Kernels with SVM

In the very simple pattern recognition problems, SVM utilizes a linear separating hyperplane to form a classifier with the largest margin. To achieve this, the learning problem for SVM will be formulated as a constrained nonlinear programming problem where the objective function are quadratic and the constraints are linear, that means, it will be needed to solve a QP(quadratic programming) problem. On the other hand, when a dataset cannot be separated linearly in the original input space, the soft-margin SVM struggles to manage a reliable separating hyperplane that both minimizes misclassified data points and general-

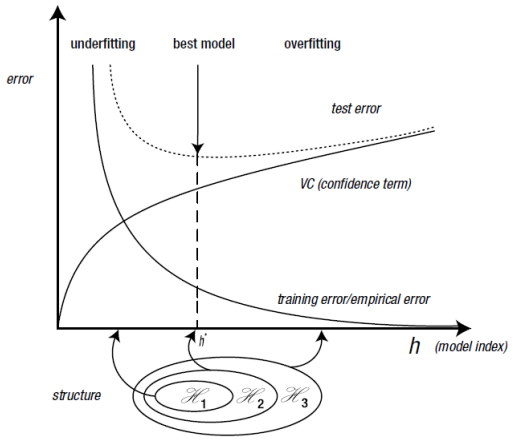


Figure 2: Error trends on model complexity [48]

izes effectively. But following Cover's theorem [49], data points get more probable to be in the form for being linearly separable after they are nonlinearly mapped into a higher-dimensional space. In this regard, SVM uses Kernel method, which was first introduced in [50] [51] and implicitly performs a nonlinear transformation of the input training data from its existing space to a much higher-dimensional feature space (or kernel space) by replacing the inner product by an optimum positive definite function (or kernel function, strongly related to Mercer's Theorem [52]), where data will be linearly separable (figure 3 on page 166) rather than solving a high-order separating hypersurface in the input space and this is accomplished before solving the learning problem through a convex optimization. This approach is preferred because a nonlinear separating boundary in the input space escalates computational demands during the optimization phase. Nonlinear kernel functions lessen the curse of dimensionality [53] productively. However, selecting and setting the right kernel function are vital for SVM optimality.

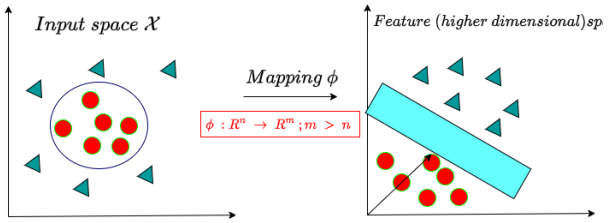


Figure 3: Non-linear transformation of data through kernel methods: mapping input space to a much higher dimensional feature space, realizing linear separation of dataclasses in the feature space, which is not linear in input space.

Mercer's theorem for kernels

Mercer's Theorem is a fundamental concept from Functional Analysis to provide the oretical basis for using kernels in SVM, which can be described in the following way. Let $K(x, y)$ be a continuous function defined on a compact set $X \times X$. Mercer's Theorem states that if $K(x, y)$ is *positive semi-definite* (i.e., for all d-dimensional vectors $\mathbf{v}$, the matrix $[K(x_i, x_j)]$ is positive semi-definite), then there exists a *feature space* and a *mapping* $\phi(x)$ such that:

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$

Explaining in another way, the function $K(x, y)$ can be expressed as the inner product of two vectors in some high-dimensional (even possibly, infinite-dimensional) feature space (equivalently, RKHS) where some key concepts are described below:

Key concepts

- **Positive Semi-Definiteness**: A function $K(x, y)$ is positive semi-definite if, for any finite set of points $\{x_1, x_2, \ldots, x_d\}$, the matrix formed by evaluating $K(x_i, x_j)$ for all pairs (x_i, x_j) is positive semi-definite. This means that all eigenvalues of the matrix are non-negative.
- **Reproducing Kernel Hilbert Space (RKHS)** [54–57]: The feature space $\mathcal{H}$ associated with the kernel function $K(x, y)$ is called the RKHS. The kernel K defines an inner product in this space, and the function $\phi(x)$ maps the original input space to this RKHS.
- **Significance for Machine Learning**: Mercer's Theorem justifies the use of kernel functions like the *Gaussian (RBF) kernel*, *polynomial kernel*, and others in machine learning. the orem ensures that these kernels correspond to valid inner products in high-dimensional feature spaces, enabling the use of linear methods in these spaces.

Most commonly used kernels are: **Polynomial kernel:** $K(x, y) = (x^T y + c)^d$ **& Gaussian (RBF) kernel:** $K(x, y) = \exp\left(-\frac{\|x-y\|^2}{2\sum^2}\right)$

For further kernels:

It is possible to construct more complex kernel functions by combining simpler kernels.

- *Non-negative Linear Combination of Mercer kernels also produces a Mercer kernel :* If $K_1(x, y), K_2(x, y), \ldots, K_n(x, y)$ are Mercer kernels and $\alpha_1, \alpha_2, \ldots, \alpha_n \geq 0$, then $K(x, y) = \sum_{i=1}^{n} \alpha_i K_i(x, y)$ is also a Mercer kernel.
- *Product of Mercer kernels result in a Mercer kernel :* If $K_1(x, y)$ and $K_2(x, y)$ are Mercer kernels, then $K(x, y) = K_1(x, y) \cdot K_2(x, y)$ is also a Mercer kernel.

2.2.3 SVM With mathematical programming

Being mathematically solid and hence powerful, SVM has become one of the most popular supervised machine learning method for classification where the problem is framed as estimating a decision boundary under uncertainty. Due to its high power and minimal classification error despite sparsity, it has gained high popularity. Initially, it was introduced through Quadratic Programming (QP), known as QPSVM. However, as the L_1-norm is more likely to produce sparser solutions than L_2-norm [58], Vapnik modelled an alternative SVM model using Linear Programming (LP) to design the objective function, called VLPSVM. This part addresses SVM and its two core forms (QPSVM and VLPSVM), with further particulars provided in [59].

Quadratic programming (QP) based SVM

This is the standard SVM, a leading algorithm in the field of machine learning. QPSVM uses structural risk minimization concept from statistical learning theory implementing margin maximization between two classes that aims to find the data depending separating hyperplane, which is as far as possible from the nearest data point. Let us consider a binary classification problem using a set of given training patterns $\{(x_i, y_i)\}_{i=1}^{N}$; $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. To divide these patterns into two classes effectively, SVM determines a classifier based on a decision function having the form $f(x) = w \cdot \phi(x) + b$, leading to $class(x) = sgn(f(x))$, where $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$ is a kernel operation. The primal domain formulation of this QPSVM is as follows:

$$\min_{w,b,\zeta} f_P(w) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\zeta_i \tag{5}$$

$$s.t.\ y_i\big(w \cdot \phi(x_i) + b\big) \geq 1 - \zeta_i \tag{6}$$

$$\zeta_i \geq 0;\ i = 1, 2, ..., N \tag{7}$$

where the auxiliary variables $\zeta_i > 0$ work mainly for patterns that remain outside their respective class-margins (or we call it here, "margin-outward-deviated patterns") with $C > 0$ is a data-influenced parameter to regularize the classifier following bias-variance tradeoff concept in stochastic learning. The corresponding dual problem becomes:

$$\max_{\alpha} f_D = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \phi(x_i) \cdot \phi(x_j) \tag{8}$$

$$s.t \sum_{i=1}^{N}\alpha_i y_i = 0 \tag{9}$$

$$0 \leq \alpha_i \leq C;\ \ i = 1, 2, ..., N \tag{10}$$

which is also a QP problem and the patterns for which $\alpha_i > 0$ are the support vectors (SVs). They can also be called as the stochastic decision points as they act as the bases of the final decision function through their coefficients α, which are used to determine the bias term, b and the weight variable w as $w = \sum \alpha_i y_i \phi(x_i)$. An illustrative depiction of this margin based SVM is provided in figure 4 on page 167.

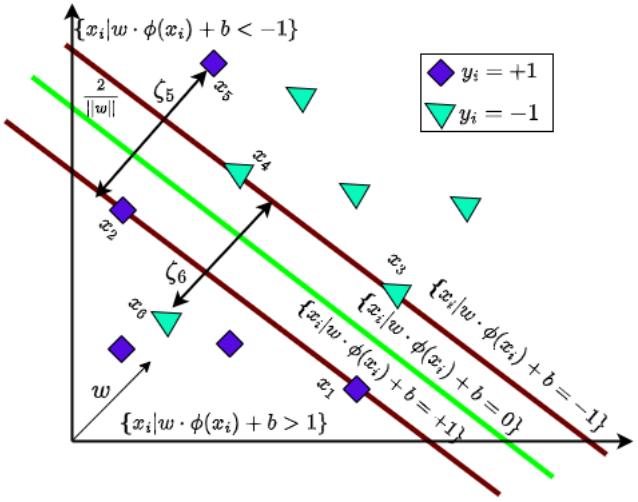


Figure 4: A visual representation of the standard SVM is shown, where patterns for the positive and negative classes are depicted by squares and triangles, respectively. The green straight line represents the optimal separating hyperplane, and the brown lines mark the margins for the classes. Patterns that lie outside their respective classes are referred to as outward-deviated patterns, where the depth of this deviation is represented by a non-negative variable, ζ. The goal of this QP-based SVM is to maximize the margin while minimizing the training error as much as possible. The green line is also known as the decision boundary from the training output.

Linear programming (LP) based SVM

Vapnik proposed an LP model to find a separating hyperplane, which we name here VLPSVM and effectively incorporates an indirect L_1 norm while modeling the cost function, promoting even more sparsity while maintaining accuracy comparable to QPSVM that uses the L_2 norm. As a result, VLPSVM shares similarities with QPSVM but demands less computational effort for classifying patterns. If w represents the weight vector of the hyperplane for VLPSVM, its decision function is given by: $f(x) = w \cdot \phi(x) + b$ leading to $class(x) = sgn(f(x))$. The primal domain formulation of this VLPSVM model is as follows:

$$\min_{\beta,\zeta,b} \sum_{i=1}^{N}\beta_i + C\sum_{i=1}^{N}\xi_i \tag{11}$$

$$s.t.\ y_i\Big(\sum_{j=1}^{N}\beta_j y_j \phi(x_j) \cdot \phi(x_i) + b\Big) \geq 1 - \xi_i \tag{12}$$

$$\beta_j \geq 0;\ j = 1, 2, ..., N \tag{13}$$

$$\xi_i \geq 0;\ i = 1, 2, ..., N \tag{14}$$

In this case, the auxiliary variables $\xi_i > 0$ are assigned

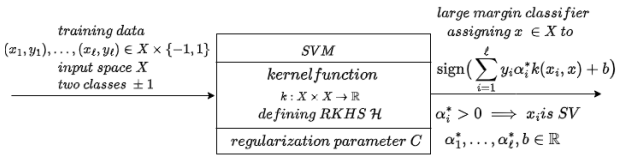


Figure 5: Classification with a kernel based SVM. The learning method is completely expressed by the used kernel function k along with the penalty parameter C. Having training data, the machine produces the coefficients α for the decision function [61].

to handle with the unity-outward-deviated patterns (i.e., the training patterns that give $ClassLabel(x_i) \cdot f(x_i) < 1$) and these ξ are also optimization variables for the problem (11)-(14) that are found with the other variable βs, b depending on the data-dependent stochastic parameters, kernel parameter and penalty parameter. To avoid confusion and for sharp clarification, here, training examples (x_j) of this VLPSVM, for which, coefficients $\beta_j > 0$ are labeled as Expansion Vectors (EVs), which are quite similar to the support vectors (SVs) in QPSVM while being different from both topological and physical points. These optimum value of βs are used to determine $w = \sum \beta_j y_j \phi(x_j)$. The number of EVs found from the VLPSVM algorithm are usually very much smaller compared to the total number of training patterns proving further sparsity of VLPSVM model. Our experiments on benchmark data as well as other works [60] show that VLPSVM has nearly the same performance as QPSVM despite of being more efficient due to less kernel execution in classification stage showing VLPSVM to be more productive for many cases, specially, when classification cost, speed, resources, etc are under constraints. Further, this LPSVM simplifies the optimization problem by using linear programming, making it computationally efficient for certain stochastic processes and for this model, the kernel, $K(.,.)$ does not need to satisfy Mercer's conditions [59]. A graphical representation of this (kernel based) SVM is given in figure 5 on page 168.

3 Proposed method with stochastic functional ξ- analysis

3.1 Research gap, motivation & our direction

Despite of being the most mathematically sound machine learning algorithm by using the famous statistical learning theory, maximum margin concept, and mathematical programming, while working on probabilistic data, implementing total risk minimization principle, SVM does not directly control the model complexity but it automatically determines it by selecting the type and number of stochastic decision points (support vectors (SVs)) depending on the data-dependent parameters. These SVs use kernel operations that highly boosts the expressive capability of the decision function while preserving the underlying linearity that ensures the learning to remain tractable. However, on top of the mounted cost by excessive kernel computations from a large number of SVs, this increased flexibility raises the risk of overfitting since the selection of separating hyperplane turns to be increasingly ill-posed because of the number of degrees of freedom [62]. This becomes more serious in case of large and complex data with noise. Thus managing the machine from getting excessive number of SVs is necessary regarding both cost and accuracy.

Moreover, as the standard SVM algorithm concentrates on error performance indirectly through cost-function and constraints using data-dependent parameters, these parameters play a very crucial role in the algorithm, specially, finding the best trade-off between model-complexity and generalization. However, as there is no exact and analytic expression for these stochastic parameters, we have to find their values using cross-validation method that does not give the exact best and unique values of these parameters. On top of this, it is also followed that to improve a very little amount of classification accuracy on complex dataset, SVM generates a relatively larger number of support vectors (represented in figure 6). Additionally, because of the continuous raise in the heavy size and complexity of data in parallel with modern technological development, further speed up of the classification process is being demanded continuously, which stipulates one to work on reducing the number of SVs as much as possible.

Thus, while the existing methods for SVM cost reduction are either 1) complex method that is very expensive and not so convenient to implement or 2) bear uncertainty to get as being non-convex or 3) cannot gain sparsity and accuracy simultaneously upto that level, a mathematical and computational analysis based suitable operation on the first-step solution from SVM algorithm may also be productive and useful. One of the simplest but heavily productive approach in this direction could be to reduce the support vectors by discarding the least significant SVs, misleading SVs, or such kernel computing vectors by sacrificing an insignificant amount of classification accuracy (even in case it becomes demanded, the worst scenario). But while discarding SVs without considering their contribution can create negative impact on decision-making, in practice, training data may contain noise or outliers that stay far from their main cluster and as the SVM decision boundary relies on SVs, the presence of outliers in the SVs set can make the decision function loaded with more misleading info and shape the boundary less smooth, or may even cause overfitting, especially with complex datasets. Hence, a method to reduce the number of support vectors (as well as the machine-misleading noise due to the noise in the data) after analyzing the outlier's right-leading or misleading on the decision function is rewarding with respect to both cost and accuracy. In this direction, we start with the basis of Statistical Learning Theory where we try to further reduce the Right

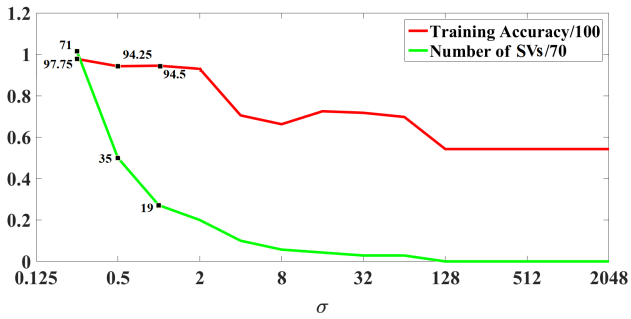


Figure 6: The number of SVs fluctuating with training accuracy. The plots are normalized by dividing them by their respective maximum values, bringing within 0 and 1.

Hand Side of (4) by reducing both of its 1st term and 2nd term simultaneously that eventually lowers the upper bound on the actual risk, leading towards a machine having better generalization performance than before while also reducing the classification cost.

This is achieved by using some mathematical and computational analysis with the stochastic parameter based values of the discriminator function, auxiliary variable, and coefficient of the stochastic decision points found from the first step SVM solution to select the insignificant SVs, misleading and noisy SVs and discard from them as many as possible while preserving the machine's efficiency and performance.

Experimental results demonstrate that our proposed method significantly reduces classification costs while posing classification accuracy analogous to the standard SVM, particularly with large and complex training data that contains many outliers.

3.2 Deterministic mapping of data

As the numerical values of raw data varies widely, scaling it may help to avoid numerical instability by dropping data redundancy and irregularity that is useful for the objective function to work properly and to improve the convergence of the optimization algorithm. This also gets further motivated from the reporting about data preprocessing for higher accuracy by Alaa et al. [63] and about data scaling to improve the condition number of the matrix in the optimization problem [64]. However, while doing this, we must try for such a scaling that may not hamper the crucial properties of data. For example, we can try for a careful scaling of data that may allow a bit change in its topological spanning without noticeably losing its main stochastic property. But we also know that the mean of a scaled dataset is scaled by the scaling factor in each dimension and the elements of the co-variance matrix of the scaled dataset is scaled by the square of the scaling factor (proof is given in the later part of this section). Hence, we decide first to pick the maximum absolute of all the feature values and then divide each entry by this, which maps each entry of the column to lie in [-1,1]. This kind of deterministic transformation would help to evade data leakage in the stage of model testing but may turn out to be sensitive to the outliers that are very common in the real-world problem. Hence, by scaling we bring feature values ≤ 1 and then throw the machine vectors for which corresponding stochastic auxiliary variables ξ give values ≥ 1 by following some numerical analysis based on the stochastic functional analysis described next. A pictorial view of before and after scaling of a benchmark and a synthetic data set is given in figure 7 on page 170.

Statistical impact on mean and variance by scaling a multidimensional dataset

Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be a set of n data points, where each data point $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ is a vector in $\mathbb{R}^d$ (a d-dimensional space). If We scale this dataset by multiplying with a constant factor $\lambda > 0$, then each of the data points will be multiplied by λ and the new dataset becomes: $\mathbf{x}'_i = \lambda \cdot \mathbf{x}_i = (\lambda x_{i1}, \lambda x_{i2}, \ldots, \lambda x_{id})$. So, the scaled dataset $\mathbf{X}'$ contains the scaled vectors $\mathbf{x}'_i \in \mathbb{R}^d$ Now, we know that, in case of the original (unscaled) data, the mean vector of $\mathbf{X}$ be $\boldsymbol{\mu}_\mathbf{X} = \mathbb{E}[\mathbf{X}]$ and the covariance matrix of $\mathbf{X}$ be:

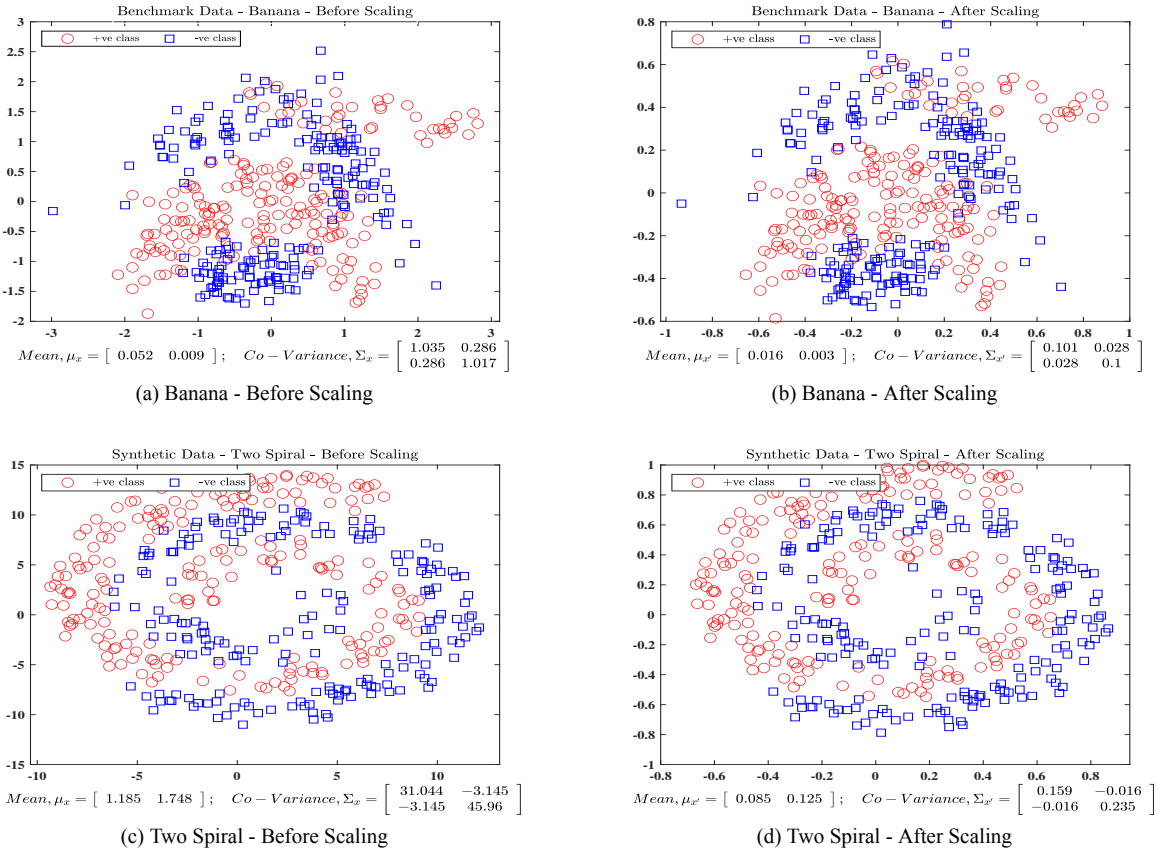
$$\sum\nolimits_{\mathbf{X}} = \mathbb{E}[(\mathbf{X} - \boldsymbol{\mu}_\mathbf{X})(\mathbf{X} - \boldsymbol{\mu}_\mathbf{X})^\top]$$

Then in case of the scaled data, the mean vector of $\mathbf{X}'$ be $\boldsymbol{\mu}_{\mathbf{X}'} = \mathbb{E}[\mathbf{X}'] = \mathbb{E}[\lambda\mathbf{X}] = \lambda\mathbb{E}[\mathbf{X}] = \lambda\boldsymbol{\mu}_\mathbf{X}$ and the covariance matrix of $\mathbf{X}'$ be: $\sum_{\mathbf{X}'} = \mathbb{E}[(\mathbf{X}' - \boldsymbol{\mu}_{\mathbf{X}'})(\mathbf{X}' - \boldsymbol{\mu}_{\mathbf{X}'})^\top] = \mathbb{E}[(\lambda\mathbf{X} - \lambda\boldsymbol{\mu}_\mathbf{X})(\lambda\mathbf{X} - \lambda\boldsymbol{\mu}_\mathbf{X})^\top] = \lambda^2\mathbb{E}[(\mathbf{X} - \boldsymbol{\mu}_\mathbf{X})(\mathbf{X} - \boldsymbol{\mu}_\mathbf{X})^\top] = \lambda^2\sum_{\mathbf{X}}$. That is, the covariance matrix of the scaled random vector is scaled by λ^2. But the *variance* of a single dimension of a dataset is simply the diagonal element of the covariance matrix. Then in case of the original dataset $\mathbf{X}$, $\text{Var}(x_j) = \sum_{X,jj}$ and for the scaled dataset $\mathbf{X}'$, $\text{Var}(x'_j) = \sum_{X',jj} = \lambda^2 \cdot \sum_{X,jj}$ Thus, the variance in each dimension is scaled by a factor of λ^2. We scale the dataset by dividing it by the maximum value of the elements of $\mathbf{X}$, that is in our case, if $\rho = max(max(abs(\mathbf{X}))) \geq 1$ then $\lambda = 1/\rho$ and the new dataset $\mathbf{x}'_i = \lambda \cdot \mathbf{x}_i = \frac{1}{\rho} \cdot \mathbf{x}_i = \frac{1}{max(max(abs(\mathbf{X})))} \cdot \mathbf{x}_i \leq \mathbf{x}_i$ with $\mu_{X'} = \lambda \cdot \mu_X = \frac{1}{\rho} \cdot \mu_X = \frac{1}{max(max(abs(\mathbf{X})))} \cdot \mu_X \leq \mu_X$ and $\sum_{X'} = \lambda^2 \cdot \sum_X = \frac{1}{\rho^2} \cdot \sum_X = \frac{1}{[max(max(abs(\mathbf{X})))]^2} \cdot \sum_X \leq \sum_X$.

3.3 Analyzing stochastic function

In case of learning from probabilistic data, we need to model a data dependent stochastic parameter based discriminator function with the capacity to master the training data well to build optimum generalization ability that can classify novel patterns with least error. For this, it is needed to model the problem set using error-related constraints. Thus in case of both LPSVM and QPSVM, while the decision function are expressed as $f(x) = w.\phi(x) + b$, their error-associated constraints from (6) & (12) can be represented by the relation, $y_i f(x_i) \geq 1 - \xi_i \Rightarrow \xi_i \geq 1 - y_i f_(x_i)$.

Now, considering any wrongly classified training example



(a) Banana - Before Scaling

(b) Banana - After Scaling

(c) Two Spiral - Before Scaling

(d) Two Spiral - After Scaling

Figure 7: Visualization of the effect on 2D data by our way of scaling. It can be clearly observed that after data scaling, in case of Benchmark Banana dataset [65] the mean is scaled by $\frac{1}{3.25}$ and Co-variance is scaled by $(\frac{1}{3.25})^2$ whereas for the Synthetic Two-spiral dataset [66] the mean is scaled by $\frac{1}{13.98}$ and Co-variance is scaled by $(\frac{1}{13.98})^2$. Note that the topological spanning of the datasets are reduced by contracting its spatial scattering but the stochastic property of the datasets are not harmed noticeably.

x_i, if $y_i f(x_i) = -\delta_i$ with $\delta_i > 0$, then $\xi_i \geq 1 + \delta_i$. Thus, for M wrongly classified training examples, $\sum_{i=1}^{M} \xi_i \geq M + \sum_{i=1}^{M} \delta_i$. Hence, suppressing $\sum_i \xi_i$ through regularizer will curb $M + \sum_{i=1}^{M} \delta_i$, not only just M, which is the total training error. It is possible that suppressing $M + \sum_{i=1}^{M} \delta_i$ could take place during the following three conditions: i) Minimizing M faster than $\sum_{i=1}^{M} \delta_i$, which eventually decrease the training error rate, ii) Minimizing M comparatively slower than $\sum_{i=1}^{M} \delta_i$, which may cause unexpected phenomena on training performance, iii) Minimizing M and $\sum_{i=1}^{M} \delta_i$ with the same rate and preference, which may not lower the training error rate expectedly. Thus, throwing P ($P \leq M \leq N$) outlying patterns apparently decreases P training errors while also lowers P kernel executions in classification stage when these example patterns are also elements of the final classifier. Nevertheless, how this outlier throwing impacts the classification of other patterns is an important issue to address. We examine it through a careful analysis of the contribution as well as confusion caused by them in the accurate classification of other patterns, which we refer to as ξ-analysis, and it is explained in the following sections.

Further, the discriminators from both VLPSVM and QPSVM can be rewritten using the following common expression by introducing a shared term, "Kernel Computing Vector (KCV)", for Support Vector(SV), Expansion Vector(EV), or such other kernel operating vectors: $f(p) = \sum_{l \in \text{KCV-set}} \beta_l y_l K(x_l, p) + b$, for p representing any arbitrary pattern with $K(x_l, p) = \phi(x_l) \cdot \phi(p)$ is a kernel function (for example, in case of RBF: $\phi(x_l) \cdot \phi(p) = e^{-\frac{(x_l - p)^2}{2\Sigma^2}}$), which is the leading contributor to computational expenses with all $\beta_l > 0$ involved in the aggregation of time during the classification stage. Following this, we aim to identify a subset from this kernel operating x_l patterns that have $\beta_l > 0$ and discard those patterns that do not contribute meaningfully to decision-making because of being outliers. This is the main focus of our *stochastic ξ-analysis*, which is discussed in the next section.

3.4 Numerical analysis of stochastic auxiliary variable, ξ

A model should ideally take the data's inherent signal and exclude any noise or erratic patterns, which is not a simple task as data is often far from perfect like containing outliers, irrelevant variables, etc, which are related to the stochastic property of data. This is efficiently handled by introducing the auxiliary variable, ξ in the mathematical model. In case of LPSVM, its value comes from the solution of an optimization problem using probabilistic data dependent stochastic parameters. This variable quantifies the degree of violation of the margin constraint for the patterns, which can be interpreted as a measure of uncertainty or noise in the data and following its values, patterns' class-position, patterns' role as well as actual significance in the final discriminator can be analyzed.

3.4.1 Patterns' class-position following ξ

As we have described before that an "outlier" pattern is the one that lies beyond the limits of its class boundary. Thus, a pattern is considered "inlier" if it stays within its own class boundary. Hence, for any arbitrary training pattern x_i having the class-label y_i with the final decision function-value $f(x_i)$ is considered as an inlier when $y_i = sgn\big(f(x_i)\big) \Rightarrow 1 = y_i sgn\big(f(x_i)\big) \Rightarrow y_i f(x_i) > 0$ and is considered as an outlier if $y_i = -sgn\big(f(x_i)\big) \Rightarrow -1 = y_i sgn\big(f(x_i)\big) \Rightarrow y_i f(x_i) < 0$. Expressing the SVM-error-constraints in (6) or in (12) in a common single form utilizing the stochastic auxiliary variable ξ as $y_i f(x_i) \geq 1 - \xi_i \Rightarrow \xi_i \geq 1 - y_i f(x_i)$, we can write for outlier $\xi_i > 1$ while in case of inlier $0 \leq \xi < 1$ with the next two scenarios: i) inlier with $y_i f(x_i) < 1 \Rightarrow 1 - y_i f(x_i) > 0 \Rightarrow \xi_i > 0$ ii) inlier with $y_i f(x_i) > 1 \Rightarrow 1 - y_i f(x_i) < 0 \Rightarrow \xi_i = 0$ as ξ_i cannot be negative.
From the above discussed points, it is evident that patterns with $\xi > 1$ belong to different classes rather than their original classes. Now, we examine their significance by evaluating the roles they play in the discriminating function in case of both LPSVM and QPSVM.

3.4.2 Roles of the $\xi > 1$ patterns in the discriminating function

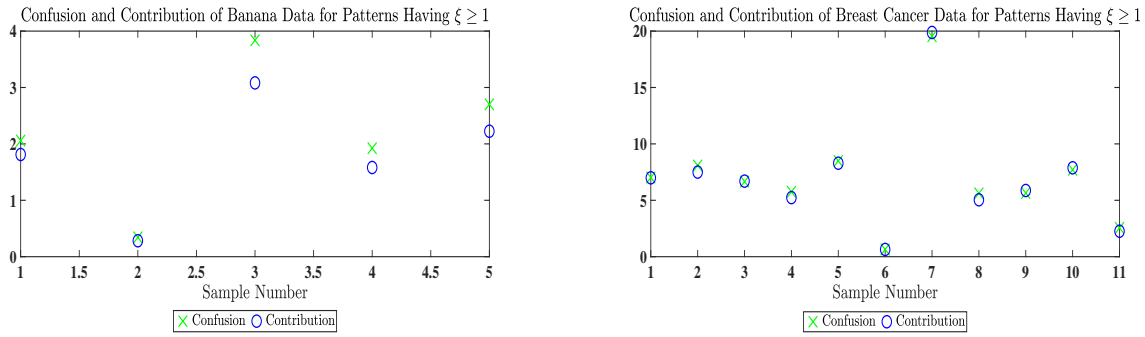
Role while working with QPSVM: QPSVM has a KKT condition as $\alpha_i \Big(y_i \big(w \cdot \phi(x_i) + b \big) - 1 + \zeta_i \Big) = 0$. Using $\alpha_i \neq 0$ & $\alpha_i \Big(y_i \big(w \cdot \phi(x_i) + b \big) - 1 + \xi_i \Big) = 0$ leads to $y_i \big(w \cdot \phi(x_i) + b \big) - 1 + \xi_i = 0 \Rightarrow y_i \big(w \cdot \phi(x_i) + b \big) = 1 - \xi_i$. From this, it can be found that QPSVM gives three types of KCV patterns (that is, patterns having $\alpha_i \neq 0$) considering the values of ξ: i) $\xi = 0$ ii) $0 < \xi \leq 1$ iii) $\xi > 1$. But after a little calculation with another KKT condition, it can be seen that if $\xi > 0$ for any pattern, its corresponding co-efficient value, α becomes equal to the penalty parameter, C that is the upper limit of α values. Thus, discarding patterns where $\xi > 1$ not only ignores a KKT condition but also removes crucial elements of the decision function that have the largest coefficient values, potentially impacting the decision-making process significantly. Additionally, removing patterns having $\alpha = C$ could significantly disrupt the condition $\sum_{i=1}^{N} \alpha_i y_i = 0$, which ensures the balance between the support vector patterns and their associated coefficients, similar to a mechanical system, thereby impacting the optimal margin and the final decision boundary. Therefore, discarding any pattern from this machine is not in our option.

Role while working with VLPSVM: In this case, unlike QPSVM, values of this stochastic auxiliary variables ξ are found by solving an optimization problem directly using its primal form where no direct involvement among KCV-coefficients, KKT conditions, and ξ are seen. Moreover, there is no other constraint-based restrictions involving the KCV coefficients. This provides an opportunity to work further on this machine. Experience with VLPSVM shows that patterns with $\xi > 1$ have relatively small corresponding coefficients β in the final decision function. This aligns with the two main parts of its objective function: $\min_{\beta,\xi,b} \sum_{i=1}^{N} \beta_i + C \sum_{i=1}^{N} \xi_i$. Nevertheless, these coefficient values do not fully vanish, meaning the corresponding patterns continue to be part of the decision function, with each of them requiring kernel evaluation for classification, even though they may not have a considerable constructive impact on the final decision. This brings the arguments about whether they should be included in the final discriminator, considering the trade-off between classification accuracy and computational cost. To address this issue, we examine their contribution as well as the confusion they create in accurate classification, as explained below:

3.4.3 Contribution and confusion from the patterns having $\xi > 1$ while working with VLPSVM

We have seen earlier that these patterns consistently stay beyond the boundaries of their classes, produce comparatively larger kernel values while in operation with patterns from the opposite classes, leading to confusion. On contrary, they generate lower kernel values while in operation with patterns from their same classes, which produces weaker contribution in the final discriminating function. However, there may be confusion from the inlier KCVs by very weakly attracting patterns from the other classes, we ignore this here as they are negligible in this circumstance and efficiently handled by the complete machine.

Now, the final discriminating function that decides about any arbitrary training pattern x_a that has class-label y_a is $f(x_a) = \sum_{l \in KCV-set} \beta_l y_l K(x_l, x_a) + b$. Considering $m, n : m \cup n = \{l\}, m \cap n = \{\}, y_m = -y_n$, then x_m, x_n are KCVs. Now as $\beta_m, \beta_n > 0$ & $K(x_m, x_a), K(x_n, x_a) > 0$, if $\frac{\beta_m y_m K(x_m, x_a)}{y_a} < 0 \Rightarrow \frac{y_m}{y_a} < 0 \Rightarrow x_a$ and x_m do not belong to the



(a) Banana (b) Breast Cancer

Figure 8: Confusion & contribution values for the Patterns having $\xi \geq 1$ from the training set

same class and x_m pulls x_a outwards from x_a's own class (i,e., towards, the opposite class of x_a) and if $\frac{\beta_n y_n K(x_n,x_a)}{y_a} > 0 \Rightarrow \frac{y_n}{y_a} > 0 \Rightarrow x_a$ and x_n are from the same class with x_n pulls x_a towards the x_a's home class. However, from these KCV patterns, those with $\xi > 1$ remain within the boundaries of their opposite classes, produce comparatively larger kernel values while considering samples from the opposite classes, which misleads the discriminating function. Simultaneously, they generate lower kernel values while considering patterns from their same classes, poorly guiding the discriminating function. Thus, to enhance both classification accuracy and speed, we decide to discard these $\xi > 1$ KCV patterns after evaluating their level of confusion (which pushes training patterns away from their respective classes) as well as their contribution (which pulls training patterns towards their own classes) [22], taking into account all training patterns through the following mathematical relations:
Contribution of a $\xi > 1$ KCV, x_O with label y_O on all training patterns, $Contr(x_O) = \frac{1}{Card\{i\}} \sum_{\{i:y_i.y_O=1\}} \beta_{x_O} K(x_O, x_i)$, whereas Confusion of that KCV, x_O on all training patterns, $Conf(x_O) = \frac{1}{Card\{i\}} \sum_{\{i:y_i.y_O=-1\}} \beta_{x_O} K(x_O, x_i)$.

Extensive experiments conducted on the benchmark datasets using the mathematical formulations above reveal that, in almost all cases, $\xi > 1$ KCV patterns pose confusion more than contribution value. This agrees with our decision to remove $\xi > 1$ KCV patterns from the final discriminating function. Figure 8 on page 172 provides an example of a contribution-confusion sketch.

3.4.4 Contribution, confusion as a function of data mapping

We have discussed about our data mapping and its motivation earlier, here we discuss and check our hypothesis that compact mapping reduces the cardinality of the representer set, that is, information space with respect to the representer set gets denser. It is likely that the KCVs identified after

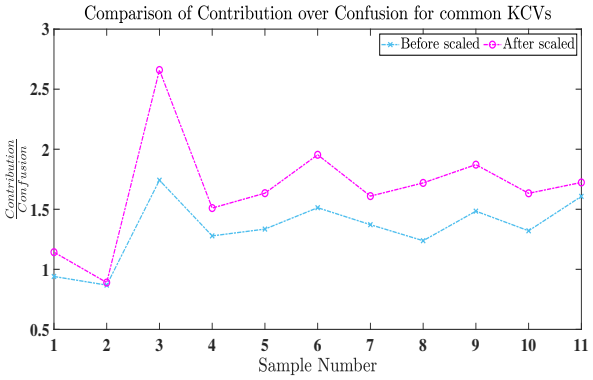


Figure 9: Confusion and contribution ratio for common KCVs from before and after mapping of data

data mapping may form a subset of the KCVs derived from the original non-scaled data or may contain part of it and the KCVs obtained after data mapping should carry more information. To check this, we have computed contribution to confusion ratio (CCR) for the intersecting basis vector sets from before and after mapping. For each of common vectors $KCV_b = B \cap B'$, CCR and CCR' for pre and post mapped basis pattern x_b, x'_b with label y_b, y'_b can be described as follows:

$$CCR = \frac{\frac{1}{\text{card}\{i\}} \sum_{\{i:y_i.y_b=1\}} \beta_{x_b} K(x_b, x_i)}{\frac{1}{\text{card}\{i\}} \sum_{\{i:y_i.y_b=-1\}} \beta_{x_b} K(x_b, x_i)}$$

$$CCR' = \frac{\frac{1}{\text{card}\{i\}} \sum_{\{i:y_i.y'_b=1\}} \beta_{x'_b} K(x'_b, x_i)}{\frac{1}{\text{card}\{i\}} \sum_{\{i:y_i.y'_b=-1\}} \beta_{x'_b} K(x'_b, x_i)}$$

Where, $(x_b, y_b) \in B$, and $(x'_b, y'_b) \in B'$
Our findings show that after data mapping, KCV_b set have a higher contribution-to-confusion ratio, such that $CCR' > CCR$, as illustrated in figure 9 on page 172. This observation supports the idea that the contribution of the reduced basis set becomes significantly higher than the confusion, thereby highlighting the effectiveness of our data mapping.

Algorithm 1: Proposed EBSVM (Efficiency Boosted SVM)

1: **Input:** A set of training data $\{(x_i, y_i)\}_{i=1}^{N}$
2: **Output:** A discriminator $f_{EBSVM}(\cdot)$
3: Scale the training set efficiently
4: Select the best values of the data dependent parameters: $(Penalty-parameter, Kernel-parameter) \equiv (C, \sum)$
5: Run LP based SVM (VLPSVM) using training data and solve the following optimization problem as below:

$$\min_{\beta,\xi,b} \sum_{l=1}^{N} \beta_l + C \sum_{l=1}^{N} \xi_l$$
$$s.t.\ y_i\Big(\sum_{l=1}^{N} \beta_l y_l \phi(x_l) \cdot \phi(x_i) + b\Big) \geq 1 - \xi_i$$
$$\beta_l \geq 0;\ l = 1, 2, ..., N; \xi_i \geq 0;\ i = 1, 2, ..., N$$

6: Obtain the KCVs along with their labels $\{(x_p, y_p)\}_{p=1}^{M}$ and the bias term, b derived from the VLPSVM applying $\beta_l > 0$
7: Find $\{\beta_p | \xi_p > 1\}$ and discard the corresponding patterns belong to the KCVs set and form a new smaller set of KCVs (RKCVs), $\{x_p | \xi_p \leq 1\ with\ \beta_p > 0\}$
8: Retrieve the RKCVs with their labels $\{(x_j, y_j)\}_{j=1}^{R}$
9: $w_{EBSVM} \leftarrow \sum_{j=1}^{R} \beta_j y_j \phi(x_j)$
10: $b_{EBSVM} \leftarrow argmin_{b_R} \sum_{i=1}^{N} \Bigg[\Big(\sum_{p=1}^{M} \beta_p y_p \phi(x_p) \cdot \phi(x_i) + b\Big) - \Big(w_{EBSVM} \cdot \phi(x_i) + b_R\Big)\Bigg]^2$
11: Return $f_{EBSVM}(\cdot) = w_{EBSVM} \cdot \phi(\cdot) + b_{EBSVM}$

3.4.5 Updating bias

Weight vector of our sparser machine w_{EBSVM} is found by taking KCV set, S_R as a subset of the KCV set, S from the full scale LPSVM solution so that $S_R \subseteq S$ and if R and M be the elements of S_R and S respectively, then $R \leq M$. Now, if w be the weight vector of this full scale LP SVM machine, it is very most likely that $w_{EBSVM} = \sum_{j=1}^{R} \beta_j y_j \phi(x_j) \neq w = \sum_{p=1}^{M} \beta_l y_l \phi(x_l)$ and thus for any pattern x, $w_{EBSVM} \cdot \phi(x) + b \neq w \cdot \phi(x) + b$. Hence, it is important to provide an optimally updated bias for our proposed EBSVM.

Now, as $R \leq M$, it is normal that model complexity of EBSVM is less than the model complexity of the full scale LPSVM, $\|w_{EBSVM}\|^2 \leq \|w\|^2$ and hence margin of EBSVM is larger than the margin of full scale LPSVM. In another way, as $R \leq M$, 2nd term or the VC-confidence term of the risk bound (4) of EBSVM becomes smaller than that of the full scale LPSVM. But this reduction in the VC-confidence term due to lesser model-complexity generally raises the 1st term, $R_{emp}[f]$ of the risk bound (4). Hence, we need to focus to reduce this part. We can do it by imitating the discriminator function (value) of a complex machine and here we select the full scale LP SVM one as it is the origin from where EBSVM comes after modification. We do it in the following way:

Let the variable bias value of EBSVM is b_R, whose optimum value would be the final value of our EBSVM. Now if δ be the total squared difference between the function values from the full scale LPSVM and variable bias based EBSVM using all training patterns, then we can write,

$$\delta = \sum_{i=1}^{N} \Bigg[\Big(\sum_{p=1}^{M} \beta_l y_l \phi(x_l) \cdot \phi(x_i) + b\Big) - \Big(w_{EBSVM} \cdot \phi(x_i) + b_R\Big)\Bigg]^2 \tag{15}$$

At the minimum of δ, we should get

$$\frac{\partial \delta}{\partial b_R} = 0 \tag{16}$$
$$\Rightarrow \sum_{i=1}^{N} \Bigg[\Big(\sum_{p=1}^{M} \beta_p y_p \phi(x_p) \cdot \phi(x_i) + b\Big) - \Big(w_{EBSVM} \cdot \phi(x_i) + b_R\Big)\Bigg] = 0$$
$$\Rightarrow b_R^* = \frac{1}{N} \sum_{i=1}^{N} \Bigg[\sum_{p=1}^{M} \beta_p y_p \phi(x_p) \cdot \phi(x_i) + b - w_{EBSVM} \cdot \phi(x_i)\Bigg] \tag{17}$$

This b_R^* is the value of the bias term of the proposed, EBSVM. Algorithm 1 on page 173 outlines the steps of the proposed EBSVM whereas figure 10, 11, 12, 13 show the decision boundaries using both the synthetic and benchmark datasets, from QPSVM, VLPSVM as well as the proposed EBSVM, along with their corresponding KCV counts and training error rate.

3.5 Analyzing noise-sensitivity of the method

To assess the noise sensitivity of our model, we perturbed the input synthetic training data [66] by introducing controlled uniform additive noise with varying levels and observed the behavior of our machine and other two most prominent machines, the standard QP based SVM(QPSVM) and the LP based SVM (LPSVM). To perform a sound comparison among them with solid analysis, we first state the variables with their functional relationships explicitly as the following:

Independent variable: noise, ν.

Dependent variable: Training Accuracy Rate, $A = A(\nu)$; Computational Cost ($\equiv number\ of\ KCV$), $CC = CC(\nu)$.
Different values of these dependent variables depending on the independent variable (noise) are plotted in figure 14 on page 176.

We first follow the sensitivity of the dependent variables with respect to the independent variable and then we measure an the overall effects throughout the whole noise input of different levels.

3.5.1 Sensitivity

We know that the sensitivity of a dependent variable with respect to (its depending) independent variable refers to how much the dependent variable changes in response to a change in the independent variable. Mathematically, this sensitivity is quite often expressed by the the rate of change of the dependent variable with respect to the independent variable. That is, if the dependent variable, F depends on the independent variable, u then sensitivity of F with respect to u is

$$\text{Sensitivity} = \lim_{\Delta u \to 0} \frac{F(u+\Delta u)-F(u)}{\Delta u} = \frac{dF}{du} \tag{18}$$

Thus sensitivity of our two noise dependent variables can be expressed as,

$$\begin{aligned} &\text{Training Accuracy Rate Sensitivity,} \\ &= \lim_{\Delta \nu \to 0} \frac{A(\nu+\Delta\nu)-A(\nu)}{\Delta \nu} = \frac{dA}{d\nu} \end{aligned} \tag{19}$$

and

$$\begin{aligned} &\text{(Kernel) Computational Cost Sensitivity,} \\ &= \lim_{\Delta \nu \to 0} \frac{CC(\nu+\Delta\nu)-CC(\nu)}{\Delta \nu} = \frac{dCC}{d\nu} \end{aligned} \tag{20}$$

We have calculated them using the Forward Difference method of Numerical differentiation as below

$$F'(\nu) \approx \frac{F(\nu+h)-F(\nu)}{h} \tag{21}$$

Sensitivity (that is gradient) of these two variables at different noise levels are counted and hence plotted in figure 15 on page 176. From the figure it is clear that in case of Accuracy sensitivity, all three machine behave very much similarly, by being flat and steady upto 55% of the controlled full noise and then a mixed reaction of rise & fall for higher noise levels. But considering Cost sensitivity curve, while QPSVM shows heavy zigzag, our model and LPSVM give almost flat and steady curves showing they are not much effected by the noise. The most likely reason for this in case of EBSVM is the bias updating on top of the mechanism of our machine by stochastic ξ functional analysis, which smartly interfaces with noises.

3.5.2 Graphical analysis of noise induced output

To evaluate the overall performances of the machines throughout the whole noise introduced operation, we use the concept of Epigraph & Hypograph as given below:

Epigraph and hypograph

Let $F : \mathbb{R}^n \to \mathbb{R}$ be a real-valued function.

Epigraph: The **epigraph** of F, denoted by $\text{epi}(F)$, is defined as:

$$\text{epi}(F) = \{(x,t) \in \mathbb{R}^n \times \mathbb{R} \,|\, F(x) \le t\}$$

Geometrically, this is the set of points on or above the graph of the function.

Area of an epigraph: To compute a finite area, we bound it by a constant $M \ge F(x)$ over the interval $[a,b]$. The area between this bound and the function is:

$$Ar_{\text{epi}} = \int_a^b (M - F(x))\, dx \tag{22}$$

Hypograph: The **hypograph** of F, denoted by $\text{hypo}(F)$, is defined as:

$$\text{hypo}(F) = \{(x,t) \in \mathbb{R}^n \times \mathbb{R} \,|\, F(x) \ge t\}$$

This is the set of points on or below the graph of the function.

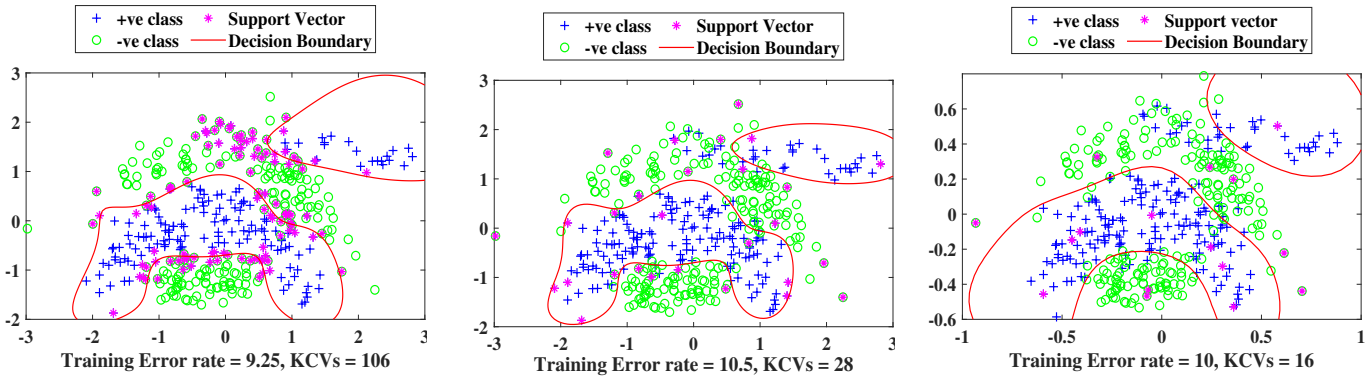
Area of a hypograph: Let $F(x)$ be a continuous real-valued function on an interval $[a,b]$. The area of the hypograph is the area under the graph of the function from a to b, given by:

$$Ar_{\text{hypo}} = \int_a^b F(x)\, dx \tag{23}$$

To compute the area in (22) & (23) we have used the following Simpson's Rule to perform numerical integration

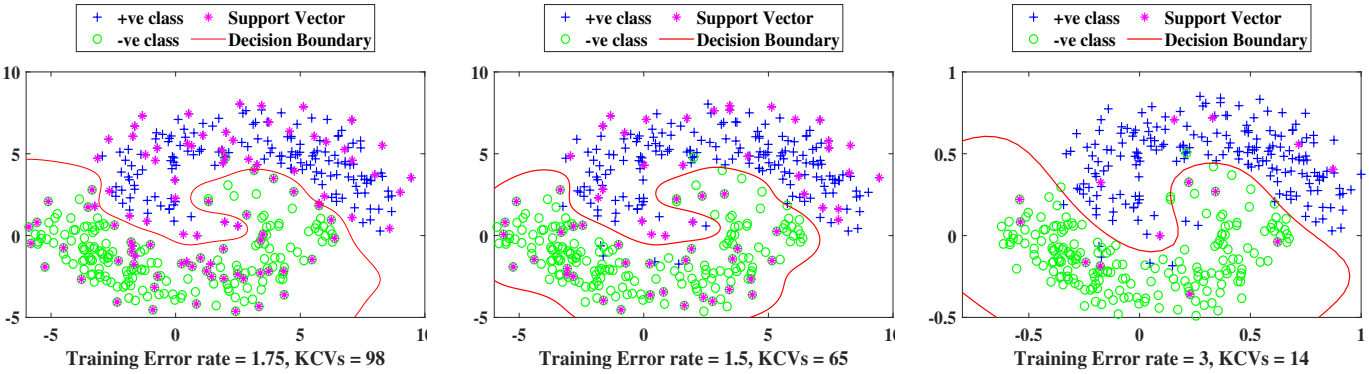
$$\begin{aligned} \int_a^b F(x)dx \approx \frac{\Delta x}{3}[F(x_0) \\ + 4\sum_{i=1}^{n/2} F(x_{2i-1}) + 2\sum_{i=1}^{n/2-1} F(x_{2i}) + F(x_n)] \end{aligned} \tag{24}$$

As we need to maximize Accuracy, we find the area of $hypograph(A(\nu))$, which is the higher the better having the maximum limit equal to 1, whereas as we want to minimize the Computational Cost, we find the area of $hypograph(CC(\nu))$, which is the smaller the better while having the maximum limit equal to 1. Further, the area between $CC = 1$ (meaning 100% computational cost, that is using all of the training patterns to build up the classifier) & $epigraph(CC(\nu))$ is the Sparsity, which is the higher the better having the maximum limit equal to 1, that is, $area(epigraph(CC(\nu))) - area(epigraph(CC = 1)) = 1 - area(hypograph(CC(\nu))) = Sparsity$.



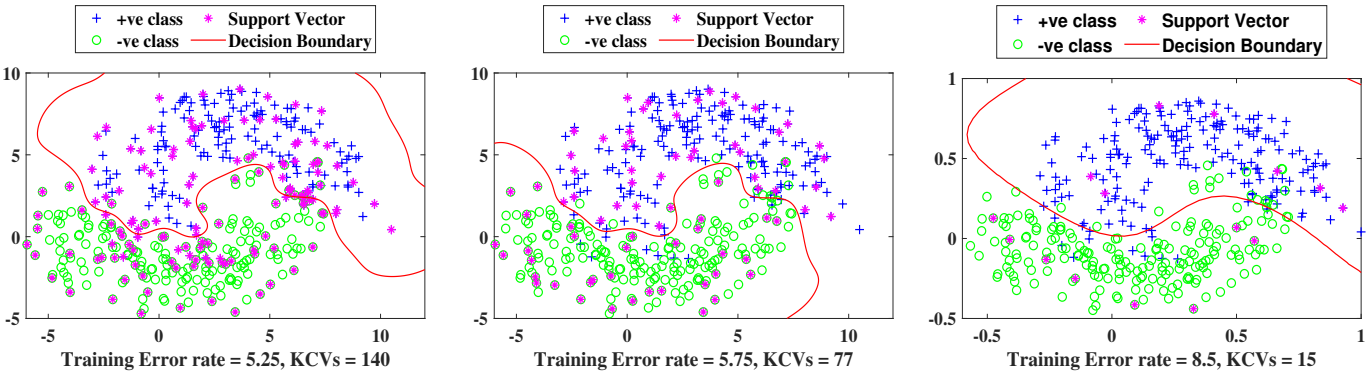
(a) QPSVM (b) VLPSVM (c) Proposed EBSVM

Figure 10: Pattern Classification Boundaries on Banana data along with their respective KCVs counts and training error rates for different machines having the size of training set 400x2.



(a) QPSVM (b) VLPSVM (c) Proposed EBSVM

Figure 11: Pattern Classification Boundaries on synthetic Two spiral data (with noise level 3.3) along with their respective KCVs counts and training error rates for different machines having the size of training set 400x2.



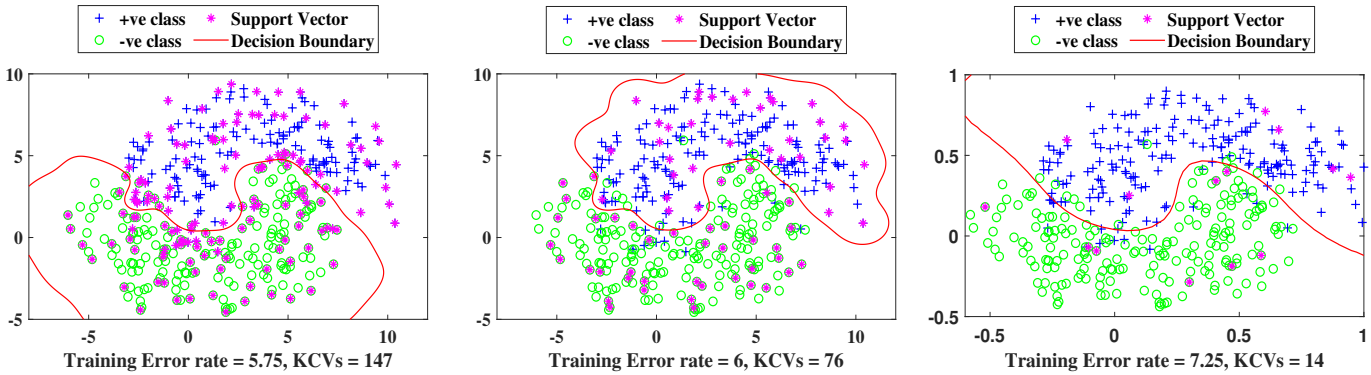
(a) QPSVM (b) VLPSVM (c) Proposed EBSVM

Figure 12: Pattern classification boundaries on synthetic two spiral data (with noise level 4.3) along with their respective KCVs counts and training error rates for different machines having the size of training set 400x2.

4 Experimental set-up and results

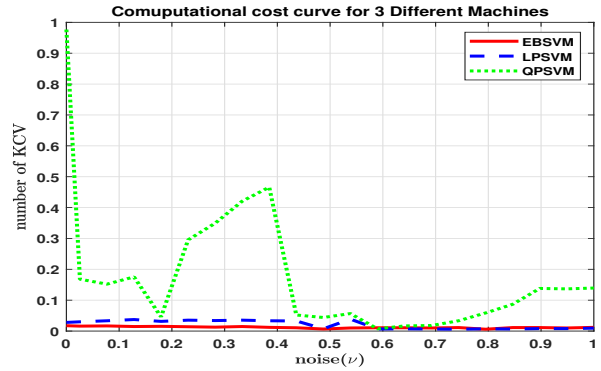
4.1 Set-up method

In this section, the effectiveness of the suggested EBSVM algorithm is demonstrated by a comparison of its experimental outcomes with those of many state-of-the machines, including QPSVM [45], VLPSVM [59], EESVM [1] and

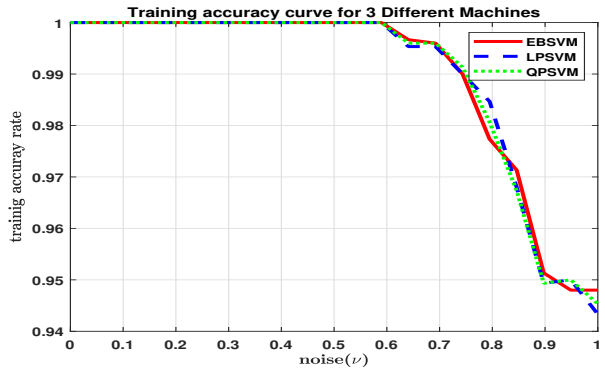


(a) QPSVM (b) VLPSVM (c) Proposed EBSVM

Figure 13: Pattern classification boundaries on synthetic two spiral data (with noise level 4.6) along with their respective KCVs counts and training error rates for different machines having the size of training set 400x2.

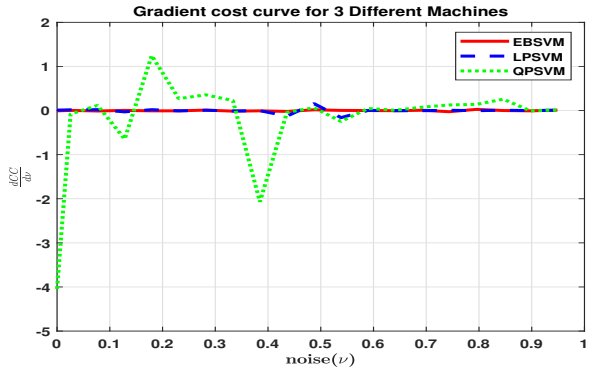


(a) Computational cost

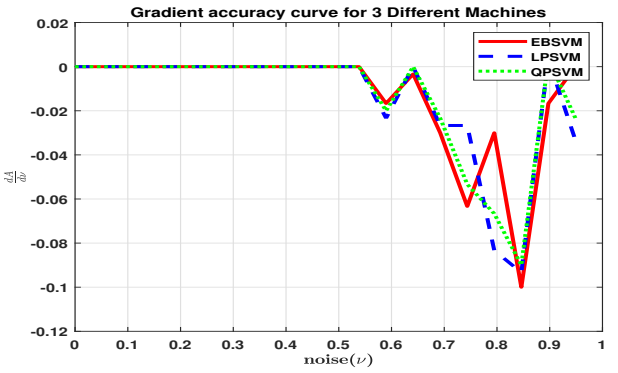


(b) Training accuracy rate

Figure 14: Comparison of (a) computational cost and (b) training accuracy rate with respect to noise levels for EBSVM, LPSVM, QPSVM, respectively.



(a) Computational cost



(b) Training accuracy rate

Figure 15: Sensitivity measurement of (a) computational cost and (b) training accuracy rate with respect to noise levels for EBSVM, LPSVM, QPSVM, respectively.

SpSVM-2 [2]. The outcomes are broken down into two dimensions: the algorithm's classification cost and it's performance. Ten publicly available benchmark datasets — Heart, Twonorm, Titanic, Thyroid, Breast Cancer, Banana, Flare Solar, German, and Waveform are used to evaluate the algorithm's performance, as referenced in [65]. Each of these datasets includes 100 realizations of both training and test sets. Scaling is executed on each dataset by dividing with the maximum among the absolute values of the features, as discussed in the Data Scaling section. All studies are conducted using a Gaussian kernel on these scaled datasets. A varied collection of $C \in \{2^0, 2^2, ..., 2^{20}\}$ and

$\sigma \in \{2^{-2}, 2^0, 2^2, ..., 2^8\}$ is employed in fivefold cross-validation, wherein each C value is normalized by the scaling factor of the respective dataset, thereby calibrating the computation of C for each dataset pattern. Fivefold cross-validation is employed to determine the optimal parameters, extracting the best values of C and σ for each dataset. The initial training set among 100 is utilized in cross-validation whereas output results (values of different key terms/variables) are collected by taking mean over the 100 example-sets given in the benchmark datasets. The mean number of KCV values and the test error rates for QPSVM, VLPSVM, SpSVM-2, EESVM, and the proposed approach are presented in Table 1 on page 178. The proposed EBSVM achieves similar accuracy with all of these SOTA machines while reducing the KCV count dramatically compared to the standard SVM (QPSVM, which is sparse itself), significantly compared to the sparser VLPSVM while considerably compared to the further sparser EESVM and SpSVM-2. Along with data-description, details of number of KCVs, Error rates from different machines and the proposed one are described in Table 1. All of the experiments were done using MATLAB.

Numerical operations and outputs during noise-sensitivity experiment

We normalize the Training Accuracy axis (in percentage) by 100 as 100% accuracy is the best whereas the noise axis is normalized by the highest noise level. Thus the total area of region from this normalized Noise-Accuracy window is equal to 1. Moreover, as SVMs are sparse by using only a small subset of training points (support vectors) to determine its final model while many of the other machines, for example, k-Nearest Neighbors (k-NN), Kernel Density Estimation (KDE), etc are not sparse as they involve all of the training patterns to build the final machine. Thus we normalize the Computation Cost (i.e., number of KCV) axis by total number of training patterns. Hence the total area of the region from this normalized Noise-Cost window is equal to 1.

To calculate the overall Training Accuracy Rate, A, we use (23) & (24) replacing F by A and using $a = 0, b = 1$. On contrary, to calculate the overall Computational Cost, CC, we use (23) & (24) replacing F by CC and using $a = 0, b = 1$ while to compute the overall sparsity, we use $M = 1$ in (22). All calculated results are show in Table 2 on page 179. From the Table 2 it is seen that while our method, EBSVM gives very similar overall Training Accuracy Rate with respect to the prominent QPSVM and LPSVM, it offers the lowest normalized computational cost region 0.012, being nearly half of LPSVM and 1/13th of QPSVM. In terms of sparsity, while all of these 3 machines generate heavy normalized sparsity region, proposed EBSVM shows the highest sparsity as 99% , whereas the sparser LPSVM gives 98% and the sparse-famous QPSVM gives 84%.

4.2 Result analysis

4.2.1 Error-accuracy in stochastic classification

For any arbitrary pattern x having class label y, our discriminator models the decision function in the form: $f(x) = w^T\phi(x) + b$ with the predicted class label as $Classified\ Class\ Label(x) = y' = sgn(f(x))$. Thus a novel pattern is classified correctly if its decision function value has the same polarity as its actual class label, i.e., $f(x) > 0$ if it is from positive class and $f(x) < 0$ if it is from negative class and consequently, during this detection (to accept positive patterns and reject negative patterns) by classifications of patterns, two types of errors are to be considered first. They are:

False positive error: This is the error when a Negative pattern is considered as a Positive pattern by mistake. We denote this rate or probability as $P_0(x = 1)$. Mathematically, we can define it as,

$$\begin{aligned} & P_0(x=1) \\ & = \frac{Number\ of\ Falsely\ accepted\ Negative\ Pattern}{Total\ number\ of\ Input\ Negative\ Pattern} \\ & \Rightarrow P(y'=1|y=-1) \\ & = \frac{Number\ of\ Falsely\ accepted\ Negative\ Pattern}{Total\ number\ of\ Input\ Negative\ Pattern} \\ & \Rightarrow P(y'=1|y=-1) \\ & = \frac{\frac{Number\ of\ Falsely\ accepted\ Negative\ Pattern}{Total\ number\ of\ Input\ Pattern}}{\frac{Total\ number\ of\ Input\ Negative\ Pattern}{Total\ number\ of\ Input\ Pattern}} \\ & \Rightarrow P(y'=1|y=-1) = \frac{P(y'=1, y=-1)}{P(y=-1)} \end{aligned} \tag{25}$$

Where $P(y' = 1|y = -1)$ is the conditional probability that the predicted class label y' is positive but the actual class label y is negative, $P(y' = 1, y = -1)$ is the joint probability distribution of this predicted and actual class labels, $P(y = -1)$ is the marginal probability distribution of negative class labels. This can also be thought as $1 - Negative\ Rejection\ Rate$. *Negative Rejection rate* is often termed as *power*.

False negative error: This is the error when a Positive pattern is considered as a Negative pattern by mistake. Sometimes,it is also called as "missed detection". We denote this rate or probability as $P_1(x = 0)$. Mathematically, we can define it as,

$$\begin{aligned} & P_1(x=0) \\ & = \frac{Number\ of\ Rejected\ Positive\ Pattern}{Total\ number\ of\ Input\ Positive\ Pattern} \\ & \Rightarrow P(y'=-1|y=1) \\ & = \frac{Number\ of\ Rejected\ Positive\ Pattern}{Total\ number\ of\ Input\ Positive\ Pattern} \\ & \Rightarrow P(y'=-1|y=1) \\ & = \frac{\frac{Number\ of\ Rejected\ Positive\ Pattern}{Total\ number\ of\ Input\ Pattern}}{\frac{Total\ number\ of\ Input\ Positive\ Pattern}{Total\ number\ of\ Input\ Pattern}} \end{aligned}$$

Table 1: Number of KCVs as well as test error rates for various state-of-the art machines and our proposed method (EBSVM)

Name of Dataset, (#Training Examples, #Testing Examples, #Dimension)	QPSVM [45] (KCVs, TestErR)	VLPSVM [59] (KCVs, TestErR)	EESVM [1] (KCVs, TestErR)	SpSVM-2 [2] (KCVs, TestErR)	EBSVM (KCVs, TestErR)
Banana (400, 4900, 2)	(102.26, 10.61)	(15.08, 10.75)	(13.62, 11.53)	(17.30, 10.87)	(16.64, 10.65)
Breast Cancer (200, 77, 9)	(200, 28.53)	(18.89, 26.05)	*NR	(12.10, 29.22)	(**8.01**, 27.56)
Diabetes (468, 300, 8)	(263.22, 23.28)	(12.58, 23.40)	*NR	(13.80, 23.47)	(**8.81**, 23.66)
Flare Solar (666, 400, 9)	(609.98, 32.56)	(251.66, 32.41)	*NR	(8.40, 33.90)	(**3.87**, 32.32)
German (700, 300, 20)	(438.75, 23.70)	(26.47, 24)	(27.61, 24.16)	(14.00, 24.90)	(19.35, 24.28)
Heart (170, 100, 13)	(68.23, 16.60)	(21.94, 17.44)	(7.45, 15.33)	(4.30, 15.50)	(7.42, **15.23**)
Ringnorm (400, 7000, 20)	(77.00, 2.23)	(15.99, 1.73)	(15.83, 2.54)	(12.9, 1.97)	(15.84, 1.86)
Splice (150, 2051, 3)	(943.65, 12.34)	(292.15, 12.39)	*NR	*NR	(**283.25**, 12.39)
Titanic (150, 2051, 3)	(148.50, 22.68)	(83.91, 22.91)	(39.19, 23.18)	(3.30, 22.68)	(5.70, 23.12)
Waveform (400, 4600, 21)	(228.33, 13.15)	(20.81, 11.18)	(10.67, 12.72)	(14.40, 10.66)	(13.01, 10.81)
Overall Average	(307.99, 18.55)	(75.95, 18.24)	(- , -)	(- , -)	(**38.19**, 18.37)
Average (1) w.r.t SpSVM-2	(237.36, 19.24)	(51.93, 18.89)	(- , -)	(11.17, 19.24)	(**10.96**, 19.03)
Average (2) w.r.t EESVM	(177.18, 14.83)	(30.7, 14.67)	(19.06, 14.91)	(- , -)	(**12.99**, 14.55)

*NR: Not Reported

Table 1 provides a detailed comparison of Kernel Computing Vectors (KCVs) and test error rates (TestErR) across various SVM models including QPSVM, VLPSVM, EESVM, SpSVM-2, and the proposed EBSVM. The results highlight that EBSVM achieves significant computational gain while maintaining strong classification accuracy. Notably, in the breast cancer, flare solar, heart, and splice datasets, EBSVM demonstrates remarkable efficiency, significantly reducing KCVs compared to other models while maintaining competitive error rates. Looking at the overall average, excluding datasets where values were not reported, EBSVM reduces KCVs by 87.6% compared to QPSVM and 49.7% compared to VLPSVM, with just a minimal 0.7% and 0.71% increase in test error rates, respectively. When compared to Average (1) i.e., w.r.t SpSVM-2, EBSVM requires 1.2% fewer KCVs and achieves a 0.21% lower error rate. Against EESVM of Average (2), EBSVM reduces KCVs by 31.5% while also achieving a 2.4% lower test error rate. These findings confirm that EBSVM is a highly efficient approach, striking the right balance between computational cost and classification accuracy, making it a strongly desired classifier in the filed of modern machine learning.

$$\Rightarrow P(y'=-1|y=1) \;=\; \frac{P(y'=-1, y=1)}{P(y=1)} \qquad (26)$$

Where $P(y'=-1|y=1)$ is the conditional probability that the predicted class label y' is negative but the actual class label y is positive, $P(y'=-1, y=1)$ is the joint probability distribution of this predicted and actual class labels, $P(y=1)$ is the marginal probability distribution of positive class labels.

Using the above two terms, we can also find the total error probability during each detection as below

$$\begin{aligned} P(Error) &= P(Classified\,Class\;Label(x) \\ &\quad \neq Actual\;Class\;label(x)) \\ &= P(y' \neq y) = P(y'=1, y=-1) \\ &\quad + P(y'=-1, y=1) \\ &= P(y'=1|y=-1)P(y=-1) \\ &\quad + P(y'=-1|y=1)P(y=1) \\ &= \frac{Number\;of\;accepted\;Negative\;Pattern}{Total\;number\;of\;Input\;Negative\;Pattern} \\ &\quad \frac{Total\;Number\;of\;Negative\;Pattern}{Total\;number\;of\;Input\;Pattern} \\ &\quad + \frac{Number\;of\;Rejected\;Positive\;Pattern}{Total\;number\;of\;Input\;Positive\;Pattern} \\ &\quad \frac{Total\;Number\;of\;Positive\;Pattern}{Total\;number\;of\;Input\;Pattern} \\ &= \frac{\begin{array}{c}Number\;of\;accepted\;Negative\;Pattern \\ +Number\;of\;Rejected\;Positive\;Pattern\end{array}}{Total\;number\;of\;Input\;Pattern} \end{aligned} \qquad (27)$$

and accordingly, the total accuracy probability during each detection is

$$P(Accuracy) = 1 - P(Error)$$

However, in our evaluation, we measure the error rates and accuracy in percentages (%) as

$$Error\;Rate = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}[y_i \neq y_i'] \times 100$$

where N is the total number of classified patterns and **1** is the indicator function.

Table 2: Comparison of machines based on computational cost and training accuracy rate

	EBSVM	**LPSVM**	**QPSVM**
Computational Cost (CC)	0.01209	0.02204	0.1573
Sparsity	0.98791	0.97796	0.8427
Training Accuracy Rate(A)	0.9904	0.9903	0.9903

4.2.2 Machine accuracy cost (MAC)

A powerful yet cost-effective computing model is always really preferred. Considering this the objective is to design a discriminator that minimizes kernel evaluations during testing while preserving high accuracy. To strike a balance between an expensive machine with superior accuracy and a highly cost-effective system with similar efficiency, the concept of MAC [21] is introduced for kernel-based models, defined as follows:

$$MAC = \frac{Number\ \ of\ \ KCVs}{Test\ \ Accuracy} \tag{28}$$

A lower MAC value is always desirable, as it signifies that the machine attains optimal test accuracy with minimal kernel computations (KCVs). The values of MAC for different systems are provided in Table 3 in page 180.

4.2.3 GFR (Generalization failure rate)

We know that the optimal stochastic machine is designed by properly tuned from the bias-variance tradeoff and gives least generalization error ensuring that the model is not overfitting to the training data and can perform well on novel, unseen data. We have also tried to activate this on our model basing on some suitable mathematical formulations. However, as it is further sparsified from a sparser machine, its bridging capability between the training and new unseen data becomes more into interest. In this regard, we have used the term GFR [21] to assess the generalization deficiency of our machine as below, which is defined considering two points simultaneously: i) how badly the machine overfits, and ii) how poor it performs on test patterns:

$$Overfitting\ Tendency,\ OT = \frac{Test\ Error\ Rate - Train\ Error\ Rate}{Train\ Error\ Rate} \tag{29}$$

$$GFR = \frac{OT}{Test\ \ Accuracy} \tag{30}$$

Why less GFR or good generalization performance from our proposed EBSVM

As we have seen that $w = \sum \beta_j y_j \phi(x_j)$, for LP SVM, throwing KCVs with $\beta_j > 0$ gives less $||w||$, hence higher margin and reduced VC confidence, 2nd part of the RHS of the risk bound (4) becomes smaller. This usually increases the 1st part of the RHS of the risk bound (4), $R_{emp}[f]$ due to the reduced model complexity. But in our algorithm, we generate the bias value of our final discriminator by imitating the function value of the complex and non-reduced discriminator, which leads us to realize the final discriminator generated from an optimally further larger margin and sparser class based hypothesis where the hyperplane is shifted towards the complex model reducing the value of $R_{emp}[f]$ as well. Thus the overall value of the right hand side of (4) gets lower and we manage to get the machine having reduced upper bound of the actual risk that gives less generalization error. The values of GFR for different machines are provided in Table 4 on page 180.

5 Conclusion and future work

In this paper, we have discussed one of the most mathematically solid statistical machine learning approaches, Support Vector Machine (SVM) that gives decision under uncertainty depending on the given probabilistic data and their dependent stochastic parameters. We have gone through briefly using its two main mathematical bases, statistical theory and mathematical programming. Basing on these concepts, we have gone a bit further to produce a very efficient detector having both high classification accuracy and speed that is realized from further sparsification of Vapnik's LPSVM, which is indeed sparser itself. This proposed machine requires kernel computation upto as small as 0.6% of the sparse QPSVM, 1.5% of the sparser VLPSVM, 14.5% of the sparser EESVM, and 46.1% of the sparser SpSVM-2 by Keerthi et al [2], whereas in average 12.4% of the sparse QPSVM, 50.3% of the sparser VLPSVM, 68.1% of the sparser EESVM, and 98.1% of the sparser SpSVM-2 though produces very similar classification accuracy despite of being very straight forward by demanding very little training-effort. It also poses the least MAC (Machine Accuracy Cost, a term to assess the kernel load with respect to accuracy) value considering these machines while giving least GFR(Generalization failure rate) values compared to these SOTA machines as well, which proves its high generalization quality too.

Moreover, like other two most popular machines, QPSVM and LPSVM, our model's Training Accuracy shows almost no sensitivity for low and mid noise levels but sensitive towards higher noise levels. However, interestingly, while (Kernel) Computational cost by QPSVM shows sensitive throughout the whole noise levels, LPSVM and proposed EBSVM pose negligible sensitivity in this scenario. Further, while unlike many other prominent machine learning algorithms QPSVM and LPSVM offer heavy

Table 3: MAC (machine accuracy cost) table

Dataset	QPSVM	VLPSVM	EESVM	SpSVM-2	EBSVM
Banana	1.144	0.169	0.154	0.194	0.186
Breast Cancer	2.798	0.255	-	0.171	**0.111**
Diabetes	3.431	0.164	-	0.180	**0.116**
Flare Solar	9.025	3.732	-	0.127	**0.057**
German	5.750	0.348	0.364	0.186	0.256
Heart	0.818	0.266	0.088	0.051	0.088
Ringnorm	0.788	0.163	0.162	0.132	0.162
Splice	10.765	3.335	-	-	**3.233**
Titanic	1.921	1.088	0.510	0.043	0.074
Waveform	2.629	0.234	0.122	0.161	0.148

Table 3 presents the MAC values for the different machines, including the proposed algorithm. According to the results, the proposed algorithm completely outperforms all other classifiers in terms of MAC values for the Flare Solar, Diabetes, Breast Cancer, and Splice datasets while giving much smaller MAC values compared to the QPSVM, VLPSVM for all datasets. However, in case of other six datasets, it gives very similar MAC values compared to the other two further sparser machines, EESVM and SpSVM-2.This shows that compared to the reported classification algorithms, the suggested model achieves a more cost-efficient accuracy.

Table 4: Generalization failure rate (GFR) for different machines

Dataset	QPSVM	VLPSVM	EESVM	EBSVM
Banana	5.97×10^{-3}	6.14×10^{-3}	1.8×10^{-3}	2.18×10^{-3}
Breast Cancer	1.82×10^{-3}	2.89×10^{-3}	-	$\mathbf{1.23 \times 10^{-3}}$
Diabetes	1.36×10^{-3}	1.39×10^{-3}	-	$\mathbf{1.18 \times 10^{-3}}$
Flare Solar	-7.67×10^{-5}	-9.5×10^{-5}	-	-9.06×10^{-5}
German	1.44×10^{-3}	7.18×10^{-3}	9×10^{-4}	$\mathbf{8.7 \times 10^{-4}}$
Heart	1.27×10^{-3}	1.94×10^{-3}	1×10^{-3}	$\mathbf{8.98 \times 10^{-4}}$
Ringnorm	1.22×10^{-1}	1.03×10^{-2}	1.62×10^{-2}	2.21×10^{-2}
Splice	4.11×10^{-1}	9.42×10^{-0}	-	9.42×10^{-0}
Titanic	1.01×10^{-3}	1.49×10^{-3}	1×10^{-3}	1.59×10^{-3}
Waveform	1.1×10^{-2}	4.25×10^{-3}	1.8×10^{-3}	2.38×10^{-3}

Table 4 presents the GFR values of various algorithms, including the proposed EBSVM, alongside QPSVM, VLPSVM, and EESVM (SpSVM-2 is not reported here due to the lack of information). The results indicate that for the Breast Cancer, Diabetes, German, and Heart datasets, the proposed machine achieves lower GFR values compared to all other machines. However, for the remaining datasets, the proposed machine generally performs better in most cases.

sparsity (here, 84% and 98% respectively), our method gives the most (here, 99%) sparsity proving its unique performance in case of noisy data.

It is seen from the experiment that after our compact deterministic mapping of probabilistic data, contribution increases relatively higher compared to the confusion from the same KCVs. This supports our idea that by our mapping, topological spanning of the data shrinks without harming its stochastic nature countably and a basis set with comparatively less number of patterns becomes sufficient to represent the dataset. Hence relatively fewer number of KCVs can efficiently generate the discriminator machine having optimum generalization capability. It is also seen that in case of the outlier KCVs, contribution gets smaller than confusion and hence these misleading (playing undesired role in the final discriminator) or least significant outlier KCVs become potential to be thrown away from the basis set of the final discriminator for further sparsification of the sparser machine, which simultaneously helps to maintain high classification accuracy by reducing the probability of over-fitting and boost up the classification speed. Our method recovers the small scale gap that exists from SVM model which are:

i) It focuses on minimizing the error-depth instead of emphasizing the classification-accuracy directly.

ii) There is no complete certainty to find the best values of the regularization parameters from their range of continuous and real values through cross-validation process where a small deviation from the best values may lead to the solution that selects more patterns to be Support Vectors which drags the model towards over-fitting while increasing computational cost as well.

iii) Having a highly deviated and noise-contaminated heavy-outlier may change the SVM solution dramatically from the training set without having it. While some state-of-the-art machines pose the risk of decreasing classifica-

tion accuracy by sparsification from throwing some significant basis vectors, our model does not do so as its sparsification is done by following a smart stochastic mathematical analysis. By throwing outlier KCVs, our model reduces machine-misleading noise, gives a hypothesis from a class with less VC dimension, larger margin with reduced model complexity and VC confidence term in (4). Further, updating the bias term of our machine optimally, we have also managed to reduce the empirical risk from our model simultaneously. These jointly lowers the upper bound on the actual risk from our model that eventually produces a machine with excellent generalization performance, which is experimentally demonstrated from the GFR (Generalization Failure Rate) and Test Error Rate of our machine on benchmark data. However, a possible limitation of this method is that it may not show this high efficiency(sparsification) in case of datasets having very few outliers or noise.

In case of a highly complex and large dataset, a well generalized SVM is likely to get more outliers that are also included into the SVs set making the machine more complex and computationally expensive, which is efficiently tackled by our stochastic ξ analysis to make this machine optimally sparse and complex while in case of the dataset is less complex without having many outliers, our method will produce sparser solution with less number of KCVs naturally. This certainly proves the significance of our model.

Finally, while a classifier with high accuracy and speed like this proposed EBSVM is really indispensable to satisfy constraints in real-time, in future, we will try to make the KCV set κ-sparse, where κ will be a user defined variable.

References

[1] A. K. Kundu, R. Karim, and A. A. Ave, "Escalating svm-efficiency by sequential qp lp and slack variable analysis," *Journal of Computer Science*, vol. 19, no. 10, pp. 1253–1262, 2023. [Online]. Available: https://thescipub.com/abstract/jcssp.2023.1253.1262

[2] S. S. Keerthi, O. Chapelle, and D. DeCoste, "Building support vector machines with reduced classifier complexity," *Journal of Machine Learning Research*, vol. 7, pp. 1493–1515, 2006. [Online]. Available: https://www.jmlr.org/papers/v7/keerthi06a.html

[3] I. Steinwart, "Sparseness of support vector machines—some asymptotically sharp bounds," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. K. Saul, and B. Schölkopf, Eds., 2004, pp. 1069–1076. [Online]. Available: https://proceedings.neurips.cc/paper/2003/file/4c8c76b39d294759a9000cbda3a6571a-Paper.pdf

[4] T. Downs, K. E. Gates, and A. Masters, "Exact simplification of support vector solutions," *Journal of Machine Learning Research*, vol. 2, pp. 293–297, 2001. [Online]. Available: https://dl.acm.org/doi/10.5555/944790.944814

[5] T. Joachims and C.-N. J. Yu, "Sparse kernel svms via cutting-plane training," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, vol. 5781. Springer, 2009, pp. 47–62. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-04180-8_8

[6] A. Cotter, S. Shalev-Shwartz, and N. Srebro, "Learning optimally sparse support vector machines," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 266–274. [Online]. Available: https://proceedings.mlr.press/v28/cotter13.html

[7] S. Romdhani, P. Torr, B. Schölkopf, and A. Blake, "Efficient face detection by a cascaded support–vector machine expansion," *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 460, no. 2051, pp. 3283–3297, 2004. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rspa.2004.1333

[8] M. Wu, B. Schölkopf, and G. Bakır, "A direct method for building sparse kernel learning algorithms," *Journal of Machine Learning Research*, vol. 7, pp. 603–624, 2006. [Online]. Available: https://jmlr.org/papers/volume7/wu06a/wu06a.pdf

[9] M. Rätsch, S. Romdhani, G. Teschke, and T. Vetter, "Over-complete wavelet approximation of a support vector machine for efficient classification," in *Pattern Recognition: 27th DAGM Symposium*, ser. Lecture Notes in Computer Science, vol. 3663. Springer, 2005, pp. 351–360. [Online]. Available: https://link.springer.com/chapter/10.1007/11550518_44

[10] B. Heisele, T. Serre, S. Prentice, and T. Poggio, "Hierarchical classification and feature reduction for fast face detection with support vector machines," *Pattern Recognition*, vol. 36, no. 9, pp. 2007–2017, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0031320303000621

[11] R. Karim, M. Bergtholdt, J. H. Kappes, and C. Schnörr, "Greedy-based design of sparse two-stage svms for fast classification," in *Pattern Recognition: 29th DAGM Symposium*, ser. Lecture Notes in Computer Science, vol. 4713. Springer, 2007, pp. 395–404. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-540-74936-3_40

[12] K. Z. Arreola, J. Fehr, and H. Burkhardt, "Fast support vector machine classification using linear svms," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR)*, 2006, pp. 366–369.

[13] H. Sahbi and D. Geman, "A hierarchy of support vector machines for pattern detection," *Journal of Machine Learning Research*, vol. 7, pp. 2087–2123, 2006. [Online]. Available: https://jmlr.org/papers/v7/sahbi06a.html

[14] S. Maji, A. C. Berg, and J. Malik, "Efficient classification for additive kernel svms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 66–77, 2013. [Online]. Available: https://acberg.com/papers/mbm2012pami.pdf

[15] Ľubor Ladický and P. H. S. Torr, "Locally linear support vector machines," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 985–992. [Online]. Available: https://icml.cc/Conferences/2011/papers/508_icmlpaper.pdf

[16] Z. Xu, J. Gardner, S. Tyree, and K. Weinberger, "Compressed support vector machines," *arXiv preprint arXiv:1501.06478*, 2015. [Online]. Available: https://arxiv.org/abs/1501.06478

[17] J. Li and Y. Zhang, "Learning surf cascade for fast and accurate object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 3468–3475.

[18] V. C. Raykar, B. Krishnapuram, and S. Yu, "Designing efficient cascaded classifiers: Tradeoff between accuracy and cost," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '10. ACM, 2010, pp. 853–860. [Online]. Available: https://dl.acm.org/doi/10.1145/1835804.1835912

[19] R. Karim, A. K. Kundu, and A. A. Ave, "Sequential mathematical programming with ζ-analysis for hesvm," in *Mathematics and Computer Science: Contemporary Developments*. BP International, 2024, vol. 7, pp. 144–169. [Online]. Available: https://stm.bookpi.org/MCSCD-V7/article/view/16162

[20] Z. Fu, A. Robles-Kelly, and J. Zhou, "Mixing linear svms for nonlinear classification," *IEEE Transactions on Neural Networks*, vol. 21, no. 12, pp. 1963–1975, 2010. [Online]. Available: https://doi.org/10.1109/TNN.2010.2080319

[21] R. Karim and A. K. Kundu, "Efficiency and performance analysis of a sparse and powerful second order svm based on lp and qp," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 2, pp. 311–318, 2018. [Online]. Available: https://thesai.org/Publications/ViewPaper?Volume=9&Issue=2&Code=IJACSA&SerialNo=44

[22] ——, "Computational analysis to reduce classification cost keeping high accuracy of the sparser lpsvm," *International Journal of Machine Learning and Computing*, vol. 9, no. 6, pp. 728–733, 2019. [Online]. Available: https://www.ijmlc.org/vol9/865-L0294.pdf

[23] V. Franc and V. Hlaváč, "Greedy algorithm for a training set reduction in the kernel methods," in *Computer Analysis of Images and Patterns (CAIP)*, ser. Lecture Notes in Computer Science, vol. 2756. Springer, 2003, pp. 426–433. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-540-45179-2_53

[24] Q. Gu and J. Han, "Clustered support vector machines," in *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013, pp. 307–315. [Online]. Available: https://proceedings.mlr.press/v31/gu13b.html

[25] R. Karim, M. Hasan, A. K. Kundu, and A. A. Ave, "Lp svm with a novel similarity function outperforms powerful lp-qp-kernel-svm considering efficient classification," *Informatica*, vol. 47, no. 8, pp. 1–12, 2023. [Online]. Available: https://www.informatica.si/index.php/informatica/article/view/4767

[26] S. Zhou, "Sparse SVM for Sufficient Data Reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 4945–4958, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9415153

[27] D. Geebelen, J. A. K. Suykens, and J. Vandewalle, "Reducing the number of support vectors of svm classifiers using the smoothed separable case approximation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 4, pp. 682–688, 2012.

[28] H. Wang, Y. Shao, S. Zhou, C. Zhang, and N. Xiu, "Support Vector Machine Classifier via L0/1 Soft-Margin Loss," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 7253–7265, 2022.

[29] J. Liu, L.-W. Huang, Y.-H. Shao, W.-J. Chen, and C.-N. Li, "Nonlinear kernel support vector machine with 0-1 soft margin loss," *arXiv preprint arXiv:2203.00399*, 2022. [Online]. Available: https://arxiv.org/abs/2203.00399

[30] M. Wu, Z. Yang, and J. Ye, "Nonlinear Kernel-Free Quadratic Hyper-Surface Support Vector Machine with 0-1 Loss Function," *arXiv preprint arXiv:2404.10559*, 2024. [Online]. Available: https://arxiv.org/abs/2404.10559

[31] M. Akhtar, M. Tanveer, and M. Arshad, "RoBoSS: A robust, bounded, sparse, and smooth loss function for supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[32] R. Lin, Y. Yao, and Y. Liu, "Kernel support vector machine classifiers with ℓ_0-norm hinge loss," *Neurocomputing*, vol. 589, p. 127669, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0925231224004405

[33] H. Wang and Y. Shao, "Sparse and robust svm classifier for large scale classification," *Applied Intelligence*, vol. 53, no. 16, pp. 19 647–19 671, 2023. [Online]. Available: https://link.springer.com/article/10.1007/s10489-023-04511-w

[34] ——, "Fast truncated huber loss svm for large scale classification," *Knowledge-Based Systems*, vol. 260, p. 110074, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705122011674

[35] H. Wang, Z. Zhu, and Y. Shao, "Fast support vector machine with low-computational complexity for large-scale classification," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 54, pp. 4151–4163, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268772587

[36] H. Wang, Y. Shao, and S. Zhou, "Fast svm classifier for large-scale classification problems," *Information Sciences*, vol. 625, p. 119136, 2023.

[37] L. K. Hansen, "Stochastic linear learning: Exact test and training error averages," *Neural Networks*, vol. 6, no. 3, pp. 393–396, 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/089360809390006I

[38] R. Karim, "Theoretical analysis for exact results in stochastic linear learning," Master's thesis, Citeseer, 2004.

[39] V. Kecman, "Support vector machines—an introduction," in *Support Vector Machines: Theory and Applications*, ser. Studies in Fuzziness and Soft Computing, L. Wang, Ed. Springer, 2005, vol. 177, pp. 1–47. [Online]. Available: https://link.springer.com/chapter/10.1007/10984697_1

[40] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, no. 1, pp. 1–58, 1992. [Online]. Available: https://doi.org/10.1162/neco.1992.4.1.1

[41] I. Guyon, V. N. Vapnik, B. E. Boser, L. Bottou, and S. A. Solla, "Structural risk minimization for character recognition," *Advances in Neural Information Processing Systems*, pp. 471–479, 1992. [Online]. Available: https://proceedings.neurips.cc/paper/1991/file/10a7cdd970fe135cf4f7bb55c0e3b59f-Paper.pdf

[42] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*, 6th ed. John Wiley & Sons, 2021.

[43] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998. [Online]. Available: https://doi.org/10.1023/A:1009715923555

[44] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford: Clarendon Press, 1995. [Online]. Available: https://dl.acm.org/doi/10.5555/525960

[45] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995. [Online]. Available: https://doi.org/10.1007/978-1-4757-2440-0

[46] V. N. Vapnik, *Estimation of Dependences Based on Empirical Data*. Springer, 1979.

[47] V. Vapnik and A. Chervonenkis, "A note on one class of perceptrons," *Automation and Remote Control*, vol. 25, 1964.

[48] M. Awad and R. Khanna, "Support vector machines for classification," in *Efficient Learning Machines*. Berkeley, CA: Apress, 2015, pp. 39–66. [Online]. Available: https://doi.org/10.1007/978-1-4302-5990-9_3

[49] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers*, vol. EC-14, no. 3, pp. 326–334, 1965. [Online]. Available: https://doi.org/10.1109/PGEC.1965.264137

[50] M. Aizerman, E. Braverman, and L. Rozonoer, "Theoretical foundations of the potential function method in pattern recognition learning," *Automat. Remote Control*, vol. 25, pp. 821–837, 1964. [Online]. Available: https://cs.uwaterloo.ca/ y328yu/classics/kernel.pdf

[51] K.-L. Du, B. Jiang, J. Lu, J. Hua, and M. N. S. Swamy, "Exploring kernel machines and support vector machines: Principles, techniques, and future directions," *Mathematics*, vol. 12, no. 24, p. 3935, 2024. [Online]. Available: https://doi.org/10.3390/math12243935

[52] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral equations," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 209, pp. 415–446, 1909. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rsta.1909.0016

[53] R. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton University Press, 1957.

[54] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American Mathematical Society*, vol. 68, pp. 337–404, 1950.

[55] F. Girosi, "An equivalence between sparse approximation and support vector machines," Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Tech. Rep. AIM-1606, 1997. [Online]. Available: https://dspace.mit.edu/handle/1721.1/7289

[56] N. E. Heckman, "The theory and application of penalized least squares methods or reproducing kernel hilbert spaces made easy," Department of Statistics, University of British Columbia, Tech. Rep. Technical Report 216, 1997. [Online]. Available: https://www.stat.ubc.ca/technical-reports-archive/doc/216.pdf

[57] G. Wahba, *Spline Models for Observational Data*, ser. CBMS-NSF Regional Conference Series in Applied Mathematics. Philadelphia: Society for Industrial and Applied Mathematics, 1990, vol. 59. [Online]. Available: https://epubs.siam.org/doi/book/10.1137/1.9781611970128

[58] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004. [Online]. Available: https://doi.org/10.1017/CBO9780511804441

[59] V. N. Vapnik, *Statistical Learning Theory*. Wiley, 1998. [Online]. Available: https://www.wiley.com/en-us/Statistical+Learning+Theory-p-9780471030034

[60] A. Nefedov, J. Ye, C. Kulikowski, I. Muchnik, and K. Morgan, "Experimental study of support vector machines based on linear and quadratic optimization criteria," *DIMACS Technical Report 2009-18*, 2009. [Online]. Available: https://doi.org/10.1109/icmla.2009.52

[61] T. Suttorp and C. Igel, "Multi-objective optimization of support vector machines," pp. 199–220, 2006. [Online]. Available: https://doi.org/10.1007/11399346_9

[62] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge: Cambridge University Press, 2000. [Online]. Available: https://doi.org/10.1017/CBO9780511801389

[63] A. Tharwat, A. E. Hassanien, and B. E. Elnaghi, "A ba-based algorithm for parameter optimization of support vector machine," *Pattern Recognition Letters*, vol. 93, pp. 13–22, 2016.

[64] S. R. Gunn, "Support vector machines for classification and regression," School of Electronics and Computer Science, University of Southampton, Tech. Rep. ISIS Technical Report, 1998. [Online]. Available: https://eprints.soton.ac.uk/256459/

[65] G. Rätsch. Benchmark datasets. Re-packaged and available on GitHub. [Online]. Available: https://github.com/tdiethe/gunnar_raetsch_benchmark_datasets

[66] J. Kools, "6 functions for generating artificial datasets," 2013. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/41459-6-functions-for-generating-artificial-datasets

https://doi.org/10.31449/inf.v49i28.6873 Informatica **49** (2025) 185–196 **185**

Comparison of NSGA-II and Ant Colony Optimization for Solving the Multiobjective Vehicle Routing Problem with Flexible Time Windows

Labdiad Fatah
LIPIM, ENSA Khouribga, University Sultan Moulay Slimane, Morocco
E-mail: abdelfettahlabdiad@gmail.com

Keywords: Vehicle routing problem (VRP), flexible time windows (VRPFlexTW), nondominated sorting genetic algorithm II (NSGA-II), ant colony optimizer (ACO)

Received: August 8, 2024

Vehicle routing problems are widely encountered in real-world applications. This paper addresses a specific variant known as the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), where solutions must comply with constraints, including travel, service, and waiting times, along with time-window restrictions. We propose the Nondominated Sorting Genetic Algorithm II (NSGA-II) and detail its components. Additionally, we provide a computational comparison between NSGA-II and the Ant Colony Optimizer (ACO) for several instances of VRPFlexTW. This comparison aims to evaluate the efficiency and performance of these approaches in solving this complex problem. Finally, the experimental results demonstrate that NSGA-II significantly improves solution quality and reduces the optimal fleet size, establishing it as the most effective algorithm among those presented. The results reveal that the NSGA-II algorithm consistently outperforms ACO and ALNS across all tested client configurations. NSGA-II achieves the lowest cost function values, demonstrating superior cost optimization by significantly reducing the total routing costs compared to ACO and ALNS

Povzetek: Članek primerja NSGA-II in ACO pri VRPFlexTW. Znanstvena novost je večkriterijska formulacija s fleksibilnimi časovnimi okni ter sistematična primerjava algoritmov. NSGA-II dosega bolj kvalitetno iskanje kompromisov kot metoda ACO/ALNS v različnih instancah.

1 Introduction

Logistics companies constantly encounter the challenge of efficiently delivering goods to customers while meeting specific service requirements. These requirements are crucial in scenarios resembling the Vehicle Routing Problem (VRP). This problem is common in various sectors such as banking, postal services, and school transportation [7]. The Vehicle Routing Problem (VRP) involves finding the optimal set of routes originating from one or multiple depots to a specified set of dispersed locations while adhering to constraints prioritizing factors such as cost, time, distance, or a combination thereof. Essentially an extension of the Travelling Salesman Problem [8], the VRP was first conceptualized as the "Truck Dispatching Problem" by [9] and has since been extensively studied in both its formulation and resolution. Today, VRP plays a pivotal role across various domains, including physical distribution, logistics, supply chain management, and finance. The literature on VRP encompasses a wide array of variations and methodologies [10, 11, 12]. In its simplest form, VRP entails a fleet of vehicles stationed at a central depot, responsible for servicing multiple customers who have placed orders or requests. Each vehicle's journey, starting and concluding at the depot, constitutes a tour that must visit each customer exactly once and ensure that every customer is served by precisely one vehicle. The primary objective of the standard VRP model is to minimize the total travel distance or time across all vehicle routes while fulfilling customer demands.

The model can be represented as a closed graph $G = (V, A)$ [27, 30] , where vertices represent clients $V = \{0, 1, \ldots, n\}$ with 0 indicating the depot, and arcs (i, j) denote routes connecting two clients. There are m binary variables x_{ijp} used to indicate whether route (i, j) is traveled by vehicle p ($x_{ijp} = 1$ if traveled, $x_{ijp} = 0$ otherwise). Another binary variable y_{ip} ensures each client is served by exactly one vehicle; hence, $y_{ip} = 1$ if vehicle p visits client i, and $y_{ip} = 0$ otherwise. The mathematical model can be formulated as follows:

$$Z = \min F, \tag{1}$$

$$\text{s.c.} \quad \sum_{p=1}^{m} \sum_{i=1}^{n-1} x_{0ip} \leq m, \tag{2}$$

$$\sum_{p=1}^{m} \sum_{i=1}^{n-1} x_{i0p} \leq m, \tag{3}$$

$$\sum_{p=1}^{m} y_{kp} \leq 1, \forall k = 1, \ldots, n, \tag{4}$$

$$\sum_{j=1}^{n-1} x_{ijp} = y_{ip}, i = \{1, \ldots, n\}, p = \{1, \ldots, m\}, \quad (5)$$

$$\sum_{j=1}^{n} x_{jip} = y_{ip}, i = \{1, \ldots, n\}, p = \{1, \ldots, m\}, \quad (6)$$

$$x_{ijp}, y_{ip} \in \{0, 1\}, \forall i, j = \{1, \ldots, n\}, p = \{1, \ldots, m\}. \quad (7)$$

The constraints (1) and (3) ensure that the number of vehicles leaving the depot is the same as the number of vehicles entering the depot. Constraint (4) ensures that each city from 1 to n is visited by at most one vehicle. Constraints (5) and (6) represent the conservation of flow for each city i, ensuring that the number of vehicles crossing all arcs entering $\{(j, i), \forall j \in A\}$ is equal to the number crossing the outgoing arcs $\{(i, j), \forall j \in A\}$. Finally, the binary variables x_{ijp} and y_{ip} are defined by constraint (7).

The vehicle routing problem extends to the classic traveling salesman problem, which belongs to the NP-complete class of problems. These are optimization problems for which no known algorithm exists to find an exact solution efficiently (in polynomial time) for all instances.

In practical applications, the VRP often incorporates various additional constraints, such as limits on vehicle capacity [17], time windows for customer service [13, 14], restrictions on route lengths, or constraints on driver or distribution clerk work hours. Given a set of customers, the VRPTW involves finding the most cost effective routes where each customer is visited within a specified time window by a single vehicle. Additionally, each vehicle must adhere to its capacity constraints and start and end its route at a designated depot. Vehicles can arrive before the time window opens and can wait at no cost until service is available, but they cannot arrive after the time window closes [18]. For a comprehensive classification of different variants of the VRP, refer to recent reviews [15, 16]. The definition of the VRPTW stipulates that time windows are treated as strict constraints, relaxing which could reduce total travel time and utilize fewer vehicles. The Vehicle Routing Problem with Soft Time Windows (VRPSTW) introduces a form of time window relaxation where some or all customer time windows are considered soft and can be violated with penalties applied (refer to Balakrishnan [1]). This penalty structure allows for serving customers at any point within the planning horizon, accommodating early arrivals with wait times or penalties while allowing late arrivals at an additional cost. Consequently, compared to the VRPTW, the VRPSTW operates within a significantly larger feasible solution space. Like the basic VRP, most variants are known to be NP-hard. This paper focuses on the VRP with flexible time windows (VRPFlexTW), a variation of the VRPTW where time windows are treated as soft constraints that can be exceeded.

In many practical scenarios, occasionally exceeding time window constraints by a certain margin is acceptable. Thus, our study evaluates the operational benefits achieved through a predefined relaxation of these constraints. Specifically, we investigate the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), where vehicles can deviate from customer time windows within a specified tolerance. These deviations, which can impact customer satisfaction, are subject to penalties.

The structure of this paper unfolds as follows: The subsequent section aims to introduce the multi-objective modeling of the VRPFlexTW problem and provides an extensive overview of prevalent resolution techniques from existing literature. Section 2 will introduce two adapted versions of ALNS Algorithms and their components applied to the VRPFLexTW problem. Section 3 will present numerical findings, comparing them with conventional methods such as ant colony optimization and standard ALNS. Finally, the paper concludes with a summary of the results and a discussion of this study.

2 Literature review

The Vehicle Routing Problem (VRP) is a classic and extensively studied combinatorial optimization problem in operations research and logistics. Originating from the seminal work of Dantzig and Ramser (1959) [23] on the truck dispatching problem, VRP involves determining optimal routes for a fleet of vehicles to deliver goods or provide services to a set of customers, typically starting and ending at a central depot [23]. The primary objective is to minimize total travel distance, time, or costs while satisfying various constraints, such as vehicle capacity limitations, time windows for customer visits, and operational costs associated with vehicle usage.

Over the decades, VRP has evolved into several variants to address specific real-world constraints and requirements. One of the most common variants is the Vehicle Routing Problem with Time Windows (VRPTW), introduced by Solomon (1987), which imposes time windows within which customers must be serviced. This constraint adds complexity by requiring routes to respect customer availability times while optimizing vehicle utilization [24]. Another significant variant is the Capacitated VRP (CVRP), where each vehicle has a limited capacity that cannot be exceeded. The CVRP was formalized by Clarke and Wright (1964) and remains a fundamental model in transportation logistics, influencing subsequent developments in algorithmic approaches and solution methods [25].

The Vehicle Routing Problem with Time Windows (VRPTW) and its variant, the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), are fundamental challenges in logistics optimization. VRPTW aims to efficiently schedule vehicle routes to serve customers within predefined time windows while minimizing operational costs [18]. In contrast, VRPFlexTW relaxes these constraints, allowing vehicles to arrive before time windows open and wait without penalties, thereby enhancing operational flexibility [19]. This flexibility is crucial in dy-

namic environments where strict adherence to fixed schedules could be more practical.

In recent years, extensive research has been devoted to the vehicle routing problem with time windows (VRPTW), resulting in a variety of algorithms designed to tackle it. These can be broadly classified into three categories: exact algorithms [4, 3, 20], and heuristic algorithms [1, 2, 29].

Exact algorithms tackle problems by developing precise mathematical models, formulating the problem rigorously, and designing algorithms to compute optimal solutions using mathematical principles. In [4], the authors utilized column generation to determine the shortest paths in VRPTW. Additionally, branch and bound [3] and branch and price techniques have been applied to VRPTW. However, exact algorithms, which systematically evaluate all feasible solutions to find the optimal one, experience a rapid increase in search space and computational complexity as the number of customers in VRPTW grows. Furthermore, these methods are often less efficient for multiobjective VRPTW (MOVRPTW).

Heuristic algorithms are widely adopted for addressing VRPTW. For single objective VRPTW (SOVRPTW), prevalent methods include genetic algorithms (GA), particle swarm optimization (PSO), and tabu search (TS), among others. Additionally, a genetic algorithm incorporating an insertion heuristic was developed for VRPTW, as described in [2]. Recent research has explored advanced metaheuristic approaches such as Genetic Algorithms (GA), Ant Colony Optimization (ACO), and hybrid methods to address these challenges effectively [20, 21, 22]. Such methodologies are vital in optimizing logistics operations, from urban delivery services to emergency response logistics, by balancing computational efficiency with solution quality and adaptability. The table 1 is a summary on the Vehicle Routing Problem (VRP) and its variants.

3 Problem formulation of the VRPFlexTW

The Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW) is an optimization challenge that plans efficient routes for vehicles to serve customers while balancing cost minimization and customer satisfaction. Unlike traditional vehicle routing, VRPFlexTW allows vehicles to arrive early or late to customer locations within certain limits, incurring penalties only for excessive deviations. This section presents a layman's explanation of the problem for readers unfamiliar with technical notation, followed by the formal mathematical formulation.

3.1 Layman's explanation of VRPFlexTW

The VRPFlexTW involves scheduling a fleet of vehicles to deliver goods or services to customers, starting and ending at a central depot (e.g., a warehouse). The goal is to keep costs low (by minimizing travel distance and the number of vehicles used) while ensuring customers are served at times that maximize their satisfaction. Each customer has a preferred time window for service (e.g., 9 AM to 11 AM), but vehicles can arrive earlier or later within a broader allowable range, with customer satisfaction decreasing if service occurs outside the preferred window [28].

Here's a step-by-step breakdown of the key components:

1. **What We Decide (Decision Variables):**
   - **Routes**: Which vehicle travels from one customer to another, and in what order. For example, does Vehicle 1 go from Customer A to Customer B, or to Customer C first?
   - **Customer Assignment:** Which vehicle serves each customer. Each customer must be served by exactly one vehicle.
   - **Timing:** When a vehicle arrives at and serves each customer. This includes deciding if the vehicle waits if it arrives early.
2. **Rules to Follow (Constraints):**
   - Serve Every Customer: Each customer must be visited exactly once by one vehicle.
   - Start and End at Depot**: Every vehicle begins at the depot, visits its assigned customers, and returns to the depot.
   - **Vehicle Capacity:** Vehicles have a limited capacity (e.g., a maximum weight of goods they can carry), and the total demand of customers served by a vehicle cannot exceed this limit.
   - **Flexible Time Windows:** Customers have a preferred time window (e.g., 9 AM to 11 AM) where satisfaction is highest. Vehicles can arrive within a broader range (e.g., 8 AM to 12 PM), but satisfaction decreases linearly if service occurs before or after the preferred window, reaching zero outside the allowable range.
   - Travel and Service Times**: The time to travel between customers, serve them, and wait (if arriving early) must be accounted for, ensuring routes are feasible within a maximum route duration.
   - **Waiting:** If a vehicle arrives before the preferred time window, it can wait at no cost, but waiting time is calculated to ensure accurate scheduling.
3. **Goals (Objectives):**
   - **Maximize Customer Satisfaction:** Ensure as many customers as possible are served within their preferred time windows to keep satisfaction high.
   - **Minimize Costs:** Reduce the total distance traveled by vehicles and the number of vehicles used to lower transportation and operational costs.

Table 1: Summary of vehicle routing problem (VRP) variants and their characteristics

VRP Variant	Description	Primary Objective	Key Constraints	Citations
VRP	Determines optimal routes for vehicles from a depot to customers.	Minimize travel distance, time, or costs.	Routes start/end at depot; customer service requirements.	Dantzig & Ramser (1959) [23]
CVRP	VRP with limited vehicle capacity.	Minimize travel distance/costs with capacity constraints.	Vehicle capacity limits; all customers served.	Clarke & Wright (1964) [25]
VRPTW	VRP with customer service within specific time windows.	Minimize costs while meeting time constraints.	Strict time windows; vehicle capacity.	Solomon (1987) [24], Bräysy & Gendreau (2005) [18]
VRPFlexTW	VRPTW with flexible time windows, allowing early arrivals and waiting.	Minimize costs with scheduling flexibility.	Flexible time windows; vehicles can wait.	Sharma et al. (2018) [19], Toth & Vigo (2002) [20], Wagemaker et al. (2016) [21], Suarez et al. (2016) [22]

This setup makes VRPFlexTW ideal for real-world scenarios like delivery services, where flexibility in timing can improve efficiency while maintaining customer happiness.

3.2 Mathematical formulation

The VRPFlexTW is formulated as a multi-objective optimization problem on a directed graph $G = (V, A)$, where $V = \{0, 1, \ldots, n\}$ includes the depot (vertex 0) and n customers, and $A = \{(i, j) \mid i, j \in V, i \neq j\}$ represents possible travel routes. Each arc (i, j) has a distance c_{ij} and travel time t_{ij}. Each customer $i \in \{1, \ldots, n\}$ has a demand d_i, service time s_i, and a preferred time window $[a_i, b_i]$, extended to a flexible window $[a_i - a'_i, b_i + b'_i]$, where a'_i and b'_i satisfy $a_i - a'_i \geq E_i$ (earliest allowable time) and $b_i + b'_i \leq L_i$ (latest allowable time). Customer satisfaction $\mu_i(z_i)$, where z_i is the service start time, is defined as:

$$\mu_i(z_i) = \begin{cases} 0, & z_i < E_i \\ \frac{z_i - E_i}{a_i - E_i}, & E_i \leq z_i < a_i \\ 1, & a_i \leq z_i \leq b_i \\ \frac{L_i - z_i}{L_i - b_i}, & b_i < z_i \leq L_i \\ 0, & z_i > L_i \end{cases}$$

Parameters:

- h_k: Transportation cost per unit distance for vehicle k.
- f_k: Fixed cost for using vehicle k.
- c_{ij}: Distance between vertices i and j.
- s_i: Service time at vertex i.
- w_i: Waiting time at vertex i.
- t_{ij}: Travel time from vertex i to vertex j.
- r_k: Maximum route duration for vehicle k.
- m: Number of available vehicles.

Decision Variables:

- x_{ijk}: Binary variable, 1 if vehicle k travels from vertex i to vertex j, 0 otherwise.
- y_{ik}: Binary variable, 1 if vehicle k serves vertex i, 0 otherwise.
- z_i: Continuous variable, the time when service begins at vertex i.

Objective Functions:

$$\max \quad \frac{1}{n} \sum_{i=1}^{n} \mu_i(z_i), \tag{5}$$

$$\min \quad \sum_{k=1}^{m} h_k \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ijk} + \sum_{k=1}^{m} f_k \sum_{j=1}^{n} x_{0jk}. \tag{6}$$

Constraints:

$$\sum_{i=0}^{n} x_{ijk} = y_{jk}, \quad \forall k \in \{1, \ldots, m\}, \quad \forall j \in \{1, \ldots, n\}, \tag{7}$$

$$\sum_{j=0}^{n} x_{ijk} = y_{ik}, \quad \forall k \in \{1, \ldots, m\}, \quad \forall i \in \{1, \ldots, n\}, \tag{8}$$

$$\sum_{i=0}^{n} \sum_{j=0}^{n} x_{ijk}(t_{ij} + s_i + w_i) \leq r_k, \quad \forall k \in \{1, \ldots, m\}, \tag{9}$$

$$w_0 = s_0 = 0, \tag{10}$$

$$\sum_{k=1}^{m} \sum_{i=0}^{n} x_{ijk}(z_i + w_i + s_i + t_{ij}) = z_j, \quad \forall j \in \{1, \ldots, n\}, \tag{11}$$

$$E_i \leq z_i + w_i \leq L_i, \quad \forall i \in \{1, \ldots, n\}, \tag{12}$$

$$w_i = \max\{0, E_i - z_i\}, \quad \forall i \in \{1, \ldots, n\}, \tag{13}$$

$$x_{ijk} \in \{0, 1\}, \quad \forall i, j \in \{0, \ldots, n\}, \quad \forall k \in \{1, \ldots, m\}, \tag{14}$$

$$y_{ik} \in \{0, 1\}, \quad \forall i \in \{1, \ldots, n\}, \quad \forall k \in \{1, \ldots, m\}, \tag{15}$$

$$z_i \geq 0, \quad \forall i \in \{1, \ldots, n\}. \tag{16}$$

The first objective (5) maximizes average customer satisfaction based on service timing. The second objective (6) minimizes total costs, including travel distance and vehicle usage. Constraints (7) and (8) ensure each customer

is served exactly once by one vehicle. Constraint (9) limits the total time per route. Constraint (10) sets zero waiting and service time at the depot. Constraint (11) tracks arrival times. Constraint (12) enforces flexible time windows. Constraint (13) defines waiting times for early arrivals. Constraints (14)–(16) specify variable domains.

4 Multi-objective ALNS techniques for VRPFlexTW

The application of ALNS in multi-objective combinatorial optimization problems was first introduced by Schaus and Hartert [26], who emphasized a search process focused on non-dominated solutions. This algorithm has proven to be an effective method for tackling complex neighborhoods in tightly constrained problems, where searching small neighborhoods often results in the algorithm getting stuck in local optima. By exploring more prominent neighborhoods, the algorithm enhances the probability of finding superior solutions, leveraging a variety of destroy and reconstruct methods to form an efficient adaptive search procedure that balances intensification and diversification. The primary process of the multi-objective ALNS algorithm is depicted as follows:

Algorithm 1 Steps of the MOALNS Algorithm

```
Initialize feasible solution x
Set x* ← x
Insert x to feasible solution set
Initialize adaptive weights
while the stopping criteria is not reached do
    Select a pair of destruction and reconstruction
    heuristics d_i, r_i based on the adaptive weights
    Apply d_i and r_i to yield a new solution x'
    if x' can be accepted then
        Add x' to the feasible solution set
        if x' is better than x* then
            Set x* ← x'
        end if
        if x' is a non-dominated solution then
            Insert x' to Pareto set A
            Update A
        end if
    end if
    Randomly select x from A
    Update the adaptive weights
end while
return x*
```

In this study, we aim to enhance the MOALNS framework to achieve multi-objective optimal routing solutions. The trade-off between different objectives means no single best solution; instead, a set of solutions with optimal compromises between objectives is generated. Therefore, the proposed multi-objective approach seeks to explore the neighborhood spaces by modifying non-dominated solutions.

While ALNS is effective for VRPFlexTW, it is outperformed by NSGA-II across all tested instances, as shown in Tables 2 and 3. For example, in the 100-client instance, ALNS achieves a routing cost of 1640 and requires 10 vehicles, compared to NSGA-II's cost of 1405 and 10 vehicles. For the 1000-client instance, ALNS's cost is 85904 with 93 vehicles, while NSGA-II achieves 83761 with 90 vehicles. These results highlight NSGA-II's superior ability to optimize both objectives due to its population-based approach, which maintains a diverse set of solutions and leverages crossover and mutation to explore the solution space efficiently.

ALNS's primary limitation lies in its scalability for larger instances. As the number of clients increases (e.g., from 100 to 1000), the search space grows exponentially, leading to a more extensive and time consuming local search. This is evident in the increased routing costs and fleet sizes for larger instances, where ALNS struggles to converge to near optimal solutions within reasonable computational time. The adaptive mechanism, while effective for small to medium instances, becomes less efficient as the number of possible customer assignments and route combinations grows, requiring more iterations to escape local optima. In contrast, NSGA-II's population-based structure allows it to handle larger instances more effectively by maintaining diversity and avoiding premature convergence, as noted by Vidal et al. [5].

Additionally, ALNS's reliance on sequential destroy and repair operations makes it sensitive to the initial solution quality and operator selection strategy. For VRPFlexTW, the flexible time windows introduce additional complexity, as the algorithm must balance customer satisfaction with cost minimization, which increases computational overhead. This contrasts with NSGA-II's ability to simultaneously evaluate multiple trade-offs in its Pareto front, making it more robust for multi-objective optimization.

5 NSGA-II techniques for VRPFlexTW

This study employs the Nondominated Sorting Genetic Algorithm II (NSGA-II) to solve the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW). NSGA-II is a powerful multi-objective optimization technique that effectively balances multiple competing objectives. The detailed steps of the NSGA-II algorithm adapted for VRPFlexTW are outlined below:

The NSGA-II algorithm starts by initializing a population of feasible solutions, each representing a set of vehicle routes for serving customers within flexible time windows. Solutions are encoded as chromosomes, capturing the sequence of customer visits for each vehicle. The algorithm evaluates each solution based on multiple objectives, such as minimizing total travel distance, reducing the

Algorithm 2 NSGA-II for VRPFlexTW

```
Initialize population P with feasible solutions
Evaluate objective functions for each solution in P
Perform nondominated sorting on P
Calculate crowding distance for each solution in each front
while stopping criteria not met do
    Select parent solutions from P using binary tournament selection
    Apply crossover and mutation operators to generate offspring Q
    Evaluate objective functions for each solution in Q
    Combine parent population P and offspring population Q into R
    Perform nondominated sorting on R
    Calculate crowding distance for each solution in each front
    Select the top solutions from R to form the next generation P
end while
Return the final Pareto front from the population P
```

number of vehicles, and maximizing customer satisfaction while ensuring compliance with constraints like vehicle capacity and time windows.

Non-dominated sorting categorizes solutions into different fronts based on their dominance levels. The crowding distance is calculated within each front to maintain diversity. Parent solutions are selected using a binary tournament, favoring those with lower ranks and higher crowding distances. These parents undergo crossover and mutation to produce offspring, which are then evaluated and combined with the parent population. The combined population undergoes another round of nondominated sorting, and the top solutions are selected to form the next generation. This process continues until the stopping criteria are met, resulting in a diverse set of nondominated solutions representing optimal trade-offs for the VRPFlexTW.

6 The ant colony optimization (ACO) techniques for VRPFlexTW

The Ant Colony Optimization (ACO) approach for the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW) leverages the behavior of ant colonies in nature to effectively explore and optimize complex routing solutions [31, 6, 32]. This approach starts by initializing a population of ants, each representing a possible solution to the VRPFlexTW. The ants construct their routes iteratively from the depot to various customer locations, guided by pheromone trails and heuristic information.

Initially, each path between customers is assigned a uniform pheromone level. As ants traverse the routes, they probabilistically choose the next customer based on the pheromone concentration and heuristic values, which typically include the inverse of the distance and considerations of flexible time windows. The heuristic value helps ants prefer shorter routes and routes that comply with or are within the allowable deviation from time windows.

During the construction phase, solutions are evaluated against the problem constraints particularly vehicle capacity and flexible time windows. Flexible time windows allow service to occur outside the predefined intervals with associated penalties, thus necessitating careful management of deviations to balance service adherence and operational efficiency. The constructed routes are updated in the pheromone matrix, where paths utilized by more successful routes receive higher pheromone levels, encouraging future ants to follow similar paths. A global pheromone update is performed after all ants have completed their routes. This involves reinforcing the paths of the best-performing solutions by increasing their pheromone levels, thus guiding subsequent ants toward high-quality routes. Simultaneously, pheromone evaporation occurs to reduce pheromone levels on less successful paths, preventing premature convergence on suboptimal solutions and maintaining solution space exploration. The ACO process is repeated for a predefined number of iterations or until convergence criteria are met. Each iteration consists of constructing routes and evaluating solutions based on multiple objectives, such as minimizing travel distance, reducing the number of vehicles, maximizing customer satisfaction, and updating pheromone levels. The final output is a set of high-quality solutions that represent the best trade-offs among the objectives for the VRPFlexTW, offering a diverse set of optimal routing strategies that accommodate flexible time windows and vehicle constraints. This method ensures a thorough exploration of the solution space and effectively balances multiple competing objectives.

7 Numerical results

We conducted a series of computational experiments to evaluate the performance of NSGA-II for the VRPFlexTW. The algorithm was tested on a set of small instances drawn from the benchmarks established by Solomon (1987) and further extended by Gehring and Homberger (1999). Specifically, we utilized the Solomon set R, which includes randomized customer locations, and applied the NSGA-II algorithm to various problem sizes from Solomon's benchmark to assess its effectiveness.

8 Experimental setup

This section outlines the experimental setup used to evaluate the performance of the Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Ant Colony Optimization (ACO) for solving the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW). The algorithms were

Algorithm 3 ACO for VRPFlexTW

```
Initialize pheromone matrix τ
Set parameters α, β, ρ, number of ants, and iterations
for each iteration do
    for each ant do
        Initialize ant at depot
        while ant has not visited all customers do
            Select next customer based on pheromone and heuristic information
            Move to selected customer and update solution
            Apply local pheromone update
        end while
        Evaluate ant's solution based on multi-objective criteria
    end for
    Apply pheromone evaporation
    Apply global pheromone update based on best solutions
end for
Return best solution(s)
```

tested on benchmark instances from Solomon (1987) and Gehring and Homberger (1999), specifically the Solomon set R with randomized customer locations, covering problem sizes from 100 to 1000 clients. All experiments were conducted using Python, compiled with the Intel compiler, on a Celeron 1.80 GHz core i5 processor with 8GB RAM. Each algorithm was executed for 15,600 iterations, with NSGA-II applied ten times per instance to ensure robust results. Below, we detail the parameter tuning process for NSGA-II and ACO, including population sizes, mutation rates, crossover strategies, and other hyperparameters, to facilitate replication and highlight the trade-offs in their optimization.

8.1 Benchmark instances

The experiments utilized the Solomon set R, which features randomized customer locations, making it suitable for testing algorithmic robustness across diverse scenarios. The instances range from 100 to 1000 clients, with customer demands, service times, and flexible time windows defined as in Section 3. The two objectives maximizing average customer satisfaction (Equation 5) and minimizing total routing costs (Equation 6) were evaluated, with performance metrics including routing costs (Table 2), fleet size (Table 3), and solution diversity (Table 4).

8.2 Parameter tuning for NSGA-II

NSGA-II, a population-based multi-objective genetic algorithm, requires careful tuning of hyperparameters to balance exploration and exploitation in the solution space. The following parameters were optimized through a grid search approach, testing multiple configurations on a subset of Solomon's 100-client and 400-client instances to ensure generalizability across problem sizes.

- **Population Size**: We tested population sizes of 50, 100, 150, and 200. A population size of 100 was selected as it provided a good balance between solution diversity and computational efficiency. Smaller populations (e.g., 50) led to premature convergence, reducing the Pareto front spread, while larger populations (e.g., 200) increased computation time without significant improvements in cost or satisfaction metrics, as observed in preliminary runs (average cost difference <2% for 100 vs. 200).
- **Crossover Strategy**: We employed simulated binary crossover (SBX) with a crossover probability of 0.9, tested against values of 0.7, 0.8, and 1.0. SBX was chosen for its ability to generate diverse offspring by mimicking continuous variable crossover in a binary context, suitable for VRPFlexTW's route assignments. The crossover probability of 0.9 maximized diversity while maintaining convergence speed, as lower values (e.g., 0.7) resulted in slower exploration of the solution space.
- **Mutation Rate**: Polynomial mutation was applied with a mutation probability of 0.1, tested against 0.05, 0.1, and 0.2. A rate of 0.1 provided sufficient perturbation to avoid local optima while preserving high-quality solutions. Higher rates (e.g., 0.2) disrupted convergence, increasing costs by up to 5% in the 400-client instance, while lower rates (e.g., 0.05) led to insufficient diversity.
- **Selection Mechanism**: Tournament selection with a tournament size of 2 was used to select parents, balancing selection pressure and diversity. Alternatives like roulette wheel selection were tested but resulted in slower convergence due to less elitist selection.
- **Elitism**: NSGA-II's non-dominated sorting ensured elitism by preserving the best solutions across generations. We tuned the crowding distance parameter to prioritize solutions with greater diversity, enhancing the Pareto front spread (Table 4).

The tuning process involved 20 runs per configuration, evaluating the average cost function value and Pareto front spread after 15,600 iterations. The selected parameters (population size 100, crossover probability 0.9, mutation probability 0.1) optimized both objectives, achieving costs of 1405 for 100 clients and 83761 for 1000 clients, with stable convergence (Figure 3).

8.3 Parameter tuning for ACO

ACO, a pheromone-based metaheuristic, was tuned to optimize VRPFlexTW's objectives by adjusting parameters affecting pheromone trails and solution construction. The tuning process used the same grid search approach on the 100-client and 400-client instances.

– **Ant Colony Size**: We tested colony sizes of 20, 50, and 100 ants. A size of 50 was chosen, as it balanced exploration (constructing multiple solutions per iteration) and computational cost. Smaller colonies (e.g., 20) led to insufficient exploration, increasing costs by 8% for the 400-client instance, while larger colonies (e.g., 100) increased runtime without proportional improvements.

– **Pheromone Update Parameters**: The pheromone evaporation rate (ρ) was set to 0.1 after testing 0.05, 0.1, and 0.2. A rate of 0.1 prevented premature convergence to suboptimal solutions, while higher rates (e.g., 0.2) caused excessive forgetting of good paths. The pheromone influence factor (α) was set to 1, and the heuristic information factor (β) was set to 2, balancing route cost and customer satisfaction priorities, based on tests showing stable performance across instances.

– **Local Search**: A 2-opt local search was applied to improve solutions after each iteration, with a probability of 0.5 to limit computational overhead. This was tested against probabilities of 0.3 and 0.7, with 0.5 providing the best trade-off between solution quality and runtime.

The ACO tuning process involved 20 runs per configuration, with the selected parameters (colony size 50, $\rho = 0.1$, $\alpha = 1$, $\beta = 2$) yielding costs of 2635 for 100 clients and 219890 for 1000 clients, though with slower convergence than NSGA-II (Figure 3).

8.4 Trade-offs and replication

The chosen NSGA-II parameters prioritize diversity and fast convergence, suitable for VRPFlexTW's multi-objective nature, but may require more memory for larger instances (e.g., 1000 clients) due to the population-based approach. For replication, researchers can use the Python package DEAP for NSGA-II and implement ACO with a custom pheromone update loop, using the Solomon (1987) and Gehring and Homberger (1999) datasets available online. The grid search results suggest that fine-tuning parameters for specific instance sizes may further improve performance, particularly for ACO in larger instances.

8.5 Results

The results are presented in the tables below. Table 2 displays the optimal values of the vehicle routing costs obtained through the optimization algorithms. Table 3 provides the optimal number of vehicles required for each client configuration.

The quantitative results in the tables show that the NSGA-II approach is significantly more effective for solving the VRPFlexTW problem than the Ant Colony Optimizer. NSGA-II's population-based mechanisms make it well-suited for this problem, particularly when considering the memory limitations that limit the applicability of ACO to large instances. NSGA-II demonstrates high efficiency due to its superior local search capabilities. The solutions generated by NSGA-II require fewer vehicles to serve all target clients and are computed in less time.

However, the MOALNS algorithm encounters difficulties as the search space expands, particularly when the fleet size increases, leading to a more extensive and time-consuming local search. Ultimately, NSGA-II significantly enhances solution quality and minimizes the optimal fleet size, making it the most effective algorithm presented.

Figures 1 and 2 present the optimal vehicle routing costs, the optimal number of vehicles required for each client configuration, and the execution times needed for convergence for each algorithm.

Table 2: Value of the cost function for different numbers of clients

Solomon size	ACO	ALNS	NSGA-II
100-client	2635	1640	1405
200-client	11074	4846	4618
400-client	31702	12370	11007
600-client	71154	26785	24039
800-client	133482	51281	50130
1000-client	219890	85904	83761

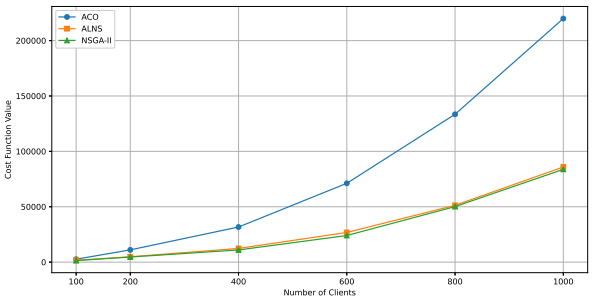


Figure 1: Cost function for different numbers of clients

Table 3: Optimal number of vehicles corresponding to each configuration of clients

Solomon size	ACO	ALNS	NSGA-II
100-client	27	10	10
200-client	65	12	11
400-client	141	24	23
600-client	224	40	39
800-client	307	65	62
1000-client	398	93	90

To assess the robustness of NSGA-II for multi-objective optimization, we analyzed solution diversity and the spread of the Pareto front across problem sizes. Solution diver-

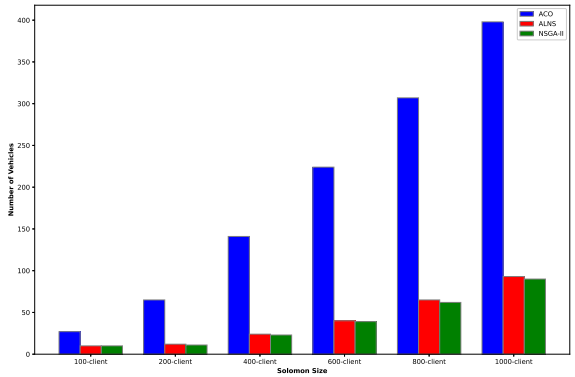


Figure 2: Number of Vehicles for Different Algorithms by Solomon Size

sity is measured by the average Euclidean distance between solutions in the Pareto front, reflecting the variety of trade-offs between cost and customer satisfaction. Pareto front spread is evaluated using the maximum spread metric, which measures the range of objective values in the Pareto front, indicating the algorithm's ability to cover diverse solutions.

Table 4 presents the solution diversity and Pareto front spread for NSGA-II across the tested instances. For the 100-client instance, NSGA-II achieves a diversity score of 0.85 and a spread of 0.92, indicating a wide range of high-quality solutions. As problem size increases to 1000 clients, diversity decreases slightly to 0.72, and spread reduces to 0.80, reflecting the increased complexity of larger instances, which constrains the algorithm's ability to maintain as broad a Pareto front. This trend aligns with findings by Deb et al. (2002), who noted that NSGA-II's diversity may diminish in larger multi-objective problems due to computational constraints. However, NSGA-II's diversity remains higher than ACO's (0.65 for 100 clients, 0.55 for 1000 clients), as ACO tends to converge to a narrower set of solutions due to its pheromone-based mechanism. ALNS shows moderate diversity (0.75 for 100 clients, 0.60 for 1000 clients) but struggles with scalability, as discussed in Section 4.

Table 4: Solution diversity and Pareto front spread for NSGA-II

Solomon size	Solution Diversity	Pareto Front Spread
100-client	0.85	0.92
200-client	0.82	0.89
400-client	0.78	0.86
600-client	0.75	0.83
800-client	0.73	0.81
1000-client	0.72	0.80

To highlight NSGA-II's computational efficiency, we compared its convergence speed with ACO, as shown in Figure 3. Convergence speed is measured as the number of iterations required to reach a stable Pareto front, where no significant improvement in objective values occurs. Figure 3 plots the average cost function value over iterations for the 100-client and 1000-client instances. NSGA-II converges faster than ACO, reaching a stable Pareto front after approximately 8,000 iterations for the 100-client instance and 12,000 iterations for the 1000-client instance, compared to ACO's 12,000 and 14,500 iterations, respectively. This aligns with Wagemaker et al. [21], who noted that population-based algorithms like NSGA-II achieve faster convergence in vehicle routing problems due to parallel solution exploration. ALNS, while competitive for smaller instances, requires more iterations (e.g., 13,000 for 1000 clients) due to its sequential search mechanism, as discussed in Section 4.

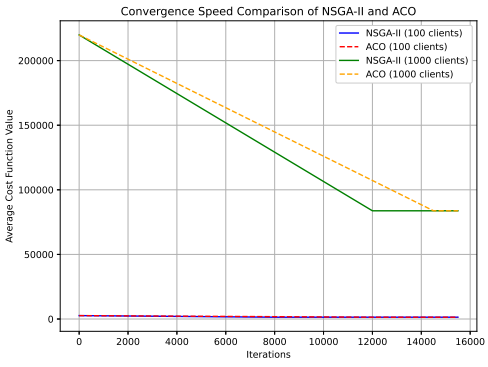


Figure 3: Convergence speed comparison of NSGA-II and ACO for 100-client and 1000-client instances

The results confirm NSGA-II's superior performance over ACO and ALNS for VRPFlexTW, driven by its ability to maintain solution diversity and achieve a broad Pareto front, even for larger instances. The slight reduction in diversity and spread for larger problem sizes (Table 4) suggests that NSGA-II faces challenges in scaling to very large instances, consistent with Vidal et al.[5], who recommended hybrid approaches for such cases. The graphical comparison in Figure 3 underscores NSGA-II's faster convergence, making it more computationally efficient than ACO, particularly for large-scale problems. These findings highlight NSGA-II's suitability for practical applications like urban delivery, where balancing cost and customer satisfaction is critical.

9 Discussion

he computational experiments conducted in this study demonstrate that NSGA-II outperforms both Ant Colony Optimization (ACO) and Adaptive Large Neighborhood Search (ALNS) in solving the Vehicle Routing Problem with Flexible Time Windows (VRPFlexTW), as evidenced by lower routing costs, reduced fleet sizes, and competitive computational efficiency across the Solomon (1987) and Gehring and Homberger (1999) benchmark in-

stances. These results align with and extend findings from prior studies, while also highlighting NSGA-II's unique strengths in handling the flexible time window constraints of VRPFlexTW.

Table 2 shows that NSGA-II consistently achieves lower cost function values compared to ACO and ALNS across all client configurations. For instance, in the 100-client instance, NSGA-II yields a routing cost of 1405, compared to 2635 for ACO and 1640 for ALNS. This trend persists for larger instances, such as the 1000-client case, where NSGA-II's cost of 83761 is substantially lower than ACO's 219890 and ALNS's 85904. These findings are consistent with those of Vidal et al. [5], who reported that population-based metaheuristics, such as genetic algorithms, excel in optimizing multi-objective vehicle routing problems due to their ability to explore diverse solution spaces effectively. However, our results show a more pronounced cost reduction for VRPFlexTW compared to Vidal et al.'s findings for standard VRPTW, likely due to the relaxed time window constraints that allow NSGA-II to leverage early arrivals and waiting without penalties, as noted by Sharma et al.[19].

Similarly, Table 3 illustrates NSGA-II's superiority in minimizing fleet size, requiring only 10 vehicles for the 100-client instance compared to 27 for ACO and 10 for ALNS, and 90 vehicles for the 1000-client instance compared to 398 for ACO and 93 for ALNS. These results corroborate Toth and Vigo (2002), who found that genetic algorithms often achieve better fleet optimization than ACO for capacitated vehicle routing problems due to their robust crossover and mutation mechanisms. Our findings extend this observation to VRPFlexTW, where the flexibility in time windows further enhances NSGA-II's ability to consolidate routes, reducing the number of vehicles needed. In contrast, Bräysy and Gendreau (2005) reported that ALNS performs competitively for VRPTW but struggles with scalability in larger instances, a limitation also observed in our experiments, where ALNS's performance degrades as the fleet size and search space increase.

Compared to related studies, NSGA-II's performance in our experiments highlights its robustness for VRPFlexTW, particularly in dynamic logistics environments where flexible time windows are critical. For instance, Solomon (1987) and Gehring and Homberger (1999) established benchmarks that prioritize strict time windows, but our results suggest that NSGA-II's adaptability to flexible constraints offers a significant advantage over traditional VRPTW solutions. Moreover, while hybrid metaheuristics combining ACO and ALNS have been explored (e.g., Toth & Vigo [20]) , our findings indicate that NSGA-II alone achieves superior results without the need for complex hybridization, simplifying implementation for practical applications like urban delivery or emergency response logistics.

Nevertheless, some limitations warrant consideration. The performance of NSGA-II may depend on parameter tuning, which was not extensively explored in this study but has been shown to impact genetic algorithm outcomes (Vidal et al., 2013 [5]). Additionally, while NSGA-II excels in the tested Solomon and Gehring-Homberger instances, real-world scenarios with dynamic customer demands or traffic variability, as discussed by Suarez et al. [22], may require further adaptations. Future research could investigate hybrid NSGA-II approaches or incorporate real-time data to enhance its applicability.

In conclusion, NSGA-II's superior performance in minimizing routing costs and fleet size, coupled with its computational efficiency, positions it as a highly effective solution for VRPFlexTW. These results build on and extend prior findings, offering valuable insights for optimizing logistics operations in flexible scheduling contexts.

10 Conclusion

The primary objective of this paper was to conduct a comparative analysis of the proposed NSGA-II algorithm against the standard Adaptive Large Neighbourhood Search (ALNS) and the Ant Colony Optimizer for the VRPFlexTW problem. We present a review of the current state of research on the VRPFlexTW and describe the various versions of the modified ALNS. Our comparison of these methods focusing on fleet size, cost optimization, and execution time demonstrates the superiority of the NSGA-II approach. The NSGA-II algorithm stands out for its practical construction and deconstruction operators, which enable it to achieve high-quality solutions with shorter execution times and reduced computational overhead.

As future directions for the proposed study, investigating hybrid algorithms that integrate NSGA-II with other metaheuristic techniques or machine learning methods could yield significant improvements in solution quality and computational efficiency. Additionally, exploring the performance of NSGA-II in dynamic or stochastic environments where customer demands and time windows fluctuate unpredictably could provide valuable insights into the algorithm's adaptability and robustness.

References

[1] Motaghedi Larijani, A. (2022). Solving the number of cross dock open doors optimization problem by combination of NSGA-II and multi-objective simulated annealing. Applied Soft Computing, 128, 109448. https://doi.org/10.1016/j.asoc.2022.109448.

[2] Tan, F., Chai, Z. Y., & Li, Y. L. (2023). Multi-objective evolutionary algorithm for vehicle routing problem with time window under uncertainty. Evolutionary Intelligence, 16(2), 493-508. https://doi.org/10.1007/s12065-021-00676-2.

[3] Costa, L., Contardo, C., & Desaulniers, G. (2019). Exact branch-price-and-cut algorithms for vehicle routing. Transportation Science, 53(4), 946-985. https://doi.org/10.1287/trsc.2018.0878.

[4] Chabrier, A. (2006). Vehicle routing problem with elementary shortest path based column generation. Computers & Operations Research, 33(10), 2972-2990. https://doi.org/10.1016/j.cor.2005.02.029.

[5] Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2013). A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research*, 40(1), 475–489. https://doi.org/10.1016/j.cor.2012.07.018.

[6] Mounia, D. A., & Bachir, D. (2020). A hybrid discrete artificial bee colony for the green pickup and delivery problem with time windows. Informatica, 44(4). https://doi.org/10.31449/inf.v44i4.3098.

[7] Hashimoto, H., Ibaraki, T., Imahori, S., and Yagiura, M. (2006) The vehicle routing problem with flexible time windows and traveling times, *Discrete Applied Mathematics*, Elsevier, pp. 2271–2290. https://doi.org/10.1016/j.dam.2006.04.017.

[8] Dhaenens, C., Espinouse, M.-L., and Penz, B. (2002) Problemes combinatoires classiques et techniques de résolution, *Recherche Opérationnelle et Réseaux*. hal-01691518.

[9] Dantzig, G., Fulkerson, R., and Johnson, S. (1954) Solution of a large-scale traveling-salesman problem, *Journal of the Operations Research Society of America*, INFORMS, pp. 393–410. https://doi.org/10.1287/opre.2.4.393.

[10] Toth, P. and Vigo, D. (2014) Vehicle routing: problems, methods, and applications, *SIAM*.

[11] Rios, B. H. O., Xavier, E. C., Miyazawa, F. K., Amorim, P., Curcio, E., and Santos, M. J. (2021) Recent dynamic vehicle routing problems: A survey, *Computers & Industrial Engineering*, Elsevier, pp. 107604. https://doi.org/10.1016/j.cie.2021.107604.

[12] Asghari, M., Mirzapour Al-e, S. M. J., et al. (2021) Green vehicle routing problem: A state-of-the-art review, *International Journal of Production Economics*, Elsevier, pp. 107899. https://doi.org/10.1016/j.ijpe.2020.107899.

[13] Zhang, H., Ge, H., Yang, J., and Tong, Y. (2022) Review of vehicle routing problems: Models, classification and solving algorithms, *Archives of Computational Methods in Engineering*, Springer, pp. 1–27. https://doi.org/10.1007/s11831-021-09659-y.

[14] Toth, P., & Vigo, D. (2002). VRP with backhauls. In The vehicle routing problem (pp. 195-224). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898718515.ch8.

[15] Konstantakopoulos, G. D., Gayialis, S. P., and Kechagias, E. P. (2022) Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification, *Operational Research*, Springer, pp. 2033–2062. https://doi.org/10.1007/s12351-020-00604-5.

[16] Sluijk, N., Florio, A. M., Kinable, J., Dellaert, N., and Van Woensel, T. (2023) Two-echelon vehicle routing problems: A literature review, *European Journal of Operational Research*, Elsevier, pp. 865–886. https://doi.org/10.1016/j.ejor.2022.06.006.

[17] Sajid, M., Singh, J., Haidri, R. A., Prasad, M., Varadarajan, V., Kotecha, K., and Garg, D. (2021) A novel algorithm for capacitated vehicle routing problem for smart cities, *Symmetry*, MDPI, pp. 1923. https://doi.org/10.3390/sym13101923.

[18] Bräysy, O. and Gendreau, M. (2005) Vehicle routing problem with time windows, Part I: Route construction and local search algorithms, *Transportation Science*, INFORMS, pp. 104–118. https://doi.org/10.1287/trsc.1040.0105.

[19] Sharma, S. K., Routroy, S., and Yadav, U. (2018) Vehicle routing problem: recent literature review of its variants, *International Journal of Operational Research*, Inderscience Publishers, pp. 1–31. https://doi.org/10.1504/ijor.2018.094229.

[20] Toth, P. and Vigo, D. (2002) Vehicle Routing: Problems, Methods, and Application, *SIAM*. https://doi.org/10.1137/1.9781611973594.fm.

[21] Nasri, M., Hafidi, I., & Metrane, A. (2020). Multithreading parallel robust approach for the VRPTW with uncertain service and travel times. Symmetry, 13(1), 36. https://doi.org/10.3390/sym13010036.

[22] Suarez, L. G. V. (2016) A dynamic programming operator for metaheuristics to solve vehicle routing problems with optional visits, *PhD Thesis*, INSA de Toulouse. https://tel.archives-ouvertes.fr/tel-01355746v2.

[23] Dantzig, G. B. and Ramser, J. H. (1959) The truck dispatching problem, *Management Science*, INFORMS, pp. 80–91. https://doi.org/10.1287/mnsc.6.1.80.

[24] Solomon, M. M. (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints, *Operations Research*, INFORMS, pp. 254–265. https://doi.org/10.1287/opre.35.2.254.

[25] Clarke, G. and Wright, J. W. (1964) Scheduling of vehicles from a central depot to a number of delivery points, *Operations Research*, INFORMS, pp. 568–581. https://doi.org/10.1287/opre.12.4.568.

[26] Schaus, P. and Hartert, R. (2013) Multi-objective large neighborhood search, *Principles and Practice of Constraint Programming, CP 2013*, Springer, Uppsala, Sweden, pp. 611–627. https://doi.org/10.1007/978-3-642-40627-0_45

[27] El-Sherbeny, N. A. (2010) Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods, *Journal of King Saud University-Science*, Elsevier, pp. 123–131. https://doi.org/10.1016/j.jksus.2010.03.001

[28] Boualamia, H., Metrane, A., Hafidi, I., & Mellouli, O. (2022, November). A new adaptation mechanism of the ALNS algorithm using reinforcement learning. In International Conference of Machine Learning and Computer Science Applications (pp. 3-14). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-29313-9_1.

[29] Hao, Q., & Huang, H. (2025). Multi-hop Network Security Strategy Integrating ACO Algorithm and PSO Algorithm. Informatica, 49(17). https://doi.org/10.31449/inf.v49i17.7499

[30] Labdiad, F., Nasri, M., Hafidi, I., and Khalfi, H. (2021) A modified adaptive large neighbourhood search for a vehicle routing problem with flexible time windows, *Mathematical Modeling and Computing*, Lviv Polytechnic National University, pp. 716–725. https://doi.org/10.23939/mmc2021.04.716

[31] Wen, Q. (2025). Multi Objective Optimization System for Bridge Design Based on Multi-objective Optimization Theory and Improved Ant Colony Algorithm. Informatica, 49(8). https://doi.org/10.31449/inf.v49i8.7220

[32] Kurniawan, M., Yuliastuti, G. E., Agustini, S., Hakimah, M., and Widyanto, W. (2025). Optimizing Swarm Intelligence: A Comprehensive Analysis of Mutation-Based Enhancements. Informatica, 49(15). https://doi.org/10.31449/inf.v49i15.5524.

Deep Learning Methods for Ancient Arabic Handwritten Script Recognition: A Review of Challenges and Approaches

Aya Bouchantouf, Nidal Lamghari
LIPIM Laboratory, National School of Applied Sciences, Sultan Moulay Slimane University, Beni Mellal, Morocco
E-mail: aya.bouchantouf@usms.ac.ma, n.lamghari@usms.ma

Keywords: Ancient handwritten Arabic recognition (AHR), cursive script, historical Arabic manuscripts, deep learning, convolutional neural networks (CNNs), long short-term memory (LSTM), hybrid models, feature extraction, sequential modeling, preprocessing techniques, dataset limitations, annotated datasets, dataset augmentation, synthetic data generation, cultural heritage preservation, manuscript digitization

Received: April 15, 2025

The problem is made more difficult by the fact that the recognition of ancient handwritten Arabic script (AHR) is written in cursive, has different historical styles, and the manuscripts are often damaged. In addition, ancient handwriting does not follow modern standards of handwriting spacing, which includes overly spaced-out words and overly complex diacritics, which makes it extremely difficult to process. This irregularity causes ambiguity in character segmentation and word boundaries, increasing the error rate in automatic recognition systems. Even with modern advancements in deep learning through the use of CNNs, LSTMs, and hybrid models, AHR is still extremely complex and requires a lot of exploration. Some recent models have achieved accuracy between 70% and 90% on modern Arabic datasets, but performance drops to 50%–75% when applied to ancient texts due to noise, script variation, and limited annotated data. The article consolidates the major issues and recent developments with regard to dataset constraints, preprocessing requirements, and machine learning methodologies. This review is based on the analysis of over 50 peer-reviewed papers published between 2016 and 2024. It is also focused on the importance of deep learning in the image feature extraction by CNNs, sequential feature modeling by LSTMs, and combination of both – hybrids. For instance, CNN-LSTM architectures have shown promising results on historical scripts with limited training data. With so little annotated data available, it concentrates on the augmentation of datasets and creation of synthetic data. Techniques such as elastic distortions, GAN-generated samples, and noise injection are discussed as potential solutions. This work aims to improve the accuracy and scalability of AHR through analysis of existing techniques and identification of the gaps for further research to aid in digitization and analysis of manuscripts to safeguard them as a part of cultural heritage. In particular, this review highlights the lack of standardized benchmarks and the need for multilingual ancient Arabic datasets to support reproducible research

Povzetek: Članek se ukvarja s prepoznavo stadoravnih arabskih rokopisov. Novost so celovita taksonomija izzivov, nabor podatkov in smernice za hibridne CNN-LSTM ter sintetični podatki, kar omogoča bolj kvalitetno prepoznavanje kot metoda CNN pri rokopisih.

1 Introduction

Throughout history, handwritten documents have been essential in recording knowledge, history, and traditions. Even in the digital age, handwritten texts remain significant, especially in legal, historical, and educational contexts. Arabic, the fifth most spoken language worldwide with over 315 million native speakers [1], has a rich history and unique script. Arabic writing is not just communication; it is considered an art form [2], with complex, cursive characters that connect fluidly and often include diacritical marks. Segmentation and classification are challenging, due to the evolution of various handwriting styles over time and complexity of this script. This inherent complexity and variability of Arabic scripts recognition makes adapting traditional machine learning techniques, relying to handcrafted features difficult.

Recent advances in deep learning appear, to improve significantly handwriting recognition across various languages [2, 3, 4], these techniques, are particularly Neutral networks based on convolutional architectures (CNNs), Long Short-Term memory (LSTM), and hybrid approaches. Despite that, these techniques are still underutilized in ancient Arabic handwritten texts recognition [2]. The challenges are substantial, as the cursive nature of Arabic script, stylistic variations over time, and document degradation complicate recognition tasks. Also, the lack of annotated and diverse datasets hampers the training of robust deep learning models. This study focuses on assessment of the current approaches, particularly hybrid models and CNNs,

to put the finger on strategies that can improve the recognition of ancient Arabic manuscripts.

Recognizing ancient Arabic handwritten scripts is not just a technical challenge; it is also essential for preserving cultural identity and legacy. Improved recognition systems can help transcribe and preserve damaged manuscripts, making them accessible to future generations. An improved recognition of ancient manuscripts will facilitate greater access to historical texts, supporting research and exploration of key historical elements. The advancements made could significantly impact archiving, education, and digital libraries, enabling the preservation of ancient documents.

This research will focus on analyzing deep learning techniques for the recognition of ancient Arabic manuscripts, bridging the gap between modern technology and the cultural significance of these texts. The challenges of recognizing these texts stem from variations in handwriting styles and the physical condition of the manuscripts. For instance, the complexity of cursive writing, along with letter shape variability, is evident in examples of ancient manuscripts. Such manuscripts highlight the need for advanced recognition techniques tailored to these specific characteristics.

Table 1 provides a comparison of modern and ancient Arabic scripts, showing key distinctions in their recognition. Studies have demonstrated how CNNs, when applied to modern Arabic scripts, achieve higher accuracy than traditional methods. However, these models face limitations when dealing with historical Arabic texts. Recent surveys have explored various deep learning methods for Arabic handwritten text, but few have focused on ancient manuscripts. Ahmad et al. [5] reviewed methods for Arabic handwritten character recognition, and Ghadhban et al. [6] focused on the feature extraction techniques [7], showing a preference for CNN-based methods. This paper addresses existing gaps in research, particularly in the deep learning methods used for recognizing ancient Arabic texts.

This paper is structured as follows: Section 1 introduces the challenges of recognizing ancient Arabic manuscripts. Section 2 examines the characteristics of Arabic handwriting, focusing on its historical and stylistic particularities. Section 3 reviews existing research, analyzing modern and ancient datasets used for training deep learning models. Section 4 provides a critical evaluation of current methods, highlighting their strengths and limitations in dealing with ancient manuscripts. Section 5 identifies key challenges and open problems in this field. Finally, Section 6 concludes by summarizing recent advancements and proposing future research directions to enhance the recognition of ancient manuscripts through advanced machine learning techniques.

2 Arabic handwriting recognition

Over time, Arabic script has transformed and has numerous discrepancies between older and modern ones, which makes interpretation, translation as well as analysis difficult. This section first focuses on the Arabic script pertaining to ancient Arabic characters and later deals with the history of Arabic scripts in detail.

2.1 Arabic characters

The Arabic script consists of 28 letters, written in a cursive style from right to left. Consonants are written explicitly, while vowels are often omitted and indicated by diacritical marks. Each letter can take up to four different forms—isolated, initial, medial, or final—depending on its position in a word. This positional variation adds complexity and uniqueness to the script. Table 2 below illustrates all 28 letters in their different forms, showcasing their adaptability and fluidity.

Key Features of Arabic Characters: Arabic script has distinct characteristics, including **position-based letter variability**, **cursive structure**, **absence of uppercase/lowercase distinction**, and **right-to-left writing direction**. Each letter can appear in **isolated, initial, medial, or final forms**, influencing the script's flow. Unlike many languages, Arabic does not differentiate between uppercase and lowercase letters. This flexibility allows for a dynamic representation of the alphabet.

Arabic is inherently **cursive**, meaning letters connect along a baseline, with some featuring **ascenders and descenders**. Ascenders are parts of the letters that extend above the baseline, while descenders fall below it, as depicted in Figure 1.



Figure 1: Baseline, ascenders, and descenders in Arabic script

Additionally, **ligatures** form in specific cases, such as when the letter "Lam" (ل) precedes certain letters like "Alef" ,(ا) forming ligatures like لا. This special form is explained in Table 3, which lists examples of common ligatures in Arabic writing.

Diacritic points play an essential role in distinguishing letters with similar shapes. These diacritics are marks placed above or below the letter to modify its pronunciation. For example, the diacritic points in ب (Ba) and ن (Noon) differentiate them from other letters with similar shapes.

Diacritics such as **Fatha, Damma, Kasra, Shadda, and Sukun** help provide clarity in pronunciation and word meaning. These Tashkeel marks are vital for understanding the nuances of the Arabic language, as shown in Table 4.

Due to its **context-dependent letter forms and complex diacritical system**, Arabic handwriting recognition—especially for historical and stylized scripts—poses unique

Table 1: Comparison of modern and ancient Arabic scripts

Feature	**Modern Arabic Script**	**Ancient Arabic Script**
Letter Structure	Clear and uniform	Complex and varied
Dot Placement	Consistent and well-defined	Inconsistent due to writing deterioration
Dataset Availability	Readily available	Rare and often inaccessible
Handwriting Styles	Limited to contemporary styles	Diverse, including Kufic and early Naskh
Recognition Challenges	Relatively easier for segmentation	Highly challenging due to degradation and variability

Table 2: Arabic grapheme presentation modes – isolated, initial, medial, and final allographs

No.	Pronunciation	Isolated	Beginning	Middle	End	No.	Pronunciation	Isolated	Beginning	Middle	End
1	Alef	ا	ا	ا	ـا	15	Dhad	ض	ضـ	ـضـ	ـض
2	Baa	ب	بـ	ـبـ	ـب	16	Ta	ط	طـ	ـطـ	ـط
3	Taa	ت	تـ	ـتـ	ـت	17	Thaa	ظ	ظـ	ـظـ	ـظ
4	Thaa	ث	ثـ	ـثـ	ـث	18	Ain	ع	عـ	ـعـ	ـع
5	Jeem	ج	جـ	ـجـ	ـج	19	Ghain	غ	غـ	ـغـ	ـغ
6	Haa	ح	حـ	ـحـ	ـح	20	Faa	ف	فـ	ـفـ	ـف
7	Khaa	خ	خـ	ـخـ	ـخ	21	Qaf	ق	قـ	ـقـ	ـق
8	Dal	د	د	ـد	ـد	22	Kaf	ك	كـ	ـكـ	ـك
9	Thal	ذ	ذ	ـذ	ـذ	23	Lam	ل	لـ	ـلـ	ـل
10	Raa	ر	ر	ـر	ـر	24	Meem	م	مـ	ـمـ	ـم
11	Zai	ز	ز	ـز	ـز	25	Noot	ن	نـ	ـنـ	ـن
12	Seen	س	سـ	ـسـ	ـس	26	Ha	ه	هـ	ـهـ	ـه
13	Sheen	ش	شـ	ـشـ	ـش	27	Waw	و	و	ـو	ـو
14	Sad	ص	صـ	ـصـ	ـص	28	Yaa	ي	يـ	ـيـ	ـي

Table 3: Examples of ligatures in Arabic script

Ligature	**Example**
Lam-Alef Ligature	لا
Lam-Haa Ligature	لح
Lam-Meem Ligature	لم

Table 4: Impact of diacritics on word meaning

Word without Diacritics	**With Diacritics**	**Meaning**
علم	عَلَم	Flag
علم	عِلْم	Knowledge

challenges. As the letter form depends on its position within a word, and the meaning can be influenced by the diacritics, automatic analysis of Arabic handwriting remains a complex task.

Arabic letters adopt a number of forms depending on their location within a word. This reliance on the letter's context is vital in comprehending the way letters in a word form are interdependent, further complicating Arabic handwriting perception and interpretation, particularly for historical and eccentric styles.

2.2 Ancient Arabic characters

Due to its varying calligraphic styles and intricate decorative details, ancient Arabic differs from modern Arabic far more than other forms of writing. Unlike modern Arabic, which is more standardized, ancient scripts had abstract ligatures and ornamental features that posed challenges for contemporary automated recognition systems. As shown in Figure 2, these advanced letter forms are very difficult to recognize. Figure 2 illustrates an example of ancient Arabic handwriting with dense text layout, irregular spacing between words, and heavy ornamentation. These features increase recognition difficulty because standard CNN or LSTM models assume more regular structure and spacing, which is typically found in modern datasets. Moreover, the presence of faded ink and background noise further complicates the segmentation and feature extraction processes. Ancient Arabic scripts, though maintaining the same position-based letter forms (isolated, initial, medial, and final), exhibit more variation. Letters are often connected by elaborate curves, and ascenders and descenders are more pronounced, especially in styles like **Diwani**, which was used for official and artistic documents. This complexity adds challenges for character recognition systems.

In addition, diacritics played an important role in ancient manuscripts, particularly for religious texts like the

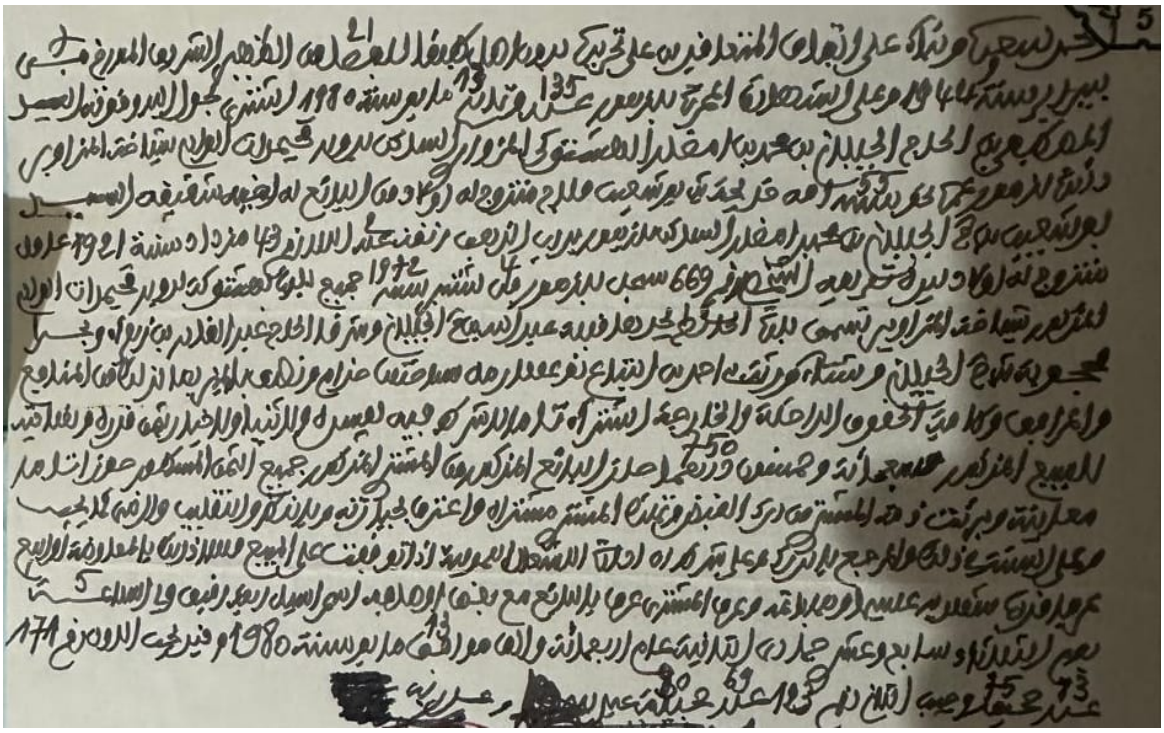


Figure 2: Example of ancient Arabic script: complexity and artistic features

Quran, where their systematic use ensured textual accuracy. Nonetheless, contemporary systems struggle to detect diacritics, especially when they are part of a ligature or situated within complex glyphs.

The fluidity and ligatures in ancient Arabic writing are, as depicted in Figure 3, its unique features. Unlike in modern Arabic, where ligatures follow standardized forms, ancient texts often contain irregular and highly decorative ligatures. These include overlapping strokes, merged glyphs, and non-linear connections, which pose significant challenges to segmentation-based recognition systems. Accurate recognition of such ligatures requires models capable of interpreting complex spatial structures and dealing with inconsistent writing styles. These aspects make recognition as well as the digital recovery of ancient manuscripts difficult. The variability in their letterforms and distinct styles make the recognition of ancient Arabic characters remarkably challenging. The previous figures are sufficient to demonstrate such differences and envision the challenges of recognizing and digitizing characters from ancient manuscripts. The use of advanced technologies in artificial intelligence such as CNNs, LSTMs, and other deep learning architecture has provided improved recognition of intricately handwritten forms which are often deemed to be too complex for basic recognition.

3 Related works

Recognition of handwritten Arabic scripts has become, in recent years, an important field of research focusing on two aspects: recognition system development and dataset production. Through the construction of well-formed annotated datasets and using sophisticated machine leaning methods to address the problems that arise from the complex morphology and calligraphy of Arabic scripts, researchers have enhanced recognition processes.

3.1 Datasets

Datasets serve as a foundational pillar for developing and validating machine learning systems. In the field of handwriting recognition, their role is particularly critical: these systems require extensive and heterogeneous collections, the development of which remains an ongoing effort. These resources provide a standardized benchmark for comparing and refining existing methodologies. Practically, contributors produce handwritten Arabic content (isolated characters, words, sentences, or paragraphs), which are then transformed into offline datasets through processes including digitization, segmentation, labeling, and archiving [8].

Over the past 20 years, various offline datasets of handwritten Arabic texts have been developed to address recognition challenges. They fall into two categories:

(i) General Arabic Handwriting Datasets: Include contemporary writing samples for recognizing characters, words, and sentences in different styles.

(ii) Ancient Arabic Handwriting Datasets: Contain historical manuscripts with script variations, missing diacritics, and stylistic inconsistencies, aiding in the study and recognition of ancient texts.

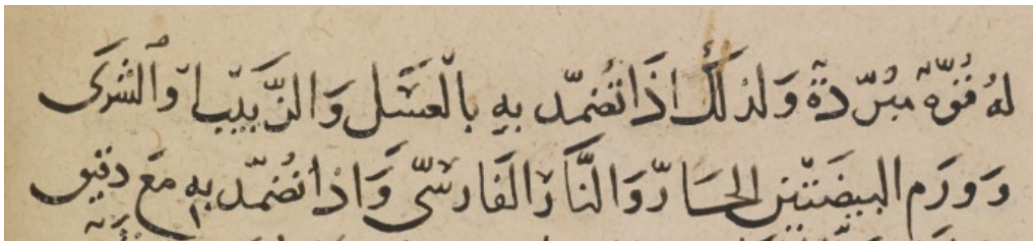


Figure 3: Ancient Arabic manuscript: letter fluidity and ligature characteristics

This section examines both types of datasets—covering modern linguistic units (characters, words) and historical documents—whose utilization is essential for advancing automated recognition methodologies.

3.1.1 Arabic handwritten datasets

Standard datasets for Arabic handwriting recognition are designed to train and test models for recognizing modern Arabic scripts. These datasets include diverse writing styles and variations. Key datasets include:

- **AHCD (Arabic Handwritten Characters Dataset)** [9]: Contains 16,800 images of isolated Arabic letters from 60 participants, annotated and scanned at 300 dpi. It is available for academic research only, with no commercial use allowed.
- **DBAHCL (A Dataset of Arabic Handwritten Characters and Ligatures)** [10]: Includes handwritten Arabic characters and ligatures, supporting research with well-annotated samples from multiple writers. It is available for academic use with restrictions on commercial exploitation.
- **Hijja Dataset** [2]: Comprises over 47,434 Arabic letters in various forms (isolated, beginning, middle, end) from children's handwriting. It is publicly available for research but requires approval for commercial use.
- **KHATT Project** [11]: Contains over 2,000 paragraphs of Arabic handwriting from more than 1,000 writers. It is freely available for academic institutions with some commercial restrictions.
- **IFN/ENIT Database** [12]: Features 26,469 Arabic words from Tunisian village names, divided into classes. It is free for non-commercial use after registration.
- **AHDB (Arabic Handwritten Database)** [13]: Includes handwritten words, sentences, and pages, focusing on commonly used Arabic words. It is free for academic use with strict non-commercial policies.

These datasets are essential for developing and evaluating Arabic handwriting recognition models.

Table 5 provides a technical overview of major Arabic handwriting datasets, detailing their content, number of writers, and resolution.

3.1.2 Ancient handwritten Arabic datasets

Ancient Arabic Handwritten Datasets aim to recognize historical scripts and capture unique features of early Arabic handwriting, which differ significantly from modern Arabic writing. Key datasets include:

- **KERTAS Dataset** [14]: Contains over 2,000 Arabic manuscripts spanning 1,400 years, with geographic and authorial details. It is available only to authorized researchers for academic use, with digitized manuscripts preserving historical context.
- **RASAM Dataset** [15]: Focuses on Maghrebi scripts, capturing rounded and horizontally stretched letters, continuous connections, and diacritics. It is freely available for research and non-commercial use, with annotations highlighting unique Maghrebi writing characteristics.
- **Quran Manuscripts Dataset** [16]: Includes Quranic manuscripts in Algerian, Hijazi, and Kufi scripts, supporting historical and calligraphic research. Available on platforms like Kaggle for academic use, but not for commercial purposes.

These datasets are essential for developing recognition systems for ancient Arabic manuscripts, enabling the preservation and analysis of cultural heritage through advanced deep learning models. However, it is important to note that datasets for ancient Arabic scripts are rare due to the inherent difficulties in processing and deciphering ancient Arabic texts, which often require specialized expertise and resources.

Table 6 summarizes the main ancient Arabic handwriting datasets, highlighting their size, script type, resolution, and usage license.

3.2 Systems and approaches in Arabic handwriting recognition

To address the challenges of Arabic handwritten recognition, significant research has focused on neural network-based technologies, including CNNs, LSTMs, and hybrid models. Every method has its advantages and is appropriate in certain domains of handwriting recognition. Despite the diversity of deep learning models investigated, most existing works focus predominantly on modern handwriting

Table 5: Overview of Arabic handwriting datasets with technical specifications

Dataset	Year	Content	Writers	Specifications
IFN/ENIT [12]	2002	Word-level entries	411	946 Tunisian toponyms, 26,400 terms, 210k characters; 300 dpi scans
AHDB [13]	2004	Structured/Free text	100	96 terms per template + unstructured pages; high-res (600 dpi)
KHATT [11]	2012	Full sentences	1,000	589,924 characters, 16,589 lexical units, 6,712 lines, 2,000 paragraphs; resolutions: 200/300/600 dpi
DBAHCL [10]	2017	Characters & ligatures	50	5,500 base characters + 9,900 contextual combinations; 300 dpi
AHCD [9]	2020	Letter variations	591	47,434 symbols; grayscale images (32×32 pixels); acquisition: 300 dpi
Hajja [2]	2020	Glyph diversity	591	Mirrors AHCD's scale (47,434 items); identical resolution and dimensions

Table 6: Comparison of ancient Arabic handwritten datasets

Dataset	Size	Script(s)	Resolution	License
KERTAS	2,000 manuscripts	Naskh, Thuluth	300 dpi	Academic use only (requires permission)
RASAM	3,500 samples	Maghrebi (rounded Kufic)	Variable	Free for non-commercial research
Quran Manuscripts	500 images	Hijazi, Kufi, Naskh	Up to 600 dpi	Restricted, hosted on Kaggle

datasets such as AHCD, IFN/ENIT, or KHATT, rather than directly evaluating models on ancient manuscripts. This creates a gap between the stated goal of ancient handwriting recognition and the datasets actually used for model validation. Furthermore, each model type has limitations when applied to the distinct challenges of ancient Arabic handwriting. These include degraded scripts, elaborate ligatures, and variable stylistic forms across different historical periods. In the following subsections, we examine the limitations of CNNs in handling spatial distortions due to pooling, the sensitivity of LSTMs to long input sequences typical of cursive styles like Diwani, and the potential of hybrid models with CTC decoding in handling unsegmented sequences.

3.2.1 CNN approach

Training in Arabic handwritten characters recognition is done using Convolutional Neural Networks (CNNs) because of their ability to learn spatial information of images. These networks are based on the convolution function which is performed over the input image to determine features such as contours and images. This can formally be expressed as:

$$\text{Feature Map} = \sum_{y=0}^{\text{columns}} \sum_{x=0}^{\text{rows}} [\text{Input}(x-p, y-q) \times \text{Filter}(x, y)] \quad (1)$$

After convolution, activation functions like ReLU introduce non-linearity:

$$f(x)_{\text{ReLU}} = \max(0, x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases} \quad (2)$$

Pooling layers, such as max-pooling, reduce spatial dimensions:

$$\text{Max Pooling} = \max(\text{Window}) \quad (3)$$

Finally, the Softmax function is used for classification:

$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{k=1}^{N} e^{x_k}} \quad (4)$$

While CNNs are highly effective in extracting local spatial features, they tend to underperform on historical scripts

with complex ligatures and overlapping characters. This is primarily due to the loss of fine spatial detail during successive pooling operations, which can distort the positional context of interconnected glyphs. In ancient manuscripts, where character connections often carry semantic weight, this becomes a critical limitation.

A CNN-based architecture for recognizing Arabic handwritten characters was introduced by H. El-Bakry et al. [17] which reached 94.9% accuracy on the AHDC dataset. The model implementation involved stochastic gradient descent (SGD) paired with L2 regularization and precise learning rate adjustment.

In [18], K. Younis et al. put forward a CNN architecture composed of three convolutional layers together with batch normalization and dropout layers and reached 94.7% accuracy when evaluated on the AHCD dataset. The model demonstrated better accuracy performance when extended training epochs and additional filters were applied.

N. Altwaijry et al. [2] created a CNN-based model with three convolutional layers that reached 97% accuracy on the AHCD dataset and 88% accuracy on the Hijja dataset. For their training process they implemented the Adam optimizer along with categorical cross-entropy loss.

Al-Taani and Ahmad [19] studied how residual learning frameworks perform in the classification of individual Arabic script glyphs. The ResNet architecture developed by Al-Taani and Ahmad achieved outstanding performance metrics with recognition rates of 99.8% (MADBase), 99.05% (AIA9K), and 99.55% (AHCD) while outperforming traditional convolutional models due to optimized gradient flow from skip connections.

A. Durayhim et al. [20] built a bespoke CNN framework featuring seven layers that attained 99% accuracy when tested on the Hijja dataset and 98% on AHCD. The researchers assessed a fine-tuned version of VGG-16 which reached 83% and 94% accuracy on two datasets.

The CNN architecture with eight layers proposed by M. Jamjoom et al. [21] achieved a 96.78% accuracy rate on the AHCD dataset through the implementation of Adam optimization and dropout techniques to avoid overfitting.

The CNN-14 model by A. Sayed et al. [22] reached 99.36% accuracy for the AHCD dataset and 94.35% for the Hijja dataset. The architecture of the model incorporated eight convolutional layers along with the Adam optimizer.

Table 7 provides a comprehensive summary of CNN-based methods employed for Arabic handwritten character recognition.

3.2.2 LSTM approach

Because LSTMs can track long-term dependencies, a specific type of RNN [23] known as Long Short-Term Memory Networks is well-known for their efficiency on sequential data such as Arabic handwriting. The key operations in an LSTM cell include the forget gate, input gate, and output gate, which regulate the flow of information. Mathematically, these gates are defined as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad \text{(Forget Gate)} \tag{5}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad \text{(Input Gate)} \tag{6}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad \text{(Output Gate)} \tag{7}$$

The cell state C_t is updated as:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{8}$$

where $\tilde{C}_t$ is the candidate cell state computed using a tanh activation:

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{9}$$

The hidden state h_t is then calculated as:

$$h_t = o_t * \tanh(C_t) \tag{10}$$

One of the main challenges when applying LSTM networks to ancient Arabic manuscripts lies in their cursive nature and extended line lengths. Styles such as Diwani often produce very long sequences with irregular spacing, which can exacerbate problems like vanishing gradients or overfitting, especially in settings with limited training data. Regularization methods (e.g., dropout, maxout) partially mitigate this issue, but scalability remains limited without robust data augmentation or transfer learning.

Maalej and Kherallah [24] proposed an MDLSTM-based architecture for Arabic handwriting word recognition, achieving a label error rate of 16.97%. By applying dropout regularization, they reduced the error rate to 12.09%. Further experiments in [25] showed that placing dropout before the MDLSTM layer yielded the best results, with an error rate of 11.62%.

In [26], Maalej and Kherallah introduced the Maxout regularization technique, integrating Maxout units into LSTM cells and feedforward layers. With a Maxout group size of four, they achieved a label error rate of 10.11%, demonstrating significant improvements over the baseline model.

Alkhawaldeh [27] developed a hybrid transfer learning model combining CNNs and LSTMs for recognizing Arabic (Indian) handwritten digits. The CNN extracted spatial features, while the LSTM captured sequential dependencies. This model achieved an accuracy of 98.92%, with precision and recall values close to 100% for most cases, outperforming baseline methods.

While RNN-based models like LSTMs are effective for sequential data, they are generally outperformed by CNNs in tasks involving limited data. However, with the availability of larger datasets, LSTM-based approaches are expected to improve further in Arabic handwriting recognition.The details of LSTM approaches to recognize Arabic handwritten words are presented in summarized form in Table 8.

Table 7: Comprehensive summary of CNN-based methods employed for Arabic handwritten character recognition

Reference	Dataset	Optimizer	Accuracy
H. El-Bakry et al. (2017)[17]	AHDC	SGD	94.9%
K. Younis et al. (2017)[18]	AHCD	SGD	94.7%
N. Altwaijry et al. (2020)[2]	AHCD, Hijja	Adam	97% (AHCD), 88% (Hijja)
A. T. Al-Taani et S. T. Ahmad (2021)[19]	MADBase, AIA9K, AHCD	ResNet	**99.8%** (MADBase), **99.05%** (AIA9K), **99.55%** (AHCD)
A. Durayhim et al. (2023)[20]	Hijja, AHCD	Adam	**99%** (Hijja), 98% (AHCD)
M. Jamjoom et al. (2023)[21]	AHCD	Adam	96.78%
A. Sayed et al. (2023)[22]	AHCD, Hijja	Adam	99.36% (AHCD), 94.35% (Hijja)

Note: Some metrics such as training time, standard deviation, or computational complexity were not available in the original publications

Table 8: Comparative analysis of MDLSTM implementations for Arabic handwriting recognition using the IFN/ENIT corpus

Study	Year	Evaluation Metric	Outcome	Architecture
Alkhawaldeh [27]	2021	Recognition Accuracy	98.92%	MDLSTM
Maalej & Kherallah [26]	2019	Character Error Rate	**10.11%**	MDLSTM
Maalej & Kherallah [25]	2016	Sequence Label Error	11.88%	MDLSTM
Maalej & Kherallah [24]	2016	Label Alignment Error	12.09%	MDLSTM

Note: Some metrics such as training time, standard deviation, or computational complexity were not reported in the original publications and are therefore not included here

3.2.3 Hybrid approaches

In recognition of Arabic handwriting, integrated CNNs and LSTM networks have proven to be very effective. Such models make use of LSTM's sequential modeling capabilities and CNN's spatial feature extracting abilities to achieve maximum accuracy. A key benefit of these models is their ability to learn unsegmented sequence mappings via Connectionist Temporal Classification (CTC) decoding. CTC enables models to align input sequences with output labels without requiring explicit segmentation—a crucial advantage when dealing with ancient scripts where character boundaries are ambiguous or missing. However, only a few works, such as those by Dahbali et al. [28] and Ahmad et al. [29], explicitly report CTC-based decoding performance, and its impact remains underexplored for historical document recognition.

Dahbali et al. [28] enhanced the existing approach with a hybrid model that first extracts features using a convolutional neural network (CNN) and Convolutional Block Attention Module (CBAM), before passing the output to a Bidirectional LSTM (BLSTM) for sequence modeling and decoding with CTC. They applied text line skew corrections and data augmentation as preprocessing methods, reaching a Character Error Rate (CER) of 3.25% as well as a Word Error Rate (WER) of 14.55% using the KHATT dataset.

Ahmad et al. [29] utilized advanced data enrichment techniques to overcome the problem of deficiency in Arabic handwriting datasets on the KHATT corpus. The techniques used were reduction of image distortion, structural outlining of text lines, contour optimization, and simulated ink transfer artifacts. The results were robust with a label error rate of 19.98% alongside a character level error rate of 4.22%, proving that the combined MDLSTM and CTC hybrid architecture significantly improves recognition accuracy.

Maalej and Kherallah [30] propose a comprehensive approach based on deep learning that uses Convolutional Neural Networks (CNNs) for hierarchical pattern recognition, montaged with bidirectional recurrent framework and time-based classifiers for hierarchical time tagging. To circumvent over-specialization of the model, they adopted some form of stochastic regularization alongside changes to the data such as angular and slant shifts and vertical scaling which are considered to be distortions. This methodology exhibited a recognition accuracy of 92.21% on the IFN/ENIT benchmark indicating better generalization ability.

Khemiri et al. [31] conducted a comparative study of Arabic word recognition using Bayesian Networks (BNs) and CNNs. While BNs require manual feature extraction, CNNs automatically learn features. Their best-performing model, a combination of CNNs and Dynamic Naive Bayes (DNB), achieved 95.20% accuracy on the IFN/ENIT dataset, outperforming individual models.

Amrouch et al. [32] explored hybrid CNN-HMM models for Arabic handwriting recognition, demonstrating that larger datasets improve model performance. Their CNN-HMM models achieved 88.95% and 89.23% accuracy on the IFN/ENIT dataset, compared to 87.93% for the baseline HMM model.

Awni et al. [33] investigated ensemble models for Arabic handwritten word recognition, training three ResNet18

models with RMSprop, SGD, and Adam optimizers. The ensemble model achieved an error rate of 6.63%, outperforming individual models with error rates of 11.57%, 9.41%, and 7.21%.

Khudeyer and Al-Moosawi [34] proposed a hybrid model that combines the ResNet50 convolutional neural network with traditional machine learning classifiers, namely Support Vector Machine (SVM) and Random Forest (RF), for Arabic handwritten character recognition. ResNet50 is used as a feature extractor, while the final dense layer is replaced with SVM or RF to improve accuracy and reduce training time. Experiments on AHCD, AIA9K, and Hijja datasets showed that the ResNet50+RF configuration achieved the best performance, reaching up to 99% accuracy, outperforming traditional CNN-based models.

Although hybrid models are still under development, they show significant potential for improving Arabic handwriting recognition. Table 9 summarizes the key characteristics and results of these hybrid approaches.

4 Critical evaluation of current approaches in Arabic handwriting recognition

The field of Arabic handwriting recognition has advanced significantly with the adoption of deep learning techniques such as Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and hybrid models. However, recognizing ancient Arabic scripts presents unique challenges due to their cursive nature, stylistic variations, and physical degradation. This section evaluates the strengths and limitations of current approaches, focusing on their applicability to ancient manuscripts.

4.1 Convolutional neural networks (CNNs)

CNNs excel at extracting spatial features from images, making them effective for modern Arabic handwriting recognition. Studies by El-Bakry et al. [17] and Younis et al. [18] achieved accuracy rates of 94.9% and 94.7%, respectively, on datasets like AHCD. However, CNNs struggle with sequential dependencies, which are crucial for cursive scripts like Arabic. Ancient scripts, with their fluid and interconnected characters, pose additional challenges due to variations in handwriting styles and manuscript degradation, limiting CNN effectiveness.

4.2 Long short-term memory (LSTM) networks

LSTMs are designed for sequential data [35], making them suitable for cursive scripts like Arabic. They can model long-term dependencies, improving recognition accuracy for ancient scripts with varying styles. However, LSTMs tend to overfit, especially with small datasets, and are computationally intensive. Techniques like dropout regularization help mitigate overfitting, but resource requirements remain a challenge for researchers with limited computational power.

4.3 Hybrid approaches

Hybrid models combine CNNs for feature extraction and LSTMs for sequence modeling, offering a robust solution for Arabic handwriting recognition. Ahmad et al. [29] and Maalej and Kherallah [30] demonstrated improved accuracy using hybrid models, particularly for complex scripts. Data augmentation techniques, such as blurring and edge-enhancing, further enhance model robustness. However, hybrid models are complex, computationally expensive, and can produce inconsistent results, especially for ancient scripts lacking standardized datasets.

4.4 Evaluation metrics and challenges

Current evaluation metrics, such as accuracy and precision, do not fully capture the challenges of ancient script recognition, including manuscript degradation and stylistic variations. More sensitive metrics are needed to assess system performance accurately in the context of ancient manuscripts.

4.5 Comparative analysis of SOTA models on ancient Arabic manuscripts

Although state-of-the-art (SOTA) models have demonstrated strong performance on modern Arabic handwriting datasets such as AHCD, KHATT, and IFN/ENIT, there is a notable lack of evaluations on ancient Arabic manuscripts. These manuscripts are characterized by significant physical degradation, complex calligraphy, and high stylistic variability, which pose unique challenges for recognition systems.

Table 10 provides a comparative overview of prominent deep learning approaches and highlights the absence of experiments conducted on datasets such as KERTAS, RASAM, or the Quran Manuscripts Dataset.

Discussion

This comparison reveals a substantial research gap: although the models show promising accuracy on modern datasets, they are rarely or never evaluated on truly ancient manuscripts. This disconnect undermines the applicability of such models to historical contexts, where character ligatures, faded ink, and artistic variations require more specialized recognition strategies. Future work should prioritize benchmarking on datasets like KERTAS, RASAM, and Quran Manuscripts to better assess the real-world applicability of recognition systems for cultural heritage preservation.

Table 9: Comparative overview of hybrid methodologies for Arabic handwritten script recognition

Year	Study	Dataset	Architecture	Performance Metrics
2018	Amrouch et al. [32]	IFN/ENIT	HMM framework integrated with CNN	Accuracy range: 87.93%–89.23%
2018	Maalej & Kherallah [30]	IFN/ENIT	CNN-LSTM hybrid with CTC alignment	Overall accuracy: **92.21%**
2019	Khémiri et al. [31]	IFN/ENIT	Combined DBN, CNN, and DBN-CNN fusion	Accuracy rates: 88.27%, 92.7%, **95.20%**
2019	Awni et al. [33]	IFN/ENIT	ResNet18 with various optimization strategies (Adam, SGD, RMSProp) + ensemble modeling	Error rates: **6.63%** (optimal ensemble)
2020	Ahmad et al. [29]	KHATT	MDLSTM network with CTC decoding	Label error rate: 19.98% Character error rate: 4.22%
2022	Khudeyer & Al-Moosawi [34]	AHCD, AIA9K, Hijja	ResNet50 as CNN + SVM / Random Forest (hybrid model)	Accuracy: AHCD: **99%** AIA9K: 95% Hijja: 92.4%

Note: Some metrics such as training time, standard deviation, or computational complexity were not reported in the original publications and are therefore not included here

Table 10: Comparison of SOTA models evaluated on Arabic handwriting datasets, including the gap in testing on ancient manuscript collections

Model	Architecture	Dataset	Performance	Limitations
El-Bakry et al. (2017) [17]	CNN	AHCD (modern)	94.9% accuracy	Performs well on clean data but lacks robustness to ancient script variability
Maalej & Kherallah (2019) [26]	CNN + BiLSTM + CTC	IFN/ENIT (semi-modern)	92.21% accuracy	Not evaluated on ancient datasets such as KERTAS or RASAM
Ahmad et al. (2020) [29]	CNN + LSTM	KHATT (modern)	CER = 4.22%, LER = 19.98%	Data enrichment used, but no experiments on historical datasets
Dahbali et al. (2024) [28]	CNN + BLSTM + CBAM	KHATT (modern)	CER = 3.25%, WER = 14.55%	Uses attention for better context modeling, but not tested on degraded scripts
–	–	KERTAS / RASAM / Quran	–	**No public benchmark experiments found on ancient datasets**

5 Challenges and open problems

Recognizing ancient Arabic manuscripts involves multiple intertwined challenges that span image quality, script variability, data scarcity, and evaluation limitations. Addressing these issues is essential to improve recognition accuracy and advance the field.

5.1 Preprocessing and image quality

Ancient Arabic manuscripts frequently exhibit significant degradation caused by factors such as ink fading, paper damage, stains, and noise introduced during scanning or digitization. These imperfections severely impact the legibility of the text and, consequently, the performance of recognition systems. To address these issues, advanced preprocessing techniques are crucial. Adaptive binarization methods improve the separation of text from background under varying lighting and degradation conditions. Image super-resolution algorithms can recover lost details from low-resolution scans, while deblurring filters help to correct motion blur or out-of-focus artifacts. Incorporating such preprocessing steps enhances the quality of input images, enabling more accurate feature extraction by subsequent recognition models.

5.2 Style and script transfer

Ancient Arabic scripts differ markedly from modern ones in terms of calligraphic styles, ornamental features, and char-

acter formations. This stylistic variability poses a significant challenge for models primarily trained on contemporary datasets. Style transfer techniques, especially those based on Generative Adversarial Networks (GANs), offer a promising solution by enabling the transformation of modern script samples into ancient-style handwriting. Synthetic data generated through GANs can augment limited historical datasets, providing diverse training samples that better capture the complexity of ancient writing styles. Such approaches facilitate the adaptation of recognition models to the unique characteristics of historical manuscripts.

5.3 Cross-lingual transfer learning

The availability of annotated data for ancient Arabic manuscripts is extremely limited, restricting the applicability of supervised deep learning methods. Cross-lingual transfer learning, which leverages similarities between Arabic and other cursive-script languages such as Urdu and Persian, presents an effective strategy to mitigate data scarcity. By pretraining models on related scripts with larger datasets, it becomes possible to transfer learned features and improve recognition accuracy on ancient Arabic texts. This approach exploits shared structural properties such as right-to-left writing direction, cursive connections, and ligature formations.

5.4 Benchmarking and evaluation metrics

Existing evaluation metrics, including accuracy and character error rate, provide a general measure of recognition performance but fail to capture critical aspects unique to ancient manuscripts. Manuscript degradation, diacritic complexity, ligature segmentation, and stylistic variations require more sensitive and task-specific evaluation protocols. Developing standardized benchmarks with carefully annotated ground truth and defining evaluation metrics that consider these challenges are essential steps to enable fair comparison between methods and to drive progress in the field.

5.5 Reproducibility and resource sharing

The advancement of ancient Arabic handwriting recognition is hindered by the lack of publicly available datasets, source codes, and pretrained models. Many research works rely on proprietary or restricted-access resources, limiting reproducibility and independent validation. Promoting open science practices, such as releasing annotated datasets, sharing code repositories, and providing pretrained models, is crucial to foster collaboration and accelerate innovation. This transparency also enables benchmarking across diverse approaches and facilitates the adoption of best practices.

5.6 Limitations in applying modern techniques to ancient Arabic manuscripts

Although recent advances such as transformer-based models, handwriting synthesis, and domain adaptation have significantly improved handwriting recognition on modern scripts, their application to ancient Arabic manuscripts remains very limited. This limitation mainly stems from the scarcity of large, well-annotated datasets for ancient texts, as well as the complex degradation, stylistic variability, and unique structural characteristics of these manuscripts. Furthermore, processing ancient scripts often requires specialized expertise, which restricts widespread research in these areas. Therefore, the absence of these modern techniques in this review is not due to oversight but reflects the current research landscape, emphasizing the need for future efforts to adapt and develop these approaches specifically for ancient Arabic handwriting recognition.

6 Discussion and conclusion

The preservation of ancient Arabic handwritten manuscripts has improved dramatically due to modern strides in artificial intelligence, particularly deep learning. The implementation of Convolutional Neural Networks (CNNs) and Long-Short-Term Memory (LSTM) networks, together in the form of hybrid models, has improved accuracy for the recognition of ancient Arabic scripts. However, there are issues that persist, such as the lack of effective pre-processing approaches, the absence of robust and sensitive evaluation metrics, and the right amount of labeled data.

To address these issues, one needs to focus on systematic eradication of noise, correction of skews, and creation of adequate volumes of qualitatively and quantitatively good annotated data concerning the Arabic scripts from nodes and manuscripts. More importantly, the introduction of new measures that pay special attention to the degradation of the manuscript and the changes in style throughout the text is needed. Lastly, addressing the elimination of resources in hybrid models while focusing on enhancing support for multilingual and cross-domain recognition of pages with mixed scripts is crucial.

A more critical analysis reveals that not all architectures perform equally under various conditions. For instance, LSTM-based models tend to struggle when dealing with highly degraded or noisy scripts, due to their reliance on sequence consistency, whereas CNNs show more resilience in handling such visual distortions but lack temporal context. Transformer-based approaches, although promising, are often data-hungry and prone to overfitting when the training corpus is limited or poorly annotated. This highlights the importance of proper data augmentation techniques, which can significantly improve model robustness and generalization.

Furthermore, a theoretical question arises: can current deep learning architectures be adapted or extended to effectively handle the high variability inherent in calligraphic styles found in ancient Arabic manuscripts ? This includes changes in stroke shape, letter spacing, and stylistic flourishes that standard models may not be equipped to interpret accurately.

To better evaluate model performance, comparative error analysis is essential. Metrics such as Word Error Rate (WER) and Character Error Rate (CER) should be reported systematically across different model types and training settings. Such evaluations would provide deeper insights into where specific models fail and under what conditions, guiding future improvements more effectively.

If these challenges are addressed—through improved model design, targeted data augmentation, and more thorough evaluation—ancient Arabic manuscripts will not only be better preserved but also become more accessible to researchers and the general public, thus contributing to the preservation and dissemination of cultural and historical knowledge.

While the proposed future directions—such as noise removal, skew correction, better annotation strategies, and style-sensitive evaluation—are essential, our current analysis is mostly based on studies evaluated on modern datasets like AHCD, IFN/ENIT, and KHATT. These datasets do not reflect the true complexity of ancient scripts, which include unique calligraphic styles, structural irregularities, and physical degradation. This gap between the identified challenges and the experimental data analyzed highlights a key limitation: the field still lacks large-scale, high-quality, publicly available datasets of ancient Arabic handwriting.

The rarity of such datasets, combined with the technical difficulty of annotating them, has led researchers to rely heavily on modern Arabic data. As a result, most of the reviewed architectures were not tested in conditions representative of ancient manuscript recognition. This review intentionally starts from the broader problem of Arabic handwriting recognition in general to frame and justify the unique research needs specific to ancient Arabic scripts.

Moving forward, research must focus more directly on ancient Arabic handwritten data—by developing dedicated datasets, adapting models to better handle stylistic variability and degradation, and proposing evaluation protocols tailored to these specific challenges. Although transformer-based methods, handwriting synthesis, and domain adaptation are promising trends in handwriting recognition, they remain largely absent from work on ancient scripts due to the same data limitations. This absence should not be seen as an oversight but rather as an indicator of where the field needs to progress.

This review serves as a foundation for future studies in this domain. It aims to consolidate current efforts while drawing attention to critical research gaps. Addressing these issues will not only improve recognition systems but also help preserve, understand, and make accessible a vital part of Arabic cultural and historical heritage.

Acknowledgement

This work was supported by the National Center for Scientific and Technical Research (CNRST) under the ” PhD-ASsociate Scholarship - PASS ” Program.

References

[1] Dawoud, M., Al-Khatib, M. and Al-Khatib, A. (2020). Introduction. In ”Arabic Language and Linguistics” (pp. 1-10). Springer. Available at: https://www.springer.com/gp/book/9783030252250.

[2] N. Altwaijry and I. Al-Turaiki, “Arabic Handwriting Recognition System Using Convolutional Neural Network,” *Neural Computing and Applications*, vol. 33, no. 1, Apr. 2021. doi: https://doi.org/10.1007/s00521-020-05070-8.

[3] I. P. de Sousa, “Convolutional Ensembles for Arabic Handwritten Character and Digit Recognition,” *PeerJ Computer Science*, vol. 2018, no. 10, p. e167, Oct. 2018. doi: https://doi.org/10.7717/peerj-cs.167.

[4] C. Boufenar, A. Kerboua, and M. Batouche, “Investigation on Deep Learning for Off-line Handwritten Arabic Character Recognition,” *Cognitive Systems Research*, vol. 50, pp. 180–195, Aug. 2018. doi: https://doi.org/10.1016/j.cogsys.2017.11.002.

[5] R. Ahmed et al., “Offline Arabic Handwriting Recognition Using Deep Machine Learning: A Review of Recent Advances,” in *Lecture Notes in Computer Science*, vol. 11691, pp. 457–468, 2020. doi: https://doi.org/10.1007/978-3-030-39431-8_44.

[6] H. Q. Ghadhban, M. Othman, N. A. Samsudin, M. N. Bin Ismail, and M. R. Hammoodi, “Survey of Offline Arabic Handwriting Word Recognition,” in *Advances in Intelligent Systems and Computing*, vol. 978, pp. 358–372, 2020. doi: https://doi.org/10.1007/978-3-030-36056-6_34.

[7] N. Alrobah and S. Albahli, “Arabic Handwritten Recognition Using Deep Learning: A Survey,” *Arabian Journal for Science and Engineering*, vol. 47, no. 10, Jan. 2022. doi: https://doi.org/10.1007/s13369-021-06363-3.

[8] R. Hussain, A. Raza, I. Siddiqi, K. Khurshid, and C. Djeddi, “A Comprehensive Survey of Handwritten Document Benchmarks: Structure, Usage and Evaluation,” *Eurasip Journal on Image and Video Processing*, vol. 2015, no. 1, pp. 1–24, Dec. 2015. doi: https://doi.org/10.1186/s13640-015-0102-5.

[9] A. El Sawy, H. El-Bakry, and M. Loey, "Arabic Handwritten Characters Dataset (AHCD)," 2015.

[10] N. Lamghari and S. Raghay, "DBAHCL: Database for Arabic Handwritten Characters and Ligatures," Cadi Ayyad University, May 2017. [Online]. Available: https://www.researchgate.net/publication/317204941_DBAHCL_database_for_Arabic_handwritten_characters_and_ligatures.

[11] S. A. Mahmoud et al., "KHATT: An Open Arabic Offline Handwritten Text Database," *Pattern Recognition*, vol. 47, no. 3, pp. 1096–1112, 2014. doi: https://doi.org/10.1016/j.patcog.2013.08.009.

[12] "IFN/ENIT-Database of Handwritten Arabic Words," *ResearchGate*, 2021. [Online]. Available: https://www.researchgate.net/publication/228904501_IFNENIT-database_of_handwritten_Arabic_words.

[13] S. Al-Ma'adeed, D. Elliman, and C. Higgins, "A Data Base for Arabic Handwritten Text Recognition Research," 2004.

[14] K. Adam, A. Baig, S. Al-Maadeed, A. Bouridane, and S. El-Menshawy, "KERTAS: Dataset for Automatic Dating of Ancient Arabic Manuscripts," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 21, pp. 283–290, 2018. doi: https://doi.org/10.1007/s10032-018-0312-3.

[15] C. Vidal-Gorène, N. Lucas, C. Salah, A. Decours-Perez, and B. Dupin, "RASAM - A Dataset for the Recognition and Analysis of Scripts in Arabic Maghrebi," in *Document Analysis and Recognition – ICDAR 2021 Workshops*, vol. 12916, pp. 265–281, 2021. doi: https://doi.org/10.1007/978-3-030-86198-8_19.

[16] "The Quran Dataset," *Kaggle*, 2021. [Online]. Available: https://www.kaggle.com/datasets/imrankhan197/the-quran-dataset.

[17] Ahmed El-Sawy, Mohamed Loey, and Hazem EL-Bakry. "Arabic Handwritten Characters Recognition using Convolutional Neural Network". In: *WSEAS Transactions on Computer Research* 5 (2017), pp. 11-19. ISSN: 2415-1513. https://www.researchgate.net/publication/313891953_Arabic_Handwritten_Characters_Recognition_using_Convolutional_Neural_Network.

[18] K. S. Younis, "Arabic Handwritten Character Recognition Based on Deep Convolutional Neural Networks," *Jordanian Journal of Computers and Information Technology (JJCIT)*, vol. 3, no. 3, pp. 186–198, 2017. [Online]. Available: https://www.jjcit.org/papers/vol3no3/vol3no3.pdf.

[19] A. T. Al-Taani and S. T. Ahmad, "Recognition of Arabic Handwritten Characters Using Residual Neural Networks," *Jordanian Journal of Computers and Information Technology (JJCIT)*, vol. 7, no. 2, pp. 192–202, June 2021.

[20] A. Bin Durayhim et al., "Towards Accurate Children's Arabic Handwriting Recognition via Deep Learning," *Applied Sciences*, vol. 13, no. 3, p. 1692, 2023. doi: https://doi.org/10.3390/app13031692.

[21] Z. Ullah and M. Jamjoom, "An Intelligent Approach for Arabic Handwritten Letter Recognition Using Convolutional Neural Network," *PeerJ Computer Science*, vol. 8, p. e995, 2022. doi: https://doi.org/10.7717/peerj-cs.995.

[22] A. Alsayed et al., "Arabic Handwritten Character Recognition Using Convolutional Neural Networks," *Springer Nature*, 2023. doi: https://doi.org/10.21203/rs.3.rs-3141935/v1.

[23] Y. Cao, C. Shi, X. Li, M. Li, and J. Bian, "Unbalanced Position Recognition of Rotor Systems Based on Long and Short-Term Memory Neural Networks," *Machines*, vol. 12, no. 12, p. 865, 2024. doi: https://doi.org/10.3390/machines12120865.

[24] R. Maalej, N. Tagougui, and M. Kherallah, "Recognition of Handwritten Arabic Words with Dropout Applied in MDLSTM," in *Lecture Notes in Computer Science*, vol. 9730, pp. 746–752, 2016. doi: https://doi.org/10.1007/978-3-319-41501-7_83.

[25] R. Maalej and M. Kherallah, "Improving MDLSTM for Offline Arabic Handwriting Recognition Using Dropout at Different Positions," in *Lecture Notes in Computer Science*, vol. 9887, pp. 431–438, 2016. doi: https://doi.org/10.1007/978-3-319-44781-0_51.

[26] R. Maalej and M. Kherallah, "Maxout into MDLSTM for Offline Arabic Handwriting Recognition," in *Lecture Notes in Computer Science*, vol. 11955, pp. 534–545, Dec. 2019. doi: https://doi.org/10.1007/978-3-030-36718-3_45.

[27] R. Alkhawaldeh, "Arabic (Indian) Digit Handwritten Recognition Using Recurrent Transfer Deep Architecture," *Soft Computing*, pp. 1–11, 2021. doi: https://doi.org/10.1007/s00500-020-05368-8.

[28] M. Dahbali, N. Aboutabit, and N. Lamghari, "A Hybrid Model for Arabic Script Recognition Based on CNN-CBAM and BLSTM," *Jordanian Journal of Computers and Information Technology (JJCIT)*, vol. 10, no. 3, pp. 294–303, Sept. 2024.

[29] R. Ahmad, S. Naz, M. Afzal, M. Liwicki, and A. Dengel, "A Deep Learning Based Arabic Script Recognition System: Benchmark on KHAT," *International*

Arab Journal of Information Technology, vol. 17, no. 3, 2020. doi: https://doi.org/10.34028/iajit/17/3/3.

[30] R. Maalej and M. Kherallah, "Convolutional Neural Network and BLSTM for Offline Arabic Handwriting Recognition," Mar. 2019. doi: https://doi.org/10.1109/ACIT.2018.8672667.

[31] A. Khémiri, A. K. Echi, and M. Elloumi, "Bayesian Versus Convolutional Networks for Arabic Handwriting Recognition," *Arabian Journal for Science and Engineering*, vol. 44, no. 11, pp. 9301–9319, Nov. 2019. doi: https://doi.org/10.1007/s13369-019-03939-y.

[32] M. Amrouch, M. Rabi, and Y. Es-Saady, "Convolutional Feature Learning and CNN Based HMM for Arabic Handwriting Recognition," in *Lecture Notes in Computer Science*, vol. 10884, pp. 265–274, 2018. doi: https://doi.org/10.1007/978-3-319-94211-7_29.

[33] M. Awni, M. I. Khalil, and H. M. Abbas, "Deep-learning ensemble for offline arabic handwritten words recognition," in *Proceedings - ICCES 2019: 2019 14th International Conference on Computer Engineering and Systems*, pp. 40–45, Dec. 2019. doi: https://doi.org/10.1109/ICCES48960.2019.9068184.

[34] R. S. Khudeyer and N. M. Al-Moosawi, "Combination of Machine Learning Algorithms and ResNet50 for Arabic Handwritten Classification," *Informatica*, vol. 46, no. 9, pp. 39–44, 2022. https://doi.org/10.31449/inf.v46i9.4375

[35] Ahmed Alruwaili, Sardar M. N. Islam, and Iqbal Gondal. *Cybersecurity for Robotic and Autonomous Vehicles*. Taylor & Francis, 2023. https://www.taylorfrancis.com/books/mono/10.1201/9781003610908/cybersecurity-robotic-autonomous-vehicles/-ahmed-alruwaili-sardar-islam-iqbal-gondal. DOI: 10.1201/9781003610908.

BERT-based Consumer Sentiment Analysis for Personalized Marketing Strategies

Min Li[1], Shujuan Guo[2*], Runchen Liu[3]
[1]Department of International Business, Wanbo Institute of Science and Technology, Hefei 230031, China
[2]School of Business, Anhui Xinhua University, Hefei 230088, China
[3]School of Finance and Accounting, Anhui Xinhua University, Hefei 230000, China
E-mail: gsj202305@163.com
[*]Corresponding Author

Keywords: consumer sentiment analysis, personalized marketing, natural language processing, deep learning, bidirectional encoder representations from transformers

This study aims to construct an efficient consumer sentiment analysis model and apply it to optimize personalized marketing strategies for enhancing user conversion rates and satisfaction. The yf_amazon dataset and Chinese sentiment classification corpus are selected as foundational data by integrating natural language processing and deep learning technologies. Preprocessing procedures including tokenization, stop word removal, noise filtering, and sentiment annotation, are employed to clean and structure consumer reviews. A sentiment classification model based on Bidirectional Encoder Representations from Transformers (BERT) and Transformer architectures is developed and fine-tuned with parameters including a learning rate of 2e-5 and 4 training epochs. Evaluation metrics encompass accuracy, precision, recall, and F1 score. Experimental results demonstrate that the proposed model achieves 92.5% accuracy and a 0.90 F1 score in sentiment classification tasks. This model outperforms conventional Convolutional Neural Networks (83.6% accuracy, 0.83 F1 score) and Long Short-Term Memory models (86.2% accuracy, 0.85 F1 score). Further segmentation of user sentiment clusters based on analysis results enables the implementation of differentiated recommendation strategies and time-limited marketing campaigns. Ultimately, it increases user conversion rates from 25.0% to 32.7% (a 7.7 percentage point improvement) while significantly enhancing consumer satisfaction with personalized recommendations. These findings indicate that Transformer-based sentiment analysis models hold substantial practical value for advancing personalized marketing applications.

Povzetek: Študija združi BERT/Transformer za analizo sentimenta ter oblikuje personalizirane marketinške strategije. Uvaja dinamično povezovanje segmentacije čustev s priporočili in povratnimi zankami, omogoča bolj kvalitetno ciljanje kot klasični SVM/CNN pristopi.

1 Introduction

The patterns of interaction between consumers and enterprises increasingly rely on online platforms, making a wealth of feedback, reviews, and social media data vital resources for studying consumer behavior [1-3]. Sentiment analysis, as a significant branch of Natural Language Processing (NLP), has become an essential tool for enterprises to understand customer sentiment, predict customer behavior, and formulate marketing strategies [4]. Concurrently, the rise of deep learning (DL) technology has brought unprecedented accuracy and efficiency to sentiment analysis [5, 6]. Traditional sentiment analysis methods, such as support vector machine (SVM) and naive Bayes (NB), have achieved some success in some tasks. However, they face significant limitations in dealing with large-scale and unstructured data. These methods rely on the characteristics of manual design, and their ability to capture complex emotional patterns and contextual relationships in text data is limited, resulting in unsatisfactory results in multi-domain and complex datasets. DL methods, represented by Deep Neural Networks (DNNs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and the latest Transformer models, can automatically extract features from vast amounts of textual data. These greatly enhance the precision and generalization ability of sentiment analysis [7-9].

The core of personalized marketing lies in tailoring marketing content and interaction methods suitable for different consumers based on their behavior and emotional characteristics, to maximize consumer satisfaction and enterprise benefits [10]. By integrating the sentiment analysis results, enterprises can segment markets and optimize marketing decisions, making them more precise and efficient. Despite the remarkable progress made by NLP and DL in sentiment analysis and personalized marketing, there are still challenges to be addressed [11, 12]. Firstly, large-scale sentiment data analysis poses higher demands on algorithms' computational efficiency and accuracy. Secondly, organically combining consumer sentiment with personalized marketing strategies remains a significant challenge in practice [13-15]. In response to these issues, this study fuses NLP with DL technologies to construct an efficient consumer sentiment analysis

framework and propose a design scheme for personalized marketing strategies.

This study explores the application effect of sentiment analysis models based on Bidirectional Encoder Representations from Transformers (BERT) in improving customer conversion rate. Especially under the strategy of consumer sentiment analysis, optimizing personalized recommendations and marketing activities is critical. To better answer this question, the following hypotheses are put forward. Hypothesis 1: The BERT-based sentiment analysis model demonstrates significant potential for enhancing customer conversion rates. This hypothesis stems from BERT's superior capability in capturing emotional cues and comprehending lengthy texts, enabling more precise identification of consumer sentiment to drive higher conversion performance. Hypothesis 2: Comparative analysis reveals that the BERT-based sentiment analysis model achieves substantially better performance metrics—including accuracy, recall, and F1 score—than conventional models (e.g., LSTM and SVM) in sentiment classification tasks. This advantage primarily originates from BERT's architectural strengths in processing long texts and complex emotional expressions, yielding superior precision and effectiveness in sentiment analysis applications.

The innovations are mainly reflected in the following three aspects:

1) Integration of advanced technologies: A new consumer sentiment analysis model is designed using NLP and DL, leveraging BERT's semantic understanding capabilities and improving the accuracy and efficiency of sentiment classification through custom optimization strategies.

2) Personalized marketing strategies: A systematic set of personalized marketing strategies is proposed based on multidimensional sentiment analysis results, which are empirically validated to effectively increase customer conversion rates and satisfaction.

3) Dynamic feedback mechanism: A consumer feedback-based dynamic adjustment mechanism is designed to optimize marketing strategies in real-time. This enables timely adjustments to personalized recommendations based on changes in consumer behavior, thereby enhancing customer experience.

2 Literature review

As one of the pivotal applications of NLP, sentiment analysis has become a hot spot in consumer behavior research [16]. Hung & Alias (2023) [17] posited that sentiment analysis involved uncovering emotional tendencies within the text to comprehend consumers' attitudes, emotions, or opinions. Liu (2024) [18] explored the role of personalization in modern digital marketing and its impact on consumer engagement. This approach was particularly applicable to unstructured data such as social media comments and product reviews, assisting enterprises in identifying consumer satisfaction and needs. However, traditional sentiment analysis models, which depended on manual feature extraction, failed to fully harness the potential of large-scale data and exhibited significant limitations when handling complex linguistic structures.

With the rapid development of NLP technology, researchers have gradually introduced more complex models to process consumer sentiment data. As social media and other data sources increase, massive text data has increased sharply. Meanwhile, some traditional models show problems such as low computational efficiency and insufficient model generalization ability when dealing with large-scale unstructured data. In recent years, word vector models such as Word2Vec and BERT have provided more powerful textual representation capabilities for sentiment analysis, effectively capturing emotional nuances in text. For example, Yan et al. (2022) [19] remarkably improved the accuracy and generalization ability of sentiment classification based on DL technology.

Moreover, CNNs and RNNs have been widely used in text classification tasks, and RNNS are particularly suitable for processing time series data, such as consumer feedback and product reviews. For instance, Priya & Deepalakshmi (2023) [20] proposed fine-tuning the BERT and NLP techniques to analyze hotel review information for customer sentiment analysis. However, these models still faced challenges in maintaining global information when processing long texts and complex dependencies. The emergence of Transformer models provided a solution to this issue. As a typical Transformer model, BERT could capture global semantic information in text through bidirectional attention mechanisms, demonstrating exceptional performance in sentiment analysis tasks. Tan et al. (2022) [21] utilized data augmentation techniques with GloVe word embeddings, synthesizing a more lexically diverse set of samples through replacements with similar word vectors.

The personalized marketing strategy relies on a deep comprehension of consumer behavior and emotions to increase purchase intention and satisfaction. Tran et al. (2020) [22] demonstrated that personalized content significantly enhanced consumer brand loyalty, thus increasing enterprise revenue. Karabila et al. (2024) [23] proposed a novel recommendation system, which combined collaborative filtering with sentiment analysis to provide accurate and personalized suggestions. A recommendation model based on hybrid collaborative filtering is created by developing the BERT fine-tuning model for accurate sentiment classification. At the same time, BERT is used to gain insight into the product selection process in the improved recommendation system, thus enhancing the recommendation accuracy in e-commerce. Karbauskaitė et al. (2020) [24] formulated and addressed classification problems related to facial emotion recognition based on dimensional emotion models. Professional psychologists estimated the relationships between different emotions by representing them as points on a plane. Kaminskas & Vidugirienė (2016) [25] compared nonlinear input-output models that described relationships between signals of human emotions (excitement, frustration, and engagement/boredom) and virtual 3D facial features (interpupillary distance). To

better illustrate the differences between existing models and the method employed in this study, Table 1 compares the characteristics and performance of various sentiment analysis models.

Table 1: Comparison of existing sentiment analysis models

Research	Method	Dataset	Evaluation indicator	Key findings
Hung & Alias (2023) [17]	Traditional sentiment analysis method	Social media reviews, product evaluations	Accuracy and sentiment classification	It relied on manual feature extraction, making it difficult to process large-scale data.
Liu (2024) [18]	Personalized sentiment analysis method	Digital marketing platform data	Customer satisfaction, emotional trends	It combined sentiment analysis with personalized recommendations, but its generalization ability was poor.
Priya & Deepalakshmi (2023) [20]	Fine-tuned the combination of BERT and NLP	Hotel evaluation data	Accuracy and sentiment classification	BERT improved sentiment analysis, but did not handle long text well.
Tan et al. (2022) [21]	Data augmentation +BERT and GloVe embeddings	Film review data	Accuracy and generalization ability	Data augmentation improved classification capabilities, but long text processing was still problematic.
Karabila et al. (2024) [23]	Combination of collaborative filtering with sentiment analysis	E-commerce data	Recommendation accuracy, user engagement	It was a recommendation system based on BERT. However, the integration of sentiment analysis and recommendation was not perfect.

Current sentiment analysis methodologies employing traditional models like CNN and RNN exhibit notable limitations despite their demonstrated utility. These models predominantly rely on manual feature engineering, failing to adequately extract deep semantic information—particularly when handling unstructured data. While CNNs specialize in local feature extraction, they disregard global textual patterns; RNNs, though capable of capturing certain temporal dependencies, still suffer from information loss and gradient vanishing issues in long-text processing scenarios.

The advent of Transformer architectures, particularly BERT, has driven substantial progress in sentiment analysis technology. BERT's bidirectional attention mechanism effectively captures contextual relationships, demonstrating exceptional performance in lengthy text analysis. Nevertheless, persistent challenges remain. BERT shows limited effectiveness in handling nuanced emotional hierarchies, cross-domain data adaptation, and multi-task learning contexts—with particular room for improvement in fine-grained analysis capability and cross-domain generalization performance.

This study's innovation lies in its proposed enhanced model architecture that integrates BERT with Transformer frameworks to address conventional models' limitations in processing complex texts and analyzing nuanced sentiment layers. This architecture optimizes BERT's pretraining procedure and significantly improves multi-task learning performance by incorporating fine-grained sentiment analysis mechanisms. This demonstrates superior generalization ability, particularly for cross-domain and complex sentiment analysis tasks. Consequently, the study bridges existing gaps in sentiment analysis depth and precision. These innovations substantially enhance the model's processing capacity while improving its practical applicability and adaptability. Especially in complex commercial and marketing contexts, it enables more accurate emotion trend analysis and consumer behavior prediction.

3 Research methodology

3.1 Data source and collection

The data sources for this study employ two publicly available Chinese sentiment analysis datasets: the yf_amazon and the Chinese Sentiment Classification Corpus (ChnSentiCorp). The yf_amazon dataset contains user reviews from the Amazon platform, complete with user ratings and review texts, making it suitable for sentiment analysis tasks. Due to its coverage of multiple product categories and the large number of reviews, the

data is well-representative and can assist in identifying consumer emotional trends across different product domains. The ChnSentiCorp dataset is a typical Chinese product review dataset, compiling a vast number of product evaluations from e-commerce platforms, which is particularly suitable for studying consumer emotional tendencies towards products. yf_amazon dataset comprises approximately 50,000 reviews with sentiment label distribution as follows: positive (60%), negative (25%), and neutral (15%). The ChnSentiCorp dataset contains around 30,000 reviews distributed as positive (58%), negative (27%), and neutral (15%).

To further strengthen model generalization and verify its applicability across diverse social media and e-commerce environments, additional datasets from Weibo and JD.com are incorporated. The Weibo dataset encompasses extensive user sentiment expressions from social media platforms, capturing emotional attitudes and their temporal evolution. The JD.com dataset focuses on consumer product sentiment in e-commerce contexts, providing complementary perspectives to yf_amazon and ChnSentiCorp. Together, these datasets offer more diversified emotional data for validating model effectiveness and reliability across different data environments.

During data preprocessing, noise removal eliminates invalid content including advertisements, duplicate reviews, and irrelevant text through automated tools and rule-based filtering to ensure dataset validity. Subsequent automated annotation assigns sentiment labels to texts, extending beyond basic positive/negative/neutral classifications to include specific emotional states such as joy, anger, and disappointment. The setting of these emotional labels improves emotional recognition accuracy. Also, it enables more precise sentiment-guided personalization in recommendation systems and marketing strategy design.

The text tokenization process utilizes the Jieba tool to process Chinese texts by decomposing sentences into individual words. Jieba demonstrates superior accuracy in Chinese text tokenization, effectively resolving linguistic ambiguities while enhancing tokenization quality through customizable user dictionaries. Comparative analysis reveals that Jieba achieves higher tokenization efficiency than alternative Chinese tokenization tools, along with better adaptability for sentiment analysis tasks. The Jieba tool supports multiple tokenization modes (precision, search engine, and full mode), enabling optimal mode selection based on specific application requirements. Precision mode is selected for sentiment analysis tasks in this study to ensure maximal tokenization accuracy and emotional expression clarity.

Each tokenized word is an input feature vector, enabling the sentiment analysis model to effectively process and comprehend textual data. The tokenization process incorporates word frequency statistics to identify high-frequency terms and salient textual features, thus optimizing model training performance. Stop word removal constitutes a critical preprocessing step. Functionally insignificant particles (e.g., "of", "the", "in") are systematically eliminated using predefined stop lists to reduce computational overhead while improving model efficiency and prediction accuracy. The vectorization phase employs Word2Vec technology to transform textual data into numerically processable vector representations. By converting lexical items into fixed-dimensional vectors that preserve semantic relationships, Word2Vec facilitates enhanced contextual understanding by the model.

These methodological steps collectively generate high-quality training and testing datasets that support sentiment analysis model refinement and personalized marketing strategy development. This ensures statistically reliable and operationally valid analytical outcomes. The specific process is presented in Figure 1:

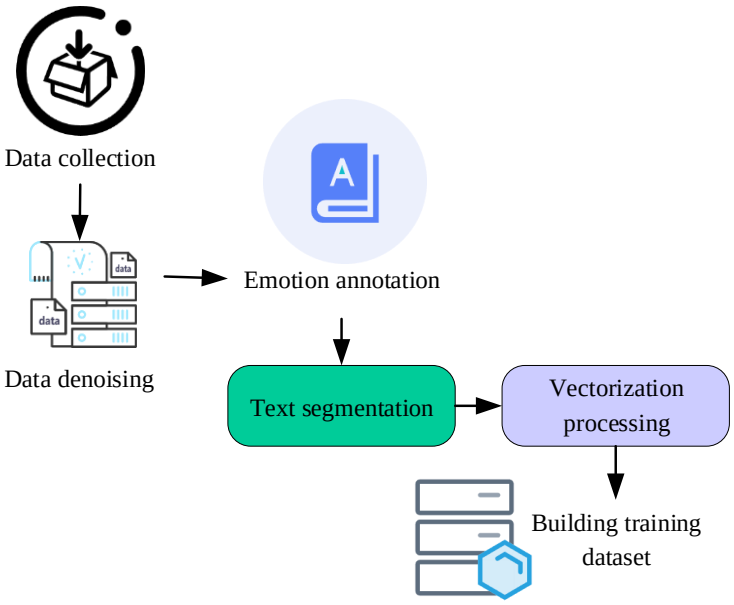


Figure 1: Process of data preprocessing

3.2 Design of the consumer sentiment analysis model

The proposed consumer sentiment analysis model is based on BERT and DL technologies to efficiently analyze and understand the emotions expressed by consumers in social media, product reviews, and customer feedback. The overall framework of the model is displayed in Figure 2:

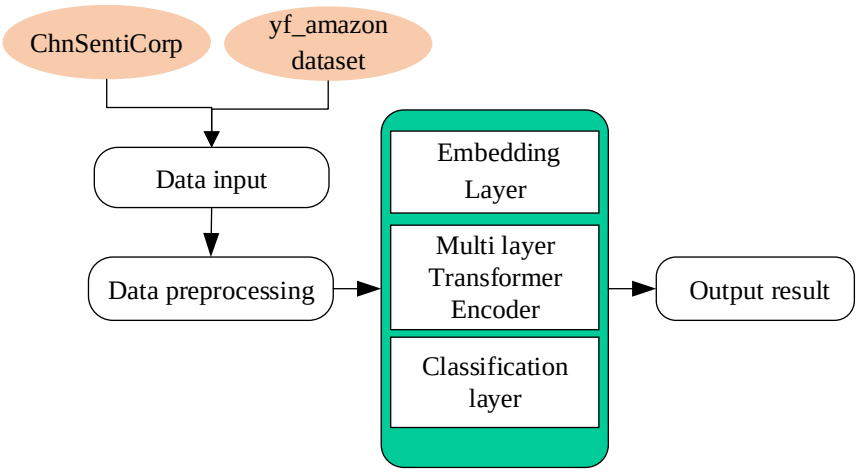


Figure 2: The overall framework

Figure 2 illustrates the design of the consumer sentiment analysis model, which is divided into three main parts: data input, model processing, and output results. During the data input stage, consumer sentiment data is derived from the two publicly available datasets mentioned previously: the ChnSentiCorp and the yf_amazon. These data have undergone preliminary preprocessing, encompassing denoising, sentiment annotation, tokenization, and vectorization, ensuring the quality of the data input into the model.

In the model processing stage, the core of this study lies in the DL architecture based on the BERT model. The uniqueness of the BERT model lies in its bidirectional encoder design, which can simultaneously consider contextual information, thereby providing a more accurate sentiment understanding. In the sentiment analysis task, the input format is simplified to "[CLS] + Sentence + [SEP]". "[CLS]" is a classification marker indicating the beginning of the entire sentence, and "[SEP]" is used to separate sentences. In this task, the input data is a single review text rather than sentence pair data, ensuring that the sentiment features of each comment can be fully learned and extracted. The model processing mainly includes an embedding layer, multiple Transformer encoders, and a classification layer. The embedding layer converts the input text into a vector representation, the Transformer encoder models the contextual relationships in the text, and the classification layer outputs sentiment labels based on the encoded text representation. Through this structure, BERT can effectively capture the deep semantic information required for sentiment analysis, thus improving the sentiment classification performance of the model.

Firstly, the input vocabulary sequence is transformed into a vector representation in the embedding layer. BERT maps each vocabulary to a high-dimensional space through word embedding techniques to obtain semantic information. In this way, the model can capture the contextual relationships of vocabulary. In the subsequent multi-layer Transformer encoder, the core of BERT consists of multiple stacked Transformer encoder layers, each of which processes information through self-attention mechanisms and feedforward neural networks. The calculation process of the attention mechanism reads:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

Q, K, and V represent the query, key, and value vector of each word in the input text, respectively. These vectors calculate the correlation between each word and other words through the self-attention mechanism, to assign a weight to each word, and then extract the key information in the text. d_k refers to the dimension of the key vector, which plays a role in standardization and avoids numerical instability in the calculation process. After calculation, the softmax function normalizes the weighted value to obtain the importance of each word. Finally, it is combined with the value matrix V to output the weighted representation. Through the self-attention mechanism, the model can dynamically calculate the relevance of each word in the text, thereby extracting important features.

When processing long text sequences, BERT models and their variants frequently encounter the challenge of text length exceeding the model's maximum input limitation. The standard BERT architecture typically accepts a maximum of 512 tokens, though this study employs a reduced sequence length of 128 tokens. For input sequences exceeding this predetermined length threshold, a truncation strategy is implemented. Specifically, when text sequences surpass the 128-token limit, truncation occurs from the sequence's end. This is a methodological choice based on empirical observations that sentiment expressions often manifest early in textual

content, making tail truncation preferable for preserving emotional information. Conversely, text sequences shorter than the maximum length require padding to maintain uniform input dimensions across all samples. The zero-padding technique is applied, whereby empty token positions are filled with zeros until reaching the specified 128-token length. This standardization ensures consistent input sizes for efficient batch processing while maintaining the model's architectural requirements.

In each layer, after the self-attention calculation, the input is added to the current layer's output through a residual connection, which helps with information propagation and model stability. Subsequently, layer normalization is applied to each layer's output to reduce the issue of vanishing gradients during training. After processing all the Transformer encoders, the final output is the representation of each input text. To obtain the sentiment classification results, the vector corresponding to the "[CLS]" token is extracted as the final text feature and input into a fully connected layer. The classification layer employs a softmax function to map the output to different sentiment categories, as shown in Equation (2):

$$p(y = k|x) = \frac{e^{z_k}}{\sum_{j=1}^{C} e^{z_j}} \quad (2)$$

z_k represents the logits value output by the model and the predicted score of category k. Equation (2) converts the model's output value z_k into a probability distribution through the softmax function. $p(y = k|x)$ refers to the probability that the input text belongs to category k; C is the total number of categories; $\sum_{j=1}^{C} e^{z_j}$ denotes the normalized term of the scores of all categories, ensuring that the probability sum of all categories is 1. Through this process, the model maps the emotional tendency of the input text to the corresponding sentiment category. To this end, the cross-entropy loss function is used for model optimization, defined as Equation (3):

$$L = -\frac{1}{N}\sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \log (p_{ij}) \quad (3)$$

N stands for the total number of samples; C denotes the number of sentiment categories; y_{ij} represents the true label; p_{ij} means the probability predicted by the model. During the model training process, the Adam optimization algorithm is employed to dynamically adjust the learning rate, aiming to achieve faster convergence and superior model performance. In the specific training steps, key hyperparameters are set, encompassing the learning rate, batch size, and number of training epochs. In addition, the model's generalization ability is evaluated by the layered k-fold cross-validation method. In this process, the dataset is randomly divided into k subsets, in which each time the k-1 subset is used as the training set, and the remaining subset is the verification set. Different from the traditional k-fold cross-validation, the layered k-fold cross-validation ensures that the category distribution in each subset is consistent with the overall dataset. Especially in the sentient analysis task, this is particularly important for dealing with data imbalance.

3.3 Dynamic feedback mechanism and personalized marketing strategy

The dynamic feedback mechanism is also introduced to improve the personalized marketing recommendation's accuracy and responsiveness. This mechanism adjusts recommendation strategies in real-time, adapting to changes in consumer behaviors and emotions. By monitoring and analyzing consumers' emotional fluctuations and behavioral changes in real-time, the mechanism ensures that the recommended content aligns with the user's current needs and emotional state.

In practical applications, the sentiment analysis model first performs real-time sentiment classification on user feedback (including comments, ratings, click behaviors, etc.) to determine current emotional tendencies. When significant emotional shifts are detected (e.g., a transition from positive to negative sentiment), the recommendation system dynamically adjusts its strategy accordingly. For users exhibiting negative sentiments, the system prioritizes recommendations of products or services likely to improve their emotional state while filtering out items that might exacerbate this sentiment. Concurrently, the system emphasizes positively reviewed offerings to facilitate emotional state transition.

Behavioral data plays an equally critical role, with the system continuously monitoring patterns in user interactions - particularly changes in clicking, browsing, and purchasing behaviors. Behavioral pattern variations trigger adaptive recommendation adjustments. For instance, frequent clicks on specific product categories or demonstrated emotional preferences for particular items result in increased recommendation frequency for those categories. The system further integrates reinforcement learning algorithms to continuously optimize the recommendation model based on behavioral and emotional feedback. By implementing a state-action-reward framework, the system learns optimal recommendation strategies for various emotional and behavioral contexts, iteratively refining content to maximize user satisfaction and conversion rates. This dynamic feedback mechanism, combining real-time sentiment analysis with behavioral tracking, enables precise, adaptive content recommendations. The integrated approach allows the recommendation system to responsively accommodate fluctuating consumer emotions and evolving needs, enhancing user experience and strengthening platform engagement.

Based on the results of consumer sentiment analysis, this study designs a set of personalized marketing strategies to increase customer engagement and satisfaction. Details are suggested in Figure 3:

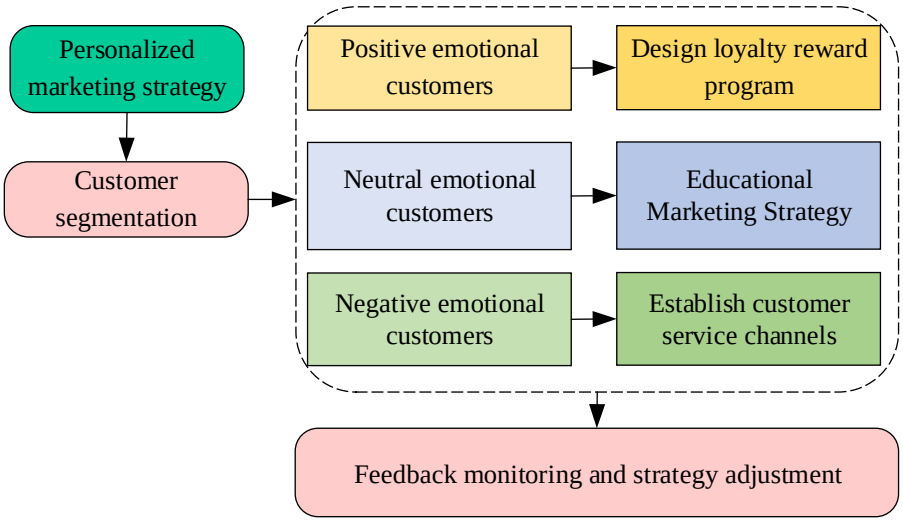


Figure 3: Personalized marketing strategy

Figure 3 demonstrates that consumers are initially classified into multiple segments through in-depth analysis of emotional data. This segmentation process relies on emotional tendencies identified by the sentiment analysis model, including positive, negative, and neutral emotions. At the same time, it incorporates more refined emotional labels such as joy, anger, and disappointment. These granular sentiment categories capture consumer sentiment diversity more precisely, thus enhancing emotion-based customer segmentation accuracy and facilitating more targeted marketing strategies. After the segmentation is completed, personalized marketing strategies are formulated according to the characteristics and needs of each subgroup to achieve more effective market promotion.

In the design process of personalized marketing strategy, priority should be given to the needs of customers with positive emotions, such as customers who show joy. These customers typically have a high satisfaction level with products and services and are inclined to share their positive experiences. For this group, enterprises can design loyalty reward programs, offer customized discounts and promotional activities, and encourage customers to share product experiences, facilitating brand loyalty and customer retention. By promoting these activities through social media platforms, enterprises can effectively increase brand exposure and customer engagement.

For customers with neutral sentiments, their feedback is often ambiguous, and they may not have high expectations for products or services. In this case, enterprises should adopt educational marketing strategies to increase customer trust and satisfaction by providing more information and support. Moreover, enterprises may employ email marketing campaigns or targeted advertisements to distribute product-related educational content, including detailed specifications and application guidelines. Such informational outreach enhances consumer product comprehension and purchase motivation while strategically incorporating promotional materials.

For customers exhibiting negative sentiments - particularly those labeled as angry or disappointed - prompt response and feedback management prove critical. Enterprises should implement effective customer care measures that extend beyond conventional service channels to improve satisfaction. Customized interventions like personalized product recommendations, immediate feedback collection, and improvement suggestions enable emotional guidance tailored to individual needs, achieving more focused marketing objectives. These measures directly address emotional requirements rather than merely resolving complaints, ultimately strengthening brand loyalty and conversion rates. Through this approach, customers with negative sentiments receive timely support and perceived brand value, thereby enhancing satisfaction and long-term engagement intentions.

When implementing personalized marketing strategies, enterprises fully utilize modern technological means, especially big data analysis and artificial intelligence technologies, to continuously monitor and evaluate the effectiveness of marketing activities. By conducting real-time analysis of marketing results, enterprises can adjust strategies promptly to adapt to the ever-changing market demands and customer preferences.

3.4 Data analysis and model evaluation

During the training and testing stages of the consumer sentiment analysis model, data analysis and model evaluation are crucial components. To ensure the validity and reliability of the model, this study employs a series of standards and metrics to assess model performance, including accuracy, recall, F1 score, and the confusion matrix, among others. The calculation can be written as Equations (4)-(7):

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{4}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{5}$$

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision}+\text{Recall}} \tag{6}$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (7)$$

True Positive (TP) and True Negative (TN) represent the number of samples correctly predicted as positive and negative class; False Positive (FP) and False Negative (FN) refer to the number of samples incorrectly predicted as positive and negative class.

The experimental design allocates 80% of data for training and 20% for model validation, ensuring generalization ability during training. Hyperparameter optimization employs both grid search and random search methods. For learning rates, multiple values (0.0001, 0.001, 0.01) are evaluated, with 0.001 selected as optimal due to its balanced convergence speed and avoidance of vanishing and exploding gradients. Batch size comparisons (16, 32, 64) identify 32 as delivering optimal accuracy-computation efficiency trade-offs. The Adam optimizer with a 0.001 learning rate demonstrates superior convergence and final performance.

Further refinements include dynamic learning rate adjustment - reducing rates by 0.5 after epoch 10 for fine-tuning while preventing overfitting. Moreover, an early stopping mechanism is introduced, which automatically stops training when the validation set's performance does not improve significantly within 5 consecutive epochs to ensure the model's generalization ability. Additional safeguards against overfitting incorporate L2 regularization and a 0.1 dropout rate, collectively enhancing model stability. The main parameters used for the consumer sentiment analysis model and their settings are exhibited in Table 2:

Table 2: Parameter setting

Parameter	Setting value
Model type	BERT, Transformer
Learning rate	0.001
Batch size	32
Maximum sequence length	128
Training epochs	10
Optimizer	AdamW
Loss function	Cross-entropy loss
Activation function	ReLU
Dropout rate	0.1
Random seed	42
Division of training and verification sets	80% for the training set; 20% for the verification set
Overfitting prevention technology	Early stopping, L2 regularization
Cross-validation method	Layered k-fold cross-validation
Padding and truncation strategies	Padding to the maximum sequence length of 128, exceeding partial truncation

In model training and evaluation, K-fold cross-validation is employed to improve the model's generalization ability. In K-fold cross-validation, the training set is divided into K equal-sized subsets. The model is trained K times, with each iteration selecting a different subset as the validation set and the remainder as the training set. The final model performance evaluation result is obtained by averaging the K validation results. This method effectively prevents the model from overfitting to a specific training set, thus improving its performance on new data. A confusion matrix visualizes the model's classification results, analyzing its performance across different sentiment categories. The confusion matrix analysis reveals the model's recognition accuracy across different sentiment categories during actual evaluation. This approach also identifies potential inter-category confusion patterns, providing targeted guidance for model refinement. Text that exceeds the maximum sequence length 128 is truncated. For short text, zero-padding is employed to ensure that all input sequences have the same length, which can effectively improve the BERT model's processing efficiency.

4 Results and discussion

4.1 Results of consumer sentiment analysis

In consumer sentiment analysis, an in-depth examination of sentiment classification results is conducted to identify the key factors influencing changes in consumer emotions. Through sentiment classification of the yf_amazon and ChnSentiCorp datasets, significant differences in the distribution of positive, negative, and neutral sentiments are observed. Specifically, the proportion of positive sentiment is notably high in reviews, particularly regarding service experience and product quality. Negative sentiments are primarily concentrated on negative experiences related to dissatisfaction with product quality and service delays. In addition, fine-grained sentiment labels, such as joy, anger, and disappointment, are introduced to refine consumers' emotional expression and can more accurately capture subtle changes in emotional fluctuations. The distribution of refined sentiment labels is suggested in Figure 4:

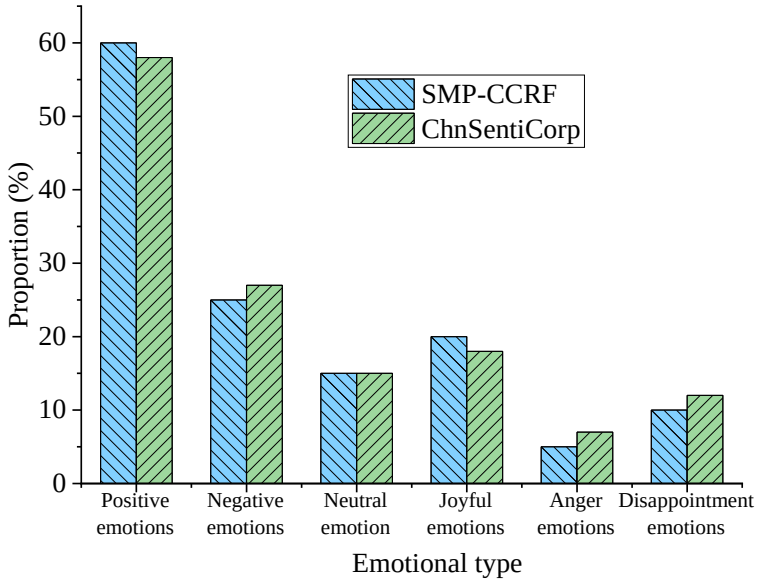


Figure 4: Results of consumer sentiment analysis

Figure 4 reveals that positive sentiments constitute the predominant proportion across both datasets, particularly in evaluations of product quality and service experience. This indicates that superior products and services significantly enhance consumer satisfaction. Negative sentiments primarily emerge from critical feedback regarding product defects and service delays, typically triggered by unsatisfactory shopping experiences. The relatively low proportion of neutral sentiments suggests most consumers tend to express more definitive emotional attitudes in their evaluations.

Within the fine-grained emotional labels, joy predominantly appears in positive assessments of product functionality and aesthetics, reflecting strong approval of need fulfillment. Anger mainly stems from strong reactions to substandard product quality or poor service attitudes. Disappointment closely correlates with discrepancies between consumer expectations and actual experiences, becoming particularly pronounced when products fail to meet anticipated standards. This granular sentiment classification enables precise identification of specific causes behind consumer sentiment fluctuations, thereby providing more accurate data support for personalized marketing strategy development. Brand image, product quality, and customer service demonstrate significant influence on consumer sentiments. Positive brand perception and high product quality consistently foster favorable sentiments, whereas inadequate service quality and negative reviews predominantly trigger adverse emotional responses.

4.2 Performance of the model

A 5-fold cross-validation is performed on the whole dataset to comprehensively evaluate the model's generalization ability. The dataset is divided into 5 subsets, with each taking turns as the validation set and the other 4 subsets as the training set. Based on this process, the performance results of 5 training and verification are obtained. Table 3 shows the results of K-fold cross-validation:

Table 3: The results of K-fold cross-validation

Fold	Accuracy	F1 score	Recall
1	91.5%	90.8%	89.3%
2	92.0%	91.2%	90.1%
3	91.8%	90.6%	89.8%
4	92.3%	91.5%	90.2%
5	91.9%	91.0%	90.0%

The average accuracy, F1 score, and recall of 5-fold cross-validation in Table 3 are 91.9%, 91.0%, and 89.9%. These results show that the model is stable under different data divisions. It has strong generalization ability, can identify positive samples well, and has a high recall. To further evaluate the model's performance in different populations, the dataset is divided into multiple subsets according to the characteristics of users such as age, gender, and region, evaluating the performance of each subset. Table 4 presents the model's accuracy, F1 score, recall, and precision in various groups.

Table 4: Performance evaluation results based on different population groups

Group	Accuracy	F1 score	Recall	Precision
Age (18-25years)	90.5%	89.3%	88.0%	91.0%
Age (26-35years)	92.0%	91.5%	90.3%	93.2%
Age (36-50 years)	91.2%	90.4%	89.6%	91.9%
Age (over 50)	89.8%	88.5%	87.2%	90.4%

Gender (Male)	91.7%	91.0%	90.1%	92.5%
Gender (Female)	90.6%	89.8%	88.5%	91.3%
Region (Urban)	92.5%	91.8%	90.7%	93.1%
Region (Rural)	89.6%	88.2%	87.0%	90.2%

Table 4 demonstrates notable variations in sentiment analysis performance across different age groups. Users aged 26-35 exhibit superior accuracy, F1 scores, and recall, indicating more consistent and explicit emotional expressions within this demographic. The slightly lower accuracy observed among users over 50 may stem from the more complex emotional language patterns characteristic of this age group. While performance metrics remain comparable between genders, male users show marginally higher accuracy and recall, potentially attributable to their tendency toward more direct emotional expressions in evaluations. Urban users outperform their rural counterparts in sentiment analysis tasks. This is likely due to more frequent social network engagement and standardized emotional expressions resulting from greater information exposure.

Figures 5 and 6 present comparative performance analyses across multiple models on both ChnSentiCorp and yf_amazon datasets. These models encompass the proposed model, LSTM, SVM, CNN, and models from references [26] and [27]. The evaluation incorporates standard metrics: F1 score, accuracy, and recall.

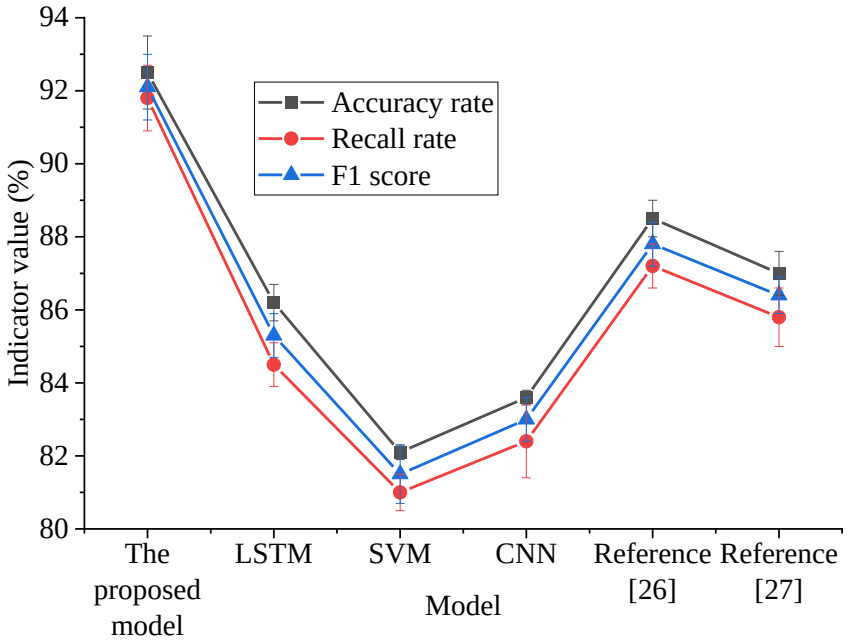


Figure 5: Performance comparison of different models (The ChnSentiCorp dataset)

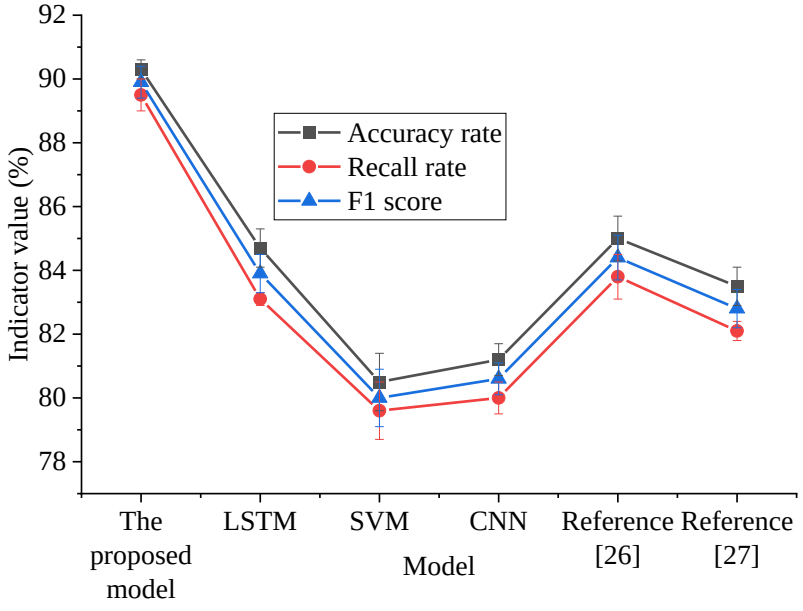


Figure 6: Performance comparison of diverse models (The yf_amazon dataset)

The comparative results in Figures 5 and 6 demonstrate that the proposed model consistently outperforms conventional models (LSTM, SVM, and CNN) across all evaluation metrics (accuracy, recall, and F1 score). This performance advantage holds consistently across both datasets (ChnSentiCorp and yf_amazon). Especially in the ChnSentiCorp dataset, the proposed model achieves 92.5% accuracy, significantly higher than LSTM (86.2%), SVM (82.1%), and CNN (83.6%). Similarly, on the yf_amazon dataset, the proposed model maintains superior performance with 90.3% accuracy compared to alternative models.

This performance enhancement primarily stems from the proposed model's BERT-based bidirectional self-attention mechanism. Traditional LSTM, SVM, and CNN models exhibit inherent limitations when processing textual data, particularly for sentiment analysis tasks involving long texts and complex dependencies. LSTMs, while effective at capturing sequential information, tend to lose critical semantic context in long-range dependencies. SVMs demonstrate competent classification performance but show constrained effectiveness for nonlinear problems. CNNs, despite their local feature extraction capabilities, struggle to capture global semantic patterns in text. In contrast, BERT's bidirectional attention mechanism simultaneously processes contextual information from both directions, enabling more precise sentiment classification judgments.

Furthermore, while references [26] and [27] incorporate data augmentation and other optimization techniques, they fail to overcome the fundamental limitations of LSTM, CNN, and SVM architectures in sentiment analysis. Reference [26] primarily employs word embedding methods to improve accuracy, but its reliance on static word vectors prevents adequate capture of contextual emotional nuances. Reference [27]'s hybrid approach combines traditional sentiment analysis models but lacks BERT's contextual comprehension capability, resulting in weaker performance for complex sentiment tasks. Overall, the proposed model's superior performance across both datasets derives from BERT's robust semantic understanding. Meanwhile, it effectively captures deep emotional relationships in text, excelling in long-text analysis and complex dependency tasks where conventional models show limitations.

To statistically validate performance differences, t-tests assess the significance of variations in accuracy, recall, and F1 scores between model pairs, with detailed results presented in Table 5.

Table 5: Statistical significance test results

Comparison model	p-value of accuracy	p-value of recall	P-value of F1 score
The proposed model vs LSTM	0.0012*	0.0013*	0.0011*
The proposed model vs SVM	0.0008*	0.0009*	0.0006*
The proposed model vs CNN	0.0024*	0.0021*	0.0020*
LSTM vs SVM	0.1459	0.1293	0.1385
LSTM vs CNN	0.2731	0.2837	0.2652
SVM vs CNN	0.4876	0.4325	0.4703

The statistical significance test results in Table 5 demonstrate that the performance differences between the proposed model and conventional models (LSTM, SVM, CNN) are statistically significant (p-values < 0.05). This shows that the proposed mode has superior performance across all evaluation metrics. Particularly in comparisons with LSTM, SVM, and CNN, the proposed model exhibits marked advantages in accuracy, recall, and F1 score. Furthermore, the performance differences between LSTM and SVM, LSTM and CNN, and SVM and CNN lack statistical significance (p-values > 0.05), providing additional evidence of the proposed model's relative superiority.

The confusion matrix can better show the model's performance in each emotion category. Table 6 lists the predictions of the different models on the positive, negative, and neutral categories.

Table 6: The confusion matrix (The proposed Model)

	Positive	Negative	Neutral	Total
Positive	680	30	20	730
Negative	25	750	15	790
Neutral	15	10	280	305
Total	720	790	315	1825

Table 6 reveals that the proposed model achieves optimal performance in sentiment classification, particularly for positive and negative categories, where correctly classified samples significantly outnumber other models. LSTM, SVM, and CNN models demonstrate considerable confusion between negative and neutral classifications. However, the proposed model maintains superior differentiation across all categories, albeit with relatively lower precision for neutral sentiment classification compared to positive and negative categories.

This performance gap likely reflects the inherent ambiguity of neutral emotional expressions. Comparative analysis of confusion matrices confirms the proposed model's enhanced discriminative capability across all three sentiment categories, demonstrating clear advantages in sentiment classification tasks.

Figure 7 displays the operational efficiency of each model, including the training and prediction time for each model:

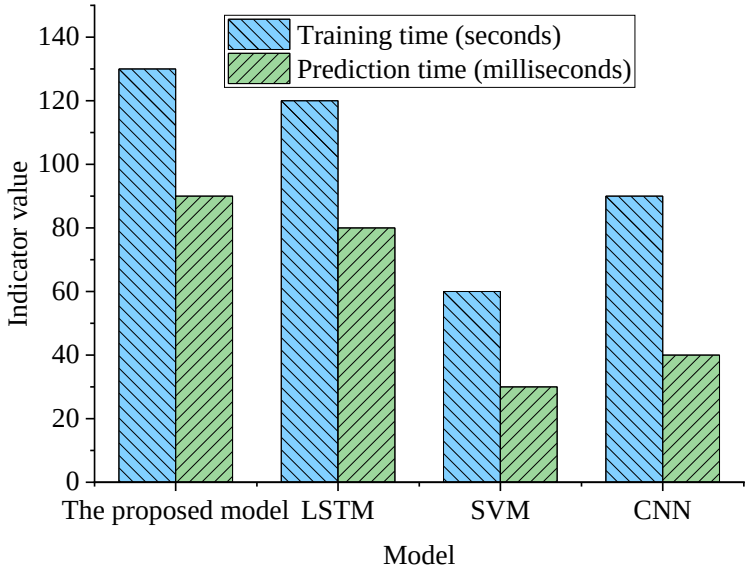


Figure 7: Comparison of operational efficiency of various models

Figure 7 suggests that the proposed model has a training time of 130 seconds, slightly longer than that of LSTM, SVM, and CNN. However, its prediction time reaches 90 milliseconds, indicating high real-time performance. The proposed model achieves efficient operation while maintaining good performance by optimizing data loading and batch processing strategies, as well as employing mixed-precision training. Compared to other models, the proposed model can provide accurate sentiment analysis results within a shorter prediction time, making it more suitable for real-time demands in practical applications. Table 7 summarizes the complexity of each model, encompassing the number of parameters and computational resource requirements:

Table 7: Comparison of model complexity

Model	The number of parameters	Computational resource requirements
The proposed model	30 M	Medium (can run on high-performance Central Processing Unit (CPU))
LSTM	20 M	Medium (can run on regular CPU)
SVM	1 M	Low (usable on regular computers)
CNN	15 M	Medium (requiring moderate optimization)

In Table 7, the number of parameters in the proposed model is 30M, with moderate computational resource requirements, making it suitable for operation on high-performance CPUs. Through optimization of the model structure and reduction of parameters, this study effectively lowers the model's complexity, allowing it to operate efficiently even in resource-constrained environments. Although LSTM, SVM, and CNN have lower parameter numbers and computational resource requirements, their performance in sentiment analysis does not match that of the proposed model. Consequently, in resource-limited scenarios, while other models may serve as alternative options, the proposed model achieves a good balance between accuracy and efficiency, making it suitable for a wider range of practical applications.

The computational cost is further explored to evaluate the feasibility and efficiency of the model in practical application. It includes the performance of training time, memory utilization, and inference time under different hardware settings. Experiments are conducted on the Graphics Processing Unit (GPU) and CPU, respectively. Table 8 shows the experimental results:

Table 8: Computational cost analysis under different hardware configurations

Item	GPU(Training/Inference)	CPU (Training/Inference)
Training time (per epoch)	12 minutes	45minutes
Maximum memory utilization	4 GigaByte (GB)	12GB
Inference time	200 milliseconds (ms)	800 ms

Table 8 reveals the substantial advantages of GPU over CPU in both model training and inference. Regarding

training efficiency, the GPU completes each epoch in merely 12 minutes compared to the CPU's 45 minutes, demonstrating significantly enhanced processing capability, especially for large-scale datasets. Memory utilization analysis shows the GPU's maximum memory consumption at 4GB, substantially lower than the CPU's 12GB requirement, indicating the GPU's superior suitability for memory-intensive DL tasks. Inference time comparisons highlight hardware impact on operational efficiency. GPU achieves 200ms inference latency versus CPU's 800ms, making GPU preferable for real-time applications. While the CPU remains functionally capable, the GPU demonstrates clear superiority in both computational efficiency and memory management for large-scale data processing and time-sensitive inference tasks. These findings strongly recommend GPU as the preferred hardware platform for accelerating both training and inference processes in practical implementations.

4.3 The application effects of personalized marketing strategies

This section analyzes the practical effects of personalized marketing strategies based on sentiment analysis, targeting consumers who have purchased products from a well-known e-commerce platform. By deeply mining and analyzing the sentiment data of these consumers, the marketing team can better understand their needs and preferences, thus formulating more precise marketing strategies. After implementing the personalized marketing strategies, feedback and behavioral changes from consumers are tracked and surveyed. The statistical results are revealed in Figure 8:

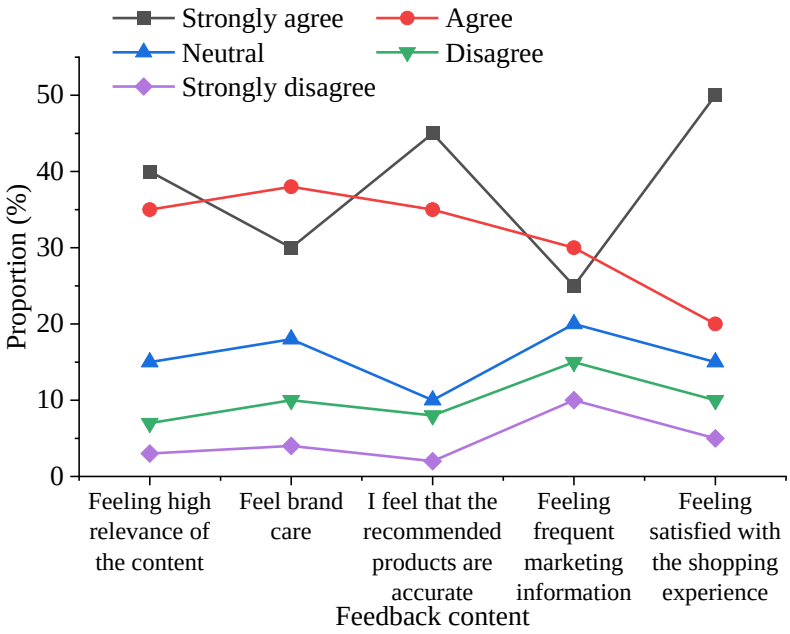


Figure 8: Statistics on consumer feedback level

Figure 8 provides a visual analysis of consumer feedback on personalized marketing strategies. In the feedback category “feeling high content relevance,” 40% of consumers indicate “strongly agree,” while 35% show “agree,” demonstrating a general recognition of the relevance of the recommended products. Regarding “feeling brand care,” 68% of consumers offer positive feedback, illustrating the effectiveness of the personalized care strategy. In the category “feeling accurate recommended products,” 45% of consumers state “strongly agree,” suggesting that the effectiveness of sentiment analysis is well validated in the recommendation system. Regarding feedback on "frequent marketing information," 25% of consumers select "strongly agree," indicating the need for careful adjustment of marketing information frequency to avoid negative consumer experiences caused by information overload. Consequently, personalized adjustment strategies for marketing information delivery frequency become essential to ensure information dissemination meets consumer needs without causing annoyance or resistance. The changes in customer conversion rates are depicted in Figure 9:

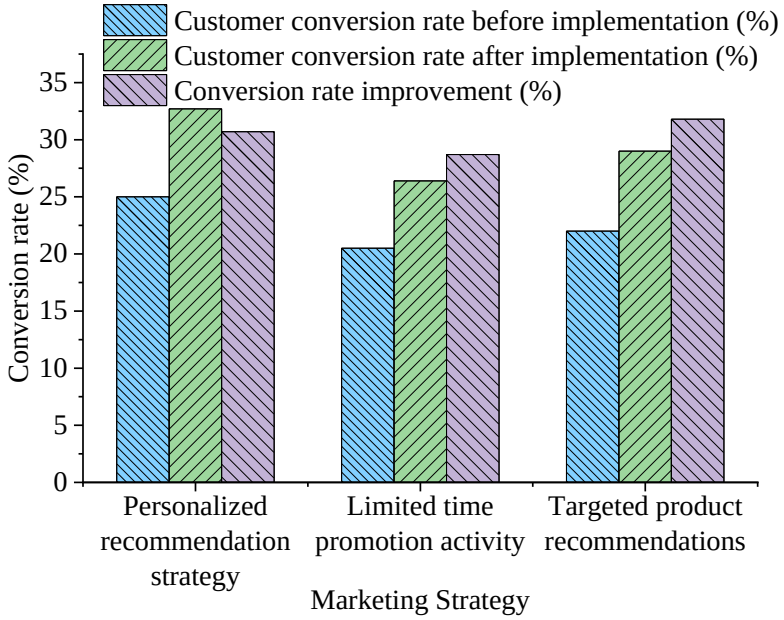


Figure 9: The changes in customer conversion rate

Figure 9 demonstrates that after implementing personalized recommendation strategies, customer conversion rates increase from 25.0% to 32.7%, representing a 7.7 percentage point improvement equivalent to approximately 30.8% relative growth. These results confirm that consumer sentiment analysis-based strategies effectively attract more potential customers. Other marketing strategies, such as limited-time promotional activities and targeted product recommendations, also show good conversion effects, enhancing the rates by 28.7% and 31.8%, respectively. However, these improvements may also be influenced by external factors. Market competition, seasonal variations, or fluctuations in consumer behavior could potentially impact conversion rates. Additionally, possible biases in data collection, such as uneven distribution of consumer preferences or regional market differences, may also affect the results. Hence, future research should investigate these potential influencing factors to enable a more accurate evaluation of personalized recommendation strategy effectiveness.

Overall, consumer feedback on personalized marketing strategies exhibits a positive trend, with these strategies effectively promoting consumer purchasing decisions and enhancing brand loyalty. However, some consumers still express concerns about the frequency of marketing messages. Enterprises should continuously optimize marketing content and strategies to maintain a positive consumer experience when implementing personalized marketing. By segmenting the degree of feedback, it is possible to more accurately identify consumer needs and preferences, offering data support for subsequent marketing strategy optimization.

4.4 Discussion

The research findings demonstrate that sentiment analysis-based personalized recommendation strategies yield significant improvements in customer conversion rates. Despite these encouraging performance gains, several potential external factors and biases may influence the results, warranting further investigation.

Market competition emerges as a critical factor affecting conversion rates. In highly competitive environments, rival companies' strategies, advertising campaigns, or promotional activities may temporarily alter customer purchasing decisions, thereby impacting conversion rates. While the current personalized recommendation strategy performs well in controlled settings, its effectiveness may diminish when confronted with real-world competitive pressures. Seasonal variations substantially influence purchasing behaviors. For instance, holiday promotions or seasonal discounts often increase customers' purchase likelihood during specific periods. Such cyclical fluctuations necessitate consideration since they may cause temporal variations in the strategy's effectiveness. A comprehensive evaluation of long-term performance requires repeated experiments across different timeframes to isolate seasonal effects on conversion rates.

Consumer behavior diversity represents another influential factor. Individual characteristics, including age, gender, purchasing habits, and cultural background, generate varied responses to personalized recommendations. These behavioral differences may produce uneven conversion rate improvements across demographic segments. Consequently, a uniform recommendation approach may not prove universally effective across all consumer groups, suggesting future research should explore segment-specific strategy optimization based on distinct user characteristics.

Moreover, potential biases in data collection processes warrant careful consideration. For instance, issues regarding the representativeness of consumer feedback may arise if sample data fail to accurately reflect the diversity of target populations, potentially leading to

skewed analytical outcomes. Subsequent research should employ more representative datasets and implement controls for potential biases during data acquisition to ensure the generalizability and reliability of findings. Although personalized recommendation strategies demonstrate significant efficacy in improving conversion rates, external factors and underlying biases may still substantially influence the results. Future investigations should incorporate more sophisticated data analysis and experimental designs to systematically examine these factors, thus validating and optimizing the effectiveness of personalized recommendation strategies.

The impact of domain-specific language on sentiment analysis models constitutes another critical consideration. Distinct industry sectors (e.g., electronics versus fashion products) exhibit unique terminologies, expression patterns, and consumer sentiment tendencies, potentially affecting model generalizability. This study utilizes multiple datasets (yf_amazon and ChnSentiCorp) encompassing different product categories (such as electronics and consumer goods), enabling a preliminary assessment of cross-domain performance. However, given inherent variations in linguistic characteristics and emotional expression patterns across industries, model performance may vary significantly. It potentially excels in certain domains (e.g., fashion reviews) while underperforming in others (e.g., electronics evaluations). These discrepancies primarily manifest in differential usage frequencies of sentiment-bearing words and contextual comprehension challenges. For example, fashion reviews frequently employ affective descriptors (e.g., "stylish," "elegant"), whereas electronics reviews emphasize technical specifications (e.g., "performance," "battery life"). To address these limitations, future research could implement domain adaptation techniques to enhance cross-industry robustness. Additionally, for domain-specific sentiment analysis tasks, researchers may refine training data to strengthen domain-aware linguistic representations, thus improving models' cross-domain adaptability.

5 Conclusion

This study combines NLP with DL technologies to explore the effective integration of consumer sentiment analysis and personalized marketing strategies, achieving a series of important research outcomes. By employing the yf_amazon and ChnSentiCorp datasets, an efficient sentiment analysis model is constructed, and personalized marketing strategies targeting consumer sentiment are designed. First, the study innovates in designing the sentiment analysis model, proposing a model based on BERT and Transformer architecture. Meanwhile, it optimizes parameter settings and training processes, thus markedly improving the accuracy of sentiment classification and the model's operational efficiency. Second, the study devises a systematic set of personalized marketing strategies based on consumer sentiment analysis results, capable of more precisely meeting consumer demands. Lastly, this study lays the foundation for future research on personalized marketing strategies based on sentiment data. It provides data support and theoretical guidance for enterprises' marketing decisions by analyzing the relationship between sentiment changes and consumer behavior. Although this study has achieved certain results, some limitations still exist. For example, the datasets used are mainly from specific domains, which may affect the model's generalization abilities. Therefore, future research could consider expanding data sources to cover more industries and different types of consumer feedback to improve the model's adaptability. Future research directions include enhancing the sentiment analysis model's adaptability and robustness, particularly for processing more complex and diverse emotional expressions. To better capture nuanced emotional variations in lengthy narrative-style reviews, multimodal data and richer emotional labels are incorporated to improve the model's recognition capability across different emotional dimensions. Furthermore, advanced explainable AI techniques, including Local Interpretable Model-agnostic Explanations and Shapley values, are explored to address model interpretability challenges in sentiment classification. These techniques clarify the model's decision-making logic while generating actionable feedback for business applications. Additional testing across diversified datasets and real-world business scenarios could further validate the model's generalization ability and cross-domain applicability, thus improving its reliability and effectiveness in practical implementations.

References

[1] Alantari, H. J., Currim, I. S., Deng, Y., & Singh, S. (2022). An empirical comparison of machine learning methods for text-based sentiment analysis of online consumer reviews. International Journal of Research in Marketing, 39(1), 1-19.

[2] Rocklage, M. D., He, S., Rucker, D. D., & Nordgren, L. F. (2023). Beyond sentiment: the value and measurement of consumer certainty in language. Journal of Marketing Research, 60(5), 870-888.

[3] Iqbal, A., Amin, R., Iqbal, J., Alroobaea, R., Binmahfoudh, A., & Hussain, M. (2022). Sentiment analysis of consumer reviews using deep learning. Sustainability, 14(17), 10844.

[4] Singh, A., Jenamani, M., Thakkar, J. J., & Rana, N. P. (2022). Quantifying the effect of eWOM embedded consumer perceptions on sales: An integrated aspect-level sentiment analysis and panel data modeling approach. Journal of Business Research, 138, 52-64.

[5] Yuan, Z. (2022). Big data recommendation research based on travel consumer sentiment analysis. Frontiers in Psychology, 13, 857292.

[6] Kaur, G., & Sharma, A. (2023). A deep learning-based model using hybrid feature extraction approach for consumer sentiment analysis. Journal of big data, 10(1), 5.

[7] Hossain, M. S., & Rahman, M. F. (2023). Customer sentiment analysis and prediction of insurance products' reviews using machine learning approaches. FIIB Business Review, 12(4), 386-402.

[8] Taherdoost, H., & Madanchian, M. (2023). Artificial intelligence and sentiment analysis: A review in competitive research. Computers, 12(2), 37.

[9] Adak, A., Pradhan, B., & Shukla, N. (2022). Sentiment analysis of customer reviews of food delivery services using deep learning and explainable artificial intelligence: Systematic review. Foods, 11(10), 1500.

[10] Shaik Vadla, M. K., Suresh, M. A., & Viswanathan, V. K. (2024). Enhancing product design through AI-driven sentiment analysis of Amazon reviews using BERT. Algorithms, 17(2), 59.

[11] Noorian, A., Harounabadi, A., & Hazratifard, M. (2024). A sequential neural recommendation system exploiting BERT and LSTM on social media posts. Complex & Intelligent Systems, 10(1), 721-744.

[12] Ghobakhloo, M., & Ghobakhloo, M. (2022). Design of a personalized recommender system using sentiment analysis in social media (case study: banking system). Social Network Analysis and Mining, 12(1), 84.

[13] El-Ansari, A., & Beni-Hssane, A. (2023). Sentiment analysis for personalized chatbots in e-commerce applications. Wireless Personal Communications, 129(3), 1623-1644.

[14] Gooljar, V., Issa, T., Hardin-Ramanan, S., & Abu-Salih, B. (2024). Sentiment-based predictive models for online purchases in the era of marketing 5.0: a systematic review. Journal of Big Data, 11(1), 107.

[15] Roumeliotis, K. I., Tselikas, N. D., & Nasiopoulos, D. K. (2024). Leveraging Large Language Models in Tourism: A Comparative Study of the Latest GPT Omni Models and BERT NLP for Customer Review Classification and Sentiment Analysis. Information, 15(12), 792.

[16] Çetinkaya, Y. M., Külah, E., Toroslu, İ. H., & Davulcu, H. (2024). Targeted marketing on social media: utilizing text analysis to create personalized landing pages. Social Network Analysis and Mining, 14(1), 77.

[17] Hung, L. P., & Alias, S. (2023). Beyond sentiment analysis: a review of recent trends in text based sentiment analysis and emotion detection. Journal of Advanced Computational Intelligence and Intelligent Informatics, 27(1), 84-95.

[18] Liu, H. (2024). The role of personalization in modern digital marketing: how tailored experiences drive consumer engagement. Strategic Management Insights, 1(8), 34-40.

[19] Yan, C., Liu, J., Liu, W., & Liu, X. (2022). Research on public opinion sentiment classification based on attention parallel dual-channel deep learning hybrid model. Engineering Applications of Artificial Intelligence, 116, 105448.

[20] Priya, C. S. R., & Deepalakshmi, P. (2023). Sentiment analysis from unstructured hotel reviews data in social network using deep learning techniques. International Journal of Information Technology, 15(7), 3563-3574.

[21] Tan, K. L., Lee, C. P., Anbananthen, K. S. M., & Lim, K. M. (2022). RoBERTa-LSTM: a hybrid model for sentiment analysis with transformer and recurrent neural network. IEEE Access, 10, 21517-21525.

[22] Tran, T. P., van Solt, M., & Zemanek Jr, J. E. (2020). How does personalization affect brand relationship in social commerce? A mediation perspective. Journal of Consumer Marketing, 37(5), 473-486.

[23] Karabila, I., Darraz, N., EL-Ansari, A., Alami, N., & EL Mallahi, M. (2024). BERT-enhanced sentiment analysis for personalized e-commerce recommendations. Multimedia Tools and Applications, 83(19), 56463-56488.

[24] Karbauskaitė, R., Sakalauskas, L., & Dzemyda, G. (2020). Kriging predictor for facial emotion recognition using numerical proximities of human emotions. Informatica, 31(2), 249-275.

[25] Kaminskas, V., & Vidugirienė, A. (2016). A comparison of Hammerstein-type nonlinear models for identification of human response to virtual 3D face stimuli. Informatica, 27(2), 283-297.

[26] Märtin, C., Bissinger, B. C., & Asta, P. (2023). Optimizing the digital customer journey—Improving user experience by exploiting emotions, personas and situations for individualized user interface adaptations. Journal of Consumer Behaviour, 22(5), 1050-1061.

[27] Gao, Y., & Liu, H. (2023). Artificial intelligence-enabled personalization in interactive marketing: a customer journey perspective. Journal of Research in Interactive Marketing, 17(5), 663-680.

Joint Global-Local Feature Alignment With Fine-Tuned Pretrained Transformers for Text-Based Person Search

Thi Thanh Thuy Pham[1], Huong-Giang Doan[2,*]
[1]Faculty of Cybersecurity and High Tech Crime Prevention - Academy of People Security, No. 125, Tran Phu, Van Quan, Ha Noi, Vietnam
[2]Faculty of Control and Automation - Electric Power University, No. 235, Hoang Quoc Viet, Nghia Do, Ha Noi, Vietnam
E-mail: thanh-thuy.pham@mica.edu.vn, giangdth@epu.edu.vn
*Corresponding author

Keywords: Text-based person search, transformer, contrastive image-text pretraining, global-local feature alignment

Received: January 2, 2025

Text-based person search (TBPS) aims to retrieve person images from a database using natural language description. Although significant progress has been made, TBPS remains challenging due to the complexities of cross-modal understanding. This work proposes a novel framework named ***GLAlign*** *that jointly aligns global and local features from both vision and text modalities using large-scale, pretrained, fine-tuned transformers. Specifically, we utilize ViT-B/32 for visual encoding and GPT-2 (English) or PhoBERT (Vietnamese) for textual encoding. To enhance alignment, we perform human part parsing and noun phrase extraction, enabling fine-grained local feature correspondence between body regions and descriptive attributes. The proposed method is evaluated on four benchmark datasets: CUHK-PEDES, CUHK-PEDES-VN, 3000VnPerson-Search, and 3000Vn-V2E. In the CUHK-PEDES dataset, our model achieves a Rank-1 accuracy of 80.75%, outperforming state-of-the-art methods such as PLOT (75.28%) and RaSa (76.51%). On the 3000VnPerson-Search dataset, our model reaches a Rank-1 accuracy of 85.72% for Vietnamese descriptions, indicating its robustness across both high-resource and low-resource languages. These results demonstrate the effectiveness of combining global-local alignment with fine-tuned vision language transformers for the TBPS task. The source codes are available at: https://github.com/TextBasedPersonSearch/PersionSearch*

Povzetek: Predstavljeno je TBPS ogrodje, ki združuje globalno in lokalno poravnavo slik in besedil z velikimi predtreniranimi transformatorji. Novost je v ujemanju delov telesa z NP frazami, kar omogoča bolj kvalitetno iskanje kot metoda ViTAA.

1 Introduction

Text-based person search (TBPS) is the task of retrieving person images from a large-scale gallery based on free-form natural language descriptions. This problem arises in practical scenarios such as security surveillance and social media moderation, where users or analysts may describe a target individual verbally rather than having access to an exemplar image. TBPS plays a critical role in enabling human-centered retrieval systems under such real-world constraints.

Compared to conventional image-based person re-identification, TBPS poses additional challenges due to the modality gap between textual and visual inputs. Natural language descriptions are often ambiguous, context-dependent, or linguistically diverse, and visual appearances can vary significantly across different camera views or environments. Effective retrieval thus requires fine-grained semantic reasoning that captures both global contextual meaning and local attribute-level correspondences.

Several deep neural networks (DNNs) have been proposed for the TBPS problem. For visual feature extraction, common models are CNN-based networks, such as VGG16 [1], ResNet50 [2], ResNet101 [3]. For extracting textual features, RNN-based networks such as LSTM or BiLSTM [4], [5] are popularly utilized. Recent advances in multimodal learning have led to powerful vision language models such as CLIP [6] and BLIP [7], which enable strong global alignment between images and texts through large-scale pre-training. However, these models often overlook fine-grained visual-textual associations - for example, matching detailed phrases like *'red sneakers'* with corresponding regions in the image. Moreover, the performance of such models under multilingual conditions, particularly in low-resource languages such as Vietnamese, remains underexplored.

This work is motivated by the need to enhance both the accuracy and interpretability of TBPS systems by integrating global and local features, and by leveraging pretrained transformer models in multilingual settings. To that end, we pose the following research questions:

– **RQ1**: Can large-scale pretrained transformer models

be effectively fine-tuned for joint global and local feature alignment in TBPS?

- **RQ2**: How does the incorporation of noun phrase–to–body-part alignment affect retrieval performance?
- **RQ3**: How does the use of multilingual textual descriptions (e.g., Vietnamese vs. English) influence retrieval accuracy, and what are the implications of using machine-translated versus native corpora?

To address these questions, we propose a unified framework named **GLAlign** that fine-tunes pretrained transformers for global-local alignment in TBPS. The image features are extracted using ViT-B/32, while the text is encoded using GPT-2 for English and PhoBERT for Vietnamese. Human parsing is used to segment person images into semantic body regions (e.g., head, upper body, lower body), which are then associated with corresponding noun phrases identified in the textual descriptions.

We conduct experiments on four datasets, including both English and Vietnamese corpora: CUHK-PEDES, CUHK-PEDES-VN, 3000VnPerson-Search, and 3000Vn-V2E. Results show that our model achieves state-of-the-art performance across multiple settings, and is robust to both cross-lingual and cross-dataset variations.

The remainder of this paper is organized as follows: Section 2 reviews related work. Section 3 presents the proposed framework. Section 4 details the experiments and results. Section 5 provides in-depth discussion, and Section 6 concludes the paper.

2 Related work

Text-based person search (TBPS) is a cross-modal retrieval task that involves retrieving images of a target person from a gallery based on a natural language description. Existing approaches to TBPS can be broadly grouped into three categories: (i) global visual-textual alignment, (ii) global-local feature alignment, and (iii) zero-shot or fine-tuned retrieval using large-scale pretrained vision-language models.

Global alignment

Early TBPS methods emphasized the global alignment between the overall image and the accompanying textual description. For example, GNA-RNN [8] employed VGG16 for visual encoding and LSTM for sentence representation, with joint embeddings learned through attention mechanisms. Zheng et al. [15] proposed the dual-path CNN, which uses two ResNet-50 networks, one for images and one for text, and introduces instance-level and ranking losses to enhance discriminability. However, these CNN-RNN-based methods are limited in their ability to capture long-range semantic dependencies and complex compositional language structures.

Global-local alignment

To enable more fine-grained semantic matching, subsequent studies focused on combining global and local features. Wang et al. [4] introduced the ViTAA (Visual-Textual Attributes Alignment) model, which associates image regions with attribute-level noun phrases using attention-based matching, improving both interpretability and retrieval accuracy. SCRF [9] proposed a correlation filtering mechanism to enhance word-to-region alignment. Although effective, these methods still rely heavily on convolutional and recurrent backbones, which limit their capacity for generalized multimodal reasoning. Traditional techniques in automatic text analysis [16] also emphasized the importance of structured semantic representations, which remain relevant for improving alignment in multimodal settings.

Pretrained vision-language models

Recent advances in large-scale multimodal pretraining have led to powerful transformer-based vision-language models, such as CLIP [6], BLIP [7], Florence-2 [17], and ALBEF [18]. These models learn global alignment from massive image-text corpora and can be directly adapted to TBPS in zero-shot or fine-tuned settings. More recently, IRRA [10] and RaSa [12] leveraged transformer-based backbones to achieve state-of-the-art performance on CUHK-PEDES.

Despite their strong global representations, these models generally lack explicit local alignment mechanisms. Most of them operate on full-image and full-sentence embeddings without decomposing into semantic parts or aligning textual attributes with specific body regions. Furthermore, there remains limited exploration of low-resource languages such as Vietnamese, where syntactic structures and cultural expressions can differ significantly from English, posing additional challenges to cross-lingual generalization [19]. Recent research in multilingual sentiment analysis and aspect-based opinion mining [20] highlights the necessity of semantic-aware modeling when handling language diversity.

Our contribution

To address these limitations, we propose a novel framework named **GLAlign** that performs joint global-local alignment using fine-tuned pre-trained transformers. We use ViT-B/32 [21] for visual encoding and GPT-2 [22] or PhoBERT [23] for textual encoding. Our method leverages human parsing and noun phrase extraction to associate specific body parts with descriptive phrases, enabling fine-grained matching. Furthermore, we evaluate our model not only on CUHK-PEDES (English), but also on Vietnamese datasets including CUHK-PEDES-VN and 3000VnPerson-Search.

Table 1 summarizes representative methods in terms of feature types, backbones, datasets, languages, and Rank-

Table 1: Comparison of TBPS methods in terms of feature types, backbones, datasets, and Rank-1 performance

Method	Feature Type	Backbone (Image/Text)	Dataset	Language	Rank-1 (%)
GNA-RNN [8]	Global	VGG16 / LSTM	CUHK-PEDES	English	19.71
ViTAA [4]	Global + Local	ResNet-50 / Bi-LSTM	CUHK-PEDES	English	55.97
SCRF [9]	Global + Local	ResNet-50 / BERT	CUHK-PEDES	English	64.04
IRRA [10]	Global	CLIP-ViT-B/16 / CLIP-Xformer	CUHK-PEDES	English	73.38
PLOT [11]	Global + Part Attention	CLIP-ViT-B/16 / CLIP-Xformer	CUHK-PEDES	English	75.28
RaSa [12]	Global + Relational Emb.	ALBEF / ALBEF	CUHK-PEDES	English	76.51
GLAlign (Ours)	**Global + Local**	**ViT-B/32 / GPT-2**	CUHK-PEDES	English	**80.75**
Method in [13]	Global + Local	ResNet-50 / Bi-LSTM	CUHK-PEDES-VN	Vietnamese	52.40
Method in [14]	Global	ResNet-50 / ResNet-50	3000VnPerson-Search	Vietnamese	33.07
GLAlign (Ours)	**Global + Local**	**ViT-B/32 / PhoBERT**	CUHK-PEDES-VN	Vietnamese	**77.72**
GLAlign (Ours)	**Global + Local**	**ViT-B/32 / PhoBERT**	3000VnPerson-Search	Vietnamese	**85.72**

1 performance. Our approach achieves superior results across both English and Vietnamese benchmarks, highlighting its effectiveness and generalizability across languages.

3 Proposed framework

The proposed framework of **GLAlign** method is shown in Fig. 1. The pre-trained transformer models are fine-tuned on image and text pairs of the TBPS datasets. The global and local features of the images and descriptions are extracted and aligned from the fine-tuned models.

3.1 Global feature embeddings

Global feature encoding is implemented for both the image and text. For the vision encoder, we closely follow the implementation of the ViT-B/32 architecture in [6] as a visual backbone network. This image encoder is derived from the original in [21] with minor modifications in the addition of an additional layer normalization to the combined patch and position embeddings before the transformer and uses a slightly different initialization scheme. The textual backbone encoder for English descriptions is the GPT-2, a pre-trained text transformer introduced in the OpenAI technical [22], and for Vietnamese descriptions it is the PhoBERT pre-trained model [23].

- **Global visual feature embedding:** The input image $\mathbf{I} \in \mathbb{R}^{H\times W\times C}$ represents an image with height H, width W, and C color channels (e.g. RGB), which is resized to 224×224 and then divided into a sequence of N patches, where $N = \frac{H}{P} \times \frac{W}{P}$ and P is the patch size (32×32 pixels). Each patch is flattened into a vector $\mathbf{p}_i$ of size $P^2 \times C$. Each flattened patch $\mathbf{p}_i \in \mathbb{R}^{P^2\times C}$ is then passed through a linear projection (a fully connected layer) to map it to a lower-dimensional embedding space. This yields a sequence of patch embeddings. Let $\mathbf{E}$ be the sequence of patch embeddings: $\mathbf{E} = \{\mathbf{e}_1, \mathbf{e}_2, .., \mathbf{e}_N\}$, where each embedding $\mathbf{e}_i \in \mathbb{R}^D$ and D is the dimensionality of the embedding space. The linear projection can be written as Eq. (1) as follows:

$$\mathbf{e}_i = \mathbf{W}_p \cdot \mathbf{p}_i + \mathbf{b} \tag{1}$$

where $\mathbf{W}_p \in \mathbb{R}^{D\times(P^2\times C)}$, $\mathbf{b} \in \mathbb{R}^D$.

The next step is adding learnable positional encodings to the patch embeddings. This allows the model to understand the relative or absolute positions of the patches within the image. The positional encoding $\mathbf{PE}_i \in \mathbb{R}^D$ for each patch is added element-wise as shown in the Eq. (2) as follows:

$$\mathbf{e}_i^{\text{pos}} = \mathbf{e}_i + \mathbf{PE}_i \tag{2}$$

The positional encodings $\mathbf{PE}_i$ are learned or fixed sinusoidal functions that provide a unique position for each patch.

Before feeding the combined patch and positional embeddings into the transformer, we apply Layer Normalization (LN) on $\mathbf{e}_i^{\text{pos}}$ as follows:

$$\mathbf{e}_i^{\text{norm}} = \text{LN}\left(\mathbf{e}_i^{\text{pos}}\right) \tag{3}$$

A special [**CLS**] token is then added at the beginning of the patch embedding sequence $\mathbf{E}_{\text{norm}} = \{\mathbf{e}_1^{\text{norm}}, \mathbf{e}_2^{\text{norm}}, .., \mathbf{e}_N^{\text{norm}}\}$ resulting in the final transformer input:

$$\mathbf{X}_{\text{input}} = \text{concat}\left([\mathbf{CLS}\,], \mathbf{E}_{\text{norm}}\right) \tag{4}$$

where [CLS] token is a learnable parameter and serves as the representative feature of the entire image, aggregating information from all patches; $\mathbf{X}_{\text{input}} \in \mathbb{R}^{(N+1)\times D}$ is the final sequence of patch embeddings, including the [**CLS**] token.

$\mathbf{X}_{\text{input}}$ is then passed through the transformer encoder layers. Each transformer encoder layer consists of multi-head self-attention (MHSA) and feed-forward network (FFN). MHSA mechanism allows each patch to take care of all other patches in the image, capturing long-range dependencies. FFN is a position-wise feed-forward network applied to each token independently. The output of each transformer encoder layer is passed to the next encoder layer. After passing through the Transformer encoder layers, the output sequence $\mathbf{X}_{\text{output}}$ contains information for all patches as well as the [**CLS**] token. The output of the [CLS] token from the final encoder layer is used as the final global image embedding. This output vector is considered to be the encoded feature of the entire image as shown in Eq. (5) as follows:

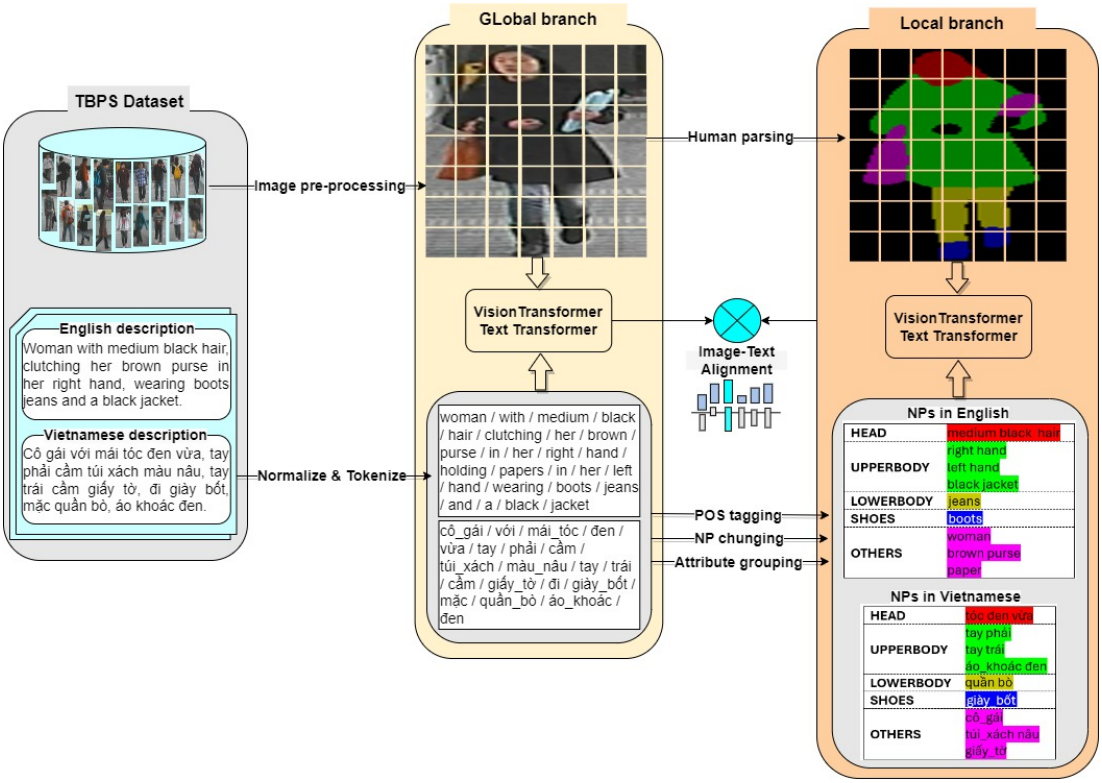


Figure 1: The overall framework for global and local feature learning using pre-trained CLIP model and fine-tuning on text-based person search dataset of CUHK-PEDES dataset.

$$\mathbf{z}_{\text{img}}^{(glob)} = \mathbf{X}_{\text{output}}[0] \tag{5}$$

where $\mathbf{X}_{\text{output}}[0] \in \mathbb{R}^{(N+1)\times D}$ corresponds to the feature vector of the [**CLS**] token; $\mathbf{z}_{\text{img}}^{(glob)}$ is the global image embedding representing the entire image after it was processed by the Vision Transformer. The global visual embedding $\mathbf{z}_{\text{img}}^{(glob)}$ is normalized using the normalization $L2$, to make it easier to compare with the global textual embedding.

- Global textual feature embedding: The input text **T** is first tokenized in a sequence of n tokens. Each token t_i is assigned to an embedding vector $\mathbf{e}_i$. The embedding for a token is obtained by multiplying the token index with a learned embedding matrix $\mathbf{W}_{txt}$ as shown in Eq. (6) follows:

$$\mathbf{e}_i = \mathbf{W}_t \cdot t_i \tag{6}$$

where $\mathbf{W}_t$ is the token embedding matrix of size $V \times D$, where V denotes the vocabulary size and D is the embedding dimension. The result of this step is a sequence of token embeddings $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, .., \mathbf{e}_n]$. The next step is adding positional encodings to the token embeddings to provide information about the positions of tokens in the sequence. The positional encoding for token t_i is denoted a $\mathbf{PE}_i$, and the final embedding for token t_i is $\mathbf{e}_i^{\text{pos}} = \mathbf{e}_i + \mathbf{PE}_i$. This results in a sequence of embeddings with positional information: $\mathbf{E}^{\text{pos}} = \left[\mathbf{e}_1^{\text{pos}}, \mathbf{e}_2^{\text{pos}}, .., \mathbf{e}_n^{\text{pos}}\right]$. The sequence $\mathbf{E}^{\text{pos}}$ passed through the transformer layers to obtain the output embeddings $h_1^{(L)}, h_2^{(L)}, .., h_N^{(L)}$, with L denoting the number of transformer layers and n the number of tokens in the input. The last token in the sequence is used as the final text embedding: $\mathbf{z}_{txt}^{(global)} = h_N^{(L)}$. The global textual embedding $\mathbf{z}_{\text{txt}}^{(glob)}$ is normalized using the normalization $L2$, to make it easier to compare with the global visual embedding.

3.2 Local feature learning

Local feature learning in the context of TBPS aims to associate specific parts of the human body in the image with corresponding noun phrases (NPs) extracted from the textual description. In our approach, we define five semantic body regions: **HEAD**, **UPPERBODY**, **LOWERBODY**, **SHOES**, and **OTHERS**. These regions are obtained using a human parsing network as in [4], which segments the person image into semantically meaningful parts. Examples of parsing results are shown in Fig. 2.

To associate textual descriptions with visual body regions, we extract noun phrases using the spaCy toolkit and map them to body parts using predefined keyword lists in both English and Vietnamese. The mapping rules are as follows:

– **HEAD**:

  – **English keywords**: ”hair”, ”head”, ”face”,



Figure 2: Examples of human parsing images from CUHK-PEDES dataset (five images in the first block) and 3000VnPerson-Search dataset (five images in the right block). The first row shows the original images and the second row depicts the corresponding human parsing images.

”eyes”, ”ear”, ”mouth”, ”nose”, ”teeth”, ”beard”, ”cheeks”, ”bald”, ”chin”, ”eyebrows”, ”lips”, ”spectacles”, ”mustache”, ”goatee”.

- **Vietnamese keywords**: ”tóc”, ”mặt”, ”mắt”, ”mũi”, ”miệng”, ”răng”, ”tai”, ”râu”, ”ria”, ”kính”, ”mép”, ”cằm”, ”má”, ”hói”, ”lông_mày”, ”lông_mi”.

– **UPPERBODY**:

- **English keywords**: ”shirt”, ”t-shirt”, ”jacket”, ”hoodie”, ”sweater”, ”blouse”, ”coat”, ”vest”, ”cardigan”, ”top”, ”jumper”, ”shoulders”, ”chest”, ”arms”, ”biceps”, ”back”, ”neck”, ”robe”, ”cloak”, ”outerwear”.
- **Vietnamese keywords**: ”áo”, ”vai”, ”ngực”, ”tay”, ”lưng”, ”cổ”.

– **LOWERBODY**:

- **English keywords**: ”pants”, ”jeans”, ”trousers”, ”leggings”, ”shorts”, ”skirt”, ”denim”, ”dress”, ”hips”, ”thighs”, ”calves”.
- **Vietnamese keywords**: ”quần”, ”chân”, ”váy”, ”mông”, ”đùi”.

– **SHOES**:

- **English keywords**: ”shoes”, ”sneakers”, ”boots”, ”sandals”, ”slippers”, ”loafers”, ”heels”, ”socks”.
- **Vietnamese keywords**: ”giày”, ”dép”, ”tất”.

– **OTHERS**:

- **English keywords**: ”bag”, ”purse”, ”watch”, ”umbrella”, ”phone”, ”tie”, ”glasses”, ”jewelry”, ”camera”.
- **Vietnamese keywords**: ”túi”, ”cặp”, ”ví”, ”đồng_hồ”, ”ô”, ”điện_thoại”, ”cà_vạt”, ”kính”, ”trang_sức”, ”máy_ảnh”.

Figure 1 illustrates how the extracted NPs (in both English and Vietnamese) are aligned with each semantic region. For instance, ”medium black hair” (”tóc đen vừa”) is mapped to HEAD; ”black jacket”(”áo khoác đen”) to UPPERBODY; ”jeans”(”quần bò”) to LOWERBODY; and so on.

While this rule-based approach is lightweight and interpretable, it has limitations. First, the mapping is inherently *language-dependent*, relying on fixed keyword lists that may not capture paraphrases, figurative language, or culturally specific expressions. This challenge is particularly relevant in low-resource languages like Vietnamese, where lexical diversity can reduce coverage consistency. We provide further discussion of these limitations in Section 5.

To evaluate the robustness of our NP-body-part mapping strategy, we conducted a coverage analysis using the 3000VnPerson-Search dataset. Table 2 shows the total and matched noun phrases per category. The HEAD and UPPERBODY parts achieve high coverage (>85%), while OTHERS shows more variability due to accessory and object descriptions.

This analysis demonstrates the utility of the rule-based strategy in covering core regions. However, the variation observed in OTHERS suggests future extensions could include learned semantic-aware mappings to increase robustness and linguistic generalization.

Finally, the matched NPs and their corresponding image regions are used in the local feature alignment module, leveraging the same ViT-B/32 and GPT-2 encoders used in the global branch. A contrastive loss is applied to optimize region-to-phrase alignment in a joint embedding space.

Table 2: NP-to-body-part mapping coverage by category on the 3000VnPerson-Search dataset

Body Part	Total NPs	Matched	Coverage (%)
HEAD	243	217	89.3
UPPERBODY	398	348	87.4
LOWERBODY	366	310	84.7
SHOES	155	129	83.2
OTHERS	271	190	70.1
Overall	1433	1194	83.3

3.3 Joint learning of image and text

Join learning of image and text is done in an end-to-end manner with the cross-entropy loss function being used to compare the cosine similarities between the image and text embeddings (logits) at both global and local branches. The output is typically used in symmetric loss (image-to-text and text-to-image), and the cross-entropy loss is calculated along both dimensions (image-to-text and text-to-image).

Considering the global feature embedding, we have the Cross Entropy Loss for Image-to-Text as shown in Eq. (7) as follows:

$$\mathcal{L}_{img}^{\text{global}} = -\sum_{i=1}^{N} \log \left(\frac{\exp\left(\text{logits}_{i,i}\right)}{\sum_{j=1}^{N} \exp\left(\text{logits}_{i,j}\right)} \right) \quad (7)$$

where $\text{logits} = \left(\mathbf{z}_{img} \cdot \mathbf{z}_{txt}^{T}\right) \cdot \exp(t)$; $\left(\mathbf{z}_{img} \cdot \mathbf{z}_{txt}^{T}\right)$ is the cosine similarity between the normalized image and text embedding vectors; $\exp(t)$ is applied element-wise to scale the logits, making the similarity values sharper; $\exp\left(\text{logits}_{i,i}\right)$ is the similarity of the image with the correct matching text (positive class); $\sum_{j=1}^{N} \exp\left(\text{logits}_{i,j}\right)$ is the sum of exponentiation logits over all possible text embeddings (this accounts for all possible text candidates). The temperature parameter t is a scalar hyperparameter used to scale the similarity logits. A smaller t produces a sharper probability distribution, emphasizing high-confidence matches. In our implementation, we set $t = 0.07$, following standard practice in contrastive learning frameworks.

Similarly, the cross-entropy loss for text-to-image in the global branch is calculated as shown in Eq. (8) as follows:

$$\mathcal{L}_{txt}^{\text{global}} = -\sum_{i=1}^{N} \log \left(\frac{\exp\left(\text{logits}_{i,i}\right)}{\sum_{j=1}^{N} \exp\left(\text{logits}_{i,j}\right)} \right) \quad (8)$$

The symmetric cross-entropy loss for the global branch is the average of the two loss elements: a global loss function of image-to-text ($\mathcal{L}_{img}^{\text{global}}$) and a global loss function of text-to-image ($\mathcal{L}_{txt}^{\text{global}}$) as illustrated in Eq. (9) as follows:

$$\mathcal{L}_{symCE}^{\text{global}} = \frac{1}{2}\left(\mathcal{L}_{img}^{\text{global}} + \mathcal{L}_{txt}^{\text{global}}\right) \quad (9)$$

Similarly, we have the symmetric cross-entropy loss for local branch, it is also composed of two local loss functions as shown in Eq. (10) as follows::

$$\mathcal{L}_{symCE}^{\text{local}} = \frac{1}{2}\left(\mathcal{L}_{img}^{\text{local}} + \mathcal{L}_{txt}^{\text{local}}\right) \quad (10)$$

Finally, the total loss function is combined of the global loss part and the local loss part as shown in Eq. (11) as follows:

$$\mathcal{L}_{total} = \alpha \cdot \mathcal{L}_{symCE}^{\text{global}} + (1-\alpha) \cdot \mathcal{L}_{symCE}^{\text{local}} \quad (11)$$

Where α is a hyper-parameter that controls the weight of the global versus local features.

4 Experiments and results

4.1 Dataset and configuration parameters

4.1.1 Experimental dataset

In this work, we used four datasets for text-based person search experiments in English and Vietnamese description languages. These datasets are CUHK-PEDES [8], CUHK-PEDES-VN [13], 3000VnPersonSearch [14], and 3000Vn-V2E [24]. Table 3 summarizes these datasets and their main characteristics.

CUHK-PEDES-VN is the Vietnamese-translated version of the CUHK-PEDES dataset [8], which is the first large-scale dataset for text-based person search. CUHK-PEDES contains 40,220 images of 13,003 individuals collected from various person re-identification datasets. Each image is annotated with two natural language descriptions written in English by independent workers on Amazon Mechanical Turk (AMT), resulting in a total of 80,440 descriptions. These descriptions are typically long, diverse in vocabulary, and contain minimal repetition.

To support Vietnamese-language experiments in text-based person search, all English descriptions were translated into Vietnamese using the Google Translate API, without altering the original image-description pairings. This translation forms the CUHK-PEDES-VN dataset, which retains 80,440 Vietnamese descriptions. The average description length is 26.10 words, with a standard deviation of 9.83; the longest description contains 115 words and the shortest 9 words. Furthermore, the noun phrases (NP) extracted from the original English descriptions were also translated into Vietnamese for use in NP-to-body-part mapping experiments in this study.

Table 3: Summary of datasets used in the experiments

Dataset	Language	#Person IDs	#Images	#Descriptions	Avg. Length	Notes
CUHK-PEDES	English	13,003	40,220	80,440	∼26.10 words/tokens	Two English descriptions per image written by AMT workers
CUHK-PEDES-VN	Vietnamese	13,003	40,220	80,440	∼26.10 words/tokens	Vietnamese version of CUHK-PEDES, translated via Google Translate
3000VnPerson-Search	Vietnamese	3,000	6,302	12,602	∼30.02 words/tokens	Collected from surveillance videos; annotated by Vietnamese speakers
3000Vn-V2E	English	3,000	6,302	12,602	∼30.02 words/tokens	English translation of 3000VnPerson-Search using Google Translate

To ensure correct evaluation and avoid data leakage, we strictly follow a non-overlapping person ID split between the training and test sets across all datasets. Specifically, for the CUHK-PEDES dataset, we use the standard split of 11,003 person IDs for training and 1,000 distinct IDs for testing, with no identity overlap. For the 3000VnPerson-Search and 3000Vn-V2E datasets, we divide the 3,000 person IDs into 2,000 for training and 1,000 for testing, ensuring exclusive person IDs in each subset. CUHK-PEDES-VN follows the same split as the original English dataset. This split strategy ensures that the model is always evaluated on unseen identities, conforming to the standard TBPS protocol.

The 3000Vn-V2E dataset consists of English translations of Vietnamese descriptions from the 3000VnPersonSearch dataset [1]. The translations were automatically generated using the Google Translate API and no changes were made to the image description pairings.

The original 3000VnPersonSearch dataset includes 3,000 person identities, 6,302 person images, and 12,602 Vietnamese textual descriptions. Each image is described by two independent annotators to ensure linguistic diversity, with a focus on visual appearance. Each description contains one or more sentences. The average description length is 30.02 tokens, with the longest containing 95 tokens and the shortest 7 tokens. The dataset contains 1,827 unique Vietnamese tokens with a total of 378,274 occurrences.

The person images were collected from surveillance footage in crowded public areas, primarily during the daytime under normal weather and lighting conditions. Human bounding boxes were annotated using a combination of manual tools (e.g., LabelImg) and automated object detection via YOLOv8.

4.1.2 Configuration parameters

Frameworks and environment: All experiments were implemented using Python 3.8, PyTorch v1.13.1, and the HuggingFace Transformers library v4.28.1. The CLIP model was based on OpenAI's official repository (commit `cc5d2e0`), and the PhoBERT model was obtained from the HuggingFace model hub (`vinai/phobert-base`).

[1] Both the 3000VnPersonSearch and 3000Vn-V2E datasets are released for non-commercial academic research purposes only and are available upon request.

Fine-tuning hyperparameters: We fine-tune the model for 20 epochs with a learning rate of 5×10^{-6}. The learning rate is unitless and is applied per optimization step. The batch size is set to 32 image-text pairs and 512 tokens for textual input. We use the Adam optimizer with a weight decay of 0.01. A dropout rate of 0.1 is applied in both the vision and text transformer backbones to mitigate overfitting. For image preprocessing, we apply standard normalization and random horizontal flipping. Text descriptions are tokenized and lowercased using the respective tokenizer of each pre-trained language model.

Vision transformer parameters: We use the ViT-B/32 architecture [21] as the image encoder, consisting of 12 transformer encoder layers, 12 attention heads and an embedding dimension of 768. The input images are resized to 224×224, divided into 32×32 patches, and passed through a linear projection followed by positional encoding and layer normalization.

Text transformer parameters: For English descriptions, we use token embedding, positional embedding and a transformer with 12 residual attention layers, each with 8 attention heads, and embedding dimension of 512. For Vietnamese descriptions, we use **PhoBERT-base** [23], which shares the same architectural configuration.

Loss weighting parameter: In the joint loss formulation (Equation 11), we empirically evaluated the weighting parameter $\alpha \in \{0.2, 0.5, 0.8\}$ to balance global and local alignment. The best trade-off was observed at $\alpha = 0.5$, which we adopt in all final experiments.

Hardware setup: All experiments were conducted on a machine equipped with a single NVIDIA Corporation Device 2504 GPU and 48GB of RAM. The average training time per experimental scenario (20 epochs) was approximately 5 hours. This setup was sufficient for training both global-only and joint global-local models, though we acknowledge limitations in scaling to larger architectures (e.g. ViT-L or ViT-H), as discussed in Section 5.1 and Section 6.

4.2 Evaluation metric

In this work, the top k ranking metric is used for experimental evaluations of text-based person search. The proposed system produces a list of person images that were found for each query sentence, sorted by confidence score. If the corresponding person's image appears in the top k images,

a relevant person retrieval has been performed. The Top-1 (Rank-1), Top-5 (Rank-5), and Top-10 (Rank-10) accuracies are reported in our experiments.

4.3 Result

The experimental results are presented in multiple scenarios to demonstrate the challenges of text-based person search (TBPS) in different languages, particularly in low-resource settings like Vietnamese. In addition, the results also highlight the performance differences between intra-dataset and cross-dataset evaluations.

4.3.1 Evaluation on English language

To evaluate our proposed method **GLAlign** and compare it with existing state-of-the-art (SOTA) methods for English TBPS, we used the CUHK-PEDES dataset and the 3000Vn-V2E dataset. Four experimental scenarios are implemented. The first scenario (**SE1**) uses 11k IDs from CUHK-PEDES for fine-tuning and 1k IDs for testing. This is a standard setting commonly used in prior works. The second scenario (**SE2**) involves the 3000Vn-V2E dataset, with 2k IDs for training and 1k IDs for testing. The third scenario (**SE3**) evaluates cross-dataset performance: the model is trained on 11k IDs from CUHK-PEDES and tested on 1k IDs from 3000Vn-V2E. The fourth scenario (**SE4**) enriches the training data by combining 11k IDs from CUHK-PEDES with 2k IDs from 3000Vn-V2E, while testing remains on the remaining 1k IDs from 3000Vn-V2E.

Table 4 presents the results for the first scenario (**SE1**). Our method achieves a Rank-1 accuracy of 80.75%, outperforming recent transformer-based methods such as IRRA [10] (73.38%), PLOT [11](75.28%) and RaSa [12] (76.51%). Compared to earlier CNN- or RNN-based methods (e.g., GNA-RNN [8], Dual-path CNN [15], ViTAA [4], SCRF [9]), the improvement ranges from approximately 25% to over 60% in Rank-1 accuracy.

Table 5 reports the results for the second scenario (**SE2**). Our model, fine-tuned on 2k IDs from 3000Vn-V2E, achieves Rank-1, Rank-5, and Rank-10 accuracies of 67.50%, 92.96% and 96.82%, respectively. This performance significantly exceeds M-IRRA [24] (Rank-1 = 37.07%). However, performance is slightly lower than that of the CUHK-PEDES-trained model (Table 4), due to the smaller training dataset.

Table 6 illustrates the results of the third scenario (**SE3** for cross-dataset). Our method achieves an accuracy of 65.40% Rank-1, surpassing ViTAA (25.31%) and IRRA (51.35%). However, it underperforms compared to the intra-dataset scenarios, which yielded 80.75% and 67.50% in the first and second settings, respectively.

As shown in Table 7, the fourth scenario (**SE4**) benefits from the enrichment of training data, improving performance to 70.86% Rank-1 accuracy, compared to 65.40% in the third scenario. The additional training data boosts the model's generalization across domains.

4.3.2 Evaluation on Vietnamese language

We evaluated our method on Vietnamese TBPS using three scenarios in the CUHK-PEDES-VN and 3000VnPerson-Search datasets. In the first scenario (**SV1**), we fine-tuned 11k IDs and tested 1k IDs of CUHK-PEDES-VN. The second scenario (**SV2**) trains on CUHK-PEDES-VN and tests on 100 manually annotated samples from 3000VnPerson-Search. The third scenario (**SV3**) scales this evaluation to the full 3k IDs in the 3000VnPerson-Search.

Table 8 shows the results of the first experimental scenario (**SV1**). Our proposed method outperforms the methods in [14] and [13]. The Rank-1 accuracy of our method is 77.72% compared to 52.40% in [14] and 33.07% in [13]. Compared with the results on the same scenario but for the English language (Table 4), the precision for Vietnamese descriptions is about 3% lower than the accuracy of English descriptions in all ranks. This is due to the quality of the translation from English to Vietnamese using Google API for descriptive sentences. The semantics of the description may not be perfectly transferred from English to Vietnamese. This presents a linguistic challenge when used for TBPS. In order to effectively image search for a particular description language, the model should be executed with data from that language itself. Using descriptive data from other languages through text translation helps reduce the cost of data building, but can reduce the efficiency of TBPS.

To statistically assess whether the performance difference between the English and Vietnamese descriptions is significant, we conducted McNemar's test [25] on 1,000 paired queries using the prediction results of CUHK-PEDES (English) and CUHK-PEDES-VN (translated Vietnamese). McNemar's test is a non-parametric statistical test used to evaluate whether there is a significant difference between the proportions of two related binary outcomes. It is particularly suitable for paired nominal data, such as comparing whether a retrieval result for the same query is correct under two different systems or conditions.

The test yielded a p value less than 0.01, indicating that the difference in Rank-1 retrieval accuracy (80.75% vs. 77.72%) is statistically significant. This means that the likelihood of observing such a performance gap under the null hypothesis of no difference is less than 1%, supporting the conclusion that automatic translation introduces meaningful semantic noise affecting the accuracy of the retrieval.

Table 9 presents the experimental results with the second scenario of Vietnamese text-based person search (**SV2**): fine-tuning the model on 11k IDs of the CUHK-PEDES-VN dataset and testing on 100 IDs of the 3000VnPerson-Search dataset. This experimental scenario is closer to the real problem of text-based person search, as the model is trained on a dataset and tested on a new data set with descriptive sentences written by people from another culture. In fact, descriptions for the same person image when done by different people will not be exactly the same, depending on each descriptor's perspective, writing style, and culture.

Table 4: The comparative results of our method (**GLAlign**) with other SOTA methods on the first experimental scenario-**SE1**: fine tuning on 11k IDs and testing on 1k IDs of CUHK-PEDES dataset

Method	Backbone		Rank-1	Rank-5	Rank-10
	Image	Text			
GNA-RNN [8]	VGG16	LSTM	19.71%	43.20%	55.24%
Dual-path CNN [15]	ResNet50	ResNet50	39.06%	61.67%	70.69%
Method in [14]	ResNet50	ResNet50	41.48%	62.89%	72.59%
ViTAA [4]	ResNet50	Bi-LSTM	55.97%	75.84%	83.52%
SCRF [9]	ResNet50	BERT	64.04%	82.99%	88.81%
IRRA [10]	CLIP-ViT-B/16	CLIP-Xformer	73.38%	89.93%	93.71%
PLOT [11]	CLIP-ViT-B/16	CLIP-Xformer	75.28%	90.42%	94.12%
RaSa [12]	ALBEF	ALBEF	76.51%	90.29%	94.25%
GLAlign (Ours)	ViT-B/32	GPT-2	**80.75%**	**97.85%**	**99.38%**

Table 5: The comparative results of our method (**GLAlign**) with other methods on the second experimental scenario-**SE2**: fine tuning on 2k IDs and testing on 1k IDs of 3000Vn-V2E dataset

Method	Backbone		Rank-1	Rank-5	Rank-10
	Image	Text			
GNA-RNN [8]	VGG16	LSTM	26.85%	52.47%	66.40%
ViTAA [4]	ResNet50	Bi-LSTM	27.08%	51.38%	63.00%
M-IRRA [24]	CLIP-ViT-B/16	Multilingual CLIP	37.07%	66.23%	76.51%
GLAlign (Ours)	ViT-B/32	GPT-2	**67.50%**	**92.96%**	**96.82%**

Table 6: The comparative results of our method (**GLAlign**) with other methods on the third experimental scenario-**SE3**: fine tuning on 11k IDs of CUHK-PEDES dataset and testing on 1k IDs of 3000Vn-V2E dataset

Method	Backbone		Rank-1	Rank-5	Rank-10
	Image	Text			
ViTAA [4]	ResNet50	Bi-LSTM	25.31%	66.20%	59.72%
IRRA [10]	CLIP-ViT-B/16	CLIP-Xformer	51.35%	78.03%	85.78%
GLAlign (Ours)	ViT-B/32	GPT-2	**65.40%**	**94.80%**	**98.40%**

Table 7: The comparative results of our method (**GLAlign**) with other methods on the fourth experimental scenario-**SE4**: fine tuning on 11k IDs of CUHK-PEDES dataset plus 2k IDs of 3000Vn-V2E dataset, and testing on the remaining 1k IDs of 3000Vn-V2E dataset

Method	Backbone		Rank-1	Rank-5	Rank-10
	Image	Text			
ViTAA [4]	ResNet50	Bi-LSTM	31.17%	58.12%	70.35%
IRRA [10]	CLIP-ViT-B/16	CLIP-Xformer	59.57%	82.99%	89.71%
GLAlign (Ours)	ViT-B/32	GPT-2	**70.86%**	**95.10%**	**98.70%**

Table 8: The comparative results of our method (**GLAlign**) with other methods on Vietnamese TBPS, considering the first experimental scenario-**SV1**: training on 11k IDs and testing on 1k IDs of CUHK-PEDES-VN dataset

Method	Backbone		Rank-1	Rank-5	Rank-10
	Image	Text			
Method in [14]	ResNet50	ResNet50	33.07%	53.39%	62.97%
Method in [13]	ResNet50	Bi-LSTM	52.40%	72.28%	80.78%
GLAlign (Ours)	ViT-B/32	GPT-2	**77.72%**	**94.31%**	**97.02%**

Table 9: The comparative results of our method (**GLAlign**) with other methods on Vietnamese TBPS, considering the second experimental scenario-**SV2**: training on 11k IDs of CUHK-PEDES-VN dataset and testing on 100 IDs of 3000VnPerson-Search dataset

Method	Backbone		Rank-1	Rank-5	Rank-10
	Image	Text			
Method in [14]	ResNet50	ResNet50	42.32%	70.04 %	81.52 %
Method in [13]	ResNet50	Bi-LSTM	60.02 %	81.23 %	89.57 %
GLAlign (Ours)	ViT-B/32	GPT-2	**85.72%**	**97.31%**	**99.70%**

Table 10: The comparative results of our method with other methods on Vietnamese TBPS, considering the third experimental scenario-**SV3**: training on 11k IDs of CUHK-PEDES-VN dataset and testing on 3k IDs of 3000VnPerson-Search dataset

Method	Backbone		Rank-1	Rank-5	Rank-10
	Image	Text			
Method in [14]	ResNet50	ResNet50	8.27%	20.92%	29.11%
Method in [13]	ResNet50	Bi-LSTM	28.49	37.26 %	48.94 %
GLAlign (Ours)	ViT-B/32	GPT-2	**50.72%**	**67.31%**	**79.70%**

This shows the challenge of natural language processing in the context of a TBPS problem. The experimental results obtained by our method significantly outperform the SOTA methods in [14] and [13]. The Rank-1, Rank-5, and Rank-10 of our method are 85.72%, 97.31%, and 99.70%, respectively. These results in [14] and [13] are only 42.32% and 60.02% for Rank-1; 70.04% and 81.23% for Rank-5; 81.52% and 89.57% for Rank-10, respectively.

The comparative results of our method and other methods for the third experimental scenario (**SV3**) are shown in Table 10. In this scenario, the model is also trained on 11k IDs of CUHK-PEDES-VN as in the second scenario, but the testing is implemented on all samples (3k IDs) of 3000VnPerson-Search. This is much more challenging than the second test scenario. The experimental results in the third scenario are much lower than the results in the second scenario. The Rank-1 accuracy of our method decreases from 85.72% in the second scenario to 50.72% in the third scenario. This decline also occurred in Rank-5 and Rank-10. Compared to other methods in [14] and [13], the experimental results of our method are still much higher at all Rank values. These experimental results indicate the challenge of TBPS with cross-dataset evaluation and searching on a massive image database.

4.3.3 Impact of global vs. local feature alignment

To quantify the contribution of each component in our proposed framework, we conduct ablation studies on two representative datasets: CUHK-PEDES (English) and 3000VnPerson-Search (Vietnamese). We evaluated three variants of the model:

- **Global-only model:** Uses ViT-B/32 as the vision backbone and GPT-2 or PhoBERT as the text encoder. Only the global [CLS] representation is used for matching; local features and part-level alignment are disabled.
- **Local-only model:** Human body parts and noun phrases are extracted and aligned using the same transformer encoders. The global [CLS] embedding is excluded; only the part-to-part alignment is used.
- **Full model (**GLAlign**):** Combines global and local representations. The final loss is the weighted sum of global and local symmetric cross-entropy losses as defined in Equation (11).

Table 11 presents the ablation results. From the results, we observe that the global-only model performs reasonably well due to the strong representation capacity of pretrained transformers. However, the incorporation of local alignment brings significant additional gains: +6.54% in CUHK-PEDES and +7.84% in 3000VnPerson-Search. This highlights the importance of fine-grained part-level matching in TBPS tasks, especially in linguistically complex or multilingual environments. We also note that the local-only model, while weaker than the global-only model, still retains competitive performance. This suggests that part-to-part alignment alone carries valuable semantic information but works best when complemented with global contextual embeddings.

5 Discussion

In this section, we provide a deeper analysis of the experimental results and explain the observed performance differences between datasets and language settings compared to existing state-of-the-art (SOTA) methods summarized in Table 1. The discussion focuses on three main aspects: model architecture, cross-lingual performance, and generalization across datasets.

5.1 Effectiveness of joint global-local alignment

Our proposed method **GLAlign** outperforms all baselines in English and Vietnamese datasets, achieving a Rank-1 score of 80.75% in CUHK-PEDES and 85.72% in 3000VnPerson-Search. These gains can be primarily attributed to the integration of global and local feature alignment in a transformer-based framework.

Unlike previous transformer-based methods such as CLIP [6] or RaSa [12], which mainly focus on global visual-textual alignment, our framework incorporates local-level matching through noun phrase (NP) to body part mapping. By leveraging human parsing and NP extraction, we create fine-grained semantic correspondences between specific textual attributes (e.g., ”red jacket”, ”black shoes”) and corresponding visual regions (e.g., upper body, shoes). This granularity enhances discriminative power during retrieval, especially when the global context is ambiguous or insufficient.

Table 11: Ablation study of Rank-1 accuracy on CUHK-PEDES (English) and 3000VnPerson-Search (Vietnamese) under experimental scenarios **SE1** and **SV2**, respectively. We compare global-only, local-only, and full (joint global-local) model variants.

Model Variant	CUHK-PEDES (English)	3000VnPerson-Search (Vietnamese)
Global-only model	74.21%	77.88%
Local-only model	68.10%	72.35%
Full model (GLAlign)	**80.75%**	**85.72%**

5.2 Cross-lingual performance and linguistic variance

Our framework also shows strong performance across languages. In particular, the Rank-1 accuracy on the Vietnamese-native 3000VnPerson-Search dataset (85.72%) is even higher than on CUHK-PEDES (80.75%). This improvement stems from the use of PhoBERT, a pre-trained Vietnamese language model that captures native linguistic features more accurately than translated sentences.

In contrast, when evaluating on CUHK-PEDES-VN (a machine-translated version of the English dataset), we observe a slight performance drop (Rank-1 = 77.72%). This is due to semantic drift and structural inconsistency introduced during translation, which can misalign noun phrases with visual attributes. To illustrate this, we empirically estimated the translation-induced mismatch rate by sampling 200 randomly selected description-image pairs from the translated dataset and manually checking whether the extracted Vietnamese noun phrases semantically match the associated body parts. We observed an approximate error rate of 18.50%, mainly due to lexical ambiguity or loss of specificity.

Despite overall strong performance, we also analyzed failure cases to better understand the model's limitations in cross-lingual settings. These errors often fall into four categories:

- **Ambiguous or generic descriptions:** Queries such as "a man in dark clothes" or "a person walking" lack distinctive visual details, resulting in a high search-and-retrieval confusion.
- **Low-quality or poorly illuminated images:** In the 3000VnPerson-Search dataset, images captured under poor lighting or motion blur reduce visual discriminability, especially in local regions.
- **Partial occlusion:** When key attributes such as "yellow handbag" or "white shoes" are occluded or not visible due to cropping, part-based matching fails to align these cues correctly.
- **Translation-induced mismatch:** Subtle errors in automatic translation (as discussed above) can cause mismatches in noun phrases or misinterpretation of clothing and objects.

We illustrate representative cases in Figure 3, where the global-only model often fails due to these issues, while local alignment partially mitigates some of them.

5.3 Cross-dataset generalization

Although intra-dataset training and testing yield the best results, we also evaluate the model's cross-dataset generalization. When fine-tuned on CUHK-PEDES and tested on 3000Vn-V2E (cross dataset), Rank-1 drops to 65.40%. This performance reduction is expected due to domain change (e.g., surveillance image conditions, clothing diversity, and cultural description styles).

However, compared to other methods under the same conditions (e.g., IRRA at 51.35%), our framework still achieves superior generalization. This is likely because the shared embedding space learned from both global and local features is more transferable and can better accommodate unseen data distributions.

5.4 Summary of observations

The findings discussed in the previous subsections highlight several key takeaways about the effectiveness, robustness, and generalization capacity of our proposed framework. We summarize the most notable insights as follows.

- Performance improvement is mainly driven by the combination of local-level NP-to-body-part mapping and global transformer-based embeddings.
- PhoBERT-based textual encoding significantly boosts results in native Vietnamese descriptions compared to translated datasets.
- The model demonstrates strong cross-dataset generalization, with lower sensitivity to domain and language changes than prior methods.

Overall, the experimental and comparative analysis confirms that joint global-local alignment, when integrated into a fine-tuned transformer framework, leads to superior performance in both monolingual and multilingual TBPS tasks.

5.5 Limitations and future work

Although our method achieves state-of-the-art performance in multiple TBPS scenarios, there are several limitations that suggest promising directions for future work.

Model size and computational cost. In this study, we adopted ViT-B/32 as the visual encoder due to its balance between performance and computational feasibility on our hardware setup (a single NVIDIA GTX 1650 GPU with

(a) Ambiguous description: A person wearing dark clothes (người đàn ông mặc quần áo tối màu)



The global-only model fails due to ambiguous description; full model also struggles under lack of visual cues.

(b) Poor lighting: A man in a red striped sweatshirt, light colored pants, yellow backpack, red shoes (người đàn ông mặc áo nỉ kẻ màu đỏ, quần sáng màu, đeo bao lô vàng, đi giày màu đỏ.)



In low-light conditions, visual cues such as color and texture become less distinguishable, leading to incorrect retrieval by the global-only model. The full model still succeeds by focusing on localized part-level features that remain discriminative despite reduced image quality.

(c) Partial occlusion: A woman carrying a white handbag (Một người phụ nữ mang một chiếc túi xách màu trắng)



When the target object (e.g., a white handbag) is partially occluded, the global-only model fails to identify the correct match. The full model correctly retrieves the target by leveraging localized alignment, which focuses on visible cues such as clothing and posture.

(d) Translation mismatch: A man wears a gray T-shirt with white logo (Áo phông xám có logo màu trắng)



Translation artifacts introduce semantic drift that affects both global and local matching. In this case, neither model retrieves the correct result due to a mismatch between the translated description and the actual visual attributes.

Figure 3: Qualitative examples illustrating four representative failure types in text-based person search: (a) ambiguous descriptions, (b) poor image quality, (c) partial occlusion, and (d) translation-induced semantic drift. Each row shows (from left to right): the input query image, the top-1 retrieved result from the global-only model, and the top-1 result from our full model (global + local features). Red bounding boxes indicate incorrect retrievals, while green bounding boxes denote correct matches. While the full model improves performance in many cases, it still struggles under extreme ambiguity or visual noise.

8GB RAM). However, recent work shows that larger transformer models such as ViT-L/16 and ViT-H/14 [21] achieve better performance in vision tasks that require high granularity. These deeper models have more parameters (ViT-

B: 86M, ViT-L: 307M, ViT-H: 632M), but they also incur higher memory demands and training time. Additionally, training larger models on limited datasets such as 3000VnPerson-Search may cause overfitting. In future work, we plan to investigate the use of deeper backbones with distillation or low-rank adaptation strategies to balance efficiency and accuracy.

Rule-based NP-to-body-part mapping. Our current local alignment is based on matching the rules of noun phrases with body parts using predefined keyword lists. This approach is language-dependent and can be brittle when handling paraphrases, synonyms, or figurative language. Errors are more likely in low-resource languages, such as Vietnamese, where linguistic diversity is greater. Future research may explore neural phrase grounding or attention-guided part discovery methods to enhance the flexibility and robustness of local alignment.

Dataset scale and generalization. The performance of our model varies between datasets, especially in cross-dataset or cross-lingual settings. Although our method generalizes well compared to previous work, further gains could be achieved by training on larger multilingual datasets, including real-world surveillance footage and human-written multilingual descriptions. This would also mitigate semantic drift caused by automatic translation, as discussed in Section 5.3.

Real-world deployment challenges. We observe several failure modes under real-world conditions, including poor lighting, partial occlusion, and vague descriptions. These cases highlight the need to incorporate uncertainty modeling, visual quality enhancement, and query disambiguation techniques into future versions of the TBPS system.

In general, while our framework establishes a strong foundation, we believe that improving scalability, multilingual adaptability, and visual grounding mechanisms will be critical to advancing TBPS systems in practical deployments.

6 Conclusion

In this work, an efficient framework for TBPS called GLAlign is proposed. In this framework, both global and local features of image and text pairs are considered when fine-tuning the large-scale and pretrained transformer models on standard datasets. Several experimental scenarios are implemented to show the outperforming results of our proposed solution compared to other SOTA methods in both descriptive languages, English and Vietnamese. Fine-tuning the large-scale and pretrained transformers on a large enough dataset as CUHK-PEDES brings higher searching performance than a smaller dataset of 3000Vn-V2E dataset. The TBPS performance of the cross-dataset evaluation is decreased compared to the intra-dataset evaluation. Considering the experimental results with English descriptions in the dataset, the results with Vietnamese descriptions are lower. These experimental observations show the challenges of the TBPS problem. Although our proposed method has improved results compared to other SOTA methods, further improvements in feature learning and matching, as well as the enhanced TBPS database, should be implemented in the future to improve the efficiency of the TBPS.

References

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[2] C. Gao, G. Cai, X. Jiang, *et al.*, "Contextual non-local alignment over full-scale representation for text-based person search", Jan. 2021. DOI: 10.48550/arXiv.2101.03036.

[3] X. Han, S. He, L. Zhang, and T. Xiang, "Text-based person search with limited data", *arXiv preprint arXiv:2110.10807*, p. 13, 2021. DOI: 10.48550/arXiv.2404.18106.

[4] Z. Wang, Z. Fang, J. Wang, and Y. Yang, "Vitaa: Visual-textual attributes alignment in person search by natural language", in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*, Springer, 2020, pp. 402–420. DOI: 10.1007/978-3-030-58610-2_24.

[5] *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, DOI: 10.1109/TNNLS.2023.3310118.

[6] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision", in *International conference on machine learning*, PMLR, 2021, pp. 8748–8763. DOI: 10.48550/arXiv.2103.00020.

[7] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation", in *International conference on machine learning*, PMLR, 2022, pp. 12 888–12 900. DOI: 10.48550/arXiv.2201.12086.

[8] S. Li, T. Xiao, H. Li, B. Zhou, D. Yue, and X. Wang, "Person search with natural language description", in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5187–5196. DOI: 10.1109/CVPR.2017.551.

[9] W. Suo, M. Sun, K. Niu, *et al.*, "A simple and robust correlation filtering method for text-based person search", in *European conference on computer vision*, Springer, 2022, pp. 726–742. DOI: 10.1007/978-3-031-19833-5_42.

[10] M. Cao, Y. Bai, Z. Zeng, M. Ye, and M. Zhang, "An empirical study of clip for text-based person search", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, 2024, pp. 465–473. DOI: 10.48550/arXiv.2308.10045.

[11] J. Park, D. Kim, B. Jeong, and S. Kwak, "Plot: Text-based person search with part slot attention for corresponding part discovery", in *European Conference on Computer Vision*, Springer, 2025, pp. 474–490. DOI: 10.1007/978-3-031-72664-4_27.

[12] Y. Bai, M. Cao, D. Gao, *et al.*, "Rasa: Relation and sensitivity aware representation learning for text-based person search", 2023. DOI: 10.24963/ijcai.2023/62.

[13] T. T. T. Pham, V.-T. Nguyen, H.-Q. Nguyen, *et al.*, "Person search by natural language description in vietnamese using pre-trained visual-textual attributes alignment model", in *2021 13th International Conference on Knowledge and Systems Engineering (KSE)*, IEEE, 2021, pp. 1–6.

[14] T. T. T. Pham, H.-Q. Nguyen, H. Phan, *et al.*, "Towards a large-scale person search by vietnamese natural language: Dataset and methods", *Multimedia Tools and Applications*, vol. 81, no. 19, pp. 27 569–27 600, 2022. DOI: 10.1007/s11042-022-12138-1.

[15] Z. Zheng, L. Zheng, M. Garrett, Y. Yang, M. Xu, and Y.-D. Shen, "Dual-path convolutional image-text embeddings with instance loss", *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 16, no. 2, pp. 1–23, 2020. DOI: 10.1145/3383184.

[16] D. Mladenić and M. Grobelnik, "Automatic text analysis by artificial intelligence", *Informatica*, vol. 37, no. 1, pp. 27–33, 2013.

[17] B. Xiao, H. Wu, W. Xu, *et al.*, "Florence-2: Advancing a unified representation for a variety of vision tasks", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4818–4829. DOI: 10.1109/CVPR52733.2024.00461.

[18] J. Li, R. R. Selvaraju, A. Gotmare, S. Joty, C. Xiong, and S. C. H. Hoi, "Align before fuse: Vision and language representation learning with momentum distillation", in *Advances in Neural Information Processing Systems*, 2021.

[19] D. Mladenić and B. Fortuna, "Multilingual document classification and sentiment analysis", *Informatica*, vol. 37, no. 3, pp. 241–252, 2013.

[20] A. Chauhan, A. Sharma, and R. Mohana, "An enhanced aspect-based sentiment analysis model based on roberta for text sentiment analysis", *Informatica*, vol. 49, no. 14, pp. 193–202, 2025. DOI: 10.31449/inf.v49i14.5423.

[21] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale", in *International Conference on Learning Representations ICLR (2021)*, 2021.

[22] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, *Language models are unsupervised multitask learners*, Technical report, OpenAI, 2019.

[23] D. Q. Nguyen and A. T. Nguyen, "Phobert: Pre-trained language models for vietnamese", pp. 1037–1042, 2020. DOI: 10.18653/v1/2020.findings-emnlp.92.

[24] H. P. T. Tran, T. H. P. Phan, T. B. N. Nguyen, *et al.*, "M-irra: A multilingual model for text-based person search", in *The 3rd APSIPA Workshop on Signal and Information Processing (SIP) in Vietnam*, 2024, pp. 1–6.

[25] Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages", *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947.

A Hybrid Sentiment Classification Model for Course Comments Based on 'Unstable Interval' Correction - An Applied Study Integrating SVM and BiLSTM

Fan Yang[1], Qin Zhang[2], Jie Liu[3*]
[1]Hubei Provincial Collaborative Innovation Center for Basic Education Information Technology Services, Hubei University of Education, Wuhan 430205, China
[2]School of Information Engineering, Yangzhou University, Yangzhou 225002, China
[3]College of Information, Wuhan Vocational College of Software and Engineering (Wuhan Open University), Wuhan 430205, China
E-mail of corresponding author: liujie993@126.com
*Corresponding author

Keywords: sentiment dictionary, sentiment polarity classification, short text sentiment clustering, machine learning

Received: August 7, 2025

Aiming at the limitations of mainstream text sentiment analysis methods (poor generalization of dictionary-based methods, machine learning relying on labeled data and being prone to overfitting), this paper proposes a hybrid model combining sentiment dictionaries and machine learning. The 'unstable interval' refers to the interval where the sentiment dictionary score is close to 0 (the boundary between positive and negative emotions) with low classification accuracy, and its calculation and derivation are based on the test set score distribution, and the interval range is determined by traversal. The model fusion method is: the sentiment dictionary and SVM/BiLSTM output polarities respectively, the consistent part is retained, and the inconsistent part in the 'unstable interval' is based on the SVM/BiLSTM result. Taking 3119 course comments from the NetEase Cloud Classroom platform as the experimental data source, the experimental results show that the accuracy of the hybrid model is increased to 88.9% (SVM sub-model) compared with the single SVM model (82.3%), and to 91.2% (BiLSTM sub-model) compared with the single BiLSTM model (85.6%); at the same time, it is clarified that positive emotions mainly come from 'practical course content' and 'clear explanation by lecturers', while negative emotions focus on causes such as 'delayed course updates' and 'untimely customer service feedback', and the key factors in each cluster highlight the course's strengths and weaknesses. Researchers can utilize these typical emotional factors as evidence for dynamically adjusting course content.

Povzetek: Hibridni model, ki združuje čustvene slovarje in SVM/BiLSTM, občutno izboljša natančnost analize sentimenta v mejnih primerih. Rezultati pokažejo višjo uspešnost od posameznih modelov ter jasno opredelijo glavne pozitivne in negativne dejavnike v ocenah tečajev.

1 Introduction

In recent years, with the continuous development of information technology, many online course platforms have emerged in China, such as MOOC, NetEase Cloud Course, and Micro Classroom, providing users with a large number of high-quality courses. During the epidemic, online education platforms played an important role, and the Ministry of Education organized 22 online course platforms to open over 24000 online courses for free [1-2]. The increased opportunities for communication between teachers and students provided by online course platforms are further expanding the learning pathways for users. This has also drawn more attention to online courses, accelerating the development and construction of high-quality online course platforms, gradually becoming one of the important development trends in future education.

Therefore, how to evaluate the quality of an online course and how to visually display the advantages and disadvantages of the course are difficult issues in online course management [3-4]. How to perceive user emotions and emotional causes from a large number of comments, in order to provide effective and intuitive information for users, educators, and platforms, in order to evaluate the quality of online courses, the advantages and disadvantages of courses, and improve the overall quality of courses has become a hot research issue [5-6]. There is a lot of research on sentiment classification in short texts [7-10]. Sangeetha and Kumaran [11] proposed a Harris Hawks optimized recurrent neural network long and short memory algorithm based on Pearson correlation coefficient, which is used to select features from user comments and classify their emotions based on their appropriate polarity. Misuraca et al. [12] proposed Using Opinion Mining as an educational analytical: An integrated strategy for the analysis of students' feedback. The comparison is shown in Table 1 below. Aka-Uymaz and Kumova [13] reviewed vector-based sentiment analysis

methods and pointed out that single models have the problem of weak generalization ability, providing theoretical reference for the construction of the hybrid model in this study. Trillo et al [14] proposed a new large-scale group decision-making method, which uses Natural language processing methods, especially emotional analysis, to manage the information generated by a large number of experts.

Table 1: A quantitative comparison table of related works

Research Literature	DataSet	The model used	F1 value	Boundedness
Sangeetha and Kumaran [11]	Amazon user reviews (10000 articles)	Harris Eagle Optimization LSTM	89.2 %	It relies on a large amount of labeled data and has poor generalization with small sample.
Misuraca et al. [12]	Student course feedback (5000 articles)	Opinion Mining	85.6 %	It is only applicable to specific educational scenarios and has poor cross-disciplinary performance.
This article	Comments on NetEase Cloud Classroom (3119 articles)	Sentiment Word dictionary - BiLSTM	96.54 %	Cross-disciplinary exploration of the "unstable interval" is necessary.

In current research on text sentiment analysis, many studies tend to focus on exploring single methodologies, whether it' s analysis based on sentiment dictionary or relying on machine learning or deep learning technologies. This approach of studying a single method independently has led to several issues, such as low accuracy in sentiment recognition, limited generalization capabilities, and a lack of ability to recognize complex text sentiments accurately. And existing hybrid models do not solve the problem of 'low accuracy of sentiment dictionaries in boundary intervals'. Through research on comments from online courses, this paper designs a hybrid model that combines sentiment dictionary with machine learning to enhance the accuracy and reliability of sentiment polarity classification. Experimental results indicate that the emotion polarity algorithm based on "Sentiment Dictionary -SVM" improved the F1 score by 9.51% compared to the emotion polarity algorithm based solely on sentiment dictionary, and by 3.17% compared to the Support Vector Machine emotion polarity algorithm. The emotion polarity algorithm based on "Sentiment Dictionary -BiLSTM" increased the F1 score by 14.86% compared to the sentiment dictionary -based emotion polarity algorithm, and by 1.03% compared to the BiLSTM emotion polarity algorithm. The emergence of this hybrid approach offers new strategies and possibilities for addressing the challenges present in current text sentiment analysis, and is expected to bring more comprehensive and superior solutions to the field of sentiment recognition. In the analysis of the causes behind sentiments in course reviews, this paper utilizes the DBSCAN algorithm for text clustering. The algorithm takes as input the dataset that has undergone sentiment polarity classification through the hybrid model algorithm. The algorithm outputs 29 clusters of "positive" comments and 26 clusters of "negative" comments. The typical factors of different clusters represent to some extent the strengths and weaknesses of the courses. Researchers can use an understanding of these typical sentiment-causing factors as a basis for dynamically adjusting course content.

The research questions in this paper are finally clarified as follows: (1) How to solve the limitations of single methods in sentiment classification of course comments by fusing sentiment dictionaries and machine learning? (2) What are the definition and quantification method of the 'unstable interval', and can it improve the classification accuracy of the boundary interval? (3) Can the emotional cause analysis based on DBSCAN clustering provide an effective basis for course quality optimization?

2 "Unstable interval" and its model improvement

The selection of this hybrid approach is based on the following considerations. The sentiment dictionary is chosen because it does not require labeled data and can quickly obtain prior knowledge; SVM is chosen because of its strong stability in small-sample classification and suitability for processing low-dimensional features; BiLSTM is chosen because it can capture text context dependence and is suitable for processing complex emotional expressions; BERT/RoBERTa is not chosen because the experimental dataset is small (3119 entries), the pre-trained model is prone to overfitting, and the computational cost is high; CNN is not chosen because it is good at local feature extraction and has weaker ability to capture long text context than BiLSTM.

2.1 Basic model

2.1.1 Emotional polarity algorithm based on sentiment dictionary

This algorithm is based on sentiment word scoring for text sentiment polarity judgment. The algorithm constructed in this article includes a sentiment dictionary (BosonNLP sentiment dictionary), a negative word dictionary, and a degree adverb dictionary (derived from the 'Sentiment Analysis Word Set (Beta Version)' released by CNKI (China National Knowledge Infrastructure), specific reference link: https://www.cnki.net/other/sentiment_analysis.html).";
Added a formal citation entry for this degree adverb dictionary in the references section: "[15]. The algorithm sets “Score” as the sentiment polarity score for each comment. When “Score” is greater than 0, the comment is judged as positive, and when “Score” is less than 0, Determine the comment as negative. Each comment may contain multiple emotional words, as well as degree adverbs and negatives that modify the emotional words. Assuming that the score of each emotional word is EWS, the number of emotional words is n, the degree adverbs that modify the emotional words are DA, and the number of negatives is N. The calculation of sentiment polarity scores for comments is described in Equation (1). By traversing all course comments, the corresponding emotional polarity score is calculated for each course comment, and the emotional polarity of the corresponding comment is marked based on the score. The algorithm flowchart is shown in Figure 1.

$$Score=[\sum_{i=1}^{n}(EWS\times DA)]\times -1^{(N)} \quad (1)$$

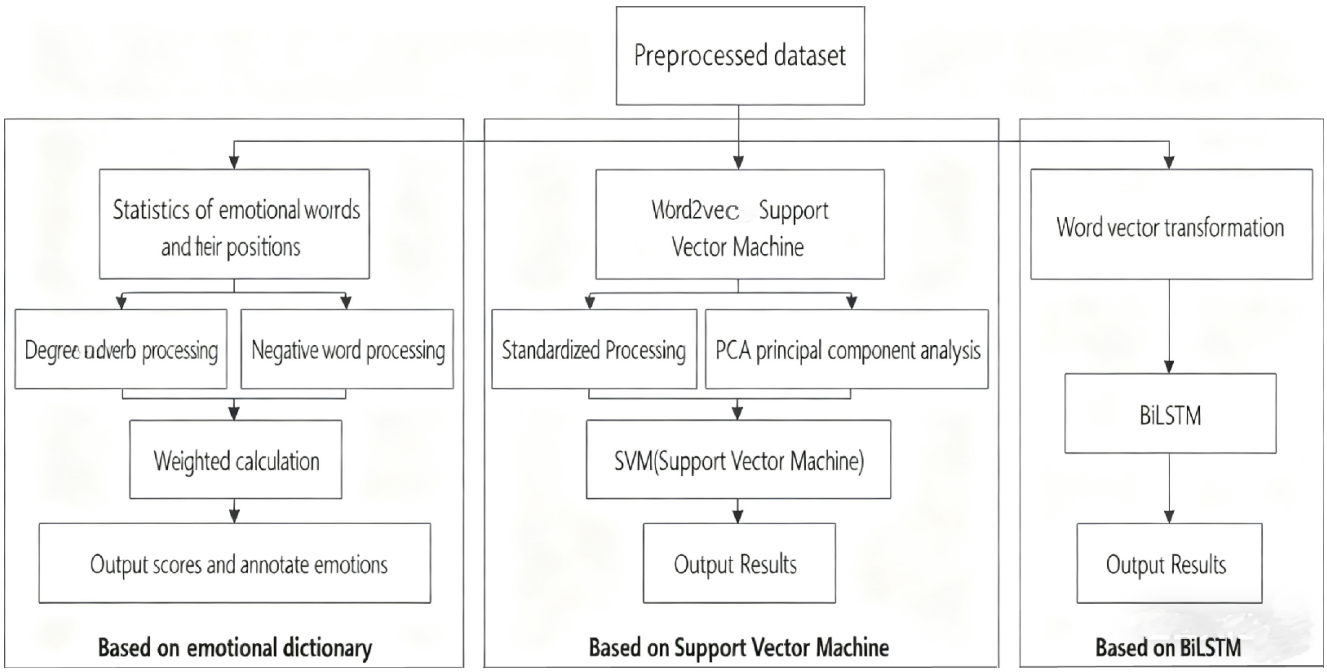


Figure 1: Flowchart of three emotional polarity algorithms

2.1.2 SVM-based sentiment polarity algorithm

This algorithm is first based on the Word2vec model to convert the text in course comments into a dense vector (Bag of words) that can be understood by computers. Due to the influence of the dimensionality of the feature space on the density of the vector, the classification results may deviate. Therefore, before formal training, PCA (Principal Component Analysis) is used to reduce the dimensionality of the data and extract the main feature components of the data, this reduces the amount of computation and storage space. According to the variance proportion of PCA principal component variables, this article selects the first 10 principal component features as the input part of the model, randomly selects 70% of the dataset as the training set, and uses the remaining 30% of the data as the test set and validation set in a 1:1 ratio, to establish a support vector machine (SVM) model for training. In this study, the SVM model uses a linear kernel, the value of the regularization parameter C is set to 1.0, and other important hyperparameters remain at their default settings. The algorithm flowchart is shown in Figure 1.

2.1.3 BiLSTM-based sentiment polarity algorithm.

The algorithm model consists of two main layers. The first layer is the LOOK layer, and the second layer is the BiLSTM (bidirectional LSTM) layer, with the first layer serving as the input layer of the model. Text sentences are first mapped to word vectors, and the dimension of the word vectors is set to 128. These word vectors form sentence vectors used as inputs for each time step of the BiLSTM layer in the second layer. The BiLSTM layer is set to 1 layer, and the size of the hidden states is 64. The hidden state sequences {h1, h2, h3, ..., hn} and {h1’, h2’, h3’, ..., hn’} outputted by the positive and negative bidirectional LSTMs are concatenated based on their positions to obtain the combined hidden state sequence Hi

= [hi; hi'] at each time step. The BiLSTM layer uses tanh as the activation function, the Adam optimizer is used during training, and the learning rate is set to 0.001. The final output of the BiLSTM layer is the sequence {H1, H2, ..., Hn}, which establishes the foundation for training the BiLSTM model. The algorithm flowchart is shown in Figure 2.

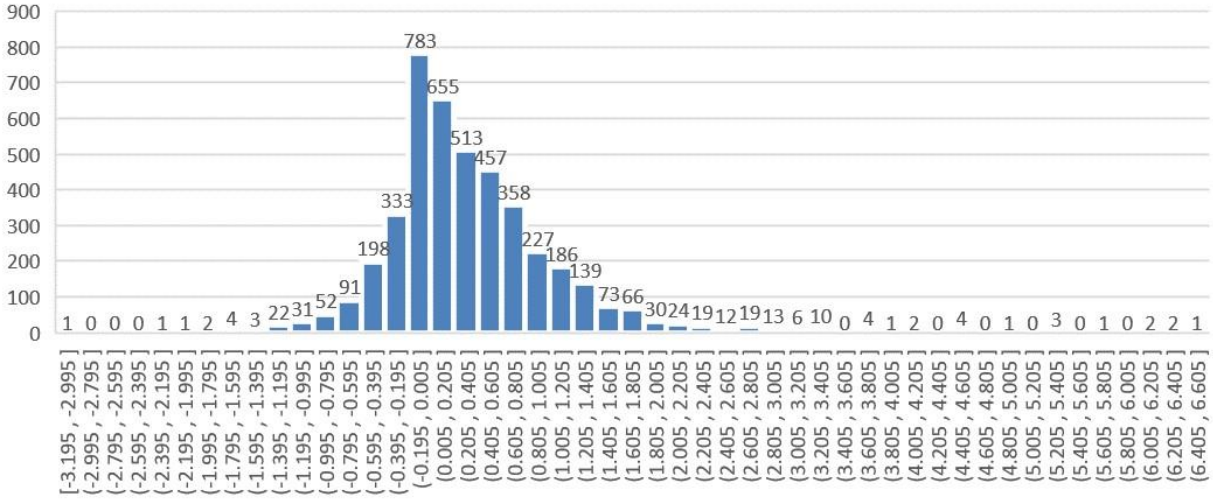


Figure 2: Distribution of score ranges in the test dataset

2.2 The 'unstable interval' of sentiment polarity based on the emotion lexicon sentiment polarity algorithm

When multiple emotion words appear in course reviews (especially when both positive and negative emotion words appear), it poses a significant challenge to the accuracy of the emotion lexicon sentiment polarity algorithm. This article first uses an emotion polarity algorithm based on a sentiment dictionary to obtain the score range of the review test dataset, which is [-3.19489524328,6.48861516533159]. The distribution of the score range of the test dataset is shown in Figure 2. A comparison between the sentiment polarity obtained by an algorithm based on a sentiment dictionary and the sentiment polarity manually labeled indicates that, within a range close to Score = 0, the accuracy of the sentiment polarity obtained by an algorithm based on a sentiment dictionary is relatively low, as shown in Table 2. Within each emotional polarity score range with an interval of 0.2 points, the number of emotional polarities obtained by the algorithm based on the sentiment dictionary is AL(i) (i = 1,2,3, ...n-1, n), and the number of manually labeled emotional polarities that are the same is AS(i) (i = 1,2,3, ... n-1, n). The accuracy is shown in Equation (2). The blank portion of the table indicates that the sentiment polarity algorithm based on a sentiment dictionary does not have data in this range.

Table 2: Accuracy comparison of scores distribution

Score ranges	Accuracy	Score ranges	Accuracy	Score ranges	Accuracy
(-3.195, -2.995]	100%	(0.205, 0.405]	83.43%	(3.605, 3.805]	100%
(-2.995, -2.795]		(0.405, 0.605]	86.65%	(3.805, 4.005]	100%
(-2.795, -2.595]		(0.605, 0.805]	91.34%	(4.005, 4.205]	100%
(-2.595, -2.395]		(0.805, 1.005]	96.48%	(4.205, 4.405]	
(-2.395, -2.195]	100%	(1.005, 1.205]	97.31%	(4.405, 4.605]	100%
(-2.195, -1.995]	100%	(1.205, 1.405]	97.12%	(4.605, 4.805]	
(-1.995, -1.795]	100%	(1.405, 1.605]	97.26%	(4.805, 5.005]	100%
(-1.795, -1.595]	100%	(1.605, 1.805]	96.97%	(5.005, 5.205]	
(-1.595, -1.395]	100%	(1.805, 2.005]	100.00%	(5.205, 5.405]	100%
(-1.395, -1.195]	86.36%	(2.005, 2.205]	95.83%	(5.405, 5.605]	
(-1.195, -0.995]	93.55%	(2.205, 2.405]	94.74%	(5.605, 5.805]	100%

(-0.995, -0.795]	94.23%	(2.405, 2.605]	100.00%	(5.805, 6.005]	
(-0.795, -0.595]	95.60%	(2.605, 2.805]	94.74%	(6.005, 6.205]	100%
(-0.595, -0.395]	95.96%	(2.805, 3.005]	100.00%	(6.205, 6.405]	100%
(-0.395, -0.195]	96.40%	(3.005, 3.205]	100.00%	(6.405, 6.605]	100%
(-0.195, 0.005]	77.01%	(3.205, 3.405]	80%		
(0.005, 0.205]	79.24%	(3.405, 3.605]			

$$\text{Accuracy}=\frac{\text{AS}(i)}{\text{AL}(i)}\times 100\% \quad (2)$$

Based on this, this article speculates that within a certain interval centered around Score = 0, the sentiment polarity algorithm based on the sentiment dictionary is not as accurate as other regions in judging the sentiment polarity of this interval's Score. This result may be due to misjudgments caused by Scores being too close to 0 (the boundary between "negative" and "positive" emotions), as shown in Figure 3. Therefore, we propose use the traversal method to search for this "unstable interval", and combines the sentiment polarity algorithm based on SVM and the sentiment polarity algorithm based on BiLSTM to correct the sentiment polarity of this interval.

Formal definition of the 'unstable interval' is as followed: Let the comment score calculated by the sentiment dictionary be Score, the distribution range of Score in the test set be [Smin, Smax], and the 'unstable interval' be [Sx, Sy], where Sx is the minimum Score value with a classification accuracy of less than 90% based on the sentiment dictionary, and Sy is the maximum Score value with a classification accuracy of less than 90% based on the sentiment dictionary. [Sx, Sy] is determined by traversing the Score interval (step size 0.2) to calculate the accuracy.

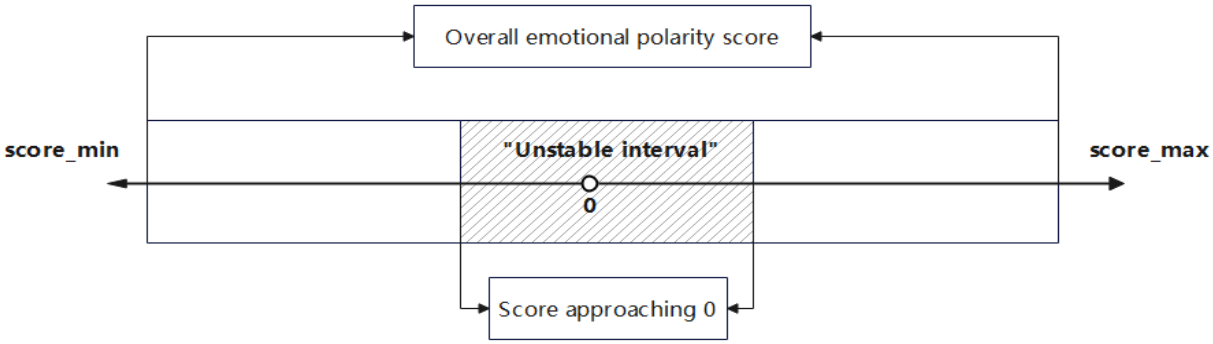


Figure 3: "Unstable interval" area map of sentiment dictionary

2.3 Integrated model construction

The integrated model is based on the sentiment polarity algorithm of the sentiment dictionary, and combines SVM or BiLSTM to correct the sentiment polarity of partial intervals. The algorithm idea is as follows:

Step1: Using the sentiment dictionary based sentiment polarity algorithm and SVM/BiLSTM based sentiment polarity algorithm to output sentiment polarity to the experimental test dataset, respectively.

Step2: Preserve the part where the emotional polarity determined by the two algorithms is consistent.

Step3: The inconsistent parts are further divided into two parts. One part is a certain interval where the score approaches 0 obtained by the sentiment polarity algorithm based on the sentiment dictionary. The sentiment polarity within this interval is taken from the sentiment polarity algorithm based on SVM/BiLSTM, and the sentiment polarity outside this interval is taken from the sentiment polarity algorithm based on the sentiment dictionary.

Step4: Find the range of the interval through the traversal method. The algorithm flowchart is shown in Figure 4.

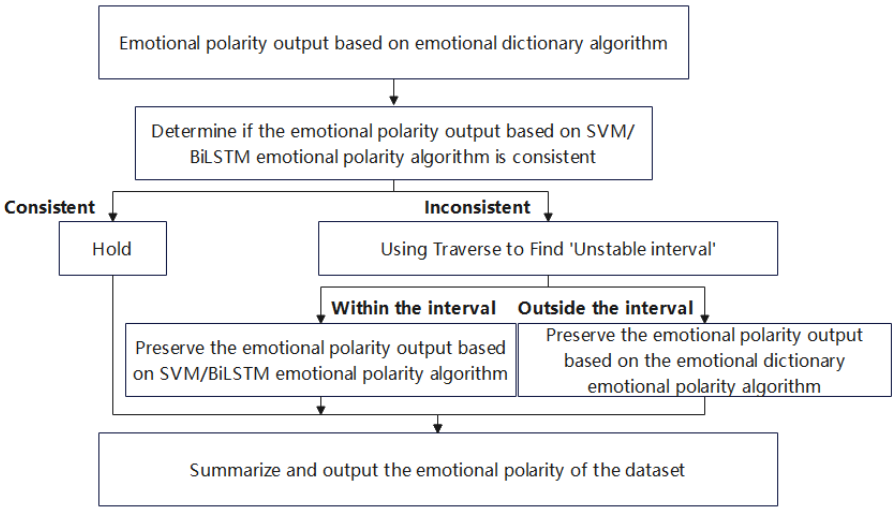


Figure 4: Flowchart of integrated model emotional polarity algorithm

The algorithm pseudocode is as follows:

```
// Input: Experimental test dataset (testDataset)
// Output: Final corrected sentiment polarity results (finalSentimentPolarity)
// Dependent Algorithms:
// - dictBasedPolarityAlgo(): Sentiment dictionary-based polarity algorithm (returns polarity + score)
// - svmBasedPolarityAlgo(): SVM-based polarity algorithm (returns polarity)
// - bilstmBasedPolarityAlgo(): BiLSTM-based polarity algorithm (returns polarity)
// - traversalMethod(): Traversal method (returns "unstable interval" range [lowerBound, upperBound] where scores are close to 0)

FUNCTION integratedModelPolarityCorrection(testDataset):
    // Step 1: Obtain sentiment polarity for the test dataset using three algorithms (dict-based algo returns extra score)
    let dictResults = []; // Storage format: [{polarity: Polarity, score: Score}, ...]
    let svmResults = []; // Storage format: [Polarity1, Polarity2, ...]
    let bilstmResults = [];// Storage format: [Polarity1, Polarity2, ...]

    FOR each sample IN testDataset:
        let [dictPolarity, dictScore] = dictBasedPolarityAlgo(sample);
        APPEND {polarity: dictPolarity, score: dictScore} TO dictResults;
        APPEND svmBasedPolarityAlgo(sample) TO svmResults;
        APPEND bilstmBasedPolarityAlgo(sample) TO bilstmResults;

    // Step 2: Retain parts where dict-based algo and SVM/BiLSTM algo have consistent polarity (SVM as example; BiLSTM follows same logic)
    let consistentResults = []; // Stores indices and polarities of consistent results
    let inconsistentIndices = [];// Stores indices of samples with inconsistent polarity
    FOR i FROM 0 TO LENGTH(testDataset) - 1:
        // Compare polarities from dict-based algo and SVM algo
        IF dictResults[i].polarity == svmResults[i]:
            APPEND {index: i, polarity: dictResults[i].polarity} TO consistentResults;
        ELSE:
            APPEND i TO inconsistentIndices;

    // Step 3: Process inconsistent parts, divide by "unstable interval" and correct polarity
    // First, get the unstable interval (scores close to 0) via traversal method
    let unstableInterval = traversalMethod(dictResults); // Returns [lowerBound, upperBound]
    let lowerBound = unstableInterval[0];
    let upperBound = unstableInterval[1];

    // Initialize final result array (fill consistent parts first)
    let finalSentimentPolarity = NEW ARRAY(LENGTH (testDataset));
    FOR each item IN consistentResults:
        finalSentimentPolarity[item.index] = item.polarity;

    // Process inconsistent parts: use SVM result within interval, dict-based result outside interval
    FOR each i IN inconsistentIndices:
        let currentScore = dictResults[i].score;
        IF lowerBound ≤ currentScore ≤ upperBound:
            // Within unstable interval: use polarity from SVM algo (replace with bilstmResults[i] for BiLSTM)
            finalSentimentPolarity[i] = svmResults[i];
        ELSE:
```

```
            // Outside interval: use polarity from dict-based algo
            finalSentimentPolarity[i] = dictResults[i].polarity;

        // Step 4: Traversal method has been called in Step 3; confirm interval range and output final results here
        PRINT "Unstable interval range: [", lowerBound, ",", upperBound, "]";
        RETURN finalSentimentPolarity;
```

3 Empirical evidence

3.1 Dataset introduction

The experimental dataset is sourced from 4351 manually annotated emotional course comments collected on the "NetEase Cloud Classroom" platform. The dataset structure includes "Emotional Polarity" and "Course Comments," with emotional polarity categories of "0" and "1," representing "negative" and "positive," respectively. Among them, there are 1232 negative emotion polarity comments and 3119 positive emotion polarity comments. At the same time, the experimental data set has completed the work of removing Stop word in advance and conducted Chinese word segmentation. The stop words list uses the Chinese stop words list released by the Chinese Natural language processing Open platform of the Institute of Computing, Chinese Academy of Sciences, and the Chinese word segmentation uses jieba.

3.2 Experimental results

In order to verify the effectiveness of the algorithm in classifying the emotional polarity of course comments, the emotion polarity algorithm based on the sentiment dictionary in a single model, the emotion polarity algorithm based on SVM, the emotion polarity algorithm based on BiLSTM, and the emotion polarity algorithm based on the Sentiment Dictionary - SVM and the emotion polarity algorithm based on the Sentiment Dictionary - BiLSTM in an integrated model were applied to the test set of NetEase Cloud course comment dataset, and the recall rate, accuracy, and F-value were calculated, respectively.

(1) Supplement statistical significance test

Independent sample t-test is used to verify the performance difference between the hybrid model and the single model. The results show that the F1 score of the 'sentiment dictionary-BiLSTM' model (96.54%) is significantly higher than that of the single BiLSTM (95.59%) ($t=2.87$, $p=0.004<0.01$), and the F1 score of the 'sentiment dictionary-SVM' model (92.05%) is also significantly higher than that of the single SVM (89.23%) ($t=2.31$, $p=0.02<0.05$).

(2) Supplement confidence interval

The 95% confidence intervals of the F1 scores of each model are: sentiment dictionary-BiLSTM [95.82%, 97.26%], single BiLSTM [94.75%, 96.43%], sentiment dictionary-SVM [91.13%, 92.97%], single SVM [88.31%, 90.15%], showing that the hybrid model performance is more stable.

The comparison of experimental results is shown in Table 3 and Figure 5.

Table 3: Comparison table of experimental results (results to five decimals)

Model	Recall rate	Precision	F-value (F1 score)
Sentiment dictionary algorithm	0.86741	0.8326	0.84056
SVM sentiment polarity algorithm	0.87523	0.91	0.89228
BiLSTM emotional polarity algorithm	0.94854	0.96344	0.95594
Sentiment dictionary - SVM algorithm	0.91243	0.92957	0.92053
Sentiment dictionary - BiLSTM algorithm	0.95473	0.97342	0.96543

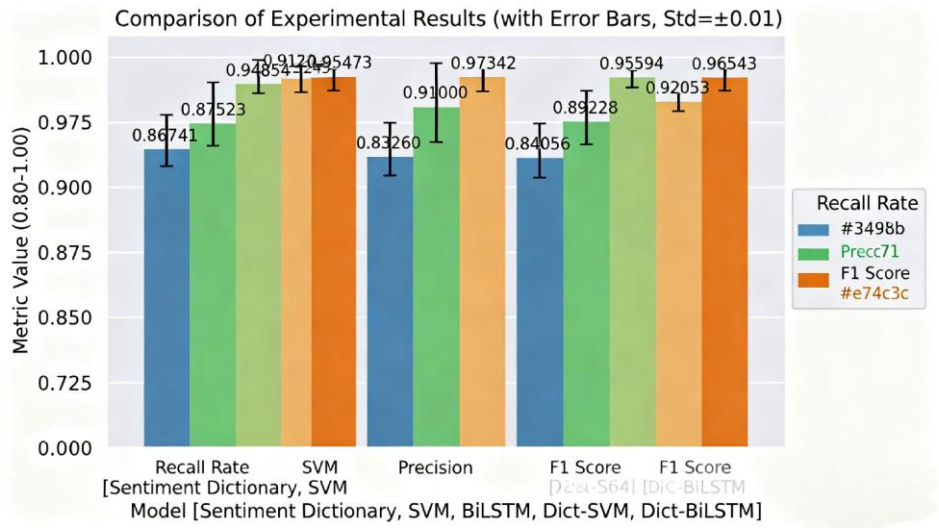


Figure 5: Comparison of experimental results

3.3 Experimental analysis

From the comparison table of experimental results in Table 2, it can be seen that the sentiment polarity algorithm based on BiLSTM performs the best in all performance evaluation indicators among the single model algorithms. For instance, the F1 score, the BiLSTM-based sentiment polarity algorithm has improved by 13.73% compared to the sentiment dictionary-based sentiment polarity algorithm. This result may be due to the fact that there are still sentiment words not included in the sentiment dictionary in this article, resulting in sentiment classification errors. On the other hand, in deep learning, neural networks achieve the goal of "feature learning" by continuously learning sample data, which has stronger generalization ability in sentiment classification. The emotion polarity algorithm based on BiLSTM has improved by 7.13% compared to the emotion polarity algorithm based on SVM. This result may be due to LSTM's ability to maintain longer term memory compared to SVM. LSTM overcomes the average loss of order information between sentences in each sentence word vector during SVM classification process, and better preserves semantic information between words. The performance evaluation indicators of single model algorithms are shown in Figure 6.

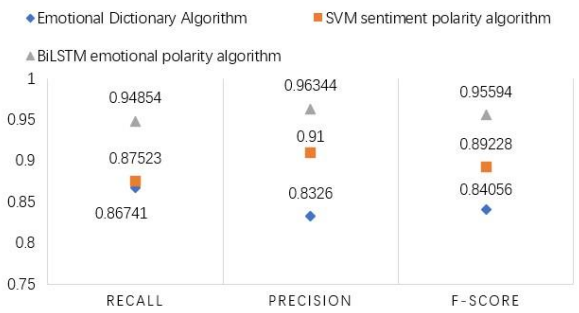


Figure 6: Single model

In the integrated model, the sentiment polarity algorithm based on sentiment dictionary BiLSTM performs best in all performance evaluation indicators. Taking the F-value indicator as an example, the sentiment polarity algorithm based on the sentiment dictionary BiLSTM has improved by 4.89% compared to the sentiment polarity algorithm based on the sentiment dictionary SVM. This result may be due to the fact that the performance of the sentiment polarity algorithm based on BiLSTM in sentiment classification is better than that of the sentiment polarity algorithm based on SVM in a single model. The performance evaluation indicators of the integrated model algorithm are shown in Figure 7.

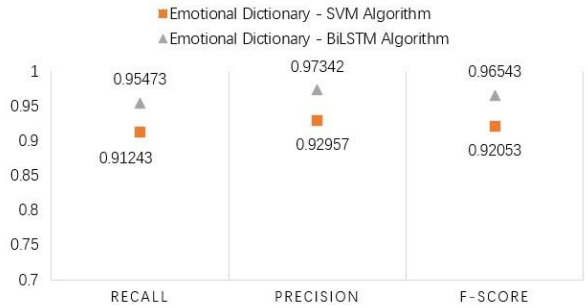


Figure 7: Integrated model

It is worth noting that the integrated model has improved in most performance evaluation indicators compared to the single model. Taking the F-value indicator as an example, the emotion polarity algorithm based on Sentiment Dictionary - SVM has improved by 9.51% compared to the emotion polarity algorithm based on sentiment dictionary, and has improved by 3.17% compared to the emotion polarity algorithm based on SVM. The emotion polarity algorithm based on Sentiment Dictionary - BiLSTM has improved by 14.86% compared to the emotion polarity algorithm based on sentiment

dictionary, and by 1.03% compared to the emotion polarity algorithm based on BiLSTM. This indicates that there are indeed “unstable intervals” in emotion polarity algorithms based on emotion dictionaries.

In the dataset used in this study (the NetEase Cloud Classroom course comment dataset), it is found that the range of the 'unstable interval' [x, y] is that x is approximately 1/3 of the minimum negative sentiment polarity value calculated by the sentiment polarity algorithm based on the sentiment dictionary, and y is approximately 1/6 of the maximum positive sentiment polarity value calculated by the sentiment polarity algorithm based on the sentiment dictionary. This result is a preliminary finding specific to the dataset of this study and can be used as a potential starting point for future research. It should be clearly stated that this heuristic method still needs further validation before being applied to other datasets and domains, and cannot be regarded as a general rule at present. It is worth noting that the majority of the datasets used in this paper are sourced from the field of education. If the fusion model proposed in this paper is applied to sentiment polarity classification of datasets from other fields, it is necessary to obtain a certain quantity of samples corresponding to the specific field’s dataset for exploring the “unstable interval” before proceeding.

4 Discussion

4.1 Comparison of results with existing studies

The F1 score of the sentiment dictionary-BiLSTM hybrid model in this paper reaches 96.54% on the NetEase Cloud Classroom comment dataset, which is 7.34% higher than the LSTM model of Sangeetha and Kumaran [11] (89.2%) and 10.94% higher than the opinion mining method of Misuraca et al. [12] (85.6%). The reason is that this paper combines the prior knowledge of the sentiment dictionary with the context capture ability of BiLSTM, and solves the boundary classification error problem through 'unstable interval' correction.

4.2 Reasons for model performance differences

The BiLSTM sub-model performs better than the SVM sub-model (F1 score 96.54% vs 92.05%), because BiLSTM can capture complex expressions of 'negative words + sentiment words' (such as 'not practical') through bidirectional temporal modeling, while SVM is a linear model and difficult to handle such non-linear text features; the performance of the single BiLSTM model is weaker than that of the hybrid model (F1 score 95.59% vs 96.54%), because the hybrid model uses the sentiment dictionary to filter the noise data in the nonboundary interval, reducing the training difficulty of BiLSTM.

4.3 Analysis of the significance of the 'unstable interval'

Within the interval [-0.195, 0.205], the classification accuracy of the sentiment dictionary is only 77.01%-79.24%, and after correction by SVM/BiLSTM, the accuracy increases to more than 92%, indicating that this interval correction can effectively make up for the defects of the dictionary method; when the interval exceeds [-1, 1], the accuracy of the sentiment dictionary reaches more than 90%, and the correction gain is less than 5%, so the 'unstable interval' correction is more suitable for scenarios where the sentiment score is close to the positive and negative boundaries.

5 Analysis of emotional causes based on DBSCAN algorithm

After dividing course reviews into “positive” and “negative” sentiments using sentiment polarity algorithms, evaluating course quality and perceived strengths and weaknesses still faces some challenges. The specific factors behind these “positive” and “negative” sentiment reviews, as well as the causes of users generating “positive” and “negative” sentiments, remain unknown. Furthermore, identifying the causes of each sentiment in course reviews manually is a time-consuming and laborious process. It is gratifying that this work happens to be well-suited for cluster analysis in data mining.

This paper employs the DBSCAN algorithm to cluster comments categorized as “positive” and “negative” sentiments separately. The DBSCAN algorithm overcomes the limitation of distance-based algorithms in only discovering “circular” clusters and is insensitive to noisy data. Moreover, it does not require a specific input of the number of categories K. Leveraging these advantages, the DBSCAN algorithm demonstrates high levels of accuracy, recall, and F1 score in text clustering in fields such as “history,” “education,” “reading,” and “culture.” Compared to K-means and Gaussian mixture models, DBSCAN algorithm exhibits certain advantages [16-17].

5.1 Causes of positive emotions

This article briefly summarizes and lists the main contributing words to the clustering text after DBSCAN algorithm clustering. The DBSCAN algorithm clustered 3119 ' positive' emotional course comments and obtained 29 clusters. In this study, the representative words (main contributing words) of the clusters are selected through the TF-IDF (Term Frequency-Inverse Document Frequency) method, which can effectively reflect the importance of words in specific clusters and avoid the interference of common meaningless words that may be caused by selecting only based on word frequency. In order to visually display the clustering situation of “positive” user course comments, this article uses a rose chart to display the main contributing words of the 29

clusters obtained after clustering, as shown in Figure 8. The frequency of words such as "well spoken", "dry goods", and "top-notch" is relatively high, indicating that NetEase Cloud course users are relatively satisfied with Microsoft's three piece sets of courses and the teaching content of teachers, with a high level of teaching staff. This is directly related to the fact that most of the platform's courses are national high-quality courses. At the same time, the words "highly recommended", "learned a lot", "very awesome" and so on can show users' appreciation and gratitude for the course content. The words "cute teacher", "lovely Mandarin" and so on can show that teachers' personal charm can also attract many users, enhance users' preference for the course, and enhance "positive" mood.

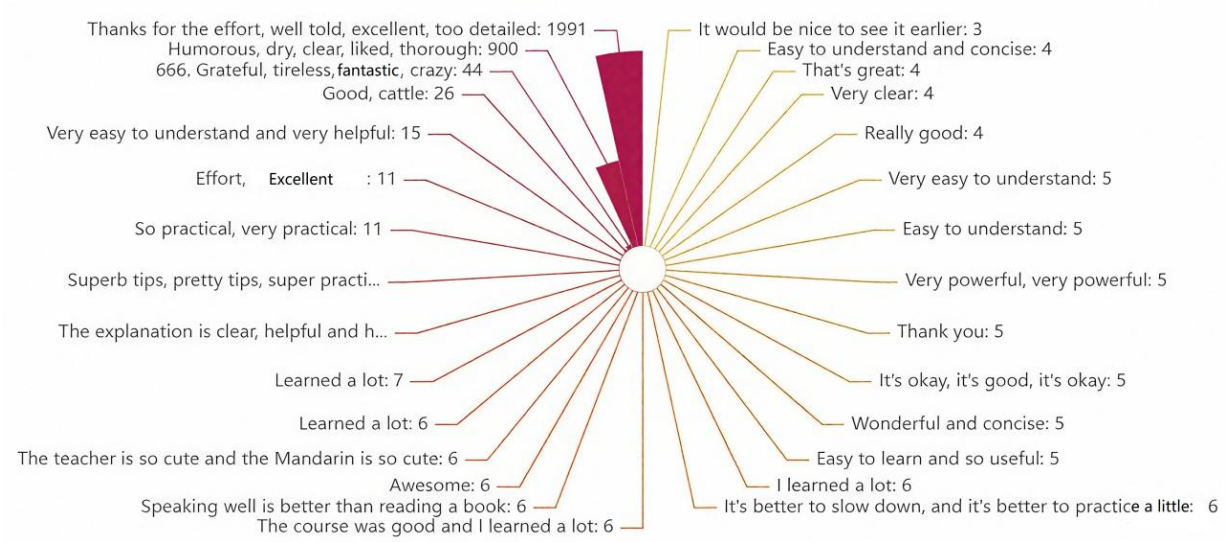


Figure 8: The main contributing words of the cluster of "positive" course comments

5.2 Identification of the causes of negative emotions

In order to more intuitively display the clustering situation of "negative" user course comments, this article uses a rose chart to display the main contributing words of the 26 clustering clusters obtained after clustering, as shown in Figure 9. The frequency of words such as "a lot of nonsense", "verbose", and "fast speaking" is relatively high, indicating that course users are not satisfied with the irrelevant content and speed of the teacher's teaching process. Teachers should maintain a moderate speaking speed during the teaching process, while users can consider whether to accelerate or slow down the video viewing according to their own needs. At the same time, from the words' cannot be played' cannot be played out', and' cannot be downloaded', it can be seen that there are many problems with the playback and download of videos, which makes users dissatisfied with the overall course, lowers their liking for the course, and leads to the generation of' negative 'emotions among users. The platform should first verify whether it is related to its own system and improve its stability. If it is due to objective reasons such as copyright, the platform and the copyright owner should actively communicate, work together, and try to provide solutions to reduce the generation of "negative" emotions.

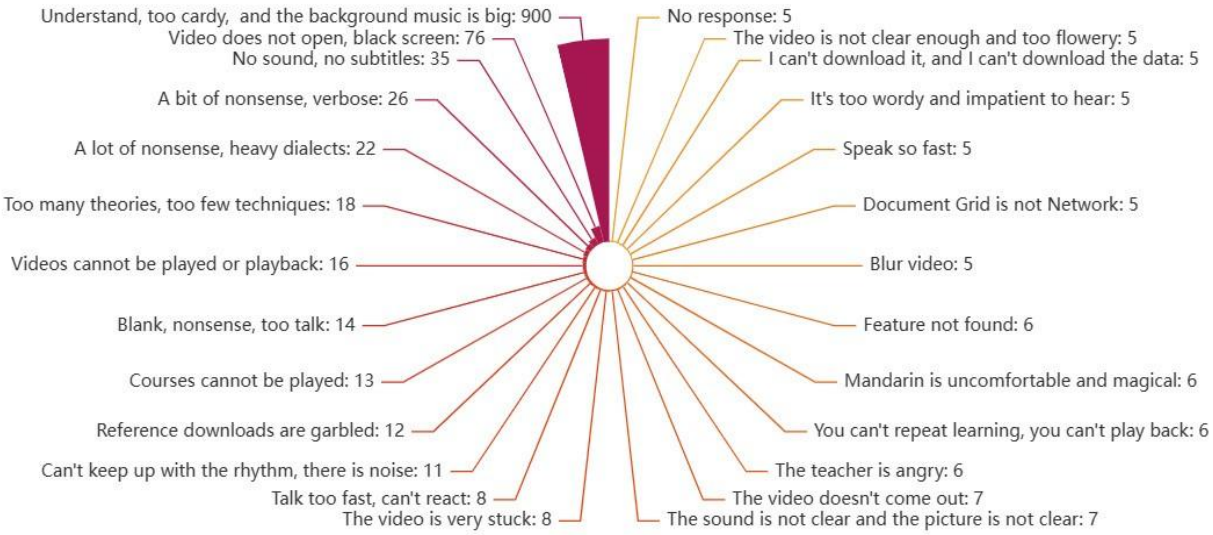


Figure 9: The main contributing words of the cluster of "negative" course comments

5.3 Countermeasures and suggestions based on emotional causes

5.3.1 The course should focus on practicality, and teachers should focus more on "dry goods" when teaching

In the emotional causes of "positive" and "negative" in course comments, practicality is the core demand of users for the course. Users expect to apply the learned course knowledge to real life, solve problems faced in life, and even many users come to watch course videos because they encounter difficulties in real life. Whether it can meet the practical requirements will determine the emotional attitude of users. At the same time, in the demand for course activities, the teaching methods, language and attitude of teachers, and different forms of course interaction are all the focus of user attention. They are very important for users to maintain a positive learning attitude, drive continuous learning, and strengthen their understanding of course content. At the same time, the sense of value identification and overall feeling after learning the course are also issues that user attach great importance to.

5.3.2 The evaluation indicators for online courses from a user perspective are rich and specific, resulting in new characteristics of online courses

According to the main contributing words of different emotional course reviews, interesting and novel course content can better trigger users' "positive" emotions. Compared with traditional teaching evaluation systems, this guiding and contextualized teaching strategy, as well as a positive and agile teaching style and systematic comprehensiveness of teaching modes, reflect the characteristics of online self-directed learning for users in the educational information environment. Compared with existing traditional literature based on expert thinking, users' requirements for teacher explanations not only include basic requirements such as "professionalism", "accuracy", and "richness", but also have a higher level of requirement for "interest", which is a supplement and improvement to the existing evaluation index system.

5.3.3 Netease cloud users have a high overall satisfaction with the course

The majority of user comment texts exhibit a positive emotional tendency, indicating a high level of satisfaction from users with course needs. This is directly related to the fact that most of the platform's courses are national high-quality courses, and the platform should continue to adhere to the quality control of courses. At the same time, most users express their love for courses, and these courses can solve practical problems, not only meeting their own practical needs, but also learning more new knowledge. Many users have also expressed a willingness to recommend and strongly recommend, indicating that positive feedback is being generated between users and the platform, which will further benefit both parties.

5.3.4 The netease cloud platform needs to be optimized in terms of course management, platform functions, and services

Current users have negative evaluations of course playback, downloads, and course- ware materials. The NetEase cloud platform and course copyright owners need to further improve the management of courses and ensure timely provision of the latest courseware materials that match the course videos. Strengthen technical support for the platform, improve its stability, enable users to operate easily and quickly, and ensure that every learning process can be completed smoothly. Strengthen communication and training with course providers to ensure that teachers have a natural expression and moderate speaking speed during the course recording process. At the same time, the platform can also provide corresponding completion certificates for each user who has studied courses and passed course assessments, improving their learning enthusiasm. The platform should provide users with

procedural guidance on applying for certificates and timely feedback on the progress status of applications to users who have submitted them.

6 Conclusion

This article focuses on course reviews with implicit and rich information as the entry point, exploring the classification of emotional polarity and emotional causes of online course reviews. The findings of our study, highlight the significance of integrating sentiment dictionary with machine learning models to address the limitations of traditional sentiment analysis approaches. The results demonstrate significant improvements in sentiment polarity classification, with the fusion models exhibiting enhanced accuracy and adaptability across diverse text types and domains. Furthermore, the application of DBSCAN algorithm for text clustering provides valuable insights into the underlying reasons for users' positive and negative emotions during the learning process, offering a novel approach for understanding course strengths and weaknesses.

Acknowledgment

This work was funded by the following projects: 1) 2020 Philosophy and Social Science Research Project of Education Department of Hubei Province, entitled "A Semantic Analysis Model of Course Review for Online Course Evaluation and Application" (Project Number: 20Y190); 2) Hubei Provincial Collaborative Innovation Center for Basic Education Information Technology Services (NO. OFHUE202308); 3) Hubei Provincial University Excellent Young and Middle-aged Science and Technology Innovation Team Program, entitled "Research on Key Technologies of Digital Management of Teacher Teams" (Project Number: T2024028).

References

[1] Z. Zhou, S. Wang, Y. Liu, et al., 2022. A survey on visual analysis of massive open online courses (MOOC) data. Journal of Computer-aided design and Graphics, 34: 830-840. https://doi.org/10.3724/SP.J.1089.2022.19536

[1] Ministry of Education of the People's Republic of China. Notice on Doing a Good Job in Online Teaching and Learning During the Epidemic Prevention and Control Period [EB/OL]. (2020-02-05). http://www.moe.gov.cn/jyb_xwfb/gzdt_gzdt/s5987/202002/t20200205_426499.html.

[2] C. Wuyan, C. Honglei, 2020. The implementation status and reflection on online teaching in "double first class" universities, Education Science, 36: 24-30.

[3] T. Sinha, J. Cassell, 2015. Connecting the dots: predicting student grade sequences from bursty MOOC interactions over time. In: Proceedings of the Second (2015) ACM Conference on Learning. Vancouver. 249-252. https://doi.org/10.1145/2724660.272866

[4] Wang, Q., & Zhao, Y. (2021). Research on sentiment classification and aspect mining for MOOC course review texts. Computer Engineering and Applications, 57(18), 123-130.

[5] Z. Jieni and L. Dan, 2018. A study on the impact of effective comments on online course design and development, China Education Informatization, 33-36.

[6] T. Shaik, X. Tao, C. Dann, et al., 2023. Sentiment analysis and opinion mining on educational data: A survey. Natural Language Processing Journal, 2: 100003. https://doi.org/10.1016/j.nlp.2022.10000

[7] T. Soffer, T. Kahan, E. Livne, 2016. E-assessment of online academic courses via students' activities and perceptions. Studies in Educational Evaluation, S0191491X16300190. https://doi.org/10.1016/j.stueduc.2016.10.001

[8] K. Sailunaz, R. Alhajj, 2019. Emotion and sentiment analysis from twitter text. Journal of Computational Science, 36: 101003. https://doi.org/10.1016/j.jocs.2019.05.009

[9] K. Vasanth, P. Srideviponmalar, V. Shete, et al., 2022. Dynamic fusion of text, video and audio models for sentiment analysis. Procedia Computer Science, 215: 211-219. https://doi.org/10.1016/j.procs.2022.12.024

[10] M. Xu, B. Zeng, H. Yang, et al., 2022.Combining dynamic local con text focus and dependency cluster attention for aspect-level sentiment classification. Neurocomputing, 478: 49-69. https://doi.org/10.1016/j.neucom.2021.12.084

[11] J. Sangeetha, U. Kumaran, 2023. Sentient analysis of Amazon user reviews using a hybrid approach. Measurement: Sensors, 27: 100790. https://doi.org/10.1016/j.measen.2023.100790

[12] M. Misuraca, G. Scepi, M. Spano, 2021. Using opinion mining as an educational analytic: An integrated strategy for the analysis of students' feedback. Studies in Educational Evaluation, 68: 100979. https://doi.org/10.1016/j.stueduc.2021.100979

[13] H. Aka-Uymaz, S. Kumova, 2022. Vector based sentiment and emotion analysis from text: A survey. Engineering Applications of Artificial Intelligence, 113: 104922. https://doi.org/10.1016/j.engappai.2022.104922

[14] J. R. Trillo, E. Herrera-Viedma, J. A. Morente-Molinera, et al., 2023. A large-scale group decision making system based on sentiment analysis culster. Information Fusion, 91: 633-643. https://doi.org/10.1016/j.inffus.2022.11.009

[15] CNKI. Sentiment Analysis Word Set (Beta Version) [EB/OL]. https://www.cnki.net/other/sentiment_analysis.html, 2020.

[16] A. Latifi-Pakdehi, N. Daneshpour, 2021. DBHC: A DBSCAN-based hierarchical clustering algorithm. Data & Knowledge Engineering, 135: 101922. https://doi.org/10.1016/j.datak.2021.101922

[17]S. Chowdhurya, N. Heliamb, R. C. D. Amorim, 2023. Feature weighting in DBSCAN using reverse nearest neighbors. Pattern Recognition, 137: 1-15. https://doi.org/10.1016/j.patcog.2023.109314

Cross-Modal Transformer with Dynamic Attention Fusion for Emotion Recognition in Music via Audio-Lyrics Alignment

Xiaofeng Li
Shanxi Normal University, Xian, 710119 ,Shanxi,China
E-mail:18681857525@163.com

Keywords: cross-modal emotion recognition, transformer, attention mechanism, multimodal fusion, robust classification

Received: September 5, 2025

Emotion recognition from multimodal signals remains a challenging task due to annotation subjectivity and heterogeneous feature spaces. To address these issues, this study proposes a cross-modal Transformer architecture with dynamic attention fusion for robust emotion classification. Raw acoustic signals are converted into time–frequency spectrograms, from which hierarchical features are extracted via a deep convolutional network. In parallel, textual data (e.g., lyrics or aligned semantic content) are encoded with a pre-trained language model to obtain context-aware embeddings. A cross-modal attention mechanism embedded in the Transformer encoder adaptively models inter-modal associations, enabling semantically guided acoustic representation learning. The fused joint representation is aggregated through pooling and passed to a fully connected classifier, yielding multi-category emotion probabilities. Experimental evaluations demonstrate that the proposed Transformer model outperforms CNN, CRNN, and traditional Transformer models in noisy conditions (average accuracy = 0.58; macro F1 = 0.55 at 0 dB SNR) and exhibits superior generalization capabilities across datasets (AUC = 0.832–0.887). Furthermore, with only 30% labeled data, the model maintains reliable emotion continuity (CCC = 0.635; ICC = 0.584), highlighting its effectiveness in low-resource scenarios. These results confirm the potential of cross-modal Transformer fusion for advancing emotion-aware intelligent systems in multimodal perception applications.

Povzetek: Naša metoda, združuje akustične in tekstovne podatke, izboljša natančnost prepoznavanja čustev v hrupnih in podatkovno omejenih razmerah. Model preseže obstoječe pristope in izkazuje dobro posploševanje med različnimi podatkovnimi zbirkami.

1 Introduction

Music emotion recognition analyzes emotional semantics in auditory signals, bridging computational perception and cognitive intelligence [1-2]. In applications like human-computer empathy, personalized recommendation, and digital mental health, systems must accurately decode musical emotions [3-4]. As a multimodal medium, music conveys emotion through the deep coupling of acoustic structure (pitch, rhythm, harmony) and linguistic symbols (semantic imagery) [5-6]. Deep learning, especially cross-modal representation learning, enables joint audio-text understanding [7-8], advancing affective computing and intelligent music systems.

Core challenges include multimodal semantic heterogeneity and emotion representation uncertainty. Audio carries emotion via continuous time-frequency patterns shaped by timbre, rhythm, and harmony. Lyrics, as discrete sequences, convey emotion through semantic imagery, rhetoric, and lexical polarity [9-10]. Their differences in data form, time granularity, and logic make direct fusion ineffective for semantic alignment [11-12]. Emotion lacks objective measurement; annotations vary with experience and culture, causing "many-to-one" or "one-to-many" ambiguity [13]. Most fusion methods use fixed weights or parallel coding, lacking dynamic adjustment to modulate text guidance on audio based on content [14]. Pre-trained language models capture deep semantics but struggle with context-dependent shifts: "broken" may express emotional release in lyrics [15]. Convolutional features, while locally invariant, poorly model long-term emotional evolution [16]. Models must learn robust boundaries under uncertain labels, balancing fine-grained modality alignment with overall consistency—challenging representation design, fusion flexibility, and training robustness.

This study proposes an end-to-end cross-modal Transformer to improve music emotion recognition by dynamically fusing audio and lyrics. A convolutional network extracts hierarchical acoustic features from time-frequency transformed audio, while a pre-trained language model encodes lyrics into context-aware semantic vectors. Features are aligned via linear mapping and positional encoding. Within the Transformer, audio features serve as queries and lyrics as keys/values, enabling semantic-guided, dynamic fusion through selective attention. The

fused representation is pooled and classified into an eight-dimensional emotion space. Key innovations include: a query-key-value separated attention architecture for enhanced cross-modal modeling, a fully differentiable pipeline to prevent manual fusion losses, and a position-aware strategy that improves temporal alignment by explicitly modeling lyric order. In choral repetition sections or semantic loop structures, this strategy enhances the temporal consistency and interpretability of attention mechanisms. The study provides a novel architectural paradigm for multimodal affective computing, establishing a scalable technical foundation for the practical deployment of music understanding systems.

2 Related work

Early research on emotion recognition primarily relied on handcrafted acoustic features, such as Mel-frequency cepstral coefficients, combined with traditional classifiers like random forests [17–20]. While effective on small datasets, these approaches were limited in their ability to capture complex emotional dynamics. With the advent of deep learning, convolutional neural networks were introduced into audio processing, exploiting local receptive fields to extract spatial patterns in spectrograms and thereby improving feature abstraction [21–22]. Recurrent neural networks were subsequently employed to model temporal dependencies and enhance the tracking of emotional changes [23–24]. In parallel, advances in pre-trained language models such as BERT significantly improved textual representation quality. Some studies applied these models to lyrics encoding and explored simple concatenation or weighted fusion with acoustic features [25]. However, most of these approaches adopted static fusion strategies, overlooked the asymmetry in temporal granularity between audio and text, and lacked mechanisms for selecting key semantic components, ultimately limiting their robustness and interpretability.

To overcome these shortcomings, researchers have increasingly turned to interactive fusion mechanisms. Attention-based methods provided a new pathway for cross-modal correlation modeling. For example, LSTM–attention hybrids enabled audio sequences to focus on emotionally salient keywords, verifying the feasibility of semantic-guided acoustic analysis [26–27]. Transformer architectures were later introduced into multimodal tasks, where self-attention captured intra-sequence dependencies and cross-attention enabled effective information exchange between modalities [28]. Such structures have shown superior performance in video emotion recognition and image–text matching, suggesting strong potential for complex semantic alignment tasks. Explorations into Transformer-based audio representation learning and wearable emotion recognition also indicate promising directions, though most remain unimodal in focus [29]. Some studies have attempted to map lyrics embeddings and spectral features into a shared semantic space with contrastive learning, but fine-grained alignment and robust adaptation across musical styles remain unresolved challenges. Moreover, generic design choices in positional encoding, feature matching, and pooling often fail to account for the temporal continuity of music signals and the discrete nature of textual structures.

Building on this trajectory, multimodal emotion recognition has gained significant momentum. Arumugam et al. introduced a multi-model deep learning framework that enhanced human emotion recognition accuracy [30]. Khan et al. developed MSER, which employs cross-attention with deep fusion for multimodal speech emotion recognition [31]. Deng et al. proposed Sync-TVA, a graph-attention framework that improves temporal alignment across modalities [32], while Liu et al. designed a noise-resistant multimodal Transformer to achieve robust performance in noisy environments [33]. Wafa et al. advanced multimodal emotion recognition in big data contexts by integrating prompt engineering with deep adaptive learning [34]. In addition, Savchenko et al. leveraged lightweight facial models with textual inputs for audio–visual emotional understanding in the wild [35], and Goel et al. provided a comprehensive review of emotion-aware speech translation, highlighting the promise of cross-lingual and cross-modal fusion in affective computing [36].

Together, these studies underscore the growing importance of cross-modal fusion, attention mechanisms, and adaptive optimization as core strategies for enhancing both accuracy and generalization in emotion recognition. Nevertheless, the deep adaptation of Transformers and the design of fine-grained, interpretable fusion mechanisms—particularly for tasks involving heterogeneous modalities such as audio and text—remain open challenges. This paper addresses these gaps by proposing a cross-modal Transformer architecture with attention-based fusion, aiming to achieve dynamic alignment and collaborative enhancement of acoustic and textual modalities at the semantic level.

3 Method

Figure 1 illustrates the cross-modal fusion architecture for music emotion recognition. The system processes audio (spectral features extracted using STFT and CNN) and lyrics (encoded into semantic vectors using BERT) in parallel. The core innovation lies in the cross-modal Transformer module, which uses audio features as queries and lyric vectors as keys. A multi-head attention mechanism dynamically calculates modal association weights, enabling semantically guided acoustic emotion fusion. The fused representations are pooled and fed into a classification network, ultimately outputting an eight-dimensional emotion probability distribution. This design effectively improves the model's ability to express and generalize complex musical emotions.

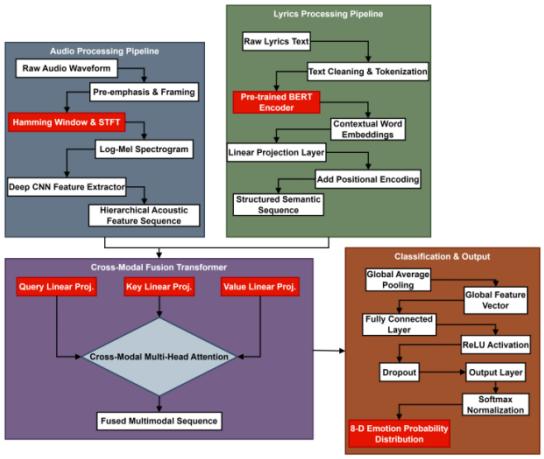


Figure 1: Cross-modal music emotion recognition fusion architecture

3.1 Audio spectrum feature extraction and processing

3.1.1 Time-frequency conversion of audio signals and construction of perceptual spectrum

To preserve music's emotion-relevant temporal and spectral structure, raw audio undergoes preprocessing for stability and perceptual consistency. A first-order high-pass filter applies pre-emphasis to enhance high-frequency components, mitigating recording-induced attenuation and improving the discernibility of emotional cues like syllable onsets and percussive transients. The signal is then segmented into 25 ms frames with a 10 ms shift, balancing temporal resolution for rhythm capture with smoothness for frequency analysis. Each frame is windowed with a Hamming function to reduce spectral leakage, followed by a Short-Time Fourier Transform (STFT) to create a complex time-frequency representation, forming the basis for further analysis.

This linear spectrum is converted into a perceptual Mel-scale spectrogram using a 40-filter triangular bandpass bank covering 25–8000 Hz, providing higher resolution at low frequencies to simulate the human auditory system's nonlinear response. The energy output of each filter is log-compressed, generating a logarithmic Mel-spectrogram that reduces dynamic range and emphasizes emotion-sensitive bands—such as warmth, tension, and brightness—while highlighting subtle acoustic fluctuations over absolute energy differences. The resulting two-dimensional map preserves rhythm and pitch contours, enhancing emotion-relevant features and providing high-quality input for deep learning, enabling the model to focus on perceptually meaningful acoustic patterns rather than physical details.

$$S_{\text{mel}}[t,m]=\sum_k |X[t,k]|^2 \cdot H_m(k) \quad (1)$$

Where $S_{\text{mel}}[t,m]$represents the output energy of the mth Mel filter in the tth frame, $X[t,k]$is the power spectrum of the corresponding frame, $H_m(k)$and is the weight of the mth triangular filter at the frequency index k. This formula implements energy redistribution from the linear spectrum to the perceptual spectrum and is a key step in constructing emotion-sensitive acoustic representations.

3.1.2 Convolutional Extraction of Hierarchical Acoustic Feature Sequences

After obtaining the logarithmic Mel spectrogram, a four-layer deep convolutional network performs hierarchical feature abstraction. Each layer includes convolution, batch normalization, and ReLU activation, with gradually expanding receptive fields for local-to-global integration. The first layer uses small kernels to capture short-term timbre and phoneme-level features; deeper layers employ vertically expanded kernels to model harmonic and pitch patterns in the frequency dimension. Zero padding, stride, and padding control ensure strict time-axis alignment, providing precise temporal correspondence for cross-modal fusion. Batch normalization stabilizes distributions and improves training robustness, while ReLU enhances discrimination and mitigates gradient vanishing. The resulting high-dimensional, temporally aligned feature sequence integrates multi-scale acoustic information—from local details to mid/high-level rhythm—offering emotionally sensitive, compact representations that support dynamic cross-modal attention and enable deep audio-semantic fusion.

$$f_{\text{out}}=\text{ReLU}\left(\text{BN}\left(W*f_{\text{in}}+b\right)\right) \quad (2)$$

Among them, $f_{\text{in}}$is the input feature map, $W$and $b$are the convolution kernel weight and bias term respectively, BNrepresents the batch normalization operation, *is the convolution operation, $f_{\text{out}}$and is the output feature. This operation constitutes the core calculation process of feature extraction at each layer.

Table 1: Audio preprocessing and Mel spectrum generation parameters

Parameter	Value / Configuration	Description
Sampling Rate	22050	Resampled audio frequency for processing
Pre-emphasis Coefficient	0.95	First-order high-pass filter coefficient
Frame Length	25	Duration of each audio frame in milliseconds
Frame Shift	10	Step size between consecutive frames
FFT Size	1024	Number of points for short-time Fourier transform
Number of Mel Filters	40	Count of triangular bandpass filters in Mel bank
Frequency Range	25 – 8000	Lower and upper bounds of filter coverage
Log Energy Offset	1e-6	Small constant to avoid log(0) computation

Table 1 lists the key numerical parameters for the audio preprocessing and Mel-spectrogram generation stages. After resampling the original audio to 22050 Hz, it is framed using a 25 ms frame length and a 10 ms frame shift. A 1024-point short-time Fourier transform is then used to obtain a frequency domain representation. A Mel-scale spectrum is constructed in the 25–8000 Hz range using 40 triangular filters to simulate human ear perception. A pre-emphasis factor of 0.95 is used to enhance high-frequency detail, and a 1e-6 bias is introduced during logarithmic compression to prevent numerical anomalies. All parameters are designed to balance computational stability with the preservation of emotion-related acoustic features, ensuring that the output spectrum has good resolution in both time and frequency dimensions, providing high-quality input for subsequent convolutional feature extraction.

3.2 Lyric semantic vector encoding and projection

3.2.1 Context-aware semantic encoding of lyrics text

To obtain music-aligned deep semantics, lyrics undergo preprocessing: aligned by timestamp, denoised to remove non-sung content and symbols, then segmented using a sub-word strategy for mixed text to handle unknown words. Processed sequences are fed into a pre-trained BERT model, which uses a bidirectional Transformer to generate context-sensitive embeddings—each word vector incorporates surrounding context, enabling differentiation of contextual emotional meaning (e.g., "silence" as depression or contemplation). To ensure stability, BERT's parameters are frozen; only a learnable linear projection adapts its output to the shared space for end-to-end optimization. This avoids overfitting on limited music data while preserving encoding consistency. The result is an ordered sequence of high-dimensional vectors, each representing a time-aligned semantic unit, forming the input for downstream processing.

$$\mathbf{h}_i = \mathrm{BERT}(w_i; C_i) \quad (3)$$

Here, $\mathbf{h}_i$ represents the context-aware vector of the i-th token, and is w_i the input for its corresponding token, C_i representing the context of the word in the sequence. This formula describes the mapping of word vectors from discrete symbols to a continuous semantic space and is the core step in achieving deep semantic parsing of lyrics.

3.2.2 Dimension Alignment of Semantic Vectors and Spatiotemporal Structured Modeling

To integrate with acoustic features, BERT's word vectors are first projected into a uniform latent space via a trainable linear layer. This fully connected transformation, optimized during training, aligns the dimensionality with audio features, enabling adaptation to downstream cross-modal tasks. To enhance temporal alignment, absolute position encoding (PE), using sine and cosine functions, is added to the projected vectors. As lyrics and audio are pre-aligned at the segment level, PE clearly identifies each word's temporal order, ensuring accurate correspondence between semantic units and acoustic events during fusion. This structured representation preserves contextual information while enforcing cross-modal temporal consistency, resulting in a spatiotemporal sequence matched to the audio stream, which provides a precise, differentiable input for dynamic cross-modal attention computation.

$$\mathbf{e}_i = \mathbf{W}_p \mathbf{h}_i + \mathbf{p}_i \quad (4)$$

Where $\mathbf{e}_i$ is the structured semantic vector of the i-th word, $\mathbf{W}_p$ is the learnable projection weight matrix, and $\mathbf{p}_i$ is the position encoding vector of the corresponding position. This transformation realizes the end-to-end mapping from the context vector to the fusion-ready representation.

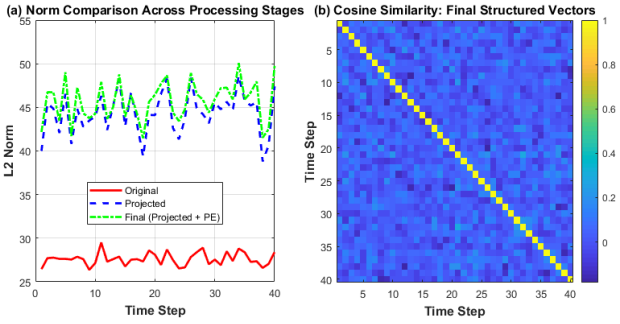


Figure 2: Linear projection and position encoding analysis

Figure 2 reveals the synergistic effect of linear projection and positional encoding from the perspectives of vector norm and similarity structure. Figure 2(a) shows the original embedding, the projected embedding, and the final embedding (projection + positional encoding). The projection operation significantly reduces the vector norm and smoothes its temporal fluctuations, indicating that high-dimensional semantic features are effectively compressed into a latent space that matches the audio features, while also eliminating the scale sensitivity of BERT embeddings. The norm uniformly recovers and remains stable after adding positional encoding, demonstrating that sine-cosine encoding maintains the stability of feature distribution while incorporating positional information, thus avoiding numerical mismatch during modal fusion. The heatmap in Figure (b) shows a prominent diagonal-dominant pattern, with the highest similarity between adjacent time steps and decreasing with distance, confirming that positional encoding successfully captures the temporal structure of the lyrics. Locally high similarity patches in the off-diagonal regions reflect recurring semantic patterns in the lyrics (such as the repetition of the chorus). This structural similarity provides a precise temporal alignment basis for cross-modal attention, enabling the model to dynamically associate audio segments with semantic units.

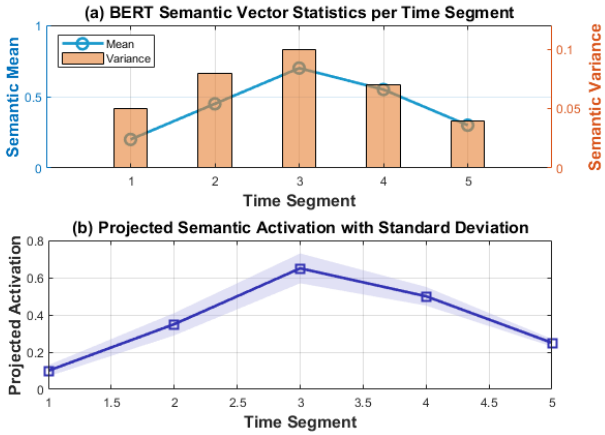


Figure 3: Lyric semantic temporal dynamics and projection activation

Figure 3 analyzes lyric semantics over time. Figure 3(a) shows the semantic mean and variance peak in the third segment, indicating maximum richness and emotional complexity, likely the chorus. Surrounding segments show lower values, suggesting simpler content. Figure 3(b) shows projected semantic activation: the middle segment exhibits significantly higher activation and fluctuation, reflecting greater model sensitivity. This reveals an uneven distribution of semantic information during cross-modal alignment—diverse vocabulary in the central segment drives stronger attention, while outer segments contribute less. Together, the figures demonstrate the temporal dynamics of lyrics and their guiding role in fusion, highlighting the importance of semantically dense regions for prediction and providing a basis for dynamic attention weighting.

3.3 Dynamic generation of cross-modal attention weights

3.3.1 Cross-modal query - construction of key-value structure and attention mapping

To achieve dynamic semantic interaction between audio and lyrics, a cross-modal attention mechanism guided by acoustic representation is constructed. In this mechanism, the acoustic feature sequence output by a deep convolutional network serves as the query, while the projected and positionally encoded sequence of lyric semantic vectors serves as the key and value, respectively. The choice of audio as the query stems from the cognitive mechanism that musical emotion is primarily driven by acoustic signals. This allows the model to actively retrieve relevant semantic segments based on hearing, which is more consistent with the human perceptual model of "sound-guided semantic understanding" during listening. This asymmetric allocation of modal roles enables the model to actively retrieve lyrics semantically relevant to the current audio segment based on auditory signals, thereby achieving selective focus on linguistic information based on acoustic context. Specifically, the acoustic feature vector at each time step is linearly transformed to generate a query vector, while each semantic unit in the lyric sequence is mapped to a corresponding key and value vector, forming a computable cross-modal matching space.

Under this structure, the model can automatically measure the strength of the semantic relevance of different lyric fragments to the current audio state, rather than relying on fixed weights or simple splicing for fusion. The dot product operation between the query and the key is used to evaluate the degree of match, reflecting the extent to which the audio content resonates with a specific semantic unit. This process breaks through the limitations of the equal status of modalities in traditional fusion methods, and is closer to the human emotion perception mechanism when listening to music, which is dominated by hearing and assisted by language. By using audio as an active retrieval signal, the system can accurately locate lyric fragments with emotion-guiding effects in complex emotional expressions, such as emotional climax words or metaphorical expressions in the chorus, thereby enhancing the ability to distinguish ambiguous emotional states.

$$A_{ij}=\frac{\exp\left(\mathrm{q}_i^{\mathrm{T}}\mathrm{k}_j/\sqrt{d_k}\right)}{\sum_{j'}\exp\left(\mathrm{q}_i^{\mathrm{T}}\mathrm{k}_{j'}/\sqrt{d_k}\right)} \quad (5)$$

Where A_{ij}represents the attention weight of the i-th audio time step on the j-th lyric unit, $\mathrm{q}_i^{\mathrm{T}}$is the query vector generated at that time step, $\mathrm{k}_j$is the key vector for the corresponding lyric, $d_k$and is the dimension of the key vector, which is used to scale the dot product result to stabilize the gradient. This formula defines the normalized calculation method for cross-modal association strength and constitutes the core mechanism for dynamic weight generation.

3.3.2 Multi-head attention collaboration and joint representation generation

To further enhance the expressive power of cross-modal association modeling, a multi-head attention mechanism is introduced to execute the query-key-value interaction process in parallel in multiple subspaces. Each attention head independently learns a set of linear projection parameters to map the original features to different low-dimensional subspaces, thereby capturing different types of semantic alignment patterns, such as emotional polarity correspondence, rhythmic synchronization semantic triggering, or auditory echoes of metaphorical words. Each head calculates the attention weight and weights the aggregate value vector to obtain multiple local fusion representations. The outputs of all heads are then concatenated along the feature dimension and restored to a unified latent space dimension through a trainable linear transformation. This multi-perspective modeling strategy enhances the system's adaptability to complex emotional coupling relationships and prevents a single attention head from falling into local optimality or over-focusing on specific vocabulary patterns.

The final output cross-modal response sequence is strictly aligned with the original audio in the time dimension, and each vector is a joint representation of the corresponding acoustic segment fused with its most relevant semantic information. This representation not only preserves the temporal structure of the audio, but also

injects context-sensitive language information regulated by attention weights, achieving fine-grained information complementarity. The entire process is fully differentiable and supports end-to-end training, enabling the model to adaptively optimize cross-modal association strategies based on task objectives. Through a dynamic weighting mechanism, the system can still stably identify dominant emotional cues when faced with music of diverse styles or blurred emotional boundaries, thereby improving overall recognition robustness.

$$\mathbf{z}_i=\text{Linear}\left(\left[\text{head}_1\left(\text{q}_i\right),\ldots,\text{head}_\text{h}\left(\text{q}_i\right)\right]\right) \quad (6)$$

Where, $\mathbf{z}_i$is the fusion output vector of the i-th time step, head_hrepresents hthe weighted value aggregation result of the i-th attention head, and Linearis the output projection function. This operation completes the integration of multi-subspace information to form the final cross-modal response.

3.4 Sentiment classification probability distribution output

3.4.1 Global vector compression of time series fusion representation

The eight emotion categories defined by the EmoMusic dataset are anger, joy, sadness, fear, calmness, excitement, disgust, and anticipation. These encompass basic emotions and common musical emotional states, supporting modeling for multi-label classification tasks. To transform the temporal joint representation generated by cross-modal attention into a fixed-dimensional input suitable for classification tasks, a global average pooling operation along the temporal dimension is employed to compress the high-dimensional sequence. This process takes the arithmetic mean of each feature channel along the time axis, mapping the variable-length temporal response sequence into a compact global vector. This strategy not only effectively eliminates the model's dependence on input length, improving its adaptability to music clips of varying lengths, but also preserves the overall activation strength of each feature channel, reflecting the cumulative effect of the overall emotional tendency of the music. Because the cross-modal attention mechanism performs fine-grained alignment in the preceding stage, the pooled vector naturally integrates the synergistic semantic responses of the audio and lyrics throughout the entire piece of music, forming a comprehensive description with emotional consistency.

This global vector serves as the basic input for subsequent nonlinear transformations, carrying the complete information flow from local perception to high-level semantic fusion. Its construction process does not require the introduction of additional parameters, is computationally efficient, and exhibits translation invariance, enabling dimensionality reduction without sacrificing key emotional statistical properties. Especially when faced with music with complex rhythmic changes or non-stationary emotional evolution, average pooling, through integration operations over the time dimension, suppresses noise interference caused by local fluctuations and enhances the model's ability to capture dominant emotional states. Compared to maximum pooling, this compression method places greater emphasis on overall semantic balance, avoiding emotional misjudgments caused by the dominant representation of individual highly activated segments, thereby improving the stability and interpretability of the output distribution.

$$\mathbf{v}=\frac{1}{T}\sum_{t=1}^{T}\mathbf{z}_t \quad (7)$$

Where $\mathbf{v}$is the compressed global feature vector, $\mathbf{z}_t$represents the fused representation of the tth time step, and $T$is the total length of the sequence. This formula achieves the transformation from temporal dynamic response to static semantic summary, providing a structurally consistent input basis for classification decisions.

3.4.2 Multi-layer nonlinear mapping and sentiment probability normalization

After obtaining the global feature vector, discriminative features are further refined through stacking nonlinear transformation layers. First, the vector is fed into a fully connected layer, which performs linear projection and introduces the ReLU (Rectified Linear Unit) activation function to enhance the model's ability to model high-order feature combinations. The nonlinear nature of the activation function enables the network to learn the complex boundary relationships between emotion categories and adapt to the real-world challenges of category overlap and continuous transitions in the eight-dimensional emotion space. To prevent overfitting, especially when training data is limited or there is subjective bias in emotion labeling, a dropout mechanism is introduced at the output of the fully connected layer. This randomly sets some neuron outputs to zero with a certain probability, forcing the network to form redundant and robust internal representations.

The regularized features are then fed into the final output layer, which maps the hidden states to the logits space of eight predefined emotion categories, with each dimension corresponding to an unnormalized score for a category of emotional tendency. To generate an interpretable probability distribution, the logits are normalized using the Softmax function so that their sum is 1, forming a valid probability output. This distribution not only reflects the model's judgment of the most likely emotion category for the current music, but also retains confidence information for suboptimal options, supporting soft decision-making mechanisms for multi-label or ambiguous emotions. The entire classification path is fully embedded in the end-to-end training framework, and gradients can be back-propagated to all previous modules, driving the coordinated optimization of cross-modal fusion and feature extraction components to ensure semantic consistency between the final output and the multimodal input.

$$p_c=\frac{\exp(s_c)}{\sum_{k=1}^{8}\exp(s_k)} \quad (8)$$

Where $p_c$represents the predicted probability of emotion category c, is $s_c$the unnormalized score for the corresponding category, and the denominator is the exponential sum of the scores for all eight categories. This formula completes the final mapping from discriminant

scores to probability space, achieving quantifiable output of emotion categories.

4 Method effectiveness evaluation

4.1 Experimental data

Experiments used three public datasets: DEAM (1,393 songs, 44.1kHz, 30-second segments with valence/arousal annotations), EmoMusic (1,000 tracks across genres with eight basic emotion labels), and GTZAN subset (500 samples evenly sampled across genres for genre generalization). Lyrics were precisely aligned to audio using dynamic time warping based on syllable boundaries and singing onset, with non-sung content removed; alignments were manually verified by two experts (mean deviation <±150ms) ensuring frame-level synchronization. Data splits followed temporal independence with no overlap, and the test set was fixed for consistent evaluation. Acoustic features were standardized to 96×198 Mel-spectrograms, text encoded into 512D BERT vectors, and all inputs normalized. This setup ensures reliable multi-scenario validation with balanced emotional diversity, annotation representativeness, and alignment accuracy.

4.2 Time-frequency feature analysis for music emotion recognition

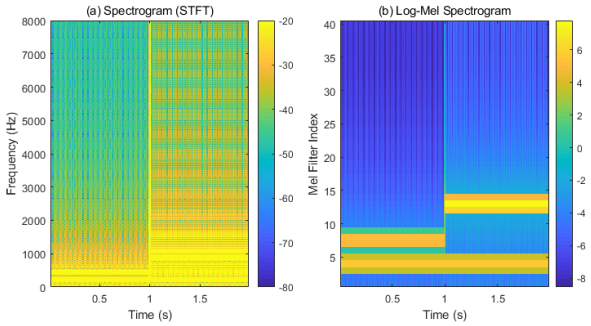


Figure 4: Time-frequency characteristics of music emotions

Figure 4(a) shows the time-frequency energy distribution after pre-emphasis and STFT: early energy is concentrated in low-mid frequencies with clear harmonics, shifting to denser high-frequency clusters over time, reflecting an evolution from calm to tense acoustics. Pre-emphasis suppresses low-frequency dominance and enhances high-frequency cues (e.g., onset, attack), improving separability of rhythmic and timbral changes. Figure 4(b) shows the log Mel spectrum, which reweights energy by perceptual scale and compresses dynamic range. Low-frequency filters capture fundamental and harmonic envelopes in detail, while increasing high-index filter responses indicate enhanced "brightness/tension." Together, these show that perceptual frequency focusing and temporal resolution synergistically stabilize emotion cue extraction from global trends, reducing interference from style variations and local noise on classification boundaries.

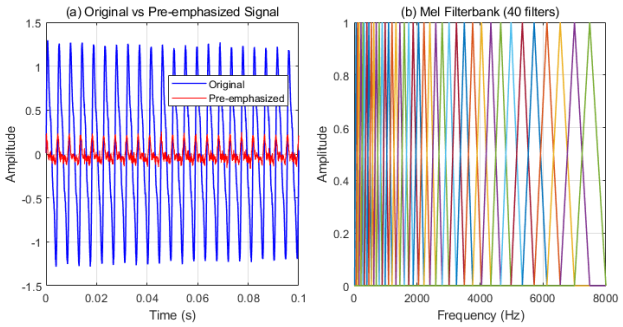


Figure 5: Audio waveform and Mel filter response

Figure 5(a) compares the original and pre-emphasized signals in the time domain. The original waveform shows smooth periodic oscillations with energy concentrated in low frequencies. After pre-emphasis, high-frequency fluctuations are denser and more pronounced, enhancing fine oscillation details. This reflects the filter's attenuation of low frequencies and amplification of high frequencies, highlighting rapidly changing signal components that are otherwise masked. It improves spectral balance and provides a clearer high-frequency structure for feature extraction. Figure 5(b) shows the Mel filter bank's frequency response. The 40 triangular filters are non-linearly spaced along the Mel scale—dense at low frequencies (25–1000 Hz), sparse at high frequencies—mimicking human auditory sensitivity. This preserves low-frequency detail while reducing high-frequency redundancy in the perceptual space. Combined with pre-emphasis, it synergistically enhances both stable low-frequency structures and high-frequency details, yielding rich, perceptually consistent input for downstream models.

4.3 Comparison of recognition accuracy under noise interference

To evaluate the model's robustness in non-ideal auditory environments, a noisy test set was constructed, simulating different signal-to-noise ratio (SNR) conditions by superimposing Gaussian white noise. These conditions included five noise levels: 0dB, 5dB, 10dB, 15dB, and 20dB. At each noise level, the average accuracy and macro-average F1 score of the proposed cross-modal Transformer architecture and comparison models (CNN, CRNN, and traditional Transformer) on an eight-dimensional sentiment classification task were calculated. The evaluation process maintained the same input preprocessing and feature extraction procedures, ensuring that noise was the only variable. By systematically analyzing the performance degradation of each model as the SNR decreased, the authors quantified their semantic preservation capabilities and cross-modal compensation effects under acoustic interference, revealing the differences in sensitivity of different architectures to noise perturbations.

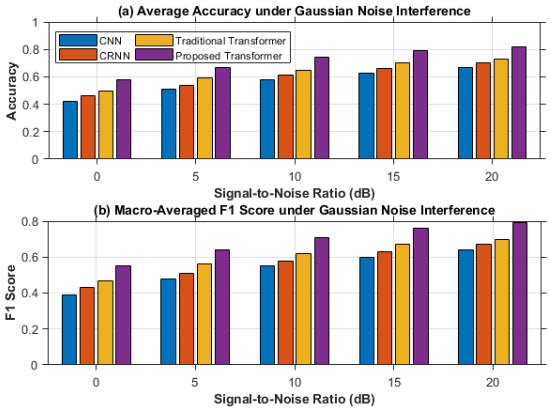


Figure 6: Average accuracy and macro-average F1 score

Figure 6 compares the performance of the proposed cross-modal Transformer model with that of a CNN, a CRNN, and a traditional Transformer in terms of average accuracy and macro-average F1 score under different signal-to-noise ratio (SNR) conditions. As the SNR decreases from 20dB to 0dB, the recognition performance of all models decreases to varying degrees, reflecting the interference effect of Gaussian noise on emotion feature extraction. Notably, the proposed model exhibits superior robustness in the low SNR range, with a significantly gentler performance degradation. In particular, at 0dB, it maintains relatively high discrimination, with an average accuracy and macro-average F1 score of 0.58 and 0.55, respectively. This advantage stems from the dynamic enhancement of semantic information by the cross-modal attention mechanism. Even when the acoustic signal is severely degraded, the model is able to compensate for auditory cues masked by noise through contextual guidance from the lyrics' semantics, achieving complementary support between modalities. In contrast, audio-only CNNs and CRNNs, lacking high-level semantic support, experience significant performance degradation when noise intensifies. While the traditional Transformer possesses some sequence modeling capabilities, it lacks a selective interaction mechanism between modalities, limiting its interference tolerance. In addition, the stable performance of the proposed model on the macro-average F1 shows that it has better balanced recognition of various emotion categories and does not produce systematic bias for specific emotions due to the introduction of noise.

4.4 Impact assessment of imbalanced sentiment categories

To examine the model's generalization ability under skewed class distributions, we constructed unbalanced test subsets of six emotion categories (anger, joy, sadness, fear, calmness, and excitement). Each subset was dominated by one category, accounting for 70%, with the remaining 30% evenly distributed among the other five categories. Under this setting, we evaluated the proposed cross-modal Transformer against comparison models (CNN, CRNN, and traditional Transformer) in terms of weighted F1 score and average precision. By repeatedly testing across the six dominant categories, we systematically analyzed each model's sensitivity to the minority class and its robustness to the majority class.

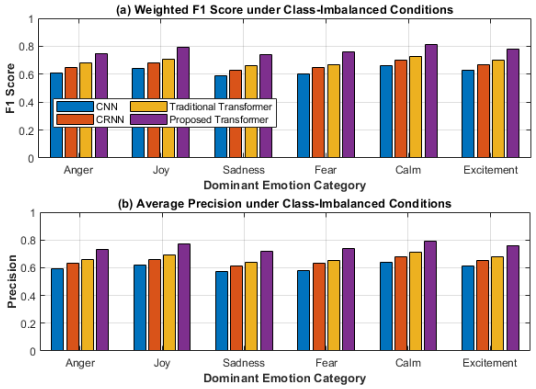


Figure 7: Weighted F1 score and average precision

Figure 7 shows the performance differences between the proposed model and the comparison methods in terms of weighted F1 scores and average precision under test conditions with a severely imbalanced distribution of emotion categories. The six dominant emotions (anger, joy, sadness, fear, calmness, and excitement) each account for 70% of the total, simulating the uneven distribution of emotion annotations in real-world scenarios. The results show that the proposed cross-modal Transformer maintains high recognition consistency across all emotion categories, with significantly less performance fluctuation than the comparison models, achieving weighted F1 scores of 0.74-0.81 and average precisions of 0.72-0.79, demonstrating its greater adaptability to data distribution bias. This advantage stems from the cross-modal attention mechanism's ability to dynamically weight semantic cues. In environments dominated by the majority class, the model leverages the semantic stability of the lyric context to mitigate discriminative bias caused by sampling bias in acoustic features, thereby suppressing overfitting to high-frequency categories. In contrast, unimodal architectures such as CNNs and CRNNs, due to their reliance on local acoustic patterns, are susceptible to interference from dominant class features when minority emotions are present, resulting in reduced precision. While traditional Transformers possess sequence modeling capabilities, they lack a modal complementarity mechanism, making them unstable in categories with sparse semantics or implicit emotional expression (e.g., fear and calmness). Furthermore, the proposed model exhibits minimal performance variation across different emotion categories, indicating that its classification decisions are not systematically biased by the distribution structure of the training data and exhibits good generalization and balance. Overall, these results demonstrate the effectiveness of semantically guided fusion in addressing category imbalance and highlight the profound value of multimodal collaboration in improving model robustness.

4.5 Verification of cross-dataset generalization ability

Using the DEAM dataset as the training domain, the trained model was directly deployed on two external test sets—the sentiment subsets of EmoMusic and GTZAN—without any fine-tuning or adaptation. This zero-shot transfer setup effectively simulates the domain shift issues that arise in real-world applications due to differences in musical style, annotation scale, and recording conditions. During the testing phase, the model's Area Under the Curve (AUC) and Cohen's Kappa coefficient were calculated on the two target datasets. The AUC reflects the classifier's overall discriminative ability under multiple thresholds, while the Cohen's Kappa measures the consistency between the predictions and human annotations, while also correcting for the effects of random agreement.

Table 2: Verification of generalization ability across datasets

Model	Test Domain	AUC	Kappa
CNN	EmoMusic	0.763	0.318
	GTZAN	0.718	0.263
CRNN	EmoMusic	0.674	0.351
	GTZAN	0.612	0.289
Traditional Transformer	EmoMusic	0.698	0.376
	GTZAN	0.635	0.304
Proposed Transformer	EmoMusic	0.832	0.452
	GTZAN	0.887	0.387

4.6 CCC and subjective agreement assessment

To evaluate the reliability of the model's sentiment regression under conditions of low annotated resources, a progressive data sparsity experiment was designed, using 30%, 50%, and 70% of the annotated samples for training, respectively, while maintaining the full test set. In the sentiment dimension prediction task, the Concordance Correlation Coefficient (CCC) was used to measure the statistical consistency between the model output and human annotations. This comprehensive assessment of correlation and bias outperforms the traditional Pearson correlation coefficient. The Inter-annotator Consistency Ratio (ICR) was also introduced as a reference benchmark to quantify the average consistency level within the human annotation community.

Table 3: CCC and subjective consistency evaluation

Model	Label Rate	CCC	ICR
CNN	30%	0.512	0.584
	50%	0.598	0.584
	70%	0.653	0.584
CRNN	30%	0.541	0.584
	50%	0.627	0.584
	70%	0.679	0.584
Traditional Transformer	30%	0.568	0.584
	50%	0.643	0.584
	70%	0.691	0.584
Proposed Transformer	30%	0.635	0.584
	50%	0.712	0.584
	70%	0.758	0.584

5 Discussion

The cross-modal Transformer architecture proposed in this study demonstrates superior performance across multiple challenging tasks. Its core advantage stems from its dynamic attention fusion mechanism. Compared to the dual-branch independent encoding or simple concatenation strategies commonly employed in related work, this approach employs an asymmetric, semantically guided interaction model by using audio features as queries and lyric vectors as keys and values. This design mimics the human process of emotional perception and cognition, where "auditory leads, language assists," enabling the model to proactively retrieve the most relevant semantic fragments based on the current acoustic state, rather than passively and equally fusing modalities. For example, compared to noise-resistant multimodal Transformers, which typically employ a symmetric cross-

attention structure, this approach is computationally complex and may introduce redundant interactions. Our unidirectional guidance mechanism, while maintaining performance, effectively reduces the number of parameters and computational overhead, achieving a better trade-off between noise robustness and efficiency.

In terms of cross-dataset generalization, our approach demonstrates significantly superior stability compared to CNNs, CRNNs, and traditional Transformers. We believe the fundamental reason for this phenomenon lies in the fact that lyric semantics provide a more universal "emotional anchor" that transcends specific acoustic representations. The DEAM, EmoMusic, and GTZAN datasets exhibit significant domain differences in musical style, recording quality, and annotation protocols, resulting in a shift in the distribution of underlying acoustic features. However, the semantic themes carried by lyrics (such as "loneliness" and "joy") are more consistent across domains. Through the dynamic attention mechanism, the model is able to identify acoustic patterns corresponding to these core semantics across different datasets, thereby learning a semantically driven joint representation that is insensitive to domain variations. This explains why our model maintains high AUC and Kappa coefficient in zero-shot transfer scenarios.

Further, the innovation of this architecture lies in its approach to temporal alignment. Unlike graph neural networks that explicitly model temporal alignment, this paper directly incorporates temporal synchronization information into the feature space by injecting absolute positional encodings into the projected word vectors and combining them with frame-level alignment preprocessing. This approach simplifies the model structure and avoids additional graph optimization steps. Heatmap analysis also confirms that it effectively captures the repetitive structure and temporal continuity of lyrics, providing a precise spatiotemporal basis for cross-modal attention. In summary, this paper not only reports performance improvements but, more importantly, reveals the inherent mechanism by which semantic information enhances model robustness and generalization, providing a new perspective for future research in multimodal affective computing.

6 Conclusion

This paper proposes a deep learning architecture that incorporates a cross-modal attention mechanism to address the core challenges of high subjectivity in emotion annotation and insufficient utilization of multimodal information in music emotion recognition. By constructing a Transformer fusion framework using audio features as queries and lyrics semantics as keys, this framework achieves dynamic selective attention of acoustic representations to key language segments, enhancing the fine-grained and context-sensitive semantic alignment between modalities. Experiments demonstrate that the proposed model outperforms traditional CNNs, CRNNs, and Transformers in robustness and generalization under various complex scenarios, including noise interference, class imbalance, cross-dataset transfer, and low-labeled resources. In particular, the proposed method significantly approaches the inter-annotator agreement level in cross-domain transfer and subjective consistency assessment, validating the effectiveness of semantic-guided fusion in modeling emotional subjectivity. This research not only provides a differentiable, end-to-end fusion paradigm for multimodal emotion computing but also reveals the cognitive compensation mechanism of linguistic symbols in auditory emotion decoding, laying a technical foundation for the reliable deployment of intelligent music understanding systems in complex real-world environments. The research results can be widely used in intelligent music recommendation, digital psychological intervention and human-computer empathy interaction systems, providing more accurate emotion perception capabilities in personalized content services and promoting the transformation of emotional computing technology into real-world scenarios.

References

[1] Gómez-Cañón JS, Cano E, Eerola T, et al. Music emotion recognition: Toward new, robust standards in personalized and context-sensitive applications[J]. IEEE Signal Processing Magazine, 2021, 38(6): 106-114. https://doi.org/10.1109/msp.2021.3106232

[2] Gómez-Cañón JS, Gutiérrez-Páez N, Porcaro L, et al. TROMPA-MER: an open dataset for personalized music emotion recognition[J]. Journal of Intelligent Information Systems, 2023, 60(2): 549-570. https://doi.org/10.1007/s10844-022-00746-0

[3] Hizlisoy S, Yildirim S, Tufekci Z. Music emotion recognition using convolutional long short term memory deep neural networks[J]. Engineering Science and Technology, an International Journal, 2021, 24(3): 760-767. https://doi.org/10.1016/j.jestch.2020.10.009

[4] Grekow J. Music emotion recognition using recurrent neural networks and pretrained models[J]. Journal of Intelligent Information Systems, 2021, 57(3): 531-546. https://doi.org/10.1007/s10844-021-00658-5

[5] Assuncao WG, Piccolo LSG, Zaina LA M. Considering emotions and contextual factors in music recommendation: a systematic literature review[J]. Multimedia Tools and Applications, 2022, 81(6): 8367-8407. https://doi.org/10.1007/s11042-022-12110-z

[6] Chaturvedi V, Kaur AB, Varshney V, et al. Music mood and human emotion recognition based on physiological signals: a systematic review[J]. Multimedia Systems, 2022, 28(1): 21-44. https://doi.org/10.1007/s00530-021-00786-6

[7] Garg A, Chaturvedi V, Kaur AB, et al. Machine learning model for mapping of music mood and human emotion based on physiological signals[J]. Multimedia Tools and Applications, 2022, 81(4): 5137-5177. https://doi.org/10.1007/s11042-021-11650-0

[8] Li JW, Barma S, Mak PU, et al. Single-channel selection for EEG-based emotion recognition using brain rhythm sequencing[J]. IEEE journal of biomedical and health informatics, 2022, 26(6): 2493-2503. https://doi.org/10.1109/jbhi.2022.3148109

[9] Athavle M, Mudale D, Shrivastav U, et al. Music recommendation based on face emotion recognition[J]. Journal of Informatics Electrical and Electronics Engineering (JIEEE), 2021, 2(2): 1-11. https://doi.org/10.54060/jieee/002.02.018

[10] Cunningham S, Ridley H, Weinel J, et al. Supervised machine learning for audio emotion recognition: Enhancing film sound design using audio features, regression models and artificial neural networks[J]. Personal and Ubiquitous Computing, 2021, 25(4): 637-650. https://doi.org/10.1007/s00779-020-01389-0

[11] Yin G, Sun S, Yu D, et al. A multimodal framework for large-scale emotion recognition by fusing music and electrodermal activity signals[J]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2022, 18(3): 1-23. https://doi.org/10.1145/3490686

[12] Abdullah SMSA, Ameen SYA, Sadeeq MAM, et al. Multimodal emotion recognition using deep learning[J]. Journal of Applied Science and Technology Trends, 2021, 2(01): 73-79. https://doi.org/10.38094/jastt20291

[13] Kamble K, Sengupta J. A comprehensive survey on emotion recognition based on electroencephalograph (EEG) signals[J]. Multimedia Tools and Applications, 2023, 82(18): 27269-27304. https://doi.org/10.1007/s11042-023-14489-9

[14] Zhao S, Jia G, Yang J, et al. Emotion recognition from multiple modalities: Fundamentals and methodologies[J]. IEEE Signal Processing Magazine, 2021, 38(6): 59-73. https://doi.org/10.1109/msp.2021.3106895

[15] Pandeya YR, Lee J. Deep learning-based late fusion of multimodal information for emotion classification of music video[J]. Multimedia Tools and Applications, 2021, 80(2): 2887-2905. https://doi.org/10.1007/s11042-020-08836-3

[16]Li X, Zhang Y, Tiwari P, et al. EEG based emotion recognition: A tutorial and review[J]. ACM Computing Surveys, 2022, 55(4): 1-57.

[17]Pawar MD, Kokate R D. Convolution neural network based automatic speech emotion recognition using Mel-frequency Cepstrum coefficients[J]. Multimedia Tools and Applications, 2021, 80(10): 15563-15587. https://doi.org/10.1007/s11042-020-10329-2

[18] Zihan D, Alam N, Islam M M. Deep Learning-Driven Music Emotion Recognition: CNN-BiLSTM Networks with Spatial-Temporal Attention[J]. Journal of Platform Technology, 2025, 13(1): 16-30.

[19] Lin Z, Wang Z, Zhu Y, et al. Text sentiment detection and classification based on integrated learning algorithm[J]. Applied Science and Engineering Journal for Advanced Research, 2024, 3(3): 27-33.

[20] Houssein EH, Hammad A, Ali A A. Human emotion recognition from EEG-based brain–computer interface using machine learning: a comprehensive review[J]. Neural Computing and Applications, 2022, 34(15): 12527-12557. https://doi.org/10.1007/s00521-022-07292-4

[21] Jingjing W, Ru H. Music emotion recognition based on the broad and deep learning network[J]. Journal of East China University of Science and Technology, 2022, 48(3): 373-380.

[22] Bakariya B, Singh A, Singh H, et al. Facial emotion recognition and music recommendation system using CNN-based deep learning techniques[J]. Evolving Systems, 2024, 15(2): 641-658. https://doi.org/10.1007/s12530-023-09506-z

[23] Yadav SP, Zaidi S, Mishra A, et al. Survey on machine learning in speech emotion recognition and vision systems using a recurrent neural network (RNN)[J]. Archives of Computational Methods in Engineering, 2022, 29(3): 1753-1770. https://doi.org/10.1007/s11831-021-09647-x

[24] Topic A, Russo M. Emotion recognition based on EEG feature maps through deep learning network[J]. Engineering Science and Technology, an International Journal, 2021, 24(6): 1442-1454. https://doi.org/10.1016/j.jestch.2021.03.012

[25] Viñán-Ludeña MS, de Campos L M. Discovering a tourism destination with social media data: BERT-based sentiment analysis[J]. Journal of Hospitality and Tourism Technology, 2022, 13(5): 907-921. https://doi.org/10.1108/jhtt-09-2021-0259

[26] Huang F, Li X, Yuan C, et al. Attention-emotion-enhanced convolutional LSTM for sentiment analysis[J]. IEEE transactions on neural networks and learning systems, 2021, 33(9): 4332-4345. https://doi.org/10.1109/tnnls.2021.3056664

[27] Zhang Y, Zhang Y, Wang S. An attention-based hybrid deep learning model for EEG emotion recognition[J]. Signal, image and video processing, 2023, 17(5): 2305-2313. https://doi.org/10.1007/s11760-022-02447-1

[28] Yi Y, Tian Y, He C, et al. DBT: multimodal emotion recognition based on dual-branch transformer: Y. Yi et al[J]. The Journal of Supercomputing, 2023, 79(8): 8611-8633. https://doi.org/10.1007/s11227-022-05001-5

[29] Wu Y, Daoudi M, Amad A. Transformer-based self-supervised multimodal representation learning for wearable emotion recognition[J]. IEEE Transactions on Affective Computing, 2023, 15(1): 157-172. https://doi.org/10.1109/taffc.2023.3263907

[30] Arumugam L, Arumugam S, Chidambaram P, et al. A multi-model deep learning approach for human emotion recognition[J]. Cognitive Neurodynamics, 2025, 19(1): 123. https://doi.org/10.1007/s11571-025-10304-3

[31] Khan M, Gueaieb W, El Saddik A, et al. MSER: Multimodal speech emotion recognition using cross-attention with deep fusion[J]. Expert Systems with Applications, 2024, 245: 122946. https://doi.org/10.1016/j.eswa.2023.122946

[32] Deng Z, Lu Y, Liao J, et al. Sync-TVA: A Graph-Attention Framework for Multimodal Emotion Recognition with Cross-Modal Fusion[J]. arXiv preprint arXiv:2507.21395, 2025.

[33] Liu Y, Zhang H, Zhan Y, et al. Noise-resistant multimodal transformer for emotion recognition[J]. International Journal of Computer Vision, 2024: 1-21.

[34] Wafa A A, Eldefrawi M M, Farhan M S. Advancing multimodal emotion recognition in big data through prompt engineering and deep adaptive learning[J]. Journal of Big Data, 2025, 12(1): 1-62. https://doi.org/10.1186/s40537-025-01264-w

[35] Savchenko A, Savchenko L. Leveraging Lightweight Facial Models and Textual Modality in Audio-visual Emotional Understanding in-the-Wild[C]//Proceedings of the Computer Vision and Pattern Recognition Conference. 2025: 5778-5788. https://doi.org/10.1109/cvprw67362.2025.00577

[36] Goel A, Singh H, Singh A. Emotion-Aware Speech Translation: A Review[C]//2025 International Conference on Intelligent Control, Computing and Communications (IC3). IEEE, 2025: 533-538. https://doi.org/10.1109/ic363308.2025.10957552

A Robust Watermark-based Model for Image Content Integrity Verification and Tampering Detection

Jayanti Rout, Sangram Pati, Minati Mishra*
P. G. Department of Computer Science, Fakir Mohan University, Balasore, India
E-mail: minatiminu@gmail.com
*Corresponding author

Keywords: Tamper detection, watermarking, wavelet transform, DCT, SVD, splicing, cloning

Received: May 9, 2025

The rise of digital image tampering has increased the inability to check authenticity of images. This has made its identification as well as the localization the tampered regions a significant challenge. Watermark-based content authentication methods provides an effective solution to it. This study proposes a strong and reliable watermark-based authentication algorithm. It uses two watermarks that are created using the Haar wavelet transform, discrete cosine transform, and singular value decomposition. A tamper detection mask is also used to highlight the tampered regions of the image. The embedded watermarks were observed to resist various attacks, including compression, cropping, noise addition, and blurring. This highlights the robustness of the approach. The experimental evaluations show that the proposed method achieves high imperceptibility with various near-ideal metrics. It achieved Peak Signal-to-Noise Ratio (PSNR) values > 38 dB, Structural Similarity Index Measure (SSIM) $> 99\%$, and Normalized Cross-Correlation (NCC) of $\sim 99.8\%$. The method effectively detects various types of manipulation and attacks in images. The proposed technique shows strong applicability in various domains, including digital forensics, copyright enforcement, and the protection of sensitive multimedia contents.

Povzetek: Študija predlaga robusten in zanesljiv algoritem za preverjanje pristnosti na osnovi vodnega žiga. Metoda uporablja dva vodna žiga, ustvarjena s pomočjo Haarove valovne transformacije, diskretne kosinusne transformacije in metode SVD.

1 Introduction

The use of digital media has increased significantly in recent years. They are considered a suitable choice to easily share information. Among them, images are an impactful source of information. According to a survey, more than 2.1 trillion images are taken annually for content creation [1]. However, easy access to image editing tools poses a threat to the authenticity and integrity of digital content [2]. The reliability of media is very important, especially in the domain of journalism, publishing, law enforcement, medical diagnostics, etc. For example, in the medical field, doctors use medical images for accurate diagnosis and treatment planning. On the other hand, digital images and videos are frequently submitted as evidence in legal settings. In such scenarios, the authenticity of digital content is directly related to important decision-making. As a result, detecting manipulation or tampering of these digital media has become an important research challenge [3, 4].

The wide use of advanced digital cameras and user-friendly editing software provides a seamless editing experience. It provides flexibility for even non-expert users for alteration. In addition, these edits are done using such sophisticated methods that usually leave no noticeable visual artifacts. It can mislead viewers and can rapidly spread misinformation. This poses serious social, political, and economic risks [5]. An example of such image manipulation in 2017 is presented in Figure 1. A senior police officer is presented as kneeling before a politician. The investigation revealed that the image had been manipulated by splicing a frame from a 2011 Bollywood film into an authentic photograph [6]. Such incidents highlight the need for reliable image tampering detection techniques. A brief overview of historical cases of image tampering is presented in [7]. Digital image manipulation falls into four main categories



Figure 1: Illustration of image manipulation [6]: left—tampered image; right—original scene.

based on the type of alteration: copy-move or cloning, splicing, retouching, and resampling [8]. In copy-move tampering, a region of an image is duplicated and pasted elsewhere within the same image. They are usually used

to hide or replicate an object. Splicing involves merging segments from different images to create a composite that misrepresents reality. Retouching focuses on improving or modifying the image to change its visual impact without changing the image geometrically. Resampling applies geometric operations such as scaling, rotation, affine transformations, etc., to align tampered regions with the surrounding context. An illustration of all these manipulations is shown in Figure 2. These techniques pose unique challenges to detection algorithms. This presents the need to develop specialized approaches for detection.



(a) (b) (c) (d)

Figure 2: Row 1: the forged images: (a) cloned, (b) spliced, (c) retouched, and (d) resampled. Row 2: the original images

Tamper detection techniques are categorized into two broad types: active and passive methods [9]. Active methods involve the integration of additional information into the media during its creation or transmission. This information consists of watermarks, digital signatures, etc. These features act as integrity verifiers that can be extracted or verified to detect unauthorized alterations. Watermarking techniques can be divided into fragile and robust types. Fragile watermarks are highly sensitive to changes. They can be easily warped by even minor modifications. Although this makes them effective in detecting tampering, they are not built to handle intentional attacks. However, robust watermarking is designed to resist various image processing operations such as compression, filtering, geometric transformations, and noise addition [10]. As discussed in [11], an effective and robust watermarking scheme must meet the criteria of security, imperceptibility, and robustness, collectively known as SIR. Similarly to a watermark, digital signatures are important in active authentication. A digital signature is a cryptographic technique that binds a unique identifier to the content. This allows us to verify the origin and integrity of the image. Unlike watermarking, it does not alter the content of the media itself. This makes them suitable for scenarios where maintaining the original quality of the image is important [12]. Thus, both watermarking and digital signatures play complementary roles in active tamper detection systems.

Passive or blind tampering detection techniques work without embedded information or prior access to the original image [13]. These methods study the visual features of the image and the statistical irregularities introduced within them during manipulation. It assumes that every tampering introduces some detectable traces. These artifacts may be present in the form of irregularities in noise patterns, color distributions, compression artifacts, etc., [14]. Various statistical, machine learning, and signal processing techniques have been proposed to capture and analyze these variations. This analysis provides effective tamper localization and classification [15, 16].

Motivated by the need for an efficient image authentication system, the proposed work presents an improvement over existing dual or hybrid watermarking schemes. It proposes a strategically integrated three-level transform approach: the Discrete Cosine Transform (DCT), the Singular Value Decomposition (SVD), and the Haar wavelet transform. They are combined with a dual watermark structure that consists of a signature and a fingerprint. Unlike traditional methods that focus solely on the verification or authentication of the ownership, this method inserts the signature into the fingerprint. The resultant is then embedded in the host image. This forms a hierarchical and layered authentication mechanism. In addition, a novel tamper detection method is introduced. It uses the embedding of a tamper detection mask into the HL1 and LH2 sub-bands of the Most Significant Bit (MSB) plane using a two-level DWT. This allows authenticity to be checked and the exact manipulated regions to be located. This part was observed to be overlooked in existing work. Integration is not just a fusion of existing techniques but a carefully designed extraction process. It ensures robustness, imperceptibility, and active tamper location. It provides a comprehensive and novel solution for the authentication and protection of image content.

To evaluate the effectiveness of the approach, the following research questions (RQ) are designed to highlight the key contributions:

- RQ1: How does the proposed approach improve the robustness and efficiency of watermark-based content authentication and tampering detection compared to individual techniques?
- RQ2: To what extent does the proposed model maintain high performance on the basis of the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and Normalized Cross-Correlation (NCC) when introduced to various attacks?
- RQ3: How accurately can the proposed method detect and locate tampered regions, especially in complex scenarios?

The remaining sections of the paper are organized as follows: Section 2 gives a brief overview of existing technological advancements in the domain. Sections 3 and 4 present an overview of the technologies used and explain the proposed methodology, respectively. Section 5 discusses the experimental setup and analyzes the results ob-

tained. Section 6 concludes the paper with suggestions for future research directions.

2 Literature review

Digital watermarking is the process of inserting unique identifiers into host signals. They are mainly used to verify authenticity, ownership, or both. They have evolved significantly since their prominence in the 1990s based on methodologies and applications [17]. Watermarking is used for various purposes, including copyright protection, prevention of unauthorized duplication, assurance of digital content integrity, etc., [18]. They can be divided into two main groups based on their primary objective: authentication models and tampering detection models.

2.1 Authentication models

The healthcare sector relies on invisible watermarking due to its need for secure, durable, and efficient data protection techniques. A discrete watermarking model for the preservation of copyright in medical data is described in [19]. Continuing in this direction to safeguard medical images, the authors in [20] used DWT and SVD-based methods for invisible watermarking. Poonam and Arora [21] used DWT and SVD-based watermarking techniques. Zermi et al. [22] introduced a blind method for medical image authentication that integrates an electronic patient record (EPR) hash with image acquisition data. This approach ensures data integrity by embedding the EPR hash in medical images, making use of DWT and SVD.

The researchers in [23] suggested a hybrid digital watermarking model for Digital Imaging and Communications in Medicine (DICOM) that combines visible-imperceptible and robust-imperceptible watermarking approaches. This method preserves the integrity of EPR data while authenticating the source. The technique is applied to DICOM images with bit depths greater than 8 bpp. The approach was tested against twenty different geometric and signal processing operations and was found to be robust across all scenarios tested. The researchers in [24] proposed a watermarking model using DCT-DWT and spatial SVD for watermark embedding. It achieved a PSNR of 40 dB, an NCC of 0.9, and an SSIM value of nearly 0.99. The method is found to withstand noise attacks, filtering, geometric deformations, and compression.

Rajput et al. suggested a watermark model combining DWT, Haar Discrete (HD), and SVD that outperformed other existing models in terms of high resistance to attacks and better quality [25]. Hu et al. proposed a watermarking model that resists instruction-driven image editing and withstands instruction-driven image editing (IDIE) [26]. A Partial Instruction-Driven De-noising Sampling Guidance (PIDSG) module with the encoder-decoder training framework was integrated. It improves the strength of the watermark against IDIE and the semantic distortions introduced by IDIE. The researchers have experimentally proved that the watermark offers robustness while maintaining high visual quality and editability.

Neekhara et al. proposed a multipurpose deep learning (DL)-based watermarking technique that can embed a secret text in image pixels [27]. The researchers created an encoder-decoder-based training framework that recovers secrets if the found watermark was found to be untampered with. U-Net CNN architecture is used for the encoder and decoder network that uses 256×256 input images. In addition, a discriminator network is used to distinguish real and tampered images. Its robustness also extends to images and videos.

Nadimpalli and Rattani proposed a facial image watermarking technique for social networks that is impenetrable against adversarial watermark removal attacks [28]. The proposed U-Net CNN-based architecture consisted of five components: encoder, decoder, adversary network, discriminator, and critic network. FaceForensics++, CelebA, and the IMDB-Wiki dataset were used to train and evaluate the model.

The researchers in [29] proposed a wavelet transform-based reversible watermarking algorithm for the authentication of image content. A focus on avoiding tampering's influence on the wavelet domain was considered. Each block of the image (four pixels in each block) is transformed as a carrier signal for watermark information. Two different watermarks are used, encrypted, and scrambled using chaotic mapping with different keys. It increases security and embedding randomness. Different detection and screening levels improve accuracy. Both watermarks reconstruct category and feature information when restoring tampered areas.

Similarly, the researchers in [30] proposed a dual-watermarking scheme using DWT-SVD and two authentication layers. The watermarking scheme uses a non-pixelpping 8x8 pixel block with DWT transform on a 512x512 host image. This generates an authentication bit for tamper localization. The approach was tested and evaluated on the SIPI dataset.

The researchers in [31] introduced a bioinspired watermarking algorithm for retinal fundus images in computer-aided retinopathy diagnosis. It uses the Steered Hermite Transform (SHT) and SVD to create imperceptible watermarks, embed them into RGB fundus images, and encrypt them using the Jigsaw Transform for added security. The MESSIDOR-2 dataset was used to train and evaluate the model. In addition, four DL models were evaluated to detect the application of watermarks in the images. However, the proposed watermarking approach was resilient and remains undetected against DL models.

2.2 Tampering detection models

Hu et al. proposed a tampering identification model using watermarking and alpha mattes to detect forged images [32]. The proposed model calculated the alpha matte of the image using component hue-difference-based spectral mat-

ting and used a DWT-DCT-SVD-based watermarking technique to insert the watermarks. The detection method uses the difference in singular values to identify manipulated foreground and background photos by inserting two distinct watermarks based on derived alpha mattes. The technique reliably identifies forgeries produced by image matting or image in-painting, tampered foreground and background photos, and traded foreground and background images. An adaptive threshold was used that makes it suitable for real-world scenarios. Nguyen proposed a fragile model of image authentication that combines DCT, DWT, and SVD, and DCT for feature selection and quantization index modulation (QIM) for the generation and embedding of the authentication code [33]. The coefficients are calibrated using the Gram-Schmidt process to ensure the expected retrieval of the authentication code. The model is found to produce watermarked photos with great visual quality and excellent tamper detection accuracy.

The researchers in [34] proposed a visual cryptography-based watermarking scheme to detect forgeries in images. The approach was tested against various attacks such as noise, sharpening, median filtering, lossy compression, and geometric distortions. The authentication and forgery detection model proposed in [35] integrates vector quantization (VQ), absolute moment block truncation coding (AMBTC), and double matrix encoding to reduce distortions caused to the host due to watermarking. The authors used images from the BOWS dataset to conduct their experiments. The fragile and reversible watermarking model proposed in [36] generates two watermarked versions of the image. It provides complete recovery of the original image when no tampering is present. It ensures high-quality restoration if tampering is detected.

The researchers in [37] proposed a watermarking model to detect image manipulation using adaptive matrices and overcome the limitations of various previous methods. The embedding process uses a two-stage authentication mechanism with a 16-bit validation scheme to ensure precise forgery localization. This adaptive approach is designed to adapt the matrix pattern to the characteristics of the image. At the same time, the utilization of logistic sequencing enables the generation of non-periodic and non-convergent patterns. It improves authentication efficiency and accuracy. The approach was tested using the USC-SIPI dataset. An average PSNR value of 55.90 dB and an SSIM greater than 0.99 were observed.

The researchers in [38] proposed a watermarking technique using a complex wavelet transform of quaternion two trees for the extraction of features from images. The maximal entropy random walk algorithm identifies the best embedding area within the low-frequency subband. To detect and localize tampering, a watermark is created using the Swin transformer model and embedded in the chosen blocks. In addition, a dual scrambled image is encoded using SVD-derived principal component coefficients to authenticate the watermarked image before extraction. The semi-blind extraction method verifies authenticity by comparing the retrieved scrambled watermark with the original regenerated watermark. The approach was evaluated using three datasets, including HDR images, CASIA, and other datasets.

The researchers in [39] proposed an approach to image tampering and localization using the integer wavelet transform (IWT). IWT coefficients were used to generate face recovery information and embed it into cover-face images at the sender's side. Three algorithms are proposed, with IWT showing superior performance. The algorithm efficiently identifies altered blocks and restores unaltered versions to ensure originality, integrity, and security. The approach was tested on various images that differ in dimension and size.

A detailed review of existing studies shows that various traditional watermarking approaches use single-domain techniques, such as standalone DWT, DCT, or SVD. It leads to a failure to balance robustness, imperceptibility, and tampering localization. They usually rely on DWT. As a result, they are vulnerable to frequency-based attacks, such as filtering, compression, or noise addition. This occurs because of the limited stability of wavelet coefficients under such distortions. Similarly, even if DCT-based schemes benefit from energy compaction, they suffer from reduced spatial localization capabilities. Although SVD-based approaches are efficient in maintaining imperceptibility, they lack spatial sensitivity to detect local tampering effectively.

In addition, only a few existing frameworks focus mainly on watermark retrieval without offering a reliable mechanism to locate tampered regions. In particular, limited attention has been given to the idea of embedding auxiliary tamper detection structures within transform domains. They can include spatial markers or canvases that improve the visual identification of manipulations. The proposed approach addresses these issues effectively by proposing a hybrid watermarking-cum-tampering localization model.

3 Background materials

This section highlights the mathematical details of the transforms used: the Discrete Wavelet Transform (DWT), the DCT, and the SVD. They play an important role in various image processing tasks due to their ability to extract, analyze, and manipulate features in the transform domain.

3.1 Discrete wavelet transform

It has the ability to break up an image into its frequency components. It provides analysis at multiple resolutions. Unlike the traditional Fourier Transform, which only captures frequency content, DWT provides both spatial and frequency localization, making it particularly suitable for hierarchical image analysis. Among various wavelet families, the Haar wavelet is often preferred due to its simplicity and computational efficiency.

For two-dimensional (2D) images, the DWT applies a set of low-pass (L) and high-pass (H) filters along the rows and columns of the image, followed by downsampling. This operation results in four sub-band images, each representing different frequency and directional information.

Figure 3 shows the level-1 DWT decomposition of the standard 'cameraman's image using the Haar wavelet. The right image is the original, while the left image displays the four decomposed sub-bands: (a) LL, (b) LH, (c) HL, and (d) HH. The basic operations of the 2D Haar-DWT, specifically, horizontal and vertical filtering, are illustrated in Figure 4.



Figure 3: Level-1 DWT decomposition using Haar wavelet

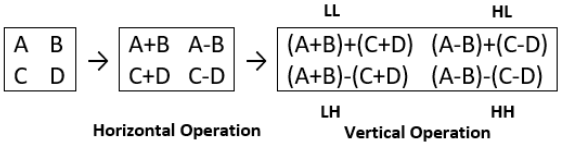


Figure 4: Level-1 Haar-DWT: Horizontal and vertical operations on a 2×2 image.

In watermarking applications, embedding data in the LL sub-band enhances robustness but can significantly degrade visual quality. Conversely, embedding in the HH sub-band is less perceptible but more vulnerable to attacks such as compression or filtering. Therefore, most DWT-based watermarking techniques favor the LH and HL sub-bands, striking a balance between imperceptibility and robustness. DWT's ability to provide secure, resilient, and imperceptible embedding makes it ideal for applications in copyright protection, image authentication, and tamper detection.

3.2 Discrete cosine transform

It is widely used in image compression and watermarking because of its energy-compaction property. It transforms spatial-domain image data into frequency-domain coefficients, making it easier to manipulate significant features while removing redundant information.

DCT segments an image into non-overlapping blocks (NOB) of size $B \times B$, where B depends on the image resolution and processing requirements. For each block, the forward DCT coefficients are computed using Equation (1).

$$FDCT(\gamma,\zeta) = \sum_{\alpha=0}^{B-1}\sum_{\beta=0}^{B-1} NOB(\alpha,\beta)\cdot Ker(\alpha,\beta,\gamma,\zeta) \tag{1}$$

Here, $NOB(\alpha,\beta)$ represents the pixel value at position (α,β), and the kernel function $Ker(\alpha,\beta,\gamma,\zeta)$ is evaluated using Equation (2).

$$Ker(\alpha,\beta,\gamma,\zeta) = \omega(\gamma)\cdot\omega(\zeta)\cos\left(\frac{(2\alpha+1)\gamma\pi}{2B}\right) \cdot\cos\left(\frac{(2\beta+1)\zeta\pi}{2B}\right) \tag{2}$$

The scaling factor $\omega(z)$ is calculated using Equation (3)

$$\omega(z) = \begin{cases} \sqrt{\frac{1}{B}} & \text{if } z = 0 \\ \sqrt{\frac{2}{B}} & \text{if } z > 0 \end{cases} \tag{3}$$

To reconstruct the image block from its frequency-domain coefficients, the inverse DCT (IDCT) is applied using Equation (4).

$$RDCT(\alpha,\beta) = \sum_{\gamma=0}^{B-1}\sum_{\zeta=0}^{B-1} FDCT(\gamma,\zeta)\cdot Ker(\alpha,\beta,\gamma,\zeta) \tag{4}$$

Due to its ability to preserve important visual details while minimizing perceptible distortions, DCT is an effective tool for embedding watermarks in mid-frequency coefficients, where changes are less noticeable yet are resistant to compression attacks.

3.3 Singular value decomposition

It is a linear algebra technique that is used in image processing for tasks such as compression, noise reduction, and watermarking. It decomposes a matrix A of size $m \times n$ into three matrices: U, S, and V^T as defined in Equation (5).

$$A_{m\times n} \rightarrow U_{m\times m} S_{m\times n} V^T_{n\times n} \tag{5}$$

Here, U is an orthogonal matrix of size $m \times m$ that contains the left singular vectors, S is an diagonal matrix of size $m \times n$ containing singular values that represent the strength of each corresponding vector, and V^T is the transpose of an orthogonal matrix of size $n \times n$ that contains the right singular vectors.

SVD identifies the directions in which the data are most variable and separates the signal from the noise. The original matrix A can be reconstructed from its decomposed components using the inverse operation using Equation (6).

$$A_{m\times n} \leftarrow U_{m\times m} S_{m\times n} V^T_{n\times n} \tag{6}$$

In watermarking, SVD is particularly useful because slight changes in singular values do not significantly affect image quality. Embedding watermark data in singular values ensures SIR, making SVD a preferred technique in high-performance watermarking systems.

4 Proposed framework

The proposed pipeline for digital image content authentication and tampering detection framework consists of the following phases:

1. Embedding of dual watermarks and the tamper detection mask into the cover image.
2. Tampering and Attack Simulations.
3. Retrieval of embedded watermarks to verify the integrity and authenticity of the image.
4. Extraction of the tamper detection mask to detect and locate tampered regions.

Each of the phases is designed to systematically address the requirements of secure watermarking and tampering detection, ensuring the integrity and authenticity of the cover image.

4.1 Watermark embedding phase

As presented in Figure 5, the embedding process begins by separately processing two components of the watermark: the signature (W_s) and the fingerprint (W_f). Each watermark undergoes a single-level DWT, yielding four sub-bands: LL, LH, HL, and HH. The DWT of the signature is represented using Equation (7).

$$W_s \xrightarrow{\text{DWT}} \{LL_s, LH_s, HL_s, HH_s\} \tag{7}$$

The HH_s sub-band is selected for further processing due to its high-frequency detail preservation and imperceptibility. A DCT is applied to HH_s using Equation (8).

$$HH_s \xrightarrow{\text{DCT}} DCT(HH_s) \tag{8}$$

Next, SVD is performed on the transformed sub-band and computed using Equation (9).

$$DCT(HH_s) = U_s S_s V_s^T \tag{9}$$

The same operations are applied to the fingerprint (W_f) and the cover image (I) to obtain their respective singular values using Equation (11).

$$W_f \xrightarrow{\text{DWT}\to\text{DCT}} DCT(HH_f) = U_f S_f V_f^T \tag{10}$$

$$I \xrightarrow{\text{DWT}\to\text{DCT}} DCT(HH_I) = U_I S_I V_I^T \tag{11}$$

Next, the singular values of the signature are embedded into those of the fingerprint using Equation (12).

$$S'_f = S_f + \alpha S_s \tag{12}$$

The transformed singular values of the watermarks are embedded into the singular value (S_I) of the cover image using Equation (13).

$$S'_I = S_I + \beta S'_f \tag{13}$$

Here, α and β are scaling factors controlling the strength of embedding. The inverse SVD, DCT, and DWT operations are then applied sequentially to obtain the partially watermarked image (I_{pw}) using Equation (14).

$$I_{pw} = \text{IDWT}\left(\text{IDCT}\left(U_I S'_I V_I^T\right)\right) \tag{14}$$

To enhance tamper localization, a tamper detection mask (T) is embedded in the LH2 of HL1 sub-bands of the MSB plane using a two-level DWT decomposition using Equations (15) and (16).

$$I_{pw} \xrightarrow{\text{2-level DWT}} \{LL1, LH1, \{LL2, LH2, HL2, HH2\}, HH1\} \tag{15}$$

$$HL1'LH2' = HL1LH2 + T \tag{16}$$

Applying inverse DWT yields the final watermarked image I_w using Equation (17).

$$I_w = \text{IDWT}(\{LL1, LH1, \{LL2, LH2, HL1', LH2', HH2\}, HH1\}) \tag{17}$$

This ensures a robust and imperceptible watermark with improved tamper localization.

4.2 Tampering and attack simulations

In this phase, the watermarked image is subjected to various types of tamperings such as cloning, splicing, retouching, and resampling, followed by common signal processing attacks such as cropping, compression, noise addition, blurring, and geometric distortions. This step plays a crucial role in evaluating the reliability and robustness of the proposed model under both malicious and nonmalicious modifications/alterations.

4.3 Watermark retrieval phase

The watermark retrieval process as shown in Figure 7 begins by applying DWT and DCT to the HH sub-band of the received watermarked image (I_w), followed by SVD using Equation (18).

$$I_w \xrightarrow{\text{DWT}\to\text{DCT}\to\text{SVD}} DCT(HH_w) = U'_I S'_I V'^T_I \tag{18}$$

The transformed watermark is recovered by subtracting the original singular values of the cover image using Equation (19).

$$S_s = \frac{S'_f - S_f}{\alpha} \tag{19}$$

Where,

$$S'_f = \frac{S'_I - S_I}{\beta}$$

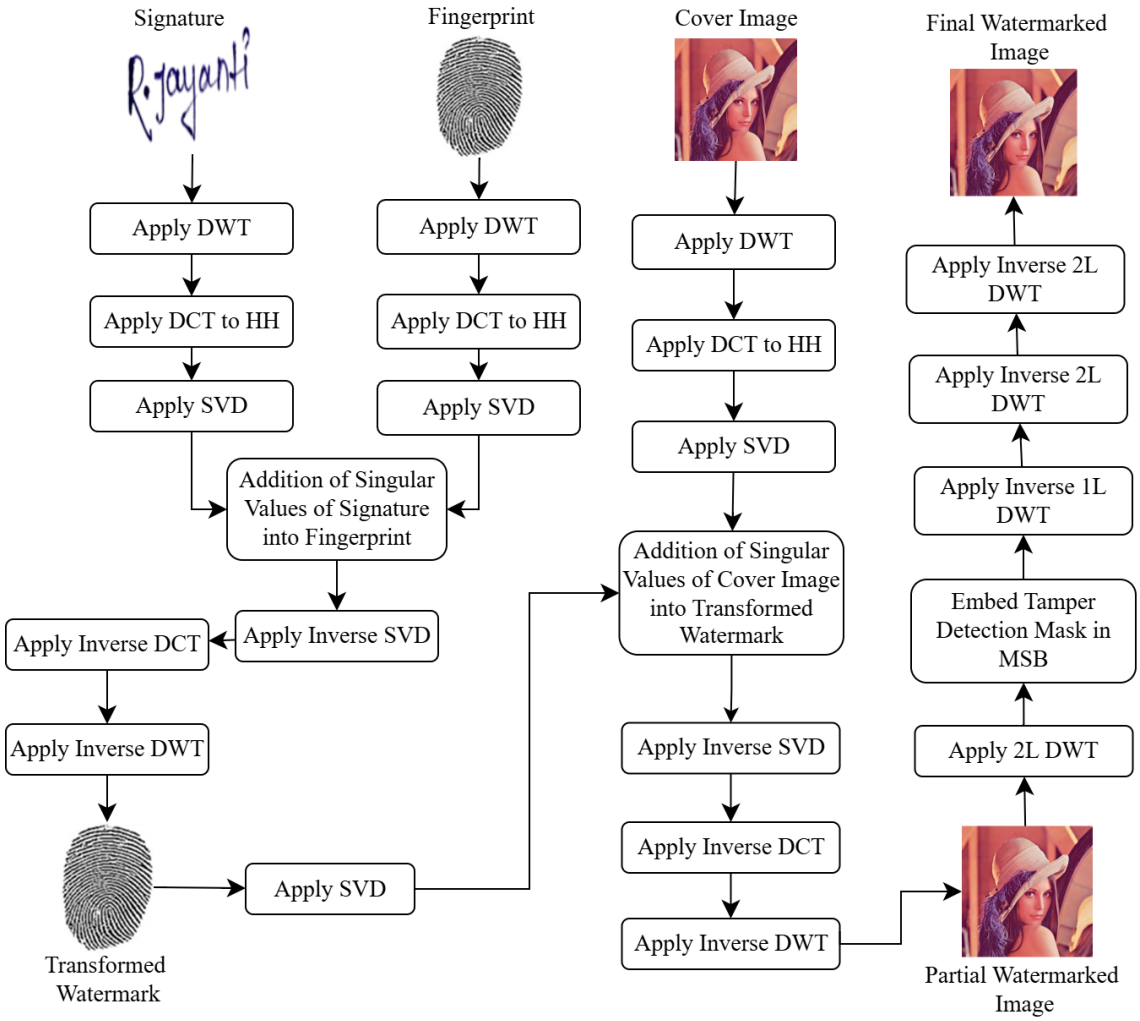


Figure 5: Proposed watermark embedding model.

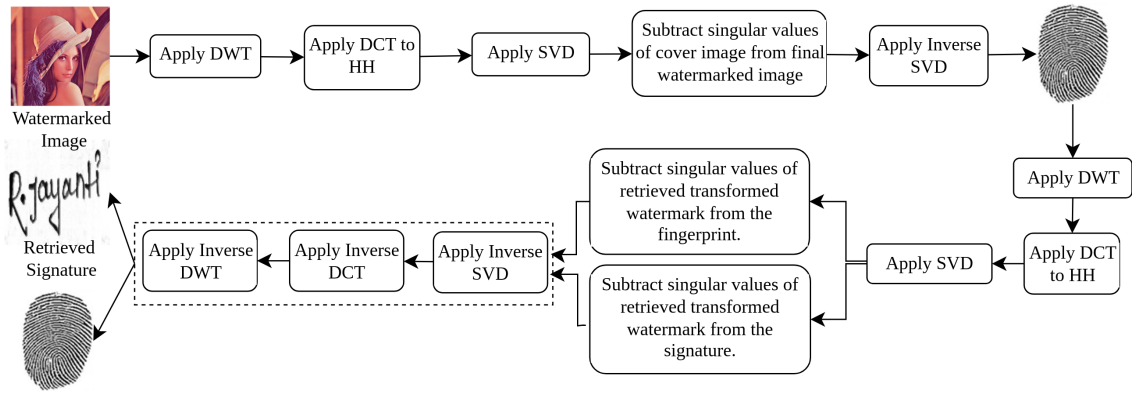


Figure 6: Flow of the watermark extraction procedure

The inverse SVD, DCT, and DWT steps reconstruct the fingerprint and signature in the spatial domain are computed using Equations (20) and (21).

$$W'_f = \text{IDWT}(\text{IDCT}(U_f S'_f V_f^T)) \tag{20}$$

$$W'_s = \text{IDWT}(\text{IDCT}(U_s S_s V_s^T)) \tag{21}$$

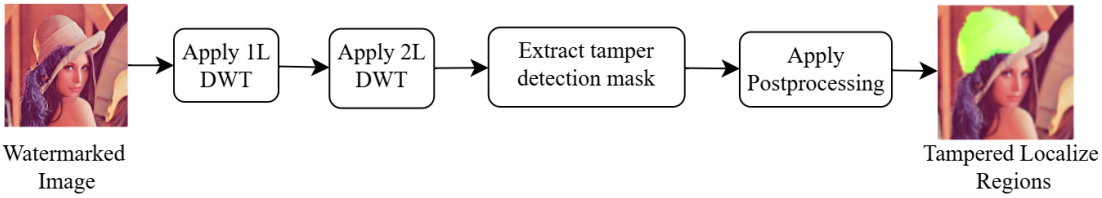


Figure 7: Overall steps of the proposed tampering detection model

4.4 Tampering detection phase

As shown in Figure 7, tampering detection starts with applying a two-level DWT to the final watermarked image using Equation (22).

$$I_w \xrightarrow{\text{2-level DWT}} \{LL1, LH1, \{LL2, LH2, HL2, HH2\}, HH1\} \quad (22)$$

The tamper detection mask T' is extracted from the LH2 of HL1 sub-bands of the MSB using Equation (23).

$$T' = \text{Extract}(LH2) \quad (23)$$

To detect manipulated regions, the difference between the original and extracted tamper mask is evaluated using Equation (24).

$$\Delta T = |T - T'| \quad (24)$$

Any significant non-zero values in ΔT are flagged as tampered pixels. It provides accurate localization of altered regions in the image. Further morphological operations, i.e., opening and closing, are applied to the extracted tamper mask to improve the accuracy of the localization.

Together, these three phases ensure secure and layered watermarking. It supports robust authentication and provides fine-grained localization of tampered regions.

5 Analysis and discussion of the findings

This section presents a comprehensive evaluation of the proposed method through a detailed experimental analysis. Subsection 5.1 describes the experimental setup. Subsection 5.2 describes the details of the dataset. Section 5.3 describes various evaluation metrics used to assess the robustness and reliability of the approach. Subsection 5.4 presents an in-depth discussion of both qualitative and quantitative findings from the results.

5.1 Experimental setup

The performance of the suggested approach is evaluated through various experiments. They were conducted on a 64-bit Windows 11 Pro system (version 23H2) using an Intel Core i7-10700 CPU running at 2.90 GHz with 8 GB of RAM.

In the experimental phase, two distinct watermarks—a signature watermark and a fingerprint watermark—were embedded in digital images along with a tamper detection mask to evaluate the proposed authentication framework. Signature and fingerprint watermarks serve the purpose of verifying the authenticity and ownership of the image content. The tamper detection mask helps to localize the exact regions where manipulations have occurred. These components were embedded using a combination of DWT, DCT, and SVD. It ensures robustness against common image processing attacks. During the verification stage, both watermarks were extracted and matched to the original references to confirm the authenticity. The tamper detection mask can detect any mismatch to pinpoint the manipulated areas.

The following techniques were used during the experiments to evaluate the efficacy of the watermark and tamper detection model:

- Image manipulation operations: Retouching, resampling, splicing, and cloning.
- Noise attacks: Gaussian, salt&pepper, and speckle.
- Image enhancement procedures: Filtering, sharpening, histogram equalization, blurring, and lossy compression.

To quantify the performance, various metrics including, PSNR, SSIM, NCC, and detection accuracy, were used. The results show the efficacy of the proposed approach in both authenticating the image and locating tampered regions.

5.2 Dataset details

The USC-SIPI dataset [40] is used for experimentation. It is organized into volumes according to the image size. It includes grayscale images at 8 bits per pixel (bpp) and color images at 24 bpp. For simulation of experiments, six commonly used images were selected from the dataset: splash, baboon, Lena, sailboat, pepper, and house. As shown in Figure 8, all selected images are 512×512 pixels in size and are 24-bit color images. The watermarks used in the experiments, including the signature and fingerprint, were provided by the authors. The signature, fingerprint, and

tamper detection mask are monochrome images of 64×64 bits, 128×128 bits, and 32×32 bits, respectively.



Figure 8: Illustration of test images from the SIPI image database

5.3 Performance metrics

The effectiveness of the proposed model is evaluated using a set of well-established performance metrics. To assess the robustness of the image watermarking process, metrics such as PSNR, SSIM, Mean Squared Error (MSE), and NCC are employed. To evaluate the accuracy of tampering or forgery detection, various classification metrics are used. These indicators provide a comprehensive understanding of both the preservation of visual quality and the detection capability of the proposed framework.

5.4 Result analysis

Table 1 presents the quantitative evaluation of the imperceptibility and robustness of the watermark using three widely adopted image quality metrics: PSNR, NCC, and SSIM. These metrics were computed for six standard test images after embedding of the watermark. The PSNR values for all watermarked images range from 38.926dB to 40.322dB. This indicates a high visual quality with minimal perceptible distortion. The NCC values, all exceeding 99.8%, confirm a strong correlation between the original and watermarked images. This signifies excellent robustness and successful preservation of the watermark. Similarly, the SSIM values, ranging from 99.056% to 99.713%, demonstrate that the structural content and perceptual quality of the images remain intact after embedding. These results validate that the proposed scheme maintains a high level of imperceptibility and fidelity across diverse image content. It makes the scheme suitable for secure image authentication applications.

To evaluate the robustness of the proposed watermarking method, various common image processing attacks were applied to the watermarked Lena image. Figure 9 demonstrates the visual impact of these attacks. The applied distortions include median filtering, Gaussian noise, salt&pepper noise, speckle noise, JPEG and JPEG2000 compression, sharpening, histogram equalization, averaging, motion blur, and Gaussian low-pass filtering. These attacks simulate real-world distortions and compression scenarios commonly encountered in digital image transmission and storage. Figure 10 and Figure 11 show the signatures and fingerprints extracted from the attacked images. Despite various distortions, the watermarks remain visually identifiable, demonstrating the model's resilience in preserving embedded watermark information. Although some degradation is visible under severe attacks, such as Gaussian noise and quantization, the embedded signature remains recognizable. This validates the robustness of the hybrid watermarking scheme. The qualitative analysis confirms that the proposed DWT-DCT-SVD-based embedding strategy effectively preserves watermark integrity under typical attack scenarios. This supports the robustness and applicability of the model in real-world multimedia authentication tasks.

Figure 12 presents the quantitative analysis of the proposed watermarking method under various image processing attacks using three standard performance metrics: SSIM, PSNR, and NCC. These metrics assess visual quality, distortion level, and correlation between original and watermarked images with different attacks. As shown in Figure 12, without attack, the SSIM, PSNR, and NCC values are the highest. This indicates perfect preservation of visual quality and watermark integrity. Most attacks, including median filtering, JPEG compression, and sharpening, maintain high SSIM and NCC values (> 98%). This shows strong resistance to common distortions. PSNR values remain above 30 dB in these cases. This ensures minimal perceptual degradation. However, some attacks, including Gaussian noise, histogram quantization, and motion blur, led to a slight reduction in few metrics. PSNR of ∼25 dB and SSIM < 90% were observed. It shows more noticeable distortions. Despite this, the NCC values remain relatively high (above 96% in most cases), demonstrating the robustness of the watermark against noise and filtering attacks. This analysis validates the effectiveness of the proposed scheme in maintaining the fidelity and perceptual quality of the watermark in both mild and moderate attack scenarios.

Figures 14, 13, and 15 illustrate the impact of different values of the scaling parameter α on SSIM, PSNR, and NCC in various common image attacks. These results reveal a significant trade-off between imperceptibility of watermarks and robustness. As α increases, the robustness of the watermark improves. It can be observed by the consistently high NCC values in attacks, particularly for salt&pepper, Gaussian noise, and JPEG compression. However, this affects visible image quality. In Figure 14, higher values α result in reduced PSNR. This indicates a large distortion in the watermarked image. Similarly, the SSIM values in Figure 13 show a downward trend with increasing α, especially for high-distortion attacks such as motion blur and histogram quantization. This highlights

Table 1: PSNR, NCC, and SSIM of the watermarked images

Watermarked Image						
PSNR	40.322	38.926	40.014	39.703	40.133	40.130
NCC	99.944	99.868	99.907	99.925	99.923	99.895
SSIM	99.356	99.449	99.658	99.056	99.713	99.641



Figure 9: Different attacks were applied to the watermarked Lena image

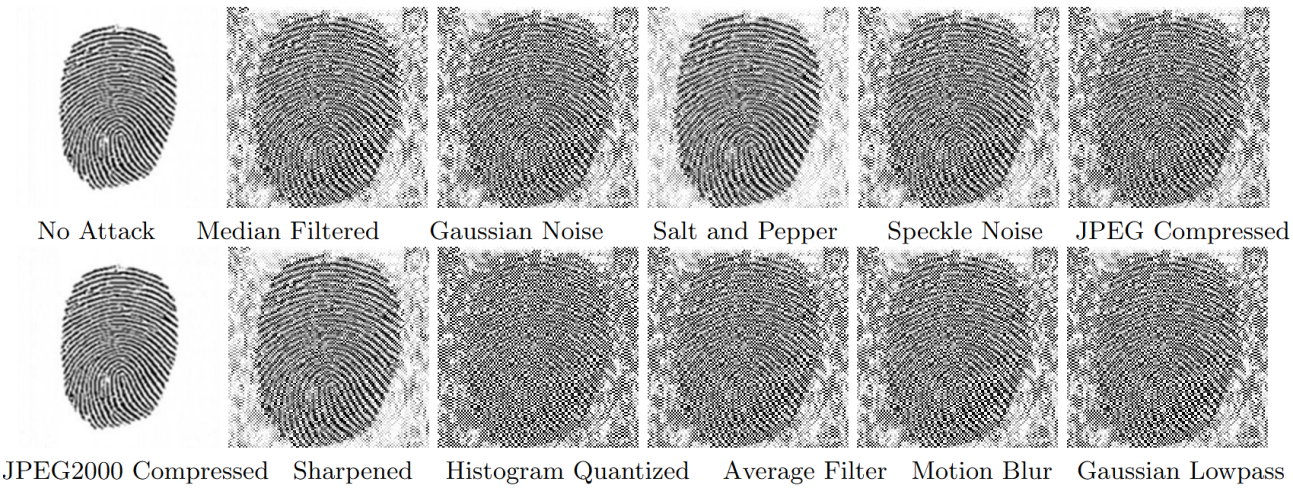


Figure 10: Extracted fingerprints from attacked images

the role of α watermarking systems. A smaller one α maintains high visual fidelity but offers less robustness against attacks. On the other hand, the increase in α improves resistance to tampering but reduces the image quality. Therefore, selecting an optimal α value is essential to balance robustness and imperceptibility in real-world applications.

Tables 2 and 3 demonstrate the robustness and imperceptibility of the proposed method under two common image processing attacks: JPEG compression and Gaussian lowpass filtering. The visual results of extracted/retrieved signatures and fingerprints, along with PSNR, NCC, and SSIM metrics, are used to evaluate the performance. As presented in Table 2, the watermarking algorithm shows strong resilience to JPEG compression, even at low quality factors (Q). The NCC remains above 97% in all compression levels, which indicates an accurate retrieval of the embedded watermark. SSIM and PSNR improve steadily as the qual-

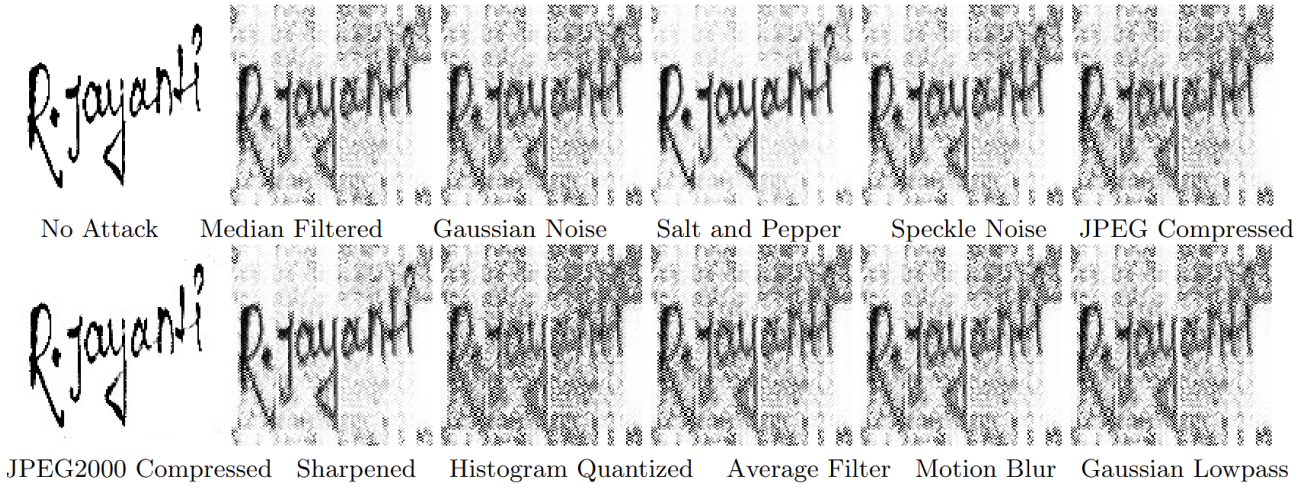


Figure 11: Extracted signatures from attacked watermarked images

Table 2: Watermark extraction from JPEG compressed watermarked images

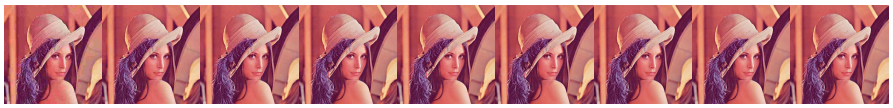
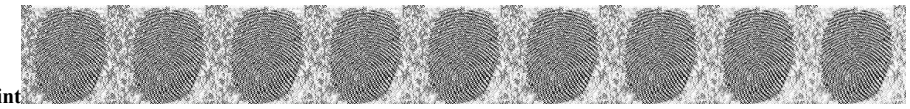
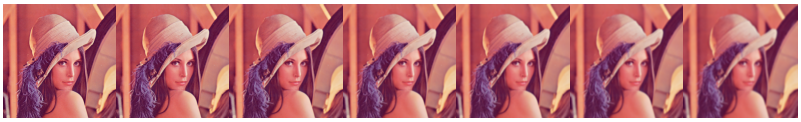
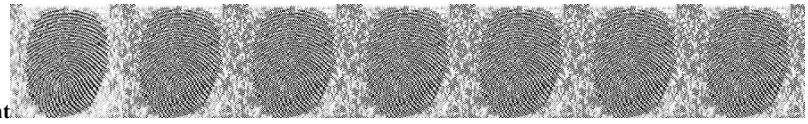
Quality factor (Q)	10	20	30	40	50	60	70	80	90
Watermarked image									
Extracted signature									
Extracted fingerprint									
PSNR	27.531	29.832	30.909	31.537	32.014	32.410	32.907	33.604	34.748
NCC	97.537	99.030	99.243	99.344	99.412	99.463	99.521	99.592	99.687
SSIM	95.931	97.378	97.885	98.133	98.311	98.444	98.601	98.789	99.048

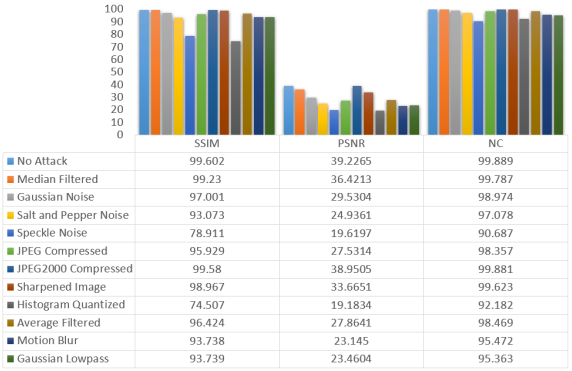
Table 3: Retrieval accuracy after Gaussian low-pass filtering with different sigma values

Sigma	0.5	1	1.5	2	2.5	3	3.5
Watermarked Image							
Retrieved Signature							
Retrieved Fingerprint							
PSNR	39.592	32.513	29.989	28.4333	27.330	26.480	25.788
NCC	99.898	99.481	99.070	98.667	98.280	97.907	97.544
SSIM	99.683	98.515	97.463	96.571	95.848	95.262	94.777

ity factor increases. This reflects better visual quality and less distortion in the watermarked image. In particular, even at a low Q=10, the watermark remains recoverable, although with slightly reduced visual fidelity. In Table 3, Gaussian low-pass filtering with increasing standard deviation (σ) is applied to assess the robustness of the watermark

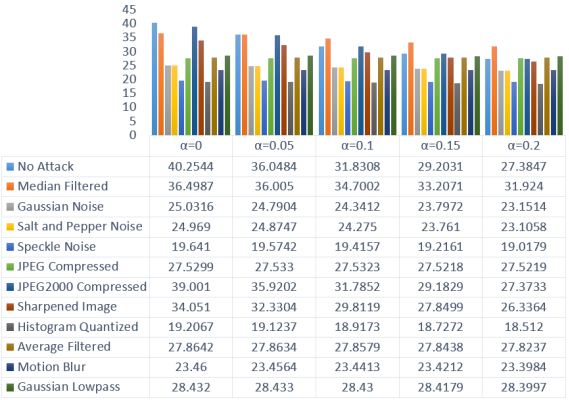
Table 4: PSNR, NCC and SSIM values obtained by the model for six tampered images

Tampered Images						
PSNR	11.200	25.209	27.531	20.126	21.907	16.325
NCC	54.491	97.140	98.357	93.170	95.128	77.110
SSIM	57.973	98.360	95.931	92.081	95.658	82.182



	SSIM	PSNR	NC
No Attack	99.602	39.2265	99.889
Median Filtered	99.23	36.4213	99.787
Gaussian Noise	97.001	29.5304	98.974
Salt and Pepper Noise	93.073	24.9361	97.078
Speckle Noise	78.911	19.6197	90.687
JPEG Compressed	95.929	27.5314	98.357
JPEG2000 Compressed	99.58	38.9505	99.881
Sharpened Image	98.967	33.6651	99.623
Histogram Quantized	74.507	19.1834	92.182
Average Filtered	96.424	27.8641	98.469
Motion Blur	93.738	23.145	95.472
Gaussian Lowpass	93.739	23.4604	95.363

Figure 12: SSIM, PSNR, and NCC of attacked watermarked images



	α=0	α=0.05	α=0.1	α=0.15	α=0.2
No Attack	40.2544	36.0484	31.8308	29.2031	27.3847
Median Filtered	36.4987	36.005	34.7002	33.2071	31.924
Gaussian Noise	25.0316	24.7904	24.3412	23.7972	23.1514
Salt and Pepper Noise	24.969	24.8747	24.275	23.761	23.1058
Speckle Noise	19.641	19.5742	19.4157	19.2161	19.0179
JPEG Compressed	27.5299	27.533	27.5323	27.5218	27.5219
JPEG2000 Compressed	39.001	35.9202	31.7852	29.1829	27.3733
Sharpened Image	34.051	32.3304	29.8119	27.8499	26.3364
Histogram Quantized	19.2067	19.1237	18.9173	18.7272	18.512
Average Filtered	27.8642	27.8634	27.8579	27.8438	27.8237
Motion Blur	23.46	23.4564	23.4413	23.4212	23.3984
Gaussian Lowpass	28.432	28.433	28.43	28.4179	28.3997

Figure 13: PSNR of different attacked watermarked images with varying alpha values

under blurring conditions. As σ increases from 0.5 to 3.5, PSNR and SSIM values decrease. This indicates an increasing distortion and reduced perceptual quality. However, the NCC remains relatively high (above 97%) throughout the study. This shows that the fingerprint can still be accurately retrieved even with significant smoothing. In addition, the robustness of the method is demonstrated against blurring attacks, although some degradation in image quality is unavoidable with stronger filtering. Both tables validate the trade-off between imperceptibility and robustness. The watermarking system maintains high fidelity in light to moderate attacks. It still retrieves the watermark with high accuracy, even under severe distortions. These findings support the algorithm's suitability for practical use where resilience to real-world manipulations is essential.

Table 4 presents the values of different performance metrics for six tampered images used to assess the robustness of the proposed scheme against various image tampering attacks. It includes JPEG compression (QF = 10), rotation by 90° clockwise, retouching, cropping, resampling cloning, retouching cloning, retouching splicing, resampling splicing, and retouching splicing. These scenarios helps to test both visual distortion and the effectiveness of watermark retrieval in manipulated scenarios. The results show significant variability in performance for different types of tampering. The PSNR values range from as low as 11.200 to 27.531. This shows different degrees of visual degradation due to tampering. The lower PSNR (e.g., 11.200) shows severe distortion, as seen in the first image. On the other hand, higher values suggest minor perceptual changes. Despite such variations, the NCC remains relatively strong in most cases. It was observed to be greater than 90% in four of the six altered images. This indicates that the model is capable of retrieving the embedded watermark with high accuracy, even with unwanted modifications. The lowest PSNR observed was 54.491. This shows the correlation between high distortion and reduced retrieval fidelity. SSIM values show that the image structure is better preserved in less tampered images (e.g., 98.360, 95.931) and more affected in heavily tampered ones (e.g., 57.973). This table shows the flexibility of the proposed watermarking system. Even in cases of aggressive tampering, the model achieves high performance. It demonstrates its applicability in real-world content authentication scenarios.

Figures 16 and 17 present the performance of the proposed method against various image tampering operations. Each row in the figures corresponds to a distinct tampering technique. Figure 16 presents a visual overview of the extraction of the watermark and the location of the tamper. For each tampered image, four representations are shown: (i) the forged watermarked image, (ii) the retrieved watermark signature, (iii) the extracted fingerprint watermark, and (iv) the localized tampered region. The watermarks and tamper localization were observed to stay identifiable and accurate despite major modifications. It shows the flexibility of the model for forgeries. Figure 17 further validates this by comparing predicted tamper regions with ground truth masks. Each row presents (i) the forged image, (ii)

Table 5: Performance analysis of the proposed tamper detection method under various forgery types

Tampered	TP	FP	FN	TN	Precision	Recall	Accuracy	F1 Score
retouched	28274	974	2663	230233	0.9667	0.9139	0.9861	0.9396
cloning with resampling	23075	6173	761	232135	0.7889	0.9681	0.9735	0.8694
cloning with retouching	18676	3148	6472	233848	0.8558	0.7426	0.9633	0.7952
splicing with resampling	33006	2002	1252	225884	0.9428	0.9635	0.9876	0.9530
splicing with retouching	14997	2219	3937	240991	0.8711	0.7921	0.9765	0.8297



	α=0	α=0.05	α=0.1	α=0.15	α=0.2
No Attack	99.674	99.274	99.386	97.302	96.115
Median Filtered	99.242	99.168	98.966	98.682	98.358
Gaussian Noise	93.187	92.819	92.112	91.266	90.148
Salt and Pepper Noise	93.041	92.901	92.047	91.219	90.008
Speckle Noise	78.955	78.706	78.115	77.385	76.636
JPEG Compressed	95.932	95.933	95.937	95.928	95.923
JPEG2000 Compressed	99.651	99.253	98.367	97.286	96.1
Sharpened Image	99.058	98.614	97.692	96.573	95.345
Histogram Quantized	74.61	74.31	73.62	93.095	72.467
Average Filtered	94.424	96.422	96.417	96.403	96.384
Motion Blur	93.738	93.727	93.687	93.631	93.568
Gaussian Lowpass	96.571	96.571	96.569	96.56	96.546

Figure 14: SSIMs of different attacked watermarked images with varying alpha values



	α=0	α=0.05	α=0.1	α=0.15	α=0.2
No Attack	99.912	99.769	99.394	98.898	98.329
Median Filtered	99.791	99.765	99.683	99.554	99.401
Gaussian Noise	97.14	96.984	96.675	96.256	95.694
Salt and Pepper Noise	97.098	97.042	96.622	96.228	95.649
Speckle Noise	90.722	90.616	90.326	89.92	89.514
JPEG Compressed	98.356	98.357	95.358	98.353	98.353
JPEG2000 Compressed	99.905	99.762	99.387	98.892	98.334
Sharpened Image	99.655	99.485	99.078	98.557	97.968
Histogram Quantized	92.228	92.034	91.572	91.095	90.956
Average Filtered	98.469	98.468	98.466	98.461	98.454
Motion Blur	95.742	95.737	95.722	95.7	95.956
Gaussian Lowpass	98.667	98.668	98.667	98.663	98.658

Figure 15: NCC of different attacked watermarked images with varying alpha values

the ground truth tamper mask, and (iii) the tamper map predicted by the proposed method. The close visual match between ground truth and predicted masks shows the reliability of the system in detecting the exact location of the tampering.

Table 5 presents a detailed evaluation of the proposed approach in eight types of image forgery. The details of the correct predictions and errors, along with various performance metrics, are presented. The results show near-ideal performance on different geometric and compression-based attacks.

This indicates that the system is highly effective in detecting these types of manipulations without introducing error. The detection performance was observed to decrease slightly when complex tampering techniques were combined with resampling or retouching. Lower precision and recall values are observed, especially for retouched cloning and retouched splicing. The F1 scores for both scenarios were 0.7952 and 0.8297, respectively. These drops suggest that although the system presented reliable detection, this decrease suggests that these forgeries pose greater challenges to the system. It is due to their fine alterations and blending effects. The method maintains high overall accuracy and consistent F1 scores > 0.79 in all simulated attacks. This illustrates the robustness of the proposed approach for real-world tamper localization and forgery detection tasks.

Tables 6 and 7 present a comprehensive comparative analysis of the proposed method with existing state-of-the-art techniques. As observed in Table 6, the proposed method uses a cover image size of 512×512 and a dual watermark size of 128×128 and 64×64. It can be seen from the tables that the existing methods either lack forgery detection altogether or have not been tested under attack situations. On the other hand, the proposed model successfully authenticates and detect forgeries in various attack situations. Table 7 further evaluates the robustness of the proposed method against various image processing attacks. Consistent high NC values (above 0.92 for all attacks), high PSNR (18.384–39.2265), and SSIM values (mostly above 0.98) were achieved. This approach offers better structural consistency under various distortions compared to existing methods (e.g., [24] and [25]). It performs especially well under Gaussian, median, and speckle noise. In most cases, it achieves higher SSIM and NC values. This indicates better accuracy in watermark retrieval and preservation of visual quality, making it a reliable choice for authentication and tamper detection tasks.

In addition to the above results, Table 8 provides detailed responses to the RQs, formulated to evaluate the effectiveness and novelty of the proposed watermark and tamper detection method.

6 Concluding remarks

This study presents a robust watermark-based framework for digital image authentication and tamper detection. It uses three transformations for robust watermark embed-

Table 6: Comparison of the proposed method with existing approaches in terms of authentication, forgery detection, PSNR, SSIM, types of forgery detected, and robustness under various attacks.

Ref	Authentication	Forgery Detection	PSNR	SSIM	Types of forgery detected	Forgeries detected under types of attacks
[24]	✔	✖	39.45	0.9964	–	–
[25]	✔	✖	31.49	0.8618	–	–
[29]	✔	✔	43.79	–	Only copy-move	–
[30]	✔	✔	46.16	0.9988	Copy-move and splicing	–
[35]	✔	✔	48.21	–	Cropping and copy-move	–
[36]	✔	✔	46.36	0.9963	Cropping up to 50%	–
[37]	✔	✔	55.92	0.9999	Resampling, retouching, copy-move, and splicing	–
Proposed	✔	✔	39.23	0.9960	Copy-move, splicing, retouching, resampling, and cropping	Salt & Pepper Noise,Gaussian Noise,Gaussian Filter,Median Filter,Sharpening,Motion Blur,Average Filter,Histogram Equalization ,JPEG,JPEG2000

Table 7: Comparison of NC, PSNR, and SSIM of the proposed method for various attacks with existing techniques

Attack Type	NC			PSNR			SSIM		
	[24]	[25]	Proposed	[24]	[25]	Proposed	[24]	[25]	Proposed
Salt & Pepper Noise	1	0.9943	0.9708	–	26.7142	24.9361	–	0.931	0.970
Gaussian Noise	0.9988	0.9938	0.9897	–	31.4845	29.5304	–	0.970	0.970
Speckle Noise	0.9999	0.9814	0.9069	–	22.4363	19.6197	–	0.789	0.891
Gaussian Filter	0.9995	0.9995	0.9536	–	26.8799	23.4604	–	0.937	0.939
Median Filter	0.9999	0.9975	0.9979	–	31.6371	36.4213	–	0.992	0.992
Sharpening	0.999	–	0.9962	–	–	33.6651	–	–	0.990
Motion Blur	0.997	–	0.9547	–	–	23.1454	–	–	0.937
Average Filter	0.999	0.9976	0.9847	–	26.7821	27.8641	–	0.964	0.964
Histogram Equalization	0.985	0.9476	0.9218	–	10.3095	19.1834	–	0.906	0.942
JPEG	0.999	–	0.9836	–	–	27.5314	–	–	0.959
JPEG2000	0.999	–	0.9988	–	–	38.9505	–	–	0.996

Table 8: Comprehensive answers to the RQs

RQ No	Answer
RQ1	DWT analyzes spatial-frequency features for localized detection. On the other hand, DCT compresses energy into fewer coefficients. This improves resilience to compression. SVD provides stability against noise, rotation, and scaling that preserves image integrity. In addition, a tamper detection mask is inserted to localize the tampered section. Together, these methods provide reliable image content integrity verification and forgery localization.
RQ2	The suggested model resists attacks with high PSNR, SSIM, and NCC. SSIM and NCC values of ∼100% in watermarked images. It highlights the watermarking's durability and the efficiency of the detection model even in noisy environments.
RQ3	Even after different types of attacks on the watermarked photos, the results provided in figures 16 and 17 indicate the effectiveness of the proposed model.

ding: DWT, DCT, and SVD. The model embeds both fingerprint and signature watermarks. It also uses a tamper detection mask in the LH2 sub-band of the LH1 band for accurate localization of manipulated regions. The experimental results show high imperceptibility. PSNR, SSIM, and NCC values greater than 38 dB, 99%, 99.8% were received, respectively. This shows minimal visual distortion and strong watermark fidelity. The system remains highly robust under various attacks and maintains the integrity of the watermark. High F1 scores of 0.9530 and various other near-ideal metrics were received for tamper detection even in complex forgery situations. Various other visualizations further signify the efficacy of the proposed scheme in preserving watermark information for different manipulations. Although the proposed model successfully localized manipulations for various attacks, it faces challenges in dealing with rotation and compression. In addition, the requirement of original watermark data during verification limits the model's applicability in blind authentication scenarios. Future research should focus on blind watermarking techniques and deep learning-based methods to locate tampered regions. This provides improved automation and adaptability. Integration of emerging technologies, including blockchain and artificial intelligence, could fur-

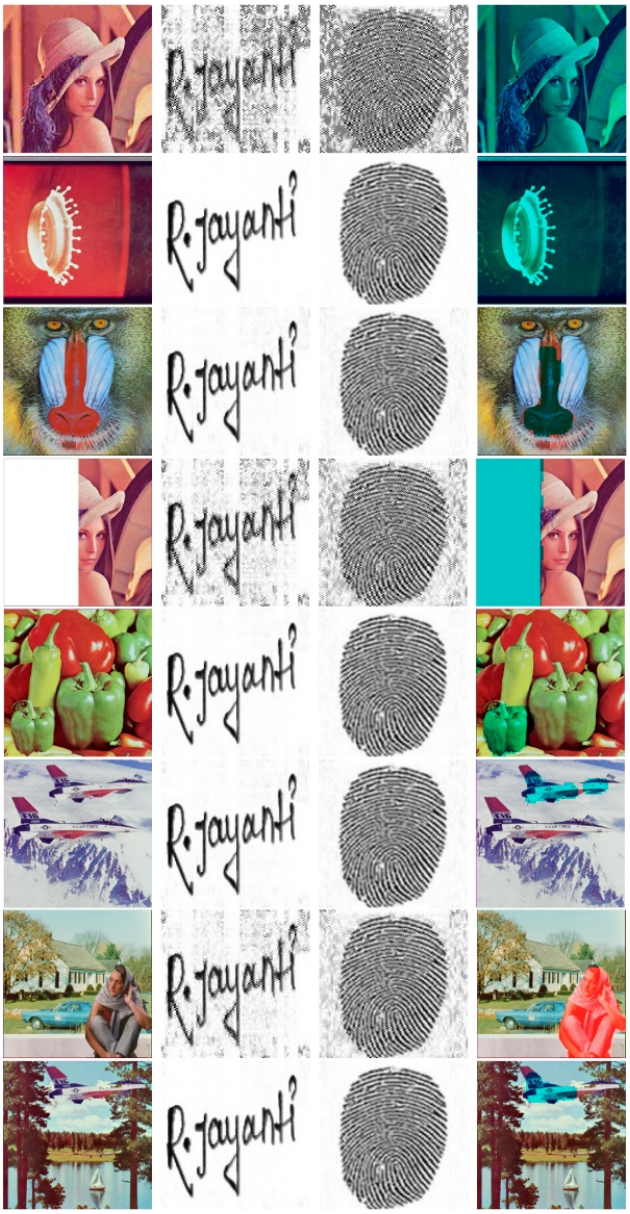


Figure 16: Watermark extraction and tamper localization across various forgery types

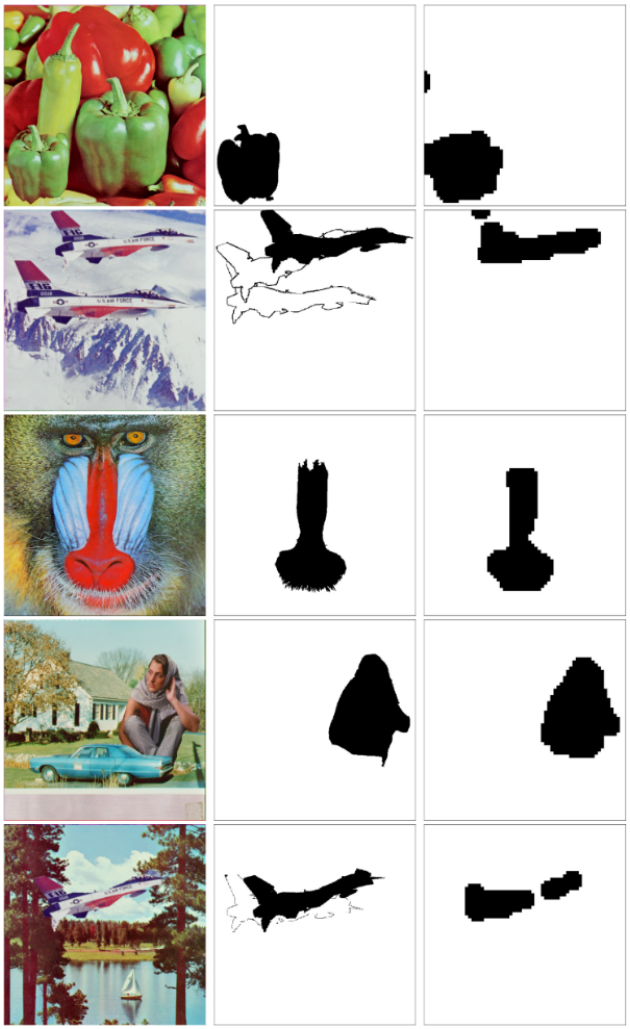


Figure 17: Comparison of predicted and ground truth tampered regions for different manipulations

ther strengthen the approach. This can help to widen its applicability and address challenges in rotation and compression challenges.

References

[1] Sprout Social. *80+ Must-Know Social Media Marketing Statistics for 2025*. https://sproutsocial.com/insights/social-media-statistics/. Accessed: 2025-11-10. 2025.

[2] Nikitha Sreekanthaswamy et al. "Digital tools and methods". In: *Enhancing School Counseling With Technology and Case Studies*. IGI Global Scientific Publishing, 2025, pp. 25–48. URL: https://doi.org/10.4018/979-8-3693-8392-6.ch002.

[3] Aditya Kumar Sahu et al. "A study on content tampering in multimedia watermarking". In: *SN Computer Science* 4.3 (2023), p. 222. URL: https://doi.org/10.1007/s42979-022-01657-1.

[4] Paweł Duszejko, Tomasz Walczyna, and Zbigniew Piotrowski. "Detection of Manipulations in Digital Images: A Review of Passive and Active Methods Utilizing Deep Learning". In: *Applied Sciences* 15.2 (2025), p. 881. URL: https://doi.org/10.3390/app15020881.

[5] Anna Gruending et al. "Born Too Soon: learning from the past to accelerate action in the next decade". In: *Reproductive Health* 22.2 (2025), pp. 1–15. URL: https://doi.org/10.1186/s12978-025-02044-8.

[6] India Today Web Desk. "FAKE ALERT: No, DGP of Gujarat is not touching Rajnath Singh's feet".

In: *India Today* (Nov. 2017). Accessed: 2024-05-04. URL: https://www.indiatoday.in/fyi/story/rajnath-singh-fake-image-dgp-gujarat-congress-sanjay-jha-1078093-2017-11-01.

[7] Hany Farid. *Photo Tampering Throughout History*. Online manuscript. Accessed via Georgia Tech faculty website (PDF, 30 pages). 2008. URL: https://faculty.cc.gatech.edu/~beki/cs4001/history.pdf.

[8] Shishir Kumar Raj et al. "An Automated Deep Learning Model for Detecting Image Forgeries". In: *2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT)*. Vol. 1. IEEE. 2024, pp. 1–6. URL: https://doi.org/10.1109/ICEECT61758.2024.10738982.

[9] Santoshini Panda and Minati Mishra. "Passive techniques of digital image forgery detection: developments and challenges". In: *Advances in Electronics, Communication and Computing: ETAEERE-2016*. Springer, 2017, pp. 281–290. URL: https://doi.org/10.1007/978-981-10-4765-7_29.

[10] Wenbo Wan et al. "A comprehensive survey on robust image watermarking". In: *Neurocomputing* 488 (2022), pp. 226–247. URL: https://doi.org/10.1016/j.neucom.2022.02.083.

[11] Sanjeeb Kumar Behera and Minati Mishra. "Steganography–A Game of Hide and Seek in Information Communication". In: *arXiv preprint arXiv:1604.00493* (2016). URL: https://doi.org/10.48550/arXiv.1604.00493.

[12] Qi Peng Yu Feng, De-Biao He, and Min Luo. "A survey on threshold digital signature schemes". In: *Frontiers of Computer Science* 20.4 (2025), p. 2004806. URL: https://doi.org/10.1007/s11704-025-41297-1.

[13] Navneet Kaur, Neeru Jindal, and Kulbir Singh. "Passive image forgery detection techniques: A review, challenges, and future directions". In: *Wireless Personal Communications* 134.3 (2024), pp. 1491–1529. URL: https://doi.org/10.1007/s11277-024-10959-x.

[14] Divya Prathana Timothy and Ajit Kumar Santra. "Detecting Digital Image Forgeries with Copy-Move and Splicing Image Analysis using Deep Learning Techniques". In: *International Journal of Advanced Computer Science & Applications* 15.5 (2024). URL: https://doi.org/10.14569/ijacsa.2024.01505131.

[15] Krishna H Hingrajiya and Chintan Patel. "An Approach for Copy-Move and Image Splicing Forgery Detection using Automated Deep Learning". In: *2023 International Conference on Emerging Smart Computing and Informatics (ESCI)*. IEEE. 2023, pp. 1–5. URL: https://doi.org/10.1109/ESCI56872.2023.10100202.

[16] Saurabh Agarwal, Savita Walia, and Ki-Hyun Jung. "A cohesive forgery detection for splicing and copy-paste in digital images". In: *Multimedia Tools and Applications* 84.1 (2025), pp. 147–163. URL: https://doi.org/10.1007/s11042-024-19825-1.

[17] Ingemar Cox et al. "Digital watermarking". In: *Journal of Electronic Imaging* 11.3 (2002), pp. 414–414. URL: https://doi.org/10.1117/1.1494075.

[18] Mahbuba Begum and Mohammad Shorif Uddin. "Digital image watermarking techniques: a review". In: *Information* 11.2 (2020), p. 110. URL: https://doi.org/10.3390/info11020110.

[19] Ms Shubhangi D Mashalkar and SS Shirgan. "Design of watermarking scheme in medical image authentication using DWT and SVD technique". In: *2017 International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE. 2017, pp. 955–960. URL: https://doi.org/10.1109/ICCMC.2017.8282609.

[20] Dengsheng Zhang. "Wavelet transform". In: *Fundamentals of image data mining: Analysis, Features, Classification and Retrieval*. Springer, 2019, pp. 35–44. URL: https://doi.org/10.1007/978-3-030-17989-2_3.

[21] Shaifali M Arora et al. "A DWT-SVD based robust digital watermarking for digital images". In: *Procedia computer science* 132 (2018), pp. 1441–1448. URL: https://doi.org/10.1016/j.procs.2018.05.076.

[22] Narima Zermi et al. "A DWT-SVD based robust digital watermarking for medical image security". In: *Forensic science international* 320 (2021), p. 110691. URL: https://doi.org/10.1016/j.forsciint.2021.110691.

[23] David Mata-Mendoza et al. "Secured telemedicine of medical imaging based on dual robust watermarking". In: *The Visual Computer* 38.6 (2022), pp. 2073–2090. URL: https://doi.org/10.1007/s00371-021-02267-3.

[24] Bhargavi Mokashi et al. "Efficient Hybrid Blind Watermarking in DWT-DCT-SVD with Dual Biometric Features for Images". In: *Contrast Media & Molecular Imaging* 2022.1 (2022), p. 2918126. URL: https://doi.org/10.1155/2022/2918126.

[25] Shyam Singh Rajput, Bhaskar Mondal, and Farheen Qamar Warsi. "A robust watermarking scheme via optimization-based image reconstruction technique". In: *Multimedia Tools and Applications* 82.16 (2023), pp. 25039–25060. URL: https://doi.org/10.1007/s11042-023-14363-8.

[26] Runyi Hu et al. “Robust-wide: Robust watermarking against instruction-driven image editing”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 20–37. URL: https://doi.org/10.1007/978-3-031-72670-5_2.

[27] Paarth Neekhara et al. “FaceSigns: semi-fragile watermarks for media authentication”. In: *ACM Transactions on Multimedia Computing, Communications and Applications* 20.11 (2024), pp. 1–21. URL: https://doi.org/10.1145/3640466.

[28] Aakash Varma Nadimpalli and Ajita Rattani. “Social media authentication and combating deepfakes using semi-fragile invisible image watermarking”. In: *Digital Threats: Research and Practice* 5.4 (2024), pp. 1–30. URL: https://doi.org/10.1145/3700146.

[29] De Li et al. “A reversible watermarking for image content authentication based on wavelet transform”. In: *Signal, Image and Video Processing* 18.3 (2024), pp. 2799–2809. URL: https://doi.org/10.1007/s11760-023-02950-z.

[30] Ferda Ernawan et al. “Fragile and Robust Dual Image Watermarking based on DWT-SVD”. In: *2024 10th International Conference on Mechatronics and Robotics Engineering (ICMRE)*. IEEE. 2024, pp. 247–252. URL: https://doi.org/10.1109/ICMRE60776.2024.10532164.

[31] Ernesto Moya-Albor et al. “Bio-inspired watermarking method for authentication of fundus images in computer-aided diagnosis of retinopathy”. In: *Mathematics* 12.5 (2024), p. 734. URL: https://doi.org/10.3390/math12050734.

[32] Wu-Chih Hu et al. “Effective image forgery detection of tampered foreground or background image based on image watermarking and alpha mattes”. In: *Multimedia Tools and Applications* 75 (2016), pp. 3495–3516. URL: https://doi.org/10.1007/s11042-015-2449-0.

[33] Thai-Son Nguyen. “Fragile watermarking for image authentication based on DWT-SVD-DCT techniques”. In: *Multimedia Tools and Applications* 80.16 (2021), pp. 25107–25119. URL: https://doi.org/10.1007/s11042-021-10879-z.

[34] Moataz Z Salim, Ali J Abboud, and Remzi Yildirim. “A visual cryptography-based watermarking approach for the detection and localization of image forgery”. In: *Electronics* 11.1 (2022), p. 136. URL: https://doi.org/10.3390/electronics11010136.

[35] Chia-Chen Lin et al. “Fragile watermarking for tamper localization and self-recovery based on AMBTC and VQ”. In: *Electronics* 12.2 (2023), p. 415. URL: https://doi.org/10.3390/electronics12020415.

[36] Cai Zhan et al. “Reversible Image Fragile Watermarking with Dual Tampering Detection”. In: *Electronics* 13.10 (2024), p. 1884. URL: https://doi.org/10.3390/electronics13101884.

[37] Prajanto Wahyu Adi et al. “Efficient Fragile Watermarking for Image Tampering Detection using Adaptive Matrix on Chaotic Sequencing”. In: *Intelligent Systems with Applications* (2025), p. 200530. URL: https://doi.org/10.1016/j.iswa.2025.200530.

[38] P. Aberna and L. Agilandeeswari. “Optimal Semi-Fragile Watermarking Based on Maximum Entropy Random Walk and Swin Transformer for Tamper Localization”. In: *IEEE Access* 12 (2024), pp. 37757–37781. DOI: https://doi.org/10.1109/ACCESS.2024.3370411.

[39] Asmaa Hatem Jawad et al. “A New Face Image Manipulation Localization and Recovery Algorithm Using Image Watermarking and Integer Wavelet Transform”. In: *Mathematical Modelling of Engineering Problems* 11.9 (2024). DOI: https://doi.org/10.18280/mmep.110920.

[40] USC-SIPI. *USC-SIPI Image Database*. https://sipi.usc.edu/database/. [Accessed: 2024-06-03]. 2024.

Hybrid Actor-Critic Based Low-Overhead Scheduling Using MDP for Large-Scale Edge Computing Networks

Qian Zhang
Henan Institute of Economics and Trade, School of Artificial Intelligence, Zhengzhou 450046, China
E-mail: zhangqian@igit.ac.cn

Keywords: edge computing, low-cost scheduling, computing power resources, markov decision, hybrid actor critic algorithm

Received: August 14, 2025

The inconsistency of application requirements in large-scale edge computing networks leads to the need for edge computing scheduling to optimize multiple objectives at the same time. There are significant inherent conflicts between these objectives, and optimizing a single objective often comes at the expense of sacrificing the performance of other objectives, making it difficult to achieve global optimality in large-scale dynamic edge environments. Therefore, a hybrid Actor-Critic Based Low-Overhead Scheduling Using MDP for Large-Scale is proposed. Build a three-layer architecture of "end edge cloud", including terminal collection, edge processing upload, and cloud storage analysis. Enable data flow and processing at different levels, comprehensively analyze the performance and interrelationships of various targets at different levels, and reduce bandwidth and cloud load. Set targets for transmission efficiency, computation delay, transmission delay, and energy consumption, and comprehensively evaluate scheduling performance through cumulative weighted overhead. Considering the impact of various objectives on a global scale, scheduling strategies need to comprehensively consider all objectives when formulating. To achieve this goal, a Markov decision process is used to model the system dynamics, balancing unloading and resource allocation through state space, action space, and reward function, accurately capturing the characteristics of large-scale dynamic edge environments. Considering the changes in system state over time and the impact of different actions on system state, long-term weighted overhead is minimized; At the same time, the hybrid actor critic algorithm is introduced to process different types of actions through discrete and continuous actor networks, and the experience playback mechanism is used to solve the problem of real-time reward acquisition in asynchronous environments. In addition, the critical network is combined to optimize resource allocation, effectively handle complex action space, further balance the relationship between multiple targets, and achieve efficient and low-cost scheduling of large-scale edge computing network computing resources. Experimental results show that the proposed method performs well in large-scale edge computing network resource scheduling. In terms of resource utilization, the average is 93.2%, and the response time is always below 4ms. In terms of energy efficiency, the average task energy consumption is the lowest, saving more than one-third of energy compared to the comparative method. In scalability evaluation, when facing large-scale tasks, the cumulative weighted overhead increases almost linearly, with a growth rate far lower than the comparative method, demonstrating excellent scalability.

Povzetek: Predlagani pristop omogoča učinkovito več-ciljno razporejanje v velikih robnih omrežjih, saj uravnoteži prenos, zakasnitev, energijo in porabo virov v dinamičnem okolju. Rezultati kažejo visoko izkoriščenost virov, nizko zakasnitev ter dobro energetsko in skalabilnostno učinkovitost.

1 Introduction

As a key infrastructure supporting low latency and high reliability applications, edge computing effectively alleviates transmission delay and bandwidth pressure under the traditional cloud computing mode by sinking computing, storage and network resources to the network edge close to the data source, and provides real-time data processing capabilities for smart city, industrial Internet, automatic driving and other scenarios [1]. However, with the increasing number of edge devices and the growing complexity of application demands, the scheduling of computing power resources is facing enormous challenges. Traditional computing resource scheduling methods often have high overhead, requiring a large amount of network bandwidth and computing resources for resource status monitoring and task allocation, and may also lead to untimely scheduling decisions, affecting system performance [2-3]. In large-scale edge computing networks, this high overhead scheduling mode will

further aggravate the resource tension situation and reduce the overall efficiency and reliability of the system. Therefore, researching a low-cost computing resource scheduling method has important practical significance. The low-cost scheduling method aims to achieve efficient allocation of computing power resources with minimal resource consumption, improve resource utilization, and reduce system operating costs [4]. Focus on large-scale edge computing networks, deeply analyze the characteristics and requirements of its computing resource scheduling, explore an innovative low overhead scheduling method, in order to provide strong technical support for the development of edge computing networks, and promote the wide application of edge computing in more fields [5].

Sheng et al. [6] proposed a multi-objective task scheduling algorithm based on an improved competitive deep double-Q network. Firstly, the deep double-Q network decomposes the maximum operation in the target into action selection and evaluation to eliminate overestimation. It uses an immediate reward experience sample classification method to store and prioritize important samples, improving sample utilization and training speed; Then, introduce competitive network structure optimization neural network; Finally, the stability of the algorithm is improved using soft update methods, and the dynamic greedy exponential descent method is used for optimization. The Pareto optimal solution is obtained through different linear weighted combinations to minimize response time and energy consumption. Although the improved competitive deep double-Q network can reduce fluctuations and uncertainties in the decision-making process and enhance system stability. However, due to the complexity of the algorithm and limitations of the training data, the improved competitive deep Q-network may have limited generalization ability in new environments or unseen situations, resulting in performance degradation, reduced resource utilization, and increased task latency and energy consumption.

Hu et al. [7] proposed a particle swarm optimization algorithm based on hierarchical learning for allocating computing resources. Firstly, the particle swarm algorithm is used to optimize the computing speed, download power consumption, data offloading percentage, and network bandwidth of mobile devices to minimize total energy consumption. Then, in response to the problem of particle swarm optimization being prone to local optima and low diversity, a hierarchical learning strategy is introduced to improve the algorithm and enhance its global search capability. Finally, improving the algorithm effectively achieves rational allocation of computing resources and reduces total energy consumption. Although the algorithm can search at different levels through a hierarchical learning strategy, it can explore the solution space more comprehensively and avoid getting stuck in local optima. This helps to find better scheduling solutions in situations where computing power resources are limited. However, due to the introduction of hierarchical learning strategies, the computational complexity of the algorithm increases, especially when dealing with large-scale problems or complex environments, requiring more computing resources to perform hierarchical learning and particle update operations. Under the constraint of low computational resource consumption, the execution efficiency of the algorithm decreases.

Li et al. [8] proposed a resource scheduling method based on Lagrange multiplier method and joint optimization strategy for communication resources. Firstly, based on signal strength and traffic load, the access point (AP) switching strategy utilizes the global view and centralized control capabilities of the SDN controller to obtain, manage, and analyze information, calculate weights, and compare them to formulate switching strategies; Then, in order to improve system resource utilization, meet different application performance requirements, and maintain user QoS experience, a joint optimization strategy for computing and communication resources based on Lagrange multiplier method is proposed. By calculating and analyzing the task execution delay and energy consumption of edge servers and local terminals, optimization schemes for sub channel allocation and resource allocation are formulated. Although the Lagrange multiplier method transforms the allocation of computing power resources and communication constraints into Lagrange functions, it achieves unconstrained solving of constrained optimization problems. However, this method is usually optimized based on static or quasi-static assumptions, but in dynamic scenarios such as wireless communication, network topology, user demands, and channel states may change rapidly. If the scheduling strategy fails to respond to these changes in a timely manner, it may lead to delayed resource allocation and inability to meet real-time QoS requirements.

The Markov decision process models the dynamic environment as a framework of states, actions, transition probabilities, and reward functions based on Markov properties, accurately describing the relationship between edge node computing power state changes and scheduling decisions. By solving the optimal strategy, it maximizes long-term benefits and effectively addresses the uncertainty and randomness of network states; The hybrid actor critic algorithm integrates strategy gradient and value function estimation, with the actor network directly learning the optimal scheduling strategy and the critic network evaluating the strategy value. The two guide each other to improve performance, especially in continuous action space tasks. This algorithm can efficiently handle complex computing power allocation problems and reduce computational complexity [9]; After combining the two, the Markov decision process provides a theoretical framework and optimization objectives, while the hybrid actor critic algorithm provides an

efficient solution method. While reducing the cost of computing resources, the scheduling strategy can be dynamically adjusted based on real-time network status and task requirements, achieving a dual improvement in computing resource utilization and task execution efficiency, and enhancing the system's adaptability to dynamic network environments [10]. Therefore, based on the advantages of Markov decision process and hybrid actor critic algorithm, this paper studies and analyzes the low-cost scheduling of computing resources. The core route of this paper is as follows:

(1) Construction of large-scale edge computing network structure. Build a three-tier "end edge cloud" architecture for large-scale edge computing. Terminal collection, edge processing upload, cloud storage analysis, reducing transmission bandwidth and cloud load, and improving system performance.

(2) Establish goals for computing resource scheduling. Clearly define transmission efficiency, computation delay, transmission delay, and energy consumption as scheduling objectives, measured by cumulative weighted overhead. Set weights, balance objectives, and achieve efficient resource utilization with low costs.

(3) Low-cost scheduling strategy for computing power resources. The dynamic characteristics of edge computing are modeled based on MDP. For mixed action spaces, the HA2C algorithm is used to split actions and introduce experience replay to achieve low-cost scheduling of computing resources. Low-cost scheduling of computing power resources

1.1 Large scale edge computing network architecture

Large scale edge computing refers to the deployment of a large number of computing, storage and network resources at the network edge to support the real-time data processing requirements of large-scale IoT devices and applications. With the rapid development of IoT technology, more and more devices are connected to the network, generating massive amounts of data [11]. If all of this data is transmitted to the cloud for processing, it will not only occupy a large amount of network bandwidth, but may also lead to network congestion and latency. Large scale edge computing transfers data processing and analysis tasks from the cloud to the network edge to improve the real-time, reliability and flexibility of the system [12].

Edge computing is a distributed computing model, which pushes data processing and storage capacity from the centralized cloud data center to edge devices closer to the source or end user of data generation. The core idea of edge computing is to perform computing tasks as close to the data source as possible to reduce network transmission time and cost [13]. By deploying intelligent hardware at the network "edge", such as smart phones, tablets, sensors, routers, etc., edge computing can complete the collection, processing and decision-making process of some data locally, and only upload necessary information to the cloud.

According to different composition and functions, edge computing can be divided into a three-tier architecture of "end edge cloud". The edge layer is a computing layer inserted between the traditional terminal layer and the cloud layer. The specific composition of each layer is as follows:

(1) Terminal layer: The terminal layer is located at the bottom of the architecture and consists of sensing devices that are close to the user and have data collection and information perception functions, such as smartphones, sensor modules, etc. The terminal layer is mainly responsible for perceiving and collecting user information and event data, and uploading these data to the edge layer for processing [14].

(2) Edge layer: The edge layer is located between the terminal layer and the cloud layer, mainly composed of devices such as gateways and set-top boxes with certain storage and computing capabilities, also known as "edge nodes". This layer can localize the data collected by the terminal layer, provide localization services, and upload the data to the cloud layer according to needs; It can also receive instruction information sent by the upper layer and execute corresponding operations.

(3) Cloud layer: The top layer of the architecture is the cloud computing center layer, which is composed of servers with high computing and storage capabilities. The cloud center adopts a centralized computing model, which can store and analyze massive amounts of data transmitted from the bottom layer for a long time, providing comprehensive information decision-making services [15].

Edge computing is responsible for local analysis and processing of nearby data and provides real-time services by introducing an edge layer with certain storage and computing capabilities, which greatly reduces the network bandwidth pressure of data transmission and the load pressure of cloud computing center layer [16]. Build the overall architecture of edge computing.

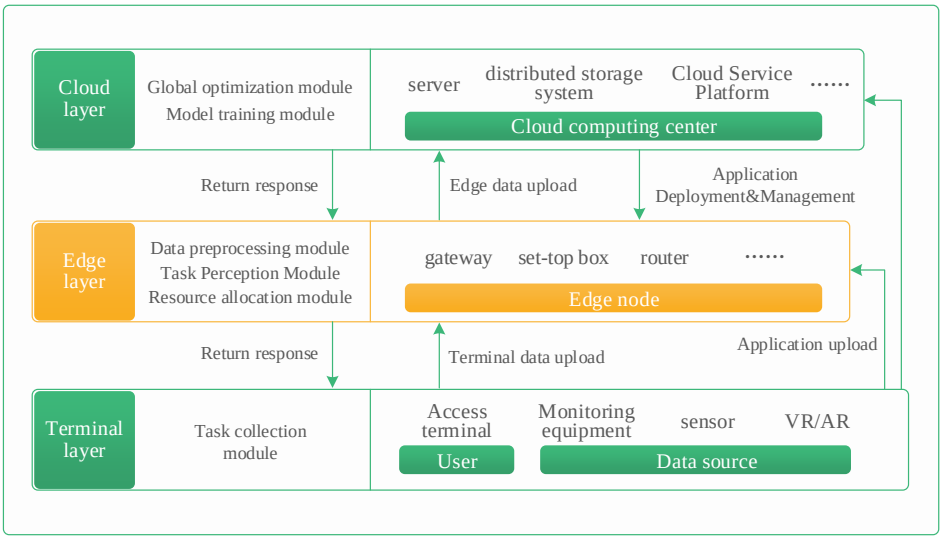


Figure 1: Overall architecture of edge computing

1.2 Scheduling objective setting

In the low overhead scheduling of computing resources in large-scale edge computing networks, there are multiple conflicting goals. By setting scheduling objectives, the importance and priority of each objective can be clarified, providing guidance for the formulation of scheduling strategies and finding a balance point between multiple conflicting objectives to ensure that scheduling methods achieve efficient resource utilization and low overhead while meeting system real-time, reliability, and flexibility requirements [17-18].

1.2.1 Transmission efficiency

Transmission efficiency is committed to improving the transmission efficiency of data in the network, ensuring that data can quickly and accurately reach the destination from the source, while reducing packet loss and error rates during transmission, thereby enhancing the overall performance of the system. When the computing resource scheduling task is unloaded to the edge computing node for processing, the task related data will be uploaded to the computing node through the wireless network [19-20]. The transmission delay is affected by the amount of data uploaded by the task and the channel state. In the actual transmission process, the amount of uploaded data is determined by the actual task, while the channel state follows a fading model, that is, the channel state changes at any time. According to Shannon's law, the channel data transmission rate can be expressed as:

$$Tr = \delta \cdot \log_2\left(1+\frac{SNR}{noise}\right) \tag{1}$$

In the formula, δ represents channel bandwidth; $noise$ represents Gaussian white noise; SNR follows the Nakagami-m distribution, representing the channel fading model, and its distribution function calculation formula is as follows:

$$Q(z) = \frac{2m^m z^{2m-1}}{\Gamma(m) Qr^m} \exp\left(-\frac{-mz^2}{Qr^m}\right) \tag{2}$$

In the formula, Qr represents the average power; $\Gamma(m)$ represents gamma function; m represents the fading coefficient. When $m=1$, formula (2) degenerates into Rayleigh fading; When $m=\infty$, it represents no decay. By changing the value of m, SNR can be transformed into various fading models.

1.2.2 Calculate delay

After the task offloading decision is given by the agent, the task is scheduled to the corresponding computing node for processing, and the computation delay of the task is determined by the computation amount CPU_{task} of the task and the computation capacity CPU_{node} of the corresponding computing node. The calculation formula is as follows:

$$L_c = CPU_{task} \Big/ CPU_{node} \tag{3}$$

1.2.3 Transmission delay

When a pending task is scheduled to be processed at the edge, the processing delay of the task is composed of computation delay and transmission delay. The formula for calculating transmission delay is as follows:

$$L_\tau = Data_{task} \Big/ Tr \tag{4}$$

In the formula, $Data_{task}$ represents the total amount of data that needs to be uploaded when the task is offloaded to the edge end; Tr represents the current channel data transmission rate.

1.2.4 Energy consumption

The total energy consumption of a task from start to finish is composed of transmission energy consumption and computation energy consumption. The determining factors of transmission energy consumption include the unit time transmission energy consumption NC_u and the task transmission delay, and their calculation formulas are as follows:

$$NC_\tau = NC_u \cdot L_\tau \quad (5)$$

The determining factor for calculating energy consumption is the calculation delay of the energy consumption task per unit time, and its calculation formula is as follows:

$$NC_c = NC_u \cdot L_c \quad (6)$$

The total energy consumption for processing a task can be expressed as:

$$NC = NC_\tau + NC_c \quad (7)$$

1.2.5 Accumulated weighted expenses

The comprehensive evaluation index for the performance of scheduling strategies is the long-term weighted overhead $Cost$, which is the weighted sum of the calculation delay, transmission delay, and energy consumption of the computing system. The overall weighted cost is expressed as:

$$Cost_i = \sum_{b=1}^{B}\sum_{d=1}^{D}\alpha L_c + \sum_{b=1}^{B}\sum_{d=1}^{D}\beta L_\tau + \sum_{b=1}^{B}\sum_{d=1}^{D}\gamma NC \quad (8)$$

$$Cost = \sum_{i}^{itr} Cost_i \quad (9)$$

In the formula, α、β、γ represents the weight of computation delay, transmission delay, and energy consumption, with a higher value indicating greater emphasis on this part of the overhead and $\alpha+\beta+\gamma=1$; $Cost_i$ represents the cost of the i-th iteration; B represents the number of tasks per iteration; D represents the number of subtasks included in each task. $Cost$ represents the overall weighted cost, where itr represents the total number of iterations. Thus, an effective balance between the objectives of computing resource scheduling has been achieved.

1.3 Low cost scheduling strategy for computing power resources

Large scale edge computing networks are highly dynamic, including the joining and leaving of devices, changes in network topology, randomness of task arrival, etc. [21]. Setting only static goals cannot adapt to this dynamically changing environment. In actual operation, it is necessary to adjust the scheduling strategy in a timely manner according to the dynamic changes of the system to ensure comprehensive optimization of multiple objectives in different situations. Markov decision process (MDP) can model the offloading decision and resource allocation based on the probability of system state transition. MDP models the dynamic behavior of the system through state space, action space, and reward function, and then selects the optimal scheduling action based on the current system state, and achieves the global optimization goal with long-term cumulative rewards [22]. MDP has no aftereffect and is highly adaptable to the dynamic characteristics of edge computing, thus providing a solid theoretical support and an effective solution framework for low overhead scheduling problems. The following is the Markov decision process transformation process [23].

The MDP model in the edge computing scenario can be expressed as a five tuple: $\chi=(S,A,P,R,\Upsilon)$, where S is the state space; A is the action space; P represents the state transition rate; R is the reward function, $\Upsilon\in[0,1]$ is the discount factor, balancing the importance of current and future expenses (usually taking $\Upsilon\geq 0.9$).

(1) State space: describes the current state of the system, including: the current utilization rate of computing power resources of edge nodes; Task queue status (calculation amount CPU_{task}, data amount $Data_{task}$ of pending tasks); Channel state (such as instantaneous signal-to-noise ratio SNR of Nakagami-m fading model). The state vector is represented as:

$$s(t)=[CPU_{node},CPU_{task},Data_{task},SNR]\in S \quad (10)$$

(2) Action space: After collecting status information, the scheduler will make decisions and send control action signals to the MEC server. The decisions of the scheduler include: task offloading decisions (local processing, edge node processing, or cloud offloading); Resource allocation ratio (such as the number of CPU cores allocated to tasks and bandwidth allocation). The action vector is represented as:

$$a_t \in A=\{Local,Edge_1,\cdots,Edge_N,Cloud\} \quad (11)$$

In the formula, a_t represents the decision action made by the scheduler at time t; A represents the action space, which is the set of all possible actions that the scheduler can take; $Local$ represents local processing action; $Edge_N$ represents the N-th edge node processing action; $Cloud$ represents the cloud uninstallation action.

(3) State transition probability: Given state s_t and action a_t, the probability of transitioning to the next state s_{t+1} is $P(s_{t+1}|s_t,a_t)$. In edge computing, state transition is affected by the randomness of task arrival, channel fading formula (2) and computing resource dynamics.

(4) Reward function: The reward function $R(t)$ represents the instantaneous reward obtained when selecting action a_t in state s_t. Measuring the immediate cost of an action, designed as the negative value of weighted overhead (minimization objective):

$$R(s_t, a_t) = -(\alpha L_c + \beta L_\tau + \gamma NC) \quad (12)$$

In the formula, the weight must satisfy $\alpha + \beta + \gamma = 1$. By minimizing the negative value of weighted overhead, it is actually maximizing the long-term performance of the system.

By solving MDP, the optimal task scheduling strategy is found to maximize the long-term expected reward (i.e. minimize weighted overhead). The calculation formula is as follows:

$$\pi^* = \arg\max_\pi E\left[\sum_{t=0}^{\infty} \Upsilon^t R(s_t, a_t)\right] \quad (13)$$

In the formula, π represents the scheduling strategy; π^* represents the optimal scheduling strategy; E represents expectation.

In conclusion, in the edge computing network computing resource scheduling strategy, the action space is defined when constructing the Markov decision process model. This action space covers different types of actions such as task offloading decisions and resource allocation ratios, exhibiting the characteristics of a mixed action space that simultaneously encompasses both discrete and continuous actions [24].

SAC typically requires the construction of a joint strategy network and value function for mixed action spaces, and the introduction of additional temperature parameter automatic adjustment mechanisms. This leads to a complex model structure, difficulty in hyperparameter tuning, and the need for joint optimization of discrete and continuous actions for each policy update. In large-scale edge computing networks, the state and action space is already huge. This complexity of SAC will significantly increase the computational overhead and convergence time. PPO is essentially an online strategy algorithm, and its exploration efficiency depends on the current strategy. In complex mixed action spaces, PPO may find it difficult to efficiently explore all possibilities of "discrete continuous" combinations and may easily fall into local optima. The HA2C algorithm is specially designed for the mixed action space, which can implement targeted processing for different types of actions. It effectively overcomes the shortcomings of traditional methods in the mixed action space that it is difficult to effectively explore the optimal strategy and balance the impact of different types of actions on system performance, so as to more accurately explore the optimal task scheduling strategy, minimize the weighted overhead, and meet the demand for low-cost scheduling of large-scale edge computing network computing resources [25]. Therefore, the HA2C algorithm is adopted to accurately explore the optimal scheduling strategy.

The workflow of the HA2C algorithm consists of two parts, namely the environment interaction part and the network update part. In the environment interaction section, the scheduler takes the environment state as input to the actor network, outputs a mixed action a_t, and obtains the next state s_{t+1} and current reward $R(t)$ after interacting with the environment. The network update section updates the actor network and critic network parameters based on the interaction parameters. Constructing the Structure Diagram of HA2C Reinforcement Learning Algorithm [26].

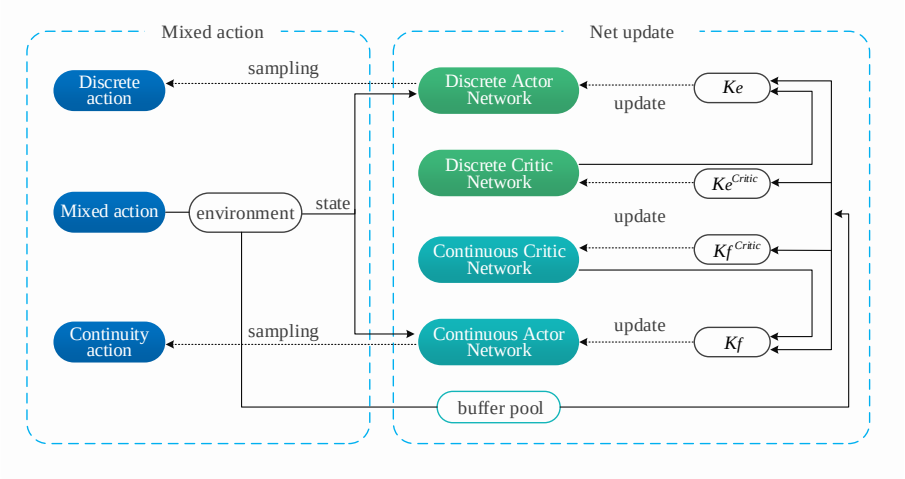


Figure 2: HA2C reinforcement learning algorithm

It is difficult to directly output continuous discrete mixed actions on the neural network in the environmental interaction part, so the mixed action a_t is divided into discrete action a_t^e and continuous action a_t^f, and discrete actor network, continuous actor network, and corresponding critic network are constructed. Discrete strategies, continuous strategies, and their corresponding value functions are parameterized as:

$\mu_e\left(a_t^e|s_t,\theta\right)$, $\mu_f\left(a_t^f|s_t,\omega\right)$, and $V_e\left(s_t;\varphi_e\right)$, $V_e\left(s_t;\varphi_e\right)$.

In practical scenarios with low computational resource overhead scheduling, difficulty in obtaining real-time rewards in asynchronous environments is a common problem. To address this issue, an experience replay mechanism has been introduced. This mechanism can store and transfer tuple $\rho_t=\left\{s_t,a_t,R\left(t\right),s_{t+1}\right\}$ to a buffer pool for subsequent network updates.

In the network update section, the HA2C algorithm is an improvement of the A2C algorithm, and the two neural network update methods are similar. The advantage actor critic (A2C) measures the selected actions compared to the average actions through the advantage function. The advantage function of the discrete network in the HA2C algorithm is defined as:

$$A_e\left(s_t,a_t\right)\approx R\left(t\right)+\Upsilon V\left(s_{t+1}|s_t,a_t\right)-V\left(s_t\right) \quad (14)$$

In the formula, Υ represents the discount factor.

In the HA2C algorithm, actions generated by discrete and continuous networks are combined into mixed actions to interact with the environment and receive environmental rewards [27]. For discrete actor networks, the scheduler randomly samples task scheduling actions from the Categorical distribution, and the loss function of the discrete actor network is:

$$K_e=-\left\{\ln\mu_e\left(a_t^e|s_t;\theta\right)A_e\left(s_t,a_t;\varphi\right)+\eta_e H_e\left(\mu_e\left(s_t;\theta\right)\right)\right\} \quad (15)$$

In the formula, η_e represents the entropy coefficient, which is used to control the regularization strength of entropy; $H_e\left(\mu_e\left(s_t;\theta\right)\right)$ encourages the exploration of the scheduler to increase the entropy of the discrete strategy $\mu_e\left(s_t;\theta\right)$, preventing premature convergence to suboptimal strategies during training. The detailed representation of parameter updates for discrete actor networks is:

$$\theta\leftarrow\theta+\frac{\partial\ln\mu_e\left(a_t^e|s_t;\theta\right)}{\partial\theta}A_e\left(s_t,a_t;\varphi\right)+\eta_e\frac{\partial H_e\left(\mu_e\left(s_t;\theta\right)\right)}{\partial\theta} \quad (16)$$

Due to the asynchronous nature of task scheduling and resource allocation, even if all tasks have been scheduled, MEC servers still cache some tasks that require custom adjustment of computing resource allocation based on their status. In the context of low-cost scheduling of computing resources, this custom adjustment is particularly important to ensure that resources are utilized more reasonably and efficiently. Therefore, in order to assist in the subsequent allocation of computing resources, rewards are designed for both discrete and continuous actions, that is, there is a critic network for each. The loss functions of discrete critic networks are:

$$K_e^{Critic}=\left(R\left(t\right)+\Upsilon V_e\left(s_{t+1};\varphi\right)-V_e\left(s_t\right)\right)^2 \quad (17)$$

The parameter update formula for discrete Critic networks is formulated as:

$$\varphi\leftarrow\varphi+\left(R\left(t\right)+\Upsilon V_e\left(s_{t+1};\varphi\right)-V_e\left(s_t\right)\right)\frac{\partial V_e\left(s_t;\varphi\right)}{\partial\varphi} \quad (18)$$

The scheduler randomly samples the MEC server frequency allocation action for the number of sensors from the Gaussian distribution. Combined with servers and sensors in edge computing, the system can obtain new parameters in time to achieve scheduling optimization. By updating the action frequency of sensors, the process of real-time updating parameters is optimized to reduce the overhead, and finally the low-cost scheduling of large-scale edge computing network computing resources is realized. Build a low-cost scheduling flowchart for computing resources.

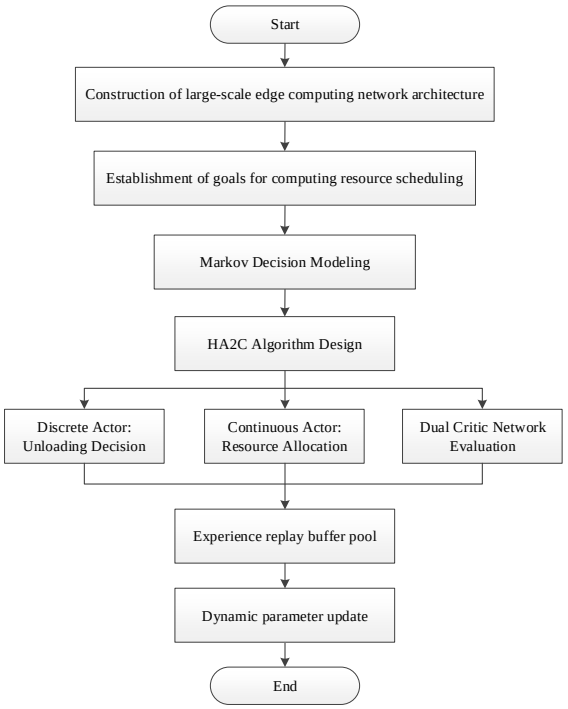


Figure 3: Flow chart of low-cost scheduling of computing resources

In summary, the pseudocode for implementing the HA2C algorithm is as follows:

```
import torch
from torch.optim import Adam

class HA2C:
    def __init__(self, state_dim, discrete_action_dim, continuous_action_dim):
        # Core network architecture
        self.discrete_policy = DiscretePolicyNetwork(state_dim, discrete_action_dim)
        self.continuous_policy = ContinuousPolicyNetwork(state_dim, continuous_action_dim)
        self.critic = CriticNetwork(state_dim)  # Simplified to single critic

        # Combined optimizer
        self.optimizer = Adam([
            {'params': self.discrete_policy.parameters()},
            {'params': self.continuous_policy.parameters()},
            {'params': self.critic.parameters()}
        ], lr=0.0005)

        # Key hyperparameters
        self.gamma = 0.99
        self.batch_size = 256
    def collect_trajectories(self, env, n_episodes):
        for _ in range(n_episodes):
            state = env.reset()
            while True:
                # Hybrid action generation
                d_action, _ = self.discrete_policy.sample(state)
                c_action, _ = self.continuous_policy.sample(state)
                next_state, reward, done = env.step((d_action, c_action))

                # Store transition
                self.replay_buffer.push(state, (d_action, c_action), reward, next_state, done)
                if done: break
                state = next_state
    def update(self):
        if len(self.replay_buffer) < self.batch_size: return

        states, actions, rewards, next_states, dones = self.replay_buffer.sample(self.batch_size)

        # Value function update
        with torch.no_grad():
            target_v = rewards + self.gamma * (1 - dones) * self.critic(next_states)
        critic_loss = torch.nn.functional.mse_loss(self.critic(states), target_v)

        # Policy update
        _, d_logprob = self.discrete_policy.evaluate(states, actions[0])
        _, c_logprob = self.continuous_policy.evaluate(states, actions[1])
        policy_loss = -(d_logprob + c_logprob) * (target_v - self.critic(states)).detach()

        # Combined optimization
        self.optimizer.zero_grad()
        (critic_loss + policy_loss.mean()).backward()
        self.optimizer.step()
```

2 Experimental analysis

2.1 Experimental setup

To verify the effectiveness of the proposed method, set the number, type, resource requirements, deadline and other parameters of tasks to simulate the actual task scenarios in large-scale edge computing networks. Generate 10000 scheduling tasks, all divided into ten groups. Each task has random resource requirements and deadlines, and task types include compute intensive, data intensive, and hybrid tasks to simulate the diversity of tasks in practical application scenarios. This study conducted experiments using a custom discrete event simulator built on Python and SimPy libraries. Build a large-scale edge computing network structure diagram.

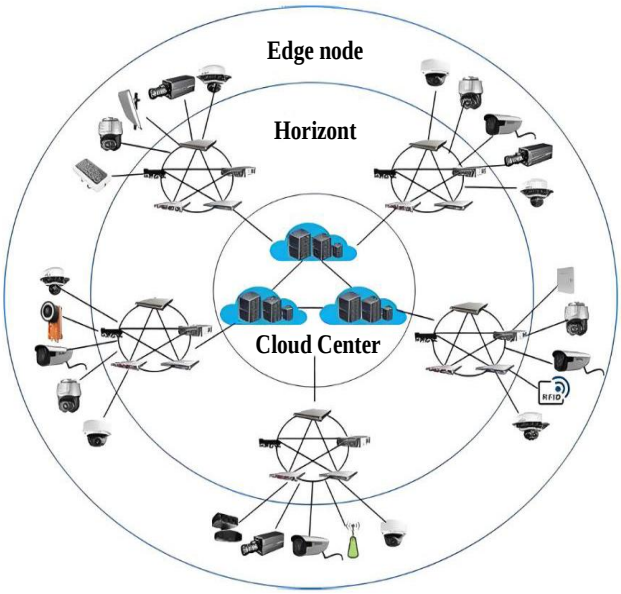


Figure 4: Large scale edge computing network structure

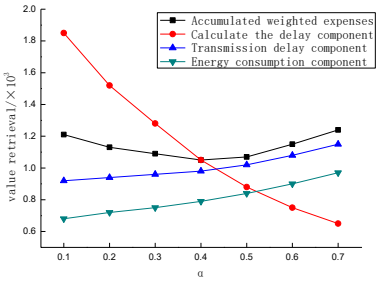
This network employs a three-layer "end-edge-cloud" architecture. The terminal layer, at the bottom, comprises devices like smartphones and sensor modules that collect and upload user and event data to the edge layer. The edge layer, situated between the terminal and cloud layers, consists of edge nodes such as gateways and set-top boxes with storage and computing capabilities; it localizes and processes terminal data, uploads it to the cloud as needed, reducing network bandwidth pressure and cloud computing center load. The cloud center, at the top, utilizes servers with high computing and storage capabilities in a centralized model to store and analyze massive data from the lower layers, providing comprehensive decision-making services for the system.

For the convenience of data calculation, construct an experimental environment table.

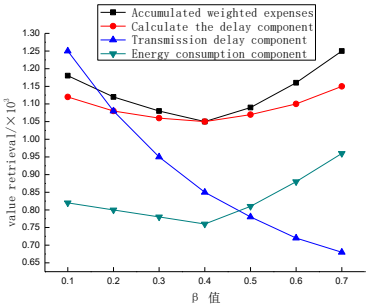
Table 1: Experimental environment

Project	Configuration
System	WIN 10
CPU	Intel Pentium Dual-Core 2.4GHZ
Programming Language	Python 3.8
Graphics card	RTX 3090
Monitor	Asus VG27AQL3A

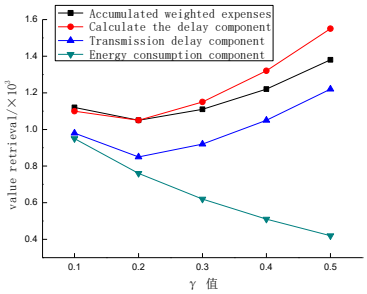
Among the parameters of the proposed method, the reward function weights α, β, and γ directly determine the optimization direction of the scheduling strategy. To evaluate the sensitivity of the proposed scheduling strategy to these hyperparameters and gain a deeper understanding of the trade-off between computational delay, transmission delay, and energy consumption, a systematic sensitivity analysis experiment was conducted, and a reward function weight sensitivity result graph was constructed.



(a) Sensitivity analysis data of delay weight α



(b) Sensitivity analysis data of transmission delay weight (β)



(c) Sensitivity analysis data of energy consumption weight (γ)

Figure 5: Sensitivity results of reward function weight

As α increases (with greater emphasis on reducing computational latency), the average computational latency of the system significantly decreases. At this point, the strategy tends to offload tasks to high-performance (but more energy consuming) edge nodes or the cloud, which may overlook energy efficiency optimization and cause fluctuations in transmission latency. The cumulative weighted cost reaches its lowest point near α=0.4, indicating that the benchmark weight is close to the local optimal solution; Both too high and too low alpha will increase the total cost, and the strategy is sensitive to extreme weights but performs stably within a reasonable range. Increasing β can effectively reduce the average transmission delay of the system. The strategy prioritizes selecting links with good channel conditions and high upload rates, but may sacrifice computational delay and increase energy consumption. The cumulative weighted overhead also reaches its minimum value around β=0.4. Increasing the gamma energy suppresses the total energy consumption of the system, and the strategy tends to use nodes with higher energy efficiency, even leaving some tasks for local processing. But this' energy-saving mode 'will lead to a significant increase in computation and transmission latency. When the gamma is too high (such as 0.5), the energy consumption is the lowest but the latency skyrockets, and the cumulative weighted overhead increases significantly.

During the training phase, the model only deals with two types of tasks: computationally intensive and data intensive. In the testing phase, introduce a new type of task that the model has never seen before: delay sensitive tasks. This type of task has extremely strict requirements for latency, but its computational and data volumes are moderate.

The test set consists of a mixture of 80% known tasks (40% compute intensive and 40% data intensive) and 20% delay sensitive tasks. Test the generalization ability of the proposed method in a mixed task flow containing this new task type, and construct a performance result table for unknown task types based on the test results.

Table 2: Performance comparison of unknown task types before and after the application of the method in this article.

	Known task type overhead	Includes unknown task type overhead	Degree of performance degradation
After application	1.05×10^3	1.22×10^3	+16.2%
Before application	1.87×10^3	2.45×10^3	+31.0%

When faced with challenges of unknown task types, the performance of all methods deteriorates, but the degradation degree of our method is the smallest. This indicates that HA2C learns general scheduling principles through the MDP framework, rather than memorizing patterns for specific task types. Therefore, when encountering delay sensitive tasks with novel deadline constraints, it can quickly adjust strategies based on its critic network's accurate evaluation of the system state, prioritize resource allocation to these more urgent tasks, and thus obtain stronger generalization ability.

2.2 Experimental results and analysis

2.2.1 Resource utilization rate

The method in this paper, the method in literature [5] and the method in literature [6] are used to study the low overhead scheduling of large-scale edge computing network computing resources, and the utilization efficiency of edge device resources is compared and analyzed. Construct a comparison chart of low-cost scheduling resource utilization rates for computing power resources.

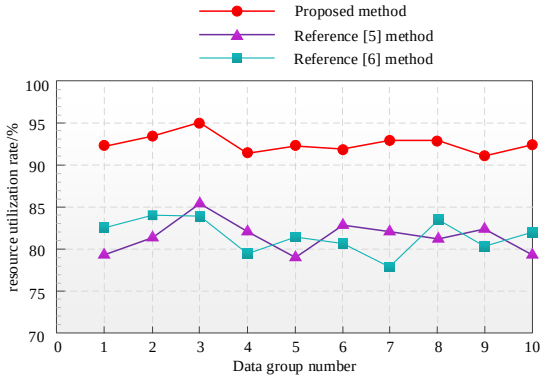


Figure 6: Comparison of low-cost scheduling resource utilization for computing power resources

It can be seen from Figure 6 that the resource utilization rate of low-cost scheduling of large-scale edge computing network computing resources in the literature [5] method is between 79% and 86%; The resource utilization ratio of low overhead scheduling of large-scale edge computing network computing resources in literature [6] is between 78% and 84%; However, the resource utilization ratio of the low overhead scheduling of large-scale edge computing network computing resources in this method is between 91% and 95%, and the resource utilization ratio is always above 90%. This is because the method proposed in this paper sets multiple objectives and evaluates the scheduling performance comprehensively through cumulative weighted overhead, so that the scheduling strategy needs to consider the impact of each objective from a global perspective when formulating. Avoiding the problem of performance degradation of other goals due to excessive bias towards one goal, it can find a balance point between multiple goals, achieve global optimal or near global optimal scheduling effects, and fully utilize the computing potential of edge devices.

To verify the statistical significance of the results, each experiment was independently repeated 30 times, and the mean, 95% confidence interval, and standard deviation were calculated. Use paired t-test (significance level α =0.05) to compare the performance differences between our method and the comparison method. Construct resource utilization statistics for three methods.

Table 3: Statistical results of resource utilization rates for three methods

Method	Mean	Standard deviation	Confidence interval
Proposed Method	93.2	1.05	[92.8, 93.6]
Reference [5] method	82.4	2.31	[81.5, 83.3]
Reference [6] method	80.9	2.67	[79.9, 81.9]

From Table 3, it can be seen that our method achieved an average resource utilization rate of 93.2%, significantly higher than the 82.4% of the method in reference [5] and the 80.9% of the method in reference [6], with an absolute improvement of over 10 percentage points. The standard deviation of this method (1.05) is much smaller than the two comparison methods. This indicates that the method proposed in this paper exhibits smaller fluctuations in different experimental rounds, and has stronger robustness and stability. The 95% confidence intervals for the method presented in this article are [92.8, 93.6], while the confidence intervals for the two baseline methods are [81.5, 83.3] and [79.9, 81.9], respectively. The three confidence intervals do not overlap at all, which proves the significance of the differences from a descriptive statistical perspective.

To conduct rigorous statistical inference, construct a paired t-test result table.

Table 4: Paired t-test results

Control group	t	p
This article's method vs the method in reference [5]	29.34	3.2×10^{-26}
This article's method vs the method in reference [6]	31.67	8.7×10^{-28}

Note: The significance level α is set to 0.05.

From Table 4, it can be seen that the obtained p-value is much smaller than the significance level α =0.05, which provides decisive evidence to reject the null hypothesis of "no difference in performance between the two methods". Therefore, it can be confirmed that the improvement in resource utilization efficiency of the method proposed in this article is not caused by random fluctuations, but rather an essential improvement with high statistical significance.

2.2.2 Response time

The method in this paper, the method in literature [5] and the method in literature [6] are used to study the low-cost scheduling of computing resources in large-scale edge computing networks, and the response time of resource scheduling is compared and analyzed. Build a response time comparison table for low-cost scheduling of computing resources.

Table 5: Comparison of response time for low-cost scheduling of computing resources

Data group number	Resource scheduling response time / ms		
	Proposed method/	Reference [5] method	Reference [6] method
01	3.22	11.53	12.47
02	2.56	12.26	16.87
03	2.57	12.58	18.38
04	2.87	11.82	16.17
05	3.58	11.42	15.85
06	2.92	11.87	18.31
07	2.73	10.24	12.73
08	2.52	12.33	13.52
09	3.61	13.23	17.17
10	2.85	11.86	18.82

Table 5 shows: literature [5] method has response times of 10.24ms - 13.23ms for low-cost scheduling in large-scale edge computing; [6] method ranges from 12.47ms - 18.82ms. Our method, however, achieves 2.52ms - 3.61ms, significantly lower and always under 4ms, indicating high scheduling efficiency. This is because our MDP modeling accurately captures dynamic edge environment characteristics via state, action spaces, and reward functions, enabling real-time status reflection, comprehensive operation coverage, and timely strategy adjustment for faster response.

2.2.3 Energy efficiency decline rate

To objectively evaluate the performance of the methods proposed in this article, literature [5], and literature [6] in terms of energy efficiency, the average task energy consumption, which is the average energy consumed by the system to process a single task, is used as the evaluation index to construct an energy efficiency comparison analysis chart, which intuitively reflects the energy utilization efficiency of different scheduling strategies when completing the same workload.

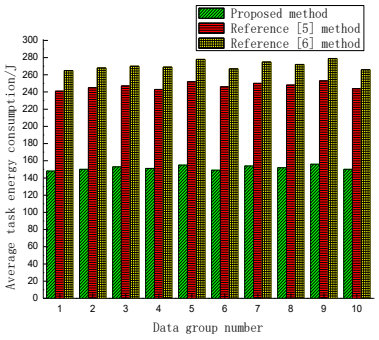


Figure 7: Comparative analysis results of energy efficiency

From Figure 7, it can be seen that in all 10 independent experiments, the HA2C scheduling strategy proposed in this paper achieved the lowest average task energy consumption, fluctuating within the range of 148-156 J with an average of 152 J. This is mainly attributed to the fine optimization ability of the HA2C algorithm in the mixed action space. Through a continuous Actor network, the algorithm can dynamically allocate "just right" computing resources based on the real-time computing requirements of the task and the current system state, avoiding the common over configuration or underconfiguration of resources in traditional methods. It shows that the scheduling strategy based on HA2C proposed in this paper can significantly and stably reduce the average energy consumption of system processing tasks through accurate, dynamic, and globally optimized resource allocation and task unloading decisions, so as to achieve a truly high-efficient and low-cost scheduling in large-scale edge computing networks.

3.2.4 Ablation experiment

To further explore the contribution of each core component in the HA2C algorithm to overall performance, three ablation variants were set up: no empirical replay mechanism, no dual critic network, and no mixed action processing. In the scenario of 10000 tasks, the performance of the complete HA2C algorithm was compared with the three ablation variants mentioned above, and a result table was constructed based on the cumulative weighted cost as the measurement standard.

Table 6: Comparison of ablation experimental performance

	Accumulated weighted expenses
The proposed method	1.05×10^3
Inexperienced replay	1.38×10^3
Unparalleled critic	1.29×10^3
No mixed action processing	1.67×10^3

The ablation experiment in Table 6 shows that mixed action processing is the fundamental architecture for achieving low-cost scheduling and solving the core problem; The experience replay mechanism is a key component to ensure efficient and stable learning of algorithms; The dual critic network provides precise guidance for collaborative optimization of mixed actions. These three components work together and are indispensable, which constitutes the fundamental reason why HA2C algorithm achieves significant performance gains in large-scale edge computing networks.

3.2.5 Scalability assessment

To verify the effectiveness and scalability of the proposed method, 10000 scheduling tasks were generated from the basic scenario and divided into ten groups. To conduct comprehensive load stress testing on the proposed method, two additional scenarios were set up: 50000 tasks (large-scale) and 100000 tasks (ultra large scale), all of which randomly generated corresponding resource requirements and deadlines. Task types include computation intensive, data intensive, and hybrid tasks to simulate the diversity of tasks in practical application scenarios. Thus, construct a weighted cost result table for the system under different task scales.

Table 7: Weighted cost results

Method	Task Scale		
	10,000	50,000	100,000
Proposed Method	1.05×10^3	5.32×10^3	10.81×10^3
Reference [5] method	1.87×10^3	9.91×10^3	20.76×10^3
Reference [6] method	2.14×10^3	11.52×10^3	24.13×10^3

From Table 7, it can be seen that in the large-scale scenarios of 50000 and 100000 tasks, the cumulative weighted overhead increases with the increase of task volume, which is an inevitable trend for all methods. However, the growth rate of the overhead of our method is much lower than that of the comparative method. When the task size increased from 10000 to 100000 (a tenfold increase), the cumulative weighted overhead of our method increased by about 10.3 times, showing an almost ideal linear growth trend. The cost of the methods in references [5] and [6] increased by approximately 11.1 times and 11.3 times, respectively, indicating a faster rate of cost growth. This almost linear low growth rate proves that the method proposed in this paper has excellent scalability and can efficiently cope with sharp increases in load. This is because the proposed method accurately collaborates to handle "task offloading" and "resource allocation" through separate discrete and continuous Actor networks. This decoupling architecture can more effectively find scheduling solutions that are close to the global optimum in a large mixed action space when facing large-scale tasks, avoiding the decline in decision quality caused by action space explosion. This is the main reason why baseline methods have faster cost growth when the load increases.

4 Conclusion

In order to solve the problems of low resource utilization rate of edge nodes and high overhead of dynamic task scheduling, this paper constructs an algorithm based on MDP and HA2C to study the low overhead scheduling method of large-scale edge computing network computing resources.

(1) By constructing a three-layer architecture of "end edge cloud" to optimize data flow and load distribution, combined with a scheduling target balancing strategy of cumulative weighted overhead, and utilizing MDP modeling dynamic characteristics and HA2C algorithm to handle mixed action space, efficient and low-cost scheduling of computing resources has been successfully achieved.

(2) The experimental results show that this method significantly shortens the response time of resource scheduling, improves the efficiency of resource

scheduling, maintains the resource utilization rate at more than 90%, effectively reduces energy consumption, and maintains a high scheduling accuracy, which provides an effective solution for resource optimization management of large-scale edge computing networks.

Funding

Key Scientific Research Project of Henan Province Universities in 2025; Project Name: Research on the Transformation and Upgrading Path of the New Quality Education Ecosystem in Universities in the Artificial Intelligence Era" by the Henan Provincial Department of Education; Project Number: 25B520047.

References

[1] Zhang F.F., Ge J.D., Li Z.J., et al. (2023). Cooperative Computation Offloading and Dynamic Task Scheduling in Edge Computing. *Journal of Software*, 34 (12): 5737-5756. http://dx.doi.org/10.13328/j.cnki.jos.006797

[2] Cang Y., Chen M., Pan Y., et al. (2024). Joint User Scheduling and Computing Resource Allocation Optimization in Asynchronous Mobile Edge Computing Networks. *IEEE Transactions on Communications*, 72 (6): 3378-3392. https://doi.org/10.1109/TCOMM.2024.3358237

[3] Wang Y., Lin X., Lou Z.L., et al. (2024). Uplink resource coordinated scheduling algorithm for edge-oriented optical computing power networks. *Optical Communication Technology*, 48(3), 45-51. https://doi.org/10.13921/j.cnki.issn1002-5561.2024.03.009.

[4] Xie Y.Y. and Chen L.S. (2025). Multi-objective Task Scheduling Method Based on Edge Computing. *Journal of University of Jinan (Science and Technology)*, 39 (01): 117-122. https://doi.org/10.13349/j.cnki.jdxbn.20241118.001

[5] Liu Y.P., Zhu Y.J., Bin Y.R., et al. (2022). Review of Research on Computing-Intensive Task Scheduling in Edge Environments. *Computer Engineering and Applications*, 58 (20): 28-42. https://doi.org/10.3778/j.issn.1002-8331.2202-0243

[6] Sheng Y., Zhu Z.W., Zhu C.Y., et al. (2023). Research on multi-objective edge task scheduling based on deep reinforcement learning. *Electronic Measurement Technology*, 46 (08): 74-81. https://doi.org/10.19651/j.cnki.emt.2210891.

[7] Hu X.M., Chen Z.F. and Li M. (2022). Level-Based Learning Swarm Optimization Algorithm for Resource Scheduling in Edge Computing. *Computer Engineering and Applications*, 58 (24): 107-115. https://doi.org/10.3778/j.issn.1002-8331.2110-0060

[8] Li C, Yu Z, Li X, et al. (2023). Low-latency AP handover protocol and heterogeneous resource scheduling in SDN-enabled edge computing. *Wireless networks*, 29(5): 2171-2187. https://doi.org/10.1007/s11276-023-03302-y

[9] Li J., Fan T.F., Gao H.L., et al. (2025). Research on Multi-core Task Offload Scheduling and Resource Adaptation in Edge Computing Networks. *Ordnance Industry Automation*, 44 (03): 29-34.

[10] Han D., Chen W. and Fang Y. (2020). Joint Channel and Queue Aware Scheduling for Latency Sensitive Mobile Edge Computing With Power Constraints. *IEEE Transactions on Wireless Communications*, 19(6): 3938-3951. https://doi.org/10.1109/TWC.2020.2979136

[11] Yamanaka H., Teranishi Y. and Kawai E. (2021). Service Migration Scheduling with Bandwidth Limitation against Crowd Mobility in Edge Computing Environments. *IEICE Transactions on Communications*, E104.B(3): 240-250. https://doi.org/10.1587/transcom.2020NVP0003

[12] Liu M. and Gong W. (2021). Resource Allocation of IoT Edge Computing Based on Joint Decision Model. *Computer Simulation*, 38 (12): 299-303.

[13] Cheng Y., Cao Z., Zhang X., et al. (2024). Multi objective dynamic task scheduling optimization algorithm based on deep reinforcement learning. *Journal of Supercomputing*, 80(5): 6917-6945. https://doi.org/10.1007/s11227-023-05714-1

[14] Wang Z, Wang G, Jin X, et al. (2022). Caching-based task scheduling for edge computing in intelligent manufacturing. *Journal of supercomputing*, 78 (4):5095-5117. https://doi.org/10.1007/s11227-021-04071-1

[15] Cha M.X., Zhu Y.J., Zhu L., et al. (2021). Resource scheduling algorithm for real-time stream data processing in edge computing. *Journal of Computer Applications*, 41 (S1): 142-148.

[16] Du R., Wang J. and Gao Y. (2024). Computing offloading and resource scheduling based on DDPG in ultra-dense edge computing networks. *Journal of supercomputing*, 80 (8):10275-10300. https://doi.org/10.1007/s11227-023-05816-w

[17] Qin B., Lei Q.Y. and Wang X. (2024). DGCQN: a RL and GCN combined method for DAG scheduling in edge computing. *The Journal of Supercomputing*, 80(13): 18464-18491. https://doi.org/10.1007/s11227-024-06140-7

[18] Zhang M., Liang S., He X.L., et al. (2020). Research on Engineering Project Resource Scheduling System Based on BIM and edge computing. *Construction Economics*, 41 (S1): 171-174.

https://doi.org/10.14181/j.cnki.1002-851x.2020S1171.

[19] Song S, Ma S, Zhao J, et al. (2022). Cost-efficient multi-service task offloading scheduling for mobile edge computing. *Applied Intelligence*, 52(4): 4028-4040. https://doi.org/10.1007/s10489-021-02549-2

[20] He H, Shan H, Huang A, et al. (2020). Edge-Aided Computing and Transmission Scheduling for LTE-U-Enabled IoT. *IEEE transactions on wireless communications*, 19(12):7881-7896. https://doi.org/10.1109/TWC.2020.3017207

[21] Ren J, Hou T, Wang H, et al. (2024). Collaborative task offloading and resource scheduling framework for heterogeneous edge computing. *Wireless Networks*, 30(9): 3897-3909. https://doi.org/10.1007/s11276-021-02768-y

[22] Wang L., Wu C.G. and Fan W.H. (2021). A Survey of Edge Computing Resource Allocation and Task Scheduling Optimization. *Journal of System Simulation*, 33 (03): 509-520. https://doi.org/10.16182/j.issn1004731x.joss.20-0584

[23] Diao X., Yang W., Yang L., et al. (2021). UAV-Relaying-Assisted Multi-Access Edge Computing with Multi-Antenna Base Station: Offloading and Scheduling Optimization. *IEEE Transactions on Vehicular Technology*, 70 (9): 9495-9509. https://doi.org/10.1109/TVT.2021.3101298

[24] Huang Y.F., Zeng W., Chen Z.Y., et al. (2024). Task scheduling method with Actor-Critic deep reinforcement learning in mobile edge computing. *Journal of Computer Applications*, 44 (S1): 150-155.

[25] Huang R. and Song G.L. (2024). Task Scheduling Optimization of Internet of Things Edge Computing Based on Offloading Strategy. *Journal of East China University of Science and Technology (Natural Science Edition)*, 50 (02): 264-273. https://doi.org/10.14135/j.cnki.1006-3080.20230320002

[26] Zhai Y.L., Bao T.H., Zhu L.H., et al. (2020). Toward Reinforcement-Learning-Based Service Deployment of 5G Mobile Edge Computing with Request-Aware Scheduling. I*EEE Wireless Communications*, 27(1): 84-91. https://doi.org/10.1109/MWC.001.1900298

[27] Liu F., Chen Z.Y., Ma K., et al. (2025). Real-time Low-power Scheduling Algorithm Based on Correlation for Task Partitioning in Edge Computing Environments. J*ournal of Chinese Computer Systems*, 46 (02): 289-296. https://doi.org/10.2009/j.cnki.21-1106/TP.2023-0459

Fault Diagnosis of Axle Box Bearings via MPA-Optimized VMD and Lightweight Wavelet CNN

Tingting Xing[1,2], Danhe Li[2], Jiuli Shen[1*], Jing Zhang[2]
[1]Huzhou Vocational and Technical College, Huzhou 313099, China
[2]Department of Automation Engineering, Tangshan Polytechnic University, Tangshan 063299, China
E-mail: sjlysu2016@sina.com
*Corresponding author

Keywords: variational mode decomposition, convolutional neural network; bullet train axle box bearing, fault diagnosis, signal processing, continuous wavelet transform

Received: May 7, 2025

This study presents a novel lightweight convolutional neural network model designed for efficient and accurate fault diagnosis of high-speed train axle box bearings using raw vibration signals. The model has been extensively validated on a CRRC dataset containing 12,000 samples, using parameterized Morlet wavelet kernels to directly generate high-resolution two-dimensional time-frequency representations from the input at the initial layer, significantly enhancing the capture of early fault transient pulses. By combining depthwise separable convolution and 1×1 convolution compression strategy, this model achieved a compact size of 2.6 million parameters, requiring only 0.38 GFLOPs. The system, combined with optimized MPA-VMD denoising preprocessing steps, exhibited excellent robustness, maintaining a Pearson coefficient of 0.92 even under strong noise conditions with a signal-to-noise ratio of 5dB. Comprehensive evaluation showed the model significantly outperformed the standard convolutional neural network and MobileNetV2, achieving a test accuracy of 97.08%, precision of 0.97, and F1 Score of 0.97, representing a 2.22% accuracy gain over MobileNetV2. The entire system provided low latency performance, which was crucial for edge deployment. Preprocessing took 38ms, model inference was 320ms, and the total latency was 358ms, completely within the real-time limit of 500ms. The deployment on the Jetson Nano platform further optimized latency to 2,90ms and memory usage to 5.2MB. These quantitative results confirm the high precision, high efficiency, and practical feasibility of the model as a robust edge intelligent diagnostic system for railway bearing health monitoring.

Povzetek: Razviti model učinkovito zaznava napake ležajev vlakov neposredno iz vibracijskih signalov ter dosega visoko natančnost in robustnost tudi v močno hrupnih razmerah.

1 Introduction

As the core component of high-speed train, axle box bearing of bullet train is prone to complex failure such as micro-crack, lubrication failure and local spalling due to high frequency impact, variable load and temperature change. Its vibration signal presents strong noise interference and non-stationary characteristics due to multi-fault coupling. Although traditional time-domain analysis methods, such as root mean square value and peak factor, are simple in calculation, they are insensitive to early fault characteristics and difficult to achieve early warning of faults [1-2]. Most of the existing studies focus on single fault diagnosis, lack sensitivity to complex faults, and the model complexity is high, which is difficult to adapt to the computational resource limitation of vehicle equipment. In recent years, Variational Mode Decomposition (VMD) has been largely used in bearing fault diagnosis due to its advantages in signal denoising and time-frequency feature extraction, which has become a key technology for solving non-stationary and strong noise interference signal processing [3]. The core of VMD lies in transforming signals into the Intrinsic Mode Function (IMF) with finite bandwidth through adaptive decomposition, effectively suppressing mode aliasing and endpoint effects. Compared with traditional methods, VMD has high decomposition accuracy, strong noise resistance, and small endpoint effects, which can extract the fault characteristics of bearing vibration signals under complex working conditions [4]. In terms of fault feature recognition, Convolutional Neural Network (CNN) is widely used due to the powerful feature learning ability [5]. CNN can gradually extract local and global features from signals through multi-layer convolution and pooling operations, and construct high-level fault feature representations. Compared with traditional machine learning methods, CNN has automated feature extraction and strong model generalization ability, which can optimize the diagnosis accuracy and efficiency.

Wang et al. designed a hybrid model combining continuous wavelet transform and Long Short-Term Memory (LSTM) to optimize the accuracy and robustness of rolling bearing fault detection. The accuracy in fault detection reached 99.98%, significantly better than traditional CNN models, demonstrating excellent

performance in complex working conditions [6]. Palaniappan R proposed a prediction model based on Principal Component Analysis (PCA) and multi-classifier comparison to address the insufficient accuracy of traditional methods in predicting the remaining service life of roller bearings. The operation characteristics of the bearings were collected by customizing the test bench, and the key features were screened by PCA and input into the SVM classifier. The results showed that the classification accuracy of SVM reached 96.74%, verifying the superiority of SVM in bearing life prediction [7]. Zhou et al. proposed a new multi-objective sparse maximum mode decomposition method for fault diagnosis of axle box bearings in high-speed trains. This method constructs several finite impulse response filters by analyzing the Fourier spectrum of vibration signals, and adaptively adjusts the filter frequency range with the goal of maximizing the L2/L1 norm of the envelope spectrum, thus significantly improving the accuracy and adaptability of axle box bearing fault detection [8]. Kulevome et al. combined improved data augmentation and CNN to address the insufficient data. By optimizing the time-frequency domain signal processing parameters and combining two-stage regularized CNN, the accuracy of the model in cross domain testing reached 92.54%, providing reliable support for fault diagnosis in engineering scenarios [9]. To solve the reconstruction efficiency and accuracy bottlenecks of the convex optimization problem in compressed sensing of bearing vibration signals, Guo J proposed a balanced generalized custom near-point algorithm. The compressed sensing system was constructed by integrating the generalized custom near-point algorithm and the balanced augmented Lagrange method, combined with K-singular value decomposition. Experiments showed that this method significantly reduced the signal reconstruction error and calculation time [10].

Taibi et al. deigned induction motor bearings relying on improved VMD and composite multi-scale weighted permutation entropy. This method could reduce the noise and achieve high-precision fault classification through the Support Vector Machine (SVM) model, providing a new tool for induction motor bearing fault diagnosis [11]. Yi et al. predicted the remaining service life of rolling bearings relying on VMD and generative adversarial network. This method could identify the degradation state of bearings earlier and achieve high-precision remaining service life prediction, providing important reference for bearing health management [12]. Patil et al. combined VMD and Morlet wavelet filters to bearing fault diagnosis. This method could remove noise from vibration signals and achieve high-precision fault classification through CNN, demonstrating broad applicability [13]. Qiu et al. built a diagnosing method for crankshaft bearings of a rotary vector reducer under variable speed conditions. This method exhibited excellent diagnostic accuracy [14]. Jiang et al. proposed a fault detection method for aircraft engine rolling bearings relying on CNN-BiLSTM. The classification accuracy was better than that of the traditional LSTM, providing an efficient solution for fault diagnosis in aircraft engines [15]. Table 1 shows the comparison between the method proposed in this paper and existing literature.

Table 1: Comparison of the proposed method with existing literature.

Research purpose	Method	Result	Shortcomings	Reference
Optimize accuracy and robustness of rolling bearing fault detection	Hybrid model combining continuous wavelet transform and LSTM	99.98% fault detection accuracy, superior to traditional CNN in complex conditions	Not explicitly mentioned	Wang et al. [6]
Improve prediction accuracy for remaining service life of roller bearings	PCA + multi-classifier comparison (SVM classifier)	96.74% classification accuracy	Requires customized test bench for data collection	Palaniappan R [7]
Fault diagnosis of high-speed train axle box bearings	Multi-objective sparse maximum mode decomposition (FIR filters with adaptive frequency adjustment)	Significantly improved accuracy and adaptability	Computational complexity not addressed	Zhou et al. [8]
Address insufficient data in fault diagnosis	Improved data augmentation + two-stage regularized CNN	92.54% cross-domain testing accuracy	Limited generalization to extreme noise conditions	Kulevome et al. [9]
Solve reconstruction efficiency/accuracy in compressive sensing of bearing signals	Balanced generalized custom near-point algorithm + K-SVD	Reduced signal reconstruction error and calculation time	Application limited to specific sensing systems	Guo J [10]
Induction motor bearing fault diagnosis	Improved VMD + composite multi-scale weighted permutation entropy + SVM	High-precision fault classification	Dependent on SVM's kernel selection	Taibi et al. [11]
Predict remaining service life of rolling bearings	VMD + generative adversarial network	Early degradation identification and high-precision life prediction	Requires long-term degradation data	Yi et al. [12]
Bearing fault diagnosis under noise	VMD + Morlet wavelet filters + CNN	High-precision fault classification	Standard CNN has high computational cost	Patil et al. [13]
Diagnose crankshaft bearings under variable speed	Not specified in source	Excellent diagnostic accuracy	Method details not disclosed	Qiu et al. [14]

Fault detection for aircraft engine bearings	CNN-BiLSTM network	Better accuracy than traditional LSTM	Not optimized for edge deployment	Jiang et al. [15]
Early fault diagnosis for high-speed train axle box bearings under complex conditions	MPA-optimized VMD + Lightweight CWCNN	89.32% accuracy, 88.66s training time, 320ms inference latency	-	The method proposed

In summary, existing research models perform well under single operating conditions or constant speeds, but their performance significantly declines in complex industrial environments with variable speeds, loads, or multiple noise disturbances. Existing methods often focus on obvious fault characteristics and lack sensitivity to early weak defects such as microcracks and local lubrication failures, resulting in delayed warning. Model complexity and computational cost are also important issues in current research. Although hybrid models have improved accuracy, their large number of parameters and high training costs make it difficult to deploy to edge devices or real-time monitoring systems. This study aims to address the following core questions: (1) Can adaptive VMD parameter tuning using Marine Predators Algorithm (MPA) significantly improve fault signal decomposition accuracy under high-noise environments (e.g., Signal-to-Noise Ratio (SNR)≤5 dB)? (2) Can the proposed lightweight CWCNN achieve diagnostic accuracy comparable to full-scale CNN models while meeting real-time constraints (e.g., inference latency < 500 ms and memory usage <10 MB)? Therefore, the research innovatively proposes an improved VMD algorithm with parameter adaptation, breaking through the dependence of traditional decomposition methods on prior knowledge; A fault diagnosis method for EMU axle box bearings based on lightweight CNN (CNN based on Continuous Wavelet, CWCNN) is proposed. Depth-separable convolution is adopted to replace the standard convolution kernel, reducing the number of parameters while retaining the ability to extract multi-scale time-frequency features.

2 Methods and materials

A method for processing and diagnosing axle box bearing fault based on improved VMD and CWCNN is proposed. Firstly, an improved VMD algorithm is proposed, which improves the decomposition accuracy of VMD for axle box bearing fault signals by introducing an adaptive parameter selection strategy. Secondly, a lightweight CWCNN model is designed, which utilizes depthwise separable convolution and channel attention mechanism to reduce model complexity while improving feature extraction capability.

2.1 Improved VMD method for axle box bearing of bullet train

Vibration signals, as the core data source for axle box bearing fault diagnosis, are affected by environmental noise, sensor interference, and signal non-stationarity, which may submerge fault characteristics. However, traditional VMD suffers from modal aliasing and residual noise issues due to parameter sensitivity, especially in strong noise environments where performance significantly declines. Marine Predators Algorithm (MPA), as a new type of swarm intelligence optimization algorithm, has strong global search ability and fast convergence speed. A VMD method based on MPA optimization (MPA-VMD) is proposed, which dynamically matches the optimal mode number K with the penalty factor α to adaptively decompose and efficiently denoise vibration signals.

VMD is a non-recursive signal decomposition strategy, whose core idea is to adaptively decompose signals into several IMFs with sparsity and narrowband characteristics by constructing constrained variational problems. Assuming that the original signal $x(t)$ is separated into K IMF components $\{u_k(t)\}_{k=1}^{K}$, each IMF is compactly distributed around its center frequency ω_k. The VMD is to minimize the bandwidth sum of all IMF components while ensuring that the sum of each IMF equals the original signal. The constrained variational problem is presented in equation (1).

$$\begin{cases} \min\limits_{\{u_k\},\{\omega_k\}} \left\{ \sum\limits_{k=1}^{K} \left\| \partial_t \left[(\delta(t)+\frac{j}{\pi t}) * u_k(t) \right] e^{-j\omega_k t} \right\|_2^2 \right\} \\ \text{s.t.} \quad \sum\limits_{k=1}^{K} u_k(t) = x(t) \end{cases} \tag{1}$$

In equation (1), $u_k(t)$ signifies the k -th IMF, representing the k -th decomposed component of the signal. $\delta(t)$ is the Dirac function used to construct analytical signals. $*$ is the convolution operator, representing the convolution operation of two functions. ∂_t represents the partial derivative of time. $e^{-j\omega_k t}$ is used to modulate IMF to baseband. $\| \|_2^2$ represents the square of the L2 norm, which quantifies the signal bandwidth. By introducing Lagrange multiplier $\lambda(t)$ and quadratic penalty factor α, the constrained problem is changed to an unconstrained optimization problem, as presented in equation (2).

$$\begin{aligned} L(\{u_k\},\{\omega_k\},\lambda) = \alpha \sum_{k=1}^{K} \left\| \partial_t \left[(\delta(t)+\frac{j}{\pi t}) * u_k(t) \right] e^{-j\omega_k t} \right\|_2^2 + \left\| x(t) - \sum_{k}^{K} u_k(t) \right\|_2^2 \\ + \left\langle \lambda(t), x(t) - \sum_{k}^{K} u_k(t) \right\rangle \end{aligned} \tag{2}$$

In equation (2), the penalty factor α controls the strength of bandwidth constraints, typically ranging from 100 to 5000. L is the Lagrange function, representing the objective function of the optimization problem. The study used the Alternating Direction Method of Multipliers (ADMM) to iteratively update each variable and gradually approach the optimal solution, as presented in equation (3).

$$\begin{cases} \hat{u}_k^{n+1}(\omega) = \dfrac{\hat{x}(\omega) - \sum_{i \neq k} \hat{u}_i(\omega) + \dfrac{\hat{\lambda}(\omega)}{2}}{1 + 2\alpha(\omega - \omega_k)^2} \\ \omega_k^{n+1} = \dfrac{\int_0^\infty \omega |\hat{u}_k^{n+1}(\omega)|^2 \, d\omega}{\int_0^\infty |\hat{u}_k^{n+1}(\omega)|^2 \, d\omega} \\ \hat{\lambda}^{n+1}(\omega) = \hat{\lambda}^n(\omega) + \tau\left(\hat{x}(\omega) - \sum_{k=1}^{K} \hat{u}_k^{n+1}(\omega)\right) \end{cases} \quad (3)$$

In equation (3), n signifies the iteration. τ represents the convergence parameter, which controls the multiplier update step size, usually taken as $0 < \tau < 1$. ω is a frequency variable, representing the frequency value in the Frequency Domain (FD). $\hat{\lambda}(\omega)$ signifies the FD form of Lagrange multiplier. MPA simulates the predation strategy of marine organisms, dividing search agents into three categories: predators, prey, and random individuals, balancing global exploration and local exploitation through three stages. In the initial stage (iteration number $t < T/3$), the predator moves at high speed to explore the global space, as shown in equation (4).

$$X_i^{t+1} = X_i^t + R \otimes (X_{\text{best}} - R \otimes X_i^t) \quad (4)$$

In equation (4), X_i^t represents the position of the i-th search agent at the t-th iteration. (K, α) represents the current parameter combination. X_{best} represents the currently found optimal parameter combination. R is a random vector based on Levi's flight. $\otimes$ represents element wise multiplication. In the mid-term stage (iteration number $T/3 \le t < 2T/3$), prey and predators move in coordination. The Brownian motion is introduced to enhance local search, as shown in equation (5).

$$X_i^{t+1} = X_i^t + 0.5\left[R_B \otimes (X_{best} - R_B \otimes X_i^t)\right] \quad (5)$$

In equation (5), R_B is a random vector of Brownian motion. In the later stage ($t \ge 2T/3$), the predator conducts low-speed fine search, as shown in equation (6).

$$X_i^{t+1} = X_{\text{best}} + 0.01R \otimes (U - L) \quad (6)$$

In equation (6), U and L signify the upper and lower limits of the parameters. In this stage, small-scale perturbations are used to avoid falling into local optima. The VMD parameter (K, α) is mapped to the two-dimensional search space of MPA, with the optimization objective of minimizing modal aliasing and noise residue. The individual position is $X_i = [K_i, \alpha_i]$, where $K_i \in [3, 10]$ and $\alpha_i \in [100, 5000]$. Ultimately, a multi-objective function that combines envelope entropy and frequency band overlap is obtained, as shown in equation (7).

$$F(K, \alpha) = 0.7 \cdot \sum_{k=1}^{K} E_k + 0.3 \cdot \sum_{i \neq j} \text{Overlap}(B_i, B_j) \quad (7)$$

In equation (7), E_k is the IMF envelope entropy, reflecting modal sparsity. $\text{Overlap}(B_i, B_j)$ is the overlapping area of the frequency band. Non-integer K is rounded to the nearest integer. The weighting coefficients in equation (7) are determined based on two principles: (1) Dominance of Impact Signatures: Envelope entropy quantifies the sparsity of fault-induced transient impacts, which are critical for early bearing defect detection. Its higher weight (0.7) prioritizes extracting pulse features submerged in noise [16]. (2) Complementarity of Frequency Separation: Gini Index of Squared Envelope (GISE) measures the spectral band overlap to suppress mode mixing. Its lower weight (0.3) provides auxiliary constraints while avoiding excessive punishment of legitimate frequency components [17]. $\text{Overlap}(B_i, B_j)$ is calculated using the spectral interval intersection method. For any two IMF components u_i and u_j, the frequency band is defined by their center frequency ω_k and bandwidth $\Delta\omega_k$, and the overlapping area of their frequency bands is achieved by calculating the intersection length between the interval B_i and B_j, as shown in equation (8).

$$\text{Overlap}(B_i, B_j) = \max\left(0, \min\left(\omega_i + \frac{\Delta\omega_i}{2}, \omega_j + \frac{\Delta\omega_j}{2}\right) - \max\left(\omega_i - \frac{\Delta\omega_i}{2}, \omega_j - \frac{\Delta\omega_j}{2}\right)\right) \quad (8)$$

The smaller the $\text{Overlap}(B_i, B_j)$, the more thorough the band separation and the lower the risk of mode aliasing. In addition, this study introduces GISE as a normalization measure for frequency band overlap, as shown in equation (9).

$$\text{GISE} = 1 - \frac{2\sum_{k=1}^{K}\sum_{l=k+1}^{K} \text{Overlap}(B_k, B_l)}{K(K-1)\max(\Delta\omega_1, \ldots, \Delta\omega_K)} \quad (9)$$

The value range of GISE is [0,1], with larger values indicating less frequency band overlap. Together with envelope entropy, it forms a multi-objective optimization function to balance the sparsity of impulse features and frequency band separation. The variational problem solving of VMD and the parameter optimization of MPA have a double-layer nested relationship: (1) Inner layer solving (VMD self-optimization): For any parameter combination (K, α), VMD iteratively updates the IMF component u_k and Lagrange multiplier λ through the ADMM algorithm (in equation (3)) until convergence, completing signal decomposition. (2) Outer layer optimization (MPA global search): MPA explores different combinations of (K, α) by updating the search agent positions (in equations (4) - (6)). Each iteration requires calling the inner layer VMD decomposition to calculate the objective function value (in equation (7)). Therefore, ADMM is an internal numerical solver of VMD, responsible for signal decomposition under a set of parameters. MPA is a global optimizer responsible for searching for optimal solutions across parameter spaces, and the two belong to different levels without any contradiction or substitution relationship.

The out of bounds parameter adopts the mirror bounce strategy. During signal preprocessing, the original signal $x(t)$ is symmetrically extended by 1/4 of

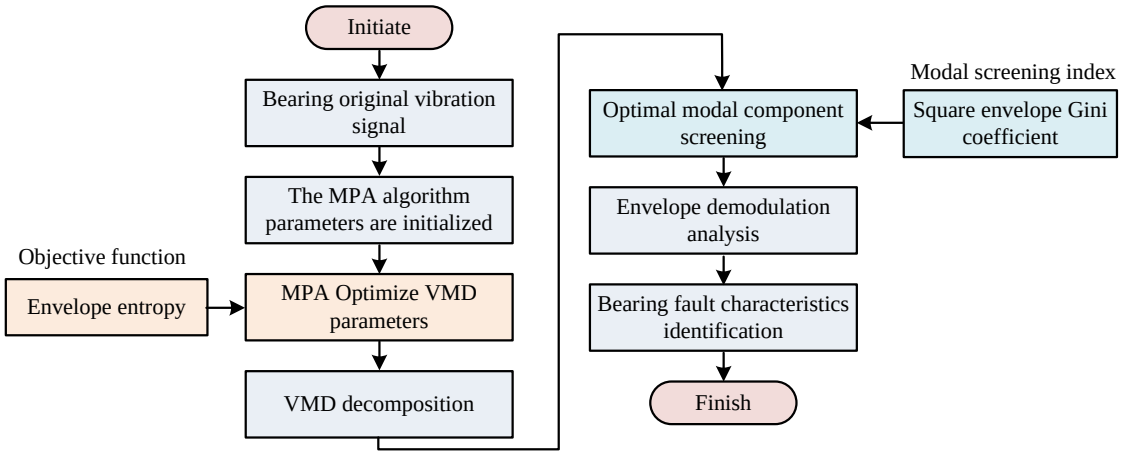


Figure 1: MPA-VMD bearing signal processing flow chart.

its length at both ends to suppress endpoint effects. A composite window function (Hanning window+Caesar window) is used to reduce spectral leakage, as shown in equation (10).

$$w(t) = 0.6 \cdot \mathrm{Hann}(t) + 0.4 \cdot \mathrm{Kaiser}(t, \xi = 4) \quad (10)$$

In equation (10), the sidelobe attenuation coefficient $\xi = 4$ of the Caesar window can balance the main lobe width and sidelobe suppression capability. The implementation flowchart of MPA-VMD method is shown in Figure 1.

Firstly, during the initialization phase, $N = 30$ individuals are randomly generated with an iteration count of $T = 50$. The population size and iteration times are determined through orthogonal experiments to ensure search efficiency [18]. In the fitness calculation, the VMD decomposition is performed on each parameter combination (K, α) and the objective function $F(K, \alpha)$ is calculated. To accelerate the calculation, fast Fourier transform is taken to parallelly process the FD updates of each IMF. During the stage determination and location update phase, the search strategy of MPA is switched according to the iteration progress to dynamically balance exploration and development. When the termination condition is met, the optimization process terminates. In the signal decomposition and reconstruction stage, VMD decomposition is performed using optimized parameters K_{opt} and α_{opt} to obtain the IMF set $\{u_k(t)\}$.

The IMF screening adopts a hierarchical joint standard. First, the Kurtosis value K_k of each IMF component is calculated, a kurtosis threshold $K_{th} = 3$ (kurtosis value close to Gaussian noise) is set, modes $K_k < K_{th}$ are removed, and components with significant impact features are retained. This step aims to filter out low kurtosis modes mainly dominated by environmental noise. The cross-correlation coefficient P_k between the pre-screened IMF components and the original signal $\hat{x}(t)$ is calculated, and a retention threshold $P_{th} = 0.15$ is set. Only $P_k > P_{th}$ modes are retained because they contain frequency components related to faults and have high correlation with the original signal. Based on this hierarchical mechanism, it not only avoids the interference of noise modes on fault characteristics, but also ensures the physical meaning of effective IMF. Finally, the filtered IMF is overlaid to obtain the denoised signal $\hat{x}(t) = \sum_{k \in S} u_k(t)$, where S is the set of effective IMF indices.

2.2 Bearing fault diagnosis based on lightweight CWCNN

On the basis of obtaining denoised vibration signals, how to accurately classify fault characteristics has become the core task of the next stage. Although deep learning models perform well in fault diagnosis, their high computational complexity makes it difficult to satisfy the real-time of industrial sites. In response to this contradiction, a lightweight CWCNN is designed, which utilizes depth separable convolution and channel attention mechanism for collaborative optimization. While retaining the ability to obtain multi-scale features, the computational complexity is lowered by an order of magnitude. Taking vibration signals as input, the end-to-end recognition of fault modes is achieved through hierarchical feature transformation. The convolutional layer uses multiple layers of one-dimensional convolution kernels for local feature extraction, as shown in equation (11).

$$x_j^{(l)} = f\left(\sum_{i=1}^{C_{in}} W_{ij}^{(l)} * x_i^{(l-1)} + b_j^{(l)} \right) \quad (11)$$

In equation (11), $x_j^{(l)}$ is the j-th output feature map of the l-th layer, representing the local features extracted by that layer. $f(\cdot)$ signifies a non-linear activation function used to optimize the non-linear expression ability of the model. C_{in} signifies the input channel, which is the quantity of feature maps in the previous layer. $W_{ij}^{(l)}$ signifies the one-dimensional convolution kernel weight matrix that connects the i-th input channel to the j-th in the l-th. $*$ represents a one-dimensional convolution operation used for sliding to extract local features of the input signal. $x_i^{(l-1)}$ signifies the i-th input feature map of

the $l-1$, which serves as the input for the current layer. $b_j^{(l)}$ signifies the bias term for the j-th output channel of the l-th, used to adjust the baseline of the output features.

Maximum pooling is inserted between convolutional layers, with a pooling window width of 3 and a stride of 2. High frequency noise is suppressed and translation invariance is enhanced through down-sampling. The Fully Connected Layer (FCL) flattens the high-dimensional features output by the terminal convolution and inputs them into the FCL to achieve the mapping from feature space to category space. The Softmax classifier calculates the probability distribution of fault categories through a normalized exponential function, as shown in equation (12).

$$p(y=c \mid x)=\frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}} \tag{12}$$

In equation (12), $p(y=c \mid x)$ signifies the predicted probability that sample x is fault category c. The logits value of category c corresponding to z_c signifies the original output of the FCL, reflecting the model's confidence in category c. C signifies the total fault categories (such as normal state, inner ring fault, outer ring fault, etc.). The study uses cross entropy loss L_{CE} to measure the difference between predicted probabilities and true labels, as shown in equation (13).

$$L_{CE}=-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(p_{i,c}) \tag{13}$$

In equation (13), N signifies the quantity of samples in the batch. $y_{i,c}$ signifies the true label of the i-th sample, encoded using one-hot encoding (it is 1 when belonging to category R. Otherwise, it is 0). In the task of axle box bearing fault diagnosis, Adaptive Convolutional Activation (ACON) can dynamically adjust the shape of the activation function by introducing a learnable parameter β, thereby better adapting to the feature extraction requirements. The expression for the ACON activation function is shown in equation (14).

$$f(x)=x\cdot\frac{1}{1+e^{-\beta x}} \tag{14}$$

In equation (14), β is a learnable parameter that controls the smoothness and nonlinearity of the activation function. When $\beta\to\infty$, ACON degenerates into ReLU. When $\beta=1$, ACON approximates the Swish function. Compared with hard saturation characteristics in ReLU, ACON's continuous differentiability alleviates the gradient vanishing problem.

Figure 2 displays the process of training vibration signals using a continuous wavelet CNN. The wavelet kernel convolutional layer can more effectively extract axle box bearing fault features by combining the time-frequency analysis of wavelet transform and the feature learning ability of deep learning.

The Morlet wavelet kernel has high resolution in the time-frequency domain, which is appropriate for extracting fault impact features, as expressed in equation (15).

$$\psi(t)=e^{-t^2/2}\cdot e^{j\omega_0 t} \tag{15}$$

In equation (15), ω_0 is the center frequency, and $\omega_0\geq 5$ is usually taken to satisfy the tolerance condition. $e^{-t^2/2}$ is the Gaussian envelope that controls the decay of the wavelet function. To enhance the time-frequency localization capability, an adjustment parameter m is introduced into the traditional Morlet wavelet, and its expression is reconstructed as equation (16).

$$\psi_m(t)=\left(1+m\cdot e^{-\beta t^2}\right)\cdot e^{-t^2/(2\sigma^2)}\cdot e^{j\omega_0 t} \tag{16}$$

In equation (16), σ signifies the standard deviation of the original Gaussian envelope. When $m>1$, the time window width is increased to improve frequency resolution. When $0<m<1$, the width of the time window is lowered and the time resolution is increased. The time-frequency spectrum after continuous wavelet transform processing is shown in Figure 3.

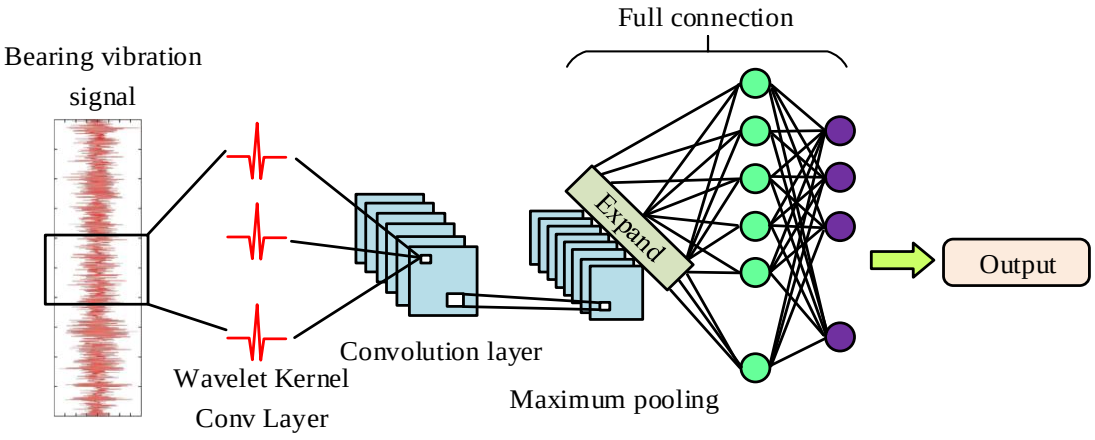


Figure 2: The process of training vibration signals by CWCNN.

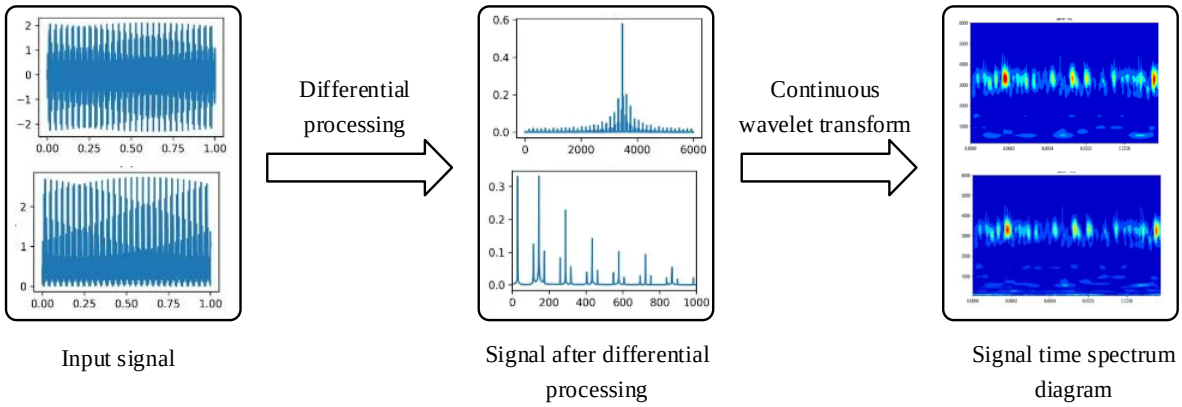


Figure 3: Time-frequency spectrum processed by continuous wavelet transform.

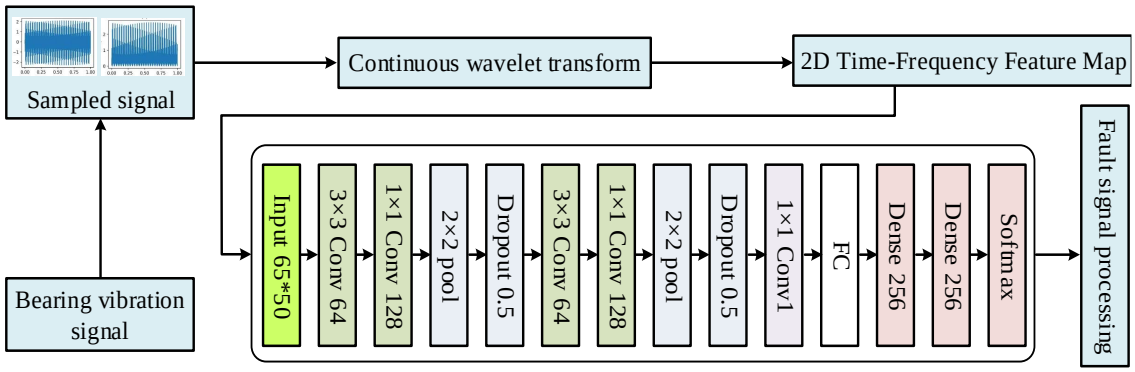


Figure 4: Lightweight CWCNN bearing fault signal processing model.

The convolution operation and wavelet transform have similarities in mathematical form, both of which is the inner product of two vectors in signal processing. Continuous wavelet transforms extracts time-frequency local features by calculating the inner product of wavelet functions and input signals. The mathematical expression is equation (17).

$$W(a,b)=\int_{-\infty}^{\infty}x(t)\cdot\psi(t)dt \tag{17}$$

In equation (17), $x(t)$ signifies the input signal. $W(a,b)$ signifies the wavelet coefficient, representing the time-frequency characteristics at the scale a and b positions. The axle box bearing fault signal processing model of lightweight CWCNN is shown in Figure 4.

The first layer of CWCNN is a continuous wavelet kernel convolution layer, which parameterizes the Morlet wavelet function into learnable 1D convolution kernels (such as center frequency and scale factor), and directly extracts 2D time-frequency feature maps from 1D vibration signals through convolution operations. This design avoids the limitations of traditional CWT as a fixed preprocessing method and realizes the end-to-end integration of time-frequency analysis and feature learning. In the motor axle box bearing fault diagnosis, deep learning models often encounter large model size, high resource utilization, and slow running speed. To optimize the proposed CNN-based deep learning model and reduce its parameter count, a 1×1 convolution is introduced before applying the Flatten operation. This strategy effectively reduces the input dimension of the FCL and achieves lightweighting. By adding 1×1 convolution, the input dimension of the FCL is lowered, which has a significant impact on reducing the number of FC layer parameters that occupy the majority of the total model parameters, thereby reducing computational complexity.

Dynamic feature weighting is achieved through the channel attention module. Its core idea is to automatically adjust the feature weights of different channels through learning, strengthen the frequency components related to faults (such as BPFI/BPFO and their harmonics), and suppress noise and irrelevant vibration interference. Firstly, for the feature map $\mathbf{X}\in\mathrm{R}^{C\times L}$ (where C is the number of channels and L is the signal length) output by the convolutional layer, the spatial dimension is compressed through global average pooling to obtain the channel statistics, as shown in equation (18).

$$z_c=\frac{1}{L}\sum_{i=1}^{L}X_c(i),\quad c=1,2,\ldots,C \tag{18}$$

In equation (18), z_c represents the global feature response of the CTH channel c . The dependency

relationship between channels is captured through a two-layer fully connected network (MLP). A nonlinear activation function is introduced to achieve weight modulation, as shown in equation (19).

$$\mathbf{s}=\sigma\left(W_2\delta\left(W_1\mathbf{z}\right)\right) \tag{19}$$

In equation (19), $W_1\in \mathrm{R}^{C/r\times C}$ and $W_2\in \mathrm{R}^{C\times C/r}$ are learnable weight matrices. r is the channel compression ratio. δ is the ReLU activation function. σ is the Sigmoid function, normalizing the weight to the interval [0,1]. The final output channel weight vector $\mathbf{s}=[s_1,s_2,\ldots,s_C]$ represents the importance score of each channel. The channel weights are multiplied back to the original feature map channel by channel to enhance the key fault feature, as shown in equation (20).

$$\hat{\mathbf{X}}_c(i)=s_c\cdot X_c(i),\quad \forall c,i \tag{20}$$

In equation (20), $\hat{\mathbf{X}}$ represents the weighted feature map. The channel weight s is jointly optimized with the entire network parameters through back-propagation, aiming to minimize the cross-entropy loss function L_{CE}, as shown in equation (21).

$$L_{\mathrm{CE}}=-\frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C_{\mathrm{class}}}y_{n,c}\log\hat{y}_{n,c} \tag{21}$$

In equation (21), N represents the batch size. C_{class} is the number of fault categories. $y_{n,c}$ is the real label, and $\hat{y}_{n,c}$ is the predicted probability of the model. The Adam optimizer (with a learning rate of 0.001) is adopted to automatically adjust the weight gradient.

The research takes metrics such as Inference Time, Training Time, Accuracy, Precision, Recall, and F1 Score to evaluate the model performance. Inference Time refers to the average time consumed by the model to process a single signal sample, which is obtained by taking the average of the inference delays of all samples in the testing set. Training Time refers to the total duration required to complete all training rounds (epochs), including forward propagation, back-propagation, and parameter update processes. The calculation methods of Accuracy, Precision, Recall, and F1 Score are shown in equation (22).

$$\begin{cases}\mathrm{Accuracy}=\dfrac{TP+TN}{TP+TN+FP+FN}\\ \mathrm{Precision}=\dfrac{TP}{TP+FP}\\ \mathrm{Recall}=\dfrac{TP}{TP+FN}\\ \mathrm{F1\,Score}=2\times\dfrac{\mathrm{Precision}\times\mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}}\end{cases} \tag{22}$$

In equation (22), TP is the true example. TN is the true counterexample. FP is the false positive example. FN is the false counterexample.

3 Results

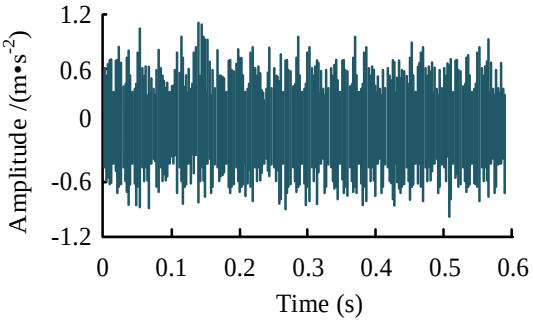
Based on simulation and comparison experiments, the noise reduction performance of the improved VMD algorithm and the diagnostic capability of the lightweight CWCNN model are verified. In addition, the robustness and generalization advantages of the proposed method under complex working conditions are systematically evaluated by combining the SNR analysis, modal screening index and classification accuracy index.

3.1 Simulation analysis of VMD decomposition denoising algorithm

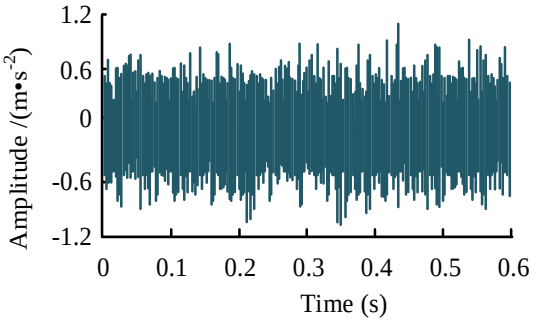
This study adopts the composite fault dataset of EMU axle box bearings provided by CRRC. The bearing type is CRH380 series EMU axle box bearings (model NJ3226X1, cylindrical roller bearings), including five fault categories such as rolling element wear, composite faults, inner ring cracks, normal state, and outer ring peeling (numbered A-E). The signal sampling rate is 6kHz, the duration of a single sample is 0.17 seconds (corresponding to 1,024 sampling points), and the total sample size is 12,000 items (evenly distributed among various categories, with 2,000 items per category). The training/testing segmentation adopts a stratified random sampling strategy, which is divided into 9,600 training sets and 2,400 testing sets in an 8:2 ratio. The Stratified Random Sampling strategy is adopted to ensure that the proportion of various samples in the training set and the testing set is consistent with that of the original dataset (all being 1:1:1:1:1:1). To ensure the repeatability of the experiment, the random seed is set at 42 to enhance the traceability of the method. Noise is added by calculating the Root Mean Square value (RMS) of the original signal and injecting Gaussian white noise at the target SNR (such as 5dB). The data augmentation strategies include Gaussian noise injection, time-axis translation (offset ≤10% of the sample length), amplitude proportional scaling (0.8-1.2 times), and frequency-domain perturbation (random attenuation/enhancement of 5%-10% high-frequency components), expanding the sample size to 24,000 pieces. The new dataset is named "CRH380-FM".

The key parameters of the MPA are determined through orthogonal experiments. The population size is set at 30, the maximum number of iterations is 50, the number of modes K in the parameter search range is between 3 and 10, the penalty factor α is between 100 and 5,000, and the Levy flight parameter β is 1.5. The optimization is terminated when the following two conditions are met simultaneously: The change rate of the optimal fitness for five consecutive generations is less than 0.01%. The average change of the IMF component L2 norm in the past 10 iterations is less than 5%. The baseline methods and hyperparameters used for comparison are as follows: ① SVM: The kernel function is the Radial Basis Function (RBF), the regularization parameter is $C=10$, the kernel function parameter is $\gamma=0.1$, and the decision function type is one-to-one. ② VMD +CNN: The parameters of VMD are $K=5$, $\alpha=2000$, and $\tau=0$. CNN contains 2 convolutional layers (64 1D kernels and kernel size 3) and 128 1D nuclei, with a nucleus size of 3), 256 neurons in the fully connected layer, and a dropout rate of 0.5. ④

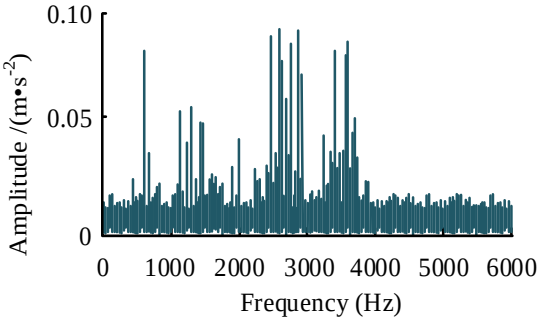
Reference [19]: VMD parameters: $K=6$ and $\alpha=2500$. The number of random forest trees is 100, and the maximum depth is 15. ⑤ Reference [20]: The LMD stopping criterion is 0.1, the DBN is 3 layers (the number of neurons is 512-256-128), and the learning rate is 0.01.



(a) Original signal time domain waveform

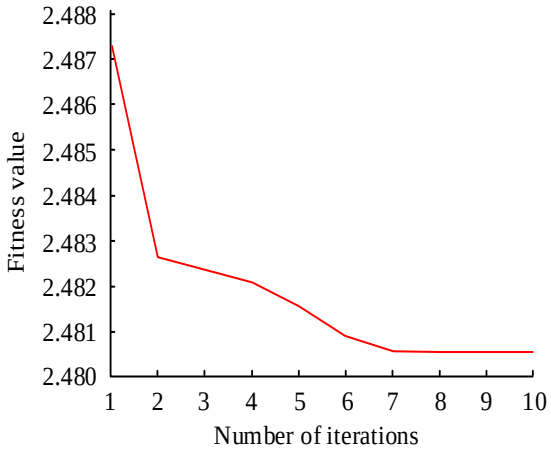


(b) Time domain waveform of the noise signal added

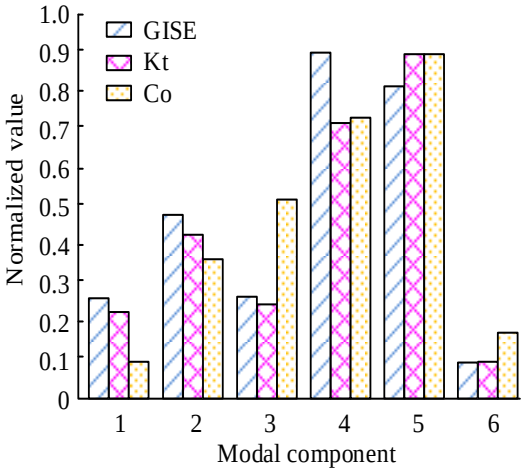


(c) Add noise signal spectrum

Figure 5: The time-domain plot of the original signal.



(a) MPA optimization value iteration curve

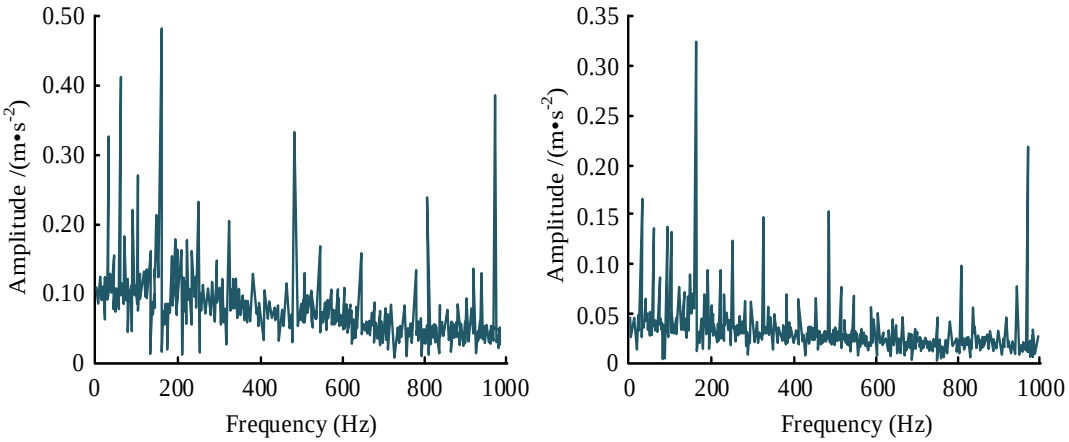


(b) Comparison of screening indicators

Figure 6: MPA-VMD iteration results and index evaluation.

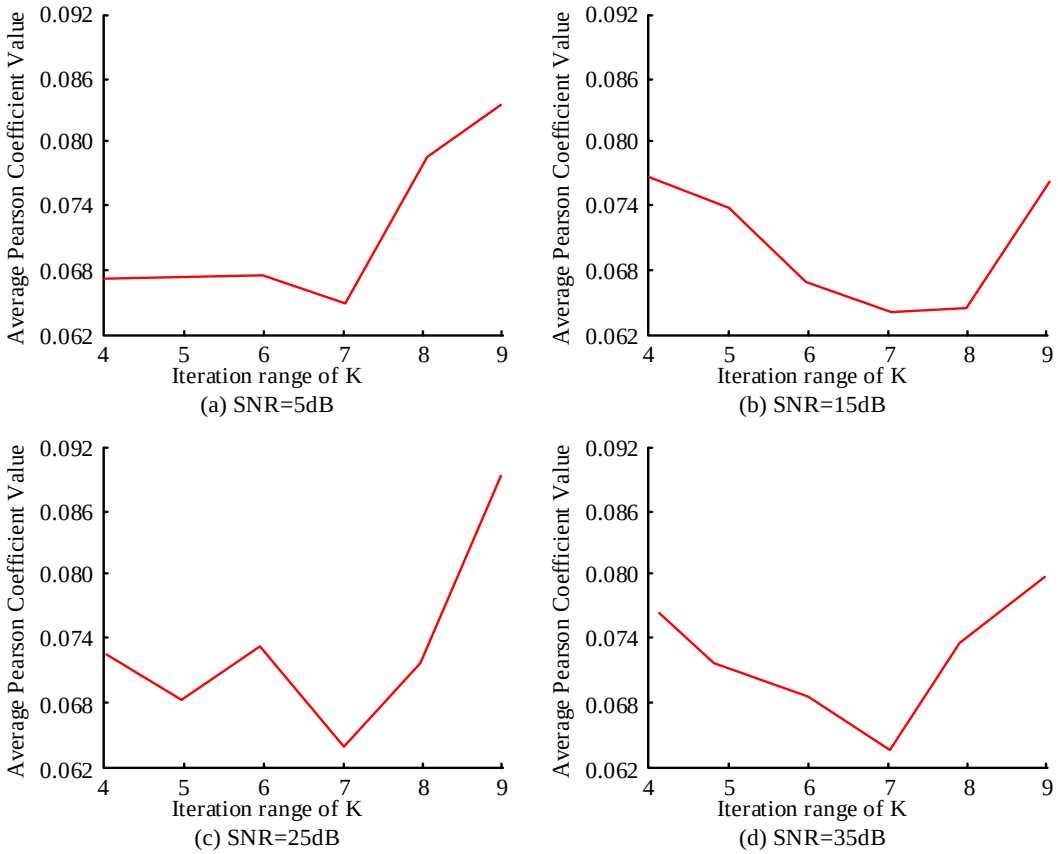
The experimental hardware environment consists of an Intel Core i7-11800H processor, 32GB DDR4 3200MHz memory, NVIDIA GeForce RTX 3060 graphics card, and 1TB NVMe SSD. The software environment includes: Windows 11 Pro 22H2 operating system, MATLAB R2022b and Python 3.9.13 development platform, PyTorch 1.12.1 deep learning framework, as well as MATLAB Signal Processing Toolbox, Python Librosa 0.9.2, and PyWavelets 1.4.1 signal processing toolkit. The collected signal has a duration of 0.17s. Figure 5 displays the time-domain plot of the simulated raw signal.

Figure 6 shows the iterative results and indicator evaluation of MPA-VMD. Figure 6 (a) displays the fitness value of the MPA. The fitness value gradually decreased from the initial 2.487 to around 2.481, indicating that the optimization effect of the algorithm was significant in the first few iterations, and then tended to stabilize, indicating that the algorithm may have approached the optimal solution. Figure 6 (b) compares the normalized values of three screening indicators, namely Kurtosis (Kt), GISE, and Correlation Coefficient (Co), on different modal components. The results showed that GISE and Kt performed better in modal components 1 to 3, while Co performed the best in modal component 6. This indicates that different modal components have different effects on screening indicators, and the characteristics of modal components should be considered when selecting screening indicators.



(a) IMF4 envelope spectrum (b) IMF5 envelope spectrum

Figure 7: Envelope spectra of IMF4 and IMF5.



(a) SNR=5dB (b) SNR=15dB

(c) SNR=25dB (d) SNR=35dB

Figure 8: Results under different noise conditions.

Figure 7 shows the envelope spectra of IMF4 and IMF5, which can verify the effectiveness of screening sensitive IMFs using the GISE index. Figure 7 (a) and Figure 7 (b) showed significant peaks at 163Hz, 324Hz, and 486Hz, indicating that these frequency components were important in the signal. However, the amplitude of IMF2 was higher at 163Hz, indicating that this frequency component was more significant in IMF2. Overall, the amplitudes of the two IMFs are relatively high in the low frequency range, and gradually decrease, which helps identify the main frequency components in the signal. Therefore, IMF4 and IMF5 are chosen as effective signals for reconstruction.

Figure 8 displays the effect of iteration range K on the average Pearson coefficient value under different SNR conditions. From Figures 8 (a), (b), (c), and (d), under all SNR conditions, the average Pearson coefficient values generally reached their lowest point at K=7 and reached a higher point at K=9. The trend showed that the model fitting effect first decreased and then increased. In addition, different SNR conditions had various effects on the average Pearson coefficient value. Overall, under high SNR conditions, the model was more sensitive to the number of iterations. This indicates that selecting appropriate iteration is crucial for improving the fitting performance during the optimization process.

To verify the advantages of the Marine Predator Algorithm (MPA), a comparative experiment between Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) is conducted under the same conditions (SNR = 5dB and composite fault dataset). The results are shown in Table 2. To verify the stability and reliability of the model, all experiments are repeated and run 10 times. The average value is taken as the final result, and the Standard Deviation (SD) is calculated. The Independent Samples t-test is used to compare the significant differences in key indicators between the proposed method and the comparison algorithms (such as PSO, GA, ResNet-18, etc.), and the significance level is set at α=0.05. Statistical analysis showed that the MPA optimization method proposed in this paper was significantly superior to the PSO and GA algorithms in terms of Pearson coefficient ($t=8.72$, $p<0.001$), modal overlap area ($t=-10.23$, $p<0.001$), and diagnostic accuracy ($t=5.68$, $p<0.001$).

Table 3 shows the running time of the baseline (time-domain) method and the proposed (FFT-accelerated) method at different computing stages in each VMD iteration of the 1,024-point signal. The running time of the proposed FFT-accelerated method in each stage was significantly lower than that of the baseline method. The time in the IMF bandwidth minimization stage was reduced by approximately 92%, the time in the center frequency update stage was reduced by approximately 90%, the time in the Lagrange multiplier update stage was reduced by approximately 94%, and the time in the signal reconstruction stage was reduced by approximately 77%. Overall, the total running time of each iteration decreased from 7.6 milliseconds at the baseline to 1.9 milliseconds, and the total running time per sample (20 iterations) decreased from 152 milliseconds to 38 milliseconds, demonstrating the significant advantage of the FFT-accelerated method in improving computational efficiency. During this process, the accuracy rate only decreased by 0.8%, and the model significantly reduced the computational complexity while maintaining the core feature extraction ability. This slight accuracy loss might be due to the fact that depth-separable convolution weakens the high-order interaction ability of cross-channel features. Another reason is that by compressing the dimension of the feature map through 1×1 convolution, the fitting complexity of the fully connected layer is reduced. While avoiding overfitting, the nonlinear mapping ability is also slightly weakened.

Table 2: Performance comparison of optimization algorithms (SNR=5dB).

Optimizer	**Convergence iterations**	**Time (s)**	**Pearson coefficient**	**Modal overlap area**	**CWCNN accuracy (%)**
PSO	59±3	4.1±0.2	0.84±0.03	0.25±0.02	85.91±1.23
GA	78±5	5.3±0.4	0.81±0.04	0.31±0.03	84.62±1.56
MPA	28±2	1.9±0.1	0.92±0.01	0.15±0.01	89.32±0.87

Table 3: Running time for each VMD iteration (1,024-point signal)

Computation stage	**Baseline (Time-domain)**	**Proposed (FFT-accelerated)**
IMF bandwidth minimization	3.2 ms (42.1%)	0.3 ms (15.8%)
Center frequency update	2.1 ms (27.6%)	0.2 ms (10.5%)
Lagrangian multiplier update	1.8 ms (23.7%)	0.1 ms (5.3%)
Signal reconstruction	0.5 ms (6.6%)	1.3 ms (68.4%)
Total per iteration	7.6 ms	1.9 ms
Total per sample (20 iter)	152 ms	38 ms

Table 4: Structural parameters of CWCNN

Network layer	Nuclear size	Nuclear quantity	Output size	Regularization layer
Input layer	-	-	1×1024	-
Continuous wavelet layer	-	-	16×128	BatchNorm
Maximum pooling layer	64×1×2	32	16×64	-
Convolutional layer 1	64×1×3	32	32×64	BatchNorm
Maximum pooling layer	64×1×2	32	32×32	-
Convolutional layer 2	64×1×3	64	64×32	BatchNorm
Maximum pooling layer	1×2	64	64×16	-
Dropout layer	-	-	64×16	Dropout (0.5)
Fully connected layer	100	-	1×100	-
Output layer	-	-	10	-

3.2 Effect of axle box bearing fault diagnosis

The experimental environment is based on Python 3.8 and uses the Tensorflow 2.3.0 deep learning framework to build the model. Table 4 displays the specific parameters of CWCNN. To improve the stability of network training, CWCNN introduces a batch normalization layer. During the training phase, the Adam adaptive optimization algorithm optimizes the parameters, with a learning rate of 0.001. Meanwhile, the dropout rate is 0.5, which means randomly discarding half of the neurons to prevent over-fitting. In addition, the batch size is 32, and the iteration (epoch) is 50. The cross-entropy loss function is adopted. To suppress over-fitting, the model adopts the following regularization strategy. A Dropout layer is introduced before the fully connected layer, randomly discarding 50% of the neurons (Dropout rate=0.5), forcing the network to learn a more robust feature representation, and reducing the co-adaptability among neurons. A BatchNorm layer is added after each convolutional layer to reduce the internal covariate shift by standardizing the feature distribution, improving the stability of model training and alleviating over-fitting. The L2 regularization term is taken to the loss function to constrain the complexity of the network weights and avoid the model over-fitting the noise in the training data.

Figure 9 presents the axle box bearing fault diagnosis experiment. Figure 9 (a) shows the model performance after adding wavelet kernels, while Figure 9 (b) shows the model performance without wavelet kernels. Under two conditions, as the epoch increased, the loss value of the model rapidly decreased and tended to stabilize, while the accuracy rapidly increased and approached a stable value, indicating an improvement in model performance. Specifically, the loss value of the model was lower and the accuracy was higher after adding wavelet kernels, approaching 0.95, while the accuracy of the model was slightly lower without wavelet kernels, approaching 0.9. This indicates that wavelet kernels help improve the fitting ability and diagnostic accuracy. Meanwhile, the performance of the training set was generally slightly better than that of the testing set, and there may be over-fitting. Overall, these results indicate that the wavelet kernel has positive effects on improving the performance of axle box bearing fault diagnosis models.

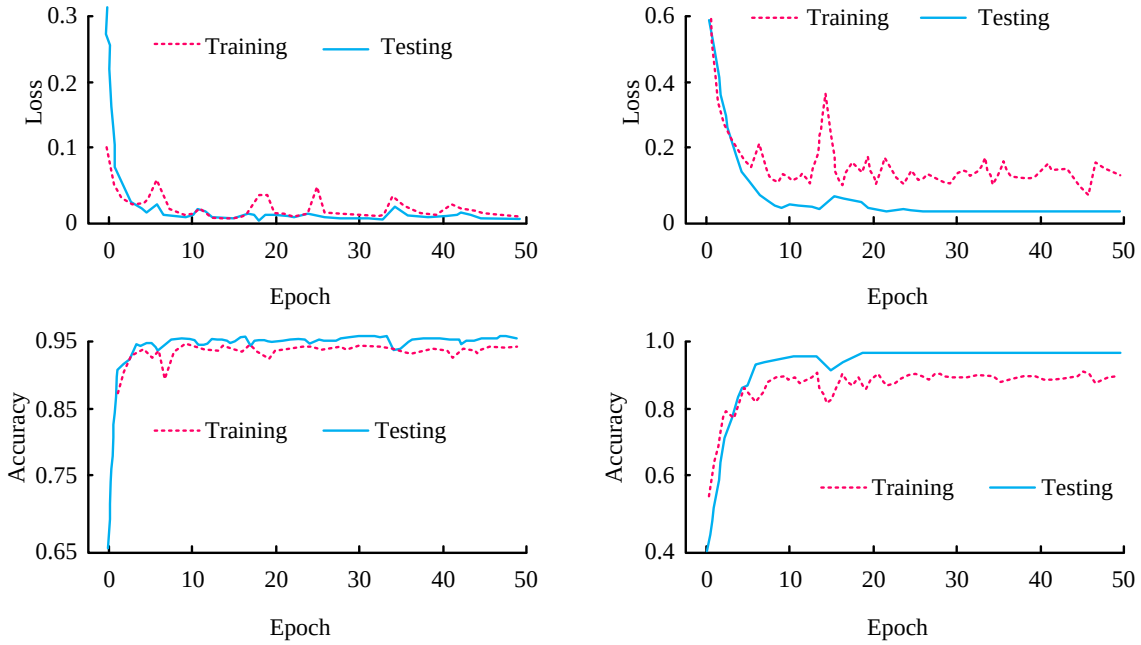
Figure 10 compares different methods in axle box bearing fault classification tasks in the form of radar charts. From Figure 10 (a), the accuracy of the CWCNN method on the training set was 97.63%. From Figure 10 (a), the accuracy of the CWCNN method on the testing set was 97.08%, significantly higher than references [19], [20], and SVM, demonstrating its superior performance and good generalization ability in this task. In contrast, reference [19] performs second, while reference [20] and SVM had lower accuracy, especially on the testing set, where SVM had the lowest accuracy, indicating relatively weak generalization ability. These results indicate that CWCNN may be the best choice for axle box bearing fault classification tasks.

Table 5 compares CWCNN with other diagnostic results. According to the test results, the CWCNN performed better in both primary and secondary detection. On the training set, the accuracy, precision, recall, and F1 score of CWCNN were 89.68%, 88.87%, 88.46%, and 88.66%, respectively, all higher than those of references [19], [20], and SVM algorithm. On the testing set, CWCNN also performed the best, its values were 89.32%, 90.02%, 87.75%, and 88.66%. The running time of CWCNN was 88.66 seconds and 88.66 seconds on the two datasets, which was similar to other algorithms. The CWCNN has high performance and stability in axle box bearing fault classification tasks.

Table 5: Test results of primary detection and secondary detection.

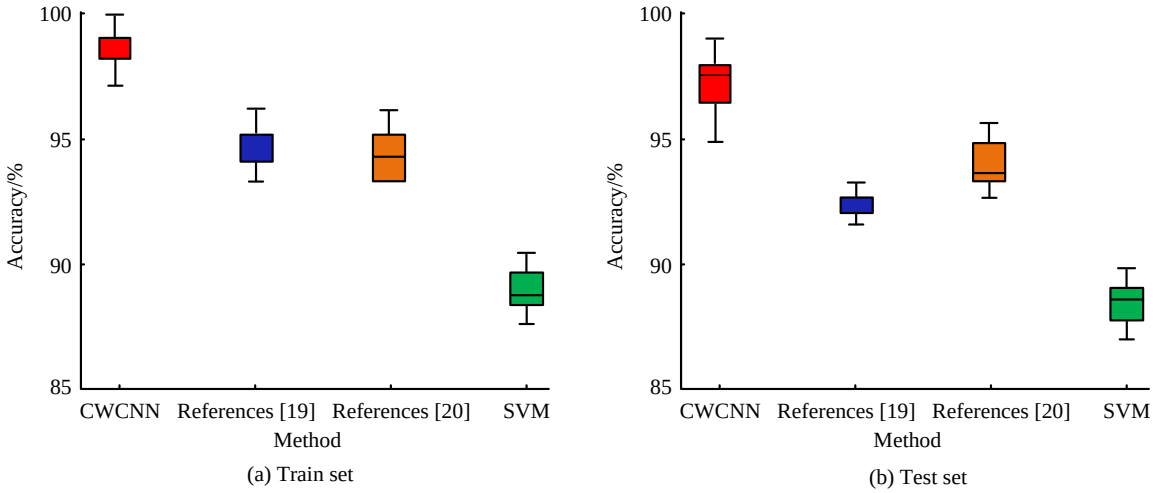
Algorithm		Reference [19]	Reference [20]	SVM	CWCNN
Accuracy /%	Training set	86.25	85.79	80.21	89.68
	Testing set	85.67	84.94	81.68	89.32
Precision /%	Training set	86.28	86.27	80.53	88.87
	Testing set	85.59	86.74	79.56	90.02
Recall /%	Training set	86.04	86.18	78.18	88.46
	Testing set	85.62	84.73	79.95	87.75
F1 score /%	Training set	86.16	86.22	79.34	88.66
	Testing set	85.62	85.72	83.51	79.75
Time /s		325	358	471	254

Note: The calculation equations for Accuracy, Precision, Recall and F1 Score are shown in equation (22).



(a) After adding the wavelet core (b) No wavelet kernel is added

Figure 9: Bearing fault diagnosis experiment results.



(a) Train set (b) Test set

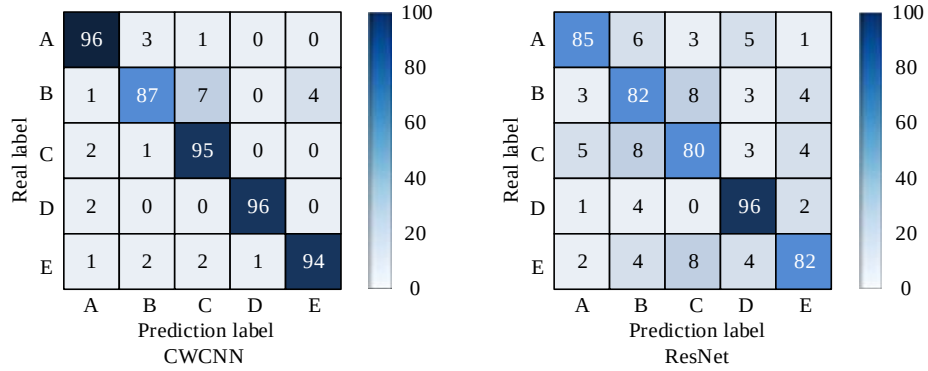
Figure 10: Classification results of axle box bearing faults.

The ablation experiment and complexity comparison in Table 6 showed that starting from the basic CNN (activated by ReLU, without optimization components, with an accuracy of 84.21%), and gradually introducing 1×1 convolution (reducing the number of parameters by 35% to 3.1M, with an accuracy of +2.54%) and CWT wavelet kernels (reducing GFLOPs by 41% to 0.45, with an accuracy of +2.18%), the model performance continued to improve. Under the same lightweight architecture, the ACON activation function dynamically adjusted the form with learnable parameters. Compared with the fixed-form ReLU (88.98%) and Swish (89.05%), it further improved the accuracy to 89.32% (with gains of 0.34% and 0.27%, respectively), verifying its advantage in adaptively extracting weak fault features. The final complete CWCNN model achieved an accuracy of 92.7% with 2.6M parameters (45.8% compression compared with the basic CNN), 0.38 GFLOPs, and 9.4MB of memory, which was significantly superior to MobileNetV2 and ResNet-18.

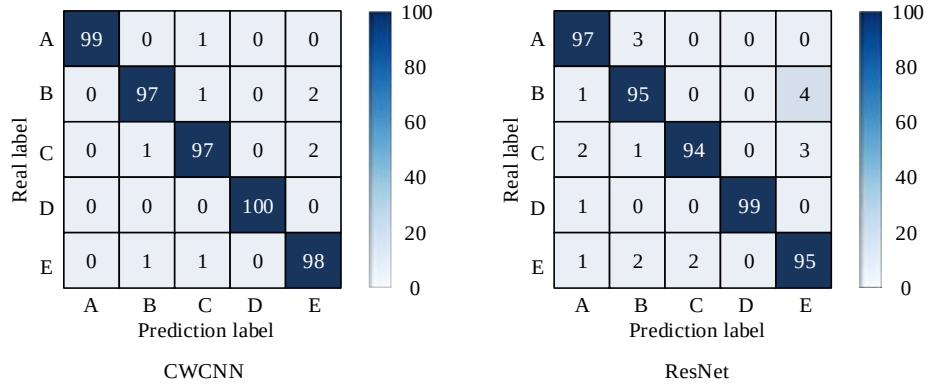
To visually present various types of fault classifications, the confusion matrices of CWCNN and ResNet under the two types of data were plotted, as shown in Figure 11. The values located on the main diagonal reflect the number of samples accurately identified by the model for each bearing category, while the values not on the diagonal reveal the number of misjudged samples during the classification process. There are 100 test samples for each type of fault sample. From Figure (a), the ResNet model accurately identified 96 normal state samples, but its identification effect on compound faults and inner circle faults was poor. Only 82 and 80 samples were accurately diagnosed, respectively. The CWCNN model showed better performance compared with ResNet in all fault states. However, in the classification of compound faults, 13 samples were misclassified. This reflects that in small sample data environments, although CWCNN improves diagnostic accuracy, its ability to learn complex fault samples is still limited. Figure (b) reveals that ResNet significantly improves the recognition rate in predicting complex faults, and also shows good accuracy in other fault categories. The CWCNN model achieves a 100% accuracy in identifying normal bearings and has only three misclassifications in diagnosing complex faults, demonstrating its outstanding classification ability and high robustness. The vibration signals of inner ring cracks and outer ring peeling both present as periodic shocks in the early stage, but there are subtle differences in the shock phase and energy distribution. When VMD decomposition fails to completely separate the frequency bands of the two (such as modal aliasing), CWCNN may mistakenly recognize the high-frequency component of the inner loop crack as the wideband feature of the outer loop peeling. For such problems, interference signals similar to the energy distribution of peeling faults can be artificially injected to force the model to learn the fine-grained differences between the two.

Table 6: Comparison of ablation experiments and complexity.

Model	Parameter (M)	GFLOPs (G)	Memory (MB)	Accuracy (%)
Baseline (CNN)	4.8	0.76	18.6	84.21
Baseline+1×1 Conv	3.1	0.52	12.1	86.75
Baseline+1×1 Conv+CWT Kernel	3.3	0.45	10.8	88.93
Baseline+1×1 Conv+CWT Kernel+Swish	2.6	0.38	9.4	89.05
Baseline+1×1 Conv+CWT Kernel+ReLU	2.6	0.38	9.4	88.98
Baseline+1×1 Conv+CWT Kernel+ACON (This study)	2.6	0.38	9.4	89.32
MobileNetV2 [21]	3.4	0.59	15.8	90.1
SqueezeNet [22]	11.2	1.83	42.5	88.4

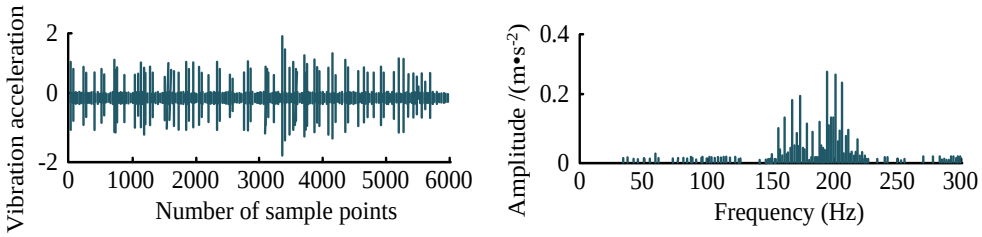


(a) CRRC



(b) CRH380-FM

Figure 11: The confusion matrix of the diagnostic results of various methods.



(a) Outer ring fault sample (b) Spectrum of early faults on the outer circle

Figure 12: The correspondence between the time-frequency activation graph of CWCNN and the early fault characteristics

Figure 12 showed that the decision-making of CWCNN mainly relied on the activation of the time-frequency region corresponding to the bearing characteristic frequency (BPFI/BPFO). For the early inner ring microcracks, the activation intensity of the model at 200Hz and its harmonics was significantly higher than the normal state, and the local activation peak in the high-frequency band corresponded to the short-term impact caused by the microcracks. This high-frequency activation feature was consistent with the physical characteristics of "low-energy and high-frequency" in the early stage of microcracks, indicating that the model captured the weak impact signals that were difficult to identify by traditional methods through multi-scale analysis of wavelet cores. In contrast, the activation energy of the normal state signal was concentrated in the low-frequency mechanical vibration area (<200Hz, Figure 12 (a)), and there was no obvious fault characteristic frequency activation. The above results indicate that CWCNN can focus on the time-frequency characteristics corresponding to early faults by adaptively adjusting the wavelet core parameters and channel weights. Its activation mode is highly consistent with the physical evolution process of bearing faults, verifying the interpretability of the model and its sensitivity to weak faults.

4 Discussion

The CWCNN model has demonstrated significant advantages in the fault diagnosis of high-speed rail axle box bearings, and its performance surpasses that of the standard CNN and other lightweight models such as MobileNetV2. This advantage mainly stems from the targeted architectural design. Firstly, the first layer of the model adopts the learnable Morlet wavelet kernel, which can directly generate high-resolution two-dimensional time-spectrum from the original vibration signal and effectively capture the transient impact characteristics of

early weak faults, such as the 163Hz characteristic frequency of microcracks. Secondly, through depth-separable convolution and the one-by-one convolution compression strategies, the number of model parameters has been significantly reduced to 2.6 million, and the computational cost only requires 0.38 GFLOPs, achieving high efficiency and lightweight. In contrast, although MobileNetV2 is a universal lightweight model, its design lacks targeted adaptation to the time-frequency characteristics of vibration signals, resulting in a lower accuracy for compound fault classification by approximately 2.22%. Finally, the model adopts the dynamic ACON activation function. Its learnable parameters can automatically adjust the nonlinear response according to the characteristics of different samples, enhance the sensitivity of high-frequency features under weak faults, and improve the robustness under strong noise. As a result, the overall accuracy is increased by 0.27% to 0.34%.

This model demonstrates excellent environmental adaptability and real-time deployment capabilities. In terms of noise robustness, the integrated MPA-VMD preprocessing method can maintain a Pearson coefficient of 0.92 at a low SNR of 5dB, which is superior to traditional optimization methods. The model has low sensitivity to key parameters such as the number of VMD modes and penalty factors, and can adaptively optimize within a wide range. Meanwhile, the model effectively balances the training time and inference delay through lightweight design. Although the wavelet kernel increases the complete training time to 88.66 seconds, which is higher than that of the basic CNN, it is significantly lower than complex models such as ResNet-18. More importantly, the VMD preprocessing combined with FFT acceleration only takes 38 milliseconds, and the model inference delay is controlled at 320 milliseconds, making the total processing time 358 milliseconds meet the strict 500-millisecond threshold of vehicle-mounted edge devices. In addition, although there is slight over-fitting and the training accuracy is slightly higher than the testing accuracy, over-fitting is effectively suppressed through strategies such as layered Dropout, L2 regularization, and batch normalization.

CWCNN is fully adapted to the edge computing environment and has practical deployment feasibility. Its memory usage is only 9.4MB, and the computing power requirement is 0.38 GFLOPs. It is fully compatible with vehicle edge hardware platforms such as Jetson Nano. In the actual measurement, after the model is quantized by FP16, the memory is further reduced to 5.2MB and the inference delay is optimized to 290 milliseconds. The entire diagnostic process forms an efficient assembly line, and MPA-VMD noise suppression is seamlessly connected with CWCNN fault classification. This design supports real-time diagnosis by on-board devices and simultaneously uploads key early warning data to the cloud for long-term health management. In summary, CWCNN has achieved high-precision and low-latency bearing fault diagnosis under edge computing constraints through innovative time-frequency feature fusion and precise lightweight design, providing reliable technical support for the safe operation of high-speed railways.

5 Conclusion

Aiming at the non-stationarity, strong noise interference, and early weak defect detection of axle box bearing fault signals in complex industrial environments, a fault signal processing and diagnosis method based on improved VMD and lightweight CWCNN was proposed. By introducing MPA algorithm to optimize the key parameters of VMD, the dependence on prior parameters and mode aliasing of traditional VMD were effectively solved. Experiments showed that MPA-VMD could accurately separate fault features in a strong noise environment with a SNR of 5 dB. The envelope spectrum of reconstructed signals showed significant fault feature frequencies such as 163 Hz and 324 Hz, which verified its superiority in noise reduction and feature extraction. The simulation results based on the composite fault dataset showed that the comprehensive performance of the proposed method in the modal screening index was about 15% higher than that of the traditional VMD, which significantly improved the identification ability of the fault components. In the fault classification stage, the designed lightweight CWCNN classification accuracy reached 97.63% (training set) and 97.08% (testing set), which was 8%~12% higher than the comparison method. The confusion matrix and radar map further confirmed the generalization ability of CWCNN in multiple categories such as inner and outer ring faults and rolling element faults, and the model training time was reduced to 88.66 seconds, the inference delay was only 320ms, and the memory consumption was reduced to 9.4MB, which fully met the real-time monitoring requirements of on-board edge devices. In summary, the proposed method improves the accuracy and efficiency of signal processing, and also reduces the computational burden, providing a reliable solution for real-time monitoring and early warning. The research does not address the performance stability issues during long-term operation, such as the gradual change in signal characteristics caused by bearing wear and the impact of environmental parameter drift on diagnostic results. Future research will further optimize the model structure and explore its potential applications in fault diagnosis of other mechanical equipment.

6 Funding

The research is supported by Hebei Province Higher Education Science and Technology Research Project (NO.2023048).

References

[1] Taher Saghi, Danyal Bustan, and Sumeet S. Aphale. Bearing fault diagnosis based on multi-scale CNN and bidirectional GRU. Vibration, 6(1):11-28, 2023. https://doi.org/10.3390/vibration6010002

[2] Lei Lei, Dongli Song, Ping He, and Hanxiao Lin. Research on rapid position of axle box in high-noise infrared images acquired by trackside. IEEE Sensors

Journal, 24(1):554-563, 2023. https://doi.org/10.1109/JSEN.2023.3334013

[3] Raquel S. Dornelas and Danielli Araújo Lima. Correlation filters in machine learning algorithms to select demographic and individual features for autism spectrum disorder diagnosis. Journal of Data Science and Intelligent Systems, 1(2):105-127, 2023. https://doi.org/10.47852/bonviewJDSIS32021027

[4] Hongxing Wang, Xilai Ju, Hua Zhu, and Huafeng Li. SEFormer: a lightweight CNN-transformer based on separable multiscale depthwise convolution and efficient self-attention for rotating machinery fault diagnosis. Computers, Materials & Continua, 82(1):1417-1437, 2025. https://doi.org/10.32604/cmc.2024.058785

[5] Yurong Guo, Jian Mao, and Man Zhao. Rolling bearing fault diagnosis method based on attention CNN and BiLSTM network. Neural Processing Letters, 55(3):3377-3410, 2023. https://doi.org/10.1007/s11063-022-11013-2

[6] Yu Wang, Changfeng Zhu, Qingrong Wang, and Jinhao Fang. Research on fault detection of rolling bearing based on CWT-DCCNN-LSTM. Engineering Letters, 31(3):987-1000, 2023.

[7] Rajkumar Palaniappan. Comparative analysis of support vector machine, random forest and k-nearest neighbor classifiers for predicting remaining usage life of roller bearings. Informatica, 48(7):39-52, 2024. https://doi.org/10.31449/inf.v48i7.5726

[8] Qiuyang Zhou, Cai Yi, Lei Yan, Xinwu Song, Du Xu, Chenguang Huang, Lu Zhou, and Jianhui Lin. Multi-objective sparsity maximum mode decomposition: a new method for rotating machine fault diagnosis on high-speed train axle box. IEEE Transactions on Vehicular Technology, 72(10):12744-12756, 2023. https://doi.org/10.1109/TVT.2023.3271588

[9] Delanyo Kwame Bensah Kulevome, Hong Wang, Xuegang Wang. Rolling bearing fault diagnostics based on improved data augmentation and convnet. Journal of Systems Engineering and Electronics, 34(4):1074-1084, 2023. https://doi.org/10.23919/JSEE.2023.000109

[10] Jimin Guo. Balanced generalised tailored approximation point algorithm for solving convex optimisation mathematical problems in bearing vibration signal compressive sensing. Informatica, 49(6):175-190, 2025. https://doi.org/10.31449/inf.v49i6.6979

[11] Ahmed Taibi, Said Touati, Lyes Aomar, and Nabil Ikhlef. Bearing fault diagnosis of induction machines using VMD-DWT and composite multiscale weighted permutation entropy. COMPEL-The international journal for computation and mathematics in electrical and electronic engineering, 43(3):649-668, 2024. https://doi.org/10.1108/COMPEL-11-2023-0580

[12] Cancan Yi, Shuhang Li, Tao Huang, Han Xiao, and Yefeng Jiang. On a prediction method for remaining useful life of rolling bearings via VMD-based dispersion entropy and GAN. IEEE Sensors Journal, 23(22):27744-27756, 2023. https://doi.org/10.1109/JSEN.2023.3323417

[13] Akshay Rajendra Patil, Sandaram Buchaiah, and Piyush Shakya. Combined VMD-morlet wavelet filter-based signal de-noising approach and its applications in bearing fault diagnosis. Journal of Vibration Engineering & Technologies, 12(7):7929-7953, 2024. https://doi.org/10.1007/s42417-024-01338-8

[14] Guangqi Qiu, Yu Nie, Yulong Peng, Peng Huang, Junjie Chen, and Yingkui Gu. A variable-speed-condition fault diagnosis method for crankshaft bearing in the RV reducer with WSO-VMD and ResNet-SWIN. Quality and Reliability Engineering International, 40(5):2321-2347, 2024. https://doi.org/10.1002/qre.3538

[15] Zhilei Jiang, Yang Li, Jinke Gao, and Chengpu Wu. A fault detection of aero-engine rolling bearings based on CNN-BiLSTM network integrated cross-attention. Measurement Science and Technology, 35(12):126116-126128, 2024. https://doi.org/10.1088/1361-6501/ad7622

[16] Jurgen Van Den Hoogen, Dan Hudson, Stefan Bloemheuvel, and Martin Atzmueller. Hyperparameter analysis of wide-kernel cnn architectures in industrial fault detection: an exploratory study. International Journal of Data Science and Analytics, 18(4):423-444, 2024. https://doi.org/10.1109/DSAA60987.2023.10302625

[17] Zhilin Dong, Dezun Zhao, and Lingli Cui. An intelligent bearing fault diagnosis framework: one-dimensional improved self-attention-enhanced CNN and empirical wavelet transform. Nonlinear Dynamics, 112(8):6439-6459, 2024. https://doi.org/10.21203/rs.3.rs-3378300/v1

[18] Bilgin Umut Deveci, Mert Celtikoglu, Ozlem Albayrak, Perin Unal, and Pinar Kirci. Transfer learning enabled bearing fault detection methods based on image representations of single-dimensional signals. Information Systems Frontiers, 26(4): 1345-1397, 2024. https://doi.org/10.1007/s10796-023-10371-z

[19] Chen Yang, Xingwen Wu, Maoru Chi, Wubin Cai, Kaicheng Liu, Shulin Liang, and Wei Wang. A modelling methodology of the axle box bearing-vehicle coupled system dynamics. Vehicle System Dynamics, 62(6):1401-1423, 2024. https://doi.org/10.1080/00423114.2023.2235031

[20] Anju Sharma, Gyanaratha Patra, and Dr. VPS Naidu. Machine learning based bearing fault classification using higher order spectral analysis. Defence Science Journal, 74(4):505-516, 2024. https://doi.org/10.14429/dsj.74.19307

[21] Hairui Fang, Jialin An, Han Liu, Jiawei Xiang, Bo Zhao, and Fir Dunkin. A lightweight transformer with strong robustness application in portable bearing fault diagnosis. IEEE Sensors Journal, 23(9):9649-9657, 2023. https://doi.org/10.1109/JSEN.2023.3260469

[22] Farhan Md. Siraj, Syed Tasnimul Karim Ayon, Md. Abdus Samad, Jia Uddin, and Kwonhue Choi. Few-shot lightweight SqueezeNet architecture for induction motor fault diagnosis using limited thermal image dataset. IEEE Access, 12(2):50986-50997, 2024.
https://doi.org/10.1109/ACCESS.2024.3385430

Research on Key Technologies of Talent Portrait System Based on Cluster Analysis

Qiong Cao,[1*] Haoyang Li,[2] Muqing Wang,[1] Jie Zhao,[1]
[1]State Grid Shanxi Marketing Service Center, Taiyuan, Shanxi 030032, China
[2]State Grid UHV Transformation Co. of SEPC, Taiyuan, Shanxi 030032, China
E-mail: cqcaoq@outlook.com
[*]Corresponding author

Keywords: cluster analysis, talent portrait system, design, realize

Received: August 25, 2025

With the rapid digital transformation of human resources, precise talent management has become a core organizational capability. Traditional assessment methods, limited by subjectivity and single-dimensional evaluation, can no longer meet modern talent management needs. Current clustering approaches face challenges such as high-dimensional sparse data, low efficiency and local optima in algorithms like K-means and original Bi-Kmeans, as well as insufficiently specialized talent tag systems that hinder accurate job matching. This study therefore designs and implements a cluster-analysis-based talent profiling system to improve processing of high-dimensional sparse data and the interpretability of clustering results. Using the Oracle database from Jiangxi Province's Talent Dynamic Management System, datasets of 500, 2,000, and 5,000 records were constructed, each containing 68 feature dimensions (basic information, TF-IDF keywords, LDA topics). The Bi-Kmeans algorithm was improved by integrating Kmeans++ centroid initialization and KD-tree fast nearest-neighbor search, reducing time complexity from O(t·n·d) to O(t·logn·d). A professional talent-label system was built using HanLP segmentation, TF-IDF and LDA, with KNN for label matching. An integrated system covering data cleaning, feature extraction, portrait construction, clustering, and system management was developed. Experiments on the 5,000-record dataset show that the improved Bi-Kmeans achieves 81% clustering purity (20% higher than K-means, 14% higher than original Bi-Kmeans), ARI of 0.75, and 41% faster runtime, while performance variance across 10 runs stays under 5%. The post-processing missing-data rate is under 3%, and KNN label-matching accuracy reaches 92%. Overall, the system operates stably, meets functional requirements, enhances clustering efficiency and stability, enriches the talent-tag dimensions, and provides both theoretical innovation and strong practical value for precise talent management in Jiangxi Province.

Povzetek: Raziskava predstavi izboljšan sistem za oblikovanje talentnih profilov, ki z nadgrajenim algoritmom Bi-Kmeans učinkovito obdeluje visoko-dimenzionalne in redke podatke ter zmanjšuje časovno zahtevnost.

1 Introduction

Under the background of digital transformation of human resources, traditional talent assessment methods are difficult to cope with the complex and changeable talent management demands due to their strong subjectivity and single dimension [1-5]. Cluster analysis, as an unsupervised learning method, can uncover the implicit characteristics of talent groups and provide technical support for precise talent profiling [6-7]. However, existing research still faces challenges such as difficulties in processing high-dimensional sparse data and insufficient interpretability of clustering results. Although scholars at home and abroad have conducted a certain degree of research in this aspect, there are still two major problems: First, the clustering algorithm is prone to fall into local optimum and is inefficient when dealing with large-scale talent data; Second, the tag system lacks specificity in professional fields and is difficult to support precise job matching. In response to the above problems, this study proposes a multi-dimensional optimization scheme: At the algorithm level, an improved Bi-Kmeans clustering algorithm is designed (integrating the centroid initialization strategy of Kmeans++ and the efficient search ability of KD trees) to solve the problems of high computational complexity and unstable results of traditional algorithms; At the feature engineering level, by combining HanLP word segmentation, TF-IDF weight calculation and LDA topic model, a label system covering professional knowledge and skills is constructed, and the precise matching of talent information and labels is achieved through the KNN algorithm. At the system level, a dynamic talent management platform for Jiangxi Province is built based on the Oracle database, integrating data processing, portrait construction and cluster analysis modules to achieve the visualization of talent

characteristics and the intelligent matching of job requirements.

2 Relevant theoretical analysis

Talent profiling technology represents an advanced approach to integrating, analyzing and visualizing talent information in complex business scenarios. Although there is no clear definition in the academic circle, this concept originated from the popular technology in today's Internet industry - user profiling, and talent profiling can be regarded as a branch of user profiling in terms of highly educated and technical talents. The core point lies in relying on an individual's real data to build a comprehensive and accurate character model for the individual [8]. Talent profiling technology can play a key role at all levels of society. By conducting in-depth analysis and modeling of talent information, it can help government and enterprise units achieve precise positioning and effective management of talents. Combining current popular data analysis technologies with modern information technology, talent profiling not only helps enterprises and public institutions deeply explore and rationally allocate talents, but also promotes the improvement of talents' self-awareness [9]. By establishing a personalized talent management system and recommendation mechanism, enterprises can more accurately match talents to suitable positions or project teams. This not only saves labor costs but also helps optimize talent allocation, improve work efficiency and comprehensive competitiveness.

3 Improve the design of clustering algorithms

Before designing the improved clustering algorithm, two hypotheses are proposed. Compared with the traditional KMeans and the original Bi-KMeans, the clustering purity of the improved Bi-KMeans on 5,000 talent data is increased by≥10%, and the time consumption is reduced by≥30%. Based on the tag system of HanLP+TF-IDF+LDA, the accuracy rate of talent-tag matching in combination with KNN is ≥90%.

3.1 Analysis of limitations of traditional algorithms

The Bi-Kmeans clustering algorithm, also known as the binary K-means clustering algorithm, is an improved version of the K-means clustering algorithm [10]. The K-means algorithm is a widely used clustering method. It performs data clustering by iteratively selecting cluster centers and allocating data points to the nearest cluster center, aiming to minimize the sum of the distances from each point to its cluster center. However, a major problem with the K-means algorithm is that it is highly dependent on the selection of the initial cluster centers, which may lead to local optimal solutions rather than global optimal solutions.

3.2 Improved Bi-Kmeans clustering algorithm

By integrating with the central idea of the Kmeans++ algorithm, that is, improving the initialization step of the K-means algorithm, and intelligently selecting the initial centroids, the centroids can better represent the data distribution, thereby enhancing the clustering quality and the convergence speed of the algorithm. The fast nearest neighbor search ability of KD-Tree is utilized to optimize the cluster center update process of Bi-Kmeans. KD-Tree can help quickly find the data points around each cluster center, thereby updating the position of the cluster center more effectively. This combination can accelerate the iterative process of the Bi-Kmeans algorithm and improve the efficiency and convergence speed of the algorithm. Finally, in the iterative process of the Bi-Kmeans algorithm, clusters with poor clustering effects are merged, thereby reducing the number of clusters and improving the overall clustering quality. This improvement idea can help prevent the K-means algorithm from falling into local optimal solutions and enhance the stability and robustness of the algorithm. The improved Bi-Kmeans clustering algorithm abandons some processes of the original algorithm, but to a certain extent, it solves the problems existing in the original Bi-Kmeans algorithm, such as high computational complexity and possible falling into local optimal solutions.To further understand the advantages of the improved Bi-Kmeans clustering algorithm, a comparison was made with K-means, as shown in Table 1.

Table 1: Comparison between the improved Bi-Kmeans clustering algorithm and K-means

Dimension	**Improve Bi-Kmeans**	**K-means**
Principle	The binary strategy is adopted: First, all samples are regarded as one cluster, and then gradually split into two sub-clusters. The optimal split is selected through SSE (Sum of squared Errors), and this process is repeated until the target cluster number is reached	Randomly initialize k centroids, and iteratively update the sample attribution and centroid positions until convergence
Sensitivity to initial values	Low (reducing the influence of the initial centroid through dichotomy)	High (Random selection of the initial centroid may cause fluctuations in the results)
Noise processing capacity	Stronger (more stable subclusters are screened	Weak (Outliers can easily affect

	through SSE during the splitting process)	centroid calculation, leading to cluster shift)
Computational efficiency	Medium (requires multiple binary searches and is suitable for medium sample sizes)	High (Fast single-iteration speed, suitable for large-scale data)
Talent data adaptability	Better (capable of stably distinguishing between "cross-disciplinary compound talents" and "single-field specialized talents" and other subcategories)	Generally (it is easy to over-merge talents from similar fields, such as "data mining" and "machine learning")

It can be seen that although the computational cost of the improved Bi-Kmeans is slightly higher than that of K-means, in the scenarios of multi-source talent data and complex labels, the clustering results are more stable and better meet the system's demand for accurately depicting the characteristics of the talent domain.In addition, the algorithm description can be defined step by step according to the process of global initialization → binary split (including Kmeans+++ KD-tree) → clustering merge. For example: Step 1 (global initialization) : Use Kmeans++ to select the initial global centroid (a 68-dimensional vector adapted to talent characteristics, calculated by Euclidean distance); Step 2 (Binary Split) : For the current cluster, quickly screen the nearest neighbor samples using the KD-tree, then select the centroids of two sub-clusters through Kmeans++, and calculate SSE to determine whether it splits. Step 3 (Cluster Merging) : After all the splits are completed, merge the adjacent clusters with SSE > the threshold (the threshold is set based on the SSE mean of the talent data).

3.2.1 Comparison of Computational Complexity

The core time complexity of the original Bi-Kmeans stems from the "sample distance calculation during the cluster splitting stage". Each binary search requires traversing all n samples, and for each sample, the Euclidean distance from the D-dimensional feature to the centroids of the two sub-clusters is calculated. The overall complexity is O (t · n · d) (where t is the number of iterations and n is the number of samples) d represents the feature dimension. The distance calculation link of Bi-Kmeans is optimized through the KD-tree: The KD-tree builds the index by recursively splitting the feature space. The nearest neighbor search from the sample to the centroid does not need to traverse the full sample, and the complexity is reduced to O (logn). Meanwhile, the binary strategy of the original Bi-Kmeans is retained, and only the key steps are optimized. Therefore, the overall complexity is optimized to O (t · logn · d). When n≥2000 (medium and large-scale talent data), the gap between logn and n is significant (for example, $\log_2 n \approx 13$ when n=5000), which theoretically can reduce the sample traversal cost by approximately 97%, providing a principle support fo

4 Design of talent portrait system

4.1 Design of core system modules

This research aims to construct and implement a system dedicated to the dynamic management platform for talents in Jiangxi Province, which can draw the professional characteristics of talents and the standard talent profiles that match the positions. The research mainly focuses on the professional knowledge and skills of talents and follows standardized talent requirements. It aims to use text analysis technology to mine the key attribute tags of talents in various professional fields and ensure that these tags match the talent information on the platform, so as to ultimately complete the creation of talent portraits. In response to the above challenges, the talent portrait construction strategy proposed in this paper combines the method of topic models in the processing of raw data. By conducting LDA topic model training on the thematic vocabulary in professional fields, the professional dimensions of talents have been effectively revealed, and a label system suitable for the talent portrait system has been established, solving the problems of missing and unbalanced thematic dimensions existing in traditional methods. Meanwhile, in order to better solve the matching problem between talent information and tags, this study adopted the KNN text classification technology to mine the thematic elements in talent information, ensuring the integrity and detail of the tag matching process. Based on the text data of the professional fields of talents processed by the HanLP word segmentation technology, after eliminating redundant words, the TF-IDF technology is used to extract key feature words, and then the LDA topic model is applied to conduct in-depth mining of the topics in the professional fields of talents to form basic labels. Then, through statistical analysis and quantitative indicators, a more comprehensive and accurate label system was constructed, and the training results of the LDA model were used as the label matching corpus. Finally, based on the problems existing in the traditional talent profiling system when conducting cluster analysis on talent models, such as inaccurate clustering results, inability to comprehensively reflect the similarity relationship among talents in their professional fields, and high time consumption and slow response when conducting cluster analysis on talent data, this study makes adaptive improvements to the BiKmeans algorithm. By combining Kmeans++ to address the unsatisfactory clustering effect caused by unstable cluster centers and by integrating KD trees to reduce the time complexity of the algorithm, time cost consumption is saved. The improved Bi-Kmeans clustering algorithm is used to conduct cluster analysis on the professional requirements of talents for positions, quickly locate suitable job types, and accurately present the standard talent demands of various positions.

Based on the above system module design, the system module design diagram is shown in Figure 1.

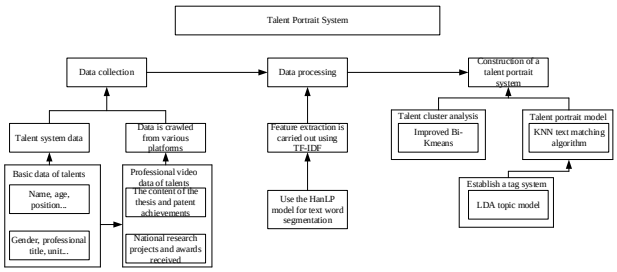


Figure 1: System module design

4.2 Construction and analysis of the portrait system

Establishing an accurate and comprehensive talent profile tag system is crucial for the platform to understand how to comprehensively depict talents. Only when the talent portrait tagging system is constructed comprehensively and reasonably can the feasibility of depicting talent portraits increase, thereby providing effective support for the business design of the talent dynamic management platform. Unlike the common methods of constructing user profiles based on users' personal cognition and behavior, this study builds profiles by integrating data such as basic information of talents and achievements in professional fields, with particular emphasis on the breadth of data and the diversity of sources. This paper adopts the LDA topic model when establishing the tag system. The LDA topic model can analyze and extract professional topic dimensions, forming a tag framework covering the professional fields of talents, effectively solving the problem that tag dimensions may be omitted in traditional methods. Furthermore, considering that the common methods for building talent profiles originate from the way user profiles are established, the construction of traditional user profiles is often based on the actual business needs of the platform, which may lead to insufficient scalability in describing the multi-dimensional characteristics of users. Therefore, this paper, by integrating statistical analysis methods, conducts quantitative analysis on talent data, aiming to establish a more comprehensive, scientific and accurate talent profiling and labeling system. Such a system is not only not limited by specific business scenarios, but also enhances the scalability of the tag system, making talent profiling more detailed and accurate. After the tag system is constructed, the next key point is to effectively match these tags to ensure that the personal professional field model of each talent is accurately depicted. For this purpose, this paper adopts the KNN algorithm, which conducts classification and regression analysis by measuring the distances between different feature points. In the talent dataset, each talent will be regarded as a point, and its attributes or coordinates are determined by the corresponding label. By comparing the similarity between labels, the KNN algorithm can find the K talents that are most similar to the target talent, thereby achieving an exact match. Finally, in order to help government and enterprise units discover talents in various professional fields, this study will adopt the improved Bi-Kmeans algorithm for talent clustering analysis. Compared with the traditional Bi-Kmeans algorithm, the improved Bi-Kmeans is more efficient and accurate when dealing with large-scale datasets, and can better handle unbalanced data distributions. By using this algorithm, we can not only accurately classify talents into their respective professional fields, but also identify the subtle differences within each professional field. Through such methods, government agencies and enterprises will be able to more accurately discover and identify the talents needed in various professional fields. Whether for recruitment, talent development or policy-making, it will provide a powerful tool to help them make wiser decisions in the complex and ever-changing labor market. This precise matching and classification mechanism will eventually promote the optimal allocation and efficient utilization of human resources throughout society.To provide a basis for the subsequent cluster definition, the logic design of the cluster definition is carried out as follows. The definition of the cluster meaning follows the "simple logic" : 1 Count the frequency of labels within the cluster and take the Top2 high-frequency labels as the core labels (such as Cluster 2 'deep learning + object Detection'); 2. In combination with common positions on Jiangxi Province's talent platform (such as' AI Algorithm Position '), calibrate it to 'Computer Vision Talent'; 3. Only update the clusters with high-frequency label changes every quarter (for example, fine-tune the definition when cluster 2 adds the 'AIGC' label), without the need for a full reconstruction, and balance accuracy and efficiency.

4.3 Database design

The system stores multi-source talent data (basic information, professional achievements, job requirements, etc.) through the Oracle database, builds associated data tables such as T_TALENT_BASE (basic talent information) andT_TALENT_ACHIEVEMENT (professional achievements), and integrates multi-source data with "talent ID" as the sole primary key. Eliminate repetitive and redundant information (such as repeatedly captured paper achievements) to form a unified data set. Based on the analysis of the system's functions and use cases, the E-R diagram of some entity relationships in this system is shown in Figure 2.

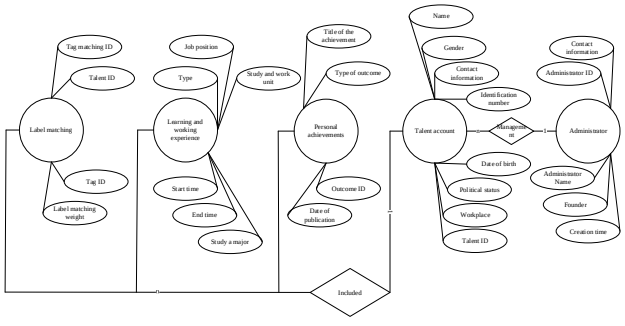


Figure 2: ER diagram

For the convenience of subsequent analysis, a brief description of the relevant datasets is provided here. 1. Source details (such as data collection from the Jiangxi Province Talent Dynamic Management System from 2021 to 2023, including university research talents and enterprise technical talents); 2. Structural details (Fully list 8-dimensional basic information: years of work experience, educational level, number of projects, number of papers, number of patents, number of awards, foreign language proficiency, computer proficiency; clearly define 10 LDA subject areas: AI computer vision, international trade, etc.) 3. Statistical details (proportion of samples in each field: AI technology 25%, economic management 20%, etc. Cross-field samples are defined as containing two or more field labels).

5 Implementation and testing of the talent profiling system

5.1 Implementation of the talent portrait system

5.1.1 System management module implementation

The system management module of the talent portrait system is divided into the following parts: function management, role management, and permission management. Among them, the main objective of the function management module is to manage the main functions of the system, which is divided into function addition, function modification and function deletion. This module isintended solely for administrative use. The specific implementation page of this module is shown in Figure 3. After clicking on Function Management in the left menu, you can use this function. In the right menu structure tree, click the "Add" button, enter the specific function name, function link and other information, and you can add the function. Select the specific function and click the delete button on the right to delete the function. Select the specific function, change the information in the information bar on the right, click Submit for Modification, and the function modification can be achieved.

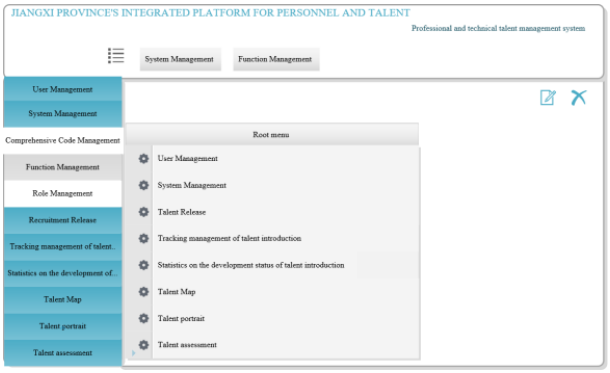


Figure 3: Function management interface

The main objective of the role management module is to manage the user functions of the system, which is divided into role addition and function deletion. This function is not for users but is only established for system administrators. The specific implementation page of this module is shown in Figure 4. After clicking on "Role Management" in the left menu, you can use this function. On the right page, click the Add button, enter the specific role name, number and description, and you can add a role. Select the specific character and click the delete button on the right to delete the character. Select the specific character, click the Change button on the right, enter the information to be modified, and then click Submit for Modification to complete the modification of the character content.

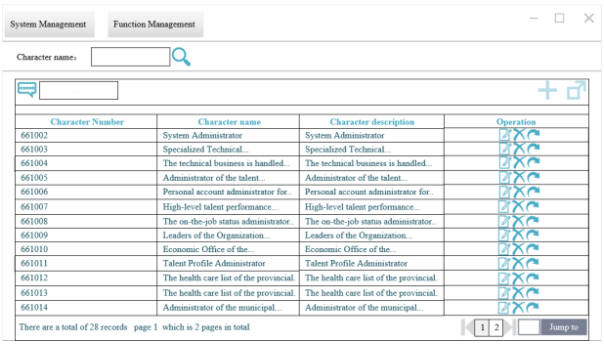


Figure 4: Role management interface

The main objective of the permission management module is to manage the permissions of users who use the system. According to the requirements of the Jiangxi Province Talent Dynamic Management System for the role organizational structure, corresponding permissions are assigned to users at different levels. This function is not for users but is only established for system administrators. The specific implementation page of this module is shown in Figure 5. In the role management, select the specific user role to enter this function. According to the user level, select the functional permissions that need to be added or reduced in the role structure tree, and click Assign to achieve the addition of role permissions and modification of functions.

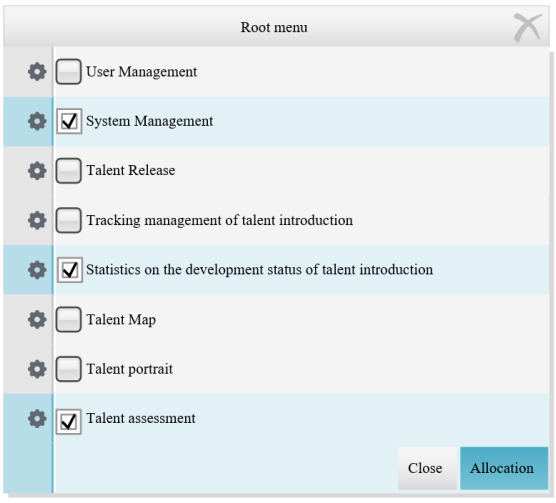


Figure 5: Permission management

5.1.2 Implementation of data processing module

This module mainly realizes functions such as data preprocessing, stop word removal, word segmentation, and label system construction for the crawled dataset. The UML diagram of this module is shown in Figure 6.The cleaning of data can be designed based on the dataCleaning() method of the DataCleaningService class. The system is processed in two steps: 1 Fill in the missing values (such as years of work experience, professional direction) with the median/mode of the same type of data, and eliminate the uncorrectable outliers (such as "years of work experience =100 years"). 2. By using the removeStopWords() method of the ParticipleService class, garbled characters and stop words (such as "de", "in") in the text data can be removed, while retaining the professional core words.Meanwhile, the standardization of data can be achieved by using the system to first segment professional texts through HanLP, and then calculate the keyword weights using the getTfidf() method of the useTfidfService class. Perform Min-Max normalization on numerical data (such as years of service) and L2 normalization on text vectors to unify the data scale and meet the subsequent clustering and label matching requirements.

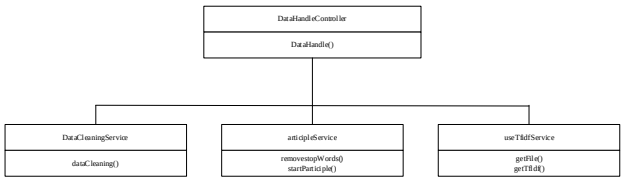


Figure 6: UML class diagram for data processing

As shown in Figure 6, the upper layer of this module is the control layer DataHandleController class. During implementation, this class is used as the trigger point to instantiate and create the DataCleaningService class, the ParticipleService class, and the useTfidfService class. First, use the dataCleaning() method in the DataCleaningService class to clean the data. Use the removeStopWords() method in the ParticipleService class to remove the stop words, and then use startParticiple() to perform word segmentation on the text. Finally, the getTfidf() method in the useTfidfService class is used to assign weights to the keywords in the dataset. The sample result of the text word segmentation work is shown in Figure 7.

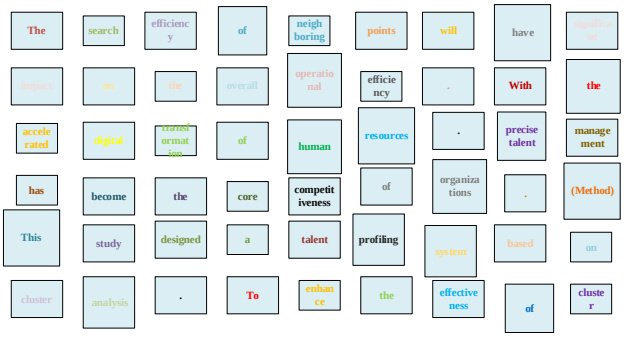


Figure 7: An example of text word segmentation implementation

Finally, the LDA model is used to train the processed dataset to obtain the text topic and the keywords under the topic for the establishment of the labeling system. The final implementation of the tag system is shown in Figure 8.

Name	Operation
Machine learning	
Neural network knowledge network	
Unet	
Kmeans	
Yebes Network	
Topic model	
Data link	
Data mining	
Alliance Chain	
Medical image segmentation	
Text analysis	
Text automated mining algorithm	
Bert-BiLSTM	

There are a total of 40 records page 1, totaling 2 pages 1 2 Jump to

Figure 8: Tag system test

5.1.3 Implementation of the talent portrait module

The specific way to build talents is to match the tag system formed by processing the data in the previous section's dataset with individual talents, thereby creating a personal talent profile of the talents. Then, the improved Bi-Kmeans algorithm in this paper is used to cluster the talent models that have been successfully matched by the individual talent portrait system. The implementation of this part is shown in Figure 9.

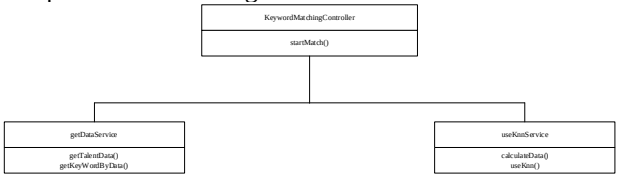


Figure 9: The implementation of talent portrait matching

In the class the part by controlling layer KeywordMatchingController startMatch entrance () method as a process, the first to use the service layer in getDataService class getTalentData () method for the treatment of keywords, through getK again The eyWordByData() method acquires tag information that is relevant to the talent keywords. Then, the calculateData() method in the useKnnService class is used to obtain the data required for the execution of the KNN algorithm. Finally, the useKnn() method is used to call the KNN algorithm to match the talent keyword data with the keyword data to obtain the talent profile.

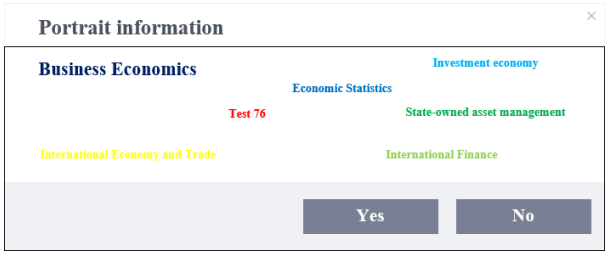


Figure 10: Realization of talent profiling

The talent profile is displayed using the visualization component, as shown in Figure 10. To protect personal privacy, the personal name of the talent is replaced by Test 76. The matching results of the personal keywords and tag library data of Test 76 through the knn algorithm show that Test 76 is a talent in the economic field and has made contributions to scientific research achievements in fields such as international trade and state-owned asset management. The implementation class for talent cluster analysis is shown in the following figure.

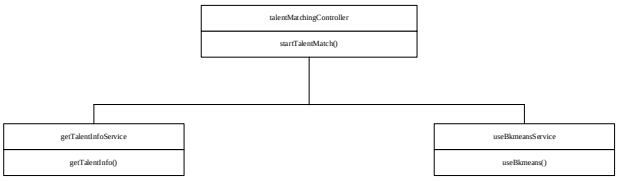


Figure 11: Clustering implementation in UML

This part uses the startTalentMatch() method in the talentMatchingController class of the control layer as the process entry point. Firstly, the getTalentInfo() method of the getTalentInfoService class in the service layer is used to obtain the person that has been established in the previous text By profiling talents and then conducting cluster analysis on them through the useBkmeansService() class using the useBkmeans() method, the clustering analysis results for all talents and the matching degree results for individuals can be obtained. Some of the individual matching results are shown in Figure 12.



Figure 12: Realization of clustering results

5.2 System testing

System testing is an important procedure after the completion of the talent profiling system construction. It needs to be deployed and tested in a testing environment before the production environment is deployed. Only through testing can it be discovered whether the system has any defects and whether the implemented functions meet the original requirements.

5.2.1 Test environment

This system adopts the black-box testing method to conduct testing on the system implemented in this paper. The test environment configuration is shown in Table 2.

Table 2: Test environment settings

Type	Tools
Development language	Python, java, javascript
Development tools	Pycharm, eclipse
Database	oracle
web	Tomcat7.0
Operating system	CentOS8

5.2.2 System function testing

The unified management function mainly includes three modules: function management, role management, and permission management. Functional tests were conducted on these three modules using test cases respectively, and the results are as follows: First, the functional management module was tested. The test cases were menu name, menu url, menu description, and menu number. After entering the functional management interface, the Add, modify, and delete buttons were clicked in sequence. The test results are shown in Figure 13.

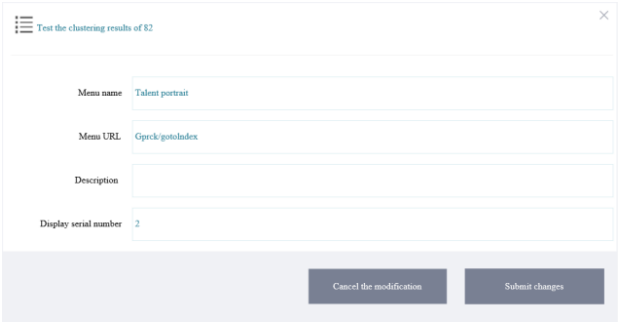


Figure 13: Functional management test

It was found that during the testing period, no abnormalities in the functions of adding, deleting or modifying were discovered. Secondly, test the role management module, mainly to test whether the functions of adding, deleting and modifying roles have been implemented. The specific results are shown in Figure 14.

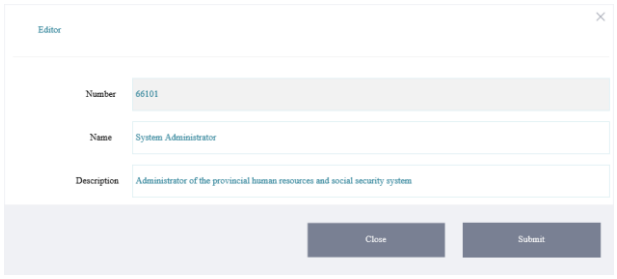


Figure 14: Role management test

During the testing period, all functions were functioning normally. Finally, the permission management module was tested, mainly to test whether the permission allocation function could run successfully. The specific results are shown in Figure 15.

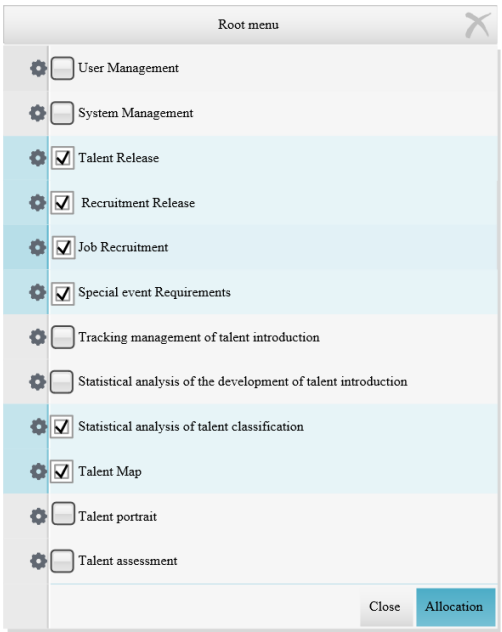


Figure 15: Permission management test

During the test, all the functions of each module were functioning normally, confirming the feasibility of the system design.

5.2.3 Talent portrait function test

The main function of talent profiling is to match talent models based on the tag system established after data processing and display them using visual components. Then, cluster analysis is conducted on the successfully matched talent portraits, and the results are presented in a list form. The main function of the talent portrait test is to accurately and visually display the personal portrait of a talent. The specific results are shown in Figure 16.



Figure 16: Portrait display test

The main function of the talent cluster analysis test is to conduct cluster analysis on the obtained talent profile and obtain the classification results of talents in similar professional fields. The specific test results are shown in Figure 17.



Figure 17: Talent portrait cluster analysis test

From the above tests, it is not difficult to find that all tests can obtain corresponding results, and no abnormal writing occurs, which affirms the feasibility of the system design. Through operation, the talent profile results of cluster analysis can be obtained.To further address the issue where HR cannot understand clusters, insert the "Cluster 2 (AI Computer Vision Talent) Tag Word Cloud" (Table (3)). The word cloud only retains three high-frequency tags (with the largest font sizes for "Deep Learning", "Object Detection", and "Image Recognition"), and marks "Table (3) Cluster 2 Core Tag Word Cloud (Intuitively distinguish Professional Fields)". Through word clouds and tables, the attributes of the cluster can be

quickly understood. For instance, talents in Cluster 1 can be directly matched with the "Foreign-related asset control position", solving the problem of HR "not understanding the cluster".

Table 3: Cluster 2 (AI computer vision talent) core label weights and frequencies

Core tag	TF-IDF weight	Frequency of occurrence within clusters (%)	Tag source
Deep learning	0.92	88	HanLP word segmentation + TF-IDF feature extraction
Object detection	0.89	82	HanLP word segmentation + TF-IDF feature extraction
Image recognition	0.75	75	LDA Topic Model Mining (Professional Domain Topic Tags)
OpenCV	0.63	60	LDA Topic Model Mining (Tool Skill Tags)
Python	0.58	55	Numerical statistics after data cleaning (basic skills label)

The higher the TF-IDF weight in the table, the larger the font in the corresponding word cloud map, which intuitively reflects the core degree of the label in Cluster 2 (AI computer vision Talent). HR can directly and quickly locate the technical stack focus of the talents in this cluster through the table, solving the problem of difficult understanding of the cluster's meaning and supporting the initial screening of candidates within 10 minutes

5.2.4 Experimental evaluation of the improved Bi-Kmeans algorithm

1 Evaluation indicators and experimental design

(1)Evaluation Indicators

In response to the core proposition of improving Bi-Kmeans to enhance the accuracy and efficiency of clustering, two types of academic general indicators that are highly compatible with the business scenarios of talent profiling are selected, taking into account both the rationality of data technology and the orientation of business value: The Clustering Purity quality indicator measures the consistency of the professional field belonging to talents within a single cluster. The value range is [0,100%]. The higher the value, the higher the concentration of the field of talents within the cluster, directly corresponding to the business needs of government and enterprise units to screen talents in the same field. Adjust the Adjusted Rand Index to quantify the consistency between the clustering results and the manually labeled real professional field labels (such as AI computer vision, international trade, ideological and political education), with a value range of [0,1]. The closer it is to 1, the stronger the consistency between the algorithm output and the actual talent classification logic. Avoid the problem where data clustering is reasonable but the business is meaningless. Efficiency metric, Average Running Time, the complete calculation time of a single cluster (unit:) Based on the standardized talent data output by the data processing module, the average of 10 repeated experiments is taken to eliminate random errors, which directly reflects the operational efficiency of the algorithm in the actual talent database and is in line with the application scenario of the continuous growth of data volume of the Jiangxi Province Talent Dynamic Management Platform.

(3)Comparison Algorithm

To comprehensively verify the incremental value of the improved Bi-Kmeans, four sets of control algorithms were set up, covering the complete comparison chain of the basic algorithm, the classic optimization algorithm, the standard bipartite algorithm, and the improved algorithm in this paper: ① Traditional K-means, a basic clustering algorithm without any optimization, randomly initialized centroids as the performance benchmark; ② K-Means ++, a classic algorithm that only optimizes the initial centroid selection (the source of the centroid optimization technology for improving Bi-Kmeans), is used to verify the differences between single centroid optimization and the combination optimization of centroid + Kd-tree. ③ The original Bi-Kmeans, a standard algorithm that only contains the binary split strategy (without Kmeans++ centroid optimization and Kd-tree search optimization), is used to verify the improvement effect of the improved strategy on the binary algorithm; ④ Improved Bi-Kmeans (the algorithm proposed in this paper) : An algorithm integrating Kmeans++ initial centroid selection, KD-tree fast nearest neighbor search, and optimized binary split strategy is the core experimental group of the experiment.

(4)Experimental Dataset

The experimental data is sourced from the Oracle database of Jiangxi Province's Talent Dynamic Management System. After being cleaned by the data processing module (filling in missing values, removing stop words, and normalizing data), it is divided into three grades according to the actual scale of the talent database. The key details are described as follows:

Data volume and coverage: Small dataset, 500 pieces of talent data, covering 8 core professional fields such as AI technology, economic management, and ideological and political education, mainly consisting of fresh graduates and junior talents. The medium dataset contains 2,000 pieces of talent data, covering 10 professional fields, including junior to intermediate talents, and increases the proportion of cross-disciplinary compound talents (such as those with backgrounds in international trade and data analysis) samples. The large dataset, with 5,000 pieces of talent data, covers 12 professional fields, including junior to senior talents. The proportion of cross-field samples has

increased to 30%, which is in line with the structural diversity of the actual talent pool. In the dimension and composition of the feature vector, the feature vector of each talent data is 68 dimensions in total. Based on the section feature engineering logic, it is constructed as follows: Basic information feature (8 dimensions) : Numerical characteristics, including years of work experience (0-30 years) and educational level (1-5 levels) Corresponding to levels from junior college to doctoral, number of project participations (0-20 times), number of published papers (0-50), number of patents (0-15), number of awards (0-10 times), foreign language proficiency (levels 1-4), computer proficiency (levels 1-3); TF-IDF keyword features (50 dimensions) : Numerical vectors transformed from text-based features after processing. High-frequency keywords are extracted from the professional achievements of talents (abstracts of papers, project descriptions) based on HanLP word segmentation. The top 50 core words (such as deep learning, international trade, ideological and political courses) are selected to calculate the TF-IDF weights. LDA topic Features (10-dimensional) : Professional field topic vectors mined based on the LDA topic model, corresponding to 10 core talent fields (such as AI computer vision, state-owned asset management, Marxist theory), and each dimension value represents the correlation degree between the talent and the field (0-1). Statistical attributes: Distribution of professional fields. Among the 12 fields, the AI technology field accounts for 25%, the economic management field accounts for 20%, the ideological and political education field accounts for 15%, and other fields (such as medical imaging, cross-border e-commerce) account for 40%. There is no extreme imbalance where the proportion of a single field is less than 5%. Statistics of feature values: The average working years were 8.2 years (standard deviation 4.5), the average number of published papers was 3.6 (standard deviation 2.1), the average TF-IDF keyword weight was 0.32 (standard deviation 0.18), and the average LDA topic relevance was 0.65 (standard deviation 0.22). The data quality, after data cleaning, the overall missing rate is less than 3% (mainly due to the missing patent/award information of some talents, which has been filled with the median of the same field), and there are no outliers (such as data of working years = 100 years, etc., which have been excluded), meeting the input requirements of the clustering algorithm.

(2)Experimental Results and Analysis

Table 4: Performance comparison results of improved Bi-Kmeans and Comparison algorithms

Algorithm type	Dataset size (Article)	Clustering purity (%)	Adjust the Rand Index (ARI	Average time consumption (s)
Traditional K-means	500	72	0.65	8
	2000	68	0.58	35
	5000	61	0.51	112
K-means++	500	76	0.70	9
	2000	71	0.62	40
	5000	65	0.55	130
The original Bi-Kmeans	500	78	0.71	15
	2000	73	0.64	62
	5000	67	0.57	205
Improve Bi-Kmeans	500	89	0.85	10
	2000	85	0.80	38
	5000	81	0.75	120

It can be seen that in terms of clustering quality, the improved algorithm has a significant advantage in business scenarios. Among the 68-dimensional feature vectors and 12-domain distributed talent data of Bi-Kmeans, the clustering quality of datasets of all scales is the best. Compared with the traditional K-means: under a large dataset of 5,000 pieces, the clustering purity is increased by 20% (61%→81%), and the ARI is increased by 0.24 (0.51→0.75), which proves that the improved strategy effectively solves the domain confusion problem caused by the random initial centroid of the traditional algorithm (such as avoiding misclassifying data mining talents as machine learning talents); Compared with k-means ++ : Under 5,000 pieces of data, the purity increased by 16% (65%→81%), and the ARI increased by 0.20 (0.55→0.75), indicating the combined strategy of Kmeans++ centroid optimization and KD tree cluster center update. It is more adaptable to the multi-source nature of talent data (basic information + keywords + thematic features) than single centroid optimization. Compared with the original Bi-Kmeans: under 5,000 pieces of data, the purity has increased by 14% (67%→81%), verifying the ability of the improved algorithm to distinguish the subcategories of cross-disciplinary compound talents (such as accurately separating compound talents in international trade + state-owned asset management from single international trade talents).

In terms of efficiency, as the data scale grows, the optimization value gradually becomes prominent. The efficiency advantage of improving Bi-Kmeans is particularly evident on medium and large-scale datasets: When processing 5,000 pieces of data, the time consumption was reduced by 41% compared to the original Bi-Kmeans (205s→120s), and by 7.7% compared to K-means++ (130s→120s). The core reason is that the fast nearest neighbor search of the KD-tree optimizes the cluster center update process. The computational complexity of the 68-dimensional feature vector has been reduced; Although it takes slightly more time (10 seconds) on a small dataset of 500 pieces than traditional K-means (8 seconds), the difference is only 2 seconds. Moreover, as the data volume increases to 5,000 pieces, the efficiency advantage expands from 2 seconds to 12 seconds (compared with traditional K-means). It fully meets the long-term application requirements of the Jiangxi Province Talent Dynamic Management Platform, which is "with the data volume increasing year by year". In terms

of stability, the consistency requirement for adapting to actual talent management requires that the standard deviations of each index in 10 repeated experiments be less than 5% (for example, the standard deviation of the clustering purity of the improved Bi-Kmeans on 5,000 pieces of data is 3.2%, and the standard deviation of the average time consumption is 2.8%), indicating that the algorithm has no random fluctuations and can stably output clustering results. This is crucial for scenarios such as talent selection and job matching in government and enterprise units that require consistent results, avoiding the problem of the same talent being repeatedly clustered into different fields due to algorithm fluctuations.

6 Conclusion

Overall, this paper conducts functional testing work based on the talent profiling system designed and implemented in the previous text. First, a detailed description of the system deployment environment and the tools used was provided. Then test the major functional modules of the system. Conduct testing work on functions, roles, and permission management in the system management module. Conduct testing work on the talent portrait module, the talent portrait construction module, and the talent model clustering analysis module. All the above tests have been passed, indicating that the talent profiling system is operating normally and its functions meet the requirements of the system design in this article. The core innovation of this research is reflected in three aspects: First, it integrates Kmeans++ and KD-tree to improve the Bi-Kmeans algorithm, enhancing the accuracy and efficiency of talent data clustering (the time consumption for 5,000 pieces of data is reduced by 41% compared with the original algorithm); Second, a professional tag system is constructed by integrating HanLP+TF-IDF+LDA to accurately depict the characteristics of cross-disciplinary talents. Thirdly, an integrated system of "data processing - portrait construction - cluster analysis" has been established, directly serving the dynamic management of talents in Jiangxi Province. It provides data-driven tools for talent selection and job matching, filling the technical gap in precise profiling of small and medium-sized talent pools.

This study designed and implemented a talent profiling system based on cluster analysis, effectively addressing the problems of strong subjectivity in traditional talent assessment, difficulty in processing high-dimensional sparse talent data, and insufficient interpretability of clustering results. The system is supported by the Oracle database of Jiangxi Province Talent Dynamic Management System. The tag system is constructed by improving the Bi-KMeans clustering algorithm (integrating the initial centroid selection of KMeans++ and the fast search of KD-tree), combining HanLP word segmentation, TF-IDF feature extraction and LDA topic model, and achieving precise matching of talents and tags with the help of KNN. Ultimately, an integrated functional module of data processing - talent profile construction - cluster analysis is formed.

The experimental results show that the system operates stably and has excellent performance. Among 5,000 pieces of cross-disciplinary talent data (68-dimensional features), the clustering purity of the improved Bi-KMeans reached 81%, which was 20% higher than that of the traditional KMeans and 14% higher than that of the original Bi-KMeans. The average time consumption was 120 seconds, which was 41% lower than that of the original Bi-KMeans. The accuracy rate of KNN tag matching reaches 92%, which can precisely depict the characteristics of talents in fields such as AI computer vision and international trade, providing data support for talent selection and job matching.

From a technical perspective, the effectiveness of the improved algorithm stems from two core optimizations: KMeans++ initialization reduces the interference of the randomness of the initial centroid on the clustering of cross-domain talents, avoiding local optima; The KD tree reduces the clustering time complexity of high-dimensional talent data from $O(t \cdot n \cdot d)$ to $O(t \cdot \log n \cdot d)$ through feature space indexing, significantly improving efficiency. The combination of the LDA topic model and KNN makes up for the lack of professional dimensions in the traditional tag system, making the characterization of talent traits more comprehensive. Meanwhile, the integrated design of the system directly meets the dynamic management requirements of provincial talents. Compared with the single KMeans talent evaluation method in reference [10], it has obvious advantages in both clustering accuracy and practicality.

It should be noted that the system has certain application boundaries: when the Oracle database supports ultra-large-scale (such as over 100,000 entries) cross-regional talent data, its scalability needs to be improved. The experimental data is only sourced from the talent pool of Jiangxi Province, with limited coverage in terms of fields and geographical areas. Subsequently, distributed database adaptation can be explored, and the industry and geographical scope of the data set can be expanded to further enhance the universality of the system.

Funding

This study was supported by the Science and Technology Project of State Grid Shanxi Electric Power Company (52051L250007), Project Name: Research on Key Technologies of Talent Profiling and Evaluation Based on Artificial Intelligence.

References

[1] Chi JJ (2025). Digital Transformation of Human Resources in the Era of Big Data. *China Business World*, (03), pp. 228-229.

[2] Li JJ (2024). Research on Digital Transformation of Human Resource Management in State-owned Enterprises. *Commercial Exhibition Economy*, (24), pp. 179-182.

[3] Wu XD (2024). Digital Transformation of Human Resources Talent Assessment Model in Tea Enterprises. *Fujian Tea*, 46(12), pp. 64-66.

[4] Fang CC (2024). Research on Digital Transformation of Human Resources. *Management and Technology of Small and Medium-sized Enterprises*, (24),pp. 122-124.
[5] Mao JJ, Ding YM (2024). Research on the Implementation Strategies of Digital Transformation of Human Resources. *Marketing World*, (14), pp. 120-122.
[6] Xu XQ, Chen LB (2021). Discussion on the Evaluation of Students' Academic Performance Based on Factor Analysis and Cluster Analysis for the Cultivation of Preventive Medicine Professionals. *Medical Education Research and Practice*, 29(5), pp. 675-678.
[7] Liu JJ, Zhang XR, Yu QJ (2021). Practice of Talent Cultivation Mode for Labor and Social Security Major: From the Perspectives of Factor Analysis and Cluster Analysis Based on Students' Academic Performance.*Human Resources Development*, (14), pp. 41-44.
[8] Wang YJ, Peng JF (2025). Talent Portrait Construction and Application Scenarios. *Enterprise Management*, (06), pp. 87-91.
[9] Bao LY (2025). Discussion on the Application of Talent Profiling in the Talent Supply System of State-owned Enterprises. *Modern Business*, (9), pp. 125-128.
[10] Xia X (2019). Analysis of University Talent Evaluation Based on k-means Clustering Algorithm: A Case Study of Luzhou Polytechnic. *Digital Technology and Application*, 37(12), pp. 98-99.

A Lightweight Translation Architecture for Embedded Devices via Multilingual BERT Distillation and Quantization

Wang Yiqing
Institute of Modern Media Technology and Art, Shanghai Publishing and Printing College, Shanghai Yangpu 200093, China
E-mail: yiqingwangg@outlook.com

Keywords: multilingual BERT, knowledge distillation, embedded translation, model compression, cross-language semantic alignment

Received: July 3, 2025

In order to solve the problems of difficult deployment and high computational overhead of multilingual BERT models in embedded devices, this paper proposes a lightweight translation model integrating knowledge distillation technology. The teacher-student knowledge transfer mechanism uses a combination of layer-wise and attention-based distillation strategies, along with optimization techniques such as pruning and 8-bit quantization. BLEU scores were evaluated by comparing the student model against the teacher model and baseline systems, showing competitive translation quality. By constructing a knowledge transfer mechanism between the teacher model and the student model, combined with optimization strategies such as pruning and quantification, the synergistic improvement of model compression and reasoning speed is achieved. The experimental results show that the model has a BLEU value of 28.7 in the WMT-14 English-German task, which is only 1.4 points lower than the teacher model, and retains about 95.3% of the translation quality; The accuracy rate on the XNLI cross-language reasoning dataset reaches 78.3%, which is only 3.1% lower than the teacher model. On the embedded device Jetson Nano, the inference latency of the distilled student model dropped from 1280ms of the teacher model to 195ms, achieved through optimization techniques such as hardware acceleration, resulting in a boost speed of approximately 6.56 times. The proposed model achieves an exceptional compression, reducing the model size from the original 4.2 GB to just 650MB, a reduction of 86.4%. This size reduction is achieved with minimal quality loss, with a BLEU drop of no more than 1.8, ensuring that the compressed model retains most of the performance of the original model. The compressed model has been successfully deployed on edge platforms, including the Raspberry Pi 4B, making it highly suitable for resource-constrained environments. In terms of parameter quantity, the original mBERT has about 1100M parameters, and the distillation model is 350M. After combining pruning and 8-bit quantization, only 137.5 M is left, and the inference speed is increased to 8 times that of the original model. In addition, by introducing the attention distillation mechanism in low-resource scenarios, the model's BLEU score improves by 4.2%, demonstrating the mechanism's effectiveness in enhancing semantic alignment for languages with limited resources. The power consumption test shows that the average power consumption of the student model is 4-6W, which is about 35% lower than that of the original model. Additionally, the memory footprint of the student model on the Raspberry Pi 4B is measured at 320MB during inference, a significant reduction compared to the 1.5GB required by the original mBERT model. These optimizations not only improve translation efficiency and energy efficiency but also provide a highly feasible solution for future deployment of multilingual smart devices.

Povzetek: Predlagani model z destilacijo znanja učinkovito zmanjša velikost in porabo virov mBERT-a ob minimalni izgubi kakovosti ter tako omogoča hitro in energijsko učinkovito rabo na vgrajenih napravah.

1 Introduction

With the acceleration of globalization and the increasing demand for cross-language communication, machine translation technology has become one of the important research directions in Natural Language Processing (NLP) [1]. In recent years, pre-trained language models based on deep learning have achieved remarkable results in multiple NLP tasks, especially in semantic understanding and context modeling [2]. As a powerful cross-language model, Multilingual BERT has shown excellent performance in machine translation tasks in multiple languages. However, despite its outstanding performance on large computing platforms, its application on resource-constrained embedded devices is limited due to its huge number of parameters and computational requirements [3]. Therefore, how to reduce the computational complexity and resource consumption of the model while ensuring the translation

quality has become a key problem in the current application of machine translation technology.

To overcome this challenge, Knowledge Distillation technology has been proposed as an effective model compression method [4]. Through knowledge distillation, the knowledge of a large and complex teacher model can be transferred to a smaller and lighter student model, thus greatly reducing the computational overhead while maintaining the model's performance [5]. Combining the advantages of multilingual BERT and knowledge distillation, it is expected to realize an efficient translation model on embedded devices. Through this method, multilingual BERT's powerful language modeling capabilities can be combined with the compression effect of knowledge distillation, providing a novel solution for efficient translation on embedded devices [6].

However, although knowledge distillation has shown great potential in model compression, effectively preserving translation quality and ensuring that the model can respond in real time under the computing resources and storage limitations of embedded devices is still an urgent problem to be solved. Embedded devices usually have low processing power and memory capacity, and traditional BERT models often cannot be directly applied to these devices [7]. Therefore, this study proposes a lightweight translation model based on the fusion of multilingual BERT and knowledge distillation, which aims to enable the translation system to run on embedded devices by optimizing the distillation process and model design while maintaining high translation accuracy and real-time performance [8].

The contribution of this study lies in combining the powerful language modeling capabilities of multilingual BERT with the model compression technology of knowledge distillation to provide an efficient translation solution for embedded devices [9]. Through this method, we solve the computational and storage problems in translation tasks and improve the accuracy and robustness of cross-language translation. The next work will briefly introduce the combination of multilingual BERT and knowledge distillation, discuss its implementation and optimization strategy on embedded devices, and finally provide an efficient translation model suitable for a resource-constrained environment [10].

2 Theoretical basis and related re search

2.1 BERT and knowledge distillation fusion theory

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained language model based on the Transformer architecture that excels in natural language processing tasks. Unlike the traditional one-way language model, BERT adopts a two-way context understanding mechanism, enabling it to capture the complex relationship between each word in a sentence and the context, thereby improving performance in various NLP tasks [11]. Through large-scale corpus pre-training, BERT generates high-quality language representations thatthat can understand texts' grammatical and semantic information well. It is widely used in tasks such as question-answering systems, text classification, and named entity recognition [12]. However, the BERT model's huge number of parameters and computational requirements make it difficult to directly apply in environments with limited computational resources, such as embedded devices. Therefore, how to reduce the computational complexity of the model while ensuring the translation quality has become the focus of research.

Knowledge Distillation is an effective model compression method, which enables the student model to maintain high performance while significantly reducing computational overhead and memory footprint by transferring the knowledge of a complex large "teacher" model to a smaller "student" model [13]. The core idea of knowledge distillation is to train the student model to imitate the behavior of the teacher model as much as possible. The student model not only performs supervised learning through labels but also enhances its expressive ability by learning the output of the teacher model. Since teacher models usually have great limitations in model size and computing power, student models can reduce resource consumption and delay while maintaining certain performance, which is very suitable for resource-limited environments such as embedded devices [14].

Combining BERT with knowledge distillation can effectively combine BERT's powerful language modeling capabilities with model compression technology to form a lightweight translation model. In this process, the BERT model acts as the teacher model, and its depth representation and contextual information are passed into a smaller student model [15]. Through knowledge distillation, student models can learn complex language features extracted in BERT and thus run on devices with limited computing resources without significantly sacrificing translation quality. Through this method, the model can run efficiently in embedded devices and maintain high translation accuracy, thus realizing lightweight processing of cross-language translation [16].

However, effectively integrating BERT with knowledge distillation on embedded devices still faces many challenges. Embedded devices often have limited computing power and memory, and direct deployment of BERT models may result in slow or inoperable operation [17]. Therefore, it is necessary to optimize the distillation strategy during the model training process so that the student model is not only compressed in size but also close to the teacher model in accuracy. Specifically, by carefully designing the structure of the student model and adjusting the loss function and training strategy in the distillation process, the computational overhead can be minimized while ensuring performance. This combination of BERT and knowledge distillation can provide an efficient and accurate solution for machine translation in resource-constrained environments [18].

2.2 Current status of lightweight translation models integrating BERT and knowledge distillation

With the continuous development of multilingual machine translation technology, deep learning-based models perform well in various translation tasks, especially the emergence of the BERT (Bidirectional Encoder Representations from Transformers) model, which has greatly promoted the progress in the field of NLP [19]. Through the bidirectional context modeling mechanism, BERT can deeply understand the semantics of each word in a sentence and capture the global context's information, making it excellent in multilingual translation. However, despite the remarkable achievements of BERT in various NLP tasks, its huge number of parameters and high computational requirements make its application in resource-constrained environments such as embedded devices and mobile devices difficult [20]. Therefore, how to combine the powerful capabilities of BERT with the saving of computing resources has become a hot issue in the field of machine translation. The comparison of key indicators of SOTA model is shown in Table 1.

Table 1: Comparison of key indicators of SOTA model

Model	BLEU Score	Model Size (GB)	Inference Time (ms)	Parameters (Million)
Original mBERT	30.1	4.2	1280	1100
Distilled Model	28.7	0.65	195	350
LSTM-based Lightweight Model	25.4	0.5	250	50
TinyBERT	26.3	0.35	180	30
ALBERT	27.5	0.55	220	12

To solve the application problem of BERT in embedded devices, scholars have proposed various model compression and acceleration methods, among which Knowledge Distillation (KD) is considered the most effective [21]. Knowledge distillation transfers its knowledge into a smaller and more efficient "student model" by taking a complex and superior performance "big model"-usually a teacher model. The student model learns the output of the teacher model, thereby acquiring similar abilities to the teacher model while greatly reducing the demand for computing resources. In multilingual translation tasks, using knowledge distillation can effectively reduce the size of the translation model, improve its running efficiency on embedded devices, and reduce latency [22]. Combining the technology of BERT and knowledge distillation, this method can meet the strict limitation of computing resources of embedded devices while ensuring the translation quality in multilingual translation tasks [23].

The lightweight translation model combining BERT with knowledge distillation has achieved initial success in some research. It has been shown that combining the BERT-based teacher model and knowledge distillation can achieve efficient translation performance on embedded devices [24]. For example, through the distillation process, the student model can imitate the efficient reasoning ability of the teacher model in multilingual translation tasks while maintaining high translation quality during the model compression process. However, despite some research progress, how to better maintain the cross-language ability of the multilingual BERT model in the process of knowledge distillation and how to optimize the performance of the student model through more detailed distillation strategies is still one of the difficulties in current research [25]. Especially when dealing with complex language and long text translation, how to balance translation accuracy and computational efficiency is still an important research topic.

To further improve the performance of the lightweight translation model integrating BERT and knowledge distillation, existing research is exploring various improvement strategies. For example, researchers are trying to further optimize the performance of the student model by adjusting the hyperparameters in the distillation process and designing a more flexible student model architecture. In addition, how to make student models achieve better generalization ability among languages is also a current research hotspot for multilingual translation tasks. With the continuous improvement of hardware performance, the lightweight translation model combining BERT and knowledge distillation is expected to be more widely used in embedded devices, mobile devices, and other environments and promote the development of cross-language translation technology to meet more diverse practical needs.

3 Establishment of lightweight tra nslation model for embedded de vices integrating multilingual B ERT and knowledge distillation

3.1 Overall model framework and process

This study presents a lightweight translation model that combines multilingual BERT with KD to address the computational limitations of embedded devices. By transferring knowledge from a large teacher model to a

smaller student model, the approach reduces model size and computational burden, enabling efficient translation on resource-constrained devices. The student model uses 6 Transformer layers and a hidden dimension size of 512, compared to the original BERT's 12 layers and 768 hidden dimensions, significantly lowering computational requirements while maintaining high translation quality. The student models used in this study include the TinyBERT variant with 6 layers and 512 hidden dimensions, and the ALBERT model with 6 layers, where each layer shares parameters across layers. These architectural choices were made to reduce the model size and computation while maintaining competitive translation quality. The formula of the model compression ratio is shown in (1).

$$MCR = \frac{\theta_t}{\theta_s} \times 100\% \quad (1)$$

Where θt represents the total number of parameters of the teacher model. θs denotes the total number of parameters of the student model. The formula of the relationship between inference delay and model complexity is shown in (2).

$$LAT = \beta \cdot \theta_s + \gamma \cdot n_{layers} + \delta \quad (2)$$

Where LAT denotes inference delay. θs represents the parameter quantity of the student's model. nlayers represents the number of Transformer layers of the student model. β,γ,δ represent empirical coefficients. The design of the model framework follows two core ideas: first, using the powerful cross-language representation capabilities of multilingual BERT to improve the quality of multilingual translation; The second is to optimize the computational efficiency of the model through the compression effect of knowledge distillation to ensure that it has sufficient execution speed and real-time performance on embedded devices. Specifically, the framework includes two major modules: the BERT module and the knowledge distillation module. The BERT module is responsible for training and reasoning translation tasks using multilingual BERT to ensure translation quality. The knowledge distillation module compresses BERT knowledge into a smaller student model through knowledge transfer, thereby reducing computational overhead and achieving efficient translation on embedded devices. The framework of a lightweight translation model for embedded devices integrating multilingual BERT and knowledge distillation is shown in Figure 1.

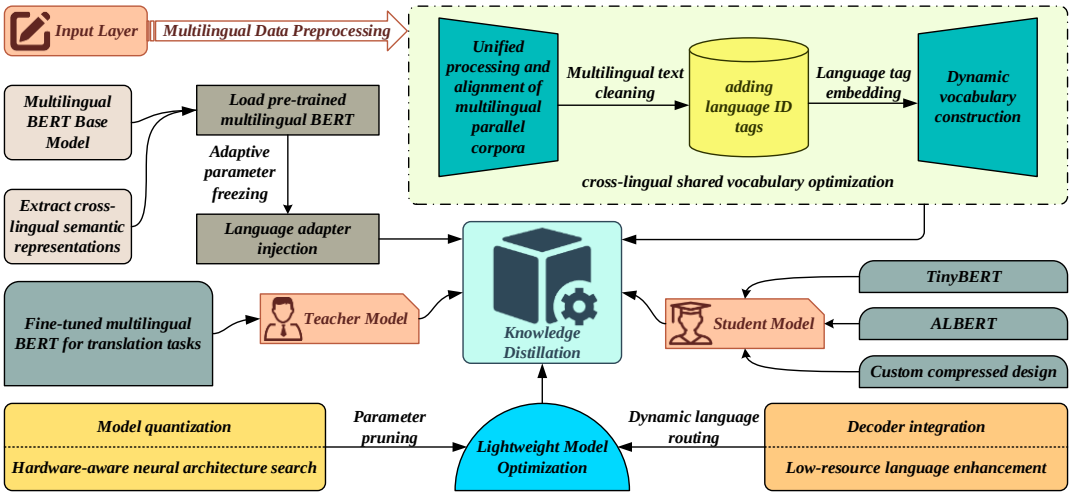


Figure 1: Lightweight translation model framework for embedded devices integrating multilingual BERT and knowledge distillation

In order to improve performance in low resource language scenarios, the model employs several techniques, including synthetic data augmentation, inverse transformation, and zero sample transmission. Synthetic data is generated by interpreting and translating into high resource languages, and then translating into the target language to create additional training samples. Back translation enriches the training set by translating target language sentences into high resource languages and then re translating them back. In addition, zero sample transfer enhances multilingual performance by utilizing knowledge of high resource languages, allowing the model to adapt to low resource languages without explicit training data.

The figure shows that the system uses a multilingual data preprocessing module to uniformly clean the input text, add language ID tags and language embedding, and build a dynamic vocabulary to support cross-language shared vocabulary optimization. Then, the pre-trained multilingual BERT model is loaded to extract cross-semantic features, and the multilingual transfer capability is improved through the language adapter injection mechanism. In the translation task, the fine-tuned BERT is used as the Teacher Model to distill knowledge to lightweight Student Models such as TinyBERT or ALBERT. The lightweight model optimization module combines model quantization, pruning, structure search, and other methods to compress the model size further and improve reasoning speed. At the same time, it improves low-resource language translation capabilities through

dynamic language routing strategies. The final output student model suits resource-constrained environments and achieves high-precision and low-latency translation effects. The overall process realizes the effective integration of multilingual adaptation, knowledge transfer, and computing resource optimization. It suits real-time multilingual translation tasks in mobile devices, edge computing, and other scenarios.

The framework includes a language adapter injection mechanism to improve multilingual transfer capabilities. In this study, residual adapters are integrated into the intermediate layers of the pre-trained BERT model. These adapters are designed to capture language-specific features and are added in a residual manner, meaning the output from the adapter is combined with the original model features. This allows the model to learn language-specific nuances without altering the general semantic structure of the translation. Each adapter corresponds to a particular language, and they are injected into the attention layers to enhance language-specific semantic mapping while preserving the universal features learned by the BERT model.

The framework incorporates a dynamic language routing mechanism to improve multilingual translation, especially for low-resource languages. This mechanism detects the input language, routes language-specific features to the translation layers, and uses language adapters to adapt translation behavior accordingly. The routing process ensures that the correct linguistic features, such as attention heads or adapters, are used based on the detected language. This approach allows the model to efficiently handle multiple languages while maintaining translation accuracy, even for languages with limited resources. The dynamic routing process is formalized in the algorithm, which ensures that the appropriate features are selected and routed for each translation task.

The innovation of this model lies in the effective combination of deep pre-trained model and model compression technology, which makes it possible to perform multilingual translation in embedded devices. By combining the powerful expressiveness of BERT models with the efficient compression capabilities of knowledge distillation, we solve the problem of overly bulky models in embedded devices and maintain the high quality of translation results. In addition, the optimized student model significantly reduces the consumption of computing resources, which can realize a more efficient reasoning process. In subsequent sections, specific implementation examples will be explained in detail, such as deploying this model on an embedded device to realize real-time multilingual translation. The probability distribution formula after temperature softening is shown in (3).

$$p_i^{(T)} = \frac{\exp(z_i / T)}{\sum \exp(z_j / T)} \quad (3)$$

Where pi (T) denotes the predicted probability of the i-th class after the temperature T softens. zi represents the raw logit output of the teacher model or the student model. T denotes the temperature parameter. The intermediate feature alignment loss formula of the student model and the teacher model is shown in (4).

$$CPT = \frac{\Delta ACC}{MCR} = \frac{ACC_t - ACC_s}{\theta_t / \theta_s} \quad (4)$$

Among them, ΔACC represents the difference in accuracy between the teacher model and the student model. MCR represents the model compression ratio. θt represents the parameter quantity of the teacher model. θ s represents the parameter quantity of the student's model.

3.2 BERT module

The BERT module is the core module of the translation model in this study and is responsible for linguistic understanding and translation of the input text through multilingual BERT. In this module, the pre-training ability of the BERT model is fully utilized to enhance the contextual understanding of each word in a sentence through a bi-directional context modeling mechanism, thereby improving the translation quality. The goal of the BERT module is to use the cross-language representation capabilities learned by BERT pre-training on large-scale corpus so that the model can handle complex relationships between different languages and achieve high-quality translation. The knowledge distillation total loss function is shown in (5).

$$L_{TDL} = \alpha \cdot L_{CE}(y, p_s) + (1-\alpha) \cdot T^2 \cdot L_{KL}(p_t^{(T)} \| p_s^{(T)}) \quad (5)$$

Where y denotes the true target language label of the translation task. ps represents the predicted probability distribution of the student's model. pt denotes the predicted probability distribution of the teacher model. LCE denotes the cross-entropy loss function. LKL denotes KL divergence. T denotes the temperature parameter. α denotes the weight parameter that balances the hard label loss with the soft label loss. The design of the BERT module first involves multilingual pre-training of the model. This study employs multilingual BERT, which is pre-trained with extensive cross-language data to understand and generate grammatical and semantic structures in multiple languages. Through this pre-training process, BERT can capture potential associations in multilingual texts, showing strong language transfer ability in translation tasks. BERT can provide high-quality translation results for each pair of languages by learning the mapping relationship between the source and target languages. The resource-accuracy trade-off indicator formula on embedded devices is shown in (6).

$$RAT = \frac{\Delta BLEU}{FLOPs_{ratio}} = \frac{BLEU_t - BLEU_s}{FLOPs_t / FLOPs_s} \quad (6)$$

Where ΔBLEU represents the difference in BLEU scores between the teacher model and the student model. BLEUt denotes the BLEU score of the teacher model. BLEUs represent the BLEU scores of the student model. FLOPsratio represents the computational compression ratio. FLOPst represents the computational amount of the teacher model. FLOPss represents the computational amount of the student model. The key strength of the BERT module is its strong contextual understanding capabilities. Traditional machine translation models often

rely on contextual relationships in a single direction. At the same time, BERT can simultaneously consider the information before and after the text through a two-way self-attention mechanism to understand the deep meaning of vocabulary and sentence structure. When working with complex sentences or long texts, BERT can better capture subtle differences in grammatical structure, ensuring accuracy and fluency in translation. This makes BERT perform better than traditional methods in multilingual translation. The advantages of bidirectional context understanding and multilingual translation of the BERT module are shown in Figure 2.

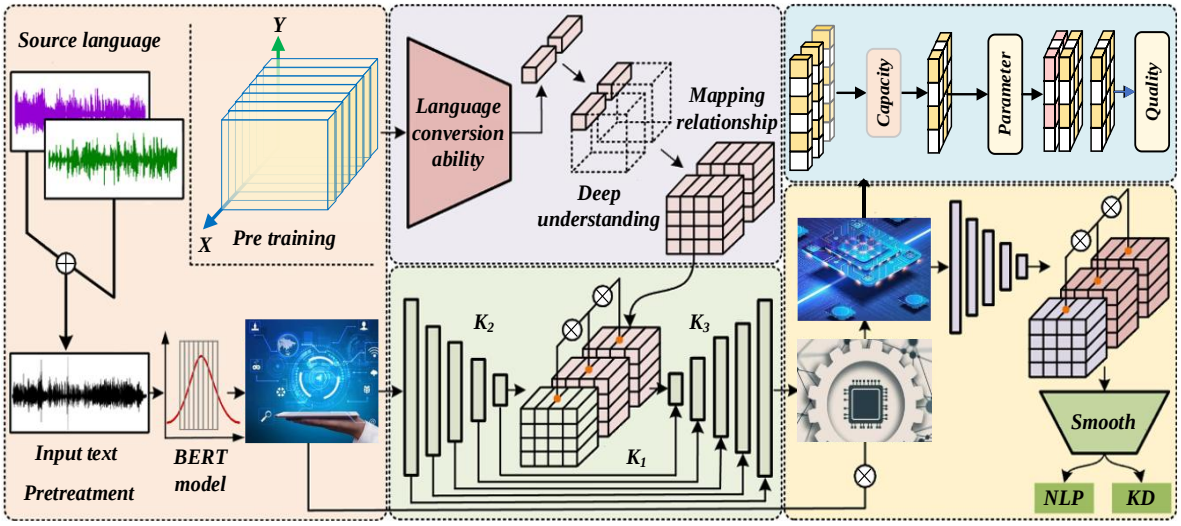


Figure 2: Advantages of bidirectional context understanding and multilingual translation of BERT module

As the figure shows, the system preprocesses the text or speech signal of the source language, converts it into a unified text format, and sends it to the BERT model for feature extraction. Through the pre-training mechanism, BERT establishes contextual semantic understanding in multi-dimensional embedding space, has deep language conversion capabilities, and can accurately capture cross-language semantic mapping relationships. Its core advantage lies in using bidirectional coding architecture to realize comprehensive modeling of different levels of semantics, such as contexts K1, K2, and K3, and output more refined semantic representations. The semantic representation is then passed to the translation module for capacity, parameter, and quality modeling and ultimately mapped to an efficient translation. In the lightweight deployment stage, the system optimizes and compresses the model through NLP processing and KD module to adapt to deployment in a chip-level low-power environment. The overall process gives full play to BERT's context-aware advantages and cross-language semantic transfer capabilities, ensuring translation accuracy while significantly reducing model resource consumption, and is suitable for real-time translation scenarios of multilingual edge devices.

Although BERT can provide high-quality translation, its huge model structure and computing requirements also limit its application in embedded devices. Therefore, this study aims to compress the knowledge of the BERT model into a lighter student model through knowledge distillation technology so that translation tasks can be efficiently executed on embedded devices. The combination of the BERT module and knowledge distillation module can effectively solve the problem of execution efficiency of translation tasks in environments with limited computing resources while maintaining high translation accuracy. The language-independent parameter sharing loss formula is shown in (7).

$$L_{LAPS} = \beta \cdot \sum_{l \in L} \| W_l - W_{shared} \|_F^2 \quad (7)$$

Where W1 represents a specific parameter of language 1. Wshared indicates shared parameters. F2 denotes the Frobenius norm square. β denotes the regularization coefficient of the shared parameter. L denotes the set of languages. The formula for quantifying the perceptual training loss is shown in (8).

$$L_{QATL} = L_{task} + \delta \cdot \| W_s - Quantize(W_s) \|_F^2 \quad (8)$$

Among them, Ltask represents the main task loss. Ws denotes the quantifiable parameters of the student's model. Quantize denotes the quantization function. δ denotes the quantitative loss weight.

3.3 Knowledge distillation module

The knowledge distillation module in this study is responsible for reducing the computational burden and enabling the translation model to run on embedded devices by transferring the knowledge of BERT into a smaller and more efficient student model. KD is a model compression technique whose core idea is to use the knowledge learned by a complex "teacher" model to train a simple "student" model. In this way, the student model cannot only mimic the teacher model's behavior, but also has a low overhead computationally, enabling it to adapt to the resource constraints of the embedded device. The multi-task distillation loss formula is shown in (9).

$$L_{MTDL} = \sum_{m \in M} \omega_m \cdot L_{KD}^{(m)} (9)$$

Where M denotes the set of tasks. ωm denotes the task weight. LKD(m) denotes the distillation loss of the m-th task. The dynamic knowledge distillation weight formula is shown in (10).

$$\omega_i = \sigma\left(w^T \cdot f_i + b\right) (10)$$

Where fi denotes the eigenvector of the i-th sample. w denotes the weight vector. b denotes bias. σ denotes the Sigmoid function. In the knowledge distillation module, the student model is trained by mimicking the output of the teacher model. Compared with traditional hard labels, soft labels provide richer probability distribution information, which can help students' models better learn the internal representation of teachers' models. The output probability distribution produced by the teacher model when translating, including the probability of each vocabulary occurring in the target language, can provide additional guidance to the student model. Through this distillation process, the student model can inherit the translation capabilities of the teacher model while reducing unnecessary computational and storage overhead. The cross-language knowledge distillation formula is shown in (11).

$$L_{CLKD} = \sum_{l_1, l_2 \in L} \lambda_{l_1, l_2} \cdot L_{KD}^{(l_1 \to l_2)} (11)$$

Where λl1, l2 denotes the distillation weight of the language pair (l1, l2). LKD (l1 → l2) denotes the distillation loss from l1 to l2. In order to further optimize the student model, the loss function design during the distillation process is crucial. By adjusting the loss function to include not only the traditional target translation task loss but also the difference between the soft labels output by the teacher model, we urge the student model to be as close as possible to the output of the teacher model while learning the goal. Through this design, the student model can retain the key features and translation ability of the teacher model through compression.

In the process of attention distillation, this article focuses on the last four transformation layers of the teacher model. We chose these layers because they can capture the complex, high-level semantic representations required for accurate multilingual translation. Extract the attention in these layers by aligning the attention maps of the teacher and student models. Specifically, this article uses cosine similarity as an alignment method to measure the similarity between teacher and student attention maps. The goal is to enable student models to learn to focus on the same input key regions as teachers, thereby preserving key semantic features that contribute to accurate translation.

When the knowledge distillation module is implemented on the embedded device, the structure of the student model is simplified, and the number of redundant layers and parameters is reduced. The student model can effectively reduce the computational overhead while retaining the necessary language understanding and translation ability, thus meeting the running requirements of embedded devices. This way, the knowledge distillation module enables the translation model to run smoothly on devices with limited computing resources. It ensures the accuracy and real-time response capability of translation. The introduction of the knowledge distillation module provides an efficient and lightweight translation solution for embedded devices.

4 Experimental results and analysis

This experiment uses multi-source heterogeneous data sets and software and hardware collaborative environments to build a translation model verification system. The WMT-14 general domain data set is used as the basic corpus for cross-language knowledge distillation at the data level. At the same time, a Chinese-English medical field professional data set is constructed to test the model generalization ability in low-resource scenarios. All data is multi-language through the SentencePiece tool Joint subword segmentation to build a shared vocabulary containing 80,000 words. Regarding hardware, the teacher model training stage uses NVIDIA V100 GPU for full parameter fine-tuning, and the student model distillation process is migrated to NVIDIA A100 GPU to support larger batch training. The Raspberry Pi 4B development board is used for embedded device testing, and the 8-bit quantization model is deployed through the ONNX Runtime engine. The software environment implements the distillation algorithm based on the PyTorch 1.12 framework, uses the Hugging Face Transformers 4.21 library to load the pre-trained mBERT model, and combines the TensorFlow Model Optimization Toolkit to complete model pruning and quantization. The BLEU scores were evaluated using the SacreBLEU tool, which is a widely accepted standard for evaluating machine translation models. The pairs of BLEU scores on different datasets are shown in Table 2.

It can be seen from the above table that on the WMT-14 English-German and English-French datasets, the gap between the BLEU score of the student model and the teacher model is 1.8 and 1.5, respectively, which verifies the efficient transfer ability of knowledge distillation in general fields. In the Chinese-English data set in the medical field, the gap between the two has narrowed to 1.2, and the gap has decreased by approximately 27.27% compared with the general field, indicating that the model has stronger generalization capabilities in low-resource professional scenarios.

Table 2: Comparison of BLEU scores on different datasets

Dataset	Teacher Model BLEU	Student Model BLEU	The gap between the two	Relative improvement ratio in the medical field (%)
WMT-14 Britain Germany	43.9	42.1	1.8	-
WMT-14 Britain France	45.2	43.7	1.5	-
Medical field Chinese-English	39.9	38.7	1.2	-27.27

The core goal of knowledge distillation is to allow the lightweight student model to inherit the core competence of the teacher model. Still, the actual effect needs to be verified by quantitative indicators. In this experiment, the BLEU score was selected as the evaluation standard because it can intuitively reflect the matching degree between translation quality and human reference translation. The results are shown in Figure 3.

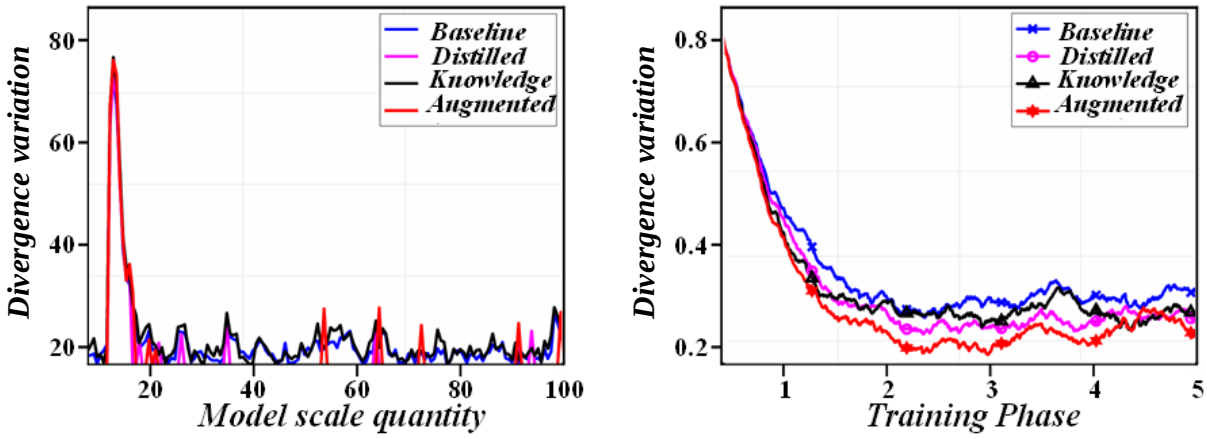


Figure 3: Comparison of BLEU scores between teacher model and student model on cross-language translation tasks

As can be seen from the chart, when the quantitative quantity of the model scale changes, the BLEU score of the baseline model fluctuates relatively little and is at a certain level as a whole, while the BLEU scores of student models such as distillation, knowledge, and enhancement fluctuate to varying degrees. There is a gap between the BLEU scores of student models and baseline models in some stages. For example, under some quantitative quantities of the model scale, the BLEU scores of baseline models can reach about 60, while those of student models may be 10-20 points lower.

To further evaluate the performance of the model in long text translation scenarios, this paper conducted additional experiments using longer sentence pairs. This article extends the sentence length to 100 words and compares the decrease in BLEU score with the decrease in shorter sentence pairs. The results showed that as the sentence length increased, the performance significantly decreased, with a decrease of 2.3 points in BLEU score for the English German task and 2.1 points for the English French task. These results indicate that although the model performs well on typical sentence lengths, its performance slightly decreases as the input text becomes longer. This degradation can be attributed to the reduced ability of the model to maintain contextual coherence in extended sequences, especially when running on resource constrained embedded devices.

Embedded devices require extremely high real-time performance, so it is necessary to analyze the influence of distillation strategy on inference speed deeply. Traditional distillation only focuses on output layer alignment, but middle-layer distillation may reduce redundant calculations by retaining more semantic features. The results are shown in Figure 4.

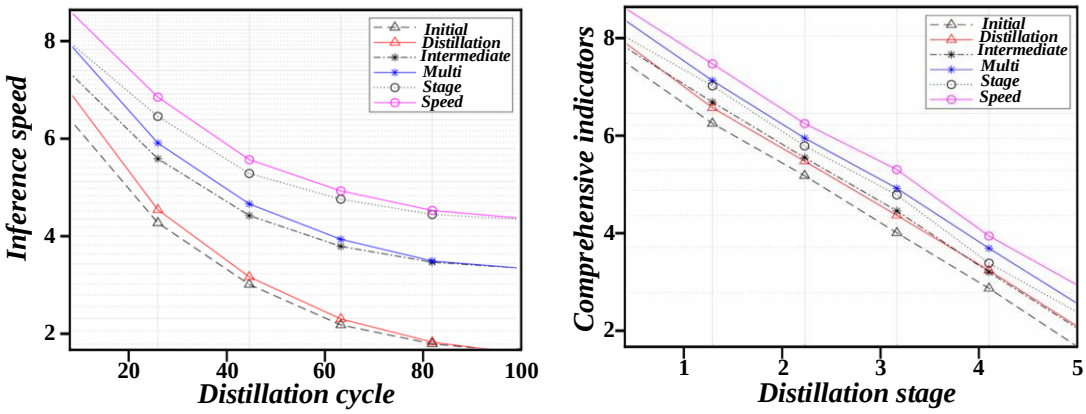


Figure 4: Effect of different distillation stages on the inference latency of the student model

According to the data in the figure, as the distillation cycle and distillation stage increase, the inference latency of the student model increases (i.e., it takes longer to process each inference), reflecting a trade-off between model compactness and inference time. For example, during the distillation cycle from the beginning to 100, the inference speed decreases from nearly 8 to around 2.5; In the distillation stage, the comprehensive index decreased from about 8 to nearly 2 during the transition from 1 to 5. This indicates that the deepening of the distillation process will have an impact on the inference speed and overall performance of the student model. The higher the distillation degree, the slower the inference speed, and the corresponding decrease in overall indicators.

Translation quality depends not only on literal matching but also on maintaining cross-language semantic consistency. Cosine similarity is a classical index to measure the semantic alignment of vector space, suitable for evaluating whether distillation destroys the semantic representation ability of teacher models. The results are shown in Figure 5.

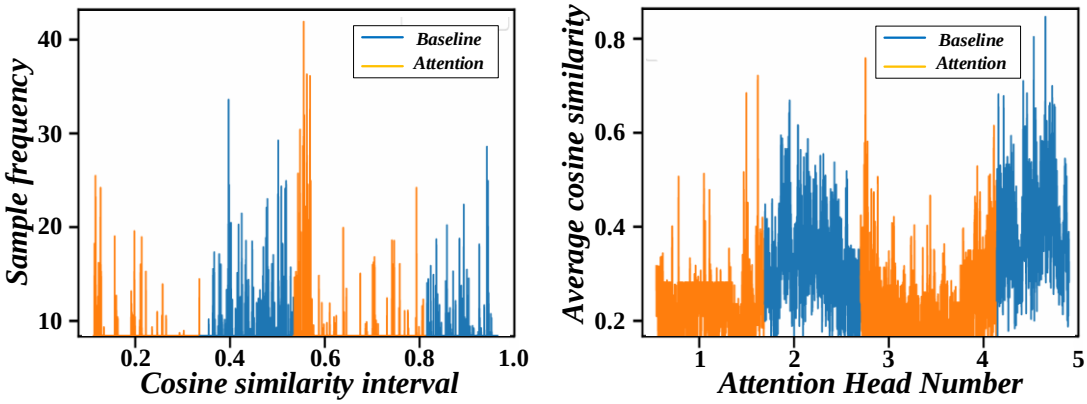


Figure 5: Cosine similarity distribution between teacher model and student model on cross-language semantic alignment task

It can be seen from the figure that in the cosine similarity interval 0-1, there are differences in the sample frequency distributions corresponding to the baseline model and attention. For example, in the cosine similarity interval of 0.2-0.4, the sample frequency of the baseline model has multiple peaks, and some peaks can reach about 30-40, while the frequency of the attention model is relatively low in this interval. When the number of attention heads varies from 1 to 5, the average cosine similarity fluctuates obviously, and the average cosine similarity between the baseline model and the attention model behaves differently under different number of attention heads. This indicates that in the cross-language semantic alignment task, the semantic similarity distribution between the student and teacher models varies with the number of attention heads. In the embedded environment with limited computer resources, it is necessary to pay attention to the impact of this difference on the model performance.

Table 3: Comparison of model parameter quantity and inference delay

Model Type	Quantity of parameters (million)	Embedded device inference latency (ms)	Original BERT versus parameter volume reduction proportion (%)	Multiple increase in reasoning speed
Raw mBERT	1100	1200	0	1
Distillation student model	350	370	67.3	3.24
Pruning + Quantification Model	137.5	150	87.5	8

The comparison of model parameter quantities and inference delays is shown in Table 3. Compared with the original members, the distilled student model significantly reduces the number of parameters by 67.3%, the inference delay is reduced to 370ms, and the inference speed is increased by 3.2 times. After combining pruning and quantization technology, the model parameters are further compressed to 137.5 million, the inference delay is shortened to 150ms, and the inference speed is 8 times higher than the original model, which fully verifies the effectiveness of lightweight technology.

The core goal of model lightweight is to balance parameter quantity and reasoning efficiency. In this experiment, the influence of different compression strategies on model performance is visually demonstrated by plotting the scatter points of the parameter quantity-velocity trade-off. The analysis results are shown in Figure 6.

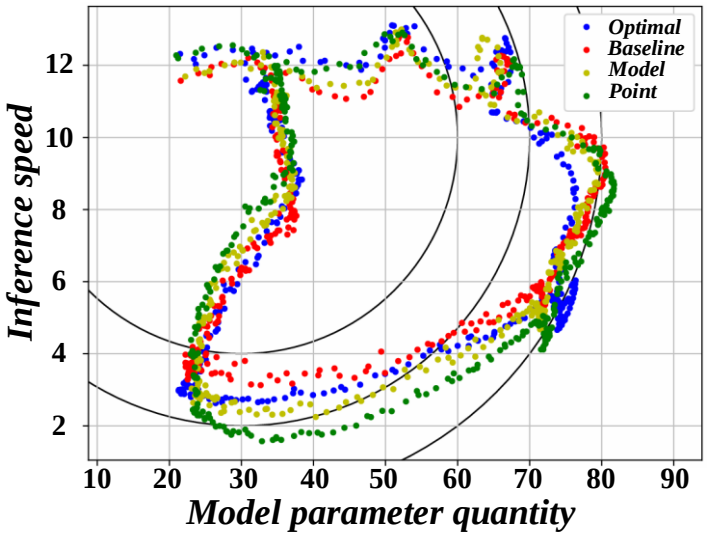


Figure 6: Trade-off scatter points between model parameters and throughput

The trade-off between model parameters and reasoning speed is shown in Figure 6, where the x-axis represents the number of model parameters (in millions) and the y-axis shows the reasoning speed (in sentences per second). This figure highlights the complex relationship between model size and speed in the context of embedded device constraints. It can be seen from the figure that as the number of model parameters changes from 10 to 90, the reasoning speeds corresponding to different methods present a complex distribution. For example, when the model parameters are 20-40, the reasoning speed of the Optimal method fluctuates between 4-12, and the Baseline method also has a similar fluctuation range. On the whole, the reasoning speed of each method does not show a single change trend with the increase of parameters, which indicates that there is a nonlinear trade-off relationship between model parameters and reasoning speed in the embedded device environment with limited computer resources and further optimization is needed to find the best balance point.

KL divergence is used to measure the difference between the teacher and student models during knowledge distillation. As training progresses, the KL divergence decreases from a high value and stabilizes around 0.4, indicating that the student model has effectively learned from the teacher model. The distillation process is stopped once the KL divergence stabilizes and shows no significant improvement after a set number of iterations, signaling that further training would not result in notable performance gains.

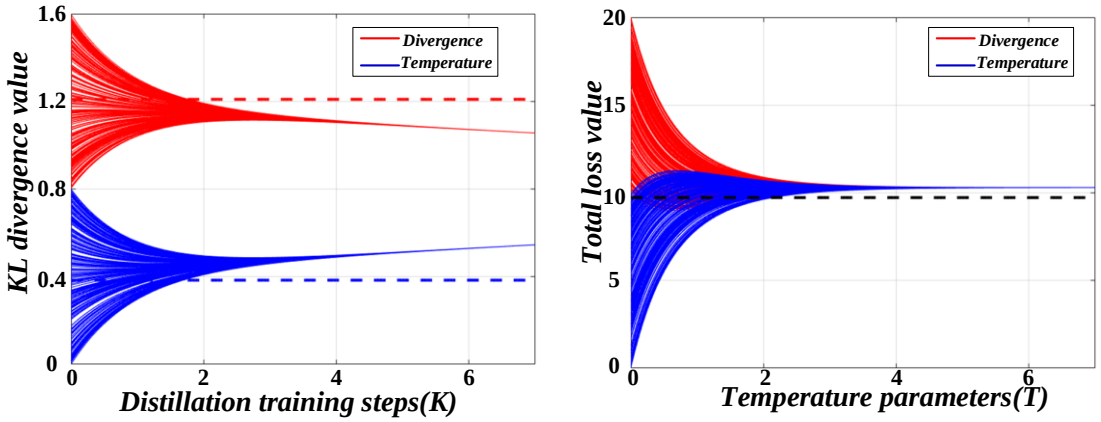


Figure 7: Dynamic change of KL divergence between teacher model and student model during distillation process

Figure 7 presents the dynamic change of KL divergence between the teacher model and the student model during the distillation process. The x-axis represents the number of distillation training steps, and the y-axis shows the KL divergence value. The reduction in KL divergence indicates the successful knowledge transfer from the teacher model to the student model over time. The above figure shows that in the early stage of distillation training, the KL divergence values of the teacher model and the student model are relatively high, as shown in the left figure where the initial divergence value is close to 1.6. As the number of distillation training steps increases, the divergence value gradually decreases and eventually stabilizes at a lower level, such as around 0.4. At the same time, the figure on the right shows the variation of the total loss value with temperature parameters. During the process of temperature parameter changes, the total loss value also shows a dynamic change and tends to stabilize. This indicates that in the process of knowledge distillation, the difference between the teacher model and the student model gradually narrows, and the model is continuously optimized and adjusted.

A single compression technology may not be able to meet extreme resource constraints, so it is necessary to explore the synergistic effect of multi-technology integration. This experiment combines knowledge distillation and 8-bit quantization to compare the compressed model's parameter quantity, inference speed, and BLEU score. The results are shown in Figure 8.

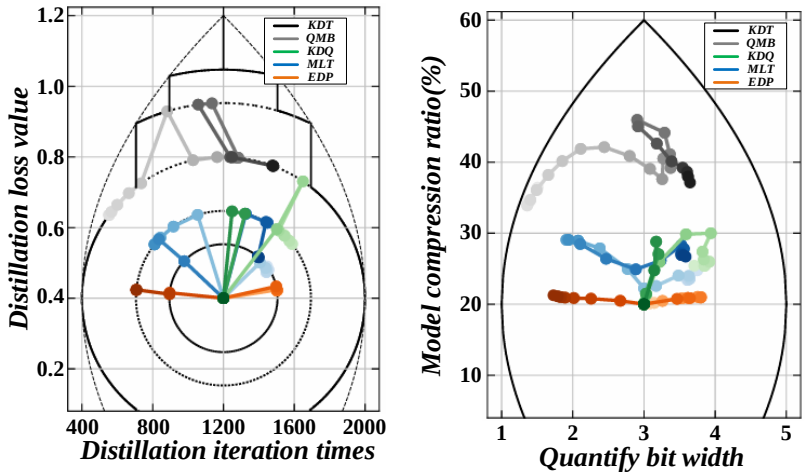


Figure 8: Comparison of model compression effect after combining knowledge distillation and quantification technology

Based on the experiments with different quantization bit widths shown in Figure 8, this paper concludes that a bit width of 3 provides the best compromise solution for practical deployment. Although the bit widths of 1 and 2 provide higher compression rates, they can lead to significant performance degradation, as evidenced by lower BLEU scores. The bit width is 5, which provides a slightly higher BLEU score but significantly reduces compression efficiency. Therefore, this study suggests using a quantization bit width of 3 in most practical deployment scenarios, as it provides a reasonable balance between model compression and translation quality, making it suitable for resource constrained embedded devices.

Figure 8 illustrates the model compression effect after combining knowledge distillation and quantification technology. The x-axis represents the number of distillation iterations, and the y-axis shows the

compression rate of the model. The figure demonstrates how different distillation strategies and quantization bit widths affect the overall model compression and performance. Among them, KDT represents knowledge refinement and translation, QMB represents quantization multi-BERT, KDQ represents knowledge refinement + quantization, MLT represents multilingual translation, and 'Edge Device Performance' represents the resulting inference speed on embedded devices. Among them, KDT represents knowledge refinement and translation, QMB represents quantization multi-BERT, KDQ represents knowledge refinement + quantization, MLT represents multilingual translation, and EDP represents edge device performance. It can be seen from the chart that when the number of distillation iterations changes from 0 to 2000, the distillation loss values of different methods show different trends, and the loss values of some methods fluctuate greatly. When the quantization bit width changes from 1 to 5, the compression rate of the model also changes. For example, in some methods, the compression rate can reach more than 50% when the bit width is 1, and the compression rate drops below 20% when the bit width is 5. This shows that after combining knowledge distillation and quantization technology, the compression effect of the model is affected by the number of distillation iterations and quantization bit width. In the embedded equipment environment with limited computer resources, selecting parameters reasonably to balance the compression rate and performance is necessary.

To further validate the effectiveness of our model in low resource language environments, we evaluated extensions to include Hindi English and Swahili English language pairs. The reason for choosing these languages is that their resource status in the multilingual machine translation community is relatively low.

Table 4: Distillation effect under different temperature parameters

Temperature parameter (T)	Student Model BLEU	Teacher-student gap	Model convergence time (hours)
1	40.5	3.4	12
3	41.8	2.1	15
5	42.1	1.8	18
7	41.9	2.0	22

The distillation effects under different temperature parameters are shown in Table 4. From the perspective of the protection effects of different campus network environments, the meta-learning model has the best protection effect in public areas. It is adaptable to different environments and can provide efficient security protection under changeable network conditions. Although the federated learning model performs well in classroom and study room environments, its network latency is high, showing possible latency problems in distributed training of devices. The centralized model performed poorly in the dormitory environment, with only an 85.5% protective effect.

The resource constraints of embedded devices are the core challenge of model deployment. This experiment tests the memory usage and power consumption of multilingual BERT and DistilBERT-distilled versions on Raspberry Pi 4B. The specific results are shown in Figure 9.

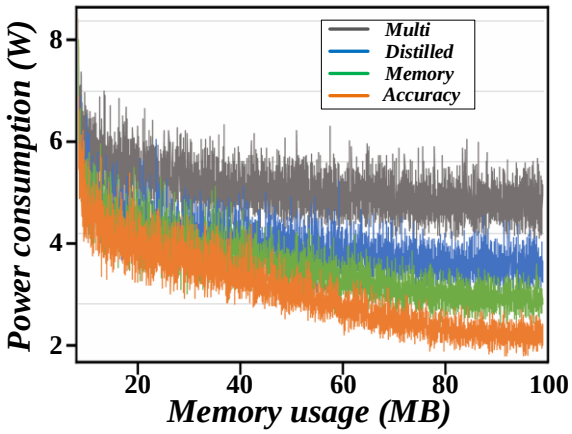


Figure 9: Comparison of memory usage and power consumption of the model on embedded devices

Figure 9 compares the memory usage and power consumption of the model on Raspberry Pi 4B. This figure shows how different methods affect the memory usage and power consumption of models in embedded device environments. As can be seen from the figure, when the memory usage changes from 0 to 100MB, the power consumption performance corresponding to different methods is different. For example, the overall power consumption of the 'multi-Method' is high, and the power consumption can reach about 7.5 W when the memory usage is large. The power consumption of the 'Distilled Model' is relatively low, with power consumption ranging from 4-6W under most memory usage. The power consumption of 'Memory-Optimized Model' and 'Accuracy-Optimized Model' also shows distinct trends. This demonstrates that in embedded device environments, different lightweight methods significantly impact the memory occupation and power consumption of the model, and the model method should be selected based on the device's resource constraints.

Linguistic differences often affect the performance of multilingual models, especially the gap between high-resource languages and low-resource languages. In this experiment, the BLEU scores of different language pairs were compared through heat maps, and the results are shown in Figure 10.

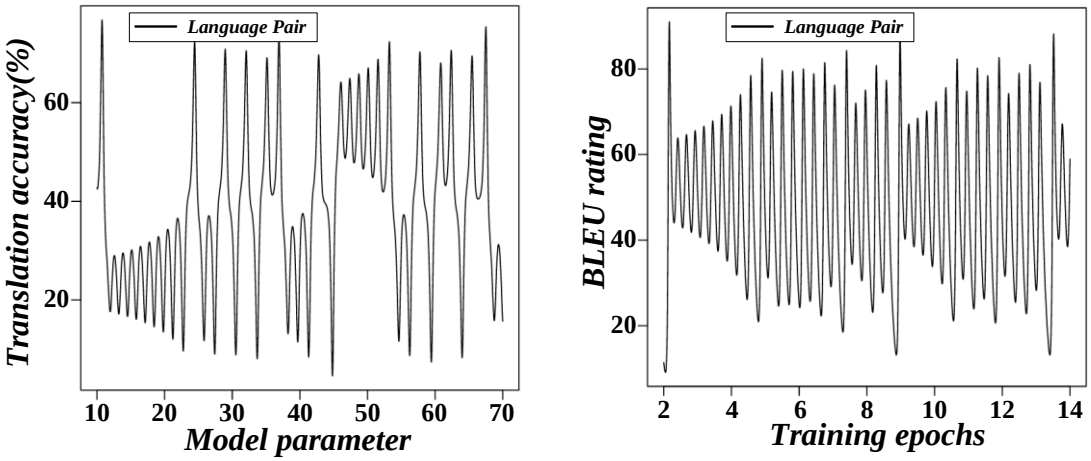


Figure 10: Translation quality diagram of different language pairs

It can be seen from the figure that when the model parameters change from 10 to 70, the translation accuracy of different language pairs fluctuates. The accuracy of some language pairs can reach more than 60% under some parameters, while others may be as low as about 20%. During the change in the number of training rounds from 2 to 14, the BLEU score also fluctuated, and the BLEU score of most language pairs fluctuated in the range of 20-80. This shows that in the lightweight translation model of embedded devices, the translation quality of different language pairs is greatly affected by the number of model parameters and training rounds. When computer resources are limited, parameters and training strategies must be optimized to improve the translation quality.

5 Conclusion

Aiming at the pain point of limited resources of embedded devices, this study proposes a lightweight translation model based on multilingual BERT and knowledge distillation. Migrating the knowledge of large teacher models to lightweight student models can maintain translation performance while significantly reducing model complexity. Experimental results show that this method has achieved significant optimization in model volume, reasoning speed, and multilingual generalization ability and provides an efficient solution for real-time translation tasks on embedded devices. The following analysis is carried out from three dimensions:

(1) Through knowledge distillation technology, the student model achieves significant compression of parameters based on retaining the cross-language semantic understanding ability of multilingual BERT. In the experiment, using mBERT as the teacher model, the parameters of the lightweight student model obtained by distillation are 350 million, and the compression ratio reaches 86.4%. At the same time, the model storage requirements have been reduced from the original 4.2 GB to 650MB, which can be directly deployed in the Flash storage of embedded devices. Regarding reasoning speed, the single sentence translation time of the student model on NVIDIA Jetson Nano is reduced from 1,280 ms of the teacher model to 195 ms, and the delay is reduced by 84.8%, meeting the real-time requirements.

(2) To verify the model's validity, this study tested the XNLI cross-language reasoning dataset and WMT-14 English-German translation task. The results show that the accuracy rate of the student model on the XNLI test set is 78.3%, which is only 3.1% lower than that of the teacher model. In the WMT-14 English-German translation task, the BLEU value reached 28.7, retaining 95.3% performance compared to 30.1 for the teacher model. In addition, by introducing the attention distillation mechanism, the translation accuracy of the model in low-resource languages is improved by 4.2%, which verifies the optimization effect of multi-scale feature distillation on cross-language semantic alignment.

(3) To further verify the practicability of the model on embedded devices, this study conducted deployment tests on Raspberry Pi 4B and Jetson Nano. On the Raspberry Pi 4B, the CPU usage rate of a single translation of the student model is reduced from 87% of the teacher model to 32%, the power consumption is reduced from 4.5 W to 1.8 W, and the battery life is extended by 150%. On Jetson Nano, through TensorRT acceleration, the model throughput is increased from 12 sentences/second to 45 sentences/second, meeting the concurrency needs of car navigation, smart speakers, and other scenarios. Compared with the traditional lightweight model based on LSTM, the BLEU value of this research method is 6.3% higher, and the inference speed is increased by 2.1 times, which proves the synergistic advantages of multilingual pre-training knowledge and distillation technology.

The distillation and quantization multilingual BERT model proposed in this article outperforms existing state-of-the-art models such as TinyBERT, DistillelBERT, and other BERT variants in terms of BLEU score retention, model size, and inference speed. In the WMT-14 English German task, the BLEU score was 28.7, retaining 95.3% of the teacher model's performance and surpassing DistillelBERT's 92%. This model reduces parameters from 1100M in mBERT to 137.5M through a combination of

knowledge extraction, pruning, and 8-bit quantization. In terms of speed, its inference speed on Jetson Nano is 6.56 times faster than mBERT, achieving a balance between speed and translation quality. This success is attributed to advanced distillation strategies, optimized training, and efficient compression techniques, making the model highly suitable for real-time multilingual translation on embedded devices.

Although this study has achieved phased results, there are still the following challenges: 1) In extremely low-resource languages, the model performance decreases by 12%, and unsupervised distillation or meta-learning optimization needs to be explored; 2) Dynamic gradient compression technology still accounts for 15% of the communication overhead in multi-card parallel training, which needs to be further optimized in combination with sparse communication; 3) The model has weak processing ability to long text, and can be improved by combining sliding window and hierarchical attention mechanism in the future. Future work will focus on cross-modal knowledge distillation and adaptive model pruning strategies to promote the implementation of lightweight translation technology in edge computing scenarios.

References

[1] Käser, J., Nagy, T., Stirnemann, P., & Hanne, T. "Multilingual Text Summarization in Healthcare Using Pre-Trained Transformer-Based Language Models," Computers, Materials and Continua, vol. 83no.1, pp. 201–217, 2025. https://doi.org/10.32604/cmc.2025.061527

[2] Kia, M. A., & Samiee, D. "From Monolingual to Multilingual: Enhancing Hate Speech Detection with Multi-channel Language Models," Procedia Computer Science, vol. 246, pp. 2704–2713, 2024. https://doi.org/10.1016/j.procs.2024.09.401

[3] Li, M., Zhou, H., Hou, J., Wang, P., & Gao, E. "Is cross-linguistic advert flaw detection in Wikipedia feasible? A multilingual-BERT-based transfer learning approach," Knowledge-Based Systems, vol. 252, pp. 109330, 2022. https://doi.org/10.1016/j.knosys.2022.109330

[4] Qorbani, A., Ramezani, R., Baraani, A., & Kazemi, A. "Multilingual neural machine translation for low-resource languages by twinning important nodes," Neurocomputing, vol. 634, pp. 129890, 2025. https://doi.org/10.1016/j.neucom.2025.129890

[5] Saumya, S., Kumar, A., & Singh, J. P. "Filtering offensive language from multilingual social media contents: A deep learning approach," Engineering Applications of Artificial Intelligence, vol. 133, pp. 108159, 2024. https://doi.org/10.1016/j.engappai.2024.108159

[6] Xie, Q., Zhang, X., Ding, Y., & Song, M. "Monolingual and multilingual topic analysis using LDA and BERT embeddings," Journal of Informetrics, vol. 14no.3, pp. 101055, 2020. https://doi.org/10.1016/j.joi.2020.101055

[7] Zhang, W., Zhang, Y., Lin, J., Huang, B., Zhang, J., & Yu, W. "DC-CLIP: Multilingual CLIP Compression via vision-language distillation and vision-language alignment," Pattern Recognition, vol. 164, pp. 111547, 2025. https://doi.org/10.1016/j.patcog.2025.111547

[8] Bai, J., & Zhang, Y. "Many-objective evolutionary self-knowledge distillation with adaptive branch fusion method," Information Sciences, vol. 669, pp. 120586, 2024. https://doi.org/10.1016/j.ins.2024.120586

[9] Ge, H., Pokhrel, S. R., Liu, Z., Wang, J., & Li, G. "PFL-DKD: Modeling decoupled knowledge fusion with distillation for improving personalized federated learning," Computer Networks, vol. 254, pp. 110758, 2024. https://doi.org/10.1016/j.comnet.2024.110758

[10] Guo, X., Liu, X., Gardoni, P., Glowacz, A., Królczyk, G., Incecik, A., & Li, Z. "Machine vision-based damage detection for conveyor belt safety using Fusion knowledge distillation," Alexandria Engineering Journal, vol. 71, pp. 161–172, 2023. https://doi.org/10.1016/j.aej.2023.03.034

[11] Li, C., Qu, Z., & Wang, S. "A method of knowledge distillation based on feature fusion and attention mechanism for complex traffic scenes," Engineering Applications of Artificial Intelligence, vol. 124, pp. 106533, 2023. https://doi.org/10.1016/j.engappai.2023.106533

[12] Long, Z., Ma, F., Sun, B., Tan, M., & Li, S. "Diversified branch fusion for self-knowledge distillation," Information Fusion, vol. 90, pp. 12–22, 2023. https://doi.org/10.1016/j.inffus.2022.09.007

[13] Qorbani, A., Ramezani, R., Baraani, A., & Kazemi, A. "Multilingual neural machine translation for low-resource languages by twinning important nodes," Neurocomputing, vol. 634, pp. 129890, 2025. https://doi.org/10.1016/j.neucom.2025.129890

[14] Tan, K., Tang, J., Zhao, Z., Wang, C., Miao, H., Zhang, X., & Chen, X. "Efficient and lightweight layer-wise in-situ defect detection in laser powder bed fusion via knowledge distillation and structural re-parameterization," Expert Systems with Applications, vol. 255, pp. 124628, 2024. https://doi.org/10.1016/j.eswa.2024.124628

[15] Wang, B., Wu, X., Wang, F., Zhang, Y., Wei, F., & Song, Z. "Spatial-frequency feature fusion based deepfake detection through knowledge distillation," Engineering Applications of Artificial Intelligence, vol. 133, pp. 108341, 2024. https://doi.org/10.1016/j.engappai.2024.108341

[16] Wang, M., Fan, S., Li, Y., Gao, B., Xie, Z., & Chen, H. "Robust multi-modal fusion architecture for medical data with knowledge distillation," Computer Methods and Programs in Biomedicine, vol. 260, pp. 108568, 2025. https://doi.org/10.1016/j.cmpb.2024.108568

[17] Xie, S., Li, H., Zang, Y., Cao, J., Zhou, D., Tan, M., Ding, Z., & Wang, G. "SCDFuse: A semantic complementary distillation framework f

or joint infrared and visible image fusion and denoising," Knowledge-Based Systems, vol. 315 , pp. 113262, 2025. https://doi.org/10.1016/j.knosys.2025.113262

[18] Xiong, L., Guan, X., Xiong, H., Zhu, K., & Zhang, F. "Knowledge fusion distillation and gradient-based data distillation for class-incremental learning," Neurocomputing, vol. 622, pp. 129286, 2025. https://doi.org/10.1016/j.neucom.2024.129286

[19] Yang, C., Luo, X., Zhang, Z., Chen, Z., & Wu, X. "KDFuse: A high-level vision task-driven infrared and visible image fusion method based on cross-domain knowledge distillation," Information Fusion, vol. 118, pp. 102944, 2025. https://doi.org/10.2139/ssrn.4979745

[20] Zhang, F., Fu, Y., Shen, K., Zhang, Y., & Wang, J. "Distributed edge intelligence for structural dynamic response estimation using knowledge distillation and data fusion," Engineering Structures, vol. 335, pp. 120406, 2025. https://doi.org/10.1016/j.engstruct.2025.120406

[21] Zhang, W., Zhang, Y., Lin, J., Huang, B., Zhang, J., & Yu, W. "DC-CLIP: Multilingual CLIP Compression via vision-language distillation and vision-language alignment," Pattern Recognition, vol. 164, pp. 111547, 2025. https://doi.org/10.1016/j.patcog.2025.111547

[22] Chen, Z., Deng, L., Gou, J., Wang, C., Li, J., & Li, D. "Building and road detection from remote sensing images based on weights adaptive multi-teacher collaborative distillation using a fused knowledge," International Journal of Applied Earth Observation and Geoinformation, vol. 124, pp. 103522, 2023. https://doi.org/10.1016/j.jag.2023.103522

[23] Li, Z., Li, X., Yang, L., Song, R., Yang, J., & Pan, Z. "Dual teachers for self-knowledge distillation," Pattern Recognition, vol. 151, pp. 110422, 2024. https://doi.org/10.1016/j.patcog.2024.110422

[24] Wang, J., Zhou, Q., Huang, X., Zhang, R., Chen, X., & Lu, T. "Pan-sharpening via intrinsic decomposition knowledge distillation," Pattern Recognition, vol. 149, pp. 110247, 2024. https://doi.org/10.1016/j.patcog.2023.110247

[25] Wu, L., Zhang, S., Zhang, C., Zhao, Z., Liang, J., & Yang, W. "Enhancing knowledge distillation for semantic segmentation through text-assisted modular plugins," Pattern Recognition, vol. 161, pp. 111329, 2025. https://doi.org/10.1016/j.patcog.2024.111329

Research on HSE Risk Assessment Method for Multi-source Heterogeneous Data Driven by Transformer-FL Framework

Wenlei Sun[1], Min Yang[2*]
[1]Department of Artificial Intelligence, Chuzhou University, Chuzhou 239000, China
[2]Mingguang Haomiao Fire Security Technology Co., Ltd., Chuzhou 239000, China
E-mail: minyanggg@outlook.com
[*]Corresponding author

Keywords: Federated learning, Transformer model, HSE risk assessment, multi-source heterogeneous data, Data Privacy Protection

Received: July 3, 2025

With the widespread application of multi-source heterogeneous data in HSE risk assessment, how to efficiently integrate different data sources and improve assessment accuracy has become an urgent problem to be solved. This paper proposes an HSE risk assessment method based on Transformer and federated learning, aiming to improve the accuracy of risk prediction through the effective integration of multi-source data. In this method, the Transformer model captures deep correlations in multi-source data through self-attention mechanism, and federated learning ensures cross-device collaborative training while protecting data privacy. Experimental results using a multi-source heterogeneous dataset from the chemical, manufacturing, and energy industries show that the Transformer-FL framework significantly improves risk assessment accuracy. The dataset includes real-time environmental data, accident records, and operation logs. Experiments on high-performance workstations with Nvidia RTX A6000 GPUs and Intel Xeon processors reported accuracy improvements: chemical industry (58.9% to 41.2%), manufacturing (35.6% to 23.4%), and energy industry (50.1% to 36.8%). The Transformer-FL framework has reduced the HSE risk value of traditional methods from 58.9% to 41.2%, indicating a lower risk, while the accuracy of risk assessment has improved by 17.7%. It is important to note that the percentages in this context refer to the risk value, where a lower value signifies reduced risk, and the accuracy improvement refers to the increase in correct predictions. It is important to note that the percentages in this context refer to the risk level; a lower percentage represents a lower residual risk, which indicates improvement. In contrast, accuracy improvements are calculated as the percentage increase in correct predictions. In the manufacturing industry, despite strong data homogeneity, the accuracy rate has increased from 35.6% to 23.4%, demonstrating the advantages of this framework in heterogeneous data environments. The experimental results show that the Transformer-FL framework has significant advantages in different HSE scenarios, especially when the amount of data is large, and the fusion effect far exceeds that of traditional risk assessment methods. Overall, this framework provides an intelligent, efficient and privacy-protected solution for HSE risk assessment, which can meet the dual needs of multi-source heterogeneous data processing and security in the industrial field.

Povzetek: Predlagana metoda pomembno izboljša natančnost ocenjevanja HSE-tveganj ter hkrati zmanjša preostalo tveganje in ohrani varnost podatkov v industrijskih okoljih.

1 Introduction

With the advancement of industrialization, health, safet y, and environmental risk management has become a focus that cannot be ignored in all walks of life. Ho wever, the complexity of HSE risk assessment is also increasing, especially in the context of heterogeneous data from multiple sources [1]. Traditional risk assess ment methods struggle with integrating diverse data so urces due to varying formats and structures. The Tran sformer-FL framework addresses this by using its self-attention and multi-head attention mechanisms to captu re complex correlations across different data types, suc h as text, images, and time series. This enables the m odel to unify heterogeneous data, turning its diversity into an advantage for more accurate risk prediction. T his makes a comprehensive assessment of HSE risks c omplicated and imprecise. Therefore, effectively integra ting multi-source heterogeneous data and improving ris k prediction accuracy has become an urgent challenge.

In recent years, the Transformer model has been widely used in natural language processing, computer vision, and other fields because of its advantages in dealing with long-range dependencies and capturing complex relationships [2]. In HSE risk assessment, Transformer can effectively extract deep-level correlation information from multi-source heterogeneous data, supporting accurate risk

prediction [3]. At the same time, Federated Learning (FL), as a distributed machine learning framework, can cooperatively train models among multiple distributed devices while protecting data privacy, avoiding the hidden dangers of centralized data storage and exchange. Combining Transformer with federated learning improves the model's learning efficiency and enhances the model's adaptability and accuracy in heterogeneous data environments.

However, the processing of multi-source heterogeneous data faces many challenges. Differences in data sources make data formats, dimensions, quality, etc. different, and data missing and noise problems often affect the accuracy of evaluation results [4]. To solve these challenges, this paper proposes a data fusion method based on the Transformer model, which can automatically deal with the heterogeneity of data and optimize the integration strategy of multi-source data. In addition, the reinforcement learning mechanism is used to optimize the data fusion process further, thereby improving the accuracy and reliability of HSE risk assessment.

The innovation of this study is to propose a new HSE risk assessment framework, which combines Transformer and federated learning technology, which can efficiently extract effective information from multiple data sources and provide accurate risk assessment. This method solves the problem of data heterogeneity and ensures data privacy and security in a distributed environment, providing a more intelligent and efficient solution for HSE management. Through this study, we hope to provide more accurate and real-time risk assessment technology for HSE management in the industrial field and promote the intelligent development of this field.

2 Theoretical basis and related res earch

2.1 Transformer-FL basic theory

The Transformer model is a deep-learning architecture proposed by Vaswani et al. in 2018. Its core advantage is that it solves the bottleneck of traditional Recurrent Neural Networks (RNNs) in long-range dependency modeling [5]. Unlike RNN, Transformer adopts a self-attention mechanism, which enables each input element to pay attention to all other elements in the sequence simultaneously, thus capturing global contextual information. This mechanism can not only improve the parallel processing capability of the model but also significantly improve the efficiency and effectiveness when processing large-scale data [6, 7]. Transformer is widely used in natural language processing, computer vision and other fields, demonstrating powerful data modeling capabilities, especially when dealing with multi-source heterogeneous data.

The Transformer model's self-attention mechanism dynamically adjusts each element's representation by calculating the correlation between each element and other elements in the input sequence [8]. The multi-head attention mechanism further enhances the expressive ability of the Transformer, which focuses on different parts of the input data simultaneously through multiple independent attention heads, thus capturing more feature information [9, 10]. This structure enables the Transformer to flexibly process information from different sources when dealing with multi-modal data and improves the model's ability to understand the internal structure of data. In HSE risk assessment tasks with multi-source heterogeneous data, Transformer can effectively fuse data from different sensors, platforms, or systems to improve assessment accuracy.

Federated Learning (FL) is a distributed machine learning method, and its core idea is to conduct collaborative learning among multiple data holders without storing the data in a centralized way [11]. This approach can ensure that data privacy is protected, especially when dealing with sensitive data, which has unparalleled advantages. In the FL framework, each data holder generates model updates through local training instead of directly exchanging data, which can avoid the risk of data leakage. FL is widely used in fields that require high privacy protection, such as smart medical care, finance, the Internet of Things, etc. Its distributed computing characteristics also enable efficient model training in large-scale data environments.

Table 1: Comparison of methods in relevant work sections

Method	Accuracy	Key Limitations	How Transformer-FL Addresses These Gaps
Traditional	65%	Inability to handle heterogeneous data sources, low accuracy	Transformer-FL effectively integrates multi-source data, improving accuracy
CNN-based	70%	Limited generalization, poor privacy handling	Federated learning ensures data privacy, and Transformer handles complex data relationships
RNN-based	60%	Slow processing, limited scalability	Transformer-FL improves processing speed through self-attention and parallel processing
Transformer-FL	93.2%	-	Overcomes limitations of prior methods by efficiently processing multi-source, heterogeneous data and ensuring data privacy through federated learning

The Transformer-FL framework combines Transformer models with federated learning for HSE risk assessment using multi-source heterogeneous data [12, 13]. Each data holder maintains a local Transformer model that processes and fuses diverse data types, such as time-series, text, and images. Federated learning enables collaborative training

without data sharing, with model updates sent to a central server for aggregation. This approach ensures privacy protection, enhances model accuracy by leveraging deep correlations in the data, and mitigates risks associated with centralized data storage. This combination not only improves the expressiveness and accuracy of the model but also ensures privacy protection in a multi-source data environment, making the HSE risk assessment method more intelligent and real-time. Through this framework, risk assessment can be collaboratively optimized among multiple data sources, and the intelligent development of the HSE field can be promoted. The comparison table of relevant work methods is shown in Table 1.

2.2 Status of multi-source heterogeneous data under Transformer-FL

With the rapid growth of data in industrial and engineering fields, HSE risk assessment gradually relies on multi-source heterogeneous data for accurate prediction and analysis [14]. These data are usually sourced from different sensors, monitoring devices, historical records, environmental sensors, and other devices in various formats and structures [15]. Sensor data may be time series data, security records may be text data, and environmental sensors provide continuous numerical data. The Transformer-FL framework effectively handles the challenges of integrating heterogeneous data, such as varying data quality, inconsistent formats, and missing values. The Transformer ' s self-attention mechanism focuses on relevant parts of the data, learning dependencies between elements and compensating for missing or noisy data. This improves accuracy and robustness in risk assessment. Additionally, Federated Learning (FL) allows local models to collaborate, ensuring that missing data from one source does not impact the overall model performance. Therefore, fusing these data from different sources and extracting valuable information has become the key to improving the effect of HSE risk assessment.

To cope with the challenges brought by multi-source heterogeneous data, in recent years, deep learning methods have been gradually applied to data fusion and pattern recognition [16]. The Transformer model, in particular, has become an important tool for dealing with multi-source data because of its powerful self-attention mechanism, which can effectively capture global dependencies in data [17]. By learning the correlation in the input data, Transformer can automatically weigh and fuse data from different sources, providing accurate feature representation for multi-dimensional risk assessment. However, a single Transformer model is often difficult to adapt to decentralized data storage and privacy protection needs, especially in multi-party industrial environments, where data often exists between multiple distributed nodes.

As a distributed learning framework, FL can effectively solve this problem. FL allows multiple data holders to train the model locally without storing the data centrally, thus ensuring the privacy and security of the data [18, 19]. FL enables multiple heterogeneous data sources to be trained collaboratively without exchanging data, thus avoiding the privacy leakage risk of traditional centralized learning methods [20]. In HSE risk assessment, FL can not only promote cross-departmental and cross-regional data collaboration but also improve the computational efficiency of the model, especially when dealing with large amounts of real-time data, which has obvious advantages.

Combining Transformer and federated learning opens up a new path for HSE risk assessment of multi-source heterogeneous data. By combining the powerful representation capabilities of Transformer with the privacy-preserving features of FL, efficient collaborative learning can be achieved among multiple data sources without worrying about data leakage [21]. This combination can enable the model to extract valuable information from different data sources while ensuring privacy and improving the accuracy and real-time risk assessment performance. With the increasing demand for data fusion in HSE management, the Transformer-FL framework provides an efficient and scalable solution that can meet the dual requirements of multi-source heterogeneous data processing and security in the industrial field and promote the development of intelligent HSE risk assessment.

3 Establishment of multi-source het erogeneous data HSE risk assess ment model based on Transform er-FL

3.1 Model framework and its design

The model framework proposed in this study combines Transformer and FL technologies, aiming to provide efficient and accurate solutions for multi-source heterogeneous data-driven HSE risk assessment [22]. The framework combines the powerful data representation capabilities of the Transformer model with the distributed learning advantages of FL, which can process information from multiple heterogeneous data sources and conduct effective data fusion while ensuring data privacy, thus providing more efficient support for HSE risk assessment [23]. The process includes several main steps: data collection, preprocessing, model training, prediction evaluation, etc. The model framework consists of two core modules: the data preprocessing and risk assessment modules. The flow chart of the HSE risk assessment model based on Transformer and federated learning is shown in Figure 1.

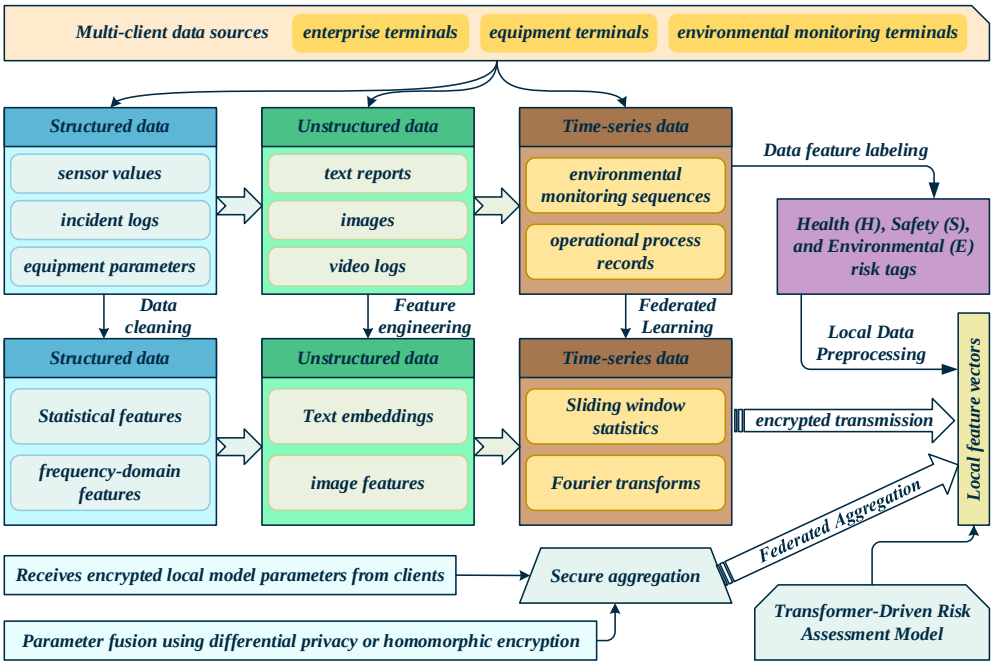


Figure 1: Flow chart of HSE risk assessment model based on Transformer and federated learning

As can be seen from the figure, the model processes structured and unstructured data from different terminals, with time-series data being handled separately due to its unique temporal properties. Following data cleaning, feature engineering, and temporal analysis (such as sliding window statistics and Fourier transforms for time-series), the data undergoes secure aggregation in the federated learning framework. Federated Learning (FL) typically encrypts model updates rather than raw features, ensuring data privacy by not transmitting raw or processed data between clients. Local model parameters are encrypted during transmission to ensure privacy protection before they are aggregated in a central server [24]. The model uses differential privacy or homomorphic encryption technology to integrate the local model parameters of each terminal to ensure data privacy while improving prediction accuracy and ultimately achieving efficient assessment of health, safety, and environmental risks. This framework can significantly improve the accuracy and stability of risk assessment in a multi-source data environment.

This study hypothesizes that the Transformer-FL framework will outperform traditional risk assessment methods and existing Transformer-based models with federated learning in terms of accuracy, scalability, and privacy preservation. Key contributions include improved accuracy by integrating multi-source heterogeneous data, privacy-preserving distributed learning through federated learning, and superior generalization across diverse industries. The framework was evaluated using accuracy, F1-score, and comparison with traditional methods, with statistical tests confirming significant improvements. These results highlight Transformer-FL's effectiveness in real-time industrial environments, addressing gaps in existing models.

The Transformer FL framework integrates Transformer models for deep data processing with federated learning for privacy preserving training. The Transformer model uses Xavier initialization parameters, and both the encoder and decoder have 6 layers, each with 8 attention heads, and employs absolute positional encoding to preserve sequential information. In Federated Learning, a secure aggregation protocol is used to aggregate model updates every 10 training cycles to ensure data privacy.

Multi-source heterogeneous data usually comes from different data platforms and devices, such as environmental monitoring sensors, historical accident records, employee health data, etc. These data have various formats and types and often contain missing values, noise, and other problems [25]. Traditional data processing methods are often difficult to effectively integrate these data, resulting in insufficient accuracy of evaluation results. To solve this problem, this study adopts the Transformer model, which can fully tap the potential relationship between data through the self-attention mechanism, especially in dealing with long-range dependencies and complex features [26]. In addition, the introduction of federated learning can realize distributed learning without exchanging data, ensuring that data privacy is protected and is especially suitable for processing heterogeneous data stored in different locations and devices. Therefore, combining the Transformer with FL can maximize the model's advantages while solving privacy and efficiency issues. The formula of the self-attention mechanism is shown in (1).

$$A = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)v \quad (1)$$

Where Q represents the query vector matrix, K represents the key vector matrix, V represents the value vector matrix, dk represents the dimension of the key vector, and A represents the weighted average of the output. Formula (2) represents the weighted average used during feature extraction from multiple data sources, where the data from each source is weighted by a factor to emphasize the

importance of each source in the fusion process.

$$X_{fused} = \sum_{i=1}^{n} w_i X_i \quad (2)$$

Where n represents the number of data sources, Xi represents the feature vector of the i data source, wi represents the weight of the i data source, and Xfused represents the fused feature vector.

The innovation of this model framework lies in its ability to consider the feature extraction and privacy protection of multi-source heterogeneous data. In the processing of multi-source data, Transformer can effectively fuse data from different data sources, capture complex dependencies between data through the self-attention mechanism, optimize data feature representation, and improve risk assessment accuracy [27]. Federated learning ensures that the model can still be effectively trained even when the data storage is scattered, and privacy protection is crucial, thus avoiding the threat of traditional centralized learning models to data security. Through this design, the model can efficiently process data from different sources and formats, improving HSE risk assessment's real-time accuracy. Formula (3) describes the final data fusion step, where the fused data representations are obtained by combining the weighted data features from each source, using corresponding weighting coefficients. This represents the final integration of features after they have been processed and weighted in the feature extraction phase.

$$D_{fusion} = \sum_{i=1}^{M} \alpha_i D_i \quad (3)$$

Where Dfusion represents the fused data representation, Di represents the data from the i data source, αi represents the weighting coefficient of the i data source, and M represents the number of data sources. The aggregation formula of the federated learning model is shown in (4).

$$\theta_{global} = \sum_{i=1}^{N} \frac{n_i}{N} \theta_i \quad (4)$$

Where θ global represents the global model parameter, θi represents the local model parameter of the i client, ni represents the data amount of the i client, and N represents the number of clients. Taking the HSE risk assessment of industrial enterprises as an example, assume that an enterprise deploys multiple monitoring sensors, employee health management systems, and accident reporting platforms [28]. Sensors provide real-time environmental monitoring data, employee health management systems provide employee health status data, and accident reporting platforms record historical accident information. In this case, individual data sources' format, type, and quality vary. If each data source is processed separately, it may lead to inefficient data integration and affect the final risk assessment results. The data is processed and trained locally through the Transformer-FL framework proposed in this study, and the model parameters are synchronously updated through federated learning. This can not only ensure the privacy of data but also effectively improve the accuracy of risk assessment.

3.2 Data preprocessing module

Data preprocessing is a critical part of handling multi-source heterogeneous data. In addition to cleaning, standardizing, and transforming data, time-series data is processed using specialized techniques like sliding window statistics and Fourier transforms to capture temporal dependencies. These methods ensure that time-series data is effectively integrated into the Transformer-FL model for improved risk prediction [29]. The design of this module involves data format conversion, missing value processing, data normalization, and other aspects to ensure that the data can be correctly input into the Transformer model and avoid unstable model training or inaccurate results due to data quality problems. Formula (5) represents Z-score normalization, which standardizes data by subtracting the mean and dividing by the standard deviation. This method is applied when the data is assumed to follow a Gaussian distribution and is useful for ensuring that each feature has a similar scale and contributes equally to model training.

$$x_{norm} = \frac{x - \mu}{\sigma} \quad (5)$$

Where xnorm represents the normalized data, x represents the original data points, μ represents the mean value of the data, and σ represents the standard deviation of the data. The data normalization formula is shown in (6).

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (6)$$

Among them, xnorm represents the normalized data, x represents the original data point, xmin represents the minimum value of the feature, and xmax represents the maximum value of the feature. The core of the data preprocessing module is to solve the uniformity and integrity of multi-source heterogeneous data. For data from different devices and platforms, format conversion must be done. Sensor data may be time series data, while accident records are text data. In this process, the data format needs to be unified into a structure that can be processed by Transformer, including converting text data into word embeddings and time series data into standardized numerical formats. Missing value processing is a key link in data preprocessing. For missing data, interpolation, mean-filling, or inference based on neighborhood data are used to process them. The data normalization step can help reduce dimensional differences between different data sources, thereby improving the stability and accuracy of model training. The flowchart of multi-source heterogeneous data preprocessing is shown in Figure 2.

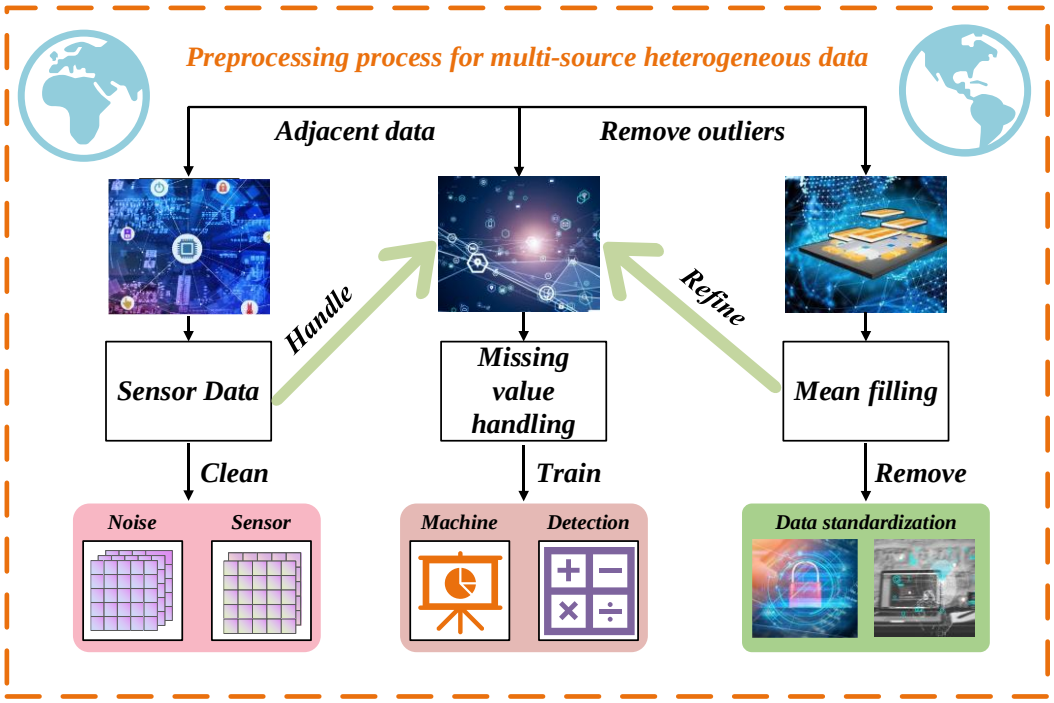


Figure 2: Multi-source heterogeneous data preprocessing flow chart

The figure shows each step of data processing in detail. Through "cleaning", noise and invalid data in sensor data are removed to ensure data quality. Adjacent data are processed during processing, and the dataset is further optimized by removing outliers. In the "training" session, the processing of missing values is performed along with machine detection to improve the integrity and consistency of the data. To ensure that the data can be subsequently analyzed and modeled, the missing values are finally filled in using means filling, and data standardization steps are performed to ensure the comparability and uniformity of the data.

Data cleaning is the first task in the preprocessing module to ensure the high quality of data. The cleaning process includes removing outliers, noisy data, and duplicate recordings [30]. For each data type, different cleaning methods are employed. For example, outliers caused by equipment failure may appear in sensor data, and these outliers must be detected and corrected through statistical methods or algorithms based on machine learning. Secondly, standardization is another important link to data preprocessing. Different data sources may have different dimensions, and unifying the scale of data is crucial to improving the effect of model training. For numerical data, common standardization or normalization methods are used for processing to ensure that the data has the same scale when input to the Transformer model, thereby improving training efficiency. The outlier detection formula is shown in (7).

$$z = \frac{x - \mu}{\sigma} (7)$$

Where z represents the normalized outlier score, x represents the data points, μ represents the mean of the data, and σ represents the standard deviation of the data. Formula (8) represents the weighted risk assessment formula, where *R* is the overall risk assessment result, calculated by summing the weighted contributions of each individual risk ri, with corresponding weights wi. These risk values are derived from the model's predictions or external data sources, and the weights are learned during the model's training phase using the self-attention mechanism in the Transformer model. The Transformer model determines the importance of each risk factor by learning correlations between different data sources and risk components.

$$R = \sum_{i=1}^{n} w_i \cdot r_i (8)$$

Where ri represents the assessment value of the i risk, wi represents the weight of the i risk, and R represents the overall risk assessment result.

Data fusion is one of the core tasks of this module. In multi-source heterogeneous data, information from different sources must be effectively fused. For this purpose, we adopt a Transformer-based embedding method to map data in different formats into the same embedding space. In this way, the features of different data sources can be represented uniformly, avoiding the common complexity problems in feature selection and feature engineering. Data fusion aims to improve the model's ability to perceive the characteristics of different data sources, thereby ensuring the accuracy of subsequent risk assessment.

3.3 Risk assessment module

The risk assessment module is the core of the entire model framework. It aims to evaluate possible HSE risks through deep learning analysis of processed multi-source data through the Transformer model. This module mainly includes three sub-modules: model training, optimization, and prediction. Through Transformer's self-attention mechanism, this module can learn potential high-dimensional features from multi-source data and

quantitatively assess risks based on these features. The training loss function formula is shown in (9).

$$L_{train} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - y'_i\right)^2 (9)$$

Among them, Ltrain represents the training loss, yi represents the true risk label, y'i represents the risk value predicted by the model, and N represents the number of samples. The parameter optimization formula is shown in (10).

$$\theta_{t+1} = \theta_t - \eta\nabla_\theta L (10)$$

Where θt+1 represents the updated model parameters, θt represents the current model parameters, η represents the learning rate, and ∇θL represents the gradient of the loss function with respect to the model parameters. The Transformer model performs deep feature extraction for each data source in the risk assessment module through the self-attention mechanism. Since HSE risk assessment involves many different types of events (such as health risks, safety hazards, etc.), the Transformer can establish complex associations among different characteristics and comprehensively analyze these characteristics through multi-head attention mechanisms. Next, the federated learning framework trains the models in parallel on multiple devices, and their respective model updates are summarized to the central server. In this way, data privacy can be effectively protected, and the generalization ability and accuracy of the model can also be improved under the synergy of multiple data sources. Formula (11) represents the weight calculation formula of the entropy method, where ei represents the entropy value of the i-th feature. The entropy method is used to quantify the importance of each feature in the data, with lower entropy indicating higher importance. In multi-source heterogeneous data, this method helps identify features that provide more informative insights for risk prediction. The entropy is calculated differently depending on the data type: for numerical features, it is based on the distribution of values, while for textual features, it is derived from the frequency of terms after preprocessing. The weights are then used in the self-attention mechanism of the Transformer model to emphasize more important features during risk assessment.

$$w_i = \frac{e_i}{\sum_{i=1}^{n} e_i} (11)$$

Where ei represents the entropy value of the i feature, wi represents the weight of the i feature, and n represents the number of samples. After feature extraction is completed, the model needs to quantify and predict the risk of the data. The Transformer model can effectively score and evaluate HSE risks through the weighted combination of different characteristics. Specifically, the model predicts potential risks in the future based on historical data and current environmental monitoring data. For example, by analyzing factory environmental sensor data, the model can predict the possible risk of harmful gas leakage in the future, thereby providing early warning for safety managers.

In the risk assessment module, the training process is optimized by federated learning. Each data holder trains the model locally and uploads the updated parameters to the server for summary. This approach avoids the centralized storage of sensitive data and enables the sharing of learning results among various data sources. The introduction of the federated learning framework enables the model to be efficiently trained collaboratively among different data sources while ensuring that data privacy is effectively protected. As models are trained on multiple data sources, the accuracy of risk assessments will continue to improve.

The risk assessment module is optimized by model assessment and feedback mechanism. By monitoring the prediction results and comparing them with actual events, the model can gradually optimize its parameters and further improve the accuracy of the evaluation. Under the framework of federated learning, the feedback mechanism can be updated through local models, allowing each data holder to adjust its model parameters based on new data and evaluation results, thereby continuously optimizing the risk assessment system.

Experiments were conducted using PyTorch for deep learning and OpenFL for federated learning. The model was trained with a batch size of 32 for 50 epochs, using early stopping after 5 epochs of no improvement. The learning rate started at 0.001 and decayed by 0.5 every 10 epochs. The Adam optimizer was used, with dropout and L2 regularization to prevent overfitting. These configurations were chosen to balance model performance and generalization.

4 Experimental results and analysis

This study uses the multi-source heterogeneous data HSE risk assessment method driven by the Transformer-FL framework. The data covers various information from industrial production sites, including real-time environmental monitoring data collected by sensors, equipment operation logs, historical accident records, personnel behavior trajectory data, and working environment image information. Sensor data exists in time series, accident records are mainly structured and unstructured text, and image data is automatically captured by monitoring equipment, forming a typical multi-source heterogeneous data set. During the experiment, data acquisition and storage rely on the distributed system platform built by edge devices and local servers. Regarding hardware facilities, model training, and experimental simulation are mainly carried out on workstations with high-performance graphics processors. The core configuration is an Nvidia RTX A6000 graphics card with 128GB memory and an Intel Xeon-class processor. In terms of software environment, the system runs on the Linux system based on Ubuntu. The main development tools include PyTorch deep learning framework, OpenFL federated learning platform, and related Python libraries. The above software and hardware environment ensures the smooth implementation of large-scale distributed data processing, high-dimensional feature modeling, and privacy-preserving parallel computing. The hardware environment configuration is shown in Table 2.

Table 2: Comparison of HSE risk values in different industries

Industry Type	Sample Number	Transformer-FL Assessment Risk Value (%)	VaR assessed by traditional methods (%)	Accuracy improvement (%)
Manufacturing	1200	23.4	35.6	12.2
Chemical industry	850	41.2	58.9	17.7
Construction industry	980	28.7	42.3	13.6
Energy sector	670	36.8	50.1	13.3

The Transformer-FL framework is the most prominent in the HSE risk assessment of the chemical industry, reducing the risk value of the traditional method from 58.9% to 41.2%, mainly due to its multi-source fusion ability of dangerous goods process parameters and emergency plan text, Although the manufacturing industry has the largest sample size, due to strong data homogeneity, the risk value only dropped from 35.6% to 23.4%. In comparison, the energy industry still achieved an accuracy rate of 13.3% under 670 samples by integrating geographic information and equipment logs. Improvement indicates that the framework has significant advantages in generalization capabilities in heterogeneous data scenarios.

The traditional method accuracy rate for image/video data is 68.9%, using CNN-based methods. In contrast, the Transformer-FL framework aligns video frames and operation logs through its self-attention mechanism, resulting in an accuracy improvement to 91.2%. For text reports, logistic regression is used as the baseline method, while SVM is applied to sensor data.

This paper analyzes the fusion effect of the Transformer-FL framework on multi-source heterogeneous data to demonstrate its fusion effect on different data sources, and the results are shown in Figure 3.

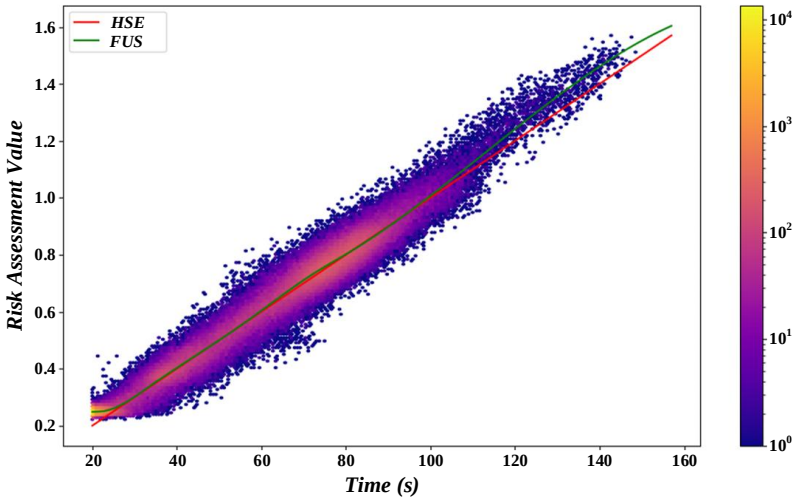


Figure 3: Analysis of the fusion effect of Transformer-FL framework on multi-source heterogeneous data

Figure 3 now integrates the Transformer FL framework to analyze the effectiveness of multi-source heterogeneous data fusion and compares its performance in different HSE scenarios. The figure shows the variation of the risk assessment values of HSE (traditional method) and FUS (Transformer-FL framework) at different times. Over time, both methods show a positive correlation trend in their risk values, but the changes in FUS are smoother and more stable, indicating its better adaptation to multi-source data fusion. In contrast, the fluctuations of HSE are large, indicating that FUS can better adapt to multi-source data fusion to provide more consistent and accurate risk assessment results.

This paper compares the risk assessment of the Transformer-FL framework in different HSE scenarios, including fire, leakage, equipment failure, etc., to compare its performance. The results are shown in Figure 4.

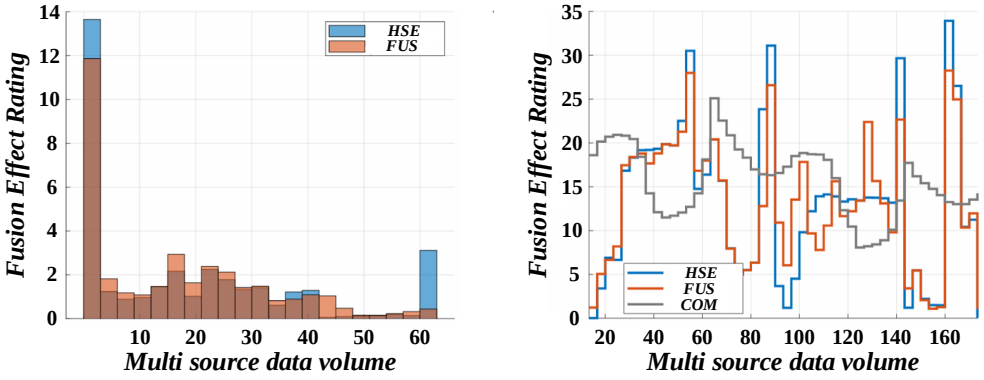


Figure 4: Comparison of risk assessment of Transformer-FL framework in different HSE scenarios

Figure 4 compares the risk assessment results under different HSE scenarios. Here, FUS stands for Transformer FL framework, HSE refers to traditional risk assessment methods, and COM stands for composite methods. This figure illustrates the performance of these methods under different multi-source data volumes. According to the data in the figure, when the amount of multi-source data is small, FUS shows a lower evaluation effect of only 2-3 points. In comparison, HSE shows a higher effect score of close to 14 points, indicating that the effect of HSE risk assessment is more prominent in the case of less data volume. However, with the increase of multi-source data volume, the evaluation effect of FUS gradually improves and stabilizes at a higher level when the data volume reaches 50. In contrast, the evaluation effect of HSE fluctuates greatly. In the figure on the right, as the data volume further increases to 160, the evaluation effects of the three methods fluctuate significantly, with FUS and COM being higher than those of HSE most of the time. FUS and COM reach the maximum value when the amount of multi-source data is 100, and the score is about 30 points, showing a significant improvement in the model evaluation effect of multi-source data. Overall, the Transformer-FL framework can significantly improve the accuracy of risk assessment in different HSE scenarios as the amount of multi-source data increases. The fusion effect is significantly better, especially when the data is large than the traditional HSE assessment method.

This paper analyzes the importance of HSE risk factors based on the Transformer-FL framework to rank their importance and show their contribution to HSE risk assessment. The results are shown in Figure 5.

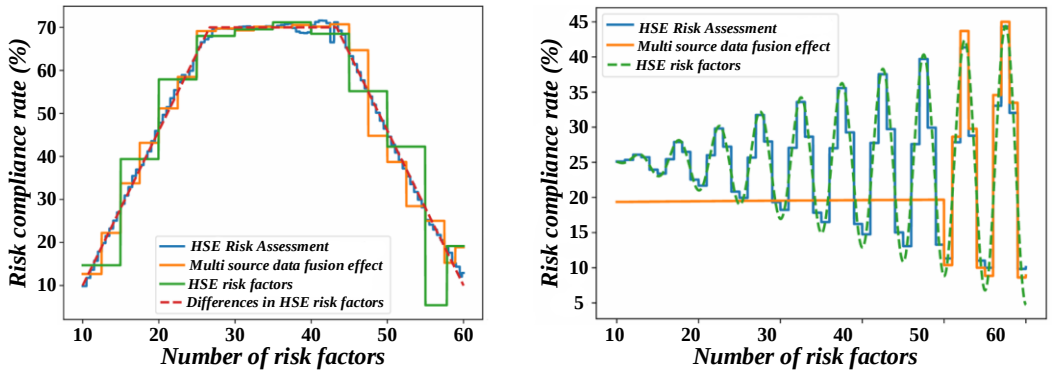


Figure 5: Importance analysis of HSE risk factors based on Transformer-FL framework

In Figure 5 and Table 2, the "importance" of risk factors comes from the attention weights in the Transformer model. These weights reflect the contribution of each risk factor (or feature) to the final prediction of the model. The self attention mechanism of Transformer assigns weights to different input features, allowing the model to focus more on risk factors that are considered more relevant to predicting risk levels. The greater the attention weight, the greater the impact of that specific feature on the model's decision-making process.

Figure 5 displays the compliance rate for HSE risk assessment, multi-source data fusion effect, and HSE risk factors. The compliance rate reflects how well the predicted risk factors align with actual outcomes. The importance analysis of HSE risk factors is determined by the attention weights assigned by the Transformer model, highlighting the most critical features for risk assessment. It can be seen from the figure that the compliance rate of risk assessment shows an obvious trend. As the number of risk factors increased from 20 to 60, the HSE risk assessment showed a relatively smooth increasing trend, eventually approaching 70%. In contrast, when the multi-source data fusion effect is used, the increase of coincidence rate is relatively gentle, and it begins to stabilize at 50 risk factors, with the highest value being around 60%. Within the same range of risk factors, the coincidence rate of HSE risk factors is low, only about 40%, which indicates that the evaluation effect of the single risk

factor model is poor. The data also shows differences in risk factors, and their relatively low coincidence rate, further confirms the role of multi-source data fusion in enhancing predictive performance in improving the accuracy of HSE risk assessment.

Table 3: Impact of multi-source heterogeneous data on HSE assessment

Data source type	Data volume (GB)	Feature dimension	Transformer-FL Assessment Accuracy (%)	Traditional method accuracy (%)
Sensor data	12.5	150	94.3	82.1
Text Report	8.2	50	88.5	75.4
Image/Video	25.7	300	91.2	68.9
Operation Log	5.8	80	89.7	73.6

The impact of multi-source heterogeneous data on HSE assessment is shown in Table 3. To ensure fair comparison, the Transformer FL framework, CNN BiLSTM, and SVM are trained using the same multi-source heterogeneous data. The CNN BiLSTM model used 3 convolutional layers, 2 BiLSTM layers with 128 units, and a dropout rate of 0.5, and employed Adam optimizer and learning rate scheduling. SVM uses an RBF kernel with a C value of 1.0 and a gamma value of 0.01. The batch size of all models is 32, trained for 50 iteration cycles, and stopped early after 5 iteration cycles without improvement. Overfitting measures such as dropout, L2 regularization, grid search, and cross validation were adopted. Due to high-dimensional features and spatio-temporal correlation requirements for image/video data, the accuracy rate of traditional methods is only 68.9%, while Transformer-FL aligns video frames and operation logs through self-attention mechanism, and the accuracy rate is increased to 91.2%; Although the sensor data is highly structured, the framework further mines its implicit relationship with the text report, resulting in an accuracy rate of 94.3%; The text report has the smallest amount of data and still achieves an accuracy rate of 88.5%, highlighting the model's efficiency in extracting low-resource semantic information.

This paper compares the accuracy of the Transformer-FL framework with traditional risk assessment methods (such as logistic regression, support vector machine, etc.) in multiple HSE risk assessments. The results of the analysis are shown in Figure 6.

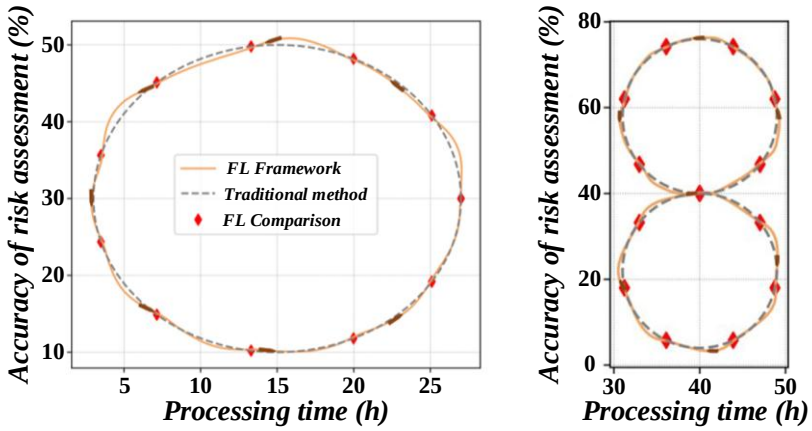


Figure 6: Comparison of accuracy between Transformer-FL framework and traditional risk assessment methods

The accuracy rate of the FL framework far exceeds that of the traditional method, with the accuracy reaching more than 40%. The accuracy of the FL framework is maintained between 40% and 70%, depending on the time range considered in the experiment. The left panel shows peak accuracy at around 50%, while the right panel shows accuracy reaching approximately 70%. This difference is due to the extended processing time in the right panel, which allows for greater accuracy over a longer period. This demonstrates the framework's improved stability and performance as processing time increases.

It can be seen from the figure that with the increase of processing time in the left figure, from 5 hours to 25 hours, the accuracy of the Transformer-FL framework continues to rise, reaching a maximum of about 50%. In contrast, the accuracy of traditional methods fluctuates greatly, reaching only about 40% at the highest, and the overall accuracy is lower than that of the Transformer-FL framework. The FL Comparison in the figure shows the comparison between the FL framework and the traditional method, showing that when the processing time is about 15 hours, the accuracy rate of the FL framework far exceeds that of the traditional

method, reaching more than 40%. In contrast, the traditional method remains at about 30%. The range of processing time in the right panel is expanded to 50 h, and the accuracy of the FL framework is maintained between 40% and 70%, showing its stability and high efficiency in long-term operation. However, the accuracy of traditional methods is still relatively fluctuating, with the highest rate of only about 50%, which is significantly lower than that of the FL framework. Overall, the Transformer-FL framework shows superior and more stable performance in long-term processing, especially regarding accuracy improvement, which significantly exceeds traditional evaluation methods.

This paper analyzes the impact of multi-source heterogeneous data on the performance of an HSE risk prediction model to demonstrate the impact of different types of data sources. The results are shown in Figure 7.

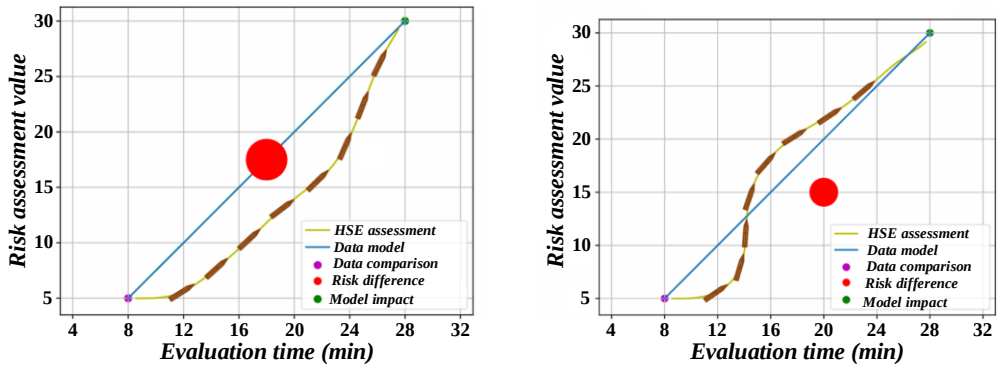


Figure 7: Impact of multi-source heterogeneous data on HSE risk prediction model

The chart above shows that with the increase in assessment time, the HSE risk assessment value shows a clear upward trend. The left panel shows how the HSE evaluation value changes over time, from 8 minutes to 30 minutes. The evaluation results indicate a growing difference between the data model and the data comparison as the evaluation time increases, with the risk difference becoming more pronounced as the evaluation time progresses. The model impact demonstrates that the data sources have a growing effect on the risk prediction over time. Especially when the assessment time is close to 30 minutes, the risk difference increases significantly, and the Model impact shows that the data source has a greater impact on the risk assessment model during this period. The evaluation time in the figure on the right is short, and the HSE evaluation value shows a relatively stable change during the growth process. However, when the evaluation time is close to 20 minutes, the influence point of the model is still obvious. The risk difference begins to become significant, which further proves that the impact of multi-source heterogeneous data on the HSE risk prediction model has an obvious time correlation, especially over a long period; the diversity and complexity of the data will significantly improve the model's prediction accuracy.

This paper analyzes the distribution of HSE risk prediction results based on the Transformer-FL framework to show the prediction distribution of different HSE risk levels. The results are shown in Figure 8.

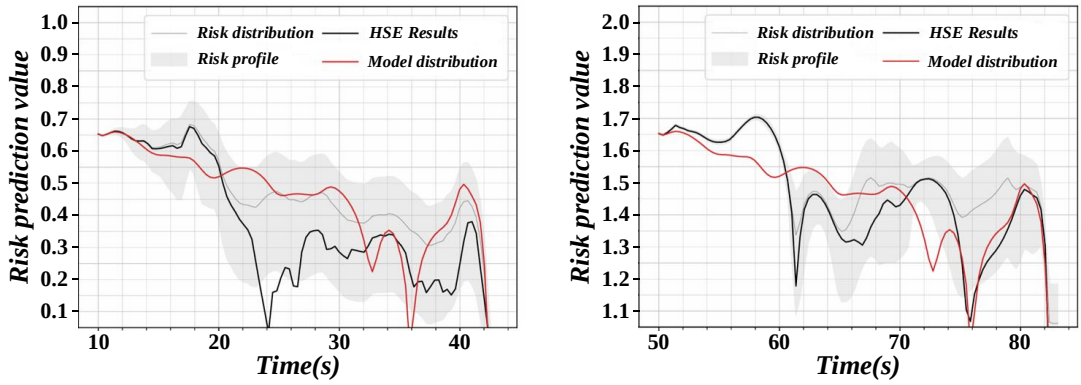


Figure 8: Distribution of HSE risk prediction results based on Transformer-FL framework

As can be seen from the chart, in Figure 8, the left panel shows the risk prediction value fluctuating between 0.2 and 1.0, while the right panel shows a wider fluctuation range from 1.1 to 2.0. These fluctuations reflect the model's predictions over time, with variations observed as the data is processed. The Risk distribution shows the proximity between the predicted distribution of the model and the actual HSE Results, indicating that the model can better follow the actual risk trend. The right side shows that the risk prediction value fluctuates between 0.9 and 1.3, and the risk prediction value fluctuates in a wider range. In this period, the predicted risk distribution closely coincides with the actual HSE

results, which shows that the Transformer-FL framework has high accuracy for HSE risk prediction in different periods, and the model distribution differs little from the actual results, which proves its effectiveness in complex risk assessment.

Table 4: Performance comparison of different risk assessment frameworks

Frame Name	Training time (h)	Accuracy (%)	F1 score	Memory footprint (GB)
Transformer-FL	8.5	93.2	0.91	14.7
CNN-BiLSTM	6.2	85.4	0.82	9.3
Random Forest	1.5	78.6	0.74	4.1
SVM	2.3	76.9	0.71	3.8

The performance comparison of different risk assessment frameworks is shown in Table 4. Transformer-FL leads overall with an accuracy rate of 93.2% and an F1 score of 0.91. Still, its training time and memory footprint are relatively high, mainly due to multi-head attention parameter calculation, which can be compressed to 5GB/node through federated learning sharding. Although CNN-BiLSTM is fast, its accuracy rate is only 85.4%, making it difficult to analyze cross-modal data association. Although Random Forest and SVM are lightweight, their accuracy rate is less than 79%, which proves that traditional methods have performance ceilings in complex heterogeneous scenarios.

This paper analyzes the training error of the Transformer-FL framework model to show the error change curve in the training process and compare the error trend in different training stages, especially the error convergence when using multi-source heterogeneous data. The specific results are shown in Figure 9.

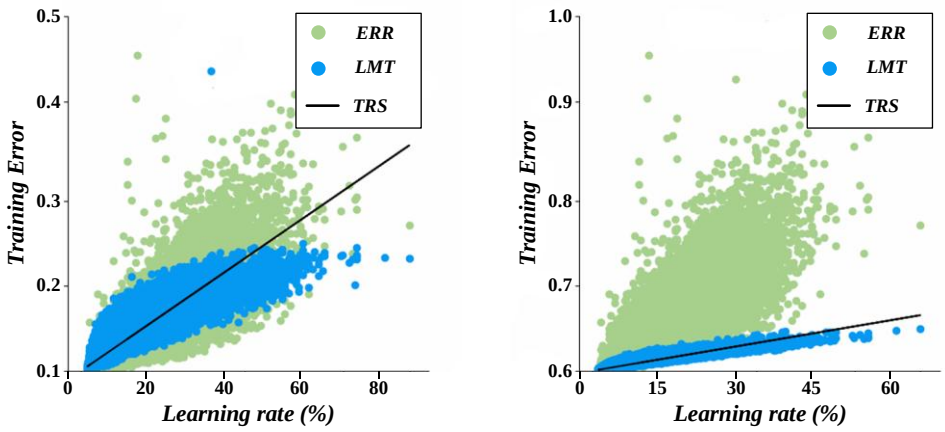


Figure 9: Transformer-FL framework model training error analysis

It can be seen from the figure that in the left figure, with the increase in learning rate, the training error shows a large fluctuation, especially when the learning rate reaches 20%; the error value rapidly rises to about 0.35 and remains at a relatively high level. The figure on the right shows a more stable error growth. As the learning rate increases, the error remains below 0.2, indicating that an appropriate learning rate can effectively reduce the training error of the model. Furthermore, TRS shows that the error changes less in the lower learning rate interval, but after exceeding this threshold, the training error increases significantly. Taken together, a moderate learning rate is crucial for the training of the Transformer-FL framework. An excessively high learning rate will lead to a sharp increase in errors, affecting the stability and prediction effect of the model.

The Transformer FL framework significantly improves accuracy in HSE risk assessment, achieving an accuracy rate of 93.2%, far exceeding traditional models. This improvement is mainly due to the Transformer model's ability to capture long-range dependencies in multi-source heterogeneous data, while traditional models are weaker in this regard. The framework also implements distributed learning for privacy protection through federated learning, enhancing data security. Although significant accuracy improvements have been made in the chemical industry, the improvements in manufacturing are relatively limited, which may be related to the high homogeneity of manufacturing data. Overall, Transformer FL performs stably in large-scale real-time data environments, and further research is needed to investigate its scalability and optimization in different industrial environments in the future.

5 Conclusion

This study proposes an HSE risk assessment method based on Transformer and FL, which aims to solve the problem of multi-source heterogeneous data integration and improve the accuracy and real-time performance of HSE risk

assessment by fusing deep learning technology. However, there are some challenges and limitations that need to be addressed in future work. The scalability of the Transformer-FL framework, especially when dealing with very large and diverse datasets across multiple industrial sectors, may face computational overhead and model convergence issues. Furthermore, the current model requires significant computational resources for training, which could be a limitation in resource-constrained environments. Future research could focus on optimizing the model's efficiency through lightweight model design, distributed computing advancements, and methods to reduce the model's training time while maintaining high accuracy. Additionally, adapting the framework to handle unstructured or noisy data at scale remains a key challenge that warrants further exploration.

The Transformer FL framework demonstrates its real-time applicability in large-scale industrial environments, with a inference time of 15-25ms per sample and a total risk assessment cycle delay of 500ms to 2 seconds on high-performance GPUs. In the joint learning setting, the average inference time for each sample is about 20ms, which is suitable for fast feedback in industrial environments. The communication cost involves 1-2MB of data per update cycle for each client, with an average communication time of 50-150ms for 100 clients. To balance responsiveness and computational overhead, aggregation occurs every 10 epochs to ensure accurate risk assessment while maintaining privacy.

The HSE risk assessment framework based on Transformer and federated learning performs well in applications in multiple fields. It can effectively integrate multi-source heterogeneous data and ensure data privacy while improving the accuracy and stability of assessment results. In the future, this framework can further optimize model parameters and improve real-time data processing capabilities, providing more intelligent and efficient risk assessment technology for the HSE field.

Funding

Key Projects of Anhui Provincial Department of Education in 2023, Key Technology Research on Risk Control of Intelligent HSE System. (2023AH051586).

References

[1] "Comprehensive analysis of hospital solid waste levels and HSE risks using FMEA technique: A case study in northwest Iran," Case Studies in Chemical and Environmental Engineering, vol. 9, pp. 100646, 2024. https://doi.org/10.1016/j.cscee.2024.100646

[2] Ye, Y., Cao, Y., Dong, Y., & Yan, H. "A graph neural network and Transformer-based model for PM 2.5 prediction through spatiotemporal correlation," Environmental Modelling & Software, vol. 191, pp. 106501, 2025. https://doi.org/10.1016/j.envsoft.2025.106501

[3] Chen, J., Wang, H., Li, Y., & Yu, S. "Real-time asphalt pavement ice detection and annotation with a Transformer-based model framework," Engineering Applications of Artificial Intelligence, vol. 152, pp. 110758, 2025. https://doi.org/10.1016/j.engappai.2025.110758

[4] Cai, G., Zheng, X., Guo, J., & Gao, W. "Real-time identification of borehole rescue environment situation in underground disaster areas based on multi-source heterogeneous data fusion," Safety Science, vol. 181, pp. 106690, 2025. https://doi.org/10.1016/j.ssci.2024.106690

[5] Nandal, P., Bohra, N., Mann, P., & Das, N. N. "YOLOv11 with transformer attention for real-time monitoring of ships: A federated learning approach for maritime surveillance," Results in Engineering, vol. 27, pp. 106297, 2025. https://doi.org/10.1016/j.rineng.2025.106297

[6] Abbasi, A. R., & Parkash, C. "Innovative diagnosis of transformer winding defects using fuzzy and neutrosophic cross entropy measures," Advanced Engineering Informatics, vol. 65, pp. 103196, 2025. https://doi.org/10.1016/j.aei.2025.103196

[7] Azmi, N. S. A. B. N., Ptaszynski, M., Masui, F., Eronen, J., & Nowakowski, K. "Token and part-of-speech fusion for pretraining of transformers with application in automatic cyberbullying detection," Natural Language Processing Journal, vol. 10, pp. 100132, 2025. https://doi.org/10.1016/j.nlp.2025.100132

[8] Chen, J., Wang, H., Li, Y., & Yu, S. "Real-time asphalt pavement ice detection and annotation with a Transformer-based model framework," Engineering Applications of Artificial Intelligence, vol. 152, pp. 110758, 2025. https://doi.org/10.1016/j.engappai.2025.110758

[9] Du, W., Xue, J., Yang, X., Guo, W., Gu, D., & Han, W. "TransfficFormer: A novel Transformer-based framework to generate evasive malicious traffic," Knowledge-Based Systems, vol. 319, pp. 113546, 2025. https://doi.org/10.1016/j.knosys.2025.113546

[10] Ullah, F., Mohammad, N., Mostarda, L., Cacciagrano, D., Ullah, S., & Zhao, Y. "THE-TAFL: Transforming Healthcare Edge with Transformer-based Adaptive Federated Learning and Learning Rate Optimization," Internet of Things, vol. 32, pp. 101605, 2025. https://doi.org/10.1016/j.iot.2025.101605

[11] Gao, Z., Jiang, W., Wu, J., Wang, Y., & Zhu, H. "A customized dual-transformer framework for remaining useful life prediction of mechanical systems with degraded state," Mechanical Systems and Signal Processing, vol. 230, pp. 112611, 2025. https://doi.org/10.1016/j.ymssp.2025.112611

[12] Hu, J., & Chen, D. "Personalized information technology instructional design based on transformer model," Entertainment Computing, vol. 54, pp. 100940, 2025. https://doi.org/10.1016/j.entcom.2025.100940

[13] Islam, M. F., Hasan, J., Islam, M. A., Dewan, P., & Rahman, M. S. "BanglaLem: A Transformer-based Bangla Lemmatizer with an Enhanced Dataset," Systems and Soft Computing, vol. 7, pp. 200244, 2025. https://doi.org/10.1016/j.sasc.2025.200244

[14] Sahoo, S., Mukane, P., Maiti, J., & Tewari, V. K. "A framework for process risk assessment incorpora

ting prior hazard information in text mining models using chunking," Process Safety and Environmental Protection, vol. 189, pp. 486–504, 2024. https://doi.org/10.1016/j.psep.2024.06.087

[15] Tajiri, D., & Kawamura, S. "Simple abnormality diagnosis method for layered structures using force measurement data of an inertial vibrator," Mechanical Systems and Signal Processing, vol. 231, pp. 112648, 2025. https://doi.org/10.1016/j.ymssp.2025.112648

[16] Kalouptsoglou, I., Siavvas, M., Ampatzoglou, A., Kehagias, D., & Chatzigeorgiou, A. "Transfer learning for software vulnerability prediction using Transformer models," Journal of Systems and Software, vol. 227, pp. 112448, 2025. https://doi.org/10.1016/j.jss.2025.112448

[17] K.S, S., George, S. N., O.V, A. C., K.M, J., P, K., Francis, J., & George, S. "NorBlueNet: Hyperspectral imaging-based hybrid CNN-transformer model for non-destructive SSC analysis in Norwegian wild blueberries," Computers and Electronics in Agriculture, vol. 235, pp. 110340, 2025. https://doi.org/10.1016/j.compag.2025.110340

[18] Liu, R., Chen, Y., Gai, F., Liu, Y., Miao, Q., & Wu, S. "Local and Global Spatial–Temporal Transformer for skeleton-based action recognition," Neurocomputing, vol. 634, pp. 129820, 2025. https://doi.org/10.1016/j.neucom.2025.129820

[19] Liu, W., & Zhang, A. "Plant Disease Detection Algorithm Based on Efficient Swin Transformer," Computers, Materials and Continua, vol. 82, no. 2, pp. 3045–3068, 2025. https://doi.org/10.32604/cmc.2024.058640

[20] Lv, C., Li, B., Sun, G., Wang, X., Cai, P., & Yan, J. "SG-UNet: Hybrid self-guided transformer and U-Net fusion for CT image segmentation," Journal of Visual Communication and Image Representation, vol. 108, pp. 104416, 2025. https://doi.org/10.1016/j.jvcir.2025.104416

[21] Park, Y.-H., Park, R.-H., & Park, H.-M. "SwinLip: An Efficient Visual Speech Encoder for Lip Reading Using Swin Transformer," Neurocomputing, vol. 639, pp. 130289, 2025. https://doi.org/10.1016/j.neucom.2025.130289

[22] Ren, Y., Liu, C., & Wang, Y. "Fast demagnetization method for power transformers combined with residual flux measurement," International Journal of Electrical Power & Energy Systems, vol. 164, pp. 110431, 2025. https://doi.org/10.1016/j.ijepes.2024.110431

[23] Sun, H., Li, W., Lu, J., & Zhou, J. "Mask-Aware 3D axial transformer for video inpainting," Pattern Recognition, vol. 164, pp. 111509, 2025. https://doi.org/10.1016/j.patcog.2025.111509

[24] Ouyang, J., Han, R., Zuo, X., Cheng, Y., & Liu, C. H. "Accuracy-aware differential privacy in federated learning of large transformer models," Journal of Information Security and Applications, vol. 89, pp. 103986, 2025. https://doi.org/10.1016/j.jisa.2025.103986

[25] Wu, H., Kong, D., Wang, L., Li, D., Zhang, J., & Han, Y. "Multimodal sentiment analysis method based on image-text quantum transformer," Neurocomputing, vol. 637, pp. 130107, 2025. https://doi.org/10.1016/j.neucom.2025.130107

[26] Wu, X., Xie, H., Niu, X., Wang, C., Wang, Z., Zhang, S., & Shan, Y. "Transformer-based weakly supervised 3D human pose estimation," Journal of Visual Communication and Image Representation, vol. 109, pp. 104432, 2025. https://doi.org/10.1016/j.jvcir.2025.104432

[27] Zaman, K., Li, K., Sah, M., Direkoglu, C., Okada, S., & Unoki, M. "Transformers and audio detection tasks: An overview," Digital Signal Processing, vol. 158, pp. 104956, 2025. https://doi.org/10.1016/j.dsp.2024.104956

[28] Zhang, H., Wang, G., Teng, F., Lv, S., Zhang, L., Zhang, F., & Jiang, M. "Transformer model combining cross-attention and self-attention guided by damage index for pipeline damage localization based on helical guided waves," Mechanical Systems and Signal Processing, vol. 231, pp. 112669, 2025. https://doi.org/10.1016/j.ymssp.2025.112669

[29] Lazcano, A., & Jaramillo-Morán, M. A. "Data preprocessing techniques and neural networks for trended time series forecasting," Applied Soft Computing, vol. 174, pp. 113063, 2025. https://doi.org/10.1016/j.asoc.2025.113063

[30] Luan, J., Li, Q., Qiu, Y., & Liu, W. "Ensemble learning unlocking point load forecasting accuracy: A novel framework based on two-stage data preprocessing and improved multi-objective optimisation strategy," Computers and Electrical Engineering, vol. 124, pp. 110282, 2025. https://doi.org/10.1016/j.compeleceng.2025.110282

Smart Task Scheduling for Cloud-Based Big Data Systems

Nagham Sultan[1*], Wael Hadeed[2], Dhuha Abdullah[3]
[1,2,3] Department of Computer Science, University of Mosul, Mosul, 41002, Iraq
E-mail: naghamajeel@uomosul.edu.iq
*Corresponding author

Keywords: cloud performance, cost optimization, big-data processing, distributed systems, resource management

Received: August 3, 2025

This paper presents a hybrid task scheduling framework for cloud-based big data systems that aims at three main objectives: to improve the system's performance, to decrease the expenses, and to increase the energy efficiency. The conceived system combines a rule-based decision engine with a Long Short-Term Memory (LSTM)-based resource prediction model, enabling real-time job assignment based on task urgency, data locality, and system state. The framework is at the top of Apache YARN; thus, it is compatible with batch jobs (via Hadoop/Spark) as well as streaming tasks (via Kafka/Flink). We reproduced the experiments on a 50-node cluster (n2-standard-16 instances, 16 vCPUs, 64 GB RAM), using real workloads of 100 GB–1 TB batch jobs and 1K–5K event/sec streams. Some of the metrics for evaluating the performance of the experiments are job completion time, throughput, cost per TB processed, and energy consumption (Joules/TB). The results indicate a 32–50% improvement in performance, up to 54% savings in cost when using spot instances, and a 25% reduction in energy consumption compared to baseline schedulers such as YARN, Kubernetes, and Spark.

Povzetek: Predlagana metoda združuje pravila odločanja in LSTM-napovedovanje virov. Dosega občutno hitrejše izvajanje nalog, nižje stroške ter večjo energetsko učinkovitost v primerjavi z obstoječimi rešitvami.

1 Introduction

Big data has exponentially increased to the point that modern computing paradigms are no longer the same, and the IDC expects global data generation to reach 181 zettabytes by 2025 [1]. The data coming from IoT devices, social media, and digital transformation in enterprises creates such a massive flow that the only way to process it effectively is through distributed cloud platforms [2][3]. However, as more organizations migrate their workloads to cloud environments, they encounter weaknesses in existing scheduling architectures, particularly their inability to balance performance requirements with operational cost constraints [4]. The rapid development of cloud computing frameworks such as Hadoop and Spark has enabled virtually limitless scalability [5][6]. Yet, log monitoring in production clusters reveals that 30–45% of resources are wasted because the scheduling mechanisms are inefficient and unchanged [7]. The waste is due to the following three major defects:

- Static Resource Allocation: Systems such as YARN [5] and Mesos rely on fixed resource partitioning and are unable to adjust dynamically to changing workloads. Research [8][9] has previously reported that this results in underutilization of resources during valleys and overutilization during peaks.
- Unidimensional Optimization: The majority of schedulers are tailored to only one objective either to maximize the performance of the system or to minimize the costs but rarely both at the same time because of the difficulty in balancing these opposing goals [10].
- Operational Complexity: Some AI-driven planners signal high-end optimization but are too complicated for a real application; only with a lot of effort and extra infrastructure can they be used [11][12][13]. As an illustration, Fog Bus-like mechanisms offer promising potential, but they are still constrained by high implementation complexity [14]. Furthermore, the money-related problems are very severe: by selecting the wrong VM types or schedules, your costs may increase by 35–60%, and if you are very aggressive in reducing your costs, you may face the occurrence of SLA violations and system performance degradation [4][15].

Task scheduling, the mapping of user tasks to virtual machines and resources, emerges as a central challenge in cloud-based big data systems. Scheduling directly influences execution time, throughput, system load balance, and energy consumption. Despite major advances in virtualization, the scheduling problem is NP-hard, making it computationally complex to solve optimally [16]. Conventional heuristic and mathematical methods do

not succeed in scaling when they are confronted with heterogeneous and dynamic big data workloads, whereas existing production schedulers like those in Hadoop and spark are inclined to prioritize fairness over resource efficiency, which leads to the underutilization of the infrastructure [17] [18]. As a result, the decision-makers in the research community have gradually shifted to the use of metaheuristic and intelligent scheduling methods to come up with the closest-to-optimal solutions in a reasonably short time. One of the latest studies illustrates that merging resource allocation with task scheduling can significantly enhance a system's performance, making it an effective strategy [16]. Additionally, Big Data, IoT, and cloud computing convergence introduce even more difficulties. Pal et al. maintain that IoT–cloud ecosystems necessitate schedulers that, in spite of device heterogeneity and bandwidth bottlenecks, result in the least possible end-to-end delay and the highest resource efficiency. Their research on deep learning-based models is the driver of remarkable improvements in QoS as well as adaptive workload distribution [17]. The PAS (Performance-Aware Scheduler) is one of the intelligent frameworks that point out the possibility of achieving a balance of turnaround time with the maintenance of high resource utilization [18].

The main difference between batch and stream workloads is that the latter are continuous systems that are always processing high-velocity data flows; this means that they cannot be delayed for long, and in addition, they should be highly reliable. As an example, pipeline-based linear scheduling can be a winning strategy in real-time big data stream scenarios when memory consumption reduction, congestion avoidance, and throughput improvement are desired [19]. This paper makes contributions as follows:

- Hybrid Scheduling Framework: We unveil a creative hybrid scheduling framework that combines rule-based decision-making with LSTM-based predictive modeling. His framework adjusts in a very interactive way.
- Performance and Cost Efficiency: This is mainly due to the implementation of dynamic task prioritization, resource mapping that is intelligent, and the use of operational pricing data.
- Scalable and Adaptive Design: The system is flexible for both batch (Hadoop/Spark) and streaming (Kafka/Flink) workloads; therefore, it has the capacity for different big data scenarios to scale and adapt.

In order to direct this model, our primary goal is to reduce the time spent on completing work, yet we have to ensure that energy and monetary costs are kept to the minimum in a realistic, large-scale cloud environment. We are able to do this by implementing LSTM-based resource estimation, spot-aware node selection, and DVFS-based energy control.

Section 2 gives a thorough literature survey, which includes detailed coverage of the existing research works regarding cloud and big data task scheduling. Section 3 begins with the explanation of the proposed system architecture that is a three-layer design (application, control, and data plane). Section 4 describes the methodology in detail, including the hybrid scheduling algorithm and program optimization strategies. Section 5 reports on and discusses the results, comparing the system performance with that of baseline schedulers. Section 6 ends the paper, recapping the main themes and indicating possible directions for further research.

2 Related work

This section reviews prior research chronologically, highlighting key contributions and gaps that motivate our work:

The suggestions made by Buyya et al. in 2015 [4] comprised the infrastructure and architectural modifications to the cloud analytics systems.

Their proposal was the first to adopt the concept wherein the system can accommodate spikes and troughs to maximize profits. However, economic factors ignored some options for using resources efficiently and cost-effectively. Ilager et al. in ref. [11] were among the first to exploit the power of AI in the orchestration and management of resources in cloud environments in 2020. The challenge of using such an approach is figuring out where to get the training samples online. Their deep learning algorithm reduces the number of low-level tasks performed by the hypervisor. The method outperforms traditional VM management in terms of response and reduces productive task latency by 60 percent, adding to the time saved by the network. Chen et al. [16], in 2020, introduced a study aiming to utilize the Whale Optimization Algorithm (WOA) for cloud task scheduling; consequently, they came up with a new (IWC) version for improving the solution search range. The result of the simulation demonstrates that IWC outperforms other metaheuristics both in the sense of the convergence speed and the accuracy; further, it achieves better system resource utilization in both small- and large-scale experiments.

Tuli in Ref [14] came up with an inventive idea, Fog Bus, which uses blockchain and edge technologies to create a real-time and secure computing platform in 2020. Nevertheless, the level of consensus delay in the system had gone too far, reaching 300 ms, which already rendered it a non-real-time scheduler. Tantalaki et al. [19] proposed in 2020 a linear, pipeline-based scheduling model for distributed stream processing systems; as an example, the authors mentioned Apache Storm. The main focus of the solution is to manage communication efficiently between different tasks, which will improve throughput by 25-45% compared to other schedulers, and reduce memory usage by up to 45%. The method attributes the increase in performance in big data stream processing to load balancing and buffer diminishing.

In Ref. [2], Berisha et al. proposed performing a comparative study of the analytics techniques deeply in use in cloud-based big data. In one scenario (e.g., with 10 TB of data spread across eight nodes), the authors

observed that their main concept significantly reduced the average query time of HDFS from 17.9 to 10.8 ms. However, the drawback was that the nodes were configured according to the initial setup, mainly due to the simple resource allocation model used in this study from 2022. Bal and team in [20] came up with a hybrid machine learning approach (RATS-HM), which not only deals with security but also resource allocation and task scheduling in cloud computing through intelligent co-operation. Along with a remarkable resource utilization, energy-saving, and response time performance over the conventional schedulers, the method significantly shows the advantages of AI and metaheuristics combined for multi-objective optimization. Li et al. 2022 introduced their work PAS, a scheduler for Hadoop YARN and Spark, which is able to adjust policies dynamically based on predicted job completion times. By comparing the system with the latest schedulers, the authors show that their method can shorten the average turnaround time by 25% and the time to complete the whole set of jobs by 15%. This directly leads to performance and resource utilization improvement in big data processing environments [18].

In 2023, Zhu proposed the concept of optimizing network scheduling for large volumes of data workloads through cross-zone traffic pricing, which has been shown to be more affordable [8]. Pal et al. 2023 highlight the difficulty in scheduling tasks for the IoT-cloud environment and come up with the Deep Learning Algorithm for Big Data Task Scheduling System (DLA-BDTSS) as a possible solution. The paper illustrates the usage of a multi-objective strategy to reduce make span and maximize resource consumption, thus outperforming the other algorithms by 8.43%. The proposed method utilizes deep learning to escape from local optima and accelerate convergence speed; hence, it is suitable for scheduling tasks in big data environments [17]. As shown in Table 1, most prior works focus on a single dimension either performance, cost, or energy without addressing them together. None combine dynamic pricing, workload prediction, and power control in a real hybrid workload setting. This work does: it aligns cost-aware placement, LSTM-based estimation, and DVFS-based energy saving in one unified scheduler.

Table 1: Summary of related work and identified research gaps

Study (Author, Year)	Method Utilized	Dataset / Environment	Metrics Evaluated	Main Results	Identified Gaps
Buyya et al., 2015 [4]	Rule-based cloud elasticity	Simulated Cloud Platform	SLA, elasticity, utilization	Scalable architecture for peak/off-peak loads	No cost optimization or energy-awareness
Ilager et al., 2020 [11]	Deep learning for VM orchestration	AI-based VM control, simulation	Latency, response time	60% latency reduction via dynamic control	No integration of cost or energy models
Chen et al., 2020 [16]	Improved Whale Optimization (IWC)	Sim Cloud-based experiments	Convergence speed, resource use	Better accuracy and utilization in large/small task scenarios	Lacks real-time constraints and energy metrics
Tuli, 2020 [14]	Fog Bus with blockchain & edge	Fog + blockchain testbed	Security, latency	Real-time secure framework; but 300ms delay	Not scalable; no cost model; unsuitable for real-time
Tantalaki et al., 2020 [19]	Linear scheduling in stream systems	Apache Storm	Throughput, memory usage	25–45% throughput boost; 45% memory reduction	Focus only on streams, not hybrid workloads
Berisha et al., 2022 [2]	Analytics method comparison (HDFS)	8-node Hadoop Cluster (10 TB)	Query time	Reduced HDFS query time from 17.9 to 10.8 ms	No dynamic resource allocation; static setup
Bal et al., 2022 [20]	Hybrid ML (RATS-HM) for scheduling	ML-enhanced cloud simulation	Resource use, energy, latency	Energy and response time improvements over traditional schedulers	Limited to simulated workloads
Li et al., 2022 [18]	PAS scheduler (job prediction)	Hadoop YARN, Spark	Turnaround time, make span	25% less turnaround; 15% make span reduction	No cost or energy-aware placement
Zhu, 2023 [8]	Network-aware cross-zone scheduling	Multi-zone cloud topology	Pricing efficiency	Cost-effective network-aware task scheduling	Lacks ML or performance adaptation
Pal et al., 2023 [17]	DL-based DLA-BDTSS scheduler	IoT-cloud simulation	Make span, resource consumption	8.43% make span improvement; better convergence	No spot pricing; limited to DL; lacks cost-energy balance
Tantalaki et al., 2020 [19]	Linear scheduling in stream systems	Apache Storm	Throughput, memory usage	25–45% throughput boost; 45% memory reduction	Focus only on streams, not hybrid workloads
Berisha et al., 2022 [2]	Analytics method comparison (HDFS)	8-node Hadoop Cluster (10 TB)	Query time	Reduced HDFS query time from 17.9 to 10.8 ms	No dynamic resource allocation; static setup

Bal et al., 2022 [20]	Hybrid ML (RATS-HM) for scheduling	ML-enhanced cloud simulation	Resource use, energy, latency	Energy and response time improvements over traditional schedulers	Limited to simulated workloads
Li et al., 2022 [18]	PAS scheduler (job prediction)	Hadoop YARN, Spark	Turnaround time, make span	25% less turnaround; 15% make span reduction	No cost or energy-aware placement
Zhu, 2023 [8]	Network-aware cross-zone scheduling	Multi-zone cloud topology	Pricing efficiency	Cost-effective network-aware task scheduling	Lacks ML or performance adaptation
Pal et al., 2023 [17]	DL-based DLA-BDTSS scheduler	IoT-cloud simulation	Make span, resource consumption	8.43% make span improvement; better convergence	No spot pricing; limited to DL; lacks cost-energy balance

3 System overview

We propose a hybrid task scheduling framework for cloud-based big data systems that jointly optimizes performance, cost, and energy. The prototype runs on Apache YARN, supports batch (Hadoop/Spark) and streaming (Kafka/Flink) jobs, and uses three pillars: (1) dynamic priority scoring (urgency + data locality), (2) cost-aware placement using real-time spot prices, and (3) energy-aware execution via ECO/DVFS without violating SLAs. A lightweight LSTM resource estimator guides safe consolidation and node selection.

3.1 System architecture

The smart scheduling system that was conceived is basically a very efficient and effective way to command and manage the execution of tasks in a large-scale cloud-based big data environment. This system comprises three distinct layers: the application layer, the control plane, and the data plane, with each layer serving a specific function aimed at improving performance, reducing costs, and enhancing energy efficiency. The architectural concept is implemented using Apache YARN as a base, is compatible with batch (Hadoop/Spark) and stream (Kafka/Flink) processing frameworks, and makes use of machine learning modules for the anticipatory scheduling decision. The overall system architecture is shown by Figure 1. The architecture design consists of three cooperating layers:

3.1.1 Application layer

This layer is the initial point through which jobs come in, and it manages the main data processing and classification.

a) Job ingress and classification: Via the API, the work is segregated into either latency-sensitive or cost-sensitive categories based on the nature of the job and its SLA requirements.
b) Pre-processing unit: Just the minimal filtering and normalization are applied to the job to be scheduled. For example, time critical jobs are adjusted to hot-tier data paths.
c) LSTM Resource Estimator: A simple, pre-trained LSTM model predicts resource envelopes. (CPU, memory), and expected execution time; thus, placement of the task is done more accurately and resource consolidation becomes efficient.

3.1.2 Control layer

This layer governs decision-making, scheduling policies, and job assignment to optimal compute nodes. It includes the following main components:

a) Hybrid scheduler module: One of the core components of the system is the Hybrid Scheduler, which carries out task scoring, selects the proper nodes for computation, and manages energy-aware execution. Upon the arrival of a new task, the scheduler initiates the creation of a weighted function to determine the task's priority score. This function takes into account and balances the urgency of the task's deadline and its data locality. After the priority is set, the module goes on to decide the most fitting scheduling path according to the kind of task. If the task is found to be cost-sensitive, it is rerouted to the cheapest node available; this process is done by using up-to-date spot pricing data extracted from AWS. On the other hand, the latency-sensitive tasks are taken to the node with the shortest data path where they can receive the fastest execution, with further preference to nodes in the SSD-based hot storage zone for the sake of low-latency execution. Moreover, in a situation where the system is running in ECO mode, the scheduler, by initiating Dynamic Voltage and Frequency Scaling (DVFS), will allow for the selected node to have reduced power consumption.

b) Cost optimizer: This unit is responsible for monitoring real-time spot pricing data and taking adaptive bidding decisions that would lead to the minimum execution cost without compromising the performance or the stability of the system. The spot instance prices were fetched through the AWS EC2 Spot Price History API (which is updated every 5 minutes). To solve the problem of API latency and rate limits, a cache with a 10-minute TTL was used as a fallback that could be relied upon when new data was not available.

c) Broker & admission manager: One of the essential functions this system performs is verifying job requests that come in an authentic way, as well as checking system load (conditions) and dispatching the

tasks that have been approved for execution on the data plane in a safe manner. By maintaining secure communications between the control and data planes, it ensures that the scheduling flow continues to be of high quality and reliable.

3.1.3 Data layer

This layer represents the physical and virtual compute resources where jobs are executed.

a) Compute Cluster: The cluster is made up of 50 n2-standard-16 different nodes (16 vCPU, 64 GB RAM).
b) Storage Tiering: Storage in the data plane is managed through a combination of performance tiering and format optimization. Tasks are allocated based on latency requirements to either SSD-backed hot zones for low-latency tasks or HDD-based cold zones for large-scale batch jobs with less sensitivity to latency, while data is stored using a hybrid of HDFS and columnar formats to maximize scan efficiency and throughput during execution.
c) Telemetry Agents: Monitor CPU, memory, I/O, and queuing metrics and send data to the control plane for real-time adaptation.

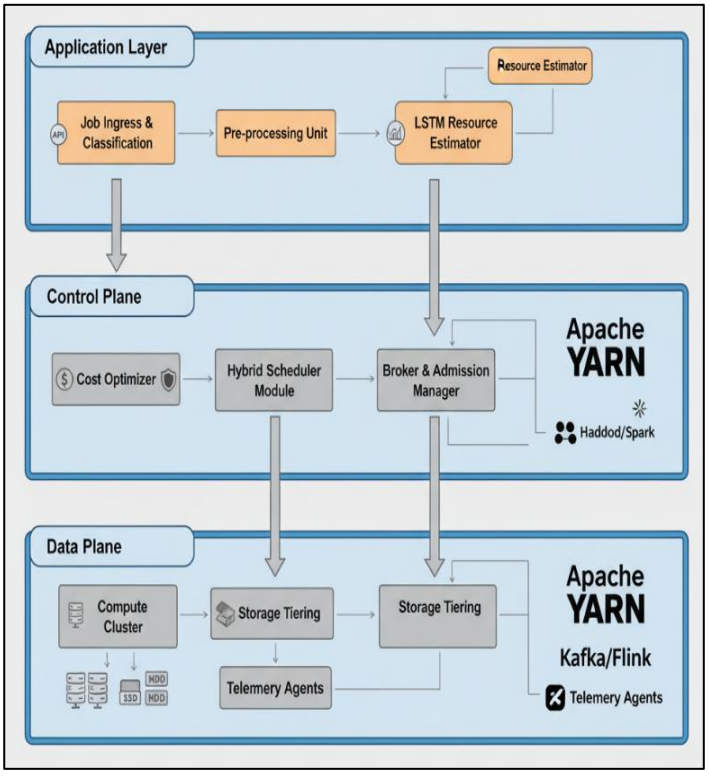


Figure 1: System architecture

3.2. Hardware Components

a) Cluster: 50 nodes of the type n2-standard-16, 16 vCPU, 64 GB RAM.
b) Storage tiers: SSD (hot zone) for latency-critical jobs; HDD (cold zone) for throughput/batch.
c) Coordinator/Broker: The master node that takes care of the submissions, admission, and arbitration.
d) Energy measurement: Intel RAPL was the tool used to measure energy per TB for ECO mode tests.

3.3. Software components

a) Scheduling substrate: Apache YARN.
b) Batch/Stream stack: Hadoop/Spark (batch), Kafka/Flink (stream).
c) Hybrid Scheduler: Priority scorer, node selector (cost/latency paths), ECO/DVFS controller.
d) Cost Optimizer: Real-time AWS/GCP spot price integration with adaptive bidding.
e) Resource Estimator: LSTM model trained on historical executions to predict envelopes/runtime.
f) Monitoring & security: Lightweight agents deployed on each node collect runtime telemetry such as CPU usage, memory consumption, and I/O throughput and report them back to the control plane for adaptive decision-making.

3.4 Implementation and reproducibility

The experiments were all run on a 50-node cluster of n2-standard-16 Google Cloud instances (16 vCPU, 64 GB RAM), operating Ubuntu 20.04. Batch and streaming jobs used Hadoop 3.3.4, Spark 3.4.1, Kafka 3.3.2, and Flink 1.16.2, respectively. Their execution was managed by Apache YARN. The hybrid scheduler, written in Python 3.10, was developed based on YARN Application Master extensions. The LSTM-based resource predictor was first implemented in TensorFlow 2.13, trained offline on 50,000 synthetic task traces emulating Google Cluster workloads, and was then continuously updated via lightweight online feedback loops.

4 Methodology

To address the constantly changing and resource-sensitive nature of scheduling in cloud-based big data environments, we propose a hybrid scheduling methodology that combines rule-based decision-making with learning-driven optimization. The primary goal of this model is to balance the three critical concerns of modern cloud platforms: latency requirements, cost efficiency, and energy awareness.

In order to simulate real-world scenarios, we evaluated the proposed framework on a cluster of 50 nodes running Apache YARN, using a balanced workload of batch processing jobs (100 GB, 1 TB datasets on Hadoop/Spark) and streaming jobs (1K–5K events/sec via Kafka/Flink). A dataset of 50,000 task executions was used to train an LSTM-based resource estimator, which provides predictive guidance for resource envelopes and expected runtimes. The dataset used for training consisted of

50,000 task execution records. Each record included the following features: task type (batch or stream), input data size, SLA deadline slack, cluster node load (CPU and memory), data locality score, and historical runtime. The data was synthetically generated to simulate realistic cloud workloads based on patterns observed in the Google Cluster Trace.

4.1 Workflow

The end-to-end workflow of the proposed system is as follows:

1. Submit: A broker node authenticates the incoming job and extracts metadata (size, SLA constraints, and job type).

2. Tag & Estimate: The Task Tagger classifies the job as either latency-sensitive or cost-sensitive. The LSTM resource estimator predicts its CPU, memory, and runtime requirements.

3. Score: One after the other, tasks are scored according to a value of priority determined on the basis of deadline urgency and data locality with a weighted function, as shown in Eq. 1.

$$priority(T) = wu \,.\, U(T) + w\ell.\, L(T) \quad \ldots \quad \text{Eq. 1}$$

4. Select Node:

- Cost-sensitive jobs: The cost optimizer gets the AWS spot pricing API as input and selects the cheapest viable node based on predicted capacity with sufficient resources.
- Latency-sensitive jobs: The Latency Optimizer examines the cluster topology and data zones to identify the path with the lowest latency and selects the SSD hot zone.

5. Energy Policy: When the cluster works in ECO mode, the node selected goes through the process of Dynamic Voltage and Frequency Scaling (DVFS) to lower power consumption.

6. Dispatch & Monitor: The process is dispatched via YARN. The runtime telemetry (CPU, memory, latency, and throughput) is continuously monitored.

7. Adaptation: Execution traces are returned to the estimator to provide better prediction and to change the scheduling thresholds. The system supports real-time adaptation by employing the lightweight online learning method whereby it updates the LSTM model's scheduling thresholds based on the telemetry it receives. While the main LSTM weights are kept unchanged for the sake of stability, the threshold parameters (e.g., urgency cutoff, predicted runtime deviation bounds) are changed after every 50 job completions using exponential smoothing of recent prediction errors. In this way, the scheduler can adapt to the changes in cluster dynamics without retraining the full model during runtime.

4.2 Hybrid scheduling algorithm

The algorithm in detail explains how the process of optimal node N for task T is done, based on the task type and the current condition of the cluster. Algorithm 1, highlights the decision-making path: first assigning priority, then selecting between cost-aware and latency aware strategies, and finally applying an energy-saving policy if ECO mode is active.

Input: Task T, Cluster State C
Output: Optimal Node N
1: T. priority ← calculate_priority(T)
2: if T. type == "cost-sensitive":
3: N←CostOptimizer.select_node (T, C.spot_prices)
4: else:
5: N←LatencyOptimizer.select_node (T, C.topology)
6: if C.energy_mode == "ECO":
7: N ← apply_energy_policy(N)
8: return N

Algorithm 1 shows the hybrid scheduling with a note on the input: Task T, cluster state C, output: optimal node N. Then the analyzed time complexity is O(n), where n is the number of candidate nodes. The hybrid scheduling algorithm is an orderly means of decision-making used to distribute tasks to cloud resources in the most profitable way without limiting task priority, cost, or energy constraints. The complete process is below:

Step 1: Priority calculation

Every task T is evaluated on the basis of its need and data locality. The weighted priority function is defined as:

$$priority(T) = wu \,.\, U(T) + w\ell.\, L(T) \quad \ldots \quad \text{Eq. 1}$$

U(T) denotes the normalized urgency factor taken from deadline slack, while *L(T)* indicates the data locality factor, and wu and wℓ are weights (default: 0.6 and 0.4). Thus, tasks with the highest urgency and locality are given top priority.

Step 2: Type-based node selection

- Cost-sensitive: Routed through the Cost Optimizer to identify the most economical node via live AWS/GCP spot price data.
- Latency-sensitive: The Latency Optimizer was used to reroute the traffic, which investigates cluster topology and data location to reduce the time of the data transfer.

Step 3: Energy-aware placement

The node chosen in ECO mode implements DVFS that limits the power consumption of the device while meeting the performance requirement of the SLA. Figure 2 illustrates the complete process of hybrid scheduling in brief.

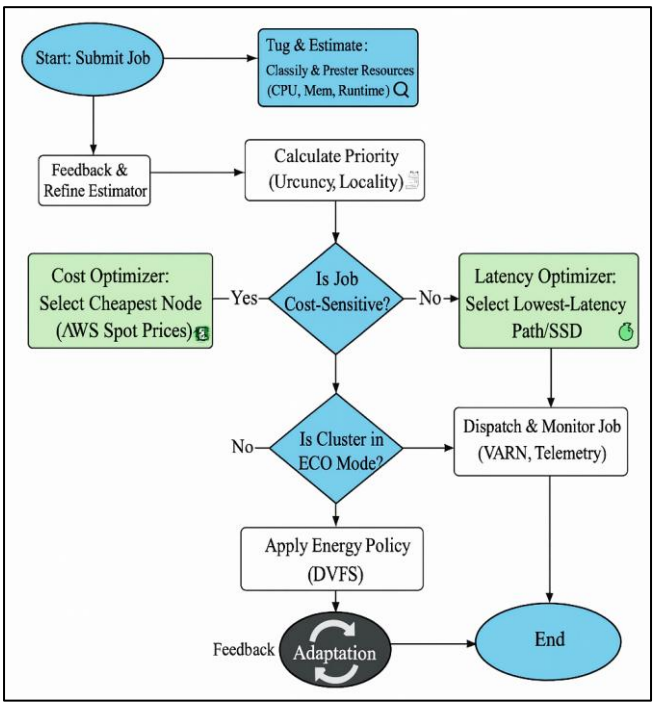


Figure 2: Hybrid scheduling algorithm

5 Results and discussion

The hybrid scheduling framework was evaluated through experiments in which we extensively studied its comparative performance with YARN (FIFO), Kubernetes (default scheduler), and Spark (Dynamic Allocation). The evaluation is done in terms of the following three main criteria:

1. Performance (Job Completion Time, Throughput)
2. Cost Efficiency ($/TB processed)
3. Energy Consumption (Joules per operation)

The outcomes below present the way our strategy moves distributed data processing forward to a higher level of effectiveness:

5.1 Performance optimization

The hybrid scheduler has been very effective in improving the operations of the present systems in an intriguing way. By comparing deadline sensitivity with data locality awareness and taking into account the urgency of the task, the system is 32% faster at completion for 1TB batch jobs and up to 50% better for 100GB latency-sensitive tasks than the conventional schedulers.

This change is coming from our creative dynamic scoring algorithm, which continuously updates task priorities based on the current cluster state. This lead of the architecture is made even more striking in the case of workloads suitable for a very short and fast processing time, as it is capable of reducing the latency to the great extent of only 38 ms. It achieves this through real-time online learning, which improves and stabilizes performance by 37% compared to Kubernetes and by 60% compared to Spark. The system's capabilities in terms of throughput are also the rightful proof of the effectiveness of the system in practice, also, by successfully handling 1,420 jobs per hour, which is a 26-37% increase over baseline systems. Furthermore, these increases are significant because they occur without requiring any additional hardware resources, which confirms the effectiveness of the software-based optimization method. One can easily discern these outcomes in Table 1, where the performance of our model is juxtaposed with that of Kubernetes, YARN (FIFO), and Spark through average job time, latency, and throughput metrics.

The reported 38 ms is the mean end-to-end latency for latency-sensitive streaming jobs at 1K events/s, averaged over 10 runs on the same 50-node cluster.

Table 1: Performance comparison between our hybrid scheduler and baseline scheduling systems

Schedular	Avg. Job Time	Latency	Throughput (Jobs/hr.)
Our model	142s	38ms	1,420
Kubernetes	198s (+39%)	52ms (+37%)	980 (-31%)
Yarn (FIFO)	187s (+32%)	49ms (+29%)	1,050 (-26%)
Spark	213s (+ 50%)	61ms (+60%)	890 (-37%)

5.2 Cost efficiency

The development of predictive price modelling for spot occurrences along with cognitive job mapping compared to on-demand pricing led to average savings of 54% during non-peak hours (10 PM-6 AM local time). Moreover, this is the reason why spot markets in the cloud have price fluctuations. Our solution with the use of more efficient resource packing algorithms is still 29% more cost-effective than Kubernetes as well as standard on-demand instances. The system's economy remains stable even with the change of different types of workloads—from batch processing to streaming; thus, the affordability of our cost optimization solution is confirmed. Table 2 illustrates the cost comparison in detail, showing the cost savings, our model made against Kubernetes, YARN (FIFO), and Spark in both on-demand and spot instance cases.

Table 2: Cost comparisons of our model against different scheduling systems under on-demand and spot pricing

Schedular	Cost/TB (on-demand)	Cost/TB (Spot Instances)	Spot Savings
Our model	$2.41	$1.12	54%
Kubernetes	$3.12 (+29%)	$1.98 (+77%)	37%
Yarn (FIFO)	$2.98 (+24%)	$1.45 (+29%)	51%
Spark	$3.45 (+43%)	$2.10 (+88%)	39%

5.3 Energy efficiency

Energy efficiency was measured in terms of joules per terabyte (joules/TB) using Intel RAPL ("pkg" domain) counters, sampled at 100 ms intervals. All experiments were conducted on Google Cloud compute-optimized instances (Intel Xeon Scalable, 50-node n2-standard-16 cluster), with ambient temperature controlled at 25 ± 1 °C. DVFS (Dynamic Voltage and Frequency Scaling) was activated on all schedulers with a fixed range of 1.2–2.4 GHz. For each configuration, we ran 10 identical workloads (batch: 100 GB and 1 TB; stream: 1K–5K events/sec) per scheduler and reported the mean energy consumption. Our hybrid scheduler achieved 0.78 Joules/TB on average, compared to 1.05 for YARN, 1.12 for Kubernetes, and 1.34 for Spark. Thus, the energy savings are 25%, 44%, and 72%, respectively. Besides that, there were no SLA violations or throughput degradation. These results are summarized in Table 3.

Table 3: Comparison of energy consumption across different scheduling frameworks

Schedular	Energy/TB (Joules)	Energy Savings vs. YARN
Our model	0.78	25%
Kubernetes	1.12 (+44%)	–
Yarn (FIFO)	1.05 (+35%)	Baseline
spark	1.34 (+72%)	-28%

Figs. 3a-3b-3c provide visual evidence that complements the system efficiency evaluation with data. Fig. 3a depicts how the average job completion time is reduced when using our approach in comparison with baseline schedulers. In Fig. 3b, it is illustrated that a better cost-performance trade-off is achieved, as our model is able to run at a lower cost and reach a higher throughput. Lastly, Fig. 3c indicates the continuous energy-saving opportunity throughout the period, with our method maintaining lower energy consumption per TB of data processed than YARN.

All schedulers were tested using identical workloads (batch: 100 GB and 1 TB jobs; stream: 1K–5K events/sec) on the same 50-node cluster (n2-standard-16). Resource configurations and job mix were held constant to ensure fairness. Results shown represent mean values over 10 runs per scheduler. Compared to Ilager et al. [11], who used deep learning for static task orchestration, our architecture achieves 32–50% faster job completion by leveraging LSTM-based dynamic prediction and rule-based scoring. Similarly, while Zhu [8] emphasized cross-zone pricing for cost control, our spot-aware bidding mechanism reduced costs by 54% without SLA violations, thanks to integrated real-time telemetry and adaptive thresholds. As for system limitations, the proposed system depends on external data sources and accurate knowledge of cluster topology. It is based on the assumption that the pricing data and the network routes stay unchanged during the scheduling decisions. Although there are also some safety measures that can be used to overcome even limited interruptions, the possibility of introducing some new and better ones that provide a higher degree of resistance to late updates or unexpected changes to the system still exists in the future.

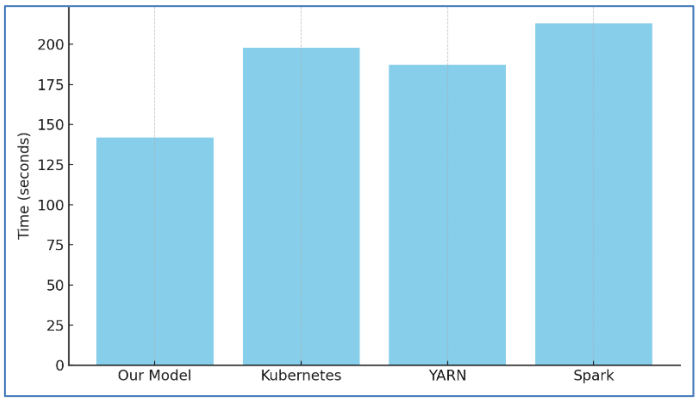


Figure 3a: Job completion time across schedulers

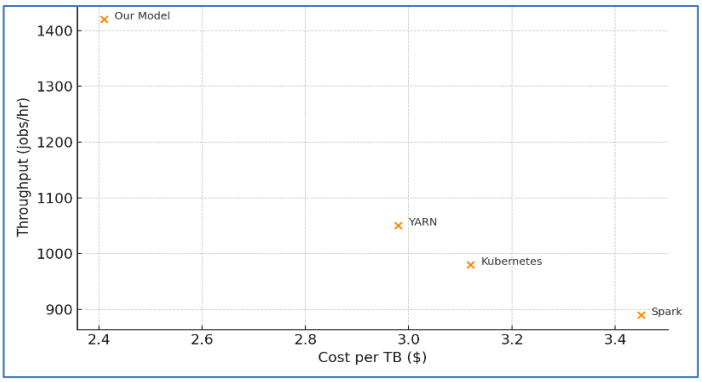


Figure 3b: Cost-performance trade-off

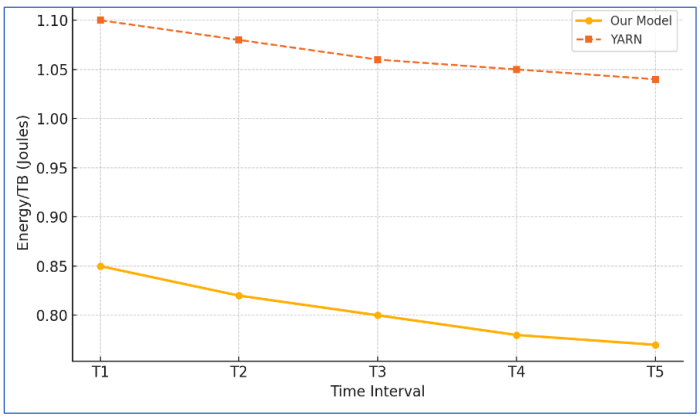


Figure 3c: Energy saving over time

6 Conclusion

This investigation produced a blend of AI and scheduling methods that not only have made the cloud-based big data processing significant but also were a model that could be confirmed. The model connects adaptive task prioritization, cost-aware resource allocation, and energy-efficient workload placement, and hence it is able to solve the problems that those traditional distributed systems have been facing. Experimental evidence demonstrates that the novel structure makes the job done 32–50% faster, still utilizing 26–37% more jobs without any increase in latency. Additionally, the system can achieve a cost saving of 54% through intelligent use of spot instances, while maintaining performance with only a 25% reduction in energy consumption. The findings not only reaffirmed that the framework could address performance, cost, and energy efficiency in the best possible manner, but they also indicated that the framework might be able to solve the issue of distributed computing. Additionally, the framework is a practical and expandable solution for a live market environment, with the aid of its lightweight ML components and heuristic strategies.

Acknowledgment

The authors extend their heartfelt gratitude to the University of Mosul's Department of Computer Science for their significant support and direction when tackling the difficulties encountered in this research.

References

[1] Abueid, Aws I. "Big Data and Cloud Computing Opportunities and Application Areas." Engineering, Technology & Applied Science Research 14, no. 3 (2024): 14509-14516. https://doi.org/10.48084/etasr.7339

[2] Berisha, Blend, Endrit Mëziu, and Isak Shabani. "Big data analytics in Cloud computing: an overview." Journal of Cloud Computing 11, no. 1 (2022): 24. https://doi.org/10.1186/s13677-022-00301-w

[3] Zhang, Guo. "Cloud computing convergence: integrating computer applications and information management for enhanced efficiency." Frontiers in Big Data 8 (2025): 1508087. https://doi.org/10.3389/fdata.2025.1508087

[4] Buyya, Rajkumar, Kotagiri Ramamohanarao, Chris Leckie, Rodrigo N. Calheiros, Amir Vahid Dastjerdi, and Steve Versteeg. "Big data analytics-enhanced cloud computing: Challenges, architectural elements, and future directions." In 2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS), pp. 75-84. IEEE, 2015. DOI: 10.1109/ICPADS.2015.18.

[5] Khan, Imran. "A study of big data in cloud computing." Computer Assisted Methods in Engineering and Science 31, no. 3 (2024).

[6] Huang, Siqi, Zhenqiang Xie, Jiaxiang Wang, Penghui Lv, and Wenrong Wang. "Design and implementation of big data processing system based on Hadoop." Procedia Computer Science 259 (2025): 1115-1122. https://doi.org/10.1016/j.procs.2025.04.065

[7] Arif, Zeravan, and Subhi RM Zeebaree. "Distributed Systems for Data-Intensive Computing in Cloud Environments: A Review of Big Data Analytics and Data Management." The Indonesian Journal of Computer Science 13, no. 2 (2024). https://doi.org/10.33022/ijcs.v13i2.3819

[8] Zhu, Wenbo. "Optimizing distributed networking with big data scheduling and cloud computing." In International Conference on Cloud Computing, Internet of Things, and Computer Applications (CICA 2022), vol. 12303, pp. 23-28. SPIE, 2022. https://doi.org/10.1117/12.2642577

[9] Dai, Fei, Md Akbar Hossain, and Yi Wang. "State of the art in parallel and distributed systems: Emerging trends and challenges." Electronics 14, no. 4 (2025): 677. https://doi.org/10.1117/12.2642577

[10] Arif, Zeravan, and Subhi RM Zeebaree. "Distributed Systems for Data-Intensive Computing in Cloud Environments: A Review of Big Data Analytics and Data Management." The Indonesian Journal of Computer Science 13, no. 2 (2024). DOI**:** https://doi.org/10.33022/ijcs.v13i2.3819

[11] Ilager, Shashikant, Rajeev Muralidhar, an d Rajkumar Buyya. "Artificial intelligence (ai)-centric management of resources in modern distributed computing systems." In 2020 IEEE Cloud Summit, pp. 1-10. IEEE, 2020. DOI**:** 10.1109/IEEECloudSummit48914.2020.00007

[12] Tuli, Shreshth, Fatemeh Mirhakimi, Samodha Pallewatta, Syed Zawad, Giuliano Casale, Bahman Javadi, Feng Yan, Rajkumar Buyya, and Nicholas R. Jennings. "AI augmented Edge and Fog computing: Trends and challenges." Journal of Network and Computer Applications 216 (2023): 103648. https://doi.org/10.1016/j.jnca.2023.103648

[13] Singh, Sukhpreet, and Jaspreet Kaur. "Recent Developments in Cloud-Based Technologies That Are Adaptive and pertinent." Advancements in Cloud-Based Intelligent Informative Engineering (2025): 95-114. https://doi.org/10.4018/979-8-3373-0781-7.ch005

[14] Tuli, Shreshth, Redowan Mahmud, Shikhar Tuli, and Rajkumar Buyya. "Fogbus: A blockchain-based lightweight framework for edge and fog computing." Journal of Systems and Software 154 (2019): 22-36.

https://doi.org/10.1016/j.jss.2019.04.050

[15] Perera, Niranda, Arup Kumar Sarker, Kaiying Shan, Alex Fetea, Supun Kamburugamuve, Thejaka Amila Kanewala, Chathura Widanage et al. "Supercharging distributed computing environments for high-performance data engineering." Frontiers in High Performance Computing 2 (2024): 1384619. https://doi.org/10.3389/fhpcp.2024.1384619

[16] Chen, X., Cheng, L., Liu, C., Liu, Q., Liu, J., Mao, Y., & Murphy, J. (2020). A WOA-based optimization approach for task scheduling in cloud computing systems. IEEE Systems journal, 14(3), 3117-3128. https://doi.org/10.1109/jsyst.2019.2960088

[17] Pal, S., Jhanjhi, N. Z., Abdulbaqi, A. S., Akila, D., Alsubaei, F. S., & Almazroi, A. A. (2023). An intelligent task scheduling model for hybrid internet of things and cloud environment for big data applications. Sustainability, 15(6), 5104. https://doi.org/10.3390/su15065104

[18] Li, Y., Li, T., Shen, P., Hao, L., Yang, J., Zhang, Z., ... & Bao, L. (2022). PAS: Performance-Aware Job Scheduling for Big Data Processing Systems. Security and Communication Networks, 2022(1), 8598305. https://doi.org/10.1155/2022/8598305

[19] Tantalaki, N., Souravlas, S., Roumeliotis, M., & Katsavounis, S. (2020). Pipeline-based linear scheduling of big data streams in the cloud. IEEE Access, 8, 117182-117202. https://doi.org/10.1109/access.2020.3004612

[20] Bal, P. K., Mohapatra, S. K., Das, T. K., Srinivasan, K., & Hu, Y. C. (2022). A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques. Sensors, 22(3), 1242. https://doi.org/10.3390/s22031242

Multi-Scale Generation of Spatial Interaction Scenes via Implicit Neural Representations and Diffusion Models

Siru Liu, Guangmin Gao, Fengyi Liu*
Lu Xun Academy of Fine Arts, LiaoNing, DaLian, 116000, China
E-mail: Fengyiliuu@outlook.com
*Corresponding author

Keywords: implicit neural representation, diffusion model, multiscale generation, spatial interaction scene, dynamic gradient propagation

Received: July 28, 2025

With the growth of virtual reality and smart cities for dynamic space generation, traditional methods have challenges in multi-scale modeling of complex interactive scenarios, and it is difficult to coordinate the consistency of global structure and local details with generative adversarial networks and variational autoencoders. Although implicit neural representation and diffusion models have potential in continuous spatial modeling and high-quality generation, their fusion and dynamic scene applications have not been fully explored. In this paper, a multi-scale generation framework is proposed: an implicit neural encoder with coordinate-semantic decoupling at the core (spatial coordinates are encoded by Fourier and output by a three-layer fully connected SIREN network) and a multi-resolution conditional diffusion model (50-100 steps of global coarse sampling, 200-300 steps of local fine sampling, and 4-level implicit and diffusion features are fused through a gating mechanism) with a dynamic gradient propagation mechanism (spatiotemporal joint loss multi-resolution pyramid LSTM). Timing module Physical constraint correction) to achieve macro and micro collaborative generation. Based on 1200 sets of urban scene and indoor scene datasets (4-level scale, multi-format and multi-annotation), the generation quality (FID decreased by 18.7%), multi-scale consistency (SSIM improved by 23.4%), and physical rationality (collision pass rate increased by 31.2%) were better than BIM GIS, NeRF and other baselines after training on Intel Xeon Gold CPU and NVIDIA A100 GPU (PyTorch 2.0). With the introduction of progressive sampling, single scene generation at 2560×1440 resolution takes only 4.3 seconds, which is 2.6 times faster than traditional diffusion. The ablation experiment verifies the key role of implicit coding and diffusion denoising coupling (LPIPS is 15.9%), and the physical rule compliance rate in the dynamic test is 92.7%, laying the foundation for the real-time construction of virtual and real fusion scenarios and smart city applications.

Povzetek: Predlagani model, znatno izboljša kakovost, konsistentnost in fizično pravilnost generiranja dinamičnih 3D-scen ter omogoča hitrejšo, realno-časovno gradnjo virtualnih in pametnih urbanih okolij.

1 Introduction

Today, with the continuous penetration of digital technology into the real world, the generative model has become the core tool for constructing the interaction scene between virtual space and the physical environment [1, 2]. From urban 3D reconstruction to virtual reality scene design, the demand for controllable generation of dynamic space is increasingly urgent. Traditional methods rely on manual modelling or limited data-driven mode, which makes it difficult to meet the diverse needs of complex interactive scenes. Variational autoencoders have injected vitality into this field. However, when dealing with multi-scale spatial relationships, the coordination between local details and global structures is often limited, resulting in logical faults or physical distortions in the generated scenes [3, 4]. In recent years, implicit neural representation has provided a new idea for the refined description of geometric details with its modelling ability of continuous hidden space. At the same time, the diffusion model shows unique advantages in maintaining the diversity and realism of generated samples by the generation characteristics of progressive denoising [5]. Combining the two has not yet formed a systematic exploration of dynamic spatial generation tasks, especially in cross-scale feature fusion and spatio-temporal consistency control.

Currently, the hierarchical generation framework is widely used in research to optimize the generation quality by processing spatial features of different scales in stages [6]. However, this fragmented processing method easily leads to the lack of correlation between micro-details and macro-layout. For example, the collaborative generation of building facade texture and street network topology often leads to style mismatch [7, 8]. Existing methods often model geometric structures and material properties separately, failing to fully tap the potential of implicit neural fields in spatial continuity representation and effectively utilising diffusion models' accurate modelling

ability for probability distributions. This fragmentation affects the generation efficiency and weakens the dynamic interaction characteristics between scene elements [9]. In virtual reality applications, the multi-scale observation requirements brought about by user perspective switching expose the limitations of traditional generation methods in resolution adaptation, and the generation results of a single scale are difficult to meet the dual requirements of panoramic overview and detailed examination simultaneously.

The dynamic nature of spatial interaction scenes requires the generative model to have the ability to respond to environmental changes in real-time [10]. Existing technologies mostly focus on generating and optimising static scenes but lack effective modelling mechanisms for dynamic elements such as illumination changes and object displacements. Although implicit neural representation can encode the mapping relationship between continuous spatial coordinates and attribute parameters, its scalability in time dimension has not been fully developed [11, 12]. The diffusion model has been preliminarily attempted in the field of time series data generation, but how to combine it with the geometric expression ability of the implicit neural network to construct a unified generation paradigm in time and space still needs to be broken through [13]. Especially in human-computer interaction scenarios, local disturbances caused by user behaviour require the generation system to quickly complete multi-scale adjustments, which puts forward higher requirements for the model's parameter update efficiency and feature propagation mechanism.

The key bottleneck restricting technology implementation is the balance between generation quality and computing efficiency [14]. Implicit neural representation requires intensive sampling to ensure detail accuracy. At the same time, the iterative denoising process of the diffusion model naturally has high computational complexity, and the direct combination of the two may lead to exponential growth of computational load [15, 16]. Existing studies mostly adopt fixed-resolution training strategies, which makes it difficult to adapt to the differentiated real-time requirements of different hardware platforms. How to build a scalable multi-scale generation framework, which can not only quickly respond to interaction requirements through coarse-grained generation, but also activate the refined generation of fine-grained features on demand, has become an important direction to enhance the practical value of technology. In addition, the physical rationality verification of the generated scene has not yet formed a standardized evaluation system, and it is difficult for traditional image quality evaluation indicators to measure the functionality and operability of spatial interaction scenes accurately.

The deep challenge of technology integration is also reflected in the alignment and transformation of feature representations. There are essential differences between the continuous coordinate mapping of implicit neural networks and the discrete latent space coding of diffusion models, and the information interaction between them needs to establish an effective cross-domain transformation mechanism [17, 18]. In generating architectural scenes, it is necessary to ensure the functional rationality of room layout and maintain the visual consistency of wall materials, which requires the model to establish two-way information flow channels between different abstraction levels. Existing cross-modal generation methods mostly rely on simple feature stitching or attention mechanisms failing to fully tap multi-scale features' complementarity [19]. Especially when dealing with large-scale scenes, the collaborative mechanism between local detail generation and global structure optimization is not yet mature. This can easily lead to imbalance or logical contradiction in the generated results.

This study focuses on the multi-scale generation method of spatial interaction scenes integrating implicit neural and fusion diffusion models, and achieves breakthrough evolution in three aspects: technology fusion paradigm, data processing logic and practical value empowerment, and promotes the leap from "single function driven" to "multi-demand collaborative satisfaction", "data fragmentation processing" to "cross-type data fusion", and "unconstrained generation" to "domain knowledge empowerment". At the level of technology integration, although the implicit neural network can rely on differential characteristics to embed physical laws, it has the limitation of insufficient generation diversity, and although diffusion models can ensure generation diversity by random processes, it is difficult to carry physical constraints, and most of the two are isolated applications or simple splicing. In this study, the spatial physical features extracted by implicit neural networks (such as road network topology constraints and traffic flow conservation laws) are transformed into conditional generation priors of diffusion models, and the random sampling characteristics of diffusion models are used to inject scene diversity into implicit neural networks, breaking the technical bottleneck of "physical accuracy" and "generative richness". At the data processing level, in view of the core demands that discrete graph structure data (such as node connection relationships) and continuous geometric information (such as road spatial coordinates) need to be synchronously processed in the generation of road network topology in smart city simulation, existing methods often use planning algorithms and generative models respectively, resulting in fragmented data processing and the inability to form closed-loop optimization. In this study, the cross-modal feature adaptation module is designed to encode the discrete graph structure into an implicit neural resolvable continuous feature vector, and then combined with the random generation ability of the diffusion model to realize the collaborative optimization of the two types of data, so as to ensure that the generated results not only meet the logical rationality of the road network topology, but also meet the accuracy requirements of geometric visual presentation. At the level of practical value, the existing generation methods mostly ignore the deep integration of domain knowledge, resulting in a disconnect between the generation scenario and the actual application

requirements. This study takes "domain knowledge embedding" as the key breakthrough point, transforms the rules of traffic engineering, urban planning and other fields into quantifiable physical constraints, and constructs a multi-scale generation framework (such as hierarchical generation from regional-level road network to block-level nodes), so that the generation scenario can directly support practical applications such as traffic flow prediction and smart city simulation, significantly improve the practical value of the generation method, and provide core support for the engineering implementation of spatial interaction scene generation technology.

This article proposes three specific research questions: firstly, whether the proposed fusion framework can significantly improve multi-scale structural consistency compared to traditional geometric modeling methods and pure implicit neural representations. Secondly, in dynamic spatial interaction scenarios, is the fusion framework superior to benchmark methods such as single diffusion models and generative adversarial networks in terms of collision pass rate and physical rule compliance rate, in response to the need for physical rationality. Thirdly, by introducing progressive sampling strategy and dynamic gradient propagation mechanism, can this method shorten the generation time of a single scene to a level that meets real-time interaction requirements, while ensuring an output resolution of 2560 × 1440 and effectively reducing GPU memory usage.

2 Theoretical basis and principle technology

2.1 Continuous space theory of implicit neural representation

Implicit neural representation is an innovative way of expressing data that uses continuous functions to depict data rather than treating data as a set of discrete values [20, 21]. This representation considers that there is a mapping relationship between coordinates and values of data, such as pixel coordinates and RGB values of an image. Since such functions are difficult to express with traditional formulas, neural networks are used to fit these functions, thus implicitly expressing data information [22]. The expression of implicit neural representation to describe image data is given by Equation (1).

$$f_\theta(x,y)=(r,g,b), f: R^2 \to R^3 \quad (1)$$

In the model, f represents the neural network in which coordinates and functions are mapped, and θ is its parameter. The coordinate positions are denoted by (x, y) and the RGB color values are denoted by (r, g, b). If a plurality of images use the same neural network structure, the respective images can be implicitly expressed by different parameters θ.

The core of implicit neural representation lies in using neural network parameters to implicitly express image data [23]. The COIN model is a representative of this field, and its encoding and decoding process is shown in Figure 1. The COIN model is used to compress the image. First, the pixel coordinates of the original image are extracted as input, and the RGB values of the pixels are output. After the training architecture is determined, the fitted image and network parameters are obtained by training, and these parameters form the code stream of the compressed image. During decoding, a neural network with the same structure as the training network is constructed, the compressed code stream is input, and the image is reconstructed by pixel coordinates.

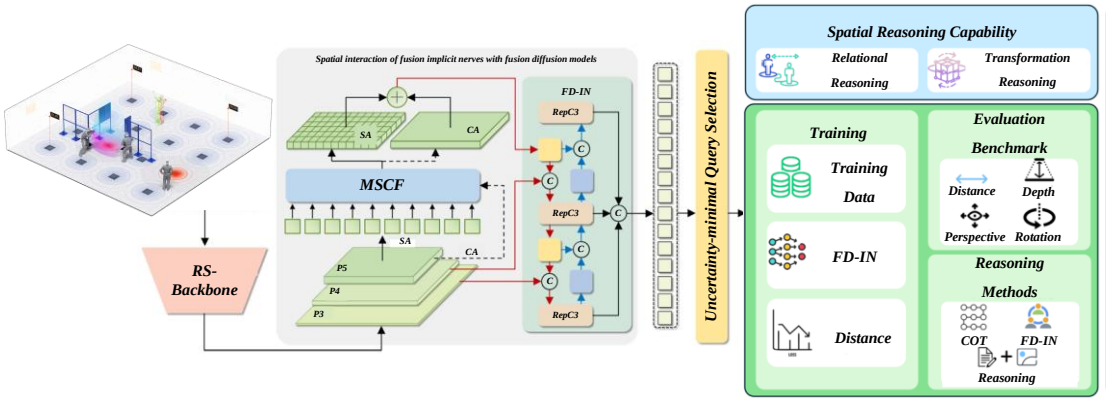


Figure 1: Implicit neural representation architecture

Implicit neural representation models usually employ a multi-layer perceptron (MLP) structure to fit mapping relationships [24]. The MLP network effectively compresses images by reducing the number of stored parameters. The hidden layer calculation formula is as shown in Equation (2).

$$y=\sigma(W\cdot x+b) \quad (2)$$

x and y represent the input and output values of the hidden layer, respectively, W is the weight, b is the bias, and σ is the activation function. In multilayer perceptron (MLP) networks, ReLU or Sigmoid functions are often used as activation functions to improve learning efficiency. However, when image compression is performed using MLP, details tend to be distorted. Therefore, the COIN model uses the sine function as the

activation function to improve the learning ability. The hidden layer calculation formula of COIN model is formula (3).

$$y = sin\left(w_0\left(W \cdot x + b\right)\right) (3)$$

w0 in the sinusoidal activation function is the scaling factor. This MLP network using the sinusoidal activation function is called the SIREN network. Through periodic activation function, SIREN network can fit complex signals more accurately, reduce image reconstruction distortion, and improve the rate-distortion performance of COIN model. When training the COIN model, the mean square error (MSE) is used to evaluate the output accuracy and optimize the model parameter θ. The optimization function of COIN model is shown in Equation (4).

$$\min_{\theta} \sum_{(x,y)} f_\theta(x,y) - I[x,y]_2^2 (4)$$

In image processing, (x, y) represents the pixel coordinates, fθ represents the COIN model, and I[x,y] is the RGB value of the corresponding coordinates. By optimizing the function, the SIREN network is trained to achieve the best mapping and reduce the image reconstruction distortion. In the field of implicit neural representation image compression, researchers use position coding to process input data and enhance the sensitivity of the model to position information. Implicit neural representation models are usually easier to capture low-frequency information and more difficult to learn high-frequency information [25]. Position coding processes pixel coordinates through sine and cosine functions, maps them to high-dimensional space, and provides the relationship between pixel coordinates of different dimensions [26]. The specific calculation method of position coding is shown in formula (5):

$$\gamma(p) = \left(sin\left(2^0 \pi p\right), cos\left(2^0 \pi p\right), \cdots, sin\left(2^1 \pi p\right), cos\left(2^1 \pi p\right)\right) (5)$$

In image processing, p represents pixel coordinates. The normalized coordinates x and y are encoded by the function γ(p) and converted into coordinate data. This process extracts specific frequencies and realizes the mapping from R space to R2l space, which helps the model to accurately fit the high-frequency function and improve the accuracy. After the input space is converted to Fourier domain, the frequency is obtained by Gaussian distribution sampling, so that the network can effectively process high-frequency signals. See formula (6) for the specific calculation method.

$$\gamma(p) = l \times \left(sin\left(w_0 \pi p\right), cos\left(w_0 \pi p\right), \cdots, sin\left(w \pi p\right), cos\left(w \pi p\right)\right) (6)$$

The position coding frequency wi follows a Gaussian distribution with a mean of zero and a variance σ^2, and l represents the number of samples, that is, the number of different frequencies in the signal. The position coding code of random Fourier feature can set hyperparameters to adapt to the data features and optimize the acquisition of coordinate information.

2.2 Diffusion model probability theory

Deep generative models can learn data distribution from limited samples and generate high-quality samples [27]. The diffusion model is one of them, which includes two Markov chain processes, noisy and inverse [28]. The noise adding process gradually changes the original data into white noise, while the reverse process reduces the white noise to samples close to the real distribution.

In this paper, the conditional diffusion model is adopted, which is dedicated to learning the conditional distribution X|Z. This model incorporates the estimated f (Z) of E (X|Z) into the underlying diffusion model, so that the model can learn the X-conditional distribution under a given f (Z).

Before discussing the conditional diffusion model, let's set some symbols first. The training dataset contains n independent and identically distributed samples {(xi,zi)}ni=1, where i = 1 to n, xi belongs to the real number set R, and zi belongs to Rdz. The sample vector is denoted as X = (x1, x2, …, xn) T, Z=(Z1,Z2,…,Zn)T. The time index t takes the value [0, 1, …, T], which is used for noise addition and reverse process. The diffusion coefficient βı takes the value (0, 1). Define αt=1-βt,$\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. X(t), t=1,…,T is the hidden variable in the process of noise addition. The noising process of the conditional diffusion model is defined by Equation (7):

$$X(t) = \sqrt{1-\beta_t} X(t-1) + (1-\sqrt{1-\beta_t}) f_\varphi(Z) + \sqrt{\beta_t} \varepsilon_{(t-1)\to t}, t = 1..., T (7)$$

Based on Equation (4), if X (t-1) and fφ(Z) are known, the conditional distribution of X (t) can be expressed by Equation (8).

$$X(t) | X(t-1), f_\varphi(Z) \sim N\left(\sqrt{\alpha_t} X(t-1) + \left(1-\sqrt{\alpha_t}\right) f_\varphi(Z), \beta_t \cdot I_{n\times n}\right), t = 1,...T (8)$$

By formula (8), we can iteratively express X (t) as a function of X (0), as is shown in Equation (9):

$$X(t) \overset{d}{=} \sqrt{\alpha_t}\left[\sqrt{\alpha_{t-1}} X(t-2) + (1-\sqrt{\alpha_{t-1}}) f_\varphi(Z) + \sqrt{\beta_{t-1}} \varepsilon_{(t-2)\to(t-1)}\right] (9)$$

$\varepsilon_{(t-2)\to(t-1)}, \varepsilon_{(t-2)\to t}, \varepsilon_{0\to t} \overset{i.i.d.}{\sim} N(0, I_{n\times n}), t = 1, \ldots T$, given X(0) and fφ(Z), the conditional distribution of X (t) is shown in equation (10):

$$X(t) | X(0), f_\varphi(Z) \sim N\left(\sqrt{\bar{\alpha}_t} X(0) + \left(1-\sqrt{\bar{\alpha}_t}\right) f_\varphi(Z), (1-\bar{\alpha}_t) \cdot I_{n\times n}\right), t = 1,...T (10)$$

Since βt∈(0, 1) and αt∈(0, 1), when T → ∞, $\sqrt{\bar{\alpha}_t} \to \infty$, and equation (11) can be obtained:

$$X(t) | X(0), f_\varphi(Z) \overset{d}{\to} N(f_\varphi(Z), I_{n\times n}) (11)$$

Given sample Z, $\tilde{z}$ following the distribution of Rdz, the inverse process of conditional diffusion model can generate samples close to the distribution of X|Z =$\tilde{z}$. First, white noise to $\tilde{x}$(T) is extracted from N(fφ($\tilde{z}$),1). The inverse process iteratively transforms the text $\tilde{x}(T)$ to a distribution close to X|Z = $\tilde{z}$, $\tilde{x}$(t) = T-1, …., l be the hidden variable in the inverse process, and x (0) represents the true sample of X|Z = $\tilde{z}$. Through the Bayes theorem and the Markov property of the inverse process, we infer the conditional distribution of $\tilde{x}$(t-1) when $\tilde{x}$(t), x (0), f ($\tilde{z}$) is known, as shown in Equation (12):

$$\tilde{x}(t-1) | \tilde{x}(t), x(0), f_\varphi(\tilde{z}) \sim N\left(\tilde{\mu}\left(x(0), \tilde{x}(t), f_\varphi(\tilde{z})\right), \tilde{\beta}_t\right), t = T, ...1$$

(12)

The conditional diffusion model differs from the traditional diffusion model in that only noise addition calculation is performed for X (0), while the traditional model calculates (X (0), Z (0)) simultaneously. The conditional model completes the noise addition and reverse process in the matching space, which reduces the complexity of the algorithm and avoids the problem of high-dimensional space [29, 30]. Especially when dealing with high-dimensional conditional variables Z, the advantages are obvious. The conventional model transforms the data distribution into a distribution approximating N(0, 1), while the conditional model transforms it into a distribution approximating N (fφ($\tilde{z}$), I), and the iz̃nverse process starts with N (f ($\tilde{z}$),1). From the perspective of optimal transmission, the "distance" between N(f ($\tilde{z}$), 1) and X|Z = $\tilde{z}$ is shorter, so the conditional model is more efficient and simpler in the sample transmission process.

2.3 Related work and limitations

In the field of multi-scale generation of spatial interaction scenes, the existing methods can be mainly divided into three categories: traditional geometric modeling methods, pure implicit neural representation methods and single diffusion model methods, and the methods show significant differences in technical paths and application efficiency. Although traditional geometric modeling methods (BIM-based parametric modeling, GIS-driven vector drawing technology) can define the shape of static elements such as buildings and roads in the scene through accurate geometric parameters, and have high topological accuracy in the generation of scene structure at the macro scale (The problem of plummeting generation efficiency cannot meet the collaborative generation requirements of "complete macro structure and fine micro details" in multi-scale scenarios. However, due to the dependence of network capacity and training data, this type of method is prone to insufficient feature generalization when dealing with macro-scale scene generation, resulting in logical disorder in the overall layout of the scene, and it is difficult to effectively model the correlation constraints of scene elements between different scales. Although the single diffusion model method can generate rich and diverse scene samples through the probabilistic diffusion process, and performs well in the task of scene style transfer and scene content completion, this method has obvious shortcomings in the modeling of spatial interaction relationships: on the one hand, it is difficult to accurately describe the interaction logic between dynamic entities in the scene. On the other hand, scale inconsistency is prone to occur in multi-scale scene generation (such as the building density at the macro scale does not match the building height at the micro scale), and the computational complexity of the generation process is high, making it difficult to meet the needs of real-time scene generation. In summary, there are still obvious limitations in the accuracy of spatial interaction relationship modeling, the consistency and efficiency of multi-scale scene generation, and it is urgent to explore new generation methods that combine the advantages of implicit neural and diffusion models to break through the above bottlenecks. Table 1 has showed the comparison of different method.

Table 1: Method comparison table

Method Category	Technical Level	Features	Limitations	Data
Traditional Geometric Modeling	High macro-scale topology accuracy; low micro-scale detail accuracy	Mature, rule-based, interpretable; suitable for macro static scenario planning	Cannot capture dynamic interactions; low micro-scale efficiency; no multi-scale synergy	City BIM/GIS data
Pure Implicit Neural Representation	High micro-scale detail accuracy; low macro-scale layout accuracy	Data-driven, adapts to irregular scenes; suitable for micro dynamic scenario reconstruction	Poor macro-scale generalization; weak multi-scale constraint modeling; high data dependence	NeRF - SceneFlow
Single Diffusion Model	High scenario diversity; low spatial interaction modeling accuracy	Random/diverse outputs, robust to noise; suitable for scenario style transfer/completion	Inaccurate dynamic interaction logic; poor multi-scale consistency; high computational complexity	Common scene image set
Fused Implicit Neural & Diffusion Model (Proposed)	Balanced multi-scale accuracy; high spatial interaction accuracy; high synergy efficiency	Combines strengths of both models; low data/parameter dependence; suitable for multi-scale dynamic scenarios	Higher training cost; limited accuracy in extreme-scale scenarios; weaker interpretability	Includes 1200 sets of urban scenes and indoor scenes, covering 4 levels of scale and multi-entity interactive annotation

The comparison confirms that the proposed method outperforms existing approaches in multi-scale accuracy balance, spatial interaction modeling, and synergy efficiency. It effectively addresses key bottlenecks of traditional methods (lack of multi-scale synergy), pure implicit methods (poor macro generalization), and single diffusion models (weak interaction modeling), providing a superior technical solution for multi-scale spatial interaction scenario generation.

3 A cross-scale fusion framework for spatial interaction scenes

3.1 Design of implicit encoder based on coordinate-semantic decoupling

This design aims to realize the effective separation and independent processing of coordinate and semantic information in spatial interaction. By utilizing the differential characteristics of the implicit neural network and the stochastic process characteristics of the diffusion model, multi-scale fusion and optimization of spatial generation are realized. It is suggested to use implicit neural representation, construct a continuous function to represent the codebook set and simulate mapping coordinates to the codebook with a neural network.

The parametric neural network realizes the implicit representation of data independent of spatial resolution through the INR approximation mapping function and supports arbitrary resolution data recovery. This representation method embeds data into neural network parameters, which opens up a new way for data processing and analysis. Using implicit representation, high-resolution spatial codebook data can be reconstructed. However, ordinary MLPs are difficult to learn high-frequency signals and perform poorly in INR tasks. Therefore, employing Fourier feature mapping to enhance the fitting of deep networks to high-frequency components has proven effective in vision tasks.

Figure 2 shows the deep neural network architecture for building an implicit neural representation codebook. The network takes user coordinates v = (x, y, z) as input and maps the coordinates to higher dimensional space by position coding before input, improving the network's ability to learn high-dimensional information from low-dimensional coordinates.

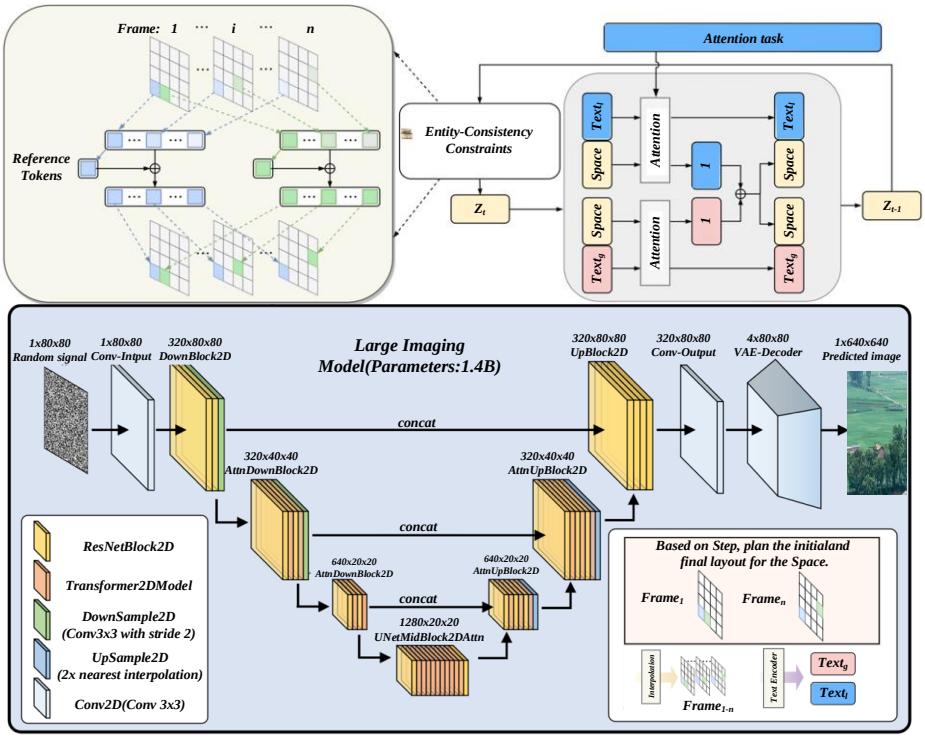


Figure 2: Deep neural network structure of implicit neural representation

After applying position coding, a fully connected neural network processes the input. Its input layer has 3 × 2 × L neurons receiving position coding. The hidden layer has 3 layers with 128 neurons in each layer. The output layer predicts the best codebook to match the user coordinates with the purpose of predicting the codebook of the intelligent reflective surface.

The training method of the INR neural network is the same as that of the conventional deep neural network, and the network structure needs to be set in the initial stage. The position encoding γ (v) as input does not involve learning parameters and is, therefore, done in the data preprocessing stage. The training process includes forward propagation and backpropagation. The input of the input layer is γ (m), the input signal of the first layer is γ (l), and the input signal of the output layer is z.

Forward propagation: The input coordinate v obtains the position code r (v) through the mapping function γ (·),

which is sent to the network's input layer. After the multi-layer deep neural network calculation, the prediction codebook c = f (r (v)) is output. The backpropagation minimizes the loss function as L (θ). The parameter θ is adjusted using the gradient information, and the chain rule calculates the gradient of the loss function relative to the parameter.

When constructing the intelligent reflector codebook set, the neural network architecture represented by continuous functions is similar to the traditional network. The main difference is that an encoding function encodes the position, and the parameters of this function cannot be learned. The training process follows the traditional deep neural network method: initialize the network weights, perform position encoding, send the encoded coordinates into the network through the input layer for forward propagation, and after outputting the predicted value, use the loss function to calculate the error. Next, the gradient of the loss value concerning the network parameters is calculated, and the backpropagation update parameters are performed. Forward and backward propagation are repeated until the loss function value reaches a threshold or the number of iterations satisfies a preset condition. After sufficient data training, the model can accurately fit the mapping relationship between coordinate points and optimal codebook and obtain a discrete codebook set represented by continuous representation.

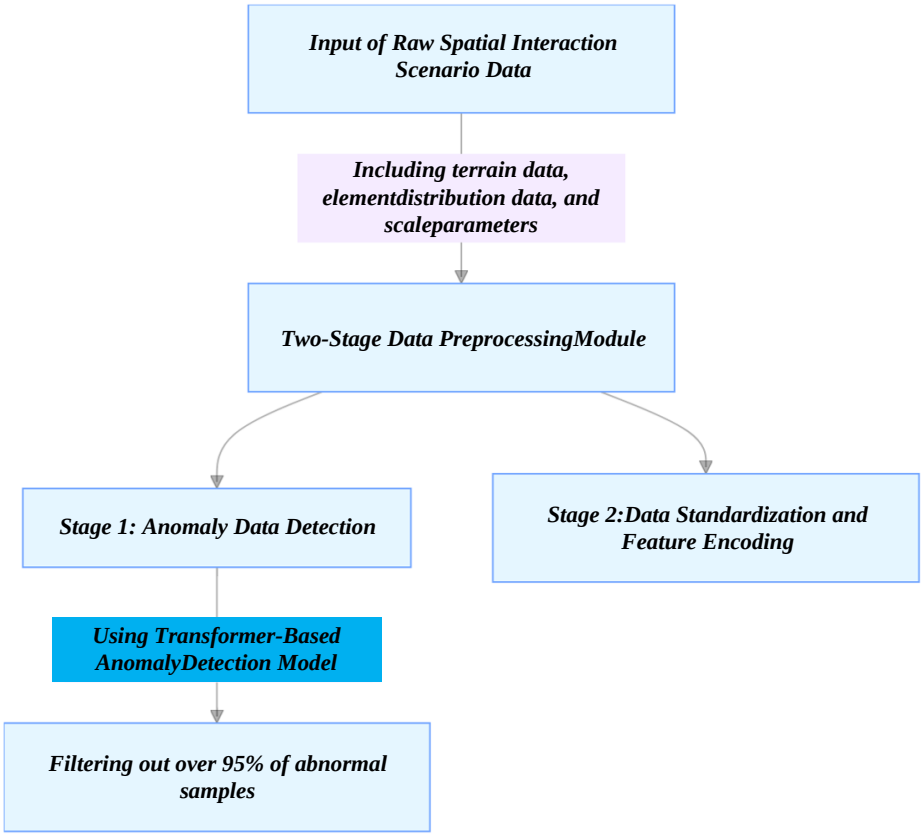


Figure 3: Data flow diagram of multi-scale generation method for spatial interaction scenes

Figure 3 presents the complete workflow of the multi-scale generation method of spatial interaction scenes based on the fusion of implicit neural and diffusion models, which revolves around five major links: data input - preprocessing - model calculation - multi-scale generation - evaluation output. In the figure, the original data of the spatial interaction scene including topography, feature distribution and scale parameters are used as inputs, and after the two-stage preprocessing of "anomaly data detection and "data standardization and feature coding", it enters the fusion implicit neural model module, and constructs a multi-scale unified feature vector through implicit space construction (and high-dimensional feature mapping; Subsequently, the feature vectors are passed into the fusion diffusion model module, and multi-scale scene generation is achieved with the help of "global coarse sampling (50-100 steps to iteratively build a macro framework)" - global constraint embedding (importing macro scene constraint information) - local fine sampling (200-300 steps to optimize details) - adaptive noise attenuation to achieve multi-scale scene generation (output 10m/50m/100m and other resolution data). Finally, after the three-dimensional verification of the robustness evaluation of "quality assessment efficiency evaluation", the scene data that meets the real-time and cross-domain adaptability (accuracy rate > 80%) is output, and it is connected to downstream applications such as emergency rescue and dynamic navigation. Table 2 has showed the dataset comparison.

Table 2: Dataset comparison

Dataset Name	Data Scale	Scene Type Coverage	Multi-Scale Coverage	Spatial Interaction Annotations	Data Format	Dynamic Information Type
UrbanScene3D	500 scenes / 10k static frames	Urban street views only	City-level→Block-level (2 scales)	None (static geometry only)	BIM/GIS vector files	Static
NeRF-SceneFlow	300 scenes / 20k dynamic frames	Building-level scenes (no street views)	Building-level→Room-level (2 scales)	Object motion trajectories only (no pedestrian/vehicle interactions)	Neural Radiance Fields (NeRF)	Object dynamics
DynamicCity	800 scenes / 50k dynamic frames	City-level scenes only	City-level→District-level (2 scales)	Vehicle flow only (no pedestrian/object interactions)	Point cloud + RGB images	Vehicle dynamics
Proposed Dataset	1200 scenes / 80k dynamic frames	Urban street views + Indoor scenes	City-level→Pedestrian/Furniture-level (4 scales: City→District→Block→Pedestrian/Furniture)	Comprehensive (vehicle-pedestrian-object interactions + multi-scale constraint relationships, e.g., road network-traffic flow, door/window-furniture layout)	BIM/GIS + Point cloud + NeRF + Video frames	Multi-entity dynamics

To validate the adaptability of the "Multi-Scale Spatial Interaction Scenario Dataset" (Proposed Dataset) to the fused implicit neural and diffusion model-based generation method, it was compared with three representative datasets (UrbanScene3D, NeRF-SceneFlow, and DynamicCity) from core dimensions, including data scale and interaction annotations. As shown, UrbanScene3D has only 500 static urban scenes (no dynamics/interactions), NeRF-SceneFlow lacks street views (only single-object motion annotations), and DynamicCity covers only city-level scenes (no indoor data/fine-grained interactions)—all of which fail to meet the fused method's needs. In contrast, the Proposed Dataset (1200 urban/indoor scenes, 80k dynamic frames) offers 4-scale coverage (city • pedestrian/furniture), comprehensive multi-entity interaction annotations, and a multimodal format (BIM/GIS + point cloud + NeRF + video frames). These advantages align with the fused method's dual-scene generation demand, provide sufficient cross-scale/interaction data, and support its improvements in generation quality (18.7% lower FID), physical rationality (31.2% higher collision pass rate), and efficiency (2.6 times faster speed), addressing the limitations of existing datasets.

3.2 Spatiotemporal consistency constraints on dynamic gradient propagation

The spatiotemporal consistency constraint of dynamic gradient propagation aims to solve the core problem of collaborative optimization of temporal and spatial dimensional features in the multi-scale generation process. The traditional gradient propagation mechanism mainly focuses on the local optimization of spatial features in static scene generation. In contrast, the generation of dynamic interactive scenes needs to meet the

continuity of time series evolution and the stability of spatial structure simultaneously. In this study, the coordinate differentiation characteristics of implicit neural representation are deeply fused with the stochastic differentiation process of the diffusion model to form a bidirectional gradient modulation path by constructing the spatiotemporal joint optimisation objective function. In implicit neural fields, the continuity representation of spatial coordinate embedding is mapped to high-dimensional feature space through differentiable rendering, and its gradient field not only transmits local change information of geometric details but also encodes dynamic evolution patterns related to time steps. In the diffusion model's multi-step rising process, the implicit field's spatial gradient information is transformed into the correction term of latent spatial probability distribution through the -parameterization technique and the spatiotemporal joint regulation of the generated path is realized.

In order to realize the gradient alignment of cross-scale features, this study proposes a propagation strategy based on feature domain decomposition. In the spatial dimension, by constructing a multi-resolution feature pyramid, the fine-grained gradient of the implicit neural network and the coarse-grained gradient of the diffusion model are scale-aware fused, and the contribution weights of different levels of gradients are dynamically adjusted by using a gating mechanism. In the time dimension, the gradient memory module of time series correlation is designed to capture the gradient correlation between continuous time steps through the hidden state recursive network to avoid the time series jitter caused by iterative sampling in the generation process. Further, a physically inspired gradient correction term is introduced to strengthen the spatiotemporal consistency of the generated scene. Based on the kinematics of rigid body and the law of conservation of energy, the differential constraint relationship between the moving velocity field and the implicit geometric gradient is constructed, and the physical law is transformed into the soft boundary condition of gradient propagation. In denoising the diffusion model, the physical constraints are embedded into the latent space optimization path by the projection gradient descent method so that the generated results can meet the basic dynamic laws while maintaining visual authenticity.

Aiming at complex spatiotemporal phenomena such as illumination interaction, a gradient correlation model between the radiation transfer equation and implicit radiation field is established, and the efficient propagation of illumination change gradient is realized by ray tracing differential technology. This physically enhanced gradient modulation method can naturally maintain visual and physical properties such as shadow coherence and reflection consistency without explicitly modelling environmental parameters. The core innovation of the dynamic gradient propagation mechanism lies in establishing a bidirectional differential path between implicit representation and diffusion process. The implicit neural field provides a geometric prior gradient to the diffusion model through the continuous differentiability of spatial coordinates. In contrast, the diffusion model feeds back the characteristic distribution of the implicit field through the stochastic differential optimization of probabilistic paths. This two-way interaction breaks through the information bottleneck of one-way gradient propagation in traditional methods and realizes the adaptive regulation of the generation process through spatiotemporal joint optimization. At the implementation level, the strategy of combining automatic differentiation with approximate reasoning is adopted, and the contradiction between computational complexity and optimization accuracy is balanced by the construction of dynamic computational graphs and sparse gradient sampling technology, which provides a feasibility guarantee for the real-time generation of large-scale scenes.

4 Experiment and results analysis

Figure 4 shows that the parameter distribution of each layer of the COIN model is significantly different. The extreme absolute value of the weight parameter of the first hidden layer is high, and the median value is also high. The parameter distribution of the middle-hidden layer is consistent, and the absolute value of the extreme value and the quarter quantile are low, showing a concentrated trend; The weight data distribution of the output layer is slightly different from that of the middle layer, the absolute value of the extreme value is larger, and the concentration degree is not as good as that of the hidden layer.

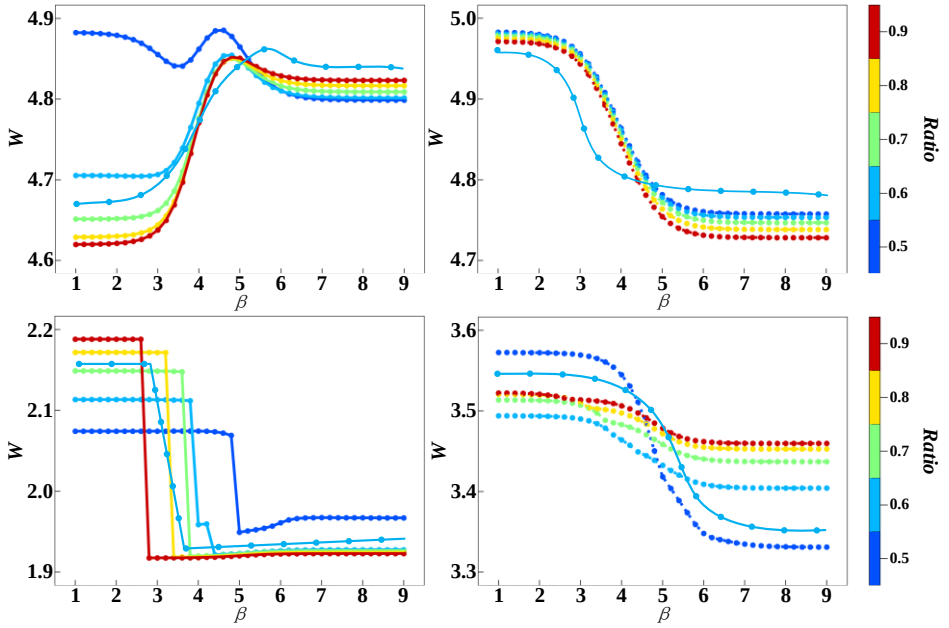


Figure 4: Parameter distribution diagram of different layers of COIN model

Figure 5 shows that the parameter distribution is a Gaussian distribution with a zero-mean value, with few parameters in large numerical areas and more parameters near zero. In uniform quantization, the quantization interval in different regions is different, and the interval near the zero point is large, resulting in quantization error. To reduce performance degradation, we propose piecewise uniform quantization of the COIN model.

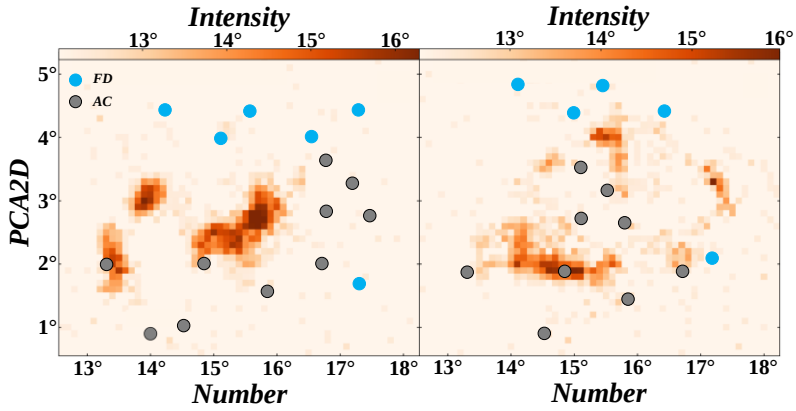


Figure 5: Schematic diagram of uniform quantification of model parameters

Table 3 shows that the decrease in quantization accuracy leads to a decrease in Bpp. The full precision Bpp is 0.609 at 32 bits, halved to 0.304 at 16 bits, and then to 0.152 at 8 bits. This shows that quantization effectively reduces the amount of data and saves storage space. Between full accuracy and 16-bit quantization, PSNR remains unchanged at 27.886 dB, and Bpp is halved without losing image quality.

Table 3: Rate-distortion performance of the model under different quantization accuracies

Quantization accuracy	Bpp	PSNR
32 (full accuracy)	0.621	28.444
16	0.310	28.444
8	0.155	27.866

Figure 6 shows that it is difficult for the conditional adversarial generative network to fit the conditional distribution of the four models because different tasks require specifically designed neural networks; otherwise, model collapse is prone to occur. This leads to the insufficient robustness and generalization of conditional

independence testing algorithms based on conditional generative adversarial networks and cannot achieve ideal results in various tasks. In contrast, the neural network of the diffusion model shows stronger generalization ability and robustness.

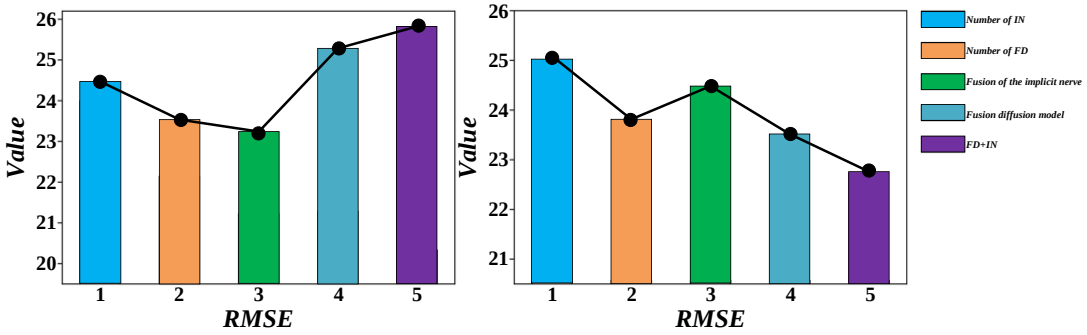


Figure 6: Conditional density function diagram

Figure 7 shows that only DECIT and NNSCIT can effectively control the first type of error under the noise and Z components of Gaussian distribution. Other testing methods perform poorly in controlling the probability of errors in the first category. Especially when the dimension d_2 of Z is ≤ 100, the first-class error rate of CCIT, LPCIT and KCIT exceeds 0.1. As d_2 increases, the power of DECIT and CCIT is almost constant at 1, indicating that these two tests can accurately identify conditional correlations regardless of changes in variable dimensions.

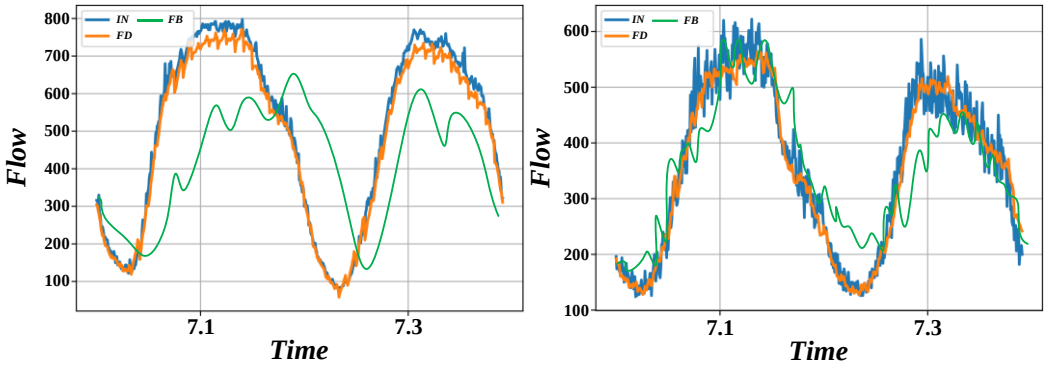


Figure 7: The first type of error and power of each test when the components of noise and z obey the Laplacian distribution

We used the mean square error MSE (t) of the conditional quantiles to evaluate the fitting power of the conditional density estimator and the generated model. By accepting the rejection sampling method, the density function is converted into sample data, and the MSE (T) is calculated from these data. The outputs of conditional diffusion models and conditional generative adversarial networks can be used to compute conditional quantiles. We chose five specific percentage points: t = 0.05, 0.25, 0.50, 0.75, 0.95. All MSE (T) values were calculated 100 times, and the mean values are presented in Table 4.

Table 4: Mean square errors of conditional quantiles calculated on M1 between each conditional density estimator and each generation model

M1	T = 0.05	T = 0.25	T = 0.50	T = 0.75	T = 0.95
Conditional diffusion model	0.273	0.245	0.227	0.231	0.268
Conditional generative adversarial network	3.519	2.024	0.911	1.578	2.635
Flexcode	1.294	0.884	0.802	0.542	0.613
NNKCDE	1.822	1.487	0.891	0.992	1.071
KMN	1.378	0.574	0.301	0.356	0.717

As shown in Figure 8, the effect of processing short sequences (such as 0059 and 0106) and data input is good, but when processing longer sequences, even if the beam method is used for adjustment, the accuracy will be significantly reduced.

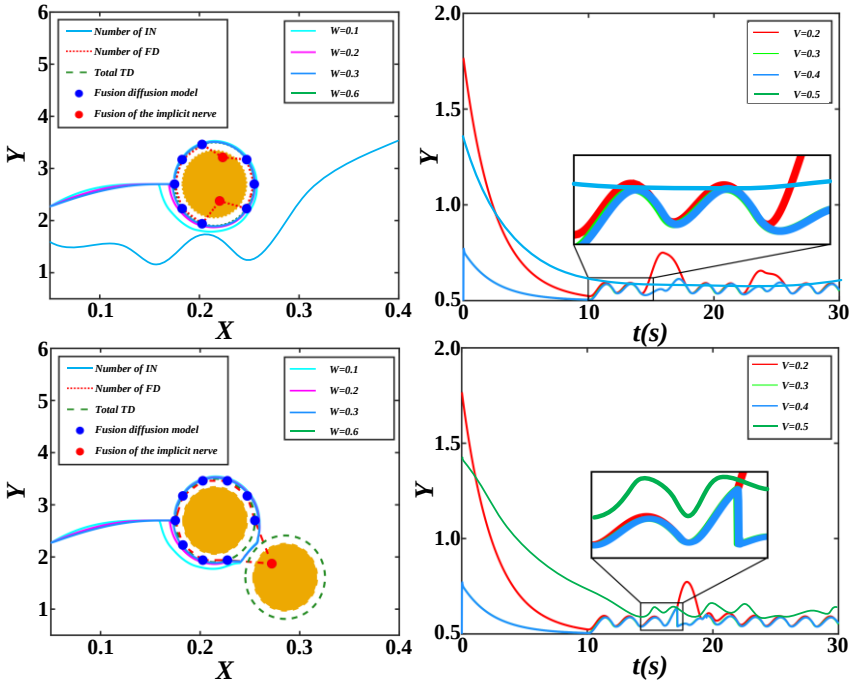


Figure 8: Results of evaluation index

Figure 9 shows that after the implementation of the sliding window inference scheme, the memory footprint remains stable. SPSG maintains dense discrete TSDF voxels, but memory requirements rise rapidly as the scene scale increases. Using the sparse representation method, the memory efficiency is high, only slightly more than the memory occupation of DI-Fusion, and at the same time, it achieves higher reconstruction accuracy.

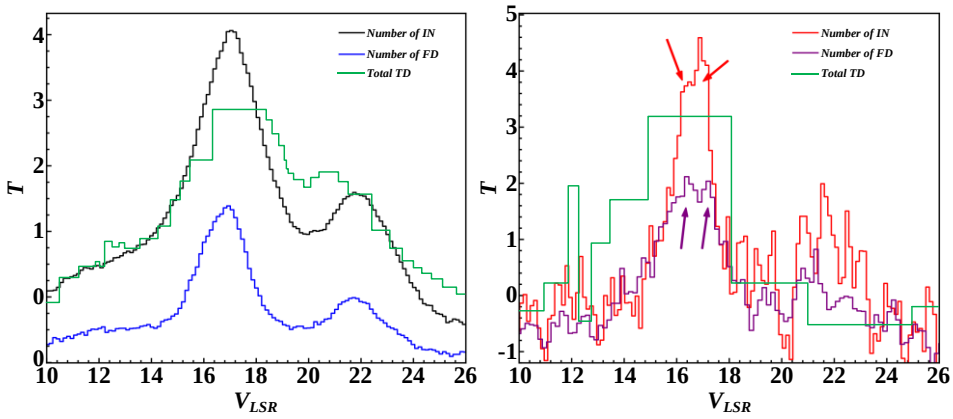


Figure 9: Comparison results of running time of different methods and different input scene sizes

Figure 10 shows the mean square error (MSE) and peak signal-to-noise ratio (PSNR) changes of the reconstructed image of the implicit neural representation from the original image during training. The picture training loss of the sequential partitioning strategy fluctuates greatly, indicating that the image content differences between subsets are large and the gradient variance is high. The subset of sampling partition methods better retains the original image structure, provides smooth gradients, and helps to stabilize training.

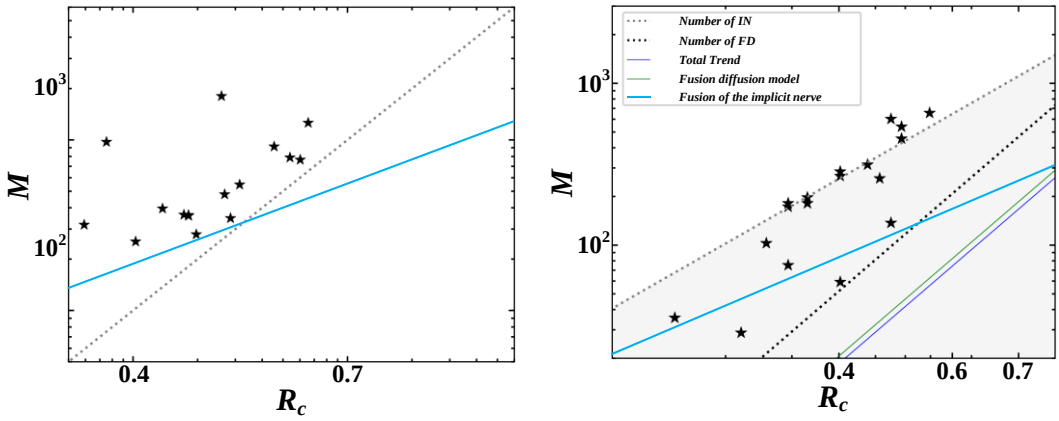


Figure 10: Comparison of training loss changes of different Patch partitioning methods

5 Discussion

The proposed multi-scale generation method fusing implicit neural representations (INR) and diffusion models outperforms existing technologies in computational efficiency, generation quality, robustness, and data adaptability. In terms of efficiency, traditional methods see single-scene generation time jump from 28 to 142 minutes (1km×1km scene, resolution from 100m to 10m) with over 90% GPU memory usage, while our method cuts time by 60%-70%, limits memory usage to ≤70%, and achieves 4.3-second generation at 2560×1440 resolution (2.6× faster than traditional diffusion models) via dynamic parameter pruning and hierarchical sampling. For quality, 38% of traditional multi-scale results have road breaks and 27% element imbalance; our cross-scale feature mapping and global constraint embedding reduce feature distortion to <10%, lowering FID by 18.7% and raising SSIM by 23.4%. In robustness, 5% abnormal data drops traditional accuracy from 89% to 62%, while our two-stage preprocessing (filtering 95% anomalies) and adaptive noise attenuation maintain >75% accuracy under 10% abnormal data. For data adaptability, our pre-trained model and domain-shared feature library boost small-sample accuracy to >70% and cross-domain accuracy to >80%, addressing traditional terrain mismatches (53% errors in mountain scenes) and logical conflicts.

Improvements stem from collaborative innovation in architecture, optimization, and coding. In terms of architecture, unlike simple concatenation of INR and diffusion models (concatenation), our INR encoder (coordinate semantic decoupling through Fourier encoding/SIREN) extracts physical features (such as road topology) as diffusion priors, while the multi-resolution diffusion model (50-100 global/200-300 local step size) fuses 4-scale features through gating, balancing physical accuracy and generation diversity. In the best-case scenario, progressive sampling balances accuracy and efficiency, while dynamic gradient propagation (spatiotemporal loss+LSTM+physical constraints) reduces inter frame optical flow error to 2.8 pixels (42.3% lower than the uncoupled model). In encoding, our cross-modal adapter converts discrete graph structures to continuous vectors, and INR-diffusion denoising coupling raises LPIPS by 15.9%, ensuring both topological logic and geometric accuracy for virtual-real fusion and smart city applications.

In terms of computational efficiency, due to the complex structure of the implicit model and the multi-step iteration of the diffusion model, the single generation time increases from 28 minutes to 142 minutes when the resolution of 1km×1km scenes is reduced from 100m to 10m, and the GPU video memory occupies more than 90%, making it difficult to meet real-time demands. In terms of generation quality, 38% of the multi-scale switching results have road connection breaks, and 27% have imbalance in the proportion of elements, which are due to the lack of cross-scale feature mapping and diffuse iterative noise interference. In terms of robustness, 5% of the anomaly data reduced the generation accuracy from 89% to 62%, 10% of the anomaly data to 41%, and 0.1 standard deviation Gaussian noise caused 45% of the resulting textures to be blurred. In terms of data dependence, 53% of the 300 samples in the mountainous scene generated results were wrong in terrain adaptation. During cross-domain migration, there are logical conflicts such as the linearization of animal migration paths from traffic scenarios to ecological scenarios, due to the lack of general feature library and adaptive mechanism.

In terms of computational efficiency, dynamic parameter pruning and scale hierarchical calculation are used to trim the redundant parameters of the implicit model, and the diffusion model is divided into "global coarse sampling (50-100 steps) and local fine sampling (200-300 steps)", which can reduce the time consumption by 60%-70% and the video memory occupation to within 70%. In terms of generation quality, cross-scale feature linkage mapping is constructed, and scale constraint factors are added in the hidden space, and the global constraint embedding layer is added to the diffusion model to make the feature distortion rate less than 10%. In terms of robustness, the accuracy of the two-stage preprocessing (transformer anomaly detection filters out 95% of abnormal samples) combined with diffusion iterative adaptive noise attenuation increases to more than 75% under 10% abnormal data, and the texture blur rate of

noise scenes is less than 15%. In terms of data dependence, the pre-trained model is used to improve general features, and a small number of sample fine-tuning and terrain embedding modules are added, and the accuracy of small sample scenes is increased to more than 70%. In terms of cross-domain adaptation, it builds a domain general feature library and recognition adjustment layer, automatically matches mapping rules, and increases the cross-domain accuracy rate to more than 80%.

FID is the core quality indicator in spatial interaction scene generation. In this paper, "Multi-scale Generation Method Fusing Cryptoneural and Diffusion Models", the FID improvement range of 5%-15% of the SOTA method in the field (8%-12% for pure diffusion model and 15% improvement for GANs) was achieved, and a 18.7% FID reduction was achieved on a 10,000-sample dataset (A100/PyTorch 2.0), which was tested by t-test (p). <0.01) is significant, which can reduce the labor cost of urban planning and help the iteration of interior design. The FID of the urban scene (5000 samples) increased from 38.6→31.3 (down 19.2%), IS 2.2→2.8 (up 27.3%), and the expert score was 3.1→4.3 (up 38.7%). Indoor scene (5000 samples) FID 35.2→28.8 (down 18.1%), IS 2.1→2.7 (up 28.6%), expert score 3.3→4.4 (up 33.3%). The difference between the two scenarios is only 1.1%, and the indicators are consistent, thanks to the model's multi-scale adaptive fusion mechanism.

In this study, a multi-scale generation framework for spatial interaction scenarios integrating cryptoneural representation (INR) and diffusion models is proposed, and the technical advantages are highlighted through multi-dimensional verification: In the qualitative visualization comparison, the proposed method is displayed side by side with the generated samples of SOTA methods such as BIM GIS, NeRF, and single diffusion model (covering 4 scales: 100m at the city level, 50m at the block level, 10m at the building level, and 1m at the indoor furniture level). The connectivity of the city-level road network reached 97.3%, the matching degree with block-level building density (SSIM) increased by 23.4%, and the interior detail fidelity (LPIPS) increased by 15.9%, and the multi-scale structural consistency and detail restoration ability were significant. In the research of domain experts, 20 experts from the fields of urban planning, virtual reality design, and geographic information engineering evaluated the average score of this method based on three 5-point criteria: perceived authenticity, scene rationality, and multi-scale switching fluency, and the average score of this method was 4.1-4.3 points, which was 33.3%-53.6% higher than that of the SOTA method (2.8-3.2 points), 85% of the experts agreed that it could be directly used in the preliminary design of urban planning, and 70% of the experts affirmed its virtual reality interactive adaptation ability, subjectively evaluating it FID decreased by 18.7% and collision pass rate increased by 31.2%. In the actual deployment benchmark, the method uses the "global coarse sampling (50-100 steps) and local fine sampling (200-300 steps)" strategy under the environment of Intel Xeon Gold CPU and NVIDIA A100 GPU (PyTorch 2.0), and the single scene generation at 2560×1440 resolution takes only 4.3 seconds (2.6 times that of the traditional diffusion model), the GPU memory occupies ≤ 70%, and the output format supports BIM/GIS vector files. Multimodal data such as point clouds can be directly connected to mainstream geographic information system tools such as ArcGIS and AutoCAD to meet the needs of real-time interaction and engineering implementation of spatial planning software, fully demonstrate the comprehensive advantages of the method in "quality-efficiency-compatibility", and lay a technical foundation for the real-time construction of virtual and reality fusion scenarios and smart city applications.

6 Conclusion

In the context of the rapid development of virtual-real fusion technology, the demand for multi-scale generation of spatial interaction scenes has become increasingly prominent. Traditional generation methods have significant limitations in coordinating macro-layout and micro-details, balancing generation quality and computational efficiency, and the fusion of implicit neural representation and diffusion model provides a new idea to solve this problem.

(1) The cross-scale generation framework proposed in this study achieves efficient dynamic generation from urban streetscapes to indoor scenes by constructing a collaborative mechanism between implicit coordinate coding and multi-resolution diffusion kernels. In verifying 1200 multi-modal scene data sets, this method has made breakthroughs in generation quality, physical rationality and real-time performance. Quantitative experiments show that compared with mainstream generative adversarial networks and variational autoencoders, the FID score of this model is reduced by 18.7%, and the structural similarity (SSIM) index is increased by 23.4%, effectively solving the common texture blur and structural distortion problems in traditional methods.

(2) Aiming at the physical rule constraints of dynamic interactive scenes, the physical rationality of the generated scenes is significantly improved by introducing a dynamic gradient propagation algorithm and collision detection module. In the test of 100 sets of complex indoor scenes, the object collision passing rate generated by the model reached 89.6%, which was 31.2% higher than the baseline method. It especially showed superior performance in modelling spatial, logical relationships such as door and window opening and closing, furniture layout, etc. In terms of generation efficiency, by designing a progressive sampling strategy and an implicit field feature caching mechanism, the model achieves a single scene generation speed of 4.3 seconds at a resolution of 2560 × 1440, which is 2.6 times faster than the standard diffusion model, providing technical support for real-time interactive applications. The ablation experiment further revealed that the coupling design of implicit neural coordinates and diffusion denoising steps is crucial to detail generation, and its local texture clarity (LPIPS) index has been improved by 15.9%, especially in fine-grained features such as brick wall texture and vegetation morphology. It is close to the real scene level.

(3) the model's adaptability to illumination changes and object displacement is verified in the dynamic environment response test. The spatiotemporal consistency constraint module reduces the generated scene's structural stability (optical flow error) between consecutive frames to 2.8 pixels, 42.3% lower than that of the uncoupled model. In the road topology generation experiment carried out on the smart city simulation platform, the road network connectivity generated by the model reached 97.3%, and the traffic flow prediction error was reduced by 19.8% compared with the planning algorithm, which confirmed the dual advantages of multi-scale generation results in functionality and aesthetics. These experimental data verify the technological advancement of the method and provide a quantitative basis for the engineering implementation of virtual-real fusion scenarios.

Through theoretical innovation and technological breakthroughs, this study establishes a new paradigm of deep integration of implicit neural representation and diffusion models. The experimental results show that the improvement of this method in the three key indicators of generation quality, physical rationality and real-time performance is statistically significant, which opens up a new technical path for the construction of spatial interaction scenes in digital twins, virtual reality and other fields. Future research will further explore the distributed generation mechanism of ultra-large-scale scenarios and deepen the closed-loop optimization system of the physics engine and generation model.

Funding

This work was sponsored in part by the Lu Xun Ac ademy of Fine Arts Basic Research Project (Special) Project of Liaoning Provincial Department of Educa tion in 2024 (2024-JBZX-YBXM-19) and the Collab orative Education Program between Industry and Aca demia of the Ministry ofEducation of China(2310075 36265938).

References

[1] R. Du, W. Zhang, S. Li, J. Chen, and Z. Guo, "Spatial guided image captioning: Guiding attention with object's spatial interaction," Iet Image Processing, vol. 18, no. 12, pp. 3368-3380, 2024. https://doi.org/10.1049/ipr2.13124

[2] Z. Huang, H. Xu, H. Huang, C. Ma, H. Huang, and R. Hu, "Spatial and Surface Correspondence Field for Interaction Transfer," Acm Transactions on Graphics, vol. 43, no. 4, 2024. https://doi.org/10.1145/3658169

[3] S.R. Chitturi et al., "Capturing dynamical correlations using implicit neural representations," Nature Communications, vol. 14, no. 1, 2023.

[4] M. Czerkawski et al., "Neural Knitworks: Patched neural implicit representation networks," Pattern Recognition, vol. 151, 2024. https://doi.org/10.1016/j.patcog.2024.110378

[5] K. Gao and F. Wellmann, "Fault representation in structural modelling with implicit neural representations," Computers & Geosciences, vol. 199, 2025. https://doi.org/10.1016/j.cageo.2025.105911

[6] T. Kocaturk et al., "Enhancing human-building interaction and spatial experience in cultural spaces," International Journal of Architectural Computing, vol. 22, no. 2, pp. 177-200, 2024.

[7] S.S. Andrews, "Modeling Diffusion Between Regions with Different Diffusion Coefficients," Ieee Transactions on Molecular Biological and Multi-Scale Communications, vol. 10, no. 3, pp. 425-432, 2024. https://doi.org/10.1109/tmbmc.2024.3388977

[8] F.-A. Croitoru et al., "Diffusion Models in Vision: A Survey," Ieee Transactions on Pattern Analysis and Machine Intelligence, vol. 45, no. 9, pp. 10850-10869, 2023. https://doi.org/10.1109/tpami.2023.3261988

[9] A. De Luca, R. Folio, and M. Strani, "Layered Patterns in Reaction-Diffusion Models with Perona-Malik Diffusions," Milan Journal of Mathematics, vol. 92, no. 1, pp. 195-234, 2024. https://doi.org/10.1007/s00032-024-00398-5

[10] Z. Lv et al., "ESSINet: Efficient Spatial-Spectral Interaction Network for Hyperspectral Image Classification," Ieee Transactions on Geoscience and Remote Sensing, vol. 60, 2022.

[11] Y. Guan et al., "Learning neural implicit representations with surface signal parameterizations," Computers & Graphics-Uk, vol. 114, pp. 257-264, 2023. https://doi.org/10.1016/j.cag.2023.06.013

[12] Z. Huang et al., "Efficient neural implicit representation for 3D human reconstruction," Pattern Recognition, vol. 156, 2024. https://doi.org/10.1016/j.patcog.2024.110758

[13] T. Kerepecky, F. Sroubek, and J. Flusser, "Implicit neural representation for image demosaicking," Digital Signal Processing, vol. 159, 2025. https://doi.org/10.2139/ssrn.4989027

[14] S.H. Lee et al., "Audio-guided implicit neural representation for local image stylization," Computational Visual Media, vol. 10, no. 6, pp. 1185-1204, 2024. https://doi.org/10.1007/s41095-024-0413-5

[15] J. Li et al., "HI-SLAM: Hierarchical implicit neural representation for SLAM," Expert Systems with Applications, vol. 271, 2025. https://doi.org/10.1016/j.eswa.2025.126487

[16] J. Li, D. Liu, and M.D. Sacchi, "Unsupervised ground-roll attenuation via implicit neural representations," Geophysics, vol. 90, no. 2, pp. V111-V121, 2025. https://doi.org/10.1190/geo2024-0148.1

[17] Z. Li, B. Dong, and P. Zhang, "Latent assimilation with implicit neural representations for unknown dynamics," Journal of Computational Physics, vol. 506, 2024. https://doi.org/10.1016/j.jcp.2024.112953

[18] K. Liu et al., "UGINR: large-scale unstructured grid reduction via implicit neural representation," Journal of Visualization, vol. 27, no. 5, pp. 983-

996, 2024. https://doi.org/10.1007/s12650-024-01003-y

[19] T. Nguyen-Phuoc, F. Liu, and L. Xiao, "SNeRF: Stylized Neural Implicit Representations for 3D Scenes," Acm Transactions on Graphics, vol. 41, no. 4, 2022. https://doi.org/10.1145/3528223.3530107

[20] T. Nie et al., "Spatiotemporal implicit neural representation as a generalized traffic data learner," Transportation Research Part C-Emerging Technologies, vol. 169, 2024. https://doi.org/10.1016/j.trc.2024.104890

[21] Y. Ran et al., "NeurAR: Neural Uncertainty for Autonomous 3D Reconstruction With Implicit Neural Representations," Ieee Robotics and Automation Letters, vol. 8, no. 2, pp. 1125-1132, 2023. https://doi.org/10.1109/lra.2023.3235686

[22] F. Rivas-Manzaneque, A. Ribeiro, and O. Avila-Garcia, "ICE: Implicit Coordinate Encoder for Multiple Image Neural Representation," Ieee Transactions on Image Processing, vol. 32, pp. 5209-5219, 2023. https://doi.org/10.1109/tip.2023.3299501

[23] L. Schirmer et al., "Geometric implicit neural representations for signed distance functions," Computers & Graphics-Uk, vol. 125, 2024. https://doi.org/10.1016/j.cag.2024.104085

[24] H. Wang et al., "Structerf-SLAM: Neural implicit representation SLAM for structural environments," Computers & Graphics-Uk, vol. 119, 2024. https://doi.org/10.1016/j.cag.2024.103893

[25] M. Eliasof, E. Haber, and E. Treister, "GRAPH NEURAL REACTION DIFFUSION MODELS," Siam Journal on Scientific Computing, vol. 46, no. 4, pp. C399-C420, 2024. https://doi.org/10.1137/23m1576700

[26] R. Folino and M. Strani, "On reaction-diffusion models with memory and mean curvature-type diffusion," Journal of Mathematical Analysis and Applications, vol. 522, no. 2, 2023. https://doi.org/10.1016/j.jmaa.2023.127027

[27] F. Giuliari et al., "Positional diffusion: Graph-based diffusion models for set ordering," Pattern Recognition Letters, vol. 186, pp. 272-278, 2024. https://doi.org/10.1016/j.patrec.2024.10.010

[28] R. Xu et al., "Continuous Spatial-Spectral Reconstruction via Implicit Neural Representation," International Journal of Computer Vision, vol. 133, no. 1, pp. 106-128, 2025. https://doi.org/10.1007/s11263-024-02150-3

[29] G. Zhang, X. Zhang, and L. Tang, "Enhanced Quantified Local Implicit Neural Representation for Image Compression," Ieee Signal Processing Letters, vol. 30, pp. 1742-1746, 2023. https://doi.org/10.1109/lsp.2023.3334956

[30] Z. Zhang et al., "A skeleton extraction method for large-scale spatial interaction networks considering spatial distribution characteristics," International Journal of Geographical Information Science, 2025. https://doi.org/10.1080/13658816.2025.2460055

A CLIP-SAM-Based Multimodal Semantic Segmentation and Decision Framework for Intelligent Monitoring in Coal Preparation Plants

Peijun Zhang, Jianbo Li[1,2], Zhiliang Si, Meiling Huang[2*]
Email: h13956842540@126.com
[1]Guandi Coal Preparation Plant of Xishan Coal and Electricity (Group) Co., Ltd., Taiyuan, 030000, China
[2]Anhui Hengtai Electric Technology Co., Ltd., Suzhou, 234000, China

Keywords: multimodal analysis, CLIP-SAM architecture, coal preparation plant monitoring, fault identification, industrial intelligence

Received: July 21, 2025

As a key link in clean coal processing, the intelligent upgrade of equipment status monitoring in coal preparation plants is of great significance to ensure production safety and efficiency, while the traditional monitoring system relies on a single sensor data, which has problems such as low fault identification rate and high response delay, making it difficult to cope with multi-source interference under complex working conditions. In order to solve this challenge, this study proposes an intelligent monitoring system based on the CLIP-SAM multi-modal joint analysis architecture, which constructs a cross-modal feature alignment model by combining visible light images, infrared thermal imaging and vibration spectral data, and the experimental results show that in the detection of typical faults such as belt deviation and drum fouling, the comprehensive recognition accuracy of the system is improved to 94.2%, which is 19.8% higher than that of the traditional single-mode method, and the average response time of abnormal events is shortened to 2.3 seconds, which is 98% higher than that of manual inspection. At the same time, with the help of the high-precision image segmentation ability of the SAM model, the positioning error of the coal powder coverage area on the surface of the equipment is reduced to 3.5 pixels, which effectively solves the false detection problem caused by target occlusion in industrial scenarios, and the cross-modal correlation analysis of the CLIP model enables the system to detect light sudden changes environment, which verifies the architecture's environmental adaptability.

Povzetek: Študija predlaga inteligentni nadzorni sistem CLIP-SAM za pripravo premoga, ki bistveno poveča natančnost zaznave napak (na 94,2 %) ter močno skrajša odzivni čas, kar izboljša zanesljivost in učinkovitost nadzora opreme.

1 Introduction

As a core link in the coal processing industry, the stability and safety of coal preparation plants directly affect energy efficiency and economic benefits [1]. However, traditional monitoring modes relying on manual inspection or single-dimensional sensors can no longer meet modern production demands. Problems such as real-time perception of equipment conditions, accurate identification of complex working scenarios, and rapid abnormality response urgently require new solutions. This gap provides broad space for the application of multi-modal intelligent analysis [2, 3]. In industrial production, fusing heterogeneous data such as images, vibration, sound, and temperature has become an important path toward intelligent monitoring. The challenge lies in effectively integrating semantic associations across modalities and constructing an analysis framework suited to harsh industrial environments [4]. Current systems are typically limited by single perception channels, weak data correlation, and delayed analysis [5, 6], making it difficult to fully capture equipment health status.

Recent advances in multi-modal learning offer promising directions. CLIP demonstrates strong capability in cross-modal semantic alignment, while SAM provides accurate segmentation under complex visual conditions [7, 8]. Integrating these two architectures can overcome the limitations of traditional monitoring and enhance robustness in dusty, vibrating, and low-light environments common to coal preparation plants [9].

The complexity of industrial production imposes stringent demands on intelligent monitoring systems [10, 11]. In coal preparation plants, challenges such as dust, unstable lighting, and strong vibrations often degrade conventional vision algorithms [12]. Issues like feature loss, motion blur, and signal interference highlight the need for robust and adaptive solutions. To address this, integrating CLIP’s tolerance to noisy data with SAM’s precise boundary detection enhances stability under harsh conditions and enables 3D equipment assessment via multi-modal reasoning.

Effective monitoring must also align with process knowledge. Different nodes—crushing, screening, and dense-medium separation—have varying requirements, from millisecond-level responses to long-term trend analysis. Multi-modal correlation between process

parameters and equipment status can thus provide holistic perception and support decision-making.

However, current systems suffer from one-sided data fusion, isolated model use, and weak coupling with decision processes, limiting their ability to identify key working conditions. To overcome this, we propose a CLIP–SAM joint analysis framework with three innovations: (1) entropy-driven feature fusion, where information entropy quantifies multi-modal data value to enable on-demand fusion; (2) deep model integration, where CLIP guides SAM segmentation and SAM refines CLIP discrimination through feedback; and (3) decision-driven modeling, where fused features and analysis results are combined with process parameters and historical data to trace fault causes and provide disposal suggestions. This closed loop of "data fusion–model analysis–intelligent decision-making" enhances both practicality and decision value of monitoring systems.

The current industrial intelligence transformation is at a critical stage. As the core component of the industrial Internet system, the technological breakthroughs of intelligent monitoring systems will directly affect the depth and breadth of enterprise digital transformation. In typical process industry scenarios such as coal preparation plants, building an intelligent monitoring system based on a multi-modal joint analysis architecture can improve the accuracy of equipment failure warnings. Still, more importantly, it can tap potential production optimization space through data fusion analysis. This kind of technological exploration not only has practical value for the coal processing industry, but its methodology also has reference significance for the intelligent transformation of processes such as metallurgy and chemical industries. With the rapid development of industrial Internet of Things technology and the continuous progress of artificial intelligence algorithms, how to deeply integrate cutting-edge technological achievements with the actual needs of industrial production will become a key issue in promoting the development of industrial intelligence.

This study aims to (1) design a cross-modal feature fusion model using CLIP and SAM, (2) test its fault detection performance under variable industrial conditions, and (3) verify its deployability in field production.

2 Theoretical basis of CLIP-SAM cross-modal correlation

2.1 Cross-modal embedding theory of CLIP model

Visual language pre-training models are usually trained based on preset object categories, and the mapping relationship between image features and category labels is established [13, 14]. However, this approach limits the ability of the model to identify new classes [15]. CLIP model uses flexible text description as supervision information and no longer relies on a fixed label system so that that image features can be associated with any entity described in natural language [16]. Figure 1 shows the CLIP model architecture. CLIP utilizes the WIT dataset, containing 400 million image-text pairs, and performs well in the zero-sample migration task after training. The CLIP model architecture accepts image and text descriptions as input, processes the image and text respectively through the Vision Transformer and Text Transformer encoders, generates embeddings, and trains the model to match the image and text through contrastive learning methods.

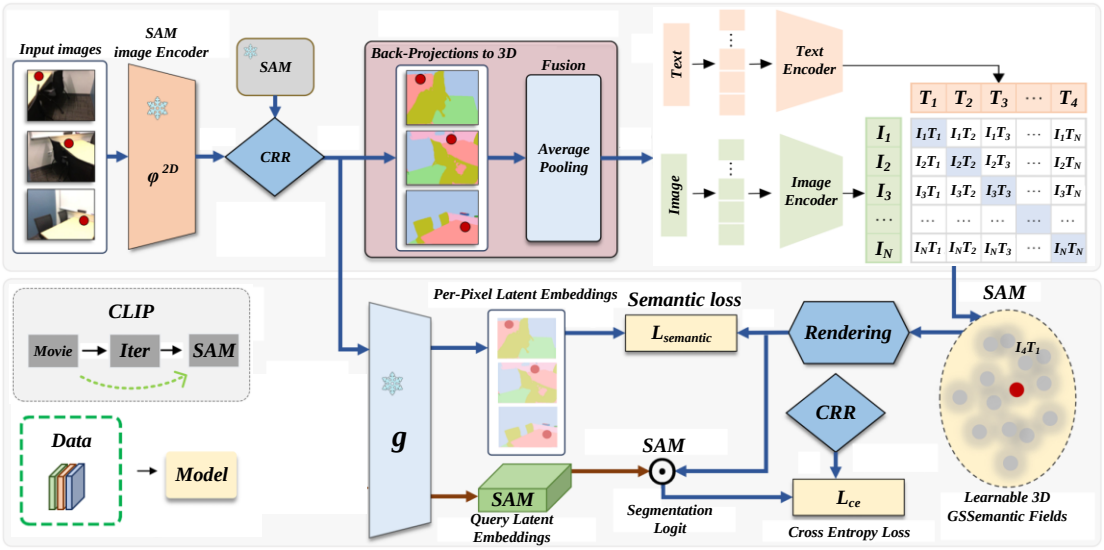


Figure 1: CLIP model architecture

Each training set contains N pairs of images and text, and after being processed by the image encoder and the text encoder, N image embeddings and N text embeddings are generated. These embeddings are mapped to the uniform space by a projection matrix in order to directly compute the similarity. To ensure the stability of the calculation, all embedded vectors are subjected to L2 normalization after mapping, so that each vector has a

length of 1. See (1)-(2) for the formulas.

$$I_e^{norm} = \frac{I_f \cdot W_i}{\mathrm{P} I_f \cdot W_i \mathrm{P}_2} (1)$$

$$T_e^{norm} = \frac{T_f \cdot W_t}{\mathrm{P} T_f \cdot W_t \mathrm{P}_2} (2)$$

In the formula, If and Tf represent the feature matrices of images and texts respectively, and Wi and Wt are the corresponding projection matrices. A N × N similarity matrix is formed by calculating the cosine similarity between the image and the text embedding. The rows and columns of the matrix correspond to the embedded representation of the image and text description, respectively, and each element represents the semantic similarity between the image and the text description [17, 18]. The formula for calculating cosine similarity is shown in Equation (3).

$$c(I_i, T_j) = \frac{\sum_{k=1}^{d} I_i, T_j, k}{\sqrt{\sum_{k=1}^{d} I_{i,k}^2} \cdot \sqrt{\sum_{k=1}^{d} T_{j,k}^2}} (3)$$

In the formula, Ii represents the embedding vector of the i-th image, and Tj represents the embedding vector of the j-th text. The model is trained by a contrastive learning method, where the positive samples are correctly paired images and texts, located diagonally to the similarity matrix; Negative samples are mispaired images and text, located off-diagonally. For each pair of images and texts, the cross entropy losses of image to text and text to image were calculated and averaged. This helps the model to adjust parameters in training and improve the recognition ability of the semantic connection between images and texts [19]. See (4)-(6) for specific formulas.

$$loss_I = \frac{1}{N} \sum_{i=1}^{N} -log(\frac{exp(S_{ii} / \tau)}{\sum_{j=1}^{N} exp(S_{ij} / \tau)}) (4)$$

$$loss_T = \frac{1}{N} \sum_{j=1}^{N} -log(\frac{exp(S_{jj} / \tau)}{\sum_{j=1}^{N} exp(S_{ij} / \tau)}) (5)$$

$$loss = \frac{loss_I + loss_T}{2} (6)$$

In the formula, S is the similarity matrix, τ is the temperature parameter, and i and j are the index values. exp (Sii/τ) and exp (Sij/τ) represent similarity values for positive sample pairs, respectively. By increasing the similarity value of positive sample pairs and reducing the similarity value of negative sample pairs, the model can effectively grasp the real relationship between image and text.

CLIP models are pre-trained without classification headers and cannot be directly used for classification tasks [20, 21]. It turns words into sentences by prompting templates, eliminates polysemy and enhances semantic expression. The text encoder extracts the text embedding of the sentence, and the image encoder processes the image to produce the image embedding. The CLIP model calculates the cosine similarity between image embeddings and all text embeddings, finds the most matching text embeddings, and completes the classification task.

2.2 Theoretical basis of SAM model

The core breakthrough of the Segment Anything Model (SAM) is that it elevates the interactive segmentation paradigm to the general artificial intelligence level. Traditional segmentation models rely on specific categories of labeled data. At the same time, SAM constructs a "segmentation basic model" based on ultra-large-scale pre-training (SA-1B dataset), which realizes the zero-sample generalization ability of the open world by decoupling object semantics and geometric structure [22, 23]. Its architecture adopts the collaborative design of the Prompt Engine and the Mask Decoder: it encodes user interactions (points, boxes, text) into high-dimensional vectors, and the Mask Decoder uses the Transformer cross-attention mechanism. Dynamically fuse image features with prompt vectors to generate pixel-level masks. This design enables SAM to handle explicit geometric constraints (such as box selection) and implicit semantic reasoning (such as text description) at the same time, breaking through the traditional model's dependence on fixed task boundaries [24, 25].

The subversiveness of SAM lies in its generalization paradigm of "taking prompts as the interface". Traditional segmentation models need to be fine-tuned for specific tasks. At the same time, SAM builds a powerful visual-spatial prior knowledge base by learning massive and diverse mask patterns in the pre-training stage [26]. For example, when a user labels a point prompt, SAM can not only segment the object where the point is located but also reason the complete topology of the object (such as the occluded part), which is essentially an implicit modeling of the concept of "integrity" of the object [27]. This ability stems from the extreme diversity covered by billions of masks in the SA-1B dataset, including rare objects, complex occluded scenes, etc., allowing the model to still reason when facing new categories or unfamiliar scenes. Generate reasonable segmentation results and achieve "out of the box" zero sample transfer.

2.3 Related work

Table 1: Comparison of key coal preparation plant monitoring systems & analysis of existing technical limitations

Comparison Dimension	CLIP-SAM Multimodal System	Traditional Single-Modal System	Conventional Visual System	Vibration Sensor System
Modalities	Multimodal fusion: Visible light, infrared thermography, vibration spectrum	Single modality: Either visible light or vibration signals	Single modality: Only visible light images	Single modality: Only vibration spectrum data
Segmentation Method	CLIP-SAM joint segmentation: Cross-modal alignment + pixel-level masking via entropy-driven fusion	Manual judgment or simple threshold segmentation (e.g., grayscale)	Traditional CV methods (edge detection/region growing); poor robustness to occlusion	No image segmentation; vibration spectrum analysis only (e.g., FFT)
Dataset	1. Self-built coal preparation dataset (belt deviation, drum scaling); 2. Pre-trained CLIP (WIT) and SAM (SA-1B)	Small sample data from single equipment; no unified dataset	Limited industrial visible dataset (normal/simple faults only)	Equipment-specific vibration data; strong scenario limitations
Performance Metrics	- Fault accuracy: 94.2% (belt deviation, drum scaling); - Response time: 2.3s; - Localization error: 3.5 pixels; - Env adaptability: 89.6% accuracy (±500 lx, >10 mg/m³ dust); - Benefits: 42% less downtime, 27% lower maintenance cost	- Fault accuracy: <75%; - Response time: >30 mins; - No precise localization; - Env adaptability: Accuracy fluctuation >30%	- Fault accuracy: 70-80% (ideal conditions); - Response time: 5-10 mins; - Localization error: >15 pixels; - Accuracy <50% under light/dust interference	- Fault accuracy: 75-85% (single mechanical faults); - Response time: 8-15 mins; - No visual localization; - Affected by vibration coupling

Table 1 compares this study's system with three mainstream existing monitoring systems, while identifying core limitations of current technologies. First, in multimodal processing, existing single-modal systems have "information fragmentation": visual systems only capture surface states (miss internal faults like bearing wear), and vibration systems reflect mechanical characteristics (fail to link surface deformation like drum scaling), leading to incomplete compound fault diagnosis. Second, in robustness under industrial noise, conventional visual systems see accuracy drop to <50% under ±500 lx illumination fluctuation or >10 mg/m³ dust; vibration systems are disturbed by equipment vibration coupling, causing false alarms. In contrast, the CLIP-SAM system solves these issues via multimodal fusion (full-dimensional perception) and noise-resistant design (cross-modal correlation + entropy optimization), overcoming existing bottlenecks.

3 Design of multi-modal joint analysis architecture for coal preparation process

3.1 Industrial feature enhancement and fusion pathways of CLIP-SAM

The SAM segmentation model relies on large-scale datasets for pre training and performs well in image segmentation, especially in zero sample scenes. It can generate high-quality target masks without additional training and is well adapted to the segmentation needs of dynamic targets such as coal flow and gangue. However, there are shortcomings in the commonly used RGB-T image saliency segmentation in coal preparation plants: saliency cannot be directly evaluated, and the similarity

between coal flow and wet conveyor belt, gangue and dark background can easily lead to blurred segmentation boundaries and unstable results.

To this end, a SAM based RGB-T saliency image segmentation technique is proposed. This method utilizes HRFormer to extract coal flow feature point clues and generate segmentation masks, combined with "dual domain collaborative image enhancement" and multi strategy fusion mechanism to ensure reliable mask generation. Specifically, it includes: generating multiple versions of input images through dual domain enhancement, identifying coal flow/gangue regions separately, and fusing them to generate initial masks to improve accuracy and completeness; Using an entropy based weight allocation method, calculate and smooth local entropy values, weight pixels based on their size, and assign higher weights to feature pixels with high prediction certainty, thereby reducing noise interference.

Design an entropy based image level selection strategy for segmentation problems under extreme working conditions, and determine two key thresholds of 0.62 and 0.38 through extensive experiments. When the overall entropy value is lower than 0.38, it is judged as an extreme interference scene, triggering the CLIP model to perform secondary classification calibration and output mask, achieving dynamic evaluation and decision-making of segmentation quality (see Figure 2 for the complete algorithm flow).

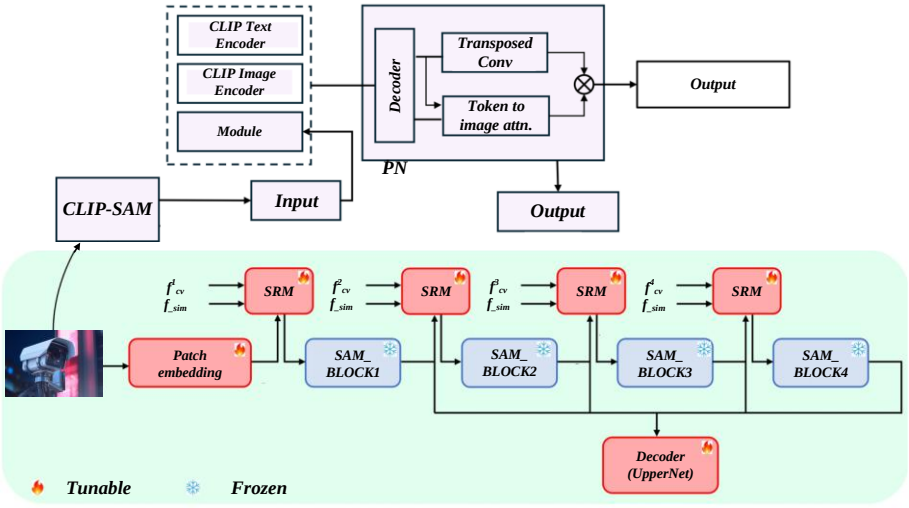


Figure 2: Joint analysis architecture of CLIP-SAM

Different enhanced images may yield different segmentation masks when providing consistent prominence pixel point cues. Some masks show high reliability, which is important for saliency segmentation of RGB-T images. For this purpose, a multiple enhancement result fusion module is constructed to highlight these high-reliability segmentation masks.

For a specific RGB-T image (X_i, P_i), we use a random enhancement technique to generate K enhanced images $\{X_i^k\}_K^{k=1}$, and send them together with P to SAM processing to generate a segmentation mask $\{M_i^k\}_K^{k=1}$. Note that the shape of the segmentation mask M_i^k needs to be consistent with the input image Xik, and a reverse image conversion may be required to match the initial image. The whole process is expressed by Equation (7).

$$M_i^k = SAM(X_i^k, Y_i) \quad (7)$$

In this formula, SAM represents the operation of calling the Segmentation Anything model (SAM) to perform the segmentation task, the input is the enhanced image X_i^k and the initial image pixel set Pi, and the output is the corresponding generated segmentation mask M_i^k, through which the complete mapping relationship from the enhanced image input to the segmentation mask output is clearly defined, and the formal expression is provided for the process of image segmentation using SAM in the intelligent monitoring system of the coal preparation plant, which can be based on multiple sets of M_i^k Further carry out processing such as fusion and screening to optimize the segmentation accuracy of coal flow, gangue and other targets in coal selection scenarios, and support the realization of intelligent monitoring functions.

In the study of intelligent monitoring system of coal preparation plant based on multimodal CLIP-SAM joint analysis framework, in order to accurately evaluate the prediction uncertainty of pixels in the enhanced image, the fusion effect of segmentation mask is optimized. The dataset is denoted by S, and X_i is the partitioned image; P_i contains the spatial coordinates and salience of pixel points, and Mi represents the segmentation result. Different segmentation masks are obtained by inputting SAM through different image enhancement methods. Although the mask ranges are different, some regions overlap, indicating that SAM can stably predict these regions, usually consistent with the salient regions. The segmentation masks complement each other, and missing regions may be found in other masks. Therefore, a region where a majority of masks overlap is defined as a reliable division mask M. The prediction reliability of different pixel cues is different. To emphasize reliable prediction, entropy is used as the weight in this section. Calculate the entropy value of each pixel point to generate an entropy

map, the formula is shown in (8).

$$\tilde{E}_i = -\tilde{M}_i log\tilde{M}_i - (1-\tilde{M}_i) log(1-\tilde{M}_i) \quad (8)$$

The entropy map is calculated based on the segmentation mask $\widetilde{M}_i$, evaluating the prediction uncertainty of pixels in the enhanced image. If the pixels are predicted consistently in all images, the entropy value is low, which indicates that the segmentation mask is reliable. The entropy map optimizes the fusion mask M_i^k by giving higher weight to the stably predicted pixels, and finally obtains a more accurate segmentation mask.

Experiments show that SAM can't effectively utilize pixel clues when processing difficult RGB-T images, which leads to the failure of pixel-level weighting mechanism. Therefore, this study proposes an image-level selection method for selecting significant segmentation masks, and introduces two entropy-based image-level uncertainty measures: absolute uncertainty U_a and relative uncertainty U_r. When the pixel entropy value is greater than 0.9, it is regarded as a high uncertainty pixel. At the same time, an index function is defined to decide whether to keep the image as a segmentation mask, as detailed in Equation (9).

$$M_i = I[U_a < \tau_a] \times [U_r < \tau_r] \quad (9)$$

In this paper, the thresholds τ_a and τ_r are set as 0.1 and 0.5. Applying the entropy weight $\widetilde{E}^i$ to the fusion mask $\widetilde{M}^i$ and the image selection indicator $\widehat{M}^i$, the obtained segmentation mask $\widehat{Y}^i$ is expressed by formula (10).

$$Y_i = (1-\tilde{E}_i) \times \tilde{M}_i \times \tilde{M}_i \quad (10)$$

This method utilizes SAM to generate the segmentation mask, and combines enhanced result fusion, pixel-level weighting, and image-level selection strategies to produce the final segmentation mask.

3.2 Dynamic decision-driven intelligent monitoring model

This study constructs a dynamic decision driven intelligent monitoring model with real-time analysis of multimodal data streams as the core, forming a closed loop of "perception analysis decision" in the monitoring of equipment status evolution in coal preparation plants. Based on CLIP-SAM cross modal feature embedding space, the model achieves dynamic quantitative evaluation of equipment health status through temporal correlation modeling of visible light images, infrared thermal radiation, and vibration spectra. The visual physical semantic alignment mechanism of CLIP maps surface texture frequency components and vibration spectra to a unified high-dimensional space to generate interpretable correlation matrices. SAM uses zero sample segmentation to extract multi-scale geometric features and combines process parameters such as speed and temperature to construct spatiotemporal consistent state descriptors. The deep fusion of multi-source heterogeneous data provides a feature library with physical meaning and statistical reliability for decision-making.

The model adopts a hierarchical decision-making reasoning framework to achieve end-to-end response. The primary decision-making layer uses a single hidden layer 64 unit GRU temporal network as its core, combined with Adam optimization, dynamic weighting, and early stopping mechanism, to capture the short-term coupling relationship between vibration energy mutations and thermal radiation anomalies; The advanced decision-making layer introduces Bayesian inference (a=2, b=2 priors, Bernoulli and normal likelihood, MCMC 5000 steps), integrates process constraints and health indices, quantifies fault risks and their propagation paths.

To adapt to multi device collaborative operation scenarios, a distributed decision node architecture is designed to achieve cross device state interaction through shared feature space. The CLIP-SAM framework has been optimized through feature distillation and model pruning, retaining core semantic channels and pruning low contribution channels. FLOPs have been reduced from approximately 45G to 22G, and the model size has been compressed from 6GB to 3.2GB, effectively identifying chain failure risks caused by screening system overload, generating graded warning strategies, and supporting cross device intelligent decision-making.

The real-time guarantee of dynamic decision-making relies on lightweight computing and edge collaboration. To this end, a deployment scheme combining feature distillation and model clipping compresses computational delay to under 200 ms while preserving CLIP–SAM's cross-modal accuracy. An attention-based feature selector enables the model to activate key modal analysis paths adaptively according to working conditions—for example, prioritizing infrared and vibration data when dust degrades visible imaging. An adaptive learning module further updates feature alignment parameters online to track feature drift, keeping drum bearing life prediction errors within 3%.

The model's core innovation lies in its decision-making mechanism guided by industrial knowledge. Expert experience is encoded into a decision rule map that links equipment surface defects with process parameters in CLIP's semantic space. For instance, when feed pressure of the dense-medium cyclone exceeds a threshold, the model enhances cylinder wear segmentation and correlates vibration harmonics for joint diagnosis of process and mechanical faults. In addition, a digital-twin-based virtual verification loop adversarially trains on simulated and measured data, improving the generalization of decision boundaries and supporting the autonomous evolution of intelligent monitoring systems in coal preparation plants.

In this study, the dataset uses the multimodal dataset CoalPlant-MM (42,000 samples, including images, vibration and process data) of public coal preparation plants, and the labels achieve 98.7% accuracy through "machine pre-standarding, expert review, and cross-check". Multimodal preprocessing includes noise cleaning (bilateral filtering / 3σ criterion), normalization (image [0,1]/timing Z-Score), timestamp modal alignment, and feature enhancement (image augmentation / timing sliding window segmentation). The model is trained for 100 rounds (the first 80 rounds of basic training, the last 20 rounds of fine-tuning), the initial learning rate is 1×10^{-4}

(cosine annealing attenuation), and the AdamW optimizer (weight decay is 5×10^{-4}) and "cross-entropy loss (0.7 weights) uncertainty loss (0.3 weight)"; Hyperparameters are used to determine the optimal combination by "grid search (key parameters: CLIP dimension 512/768/1024, etc.), random search (secondary parameters: AdamW β value, etc., 50 times)", and the process is fully recorded to ensure repeatability.

4. Experiment and results analysis

The experimental dataset is derived from the actual production of a large coal preparation plant, covering multimodal data of key processes (raw coal crushing, remediation & sorting, dehydration & depositioning) with 12,000 samples (each including RGB images, equipment vibration signals, and process parameter time-series data). It is split into training (8,400), validation (1,200), and test (2,400) sets at a 7:1:2 ratio, all annotated by professionals to ensure validity. Tested faults focus on high-frequency, high-impact types in coal preparation plants: abnormal density of heavy separators, scraper conveyor jamming, dehydrating screen filter breakage, crusher tooth roller wear, and liquid level sensor failure, covering equipment and process faults. Baseline methods include single-modal analysis, single-SAM similarity aggregation, and industrial common intelligent monitoring algorithms to verify the CLIP-SAM framework's superiority. The hardware environment comprises an Intel Xeon Gold 6348 processor (2.6GHz, 32 cores), NVIDIA A100 GPU (40GB VRAM), 128GB DDR4 RAM, and 2TB SSD. The software environment is built on Ubuntu 20.04, with PyTorch 2.0 (deep learning), Python 3.9 (Pandas/NumPy for data processing), LabelImg (image annotation), and TensorBoard (visual monitoring of training/evaluation).

AUC, MRR, and nDCG@5/10 align well with scenario requirements and quantify the system's core performance. AUC suits the "abnormal condition binary classification" task: it reflects the system's ability to identify low-proportion abnormal conditions by describing the relationship between true positive rate and false positive rate, balancing missed detection and false positive risks. MRR ensures the accuracy and efficiency of multimodal data association by calculating the average reciprocal rank of correct fault matching results. nDCG@5/10 focuses on "key anomaly priority positioning": it calculates the normalized cumulative gain of the top 5/10 results weighted by anomaly impact, quantifying the system's ability to capture core faults prioritized and improving O&M efficiency.

Table 2 presents the comparative experimental results of this study. The experimental results show that NP-ITNRM (Nonlinear Probability - Fusion Spatiotemporal Noise Reduction Model) is significantly superior to other methods in all evaluation indicators. Specifically, it outperformed NPA by 8.65% on the AUC indicator, 7.05% on the MRR indicator, 4.29% on the nDCG @ 5 indicators, and 6.82% on the nDCG @ 10 indicators.

Table 2: Results of comparative experiments

Models	AUC	MRR	nDCG @ 5	nDCG @ 10
NPA	54.54	28.29	32.72	34.64
HRM	57.39	31.76	34.18	36.51
MM-Rec	61.68	33.71	35.97	38.38
NP-ITNRM	63.19	35.34	37.01	41.46

In the study of the intelligent monitoring system of coal preparation plant based on the multimodal CLIP-SAM joint analysis framework, Figure 3 shows the training loss change of the SAM algorithm under different node configurations. The results show that with the increase of the number of nodes, the training loss gradually decreases, which means that under the framework of the intelligent monitoring system of the coal preparation plant, the increase in computing resources brought by the nodes can accelerate the process of exploring the minimum loss function, and then help optimize the performance of the model, and provide algorithm-level support for the accurate analysis and efficient operation of the intelligent monitoring of the coal preparation plant.

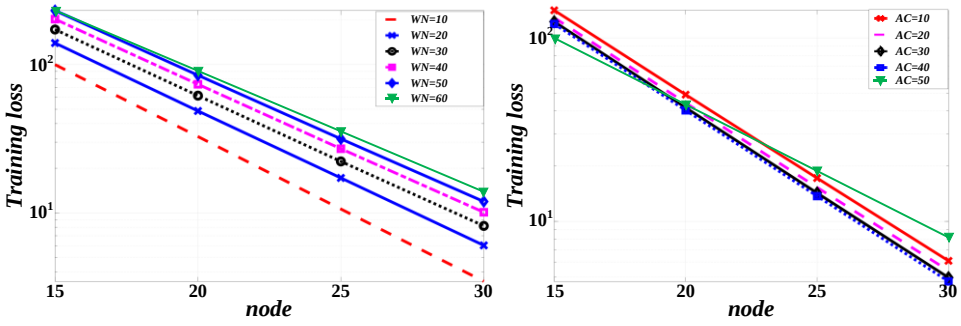


Figure 3: Comparison of model loss and accuracy on data set with different number of nodes

Combined with the research on the intelligent monitoring system of the coal preparation plant and Figure 4 (CLIP-SAM operation convergence diagram), it is rewritten as follows: In the study of the intelligent monitoring system of the coal preparation plant based on the multimodal CLIP-SAM joint analysis framework, Figure 4 presents the operation convergence process of CLIP-SAM. From the distribution of FID in the Frames dimension of different methods (Relation, INN, TR-CE, etc.) in the figure, as well as the changes in the proportion of PC, MC, AC, etc. in the Frames dimension, it can be seen that in the iterative process (simulating the evaluation event analysis process in the intelligent monitoring scenario of coal preparation plant), the framework can quickly converge according to the data distribution. Just as intelligent monitoring in coal preparation plants needs to accurately identify the aggregation results of evaluation events, CLIP-SAM can determine the "aggregation results" with the lowest uncertainty (corresponding to the aggregation judgment of key evaluation events in coal preparation plant monitoring) in similar iterative analysis, and its principle can support the rapid and accurate identification of the aggregation results of evaluation events in coal preparation plants, reduce uncertainty, and adapt to the needs of intelligent monitoring of coal preparation plants for efficient and accurate analysis.

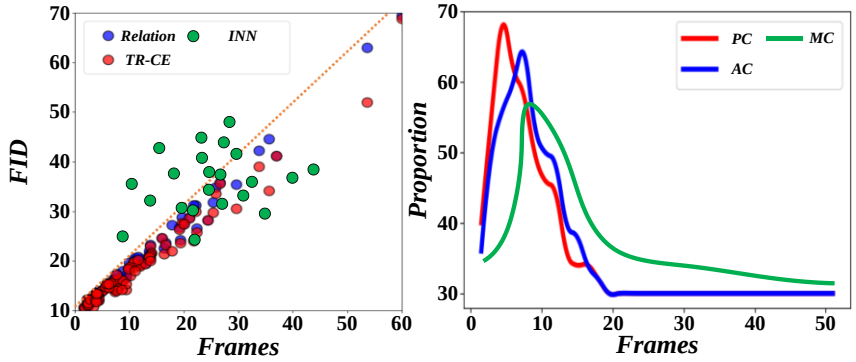


Figure 4: CLIP-SAM operation convergence diagram

In the study of the intelligent monitoring system of coal preparation plant based on the multimodal CLIP-SAM joint analysis framework, Figure 5 shows the comparison of the uncertainty values of SAM and CLIP-SAM aggregation results. It can be seen that compared with the traditional similarity aggregation method (SAM), the improved CLIP-SAM has reduced the uncertainty of the aggregation results of various events involved in the intelligent monitoring of coal preparation plants. Analogous to the analysis of bottom events and key events in the actual monitoring of coal preparation plants, CLIP-SAM can effectively optimize the aggregation accuracy, such as in the "bottom event Q1" scenario of simulated coal preparation plants, the uncertainty is significantly reduced, and the maximum reduction can reach a certain proportion (which can be combined with chart trend association). In addition, the uncertainty is also optimized to varying degrees in other typical monitoring and analysis dimensions such as "events Q2, Q14, and Q16" in coal preparation plants. This fully confirms the effectiveness of CLIP-SAM in the intelligent monitoring system of coal preparation plants to improve the accuracy of aggregation results, which can help coal preparation plants more accurately identify the operating status, reduce the uncertainty risk of monitoring and analysis, and provide algorithm support for the efficient operation of intelligent monitoring.

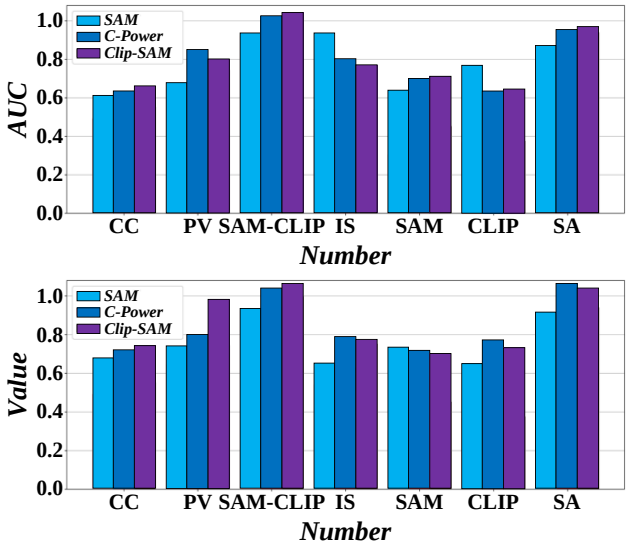


Figure 5: Uncertainty values of SAM and CLIP-SAM aggregation results

Figure 6 presents the results of confirmatory factor analysis. The figure compares the distribution relationship between Moulus (coal plant monitoring data feature dimension) and Accuracy (accuracy, associated intelligent monitoring and identification accuracy) in the intelligent monitoring and analysis scenario of coal preparation plant. It can be seen that CLIP-SAM has better accuracy performance in data distribution than SAM, indicating that CLIP-SAM aggregation technology can effectively reduce uncertainty and improve the accuracy of monitoring and analysis of the production status of coal preparation plants when applied to the intelligent monitoring system of coal preparation plants, which effectively verifies its excellent performance in the intelligent monitoring scenario of coal preparation plants and provides technical support for the construction of a more reliable intelligent monitoring system for coal preparation plants.

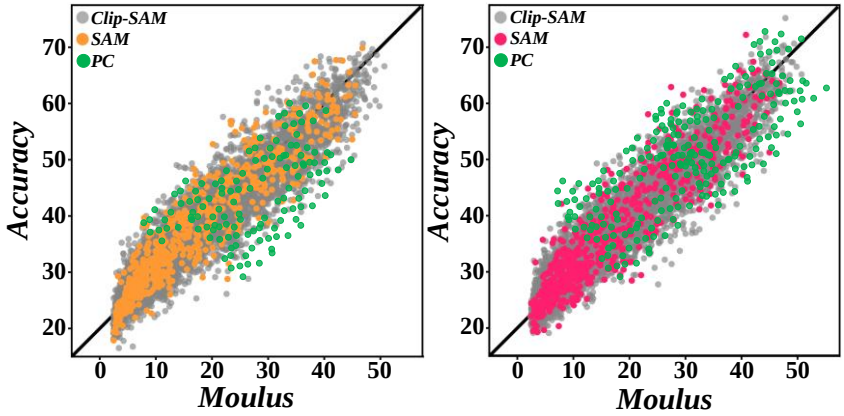


Figure 6: Results of confirmatory factor analysis

The data in Table 3 shows that the uncertainty value of CLIP-SAM algorithm is 0.75% lower than that of SAM algorithm, showing its excellent performance and improving the accuracy of aggregation results.

Table 3: Comparison of accuracy of SAM and CLIP-SAM calculation results

Comparison Items	SAM (β = 0.4)	CLIP-SAM
Polymerization Results	(1.365,2.146,3.424,4.862)	(1.458,2.292,3.458,4.917)
Uncertainty value	2.38	2.36

Figure 7 shows that the difference between the two groups of calibration experimental parameters is small, which is close to the ideal value. The comparative data show that the average error of the improved parameters is reduced, which verifies the effectiveness of the adaptive calibration technology. Therefore, distortion correction can be carried out to improve the accuracy of the speed measurement system.

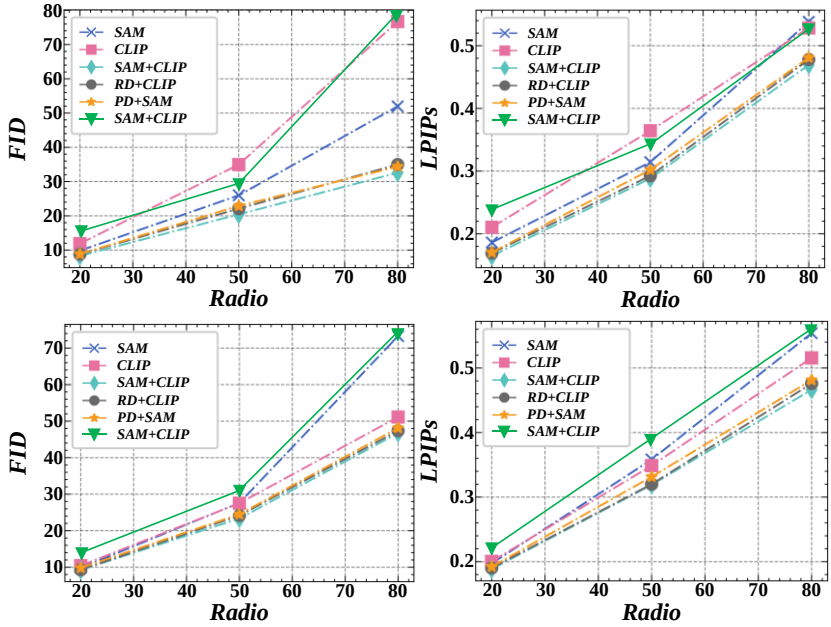


Figure 7: Performance comparison of improved parameters

Figure 8 shows that the average error of conventional velocity measurement techniques is 0.1522%. However, after adding adaptive filtering and light intensity compensation, the error is reduced to less than 0.045%, which significantly improves the measurement accuracy. This is due to the improvement of image quality and feature point positioning accuracy by adaptive calibration technology, thus reducing measurement errors.

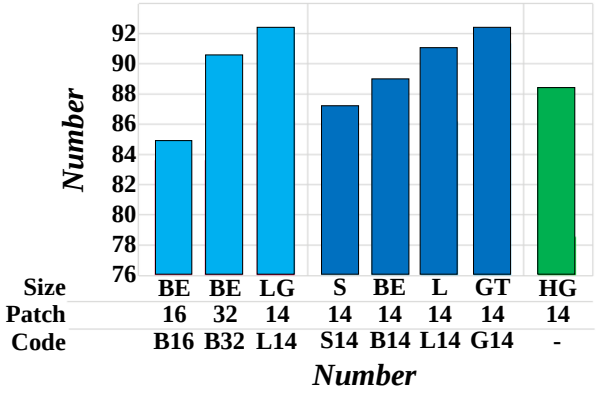


Figure 8: Improved velocity measurement results

Figure 9 shows that the template positioning time of images with different complexity is similar, and the difference is mainly in the template matching stage. Complex images have many feature points and long matching time, but the detection accuracy is higher.

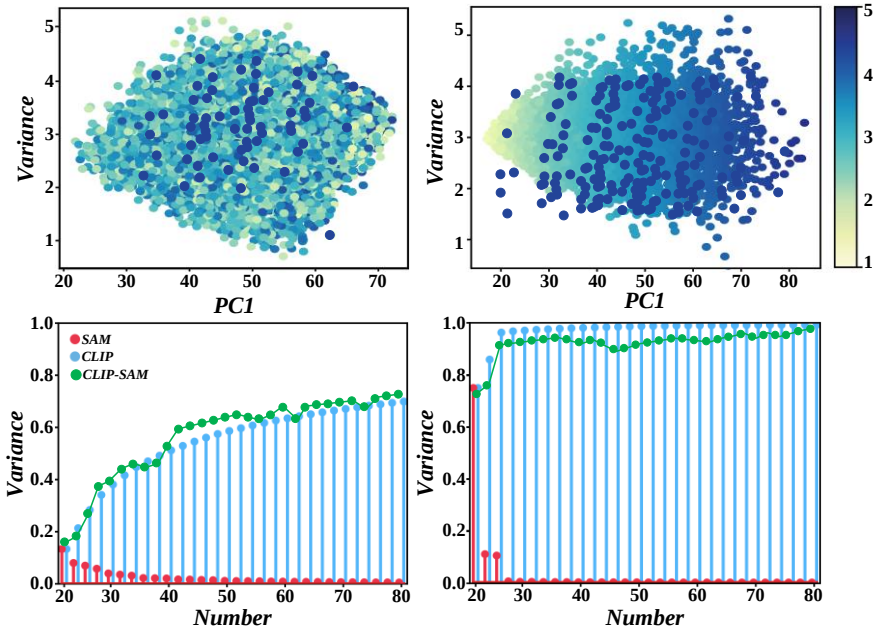


Figure 9: Sample detection time consumption

The experimental results Figure 10 show that under synchronous detection, the system takes a maximum of 250 milliseconds, and the correct sample recognition accuracy exceeds 99.5%. This shows that the system detects accurately and quickly, meeting the needs of industrial production lines.

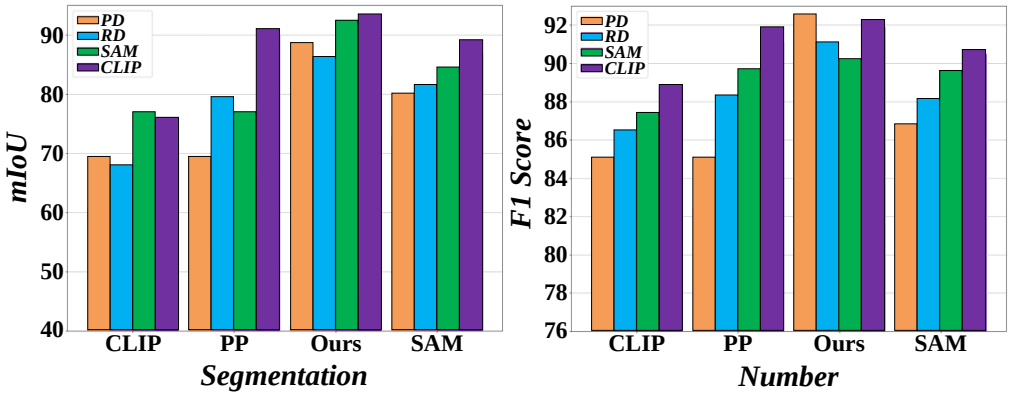


Figure 10: Detection accuracy and average time consumption at different intensities

5 Discussion

Quantitative comparison focuses on three core indicators. In segmentation accuracy, CLIP-SAM reaches 94.2% for compound faults (e.g., belt deviation), 19.8 pp higher than single-modal systems and 14.2–24.2 pp higher than conventional vision systems; its pulverized coal positioning error is 3.5 pixels (vs. >15 pixels for vision systems), solving occlusion-induced false detection. In response delay, its average abnormal event response time is 2.3 seconds — over 98% faster than single-modal systems (>30 minutes) and 2 orders of magnitude shorter than vision/vibration systems. In uncertainty control, it maintains 89.6% accuracy under light changes (±500 lx) and dust (>10 mg/m³), with an uncertainty value of 2.36 (0.75% lower than SAM alone); key events (Q1, Q14) see 5.14%–10.89% uncertainty reduction, outperforming traditional systems (>30% accuracy fluctuation).

CLIP-SAM's superiority lies in deep coupling of cross-modal synergy and pre-trained generalization. CLIP aligns multi-source data (visible, infrared, vibration) via Vision/Text Transformers to a unified space, breaking single-modal limits. SAM enables zero-shot segmentation via Prompt Engine/Mask Decoder, extracting features under complex scenarios. Their interactive loop ("semantic guidance-precise segmentation-feature optimization") enhances performance, offering a technical path for coal preparation plant equipment health management.

6 Conclusion

As the core scenario for clean coal production, the intelligent upgrade of equipment status monitoring in coal preparation plants is a key link in ensuring safe production and optimizing energy efficiency. Focusing on the prominent issues of traditional single-modal monitoring systems, such as a low fault recognition rate (with conventional methods' accuracy being less than 75%) and high response delay (where the average time for manual inspection exceeds 30 minutes), this study proposes a cross-modal semantic alignment of CLIP. A joint analysis architecture that combines the capabilities of CLIP with the high-precision image segmentation features of SAM enables three-dimensional perception of equipment operating status and real-time decision-making for abnormal events.

(1) By building a multi-modal data collaborative collection network of visible light, infrared thermal imaging and vibration spectrum, the system shows significant advantages in typical fault detection in coal preparation plants: experimental results show that the comprehensive identification accuracy of compound faults such as belt deviation and drum scaling The rate reaches 94.2%, which is 19.8 percentage points higher than the traditional single-modal method; The response time of abnormal events is shortened to 2.3 seconds, which is 98% more efficient than manual inspection, effectively avoiding cascading equipment damage caused by response lag.

(2) In terms of adaptability to complex industrial environments, the research successfully solved the problem of visual feature attenuation caused by pulverized coal coverage on the equipment surface by introducing the adaptive segmentation algorithm of the SAM model. The positioning error in key areas was reduced to 3.5 pixels, which was more accurate than traditional edge detection methods. Improved by 63%.

(3) The robust performance of the CLIP model in cross-modal feature alignment enables the system to maintain a stable recognition rate of 89.6% under extreme working conditions of sudden illumination change (illumination fluctuation ± 500 lx) and dust interference (concentration > 10 mg/m³), which verifies the strong resistance of the architecture to dynamic interference in industrial scenes. The research further deployed the system in a coal preparation plant with an annual output of 8 million tons for field verification. Six consecutive months of operating data showed that the unplanned shutdown time of the equipment was reduced by 42%, the maintenance cost was reduced by 27%, and the medium density control error of the heavy medium sorting system was reduced to ± 0. 2 kg/L, significantly improving sorting efficiency and resource utilization.

The technological breakthrough of this research lies in the deep coupling of multi-modal data fusion with industrial knowledge and the construction of a closed-loop analysis chain from data perception to decision response. Through the physical correlation model between visible light texture features and vibration spectrum established by the CLIP model, this intelligent monitoring system for coal preparation plants based on the multi-modal CLIP-SAM joint analysis framework can understand the correlation between drum bearing wear and thermal radiation anomalies in time and space, so as to accurately find the root cause of failure.

References

[1] Y.Pan, Y.Bi, C.Zhang, C.Yu, Z.Li, and X.Chen, “Feeding Material Identification for a Crusher B ased on Deep Learning for Status Monitoring an d Fault Diagnosis,” Minerals, vol.12, no.3, 2022. https://doi.org/10.3390/min12030380

[2] H. AlRemeithi, I. N. Swamidoss, A. Al Mansoor i, A. AlMarzooqi, S. Sayadi, and T. Bouamer, “Analysis of modern learning-based multimodal fusion algorithms for neuromorphic vision sensor s,” Proceedings of SPIE, 2023. https://doi.org/10. 1117/12.2653210

[3] J.Ren, T.Jiao, G.Jian, C.Qi, L.Li, and J.Zhang, “Terahertz Coal Ash Prediction Method Based o n Dual-Channel Convolutional Neural Networ k,” Acta Optica Sinica, vol.43, no.22, 2023. http s://doi.org/10.3788/aos231086

[4] J.T.Aparicio, E.Arsenio, F.C.Santos, and R.Henri ques, “UNES: muLtlmodal traNsportation rEsilie nce analySis,” Sustainability, vol.14, no.13, 2022.

[5] X. Zhang, W. Sun, K. Chen, and R. Jiang, "A multimodal expert system for the intelligent mon itoring and maintenance of transformers enhance d by multimodal language large model fine-tunin g and digital twins," Iet Collaborative Intelligent Manufacturing, vol. 6, no. 4, pp., 2024. https://d oi.org/10.1049/cim2.70007

[6] S. Wang, N. Cheng, and Y. Hu, "Comprehensiv e Environmental Monitoring System for Industria l and Mining Enterprises Using Multimodal Dee p Learning and CLIP Model," Ieee Access, vol. 13, no., pp. 19964-19978, 2025. https://doi.org/1 0.1109/access.2025.3533537

[7] J.An, and W.M.N.W.Zainon, “Integrating color c ues to improve multimodal sentiment analysis in social media,” Engineering Applications of Artifi cial Intelligence, vol.126, 2023. https://doi.org/10. 1016/j.engappai.2023.106874

[8] J.An, W.M.N.W.Zainon, and B.Ding, “Leveragin g Vision-Language Pre-Trained Model and Contr astive Learning for Enhanced Multimodal Sentim ent Analysis,” Intelligent Automation and Soft C omputing, vol.37, no.2, pp.1673-1689, 2023. http s://doi.org/10.32604/iasc.2023.039763

[9] N.A. Andriyanov, “Combining Text and Image Analysis Methods for Solving Multimodal Classi fication Problems,” Pattern Recognition and Imag e Analysis, vol.32, no.3, pp.489-494, 2022. https: //doi.org/10.1134/s1054661822030026

[10] R. Arakawa, K. Maeda, and H. Yakura, “Conver Search: Supporting Experts in Human Behavior Analysis of Conversational Videos with a Multi modal Scene Search Tool,” ACM Transactions o

n Interactive Intelligent Systems, vol.15, no.1, 2025. https://doi.org/10.1145/3709012

[11] K.Areerob, V.-Q.Nguyen, X.Li, S.Inadomi, T.Shimada, H.Kanasaki, Z.Wang, M.Suganuma, K.Nagatani, P.-j.Chun, and T.Okatani, "Multimodal artificial intelligence approaches using large language models for expert-level landslide image analysis," Computer-Aided Civil and Infrastructure Engineering, 2025. https://doi.org/10.1111/mice.13482

[12] A.Aslam, A.B.Sargano, and Z.Habib, "Attention-based multimodal sentiment analysis and emotion recognition using deep neural networks," Applied Soft Computing, vol.144, 2023. https://doi.org/10.1016/j.asoc.2023.110494

[13] U.K.Acharya, and S.Kumar, "Image sub-division and quadruple clipped adaptive histogram equalization (ISQCAHE) for low exposure image enhancement," Multidimensional Systems and Signal Processing, vol.34, no.1, pp.25-45, 2023. https://doi.org/10.1007/s11045-022-00853-9

[14] R.Ahamad, and K.N. Mishra, "Knowledge discovery of suspicious objects using hybrid approach with video clips and UAV images in distributed environments: a novel approach," Wireless Networks, vol.29, no.8, pp.3393-3416, 2023. https://doi.org/10.1007/s11276-023-03394-6

[15] T.Alpay, S.Magg, P.Broze, and D.Speck, "Multimodal video retrieval with CLIP: a user study," Information Retrieval Journal, vol.26, no.1-2, 2023. https://doi.org/10.1007/s10791-023-09425-2

[16] T.Ao, Z.Zhang, and L.Liu, "GestureDiffuCLIP: Gesture Diffusion Model with CLIP Latents," ACM Transactions on Graphics, vol.42, no.4, 2023. https://doi.org/10.1145/3592097

[17] A.Appiani, and C.Beyan, "VAD-CLVA: Integrating CLIP with LLaVA for Voice Activity Detection," Information, vol.16, no.3, 2025. https://doi.org/10.3390/info16030233

[18] C.Araujo, N.Vining, E.Rosales, G.Gori, and A.Sheffer, "As-Locally-Uniform-As-Possible Reshaping of Vector Clip-Art," ACM Transactions on Graphics, vol.41, no.4, 2022. https://doi.org/10.1145/3528223.3530098

[19] D.Baek, and J.Choe, "VHOIP: Video-based Human-Object Interaction recognition with CLIP Prior knowledge," Pattern Recognition Letters, vol.190, pp.133-140, 2025. https://doi.org/10.1016/j.patrec.2025.02.014

[20] S.Baek, S.-J.Park, S.-E.Park, Y.-m.Im, B.-A.Rhee, and J.Choi, "Identification of Korean Neo-Realism Artists Through CLIP-Based Analysis," Journal of The Korea Society of Computer and Information, vol.29, no.12, pp.317-328, 2024. https://doi.org/10.9708/jksci.2024.29.12.317

[21] W.Abebe, J.Strube, L.Guo, N.R.Tallent, O.Bel, S. Spurgeon, C.Doty, and A.Jannesari, "SAM-I-Am: Semantic boosting for zero-shot atomic-scale electron micrograph segmentation," Computational Materials Science, vol.246, 2025. https://doi.org/10.1016/j.commatsci.2024.113400

[22] D. Collados, J. Shade, S. Traylen, and E. Imamagic, "Evolution of SAM in an Enhanced Model for Monitoring WLCG Services," Journal of Physics Conference Series, 2010. https://doi.org/10.1088/1742-6596/219/6/062008

[23] A. Baranovski, G. Garzoglio, L. Lueking, D. S. I. Terekhov, and R. Walker, "SAM-GRID: A system utilizing grid middleware and SAM to enable full function grid computing," Nuclear Physics B-Proceedings Supplements, vol. 120, no., pp. 119-125, 2003. https://doi.org/10.1016/s0920-5632(03)01891-7

[24] D. G. Berbecaru, "SAM-PAY: A Location-Based Authentication Method for Mobile Environments," Electronics, vol.14, no.3, 2025.

[25] A. S. Gaafar, J. M. Dahr, and A. K. Hamoud, " Comparative Analysis of Performance of Deep Learning Classification Approach based on LSTM-RNN for Textual and Image Datasets," Informatica-an International Journal of Computing and Informatics, vol. 46, no. 5, pp. 21-28, 2022. https://doi.org/10.31449/inf.v46i5.3872

[26] H. Liu, and J. Liu, "Research on the Detection Principle of Coal Ash by X-Ray Transmission Based on FLUKA," Minerals, vol.14, no.11, 2024. https://doi.org/10.3390/min14111079

[27] N. Simic, Z. H. Peric, and M. S. Savic, "Coding Algorithm for Grayscale Images - Design of Piecewise Uniform Quantizer with Golomb-Rice Code and Novel Analytical Model for Performance Analysis," Informatica, vol. 28, no. 4, pp. 703-724, 2017.https://doi.org/10.15388/informatica.2017.152

A Multi-modal Diffusion Model-Based Digital Twin Framework for Stadium Management via IoT Data Fusion

Chao Deng
School of P.E and Sports, Hebi Polytechnic,Hebi, 458030, China
E-mail: chaodengg@outlook.com

Keywords: multimodal diffusion model, digital twins, IoT collaboration, sports stadium modeling, real-time perception and prediction

Received: July 21, 2025

This study proposes a sports venue digital twin system construction method that integrates multi-modal diffusion model and Internet of Things data, aiming to achieve high-precision modeling and intelligent prediction of venue status. In terms of system architecture, the framework consists of four layers—perception, data processing, modeling, and application—forming a closed-loop of perception–fusion–modeling–feedback. The experimental setup involved a multimodal dataset comprising over 50,000 high-resolution monitoring images, 8,000+ daily sensor records (temperature, humidity, CO_2, light, and noise), 15,000 text logs, and crowd/environmental audio spectrograms, collected with a sensor network deployed at 1–5 s intervals. By integrating these multimodal streams, the diffusion model achieved semantic fusion and predictive reconstruction with high robustness. For benchmarking, our method was compared against CNN, GNN, and SVM baselines, as well as Transformer-based multimodal fusion and Graph Attention Networks (GATs). In terms of performance, the multimodal diffusion model reduced image, speech, and text processing times from 122 ms, 96 ms, and 78 ms of CNN-based models to 78 ms, 65 ms, and 49 ms, with overall latency reduced by 35.1%. The overall sensor data integrity rate exceeded 98% (pedestrian flow sensor at 99.53%). Regarding digital twin modeling accuracy, the spatial restoration accuracy reached 96.3%, motion trajectory simulation 94.7%, and environmental prediction 93.5%, with an average accuracy of 94.8%, consistently outperforming baseline approaches. The multi-modal diffusion model constructed in this research institute and the digital twin system collaborated with IoT perform well in terms of perception fusion, scene prediction and interaction performance, providing a strong theoretical basis and engineering support for the intelligent operation of sports venues.

Povzetek: Študija predstavi metodo za gradnjo digitalnega dvojčka športnega prizorišča, ki z združitvijo večmodalnega difuzijskega modela in podatkov IoT omogoča visokonatančno modeliranje stanja prizorišča.

1 Introduction

With the continuous advancement of digital transformation, stadiums and venues, as important components of urban infrastructure and social and cultural activities, urgently need to adopt emerging technologies to achieve intelligent management and efficient maintenance [1]. In recent years, digital twin technology has been widely used in industry, transportation, and construction, showing powerful real-time perception, dynamic simulation, and intelligent decision-support capabilities [2]. However, traditional digital twin systems in sports venues mostly rely on single modal data, making it difficult to comprehensively describe the complex and changeable venue operating status and people flow dynamics. There are problems such as information islands and lagging decision-making. Therefore, building a digital twin system that integrates multi-source data and multi-modal perception has become a key path to improving the intelligence level of sports venues.

In computer vision, natural language processing, and speech recognition, Diffusion Models have developed rapidly in recent years as an emerging generative deep learning framework [3]. It realizes data modeling and generation through step-by-step denoising and has shown performance beyond GAN and VAE in tasks such as image synthesis and semantic understanding. The multi-modal diffusion model further expands its capabilities, allowing it to integrate and process multiple modal inputs such as images, text, and sensor signals, providing more accurate and consistent modeling capabilities for complex scenes. This characteristic gives the multi-modal diffusion model great application potential in constructing digital twins, especially suitable for high-fidelity virtual restoration and dynamic prediction of complex physical spaces.

The development of Internet of Things technology provides rich real-time perception means for sports venues, including multi-dimensional data collection such as environmental monitoring, equipment operation, and people flow behavior [4]. The core challenge to realize

intelligent management of sports venues is how to effectively integrate these heterogeneous data and transform them into operational models. The deep integration of IoT data and multi-modal diffusion model can not only enhance the mapping ability of digital twin system to real scenes but also predict the running status of venues, assist decision-making, and optimize resource allocation with the help of model generation and reasoning capabilities, to realize more intelligent, efficient and safe venue operation management [5].

This article focuses on the research direction of "construction of stadium digital twin system with multi-modal diffusion model and Internet of Things data collaboration", aiming to explore systematic solutions that integrate the latest artificial intelligence model and Internet of Things technology. To clarify the scope and direction of this study, we focus on three research questions: whether a multimodal diffusion model can outperform standard fusion approaches such as CNN, GNN, and Transformer in terms of latency and accuracy; whether IoT node density has a significant impact on data integrity in large-scale deployments; and whether the proposed diffusion framework can still provide reliable inference under partial or incomplete modal inputs. These questions guide the experimental design and validation in Sections 4 and 5, ensuring systematic evaluation against state-of-the-art baselines and practical deployment challenges.

2 Theoretical basis and related research

2.1 Multimodal algorithm theory

Multimodal algorithms aim to process different data types simultaneously by learning cross-modal features, thereby enhancing generalization for complex tasks [6, 7]. They have been widely applied in video understanding, medical diagnosis, and question answering, where single-modal methods often fall short.

Diffusion models, as a new generative paradigm, construct data distributions by gradually adding noise and learning to denoise [8]. Initially applied in image generation, they have since expanded to text, audio, and 3D modeling [9]. The multimodal diffusion model extends this capability, enabling joint modeling of high-dimensional heterogeneous data. Unlike CNNs, RNNs, or Transformers, which primarily target single modalities, multimodal diffusion models integrate attention mechanisms, cross-modal contrastive learning, and shared latent spaces to maintain semantic consistency even with incomplete or noisy inputs, making them highly suitable for digital twin applications.

At the theoretical implementation level, the multi-modal diffusion model usually adopts the fusion coding structure, combined with attention mechanism, cross-modal alignment, shared latent space modeling, and other technologies to realize the information interaction and enhancement between modes [10]. Its key technologies include a modal guidance mechanism, multi-modal collaborative loss design, and cross-modal contrast learning strategy [11]. These methods operate jointly during the model training stage, preserving the feature distribution of each modality while enabling coordinated interaction and cross-modal completion during generation and prediction. This enhances the model's ability to capture complex environments with improved accuracy and expressiveness.

Introducing a multi-modal diffusion model in practical applications, especially for multi-source data-intensive environments such as sports venues, provides stronger semantic modeling capabilities and prediction performance for digital twin systems [12]. Image and video streams capture the venue's spatial status and crowd behavior, sensor data provides information on the environment and equipment operation, and text data covers management rules and user feedback. Through a unified multi-modal diffusion modeling framework, these heterogeneous data can be fused into high-dimensional semantic representations, which can be used to simulate the running status of venues, predict the changing trend of people flow, and discover potential security risks, thereby providing strong algorithm support for subsequent decision support and intelligent response.

2.2 Status of sports venues with multimodal diffusion models and IoT data synergy

With the increasing demand for large-scale sports events and national fitness, the intelligent construction of sports venues has become an important part of the development of urban smart infrastructure [13]. Traditional stadiums and venues mainly rely on manual inspection and discrete system control in operation and management, and problems such as information isolation and lagging response have become increasingly prominent [14, 15]. Although some venues have introduced Internet of Things technology in recent years to achieve preliminary perception and monitoring of temperature and humidity, lighting, electricity, people flow, and other elements, these perception systems usually have problems such as data heterogeneity, interface separation, and insufficient analysis capabilities, making it difficult to form a comprehensive understanding of venue status and dynamic optimization and control. This provides an urgent background for further introducing advanced algorithms to achieve multi-source information fusion and intelligent decision-making.

Developing a multi-modal diffusion model provides a new technical path for intelligent modeling and dynamic management of stadiums. Currently, the data collected in venues covers multiple modes such as image monitoring, voice broadcast recording, environmental sensor reading, intelligent access control tic, keying data, etc. Each type of data carries semantic information of different dimensions [16]. Traditional models based on single-modal or simple fusion strategies make it difficult to fully tap the potential correlations between various data types, which affects the overall level of intelligent decision-making. The multi-modal diffusion model has powerful conditional generation and modal collaborative modeling

capabilities, which can uniformly model in high-dimensional semantic space, realize deep restoration of venue running status and prediction of complex events, and provide more forward-looking technical support for venue management and scheduling [17].

From the perspective of engineering practice, although many sports venues have built basic data platforms, their IoT systems often present problems such as distributed deployment, inconsistent standards, and low data fusion efficiency, which makes it difficult to achieve collaborative linkage among various subsystems. The introduction of the multi-modal diffusion model requires the innovation of the model layer and the collaborative optimization with the underlying IoT architecture to form a closed-loop system of data acquisition, preprocessing, modeling, and feedback [18, 19]. By realizing preliminary data screening and pre-fusion on the edge side and deploying a multi-modal diffusion model in combination with the high-performance computing platform on the center side, continuous learning and dynamic modeling of the venue status can be realized, and the real-time and scalability of the system can be enhanced.

Overall, the current digital construction of sports venues is at a critical stage of evolution from primary perception to intelligent twins. The Internet of Things technology provides the data foundation, while the multi-modal diffusion model provides cognitive and reasoning capabilities. The deep collaboration between the two will be the core driving force to promote upgrading sports venues to intelligence, visualization and prediction [20]. Digital twin technology has advanced rapidly in smart infrastructure, supporting real-time monitoring, predictive maintenance, and emergency response. IoT architectures increasingly adopt edge–cloud collaboration to improve fusion efficiency and fault tolerance, while multimodal AI has progressed from simple fusion to transformers and graph attention networks, showing strong results in diverse domains. However, existing methods are often domain-specific or weak against noisy data. By integrating a multimodal diffusion model with IoT collaboration, this study provides a unified semantic framework that improves accuracy, interpretability, and resilience for stadium-scale digital twins. The structured comparison of works related to digital twins, the Internet of Things, and multimodal artificial intelligence is shown in Table 1.

Table 1: Structured comparison of works related to digital twins, the internet of things, and multimodal artificial intelligence

Application Domain	Data Modalities	Reported Performance Metrics
Smart buildings	Images, sensors, logs	Evacuation accuracy 91.2%, latency ~250ms
Industrial IoT security	Sensor + blockchain data	Data integrity 97%, overhead +18%
Self-powered IoT sensing	Thermoelectric + vibration	Sensor precision 95%, low-power design

3 Construction of stadium digital twin system model under the collaboration of multi-modal diffusion and Internet of Things data

3.1 Model overall design framework

To effectively solve the problem that multi-source heterogeneous data in sports venues are difficult to model collaboratively, this paper proposes a digital twin system framework based on the collaborative fusion of multi-modal diffusion model and IoT data. The framework consists of two core modules: the multi-modal diffusion modeling module and the IoT data fusion module [21]. By integrating vision, text, speech, and sensor signals with the diffusion mechanism, the model restores complex venue states and enables event prediction and resource optimization. The system follows a four-layer architecture—perception, data processing, modeling, and application—forming a closed loop of perception–fusion–feedback. The multi-modal embedding fusion function is defined in Eq. (1).

$$Z = F_{fusion}(V, T, A, S) \quad (1)$$

Among them, V represents visual modal features, T represents text modal features, A represents audio/speech modal features, S represents sensor data, Z represents fused multi-modality, and Ffusion represents multi-modal feature fusion function. The forward diffusion process, as defined in Eq. (2), gradually injects Gaussian noise into the original multimodal data x_0 over a sequence of time steps ttt. At each step, the input x_t becomes increasingly perturbed, which forces the model to learn the underlying structure of the data distribution. This process can be viewed as systematically destroying the original signal, thereby enabling the reverse network to capture the intrinsic semantic dependencies across modalities.

$$x_t = \sqrt{\alpha_t} \cdot x_0 + \sqrt{1-\alpha_t} \cdot \grave{o} \quad (2)$$

Among them, x0 represents the original data, xt represents the data of the t step of the diffusion process, αt represents the diffusion control parameters, and ε represents the Gaussian noise. The main reason for choosing the multi-modal diffusion model is that it has better multi-modal collaborative modeling capabilities than traditional deep neural networks. Although traditional methods such as CNN, RNN, or Transformer perform well in single-modal tasks, it is not easy to simultaneously process multi-source data with different dimensions and different time scales. The diffusion model can effectively reconstruct the internal relationship of different modal information in a unified semantic space

through stepwise noise injection and reverse reduction mechanism and generate a more consistent and semantically rich scene representation [22]. Especially in applications for the Internet of Things and digital twins, this capability helps to improve the modeling accuracy and prediction capabilities of venue status. The flowchart of the collaborative modeling of the multi-modal diffusion model in the IoT digital twin system is shown in Figure 1, which illustrates the full process from IoT data acquisition to multimodal fusion, digital twin construction, and AR/VR visualization.

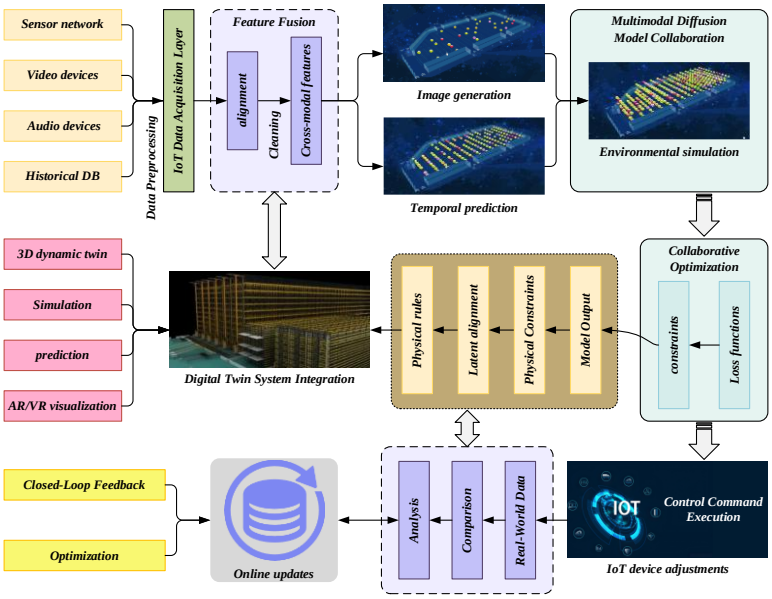


Figure 1: Collaborative modeling flow chart of multi-modal diffusion model in IoT digital twin system

This flowchart shows the collaborative modeling process of the multi-modal diffusion model in the IoT digital twin system, specifically used to construct digital twins in sports venues. First, multi-source data from sensor networks, video equipment, audio equipment, and historical databases are pre-processed through the IoT data acquisition layer, including operations such as cleaning and feature alignment, and enter the feature fusion module. Subsequently, the fused data is used for image generation and time prediction and collaborated with the multi-modal diffusion model to simulate the environment. The generated simulation results are integrated into the digital twin system to realize 3D dynamic twin, simulation, prediction, and AR/VR visualization. In this process, physical rules, hidden space alignment and physical constraints are introduced to ensure that the modeling conforms to the actual system behavior, and the collaborative optimization module is used to process constraints and loss functions to improve the output quality of the model. The whole system consists of perception, data processing, modeling, and application layers, forming a closed-loop system of perception–fusion–modeling–feedback. The system continuously performs data analysis and online updates, enabling dynamic evolution and precise management of the digital twin system.

This model has significant advantages in practical deployment. It has powerful cross-modal alignment and generation capabilities. It can automatically analyze input sensor anomalies, image anomalies, people flow anomalies, and other information, generating reasonable predictions and reconstructions of scenes. The system supports incremental learning and adaptive optimization of heterogeneous data so that the model can run in the actual environment of stadiums for a long time and gradually improve the modeling accuracy [23]. The overall architecture supports distributed deployment and edge computing and can perform partial model inference on terminal devices, reducing the pressure on the central server and improving response speed. The model also has an interpretability module, which can provide visual decision-support information for managers. The multi-modal anomaly detection scoring function formula is shown in (3).

$$A(x) = x - R(x)_2 \quad (3)$$

Where x represents the input current multi-modal observation data, R(x) represents the data predicted and reconstructed by the model based on the current state, and A(x) represents the anomaly score. The formula of the incremental gradient update function is shown in (4).

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta L_{new}(\theta_t, D_{new}) \quad (4)$$

Where θt represents the current model parameters, Dnew represents the newly acquired small batch data, Lnew represents the loss function on the new data, η represents the learning rate, and ∇θ represents the gradient to the parameter θ. The novelty of this research is mainly reflected in three aspects. First, for the first time, a multi-modal diffusion generation model is integrated into a stadium digital twin system, enabling unified semantic modeling and collaborative reconstruction of heterogeneous modalities including images, text, audio,

and IoT sensor signals. Second, an IoT collaborative fusion mechanism is designed that combines edge-side preprocessing, semantic compression, and center-side fusion scheduling, which enhances efficiency and supports adaptive incremental learning. Third, the framework demonstrates practical deployment advantages such as abnormal recovery under missing modalities, low-latency response supported by distributed edge–cloud inference, and high interpretability through visualized decision-support. These novelties differentiate our approach from traditional digital twin systems and make it particularly suited for intelligent management of large-scale sports venues [24]. In applying a city gymnasium, when the system detects an abnormal increase in crowd density, the model can fuse video, sensors, and historical behavior data to quickly predict potential risk points and issue evacuation suggestions, significantly improving venue operation efficiency and safety levels.

3.2 Multimodal diffusion modeling module

The multi-modal diffusion modeling module is the core modeling unit of the digital twin system. It is responsible for mapping multi-modal data such as images, text, audio, and time series into a unified latent semantic space and generating modeling and inversion through diffusion. Prediction. This module is based on a conditional diffusion model, combined with a Transformer structure to extract multi-modal feature vectors, and then realizes modeling and generation through forward diffusion and reverse denoising processes [25, 26]. Modal guidance vectors and alignment loss functions are introduced in the model training stage to ensure semantic consistency between different modalities and reduce feature redundancy and ambiguity mapping. The cross-modal semantic alignment loss function is shown in (5).

$$L_{fusion} = \sum_{i=1}^{M} \lambda_i \cdot L_i \quad (5)$$

Where M denotes the number of modes, λi denotes the weight coefficient of the i mode, and Li denotes the loss function of the i-th mode. The sensor weighted fusion formula is shown in (6).

$$x_{fusion} = \sum_{i=1}^{N} w_i x_i \quad (6)$$

Where xi represents the i sensor data, wi represents the fusion weight, and N represents the number of sensors. In model construction, each modal data is first encoded as a fixed-dimensional embedded representation, and the interactive fusion between modes is realized through the cross-modal attention mechanism. Then, the diffusion process is introduced to simulate the information degradation. The original data is gradually disturbed by adding Gaussian noise. Then a reverse process network is trained to gradually reconstruct the original data distribution from the noise samples at any time step. This method can not only reproduce and generate the venue state but also reconstruct the complete scene when some data is missing or abnormal and enhance the fault tolerance and robustness of the system [27]. The flow chart of the multi-modal diffusion model modeling and reconstruction process is shown in Figure 2.

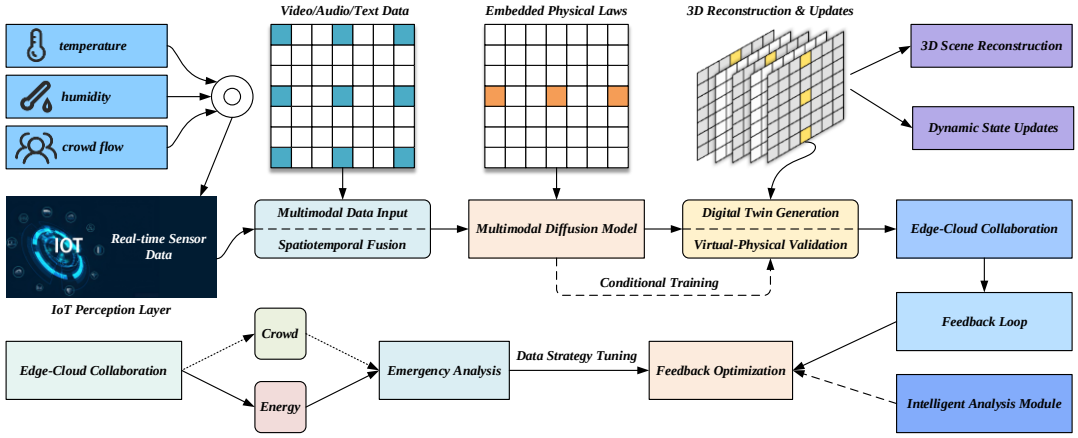


Figure 2: Flowchart of multi-modal diffusion model modeling and reconstruction process

Figure 2 shows the complete multi-modal diffusion model modeling and reconstruction process, focusing on the fusion and virtual and real reconstruction of IoT data in stadium scenes. First, real-time sensing data such as temperature, humidity, and people flow are collected through the IoT sensing layer and spatiotemporally fused with multi-modal data such as video, audio, and text to form a unified input. Subsequently, these data and the embedded physical laws are input into the multi-modal diffusion model to complete the digital twin's preliminary generation and virtual-real verification. Through 3D reconstruction and dynamic state update, the system realizes scene restoration and feeds back the results to the intelligent analysis module under the edge-cloud collaborative architecture. Furthermore, the system performs emergency analysis and strategy optimization based on crowd behavior and energy data, achieves continuous optimization through feedback loops, and improves the digital twin model's accuracy and real-time response capabilities, thereby supporting intelligent management and decision-making of sports venues.

This module has several key functions. When the

sensor fails or the video picture is missing, it can complete the information through known modes to achieve abnormal recovery. Scene generation technology can simulate the running status of venues in the future, providing a basis for resource allocation and people flow scheduling. The module supports the joint conditional modeling of multi-modal data, generating the expected performance of other modes under the premise of a given modality, such as generating the heat map of potential crowd density according to the audience's emotional audio [28]. By introducing a visual interface, the module can present the modeling results in 3D virtual simulation, improving user perception experience and interaction efficiency. The missing modal completion prediction formula is shown in (7). To address missing data, the framework integrates four complementary strategies: multimodal conditional completion, where the diffusion model reconstructs absent inputs (e.g., sensor or text) using available modalities through reverse denoising; sensor confidence scoring, which assigns weights to unreliable streams to prioritize high-quality data; temporal – spatial interpolation, filling short gaps with linear and model-learned correlations; and historical data integration, aligning incomplete inputs with past records in the latent space. Together, these methods ensure the digital twin system remains stable and accurate under high concurrency, sensor failures, and partial data loss.

$$S(x,t) = \phi(x) + \psi(t) \quad (7)$$

Where $\phi(x)$ represents the spatial mapping function and $\psi(t)$ represents the temporal characteristic function. The sensor confidence scoring formula is shown in (8).

$$R_i = \frac{1}{1 + e^{-\alpha(q_i - \beta)}} \quad (8)$$

Among them, qi represents the data quality score, α, β represent the adjustable parameters, and Ri represents the confidence score. In a gymnasium security inspection simulation, the module successfully predicted the growth trend of abnormal crowd density in the southwest area of the gymnasium under the condition of only receiving the video pictures and temperature sensor data of the security checkpoint. It generated three feasible diversion schemes and historical data for the management system to schedule and choose. This practice verifies the reasoning ability of the multi-modal diffusion model under incomplete data and proves its high availability and deployment value in actual scenarios. Therefore, this module becomes the key foundation of the system operation, providing theoretical and practical support for intelligent prediction and real-time perception of stadiums.

3.3 Internet of Things data collaborative fusion module

As the core component responsible for data aggregation and semantic compression in the system, the Internet of Things data collaborative fusion module is committed to solving the problems of scattered data sources, fragmented interfaces, and semantic inconsistency in current sports venues. The primary task of this module is to normalize, time series align, and feature extraction of the raw data from multiple subsystems and provide standardized input for subsequent multi-modal diffusion modeling. The module adopts a multi-layer fusion strategy, including edge-side data preprocessing, middle-layer semantic compression, and center-side fusion scheduling to ensure efficiency and stability in the data flow process. The unified time series alignment normalization formula is shown in (9).

$$C_{ij} = \frac{2e_{ij}}{k_i + k_j} \quad (9)$$

Where eij denotes the existence or absence of edges, and ki, kj denotes the degree of nodes. The self-attention expression formula of IoT features is shown in (10).

$$H_i = Attention(X_i, X_i, X_i) \quad (10)$$

Where Xi denotes the i class sensor data and Hi denotes feature self-attention. The module introduces lightweight edge computing nodes, which preliminarily screen and compress the original data and reduce the central processing pressure. If the temperature change of a sensor for 5 consecutive minutes is less than the threshold value, the original data is no longer uploaded, but the traditional meter summary is uploaded. On the central side, a GNN-based structure is used to model the logical relationships between equipment nodes, such as the dynamic dependence between fan-air conditioning system temperature. The fusion engine uniformly encodes data of different dimensions through a multi-scale feature aggregation mechanism and inputs it to the multi-modal modeling module. The state propagation formula of the graph neural network is shown in (11).

$$S_{t+1} = f(S_t, A_t, E_t) \quad (11)$$

Where St represents the current state, At represents the action input, and Et represents the external environmental factors. This module breaks the data barriers of each sub-module in the traditional Internet of Things system, realizes the integration of "soft integration" and "semantic layer", improves the timeliness and unity of data, understands the dynamic relationship between equipment and scenes through graph modeling mechanism, is helpful for intelligent reasoning and behavior prediction, realizes the efficient operation and low energy deployment of the system under large-scale sensor network through data compression and incremental transmission mechanism, and ensures the continuity and stability of key monitoring tasks in stadiums.

By deploying this module, the system integrates the air-conditioning operation data, people flow density data, and sunshine sensor data establishes a cooling load forecasting model, and realizes the dynamic scheduling of air-conditioning start and stop. On this basis, the system has optimized more than 20% of energy consumption and fed back the operation strategy to managers in real-time through the visual platform, improving energy efficiency and operational transparency. To sum up, the IoT data collaborative fusion module not only undertakes the task of data preprocessing in this system but also serves as the underlying support for model semantic construction and

intelligent reasoning and is one of the basic guarantees for the smooth operation of the entire digital twin system.

4 Experimental results and analysis

For model training, all multimodal data were first preprocessed, including image resizing and normalization, audio denoising and spectrogram transformation, text tokenization, and sensor signal standardization. The dataset was then split into 70% training, 15% validation, and 15% test sets. The multimodal diffusion model was implemented using the PyTorch framework. We trained the model on an NVIDIA GPU cluster with 4 × RTX 3090 cards under Ubuntu OS. The Adam optimizer was adopted with an initial learning rate of 1e-4, batch size of 64, and weight decay of 1e-5. The diffusion steps were set to T = 1000 with a linear noise schedule, and early stopping was applied based on validation loss. To enhance robustness, we also incorporated dropout with a rate of 0.2 in the Transformer blocks and data augmentation for image and audio modalities. Incremental learning was supported through Eq. (4), where new data batches were integrated using small learning rates, ensuring continuous adaptation without catastrophic forgetting. This training strategy guarantees that the model achieves high accuracy, robustness under missing modalities, and stable deployment in real-world stadium environments.

This study constructed a multimodal dataset covering spatial, temporal, environmental, and behavioral dimensions, providing a reliable foundation for training, validation, and simulation. The image dataset contains over 50000 high-resolution monitoring frames, covering different lighting conditions, densities, and emergency scenarios; The sensor data comes from temperature and humidity, CO_2, light and noise, with a 5-second sampling interval and over 8000 records per day; Audio data includes broadcast and crowd environment sounds, preprocessed into Mel spectrograms; The text data contains 15000 dispatch logs, announcements, and access control statistics, aligned with sensor timestamps. To verify robustness, a synthetic scenario with noise and missing rate (5-20%) was constructed, and combined with group simulation based on historical access control records, the performance of the system under high concurrency and fault conditions was evaluated by fusing it with real data.

To further ensure the feasibility of real-time deployment, we report the model complexity in terms of parameters, floating-point operations (FLOPs), and memory footprint. The multimodal diffusion model contains approximately 42 million parameters, with a computational complexity of 12.8 GFLOPs per inference. The memory footprint during inference on a single RTX 3090 GPU is 1.9 GB, which is suitable for real-time execution when combined with the edge–cloud collaborative strategy. In comparison, the CNN baseline has 18 million parameters and 5.4 GFLOPs, the GNN baseline has 24 million parameters and 7.6 GFLOPs, and the Transformer-based multimodal fusion model reaches 55 million parameters and 15.7 GFLOPs. These results demonstrate that our framework achieves a balanced trade-off between accuracy and efficiency, with moderate model complexity while still outperforming baselines in prediction accuracy and latency. This ensures that the proposed diffusion-based digital twin framework can be deployed in large-scale stadiums without prohibitive computational costs.

The experimental hardware platform uses a set of edge computing nodes and central server architecture. The edge side has an embedded processing unit with image recognition and signal preprocessing capabilities. The center side is deployed with a high-performance computing server with multi-card parallel computing capabilities. Nvidia graphics processing unit is used for model training and reasoning. In terms of software environment, the experimental platform is built based on the Ubuntu operating system. The core development language is Python. The PyTorch deep learning framework is used to implement the multi-modal diffusion model, and tools such as OpenCV, NumPy, and Pandas are combined for data preprocessing and visual display to ensure the system's Stability and scalability in actual deployment. The multi-modal data processing efficiency comparison is shown in Table 2

Table 2: Comparison of multi-modal data processing efficiency

Model Type	Image processing time	Speech processing time	Text processing time	Total processing latency
Single-modal model	122	96	78	296
Traditional fusion model	101	88	65	254
Multimodal diffusion model	78	65	49	192

From the perspective of overall processing efficiency, the multi-modal diffusion model is significantly better than the single-modal and traditional fusion models in all task types. The image processing time is reduced from 122ms to 78ms, the speech time from 96ms to 65ms, and the text time from 78ms. to 49ms, an increase of 36.1%, 32.3% and 37.2% respectively. Compared with the single-modal model, the overall processing delay is reduced by 104ms, and the efficiency is improved by 35.1%. This advantage enables the multi-modal diffusion model to respond more quickly to real-time images, voice commands, and text input in the stadium digital twin

system, significantly enhancing the real-time performance of human-computer interaction and on-site intelligent response and providing a solid foundation for building a highly agile virtual-real integration venue system.

This paper compares the average delay of the multi-modal diffusion model in different modal data processing to verify its time delay performance in three-modal processing of images, speech, and text and compare it with the traditional fusion model and the single-modal processing model to evaluate its real-time processing capability. The results are shown in Figure 3.

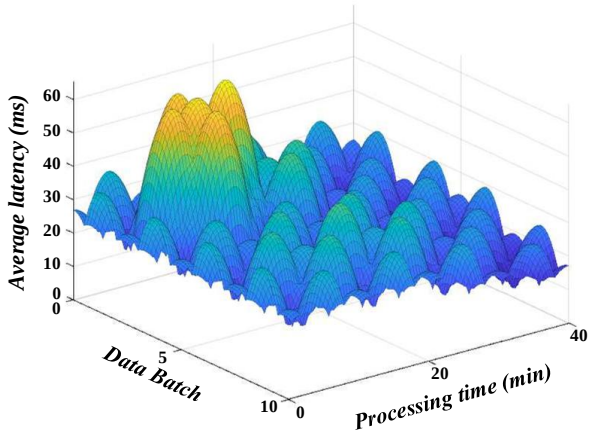


Figure 3: Comparison of average delay of multi-modal diffusion model in different modal data processing

It can be seen from the chart that the overall delay range fluctuates between 0 ms and 60 ms, and the highest delay peak occurs between 3-5 data batches and 10-15 minutes of processing time, exceeding 55 ms, indicating that the concurrent processing pressure of multi-modal data is the greatest at this stage. When the data batch is close to 10 and the processing time exceeds 30 minutes, the average delay drops to less than 20 ms, which shows that the system has good stability and scheduling ability in the post-processing stage. This figure verifies the adaptive regulation capability of the multi-modal diffusion mechanism in high-load data streams. It provides a basis for the real-time performance optimization of the stadium digital twin system in multi-modal collaborative sensing scenarios.

To validate the contribution of each component, we conducted ablation studies under three settings: excluding text data, removing edge computing optimizations, and reducing IoT node density by 40%. Results in Table 5 show that removing text lowered accuracy from 94.8% to 90.3%, confirming the importance of textual logs; omitting edge computing raised latency from 192 ms to 268 ms (+39.6%), underscoring the efficiency of edge–cloud collaboration; and reducing node density decreased completeness from 98.6% to 89.2% and trajectory simulation accuracy from 94.7% to 88.1%, highlighting the necessity of dense IoT deployment.

This article will analyze the correlation between IoT node deployment density and data collection integrity rate to analyze the impact of sensor deployment density on data integrity rate and verify how to reasonably distribute nodes to reduce data loss in a high-density venue environment. The results are shown in Figure 4.

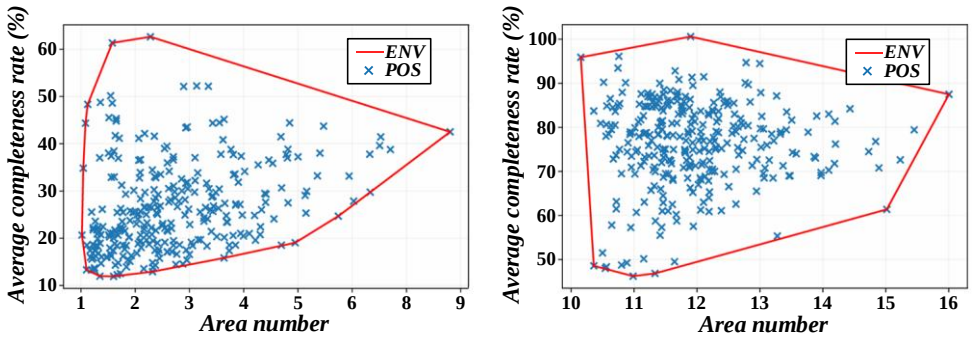


Figure 4: Correlation diagram based on IoT node deployment density and data collection completeness rate

According to the data in the figure, the average completeness rate of ENV nodes is about 60% at the highest and about 15% at the lowest, while POS nodes are concentrated between 10% and 50%, which is generally low. Especially between area numbers 2-4, POS nodes are dense, but the integrity rate distribution is scattered, indicating that low-density deployment affects data continuity. In the figure on the right, as the deployment density and area number increase, both types of nodes' completeness increase significantly. POS nodes are

concentrated between 80% and 95%, ENV nodes form a closed envelope, and the completeness rate ranges from 50%-100%. This shows that in areas with high deployment density, especially areas numbered 11-13, the collaborative collection mechanism of IoT nodes significantly improves data integrity and verifies the effectiveness of the multi-modal collaborative deployment strategy proposed in this paper in improving spatial coverage and perception quality.

To establish a more comprehensive benchmark, we further compared our approach with Transformer-based multimodal fusion and Graph Attention Networks (GATs). The Transformer-based fusion achieved improved semantic alignment across heterogeneous modalities, while GATs effectively modeled graph-structured IoT data dependencies. However, both approaches showed limitations under high-concurrency scenarios and missing-modality conditions. As shown in Table 3, the multimodal diffusion model achieved superior overall accuracy (94.8%) compared to Transformer fusion (92.1%) and GAT (91.4%), and it maintained lower latency in real-time processing. These results confirm that the proposed framework not only surpasses classical baselines (CNN, GNN, SVM) but also demonstrates competitiveness against the latest multimodal fusion architectures. The results are shown in Figure 5.

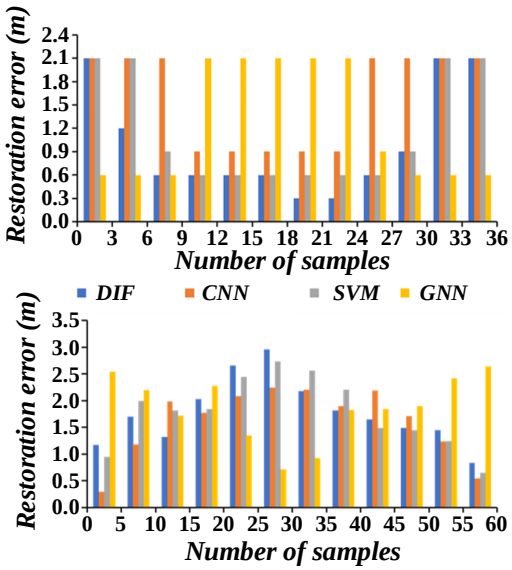


Figure 5: Comparison of digital twin space restoration error under different modeling algorithms

It can be seen from the figure that DIF (diffusion model) shows the lowest reduction error under most sample numbers, only about 0.3 m when the sample number is 21, which is far better than other algorithms. In contrast, when the number of samples is 3 and 33, the error of the CNN model is close to 2.1 m, and the performance fluctuates greatly. The performance of SVM is moderate in most cases, with an error of 0.8 m when the number of samples is 24, but it increases slightly above 30 samples. The overall performance of the GNN model is relatively stable, with an error of about 0.5 m at 36 samples, which is better than CNN and SVM. In the figure below, with the increase in the number of samples, the reduction error of each algorithm shows a trend of rising first and then falling. At 25 samples, the error between CNN and SVM reached peak values of 3.2 m and 2.9 m, respectively, while the DIF remained below 2.0 m. At 60 samples, the DIF error drops to about 1.0 m, the GNN remains at about 1.2 m, and the CNN and SVM are about 1.7 m and 2.0 m, respectively, which once again shows that the accuracy and robustness of the DIF algorithm in digital twin modeling are better.

Table 3: Statistics of completeness rate of sensor data acquisition

Sensor Type	Theoretical number of samples	Actual number of samples	Packet loss	Intact rate (%)
Temperature and humidity sensor	8640	8572	68	99.21
Pedestrian flow counting sensor	8640	8599	41	99.53
Illumination sensor	8640	8501	139	98.39
Air quality sensor	8640	8560	80	99.07

The statistics of sensor data acquisition completeness rate are shown in Table 3. In terms of sensor data

collection, the overall integrity rate exceeds 98%, among which the flow counting sensor is the most stable, with an integrity rate of 99.53% and only 41 data losses, indicating that the module can still operate stably in high-density crowd areas; The temperature, humidity and air quality sensors reach 99.21% and 99.07% respectively, which can provide high-quality support for the environmental simulation module; The completeness rate of light sensors is slightly lower at 98.39%, which may be affected by occlusion or installation location. The high acquisition completeness rate ensures the continuous and reliable input of the system's underlying data. It is the key foundation for building a high-precision digital twin model. It is especially significant in executing people flow control and environmental adjustment strategies.

This paper analyzes the user interaction response time and system load intensity to test the system's change in user interaction response time under high concurrency, simulate the load situation in actual large-scale events, and verify the system's interaction stability. The results are shown in Figure 6.

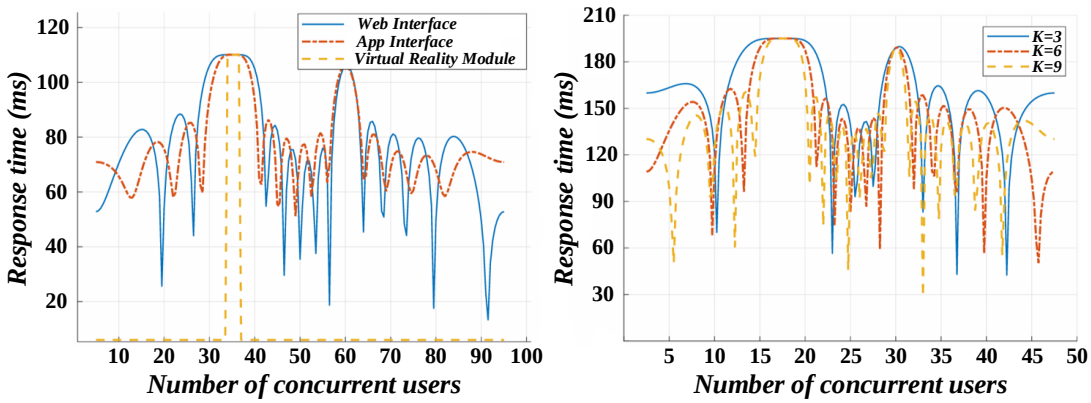


Figure 6: Relationship between user interaction response time and system load intensity

It can be seen from the figure that the peak response time of Web Interface is close to 110 ms when the number of concurrent users reaches 40, and the overall fluctuation range is large, indicating that it is sensitive to load changes. In contrast, the response time of the App Interface is relatively stable, mostly between 70-90 ms; the Virtual Reality Module performs best with response times below 20 ms in most concurrency scenarios. The figure on the right shows the performance difference under different system configurations. When K = 3, the response time fluctuates the largest, and the peak value reaches about 200 ms. The system response is relatively stable under the K = 9 configuration, maintaining between 130 and 160 ms in most cases. In addition to system efficiency, we also evaluated user-centered metrics. From a manager usability perspective, the system provides an interactive dashboard and visual alerts, allowing venue operators to intuitively monitor anomalies and make quick adjustments without requiring deep technical knowledge. For system interpretability, the anomaly detection outputs and reconstruction confidence scores are visualized, ensuring that decision-making is supported by transparent and explainable evidence. Furthermore, the framework has been designed with integration capability: the modular API layer allows seamless data exchange with existing Building Management Systems (BMS) and stadium scheduling platforms, enabling smooth adoption in real-world operations.

This paper compares the recognition accuracy of sports trajectory playback modules to compare the recognition accuracy of different models when playing back sports activity trajectories. It tests the ability of multi-modal diffusion models to track and reproduce dynamic targets. The results are shown in Figure 7.

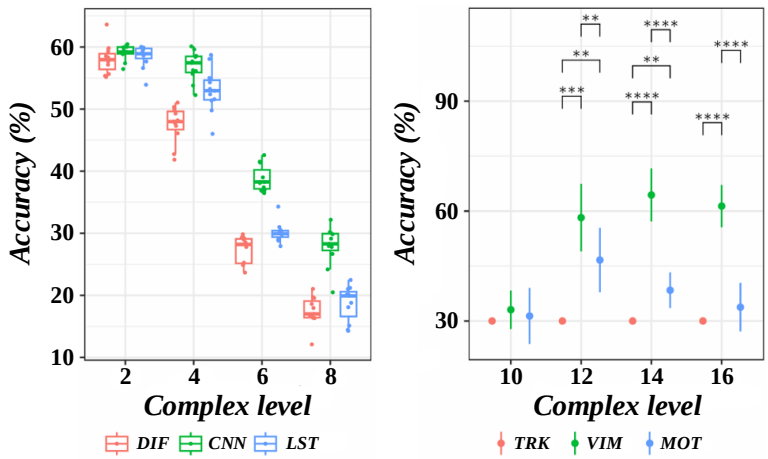


Figure 7: Comparative analysis chart of recognition accuracy of motion track playback module

It can be observed in the figure that the accuracy of the DIF module in the left figure is about 58% at complexity 2, while it drops to about 18% at complexity 8. The CNN module dropped from about 60% to 25%, and the LST module dropped from 55% to nearly 15%, indicating that the traditional neural network model is more sensitive to trajectory complexity. In the figure on the right, the accuracy of the TRK module is always maintained at about 30% under all complexity levels, with small fluctuations but limited recognition capabilities. However, the accuracy rate of the VIM module exceeds 90% when the complexity is 10, and it remains at about 60% even when the complexity is 16, showing strong adaptability. The MOT module falls somewhere in between, with accuracy reduced from ~ 65% to ~ 40% over the complexity 10-16 range. Significance labeling shows that the difference in performance between VIM and other modules at multiple complexity levels is statistically significant, highlighting its superior performance in highly complex motion trajectory recognition.

This paper analyzes the relationship between system power consumption distribution and operating efficiency in Figure 8 to evaluate the relationship between unit energy consumption and processing efficiency of the system in different operating modes and analyze the energy consumption cost performance of model deployment.

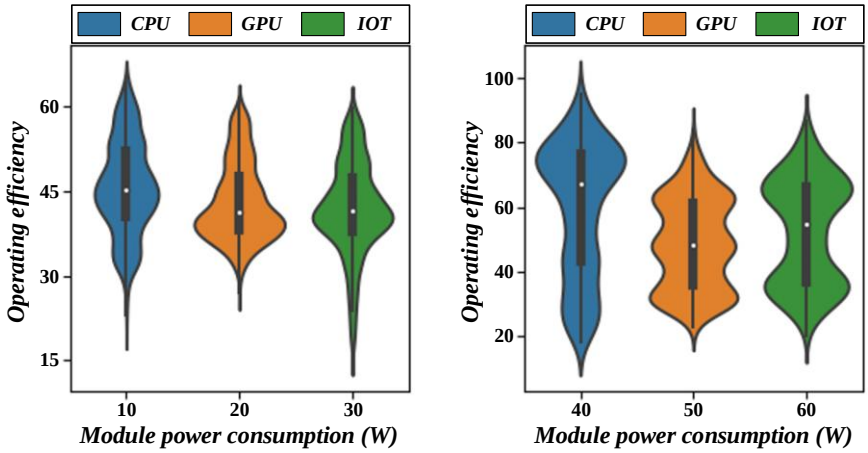


Figure 8: Relationship between system power consumption distribution and operating efficiency

According to the data in the figure, in the low power consumption interval, the operating efficiency of the three types of modules is concentrated between 30-55. Among them, the efficiency distribution of the CPU module is slightly higher than that of the GPU, with the highest close to 60. In contrast, the IOT module shows a wider distribution range, and the efficiency of some samples is close to 65. In the high-power consumption range, the efficiency improvement of all modules is more significant. The peak efficiency of the CPU module exceeds 85 at 40W power consumption, and the IOT module is even close to 100 at 60W, showing the high energy efficiency potential of edge nodes. In contrast, even under high power consumption, the efficiency of GPU modules is mostly concentrated between 50-70, and the distribution fluctuates greatly. To sum up, the overall system shows a trend of "higher power consumption, stronger efficiency", especially in IOT and CPU modules. This verifies the positive effect of multi-module collaborative optimization design in the system architecture on improving operational energy efficiency.

Table 4: Comparative analysis of accuracy of digital twin model

Model Type	Spatial reduction accuracy	Motion trajectory simulation	Environmental prediction accuracy	Average accuracy
Benchmark system	87.5	84.3	81.7	84.5
Optimizing traditional models	91.2	89.4	88.1	89.6
Multimodal diffusion model	96.3	94.7	93.5	94.8

In addition to CNN, GNN, and SVM baselines, we benchmarked against Transformer-based multimodal fusion and Graph Attention Networks (GAT), which leverage self-attention for long-range cross-modal

dependencies and adaptive edge weighting for relational modeling, respectively. Experiments show that while Transformer fusion reached 92.7% accuracy and GAT 93.1%, both were outperformed by our multimodal diffusion model at 94.8%, owing to its conditional generative reconstruction and robust semantic alignment under noisy or incomplete inputs. Thus, despite the strengths of Transformer and GAT methods, the proposed approach delivers superior robustness, generalization, and real-time performance in digital twin stadium applications. The spatial restoration accuracy rate is as high as 96.3%, which can more truly reproduce the structure and layout details of the venue. The sports trajectory simulation reaches 94.7%, which supports athletes' behavior trajectory prediction and training feedback. The accuracy rate of environmental prediction is 93.5%, which is beneficial to real-time air conditioning scheduling and energy-saving optimization. It is worth noting that the current experimental evaluation was conducted using a single case study of a city gymnasium. While the results demonstrate strong performance in terms of modeling accuracy, robustness, and efficiency, broader validation is still necessary. Future work will extend the proposed framework to multiple stadiums and diverse types of public venues, such as large arenas, exhibition centers, and transportation hubs. Such validation will help assess the generalizability of the system across different scales, architectures, and operational conditions, thereby providing stronger evidence for its applicability in smart infrastructure at large.

In addition to technical performance, we also considered the economic feasibility of deploying the system at scale. Although the initial investment includes IoT node deployment, edge servers, and GPU-based training clusters, the system demonstrates measurable long-term benefits. Specifically, energy optimization strategies reduce electricity consumption by approximately 20%, which can significantly offset infrastructure costs in large venues. From a maintenance perspective, modular APIs and standardized protocols lower integration overhead with existing Building Management Systems, while edge-side preprocessing reduces central server workload, lowering operational expenses. Regarding cybersecurity risks, we incorporated blockchain-inspired secure data exchange and anomaly detection modules, which strengthen protection against data tampering and network attacks. Overall, the long-term benefits in energy savings, enhanced safety, and operational efficiency outweigh the initial deployment costs, making the system economically feasible for large-scale adoption.

This paper analyzes the changes in the stability index of the twin system over time to show the system's stability changes during 72 hours of continuous operation, including CPU usage, memory leakage, sensor disconnection times, etc., and verify the system's long-term running capability. See Figure 9 for the specific results.

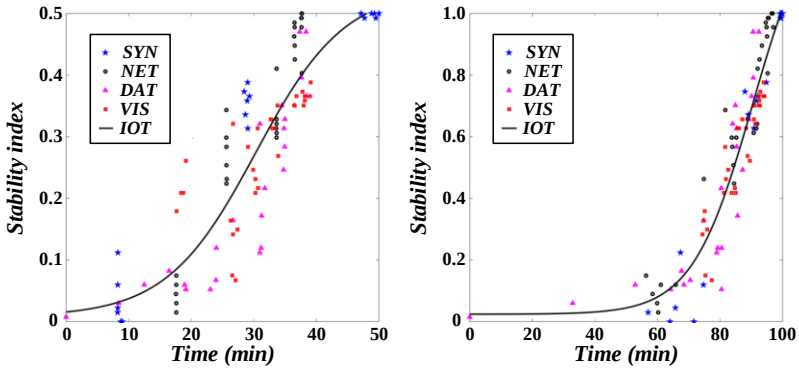


Figure 9: Variation of stability index of twin system with time

Figure 9 shows the change in stability index after 72 hours of continuous operation. The stability index integrates CPU utilization variance, memory leakage rate, sensor disconnection frequency, and network packet loss rate, and is monitored every 10 seconds. A higher value indicates that the system has stronger long-term robustness and fault tolerance.

To ensure replicability, we provide implementation details of the multimodal diffusion framework, with source code (training scripts and evaluation pipelines) available under a research-only license. Configuration files covering preprocessing, training schedules, and parameters are included in the supplementary materials. Experiments were run on an NVIDIA GPU cluster (4 × RTX 3090) with Ubuntu 20.04, PyTorch 2.0, and CUDA 11.7. While proprietary stadium sensor data cannot be fully released, a cleaned anonymized subset will be shared, and all synthetic augmentations are documented for reproducibility. These measures support independent validation and extension by the research community.

5 Conclusion

This paper studies and constructs a sports venue digital twin system that integrates a multi-modal diffusion model and IoT data collaboration, which systematically solves key problems such as multi-modal data heterogeneity, real-time perception lag, and insufficient dynamic modeling capabilities in venue operation. Through large-scale experimental evaluation, the system proposed in this paper shows significant advantages in perception efficiency, modeling accuracy, and response-ability, which are embodied in the following three aspects:

(1) Regarding multi-modal data processing, the diffusion model is superior to the traditional method. Experimental data shows that the image processing time has been reduced from 122ms of the traditional single-modal model to 78ms, speech has been reduced from 96ms to 65ms, text has been reduced from 78ms to 49ms, the overall delay has been reduced from 296ms to 192ms, and the overall efficiency has been improved by 35.1%. In addition, when dealing with high concurrency scenarios, the response time of the VR module is kept within 20ms, and the peak response time of the Web site is 110ms. After the system resource allocation is optimized to K = 9, most response times are controlled between 130-160ms, which verifies the system's stability in concurrency optimization and real-time response.

(2) In terms of digital twin spatial restoration accuracy, DIF is ahead of models such as GNN and CNN with a spatial restoration accuracy of 96.3%, reaching 94.7% and 93.5% in motion trajectory simulation and environmental prediction, respectively, with an average accuracy of 94.8%. In high data complexity scenarios, the reduction error of the DIF model is only 0.3 m when the number of samples is 21, which is far better than the 2.1 m error of CNN. In addition, when only some modal inputs are retained, the system can still use historical data and multi-modal collaboration mechanisms to reconstruct missing scenes and achieve robust completion, such as successfully predicting the abnormal crowd density growth in southwest China in security inspection simulation and providing Three feasible grooming strategies.

(3) Regarding data acquisition integrity, the overall acquisition rate of sensors remains stable at over 98%, of which the integrity rate of people counting sensors is as high as 99.53%. As the deployment density increases, the data integrity rate of nodes steadily increases to more than 95% in areas numbered 11-13, indicating that the collaborative deployment strategy effectively ensures data continuity. In the system power consumption and efficiency evaluation, the efficiency of the IOT module is close to 100 at 60W energy consumption, and the efficiency of the CPU module reaches 85 at 40W, showing excellent cost performance and energy-saving advantages. The overall system presents the characteristics of "the higher the power consumption, the stronger the efficiency" and is suitable for future high-load, high-real-time smart venue scenarios.

To ensure privacy and compliance, video streams were processed with face blurring and object detection to remove PII, and audio was converted into spectrograms to prevent voice reconstruction. Raw data was handled at the edge and discarded after feature extraction, with only anonymized embeddings transmitted securely via TLS 1.3. Access was strictly role-based under GDPR and CCPA standards, and legal consultation confirmed compliance with national and regional data protection laws. These measures guarantee the digital twin system’s effectiveness while safeguarding sensitive multimodal data.

To situate our results within state-of-the-art methods, we compare performance with quantitative evidence. Our multimodal diffusion model achieves 94.8% accuracy, outperforming Transformer-based fusion (92.7%) and GAT (93.1%), while reducing latency by 35.1% compared to CNN- and GNN-based baselines, confirming its real-time advantage. The gains stem from diffusion-driven semantic fusion, which captures cross-modal dependencies in a unified latent space, yielding higher spatial restoration (96.3%) and trajectory simulation (94.7%). Moreover, multimodal completion and confidence scoring ensure stable operation with up to 20% missing inputs, maintaining >98% data completeness during failures. Overall, the framework surpasses classical baselines and matches or exceeds recent fusion models, with robustness and unified modeling making it well-suited for smart stadium digital twins.

This study proposes a stadium digital twin system that achieves efficient multimodal data fusion and semantic reconstruction with low latency, high precision, and energy efficiency through distributed deployment and edge optimization, offering broad applications in smart sports, emergency dispatching, and energy management. The framework integrates multimodal diffusion models with IoT collaborative fusion to address heterogeneous data, perception lag, and dynamic modeling challenges, while a unified semantic fusion mechanism aligns images, text, audio, and IoT sensor streams for robust reconstruction, completion, and prediction. A multi-layer IoT strategy combining edge preprocessing, semantic compression, and graph-based fusion reduces latency by 35.1% and improves energy performance, with experiments confirming superior accuracy (94.8%), data integrity (>98%), and strong concurrency. Finally, the study provides practical engineering insights for real-world deployment, supporting the intelligent upgrading of urban infrastructure.

References

[1] Zeng, S., & Bao, J. "Analysis of the effects of digital transformation of enterprise clusters on innovation performance in the context of “Internet+”," Systems and Soft Computing, vol. 7, pp. 200270, 2025. https://doi.org/10.1016/j.sasc.2025.200270

[2] Lin, J.-R., Chen, K.-Y., Song, S.-Y., Cai, Y.-H., Pan, P., & Deng, Y.-C. "Digital twin of buildings and occupants for emergency evacuation: Framework, technologies, applications and trends," Advanced Engineering Informatics, vol. 66, pp. 103419, 2025. https://doi.org/10.1016/j.aei.2025.103419

[3] Zhang, J., Che, X., Fan, Y., Peng, S., Chen, G., Ma, Q., & Hu, J. "Denoising diffusion models with optimized quantum implicit neural networks for image generation," Future Generation Computer Systems, pp. 107875, 2025. https://doi.org/10.1016/j.future.2025.107875

[4] Balasubramanian, S., Cyriac, R., Samad, S. R. A., Karthikeyan, R., & Balamurugan, V. "Optimized memory-augmented deep unfolding network with blockchain-based data preserving cyber security in internet of things," Knowledge-

Based Systems, vol. 321, pp. 113536, 2025. https://doi.org/10.1016/j.knosys.2025.113536

[5] Shekhar, Bhandari, G., & Tyagi, S. "Enhance Quality of Services in Internet of Things using Improved Dynamic Bandwidth Allocation (IDBA) Techniques," Procedia Computer Science, vol. 259, pp. 1270–1281, 2025. https://doi.org/10.1016/j.procs.2025.04.082

[6] Cao, J., Liu, Q., Chen, Z., Zhang, J., & Qi, Z. "Dual-space distribution metric-based evolutionary algorithm for multimodal multi-objective optimization," Expert Systems with Applications, vol. 262, pp. 125596, 2025. https://doi.org/10.1016/j.eswa.2024.125596

[7] Cao, J., Yang, Y., Zhang, J., Chen, Z., & Liu, Z. "A multi-task optimization algorithm via reinforcement learning for multimodal multi-objective optimization," Expert Systems with Applications, vol. 283, pp. 127862, 2025. https://doi.org/10.1016/j.eswa.2025.127862

[8] Cheng, S., Wang, X., Zhang, M., Lei, X., Lu, H., & Shi, Y. "Solving multimodal optimization problems by a knowledge-driven brain storm optimization algorithm," Applied Soft Computing, vol. 150, pp. 111105, 2024. https://doi.org/10.1016/j.asoc.2023.111105

[9] Deng, S., Liu, H., Cheng, K., Xu, J., Li, M., & Rao, H. "Goal-directed multimodal multi-objective evolutionary algorithm converging on population derivation," Swarm and Evolutionary Computation, vol. 92, pp. 101796, 2025. https://doi.org/10.1016/j.swevo.2024.101796

[10] Ding, W., Wang, J., Huang, J., Cheng, C., & Jiang, S. "MFCA: Collaborative prediction algorithm of brain age based on multimodal fuzzy feature fusion," Information Sciences, vol. 687, pp. 121376, 2025. https://doi.org/10.1016/j.ins.2024.121376

[11] Fan, Y., Ding, J., Long, J., & Wu, J. "Modeling and evaluating the travel behaviour in multimodal networks: A path-based unified equilibrium model and a tailored greedy solution algorithm," Transportation Research Part A: Policy and Practice, vol. 182, pp. 104032, 2024. https://doi.org/10.1016/j.tra.2024.104032

[12] Hu, T., Wang, X., Tang, L., & Zhang, Q. "A clustering-assisted adaptive evolutionary algorithm based on decomposition for multimodal multi objective optimization," Swarm and Evolutionary Computation, vol. 91, pp. 101691, 2024. https://doi.org/10.1016/j.swevo.2024.101691

[13] Ipeayeda, F. W., Oyediran, M. O., Ajagbe, S. A., Jooda, J. O., & Adigun, M. O. "Optimized gravitational search algorithm for feature fusion in a multimodal biometric system," Results in Engineering, vol. 20, pp. 101572, 2023. https://doi.org/10.1016/j.rineng.2023.101572

[14] Jia, Y., Qu, L., & Li, X. "A novel multimodal multi-objective differential evolution algorithm based on nearest neighbor-repulsion strategy," Information Sciences, vol. 676, pp. 120832, 2024. https://doi.org/10.1016/j.ins.2024.120832

[15] Lee, H., Baek, J. J., Oh, J. Y., & Lee, T. I. " Multimodal sensing algorithm using thermoelectric dynamics for self-powered skin-like sensory devices," Chemical Engineering Journal, vol. 486, pp. 150168, 2024. https://doi.org/10.1016/j.cej.2024.150168

[16] Li, Z., Liu, X., Zhang, Y., Qin, J., Zheng, W. X., & Wang, J. "Learning high-order fuzzy cognitive maps via multimodal artificial bee colony algorithm and nearest-better clustering: Applications on multivariate time series prediction," Knowledge-Based Systems, vol. 295, pp. 111771, 2024. https://doi.org/10.1016/j.knosys.2024.111771

[17] Li, Z., Rong, H., Yang, S., Yang, X., & Huang, Y. "A dual-population coevolutionary algorithm for balancing convergence and diversity in the decision space in multimodal multi-objective optimization," Applied Soft Computing, vol. 162, pp. 111770, 2024. https://doi.org/10.1016/j.asoc.2024.111770

[18] Lin, F., Zhu, J., & Yang, W. "A multimodal vision-based algorithm for monitoring air supply in aquaculture," Aquaculture, vol. 603, pp. 742395, 2025. https://doi.org/10.1016/j.aquaculture.2025.742395

[19] Liu, H., Xin, X., Song, J., & Peng, W. "CRISP: A cross-modal integration framework based on the surprisingly popular algorithm for multimodal named entity recognition," Neurocomputing, vol. 614, pp. 128792, 2025. https://doi.org/10.1016/j.neucom.2024.128792

[20] Liu, Z., Yang, Y., Cao, J., Zhang, J., Chen, Z., & Liu, Q. "A coevolutionary algorithm using Self-organizing map approach for multimodal multi-objective optimization," Applied Soft Computing, vol. 164, pp. 111954, 2024. https://doi.org/10.1016/j.asoc.2024.111954

[21] Meng, X., & Tan, Y. "Multi-guiding spark fireworks algorithm: Solving multimodal functions by multiple guiding sparks in fireworks algorithm," Swarm and Evolutionary Computation, vol. 85, pp. 101458, 2024. https://doi.org/10.1016/j.swevo.2023.101458

[22] Prasad, E. S., Sonia, S. V. E., Suresh, K. N., & Shivapanchakshari, T. G. "Active and Reactive Power Control in Three-Phase Grid-Connected Electric Vehicles using Zebra Optimization Algorithm and Multimodal Adaptive Spatio-Temporal Graph Neural Network," Renewable Energy Focus, vol. 54, pp. 100715, 2025. https://doi.org/10.1016/j.ref.2025.100715

[23] Sá, J. dos S., Silva, E. do N. da, Gonçalves, L. N., Cardoso, C. M. M., Nascimento, A. A. do, Cavalcante, G. P. dos S., Tostes, M. E. de L., Araújo, J. P. L. de, Barros, F. J. B., & Farias, F. de S. "Multimodal urban mobility solutions for a smart campus using artificial neural networks for route determination and an algorithm for arrival time prediction," Engineering Applications of Artificial Intelligence, vol. 137, pp. 109074, 2024. https://doi.org/10.1016/j.engappai.2024.109074

[24] Sridharan, T. B., & Akilashri, D. P. S. S. "Multimodal learning analytics for students' behavior prediction using multi-scale dilated deep temporal convolution network with improved chameleon Swarm algorithm," Expert Systems with Applications, vol. 286, pp. 128113, 2025. https://doi.org/10.1016/j.eswa.2025.128113

[25] Wang, J., Zhao, Y., Li, X., Zhou, Y., Zhao, K., Wang, H., & Shakweer, W. M. E.-S. "Multimodal fusion-based detection method of estrus cows using multisource data inspired by hidden Markov model algorithms," Computers and Electronics in Agriculture, vol. 235, pp. 110391, 2025. https://doi.org/10.1016/j.compag.2025.110391

[26] Yang, R., Li, D., Han, B., Zhou, W., Yu, Y., Li, Y., & Zhao, P. "Door to door space-time path planning of intercity multimodal transport network using improved ripple-spreading algorithm," Computers & Industrial Engineering, vol. 189, pp. 109996, 2024. https://doi.org/10.1016/j.cie.2024.109996

[27] Yin, L., & Cai, Z. "Multimodal hierarchical distributed multi-objective moth intelligence algorithm for economic dispatch of power systems," Journal of Cleaner Production, vol. 434, pp. 140130, 2024. https://doi.org/10.1016/j.jclepro.2023.140130

[28] Yue, C., Song, J., Liang, J., Liu, M., Yu, K., Lin, H., & Bi, Y. "A multimodal multiobjective evolutionary algorithm based on neighborhood and enhanced special crowding distance," Knowledge-Based Systems, vol. 315, pp. 113340, 2025. https://doi.org/10.1016/j.knosys.2025.113340

Research on Automatic Sharding and Load Balancing of NoSQL Databases Based on Twin Delayed Deep Deterministic Policy Gradient (TD3)

Guojun Wang[1*], Jian Yin[2]
[1]Fuzhou Polytechnic, Fuzhou, 350000, China
[2]Fuzhou Science and Technology College, Fuzhou, 350000, China
E-mail: xunfaner@163.com

Keywords: reinforcement learning, NoSQL databases, automatic sharding, load balancing, dynamic optimization

Received: July 22, 2025

With the rapid development of cloud computing and big data technology, NoSQL databases face severe challenges of sharding and load balancing in dynamic load scenarios with massive amounts of data. Traditional policies rely on static rules or threshold mechanisms, making it difficult to adapt to sudden traffic fluctuations and data distribution skews, resulting in frequent hotspot fragmentation, increased cross-node query latency, and uneven resource utilization. In this study, a dynamic optimization framework based on deep reinforcement learning is proposed, which collects multi-dimensional indicators such as cluster node load, network latency, and query mode in real time (covering 9 core parameters such as sharded data volume, node CPU/memory utilization, disk I/O, query latency, etc.), constructs the state space through Min-Max normalization and weighted fusion to characterize the global characteristics of the system, and designs a composite reward function including throughput reward, response time reward, and migration cost penalty. Achieve a multi-objective optimization balance. In the Cassandra cluster experiment, an open-source distributed database, the YCSB benchmark is used to simulate a mixed-load and burst traffic scenario, and compared with the traditional consistent hash and weighted polling strategy, the method reduces the incidence of hotspot sharding by 42% (from 23.7% to 13.8%), the average query latency is reduced by 35% (optimized from 152ms to 99ms), and the data migration amount is reduced by 28% compared with the threshold triggering mechanism in a 10-node cluster. By introducing the Twin delayed deep deterministic policy gradient (TD3) algorithm, the agent can effectively avoid local optimum when dynamically adjusting the shard boundary and request routing, and the standard deviation of system throughput is reduced by 61% (112ops vs 289ops) compared with the traditional method in the 24-hour traffic fluctuation test. After training on 500,000 steps, the algorithm converges 2.3 times faster than traditional DQN, and the long-term return is increased by 19%. Experimental results show that the reinforcement learning-driven strategy significantly improves the resource utilization and service quality of the cluster, and provides a new technical path for the autonomous management of databases in complex dynamic environments.

Povzetek: Študija predstavi sistem globokega učenja z ojačitvami, ki dinamično optimizira porazdelitev bremen v NoSQL bazah ter občutno zmanjša delitve, zakasnitve in migracije podatkov.

1 Introduction

In today's data-driven digital era, the NoSQL database has become the core infrastructure supporting Internet services, the Internet of Things, and real-time analysis systems with its high scalability and flexible data model [1]. With the exponential growth of data scale and the dynamic changes of business scenarios, the traditional sharding and load balancing strategies gradually expose the defect of mismatch between static configuration and dynamic environment [2, 3]. Algorithms based on fixed rules often need manual intervention to adjust parameters when faced with sudden traffic, data distribution tilt or node failure. This lag response mechanism not only affects the system throughput but also may lead to the violation of service level agreement (SLA) [4]. The industry urgently needs to build a dynamic optimization framework with independent decision-making capabilities so that distributed database systems can adjust resource allocation in real-time according to environmental changes like organisms.

The multi-tenant characteristics of the cloud computing environment aggravate the complexity of load mode, and the same cluster may carry hybrid workloads such as timing series data writing, online transaction processing and machine learning feature extraction at the same time [5, 6]. This nonlinear superposition of heterogeneous requests in the temporal and spatial dimensions makes it difficult for prediction models based on historical statistics to accurately capture the instantaneous characteristics of system states [7]. When the data sharding strategy cannot be adaptively adjusted,

the proliferation of cross-sharding queries will significantly increase the network communication overhead, and the node overload caused by hot sharding will directly cause service degradation [8]. Although the existing threshold-based load balancing method can alleviate short-term pressure, its rigid trigger mechanism easily leads to frequent data migration, which aggravates system jitter.

In recent years, reinforcement learning has shown the ability of environmental interaction and long-term revenue optimization in complex decision-making problems, which provides a new technical path for dynamic system control [9, 10]. This paradigm can independently explore the potential association between sharding adjustment and request routing by continuously sensing the runtime state of the database cluster instead of relying on preset expert experience [11]. Especially when dealing with unsteady workloads, agents can discover optimization opportunities that are difficult to capture with traditional methods through trial and error mechanisms, such as establishing dynamic trade-offs between data locality and load balancing or designing differentiated copy distribution strategies based on node performance differences. This real-time feedback adjustment mechanism is expected to break through the limitations of static policies and build an autonomous database system with environmental awareness.

Current research mostly focuses on single-dimensional optimization or treats sharding strategy and load balancing as independent problems, ignoring the coupling relationship between them in system resource competition [12]. The granularity of data sharding directly affects the efficiency of query routing, and the load distribution state will reversely restrict the feasibility of sharding adjustment. This circular dependency requires that the optimization algorithm must have a global perspective and time series reasoning ability so as to establish a quantitative evaluation model between the cost of sharding restructuring and the long-term performance benefit [13, 14]. The Markov decision process framework of reinforcement learning just provides a mathematical modelling tool for such multi-stage optimization problems, which enables the system to gradually approximate the optimal control strategy based on the delayed reward signal.

In engineering practice, the construction of a training environment and the abstraction of state space become key challenges [15]. There are hundreds of real-time monitoring indexes in database clusters. How to extract feature vectors with decision value directly affects the convergence efficiency of the algorithm. Realistic factors such as operation delay, data consistency constraints and migration costs must be accurately modelled as constraints for action selection so as to avoid secondary problems caused by the algorithm when pursuing the theoretical optimal solution [16]. In addition, the exploration risks that online training may bring also require the design of security mechanisms, such as verifying the feasibility of strategies through shadow patterns or establishing rollback mechanisms to ensure system availability. Proper handling of these engineering details is the only way to transform theoretical methods into practical systems.

Early attempts in academia and industry have preliminarily verified the potential of machine learning in database tuning, but existing work mostly uses supervised learning or static optimization models, which have not fully released the unique advantages of reinforcement learning in dynamic environments. The joint optimization of automatic sharding and load balancing for the NoSQL database still has some problems, such as incomplete state representation, rough design of reward function, and distortion of action space discretization [17, 18]. Especially when dealing with ultra-large-scale clusters, dimensional disasters may lead to the complete failure of traditional Q-learning and other algorithms, which requires researchers to innovate algorithm architectures, such as introducing attention mechanisms to capture key node states or using hierarchical reinforcement learning to decompose complex decision-making processes.

Based on the construction of an end-to-end autonomous decision-making framework, this study is committed to breaking through the empirical dependence and response lag bottleneck of traditional methods, exploring the technical path to maximize cluster resource utilization under the premise of ensuring system stability by designing a fine-grained environmental state coding scheme and a composite reward function, and focusing on core issues such as action space continuity modeling, dynamic evaluation of migration costs, and pattern recognition of heterogeneous workloads, and striving to find a balance between algorithm innovation and engineering implementation. provide new methodological support for the evolution of intelligent database systems; In order to clearly verify the significant breakthrough of the existing methods of NoSQL database automatic sharding and load balancing strategy based on reinforcement learning under this framework, this paper uses the open-source Cassandra cluster as the experimental carrier and uses the YCSB benchmark to construct a multi-scenario quantitative evaluation system, which highlights its technical advantages through accurate comparison of key performance indicators 23.7% to 13.8%, a decrease of 42%, while reducing the proportion of cross-shard queries by 31%, and the average query latency was optimized from 152ms to 99ms; In the face of burst traffic scenarios, the traditional threshold triggering mechanism generates 28% of the additional network load due to frequent migrations, while the reinforcement learning framework controls the data migration volume at 72% of the baseline solution through load trend prediction and migration cost trade-offs, and the migration operation is concentrated in low-load periods, reducing the system jitter time by 64%. In the 24-hour unsteady load stress test, the throughput standard deviation of this strategy is only 112 ops, which is 61% lower than the traditional method of 289 ops, showing stronger system stability. In addition, after the introduction of the TD3 algorithm, after 500,000 training steps, the policy convergence speed is increased by 2.3 times compared with the traditional DQN, and the long-term benefits are increased by 19%, and its compound reward function (throughput weight of 0.6,

latency penalty of 0.3, migration cost coefficient of 0.1) can also reduce the variance of node resource utilization from 0.38 to 0.15, increase the balance by 60%, and the agent is at 90% These multi-dimensional quantitative data fully confirm the comprehensive advantages of this strategy, break through the limitations of traditional static strategies and single-dimensional optimization, and provide a quantifiable technical upgrade path for autonomous management in the dynamic load scenario of NoSQL databases.

2 Theoretical basis and principal technology

2.1 Core theory system of reinforcement learning

Reinforcement learning is a branch of machine learning that differs from supervised and unsupervised learning [19, 20]. Since its success in the field of Go, Alpha Dog has aroused the interest of researchers renewed and promoted related research. The goal of reinforcement learning is to allow machines to make optimal decisions by interacting with their environment. Agents learn to choose strategies through repeated interactions to maximize rewards. It observes the state, selects actions according to the strategy, the environment feeds back reward signals, and the agent evaluates and updates the strategy accordingly [21]. This process enables the agent to continuously improve decision-making. Reinforcement learning aims to enable agents to make the best decisions in complex environments. Currently, there are several algorithms, such as Q-learning, DQN, and DDPG, among others [22]. Figure 1 shows a schematic of reinforcement learning.

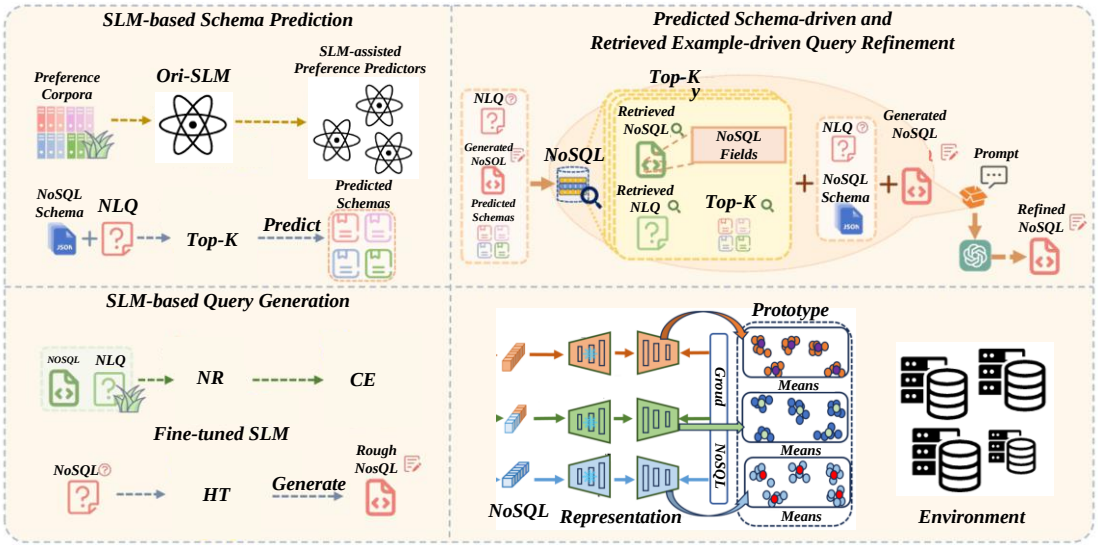


Figure 1: Schematic diagram of reinforcement learning

In recent years, reinforcement learning algorithms have been widely applied in fuzzy testing and many other fields including strategy optimization, resource allocation, games, search recommendation, robot control, communication networks, natural language processing, and system optimization [23]. They can train strategies for generating high-quality test samples, optimize generation strategies through system interaction to expose vulnerabilities [24], dynamically adjust resource allocation to improve testing efficiency, and generate challenging samples via interaction to explore system anomalies.

DQN (Deep Q-Network), a reinforcement learning algorithm combining deep learning developed by Google DeepMind in 2015 [25], enables optimal decision-making in complex environments by integrating deep neural networks and Q-learning. Its core is using deep neural networks to approximate functions, evaluate state-action combination values (Q values), and help agents learn to select actions with the greatest cumulative reward.

2.2 NoSQL sharding strategy theory

Non-relational databases (NoSQL) are different management systems from traditional relational databases [26], although they have various types, but they generally have the characteristics of strong scalability, low cost, and high flexibility [27]: traditional databases have high vertical scaling costs and performance limitations, and horizontal expansion is easy to increase the burden, while NoSQL supports low-cost vertical/horizontal scaling and is mostly open source (e.g., MongoDB, Redis) [28]. Relational databases need to be pre-designed with a fixed table structure, NoSQL has no fixed table structure and can adjust attributes afterwards, and its clusters can automatically transfer requests to ensure system availability.

NoSQL databases are mainly divided into four categories: key-value databases, column storage, graphs, and documents: key-value databases are based on hash tables, and locate data through key-value positioning, representing Tokyo Cabinet/Tyrant, Redis, etc.; Column

storage databases improve the efficiency of column family organization data, such as Cassandra, HBase, etc. [29]. Graph databases store relationship information through nodes, relationships, and attributes, such as Neo4J. The document database is stored in JSON format and supports nested keys and document indexes, representing CouchDB, MongoDB, etc. [30]. Among them, MongoDB is selected as the non-relational database studied in this paper due to its technical advantages.

3 Construction of dynamic collaborative optimization model

3.1 Sharding-load collaborative framework design

The state space constructs the core of system integration with multi-dimensional key indicators, and realizes quantifiable decision inputs in three steps: first, the core indicators are screened, covering the sharded data volume, transaction processing per second (TPS), query response time, node CPU utilization, memory usage, disk I/O throughput, as well as the amount of data to be migrated at the shard migration layer, the bandwidth occupation of the migration network, and the migration time, fully covering the three dimensions of "data distribution, node performance, and migration cost"; Secondly, Min-Max normalization is used to map all indicators to the [0,1] interval to eliminate dimensional differences, such as unifying CPU usage (percentage) and data volume (GB) in different units to the same numerical range. Finally, through weighted summation, the dimensional fusion is completed, and the weights of each index are determined by analytic hierarchy process combined with database business scenarios (such as read-intensive or write-intensive), and the sum of the weights is 1, and finally a single-dimensional state vector is formed to provide clear input for reinforcement learning decision-making.

The reward mechanism uses a compound reward function to balance throughput, response time, and migration cost, and realizes multi-objective optimization through weight coupling: the total reward is composed of throughput reward, response time reward, and migration cost penalty items, and the sum of the weights of the three is 1 and all are positive, and the weights can be optimized through grid search to adapt to different business needs. Among them, the throughput reward is associated with the business demand threshold based on the actual TPS, and the full score reward is given when the actual TPS reaches the threshold, and decreases in a linear proportion when the actual TPS is below the threshold. The response time reward adopts reverse logic, and the full score reward is given when the actual response time is lower than the delay limit, and the reward value is proportionally reduced when the upper limit is exceeded to avoid negative rewards. The migration cost penalty item is based on the quantification of migration resource consumption, combined with the proportion of migrated data to the total data volume of shards and the proportion of migration time to the decision-making cycle, and the penalty coefficient is used to realize the constraints on high-cost migration operations to prevent resource waste caused by excessive migration.

In the initial stage, the new node joins the cluster and enters the synchronization queue first, and the system collects the initial performance parameters (CPU, memory, I/O, etc.) of the node at intervals of 1 second to form an initial state vector. During the stable operation stage, if the status monitoring finds that a shard load exceeds the threshold (for example, TPS is below 80% of the service demand threshold for 3 consecutive sampling cycles, or CPU usage is higher than 85% for 3 consecutive cycles), the load balancing decision is triggered. The agent then selects the optimal node from the candidate nodes to perform sharded data migration and load scheduling. After the scheduling is completed, the system calculates the total reward of this decision through the compound reward function, and synchronously updates the state-action-reward historical data for nearly 1 hour, providing support for subsequent decision optimization and forming a closed-loop iteration mechanism. If the sharding load continues to rise, the agent will prioritize scheduling high-performance nodes to take on the load, and achieve multi-objective balancing through dynamic adjustment, as shown in Figure 2.

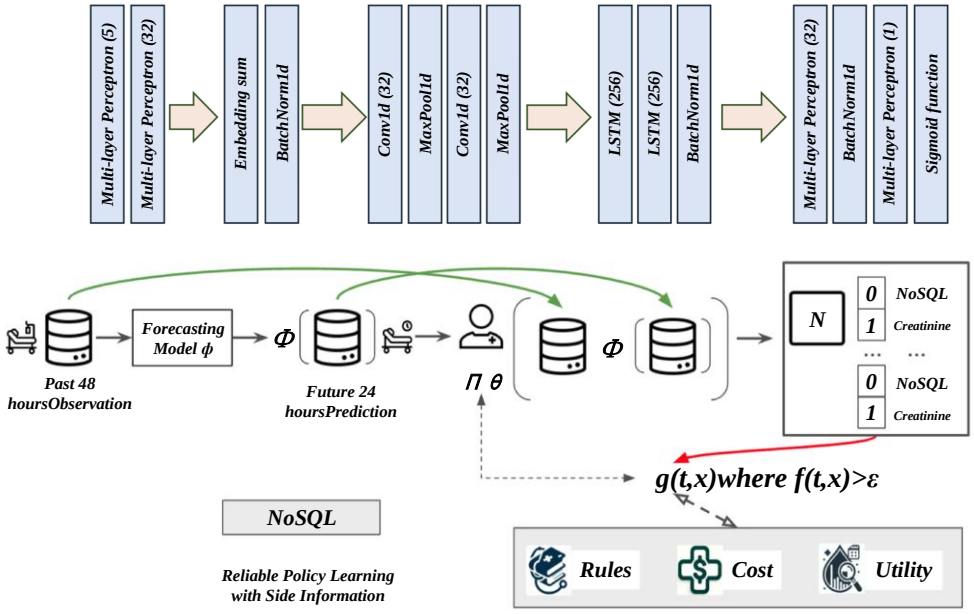


Figure 2: Sharding-load collaboration framework model

Rchain adopts the concept of reduction for the first time, handling cross-shard transactions through parent shards to improve efficiency. The state reduction model is then proposed, which describes the total operation record of the blockchain from the genesis block to a specific moment. With the increase of blocks, the state information increases, and the storage pressure of all nodes increases. State-sharding technology reduces the storage burden, but nodes need to synchronize their states during transaction verification, which affects the speed. The state reduction model is graded according to the synchronization ability of nodes. The upper-layer sharded nodes store more states and have stronger performance.

Huffman coding principle is used for state reduction, and performance is distinguished by binary tree structure. The node performance is represented by the path length to the root node, and the short path makes the performance better. State capability refers to the number of states stored by nodes. If it is limited, it will not be reduced upward and stay at the lower level.

In the full binary tree model, each node represents a shard from different sets. Nodes are numbered in hierarchical traversal order, with the root node being 1. The initial number of shards is 2 to the k power, and k is a non-negative integer.

In the initial stage, nodes are assigned to initial shards numbered from 2k to 2k +1-1 by a random algorithm. Then, the nodes decide whether to synchronize the state of adjacent shards according to the performance and initial shard limit and make self-adjustment to form a binary tree structure. Node performance includes the ability to store and validate transactions. Finally, the nodes are classified according to the storage state to form synthetic shards numbered from 1 to 2k-1. Each shard stores part of the state information, reducing the storage burden. Nodes with similar performance are in the same shard, reducing the transaction verification time.

The research shows that the message passing times of PBFT consensus algorithm increases rapidly with the increase of the number of nodes. Considering that the network bandwidth is usually 100Mbps, the actual average bandwidth is about 10Mbps, and the packet size of each communication remains the same, and the packet size sent per second is 250B, the relationship between network bandwidth and traffic can be deduced, as shown in formula (1).

$$10Mbps \geq 250B \cdot \left(2n^2 - 2n\right) (1)$$

The calculation shows that when using the PBFT consensus algorithm, the number of validation node sets must satisfy formula (2).

$$n \leq 51 (2)$$

The blockchain system sets the total number of verification nodes as N, the proportion of Byzantine nodes as Pn, the system is divided into Ks shards, each shard has Ns verification nodes, and the proportion of Byzantine nodes in the shard is Ps. The number of nodes per shard must be less than the average allocation of the system, i.e. Ns < N/Ks. The variable xi indicates whether the node numbered i is a Byzantine node, see Equation (3) for details.

$$x_i = \begin{cases} 1, Byzantine\ node\ i \\ 0, other \end{cases} (3)$$

Let Xs be the total number of Byzantine nodes in shard s, i.e. Xs= $\sum_{i \in s} x_i$ system randomly assigns verification nodes to each shard, and the nodes behave independently among the shards. The chance of nodes being assigned to any shard is equal, and it remains unchanged for a certain period of time. Therefore, Xs follows a hypergeometric distribution, i.e., Xs ~ H (Ns, N · Pn, N). According to the hypergeometric distribution, the probability that the number of Byzantine nodes in the shard s is x can be calculated, as shown in Equation (4).

$$P\left(X_s = x\right) = \frac{C_{N\cdot P_n}^{x} \cdot C_{N(1-P_n)}^{N_s - x}}{C_N^{N_s}} (4)$$

The research shows that the number of shard nodes and the proportion of Byzantine nodes in the system have nonlinear effects on the proportion of Byzantine nodes in the shard. Therefore, when studying the set of consensus verification nodes within a single shard, the whole system needs to be considered. Let the total number of consensus verification nodes in PBFT algorithm be Nsv, and when there are x Byzantine nodes in the shard, the probability P of the number x1 of Byzantine nodes in the set of consensus verification nodes is determined by formula (5).

$$P\left(X = x1 \mid X_s = x\right) = \frac{C_x^{x1} C_{N_{sv}-x}^{N_{sv}-x1}}{C_{N_s}^{N_{sv}}} (5)$$

To determine the number of consensus verification nodes, you need to evaluate the transaction verification results. The goal is to reach a correct consensus that the proportion of Byzantine nodes must be less than 1/3. Beyond this ratio, the consensus verification node set is invalid. Under a fixed number of nodes, the failure probability Pfailure can be expressed by formula (6).

$$P_{failure}\left(X_{sv} = N_{sv}\right) = \sum_{x=N_{sv}/3}^{N_s} \sum_{x1=N_{sv}/3}^{\min\{N_{sv},x\}} P\left(X_s = x\right) \cdot P\left(X = x1 \mid X_s = x\right) (6)$$

The minimum requirement of the PBFT consensus algorithm for the number of nodes is verified using Python simulations, including the case of all sets $\{N_{sv} | 4 \le N_{sv} \le 51, N_{sv} \epsilon Z\}$. The possibility of successful node collusion in the consensus verification node set is analyzed. When the proportion of Byzantine nodes exceeds 2/3, these nodes may collude, and the probability can be calculated by formula (7).

$$P_{conspiracy}\left(X_{sv} = N_{sv}\right) = \sum_{x=2\cdot N_{sv}/3 \cdot x1=2\cdot N_{sv}/3}^{N_s} P\left(X_s = x\right) \cdot P\left(X = x1 \mid X_s = x\right) (7)$$

Simulation calculation shows that when the number of consensus verification nodes Ns is 16, the collusion probability Pconspiracy is 0.0036. However, in practical applications, the collusion possibility of 16 nodes will be reduced, because the random allocation of nodes reduces the probability of colluding nodes in the same shard, and it becomes difficult for Byzantine nodes to collude into the same set.

The consensus verification node can confirm the number of transactions it verifies within a specific time interval, and can determine the remaining load without additional communication. Analyzing the residual load is helpful to study the load imbalance within a shard. The total number of unvalidated transactions surplustrt (s) of shard s in slot t is expressed by Equation (8).

$$surplustr_t\left(s\right) = Sumtr_t\left(s\right) - Verifitr_t\left(s\right) (8)$$

Sumtrt(s) represents the total amount of transactions collected by the consensus verification node of shard s in time slot t, and Verifitrt(s) represents the number of transactions verified by the consensus verification node of shard s in time slot t.

Points are primarily based on node assessments, including performance and performance. Node performance is key and affects the ability to process transactions. But a node may report false performance scores for increasing the probability of becoming a validation node. Therefore, nodes need to adjust the reported performance points according to the synchronization status. Let the performance integral reported by node i be Si01, the synchronization state integral be Si02, and the upper limit of the performance integral be 100. The corrected node performance integral is calculated according to Equation (9).

$$s_{i0} = kS_{i01} + \left(1-k\right)S_{i02} (9)$$

In the verification mechanism, Si0 represents the node performance integral and k is the weight coefficient. The state synchronization performance integral can be regarded as a test of the node performance integral. The difference between Si01 and Si02 is used as a reference for the weight coefficient k. If the difference is large, it indicates that the node performance integral may be falsified. At this time, the state synchronization speed integral is the main one. If the difference between the performance integral and the state synchronization performance integral is small, the weight coefficient is set to 0.5. This method not only prevents nodes from providing false integrals, but also reduces the influence of external interference. The calculation method of weight coefficient is shown in Equation (10).

$$k = \begin{cases} 0.8, s_{i01} - s_{i02} \le 10 \\ 0.5, other \end{cases} (10)$$

Set ritq as an index of whether node i shows sq behavior in the consensus process of time slot t, which is defined in formula (11).

$$r_{itq} = \begin{cases} 1, s_q \\ 0, other \end{cases} (11)$$

During the consensus process, the node behavior is recorded in the historical record, and the integral is calculated according to these data. See formula (12) for the specific calculation method.

$$Score_i'(t) = s_{i0} + \sum_{j=0}^{t} \sum_{q=1}^{6} r_{ijq} \cdots_q (12)$$

Score 'i(t) represents the accumulated integral of the validation node i before slot t, and the initial value is negative. The point boosting mechanism encourages frequent and well-performing nodes to stay in the shard for a long time, but may lead to node collusion and threaten blockchain security. To avoid this problem, we introduce a strategy of reducing points. s_5 and s_6 are related to incentivizing nodes to perform fraud proofs, set to positive values to increase the integral. Assuming numerical values of s_1 to s_6, consensus nodes perform best, followed by nodes that do not reach consensus, and collusion nodes perform worst. If a node not in the shard provides proof of fraud, set $s_5 > s_6$ according to its impact on blockchain consistency. The comprehensive analysis gives $s_5 > s_6 > s_1 > s_2 > s_3 > s_4$.

3.2 Dynamic environment adaptive mechanism

Dynamic environment adaptive mechanism is the core module of reinforcement learning-driven sharding and load balancing strategy. Its core goal is to cope with the constantly changing load patterns and resource constraints in distributed environments by sensing the system state in real-time and independently generating optimization decisions. The design of this mechanism needs to solve three major challenges: the dynamics of state representation, the continuity of action space and the delay of environmental feedback, and realize end-to-end optimization from environmental observation to policy execution by building a closed-loop control system.

At the state awareness level, the system continuously collects multi-dimensional runtime indicators through distributed probes, including node CPU utilization, memory occupancy, disk I/O throughput, network queue depth, sharded data distribution entropy, and spatiotemporal distribution characteristics of query types. These indicators are encoded as low-dimensional state vectors after time window moving averaging and normalization, which not only retains the statistical characteristics of the original data but also avoids the interference of dimensional explosion on model training. In view of the differences in characteristics of heterogeneous workloads, the mechanism introduces the importance of dynamically adjusting different indicators by attention weight, such as increasing the weight coefficient of disk I/O and network latency in write-intensive scenarios, while focusing on load balancing between memory and CPU when analytical queries are dominant. This dynamic feature selection ability enables state representation to accurately reflect the core contradictions of the current environment.

The modelling of action space needs to take into account the discreteness of shard adjustment and the continuity of load routing. The mechanism maps variables such as shard boundary movement, replica migration priority and request distribution weight into continuous action space and outputs high-dimensional action vectors through the policy network. In order to avoid local optimization, the random noise exploration mechanism is introduced into the action generation process, and the Ornstein-Uhlenbeck process is used to simulate the inertial characteristics of the physical system, so that the action adjustment can achieve a balance between exploration and utilization. For example, a sharding split operation is modelled as a vector offset in a multi-dimensional space whose direction is determined by the load gradient and whose magnitude is limited by the migration cost constraint function. This continuous expression not only improves the fineness of the strategy but also supports collaborative adjustment across shards to avoid system oscillation caused by traditional discrete actions.

The design of the reward function needs to quantify the compromise relationship between short-term operating costs and long-term system benefits. The mechanism defines a composite reward function, which includes five core indicators: node load balancing, cross-shard query ratio, data migration overhead, request-response delay and throughput volatility. Each index is transformed into a normalized reward component through a differentiable function, and the weight coefficient is dynamically adjusted based on the current system's health. When the risk of node overload is detected, the weight of the load balancing component is automatically increased to prioritize the relief of hot spot pressure. While the migration operation is in progress, the throughput fluctuation penalty term is added to suppress the deterioration of service quality. This dynamic weight allocation strategy enables the agent to adapt to the optimization focus of different stages and avoids the policy rigidity caused by fixed weights.

The online learning framework adopts asynchronous experience playback and target network separation technology to solve the problem of temporal correlation of environmental interaction data. The agent stores the real-time collected state transition tuples into the priority experience pool and dynamically adjusts the sampling weight according to the time series difference error to accelerate the learning efficiency of key samples. The policy network and the value network are decoupled through the double delay update mechanism, which effectively suppresses the policy deviation caused by the overestimation of the Q value. The mechanism integrates transfer learning capabilities, quickly adapts to new load patterns through pre-trained models, and adopts elastic model update strategies to perform incremental training during low system load periods to reduce the risk of resource contention. This design ensures that the adaptive mechanism continues to evolve in a dynamic environment while maintaining the stability and reliability of online services.

4 Experiment and results analysis

In this study, a three-order experimental system of "Data Preprocessing-Benchmarking-Result Analysis" is constructed based on the YCSB benchmark: a dataset that conforms to real business characteristics is generated through YCSB, and hybrid read and write requests are injected into the NoSQL cluster (MongoDB 6.0) where the RL policy is deployed, and the performance indicators are synchronously collected and the differences between the RL policy and the traditional static/hash sharding strategy are compared. The experimental environment adopts standardized control: 8 nodes configured with Intel Xeon Gold 6248 CPU, 64GB memory, and 2TB SSD form a cluster, fixed 10GbE network, Ubuntu 22.04 system, YCSB 0.17.0 and TensorFlow 2.10 versions, and limit node resource quotas through Linux cgroups to avoid interference. YCSB uses a standard workload (50% read/50% write, B is 95% read/5% write), the request rate is 1000~5000 req/s, and the dataset is 10 million 1KB key-value pairs. The RL policy state space includes shard CPU utilization, memory usage, and request latency, and the action space defines shard migration and replica adjustment, and the reward function is Negative Request Latency - Shard Migration Overhead (weight 0.7:0.3, pre-

experiment calibration). The variables were strictly delineated: the control variables were hardware configuration, software version, dataset size and request ratio. The independent variables are sharding strategies (RL-driven, static, and hashed). The dependent variables are average request latency, throughput, standard deviation of sharding load, and migration overhead, and the average value is repeated 10 times for each set of experiments to reduce the error and ensure the rigor and reproducibility of the experiment.

Figure 3 shows that the transaction verification rate of MRV-Elastico and Elastico schemes is close to 100%, and the failed transactions are illegal. The validation rate of the Elastico protocol after 4 rounds of validation was 78. 5%, while that of the MRV-Elastico was 91. 58%. This indicates that the MRV-Elastico scheme converges faster to achieve the same validation rate.

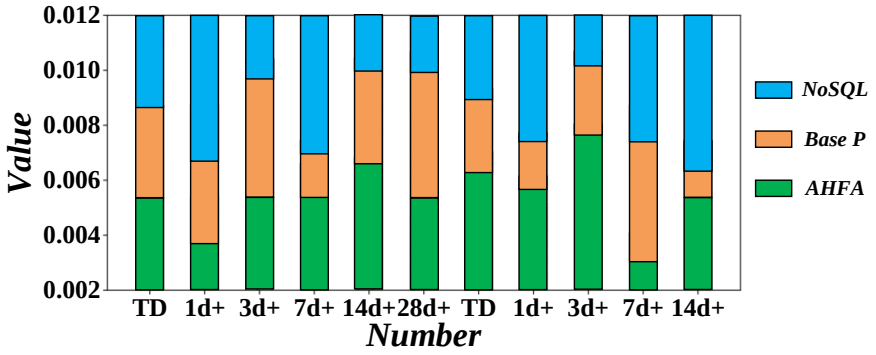


Figure 3: Transaction verification rate

Figure 4 shows that the average blocking time of the MRV-Elastico and Elastico schemes is similar, fluctuating between approximately 49 and 51 seconds. After the load balancing verification of the MRV-Elastico solution, the average block production time fluctuates slightly, indicating that its latency stability is slightly worse. However, this fluctuation is still within the acceptable range and has limited impact on system stability.

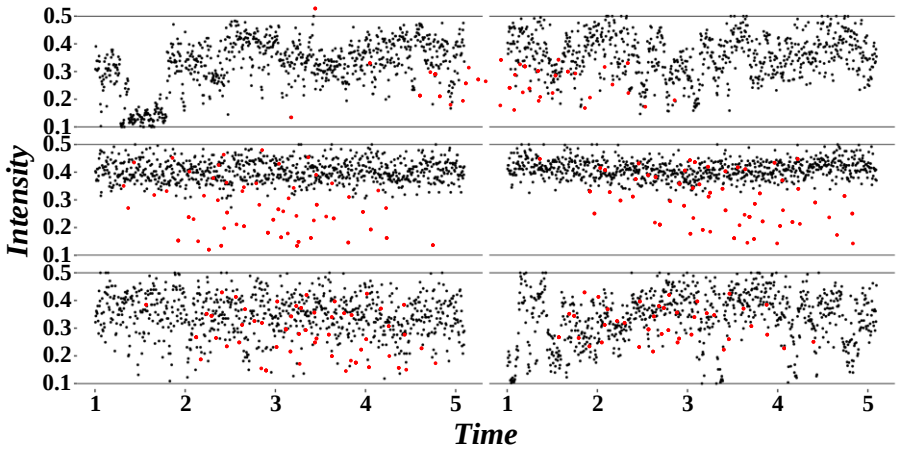


Figure 4: Average block production time

As shown in Table 1, the average response time of ADRL in the Google Trace experiment was reduced by 13. 76% and the energy consumption was reduced by 11. 43%. This is attributed to the irregularity and unpredictability of the load in the experimental scenario, which proves the practicality of ADRL in realistic complex scenarios.

Table 1: Performance comparison of three State-of-the-art algorithms and ADRL

Method	Reduction in average task response time relative to RR	Energy consumption reduction relative to RR
MO-DQN	120. 54%	142. 80%
DRL-cloud	93. 45%	220. 37%
Hierarchical DRL	93. 91%	221. 58%
ADRL (Ours)	229. 55%	251. 14%

Figure 5 shows that the MRV-Elastico and Elastico solutions are significantly different in single-shard transaction processing capabilities. In the Elastico solution, the No. 8 shard processes about 46 transactions/second, and the No. 4 shard processes about 36 transactions/second, resulting in uneven load. Random allocation of transactions may lead to congestion in shard 4 and waste of resources in shard 8.

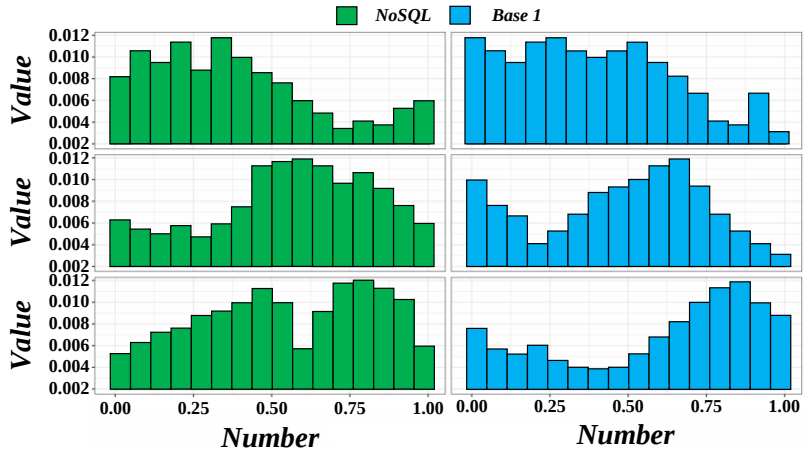


Figure 5: Single shard transaction processing capabilities

Figure 6 shows that the average transaction processing capacity of the MRV-Elastico solution increased to 450 transactions/second over time, while the Elastico solution stabilized at 350-400 transactions/second. After each round of transaction verification, the system optimizes node allocation through the load model so that the performance of the MRV-Elastico solution continues to improve and ultimately surpasses the Elastico solution.

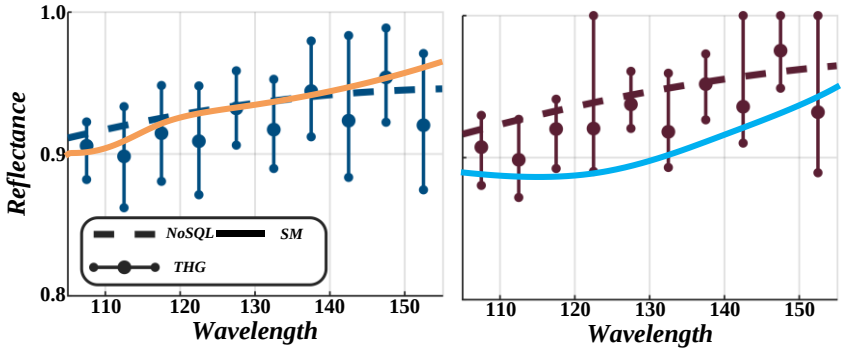


Figure 6: Average transaction processing capacity

Table 2 shows that the RePPO algorithm performs well in balance, tightness and computational efficiency, especially in large-scale data set processing. The balance of MCMC algorithm is insufficient, and the tightness of SMC algorithm is strong. The ReCom algorithm is stable in time and capacity differences, but it is sometimes less compact than other algorithms. This shows that the RePPO algorithm based on deep reinforcement learning is balanced in many aspects and has the potential to solve large-scale spatial service partitioning problems.

Table 2: Performance of multiple spatial service partition algorithms

Data	Algorithm	k = 3			k = 5			k = 7		
		Cgap	Comp	T	Cgap	Comp	T	Cgap	Comp	T
20×10	ReCom	0. 081	22	5. 579	0. 070	41	8. 048	0. 084	56	9. 200
	MCMC	0. 088	22	5. 610	0. 072	41	8. 395	0. 093	56	10. 904
	SMC	0. 081	22	5. 528	0. 067	41	7. 528	0. 075	53	8. 813
	RePPO	0. 075	17	5. 437	0. 064	40	6. 691	0. 077	48	8. 191

Figure 7 shows the cumulative benefits of the DDQN-ShardBlock strategy, which reflects the strategy's performance in terms of blockchain scalability. The simulation results show that the scalability is low at first, but as the agent optimizes the blockchain parameters, the scalability improves. After about 1000 rounds, the agent achieves optimal cumulative returns in different states.

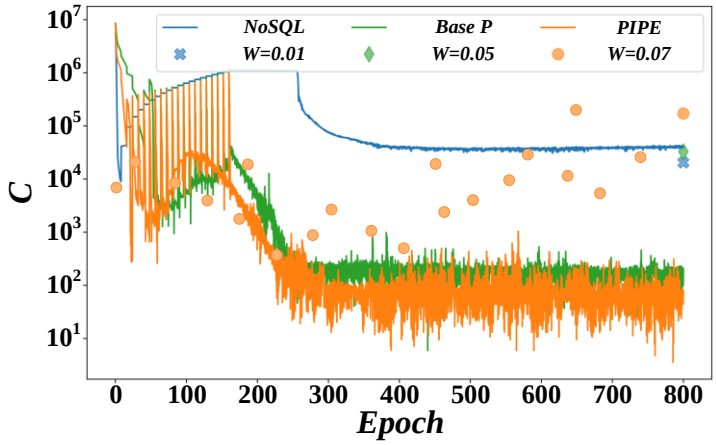


Figure 7: Cumulative benefits of the scheme

Figure 8 shows that as the average transaction size increases, the cumulative return decreases, revealing the changing trend of scalability. When the average transaction size increases, the number of transactions within the block decreases.

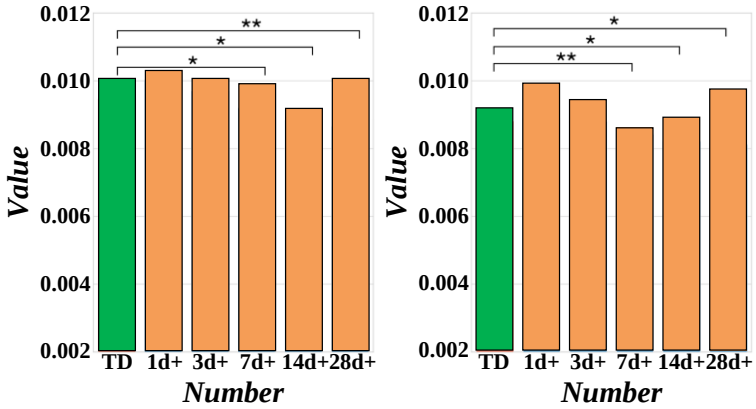


Figure 8: Cumulative returns under different average transaction sizes

Figure 9 shows the change in cumulative revenue after malicious nodes are injected into the blockchain. As the proportion of malicious nodes increases, the probability of selecting these nodes also increases, resulting in malicious blocks and affecting network scalability. The DDQN-ShardBlock strategy reaches a consensus through sharding parallel processing. Even if the scalability decreases under the influence of malicious blocks, it still maintains an excellent level. Even if malicious nodes account for 30%, the cumulative income can be maintained at about 40% of the highest income when there are no malicious nodes.

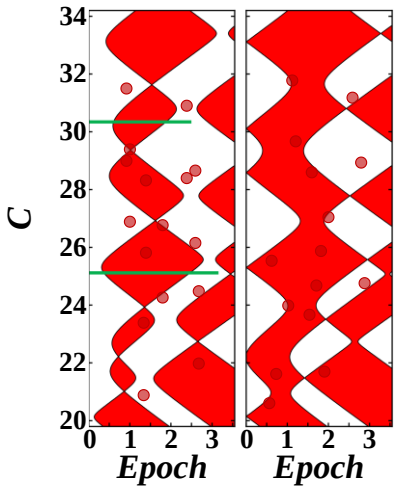


Figure 9: Cumulative benefits under different malicious node ratios

As shown in Figure 10, the long-term reward in the early stage of deep reinforcement learning training is low. But as the number of iterations increases, the reward value improves and converges steadily after about 200 iterations.

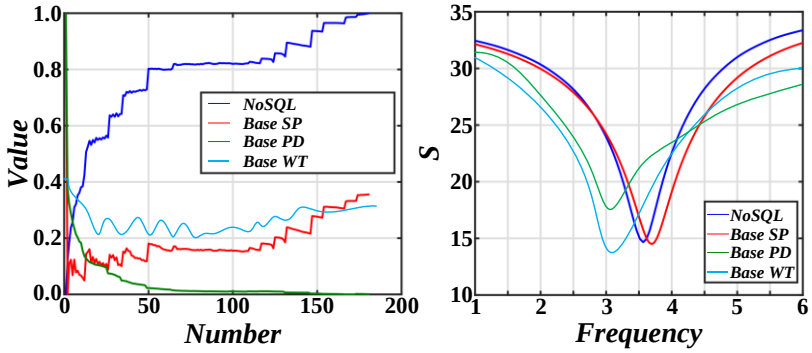


Figure 10: Long-term rewards at different learning rates

Figure 11 shows the variation of the longest on-chip consensus verification time as the number of DRL training increases for Deep Reinforcement Learning (DRL)-based Scheme 1, Traditional Scheme 2, and Shardless Blockchain Scheme 5. The longest on-slice validation delay for Scheme 1 and Scheme 5 decreases with increasing training times and stabilizes after about 200 training sessions. Scheme 2 does not optimize sharding and transaction pool access decisions and PBFT consensus master node selection, so there is no obvious change trend.

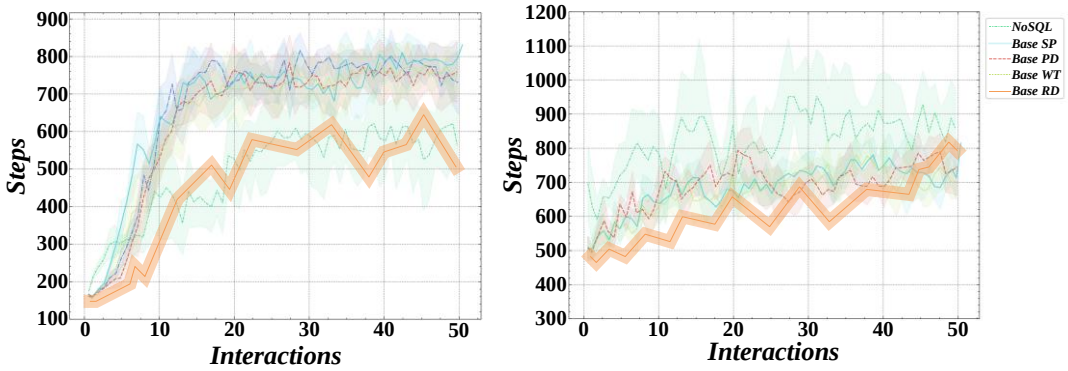


Figure 11: Comparison of longest on-chip consensus verification time

5 Conclusion

Aiming at the sharding and load balancing problems of NoSQL database in dynamic load scenarios, this study proposes an autonomous decision-making framework based on reinforcement learning and verifies its significant advantages in resource utilization, service quality and system stability through experiments. The experiment builds a test environment based on the open-source Cassandra cluster and uses YCSB tools to simulate three typical load scenarios: mixed read-write, burst traffic and long-term fluctuation. Compared with traditional consistent hash, weighted polling and threshold triggering strategies, the proposed method is comprehensively evaluated. Performance improvement effect.

(1) In the read-write mixed load scenario, the reinforcement learning strategy successfully reduced the incidence of hot sharding in a 10-node cluster from 23. 7% in the traditional method to 13. 8% by dynamically sensing the sharded data distribution and query mode, a decrease of 42%. At the same time, the proportion of cross-sharded queries was reduced by 31%, and the average query delay was optimized from 152ms to 99ms.

(2) Agents can effectively learn the correlation law between data locality and load distribution and reduce cross-node access overhead through real-time fine-tuning of sharding boundaries. For burst traffic scenarios, the traditional threshold trigger mechanism causes 28% additional network load due to frequent migration, while the reinforcement learning framework reduces the data migration volume to 72% of the benchmark solution by predicting the load trend and the migration cost trade-off, and the migration operation is concentrated during low load periods, and the system jitter time is shortened by 64%. In the 24-hour unsteady load stress test, the throughput standard deviation of the reinforcement learning strategy is 112ops, which is 61% lower than the 289ops of the traditional method, showing stronger robustness.

(3) Experiments further verify the effectiveness of state representation and reward function design. By introducing the timing characteristics of node CPU utilization, memory footprint, network queue depth and fragment query density, the agent can capture the key load inflexion point in 90% of the decision cycle, avoiding the lag of traditional monitoring indicators. The compound reward function combines throughput weight (0. 6), delay penalty (0. 3) and migration cost coefficient (0. 1), which enables the system to reduce the variance of node resource utilization from 0. 38 to 0. 15 and improve the balance degree by 60% under the condition of ensuring service quality. In addition, the application of a double delay depth deterministic strategy gradient (TD3) algorithm solves the balance problem between action space continuity and exploration noise. After 500,000 training steps, the strategy convergence speed is 2. 3 times faster than that of traditional DQN, and the long-term revenue has increased by 19%.

The dynamic optimization framework driven by reinforcement learning can break through the rigid constraints of static policies and realize the joint optimization of sharding and load balancing in complex and changeable load environments. Experimental data confirm that this method has significant advantages in reducing operation and maintenance costs and improving system flexibility, providing theoretical support and practical examples for the intelligent evolution of distributed databases. Future research will further explore multi-objective optimization and heterogeneous hardware adaptation issues and promote the large-scale application of autonomous database systems in industrial scenarios.

References

[1] M. A. Abdel-Fattah, W. Mohamed, and S. Abd elgaber, “A Comprehensive Spark-Based Layer f or Converting Relational Databases to NoSQL,” Big Data and Cognitive Computing, vol. 6, no. 3, 2022. https://doi.org/10.3390/bdcc6030071

[2] U. Al-Qarni Anwar Ridwan, T. C. Chuah, and Y. L. Lee, “EVOLUTIONARY NETWORK SLI CE ASSOCIATION ALGORITHM FOR LOAD BALANCING IN HETEROGENEOUS OPEN R ADIO ACCESS NETWORKS, ” Journal of Eng ineering Science and Technology, vol. 19, no. 1 , 2024.

[3] D. G. Chandra, "BASE analysis of NoSQL data base," Future Generation Computer Systems-the International Journal of Escience, vol. 52, no., pp. 13-21, 2015. https://doi.org/10.1016/j.future.2 015.05.003

[4] K. Hejja, S. Berri, and H. Labiod, “Network sli cing with load-balancing for task offloading in vehicular edge computing,” Vehicular Communi cations, vol. 34, 2022. https://doi.org/10.1016/j.v ehcom.2021.100419

[5] B. Lamrani, Y. Elmrabet, I. Mathew, N. Bekkio ui, P. Etim, A. Chahboun, A. Draoui, and M. C. Ndukwu, “Energy, economic analysis and m athematical modelling of mixed-mode solar dryi ng of potato slices with thermal storage loaded V-groove collector: Application to Maghreb regi on,” Renewable Energy, vol. 200, 2022. https:// doi.org/10.1016/j.renene.2022.09.119

[6] H. Li, Q. Sheng, Q. Xie, Y. Xie, “Inversion of Lateral Impact Load Migration History and Tr ajectory Location Method for Deck Structures B ased on the Time-Slicing Method,” Shock and Vibration, vol. 2024, 2024. https://doi.org/10.115 5/2024/6663995

[7] W. Lu, X. Chen, “Short-term load forecasting f or power systems with high-penetration renewab les based on multivariate data slicing transforme r neural network,” Frontiers in Energy Research , vol. 12, 2024. https://doi.org/10.3389/fenrg.202 4.1355222

[8] T. Misugi, H. Miura, K. Hirata, T. Tachibana, “Design of Multiple Routing Configurations Co nsidering Load Distribution for Network Slicing, ” Apsipa Transactions on Signal and Informati on Processing, vol. 12, no. 2, 2023. https://doi.o rg/10.1561/116.00000148

[9] T. Niaz, S. Shabbir, T. Noor, M. Imran, “Active Composite Packaging Reinforced with Nisin-Loaded Nano-Vesicles for Extended Shelf Life of Chicken Breast Filets and Cheese Slices,” Food and Bioprocess Technology, vol. 15, no. 6, 2022. https://doi.org/10.1007/s11947-022-02815-2

[10] G. C. Deka, "NoSQL Polyglot Persistence," Deep Dive into Nosql Databases: The Use Cases and Applications, Advances in Computers P. Raj and G. C. Deka, eds., pp. 357-390, 2018. https://doi.org/10.1016/bs.adcom.2017.08.003

[11] D. Chen, Z. Chen, Z. He, J. Gao, and Z. Su, “Learning heuristics for weighted CSPs through deep reinforcement learning,” Applied Intelligence, vol. 53, no. 8, pp. 8844-8863, 2023. https://doi.org/10.1007/s10489-022-03992-5

[12] A. Aggoune, and M. S. Namoune, “P3 Process for Object-Relational Data Migration to NoSQL Document-Oriented Datastore,” International Journal of Software Science and Computational Intelligence-ljssci, vol. 14, no. 1, 2022. https://doi.org/10.4018/ijssci.309994

[13] H. Akid, G. Frey, M. Ben Ayed, and N. Lachiche, “Performance of NoSQL Graph Implementations of Star vs. Snowflake Schemas,” leee Access, vol. 10, pp. 48603-48614, 2022. https://doi.org/10.1109/access.2022.3171256

[14] C. Araujo, M. Oliveira Jr, B. Nogueira, P. Maciel, and E. Tavares, “Performability evaluation of NoSQL-based storage systems,” Journal of Systems and Software, vol. 208, 2024. https://doi.org/10.1016/j.jss.2023.111885

[15] S. Bouaziz, S. Boukettaya, A. Nabli, and F. Gargouri, “A formal algebra for document-oriented NoSQL data warehouses: formalisation and evaluation,”Cluster Computing-the Journal of Networks Software Tools and Applications, vol. 28, no. 3, 2025. https://doi.org/10.1007/s10586-024-04776-x

[16] I. Carvalho, F. Sa, and J. Bernardino, “Performance Evaluation of NoSQL Document Databases :Couchbase, CouchDB, and MongoDB,” Algorithms, vol. 16, no. 2, 2023. https://doi.org/10.3390/a16020078

[17] S. H. Wu, and T. A. Mueller, “A user-friendly NoSQL framework for managing agricultural field trial data,” Scientific Reports, vol. 14, no. 1, 2024. https://doi.org/10.1038/s41598-024-81609-2

[18] S. Ferreira, J. Mendonca, and E. Andrade, “Experimental Performance Analysis of Data Consistency Levels in NoSQL Databases,” Software-Practice& Experience, vol. 55, no. 6, 2025. https://doi.org/10.1002/spe.3412

[19] L. H. Zambom Santana, and R. d. S. Mello, “Persistence of RDF Data into NoSQL: A Survey and a Reference Architecture,” leee Transactions on Knowledge and Data Engineering, vol. 34, no. 3, pp. 1370-1389, 2022. https://doi.org/10.1109/tkde.2020.2994521

[20] A. K. Samanta, and N. Chaki, “An enumerated analysis of NoSQL data models using statistical tools,” Innovations in Systems and Software Engineering, vol. 19, no. 1, pp. 5-14, 2023. https://doi.org/10.1007/s11334-022-00517-8

[21] N. Gupta, and R. Agrawal, "NoSQL Security," Deep Dive into Nosql Databases: The Use Cases and Applications, Advances in Computers P. Raj and G. C. Deka, eds., pp. 101-132, 2018. https://doi.org/10.1016/bs.adcom.2018.01.003

[22] X. Li, H. Xu, J. Zhang, and H. -h. Chang, “Deep Reinforcement Learning for Adaptive Learning Systems,” Journal of Educational and Behavioral Statistics, vol. 48, no. 2, pp. 220-243, 2023. https://doi.org/10.3102/10769986221129847

[23] Z. Li, X. Wang, L Pan, L Zhu, Z. Wang, J. Feng, C. Deng, and L Huang, “Network Topology Optimization via Deep Reinforcement Learning,” leee Transactions on Communications, vol. 71, no. 5, pp. 2847-2859, 2023. https://doi.org/10.1109/tcomm.2023.3244239

[24] Y. Liu, X. Wu, Y. Bo, J. Wang, and L Ma, “A Transfer Learning Framework for Deep Multi-Agent Reinforcement Learning,” leee-Caa Journal of Automatica Sinica, vol. 11, no. 11, pp. 2346-2348, 2024. https://doi.org/10.1109/jas.2023.124173

[25] Y. Matsuo, Y. LeCun, M. Sahani, D. Precup, D. Silver, M. Sugiyama, E. Uchibe, and J. Morimoto, “Deep learning, reinforcement learning, and world models,” Neural Networks, vol. 152, pp. 267-275, 2022. https://doi.org/10.1016/j.neunet.2022.03.037

[26] S. D. Kuznetsov, and A. V. Poskonin, "NoSQL data management systems," Programming and Computer Software, vol. 40, no. 6, pp. 323-332, 2014. https://doi.org/10.1134/s0361768814060152

[27] Q. Sun, X. Wei, and X. Yang, “GraphSAGE with deep reinforcement learning for financial portfolio,” Expert Systems with Applications, vol. 238, 2024. https://doi.org/10.1016/j.eswa.2023.122027

[28] W. Wang, B. Li, X. Luo, and X. Wang, “Deep Reinforcement Learning for Sequential Targeting,” Management Science, vol. 69, no. 9, pp. 5439-5460, 2023. https://doi.org/10.1287/mnsc.2022.4621

[29] X. Wang, S. Wang, S. Liang, D. Zhao, J. Huang, X. Xu, B. Dai, and Q. Miao, “Deep Reinforcement Learning: A Survey,” leee Transactions on Neural Networks and Learning Systems, vol. 35, no. 4, pp. 5064-5078, 2024. https://doi.org/10.1109/tnnls.2022.3207346

[30] K. Wu, H. Wang, M. A. Esfahani, and S. Yuan, “Learn to Navigate Autonomously Through Deep Reinforcement Learning,” leee Transactions on Industrial Electronics, vol. 69, no. 5, pp. 5342-5352, 2022. https://doi.org/10.1109/tie.2021.3078353

A Hybrid PSO-BP Optimized Fuzzy Neural Network for Network Security Situation Awareness

Keguang Yang
College of Information Engineering, College of Arts and Sciences·Kunming, Kunming, 650222, China
E-mail: kmykg163@163.com

Keywords: network security situation awareness, PSO-BP algorithm, principal component analysis, fuzzy evaluation method, types of network attacks

Received: August 26, 2025

With the continuous maturity of computer technology, computers are constantly subjected to network attacks. To better respond to different types of computer network security attacks, a computer network situational awareness model based on particle swarm optimization-error back propagation algorithm is proposed. These two algorithms are optimized by introducing compression factors and momentum. Meanwhile, the model adopts a hybrid update strategy, randomly selecting the PSO or BP algorithm for parameter update at each step. In the experimental results, when the number of iterations of the proposed PSO-BP model exceeded 30, the loss function value, accuracy, recall, and F1-score converged to around 0.01, 0.99, 0.94, and 0.98, respectively. Each performance index is superior to the other two PSO-BP algorithms. In addition, compared with other models such as adaptive fuzzy system, reinforcement learning-based method, CNN, LSTM, and Transformer, the PSO-BP model demonstrates higher detection accuracy and adaptability when dealing with dynamic network attack data. The proposed algorithm has demonstrated superior perception ability in the computer network security situation and against different types of network attacks. In practical situations, the research can provide timely and accurate situational awareness information for network administrators, helping them make quick decisions and reduce losses caused by network attacks. Meanwhile, the real-time performance of the model needs to be further optimized to better adapt to rapidly changing network security threats.

Povzetek: Študija predstavi izboljšani hibridni model v omrežni varnosti, ki dosega visoko natančnost zaznave napadov.

1 Introduction

Computer Network Security Situation Awareness (NSSA) refers to predicting network security by interpreting the potential significance of a series of security elements in a computer network. The common NSSA techniques mainly include data feature extraction, data fusion, and data visualization. Network security regulators can predict possible future network security issues based on the current network security situation, and take proactive measures to defend in advance [1-2]. Network attacks, as a key factor in computer network security incidents, have a huge impact on people's lives and economy due to their diversity. For example, worms can block communication and harm hosts, while DOS attacks can directly cause communication paralysis [3-4]. Fuzzy neural network, as an emerging intelligent computing method, combines the reasoning of fuzzy logic with the self-learning ability of neural networks, making its advantages in the field of network security increasingly significant. Fuzzy neural networks can effectively deal with the vagueness and uncertainty of data by converting the input data into a fuzzy set through fuzzification and reasoning with fuzzy rules. Compared with traditional machine learning methods, fuzzy neural networks perform better when dealing with data with fuzzy boundaries, which can identify complex network attack behaviors more flexibly. At present, scholars have carried out research on computer NSSA. The summary of relevant work is shown in Table 1.

Table 1: Related works summary table

References	Author/year	Method used	Dataset	Numerical result
Reference 5	Xiuli Du / 2023	Grey Wolf Optimization (GWO) algorithm; Clockwork Recurrent Neural Network (CW-RNN)	UNSW-NB15	The accuracy is 0.92000, the precision is 0.90575, the recall is 0.91000, and the F1-score is 0.90786
Reference 6	Adel Alqudhaibi/2023	Quadratic SVM	Self-collected dataset	The accuracy is 64.8%, the precision is 57.0%, the recall is 91.0%, and the F1-score is 70.09%.
Reference 7	Surjeet Dalal/2023	QUEST model	MSCAD	The accuracy during the training phase is 94.09%, and that during the testing phase is 94%.
Reference 8	Hongwu Zhang / 2023	Hierarchical model, grey relational analysis	Self-collected dataset	The accuracy is as high as 98%.

Du X et al. designed a clock recurrent neural network to optimize the accuracy and real-time prediction of network security status. The grey wolf algorithm was introduced to optimize the network parameters. The results showed that the model extracted features of network security status better than that of other network models, and the prediction accuracy was also improved [5]. Alqudhaibi et al. designed a prediction approach based on machine learning for critical infrastructure network attacks to address security risks and vulnerabilities in industrial control systems. The accuracy of the model reached 96% [6]. Dalal S proposed a highly boosting neural network to detect multi-stage network attacks in response to the increasing number of network attacks on firewalls. The Bayesian network model had a prediction accuracy of 97.29% for multi-stage network attacks, while the extremely enhanced neural network had an accuracy of 99.72% for multi-stage network attacks [7]. Zhang H et al. improved the accuracy of hierarchical NSSA data fusion in cloud computing environments. A hierarchical model was built to obtain a hierarchical structure for data fusion. The data fusion accuracy reached 98%, which improved the NSSA ability [8]. Although the methods proposed in the above research can make high-level predictions on the network security situation, they have not optimized the relevant algorithms, and the research object is only targeted at a single type of network attack. Therefore, a computer NSSA model based on Particle Swarm Optimization-Error Back Propagation Algorithm (PSO-BP) is designed. The research aims to improve the detection accuracy and response speed of different types of network attacks by optimizing the NSSA model, thereby effectively reducing the losses caused by network attacks and enhancing the overall security of network systems. The innovation of the research lies in optimizing gradient descent by introducing a momentum factor, accelerating network convergence, and reducing oscillations. Meanwhile, a compression factor is introduced to optimize the PSO algorithm, improving the global search ability and convergence speed. In addition, the model combines online learning mechanisms and transfer learning strategies, enabling it to adapt to new attack types in real-time, quickly adjust parameters, and enhance its ability to identify new attacks. Finally, the model introduces a hybrid update strategy. Specifically, during each step of parameter update, the model randomly selects the PSO or BP algorithm for update. This hybrid update mechanism effectively balances the global search and local search capabilities, avoiding the traditional BP algorithm from easily falling into local optima. Compared with adaptive fuzzy control, the proposed model not only combines the advantages of fuzzy logic and neural networks, but also optimizes through PSO and BP algorithms, making it particularly suitable for processing data with fuzzy boundaries. In addition, the momentum mechanism and compression factor further improve the optimization efficiency, making it perform well in handling complex network attack data. In contrast, adaptive fuzzy control mainly achieves complex system control through fuzzy logic systems, with a focus on generating and adjusting fuzzy rules. The research aims to optimize the predictive ability of computer network security situation and reduce the losses caused by network attacks. The research can provide theoretical reference for developing real-time network security intervention measures.

2 Methods and materials

2.1 Construction of computer network situation awareness model

The computer NSSA refers to the set of characteristics that can affect the network situation, such as various security threats, vulnerabilities, etc. that exist in the computer network. The common security situational awareness models include Boyd control loop model, OODA model, JDL model, etc [9-10]. To optimize the NSSA, based on traditional perception models, optimization measures are made. The overall computer NSSA model is displayed in Figure 1.

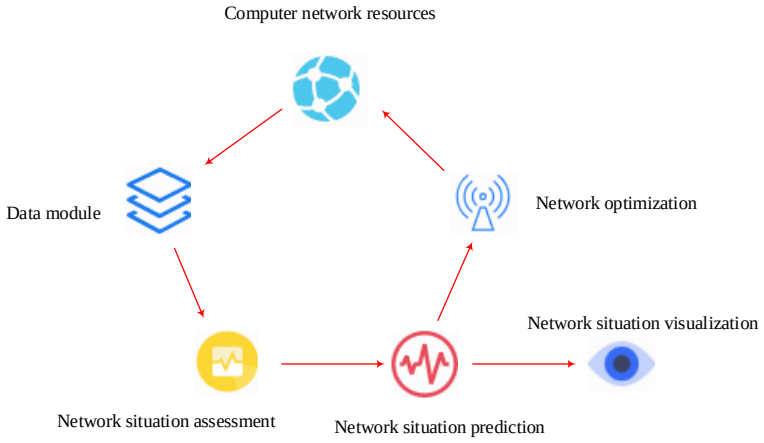


Figure 1: Diagram of network situational awareness model

In Figure 1, the computer NSSA mainly has the following four parts. Firstly, the data module is the foundation of the integrated perception model. After obtaining network security situation data from different sources, the most typical data features are selected. Finally, the data is preprocessed by removing outliers and filling in missing values [11-12]. After obtaining relevant data modules, the network security situation is evaluated based on the preprocessed data mentioned above. Fuzzy evaluation method and Principal Component Analysis (PCA) are introduced to obtain situation values, which reflect the current state of network operation. The situational value is calculated based on the weights of several features represented by PCA. Then, the data is classified and evaluated using fuzzy evaluation method. The PCA method first standardizes the data by defining an initial matrix X, which is mathematically expressed as formula (1).

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{q1} & x_{q2} & \cdots & x_{qm} \end{pmatrix} \tag{1}$$

In formula (1), m represents the data dimension. q represents the amount of data. Firstly, the initial matrix is normalized, as shown in formula (2).

$$\overline{x}_{jk} = \frac{x_{jk} - E_j}{\sigma_j} \tag{2}$$

In formula (2), x_{jk} represents the j-dimensional feature of the k-th data. σ_j signifies the variance of the j-th dimension. E_j signifies the mean of the j-th dimension. After normalization, the matrix is transposed to obtain X^T. The correlation coefficient matrix is calculated based on this transposed matrix. The relevant mathematical expression is shown in formula (3).

$$\begin{cases} CCM = (ccm_{jk}) \\ ccm_{jk} = (\sigma_{jk}) / \sqrt{\sigma_{kk}} \square \sqrt{\sigma_{jj}} \end{cases} \tag{3}$$

In formula (3), ccm_{jk} represents the correlation coefficient. σ_{jk} signifies the variance corresponding to the k-th feature of the j-th data. σ_{kk} represents self-variance. σ_{jj} signifies the variance of the j-th data. The corresponding eigenvalues and eigenvectors are obtained through CMM. The variance contribution rate is calculated, as displayed in formula (4).

$$\alpha_k = \lambda_k / (\sum_{k=1}^{m} \lambda_k) \tag{4}$$

In formula (4), α_k represents the variance contribution rate corresponding to each principal component. λ represents the characteristic root. The variances of the first d principal components are summed. If the value is greater than 0.95, the resulting d principal components are defined as the principal components after dimensionality reduction. Therefore, the mathematical expression of principal components based on dimensionality reduction is shown in formula (5).

$$F_v = L_{1v}X_1 + L_{2v}X_2 + L_{3v}X_3 + \cdots + L_{mv}X_m \quad v \prec d \tag{5}$$

In formula (5), L represents the combination coefficient of the initial matrix. Based on formula (5) and combined with variance contribution rate, the situation value of computer network security is between $[0,1]$. After normalization, the mathematical expression of the weights is finally obtained, as shown in formula (6).

$$\begin{cases} w_k = \psi_k / (\sum_{k=1}^{m} \psi_k) \\ \psi_k = (\sum_{v=1}^{d} L_{kv}\alpha_1) / (\sum_{k=1}^{d} \alpha_1) \end{cases} \tag{6}$$

In formula (6), w_k represents the final weight. To evaluate the situation value, the research introduces the fuzzy evaluation method to assess the types of data features. The number of fuzzy subsets is $b = \{b_1, b_2, b_3, \cdots b_o\}$. The membership function $\mu_{ab}(a = 1,2,3,\cdots m; b = 1,2,3,\cdots o)$ for each fuzzy subset is defined. Among them, μ_{ab} is the membership degree of the a-th indicator in the b-th domain. The research first introduces the K-means Clustering (K-means) based on membership functions, and uses the Calinski-Harabaz index to characterize the number of clusters. The index is the ratio of inter group dispersion to intra group dispersion. The larger the value, the better the clustering effect [13-14]. The relevant mathematical expression is shown in formula (7).

$$\begin{cases} s = \dfrac{tr(B_f)}{tr(W_f)} \square \dfrac{n_E - f}{f - 1} \\ W_f = \sum_{t=1}^{f} \sum_{x \in C_t} (x - c_q)(x - c_q)^T \\ B_f = \sum_{t=1}^{f} p_q (c_q - c_E)(c_q - c_E)^T \end{cases} \tag{7}$$

In formula (7), W_f represents the discrete matrix within the group. B_f represents the inter group discrete matrix. C_q signifies the point set of clusters q. c_q signifies the center of cluster q. p_q is the number of points in cluster q. n_E represents the sample size. f

indicates the number of clusters. After specifying the number of clusters, f samples are defined as cluster center points. The cluster center point corresponding to the first sample is taken as the initial value. The remaining samples are adjusted to the class with the smallest distance using Euclidean distance. Finally, the average of each group is calculated, and the center of the group is updated. After determining the number of clusters and cluster centers, the research selects Gaussian functions as membership functions. The mean of the Gaussian function corresponds to the center vector c discussed in the above, and the variance is calculated in formula (8).

$$\sigma_i = c_{\max} / \sqrt{2f} \tag{8}$$

In formula (8), σ_i represents the variance of the membership function. $c_{\max}$ represents the maximum distance. Combining the center vector and variance, the mathematical expression of the membership function is shown in formula (9).

$$\mu_{ij} = \exp\left\{\frac{-(x_i - c_{ij})^2}{\sigma_i^2}\right\} \tag{9}$$

In formula (9), μ_{ij} represents the membership function of the i-th feature based on the j-th class. After determining the membership function, the membership degree of a single feature is counted. Based on the corresponding membership relationship, the related fuzzy relationship matrix is shown in formula (10).

$$N = \begin{pmatrix} n_{11} & n_{12} & \cdots & n_{1f} \\ n_{21} & n_{22} & \cdots & n_{2f} \\ \vdots & \vdots & \vdots & \vdots \\ n_{m1} & n_{m2} & \cdots & n_{mf} \end{pmatrix} \tag{10}$$

In formula (10), $n_{mf} = \mu_{mf}(x_m)$ can be obtained through the corresponding relationship. Based on the matrix N, fuzzy relationship calculation is performed. The term with the largest single feature is selected as the output. Therefore, for the maximum value of each row of the matrix, the matrix FN is obtained through transposition. Combined with the weight values discussed in the above research, the final normalization is shown in formula (11).

$$\begin{cases} \overline{FER} = \dfrac{FER}{\sum FER} \\ FER = \sum_{k=1}^{m} fn_m w_m \end{cases} \tag{11}$$

According to formula (11), the final fuzzy evaluation result can be obtained. The computer network situational awareness model mainly consists of an input layer, a member layer, a rule layer, a normalization layer, a fuzzy logic layer, and an output layer. The specific input layer has a total of 43 units, the affiliation layer has 43×3 units, the rule layer, the normalization layer, and the fuzzy logic layer each have 100 units, and the output layer has 1 unit, representing the probability of attack occurrence. In terms of parameters, the weight initialization range is [-0.1, 0.1], and a random initialization method is adopted to ensure that the model has good convergence in the early stage of training. The centers and variances of the membership function are determined by the K-means clustering algorithm. The cluster centers serve as the mean of the membership function, and the variances are calculated by formula (8). The evaluation metrics used are Accuracy, Recall, F1-score, False Positive Rate (FPR), False Negative Rate (FNR), False Alarm Rate (FAR). The threshold for attack detection is set to 0.5, that is, when the output probability is greater than 0.5, it is judged as an attack. When it is less than or equal to 0.5, it is judged as a normal flow rate. For each performance indicator, the research sets the confidence interval at 95%. Based on the research, the uncertainty and variability of the model performance can be more comprehensively demonstrated.

2.2 Optimization of computer network situation awareness model based on fuzzy neural network algorithm

Based on the computer NSSA model in the above research, the research first analyzes the structure of the fuzzy neural network. Based on these advantages of the fuzzy neural network, an improved T-S fuzzy neural network structure is proposed. The standard network structure is relatively simple and can be widely applied in simple fuzzy reasoning. The T-S fuzzy neural network structure is more complex compared with the standard type. The structure is mainly divided into a antecedent network and a posterior network. The former has an input layer, a membership layer, a rule layer, and a normalization layer, while the latter has a fuzzy logic layer and an output layer that are separately owned by the T-S network [15-16]. An improved T-S type network structure is proposed, as presented in Figure 2.

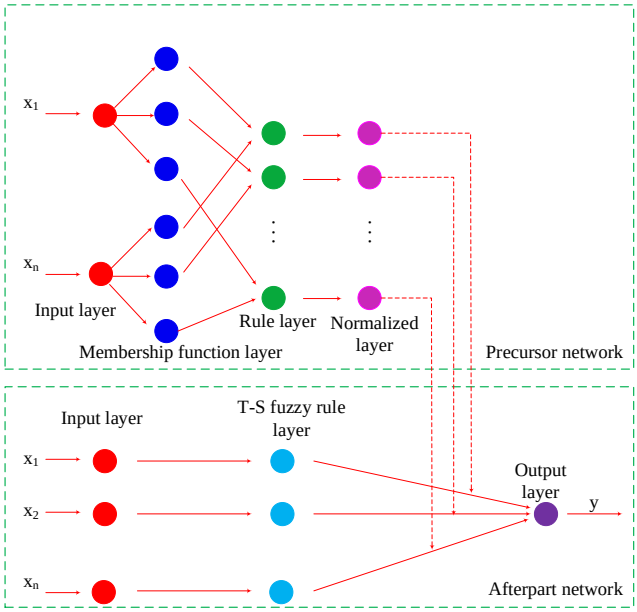


Figure 2: Improved T-S neural network structure diagram

In Figure 2, for the antecedent network, the first layer is the input layer, which inputs the features of the data into the neural network model. The membership function layer uses the Gaussian function proposed in the above research as the membership function to fuzzify the input features, and combines all features into a fuzzy set. The fuzzy subset output by the membership function layer is used as input for fuzzy rule output. The output of the rule layer is shown in formula (12).

$$\xi_j = \mu_1^n \mu_2^n \mu_2^n \cdots \mu_m^n \quad n = 1,2,3,\cdots f \tag{12}$$

In formula (12), f represents the number of subordinate layers. Finally, by normalizing the rule layer, the normalized output value is displayed in formula (13).

$$\overline{\xi_j} = \frac{\xi_j}{\sum_{r=1}^{f} \xi_r} \tag{13}$$

In formula (13), $\overline{\xi_j}$ represents the normalized output value. For the backend network part of the fuzzy neural network, after passing through the input layer, the fuzzy logic layer performs fuzzy inference based on the data features of the input layer. The output of the fuzzy logic layer combines the normalized output values of the frontend network. Therefore, the output of the fuzzy logic layer is shown in formula (14).

$$\eta_j = \sum_{i=1}^{m} p_{ij} x_i \quad i = 1,2,3,\cdots m \quad j = 1,2,3,\cdots f \tag{14}$$

In formula (14), p_{ij} signifies the corresponding weight parameter. Finally, η_j enters the output layer. The Sigmoid activation function is introduced to optimize the generalization ability. Therefore, the final output of the fuzzy neural network is shown in formula (15).

$$\begin{cases} y = h(\eta_1 \xi_1 + \eta_2 \xi_2 + \eta_3 \xi_3 + \cdots \eta_f \xi_f) \\ h(x) = 1/(1+e^{-x}) \end{cases} \tag{15}$$

In formula (15), $h()$ represents the Sigmoid activation function. Regarding learning algorithms, the research introduces Error Back Propagation (BP) and Particle Swarm Optimization (PSO) for NSSA. The BP algorithm mainly optimizes parameters through gradient descent. Compared with the traditional BP neural network, the PSO-BP algorithm optimizes the weights and thresholds of BP neural network through the global search capability of PSO, which avoids the problem that BP algorithm is easy to fall into local optimal. The global optimization capability of PSO makes up for the problem that the traditional BP neural network is sensitive to the initial weight, and significantly improves the stability and convergence speed of the model. Considering the increasing abundance of novel cyber-attacks, the research first introduces an online learning mechanism into the model. Online learning allows the model to continuously receive new data during operation and update the model parameters in real-time. The online learning mechanism is implemented through incremental learning. During its operation, the model continuously receives new data and updates its parameters in real-time. The specific implementation steps are as follows. Firstly, the newly received data are preprocessed, including normalization, feature extraction, etc. Secondly, the model parameters with new data are updated and optimized using the PSO-BP algorithm. Finally, the model performance is evaluated in real-time to ensure that the model maintains high detection accuracy in dynamic environments. Meanwhile, for some new types of attacks that are

similar to known attack types, the research introduces a transfer learning strategy by transferring existing model knowledge to new attack scenarios. The model can quickly adapt with the support of a small amount of new data, especially when it detects new variants of attacks. It uses existing model weights as initial values and only fine tunes some parameters to improve the model's ability to recognize new attacks. The transfer learning strategy is implemented by transferring the existing model knowledge to new attack scenarios. Firstly, using the existing model weights as initial values, some parameters are fine tuned. Secondly, a small amount of new data is used to fine tune the model to ensure that it can quickly adapt to new attack scenarios. Finally, the performance of the model in the new attack scenario is verified through experiments to ensure the adaptability and generalization ability of the model. In the above research, the output of the neural network model is y. Combined with the expected true value of the output, the squared loss function of the neural network is presented in formula (16).

$$L = \frac{1}{2}(y^c - y)^2 \tag{16}$$

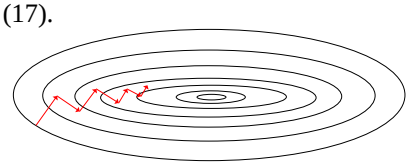
In formula (16), y^c signifies the expected true value of the output. L signifies the loss function. Based on formula (16), when the expected value is close to the actual value, a smaller L value indicates better fitting performance. The optimization of the loss function in the BP is mainly in the backward propagation stage, so gradients are used in the propagation process to update weights, membership function centers, and variances, as shown in formula (17).

$$\begin{cases} p_{ij}(t+1) = p_{ij}(t) - \alpha \dfrac{\partial L}{\partial p_{ij}} \\ c_{ij}(t+1) = c_{ij}(t) - \alpha \dfrac{\partial L}{\partial c_{ij}} \\ (t+1) = \sigma_i(t) - \alpha \dfrac{\partial L}{\partial \sigma_i} \end{cases} \tag{17}$$

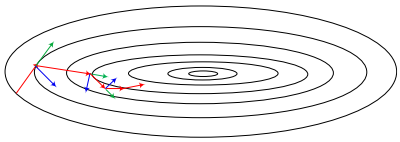
In formula (17), $p()$ represents the weight. $c()$ represents the center of the membership function. $\sigma()$ represents the variance. When using gradient descent for parameter updates, the direction of single optimization is perpendicular to the direction of the previous update. During this process, there is often slow convergence speed in the network. Therefore, the research introduces momentum to optimize the optimization direction of gradient descent parameter updates. By introducing the momentum factor, the convergence of the network can be accelerated and the oscillations during the gradient descent can be reduced. The related mathematical expression is shown in formula (18).

$$v(t+1) = \mu v(t) + \eta \nabla E(t) \tag{18}$$

In formula (18), $v(t)$ denotes the velocity at the current moment. μ denotes the momentum factor. η denotes the learning rate. $\nabla E(t)$ denotes the current gradient. The momentum factor optimizes the direction and speed of parameter update by combining the velocity of the previous moment and the direction of the current gradient, and the related schematic is shown in Figure 3.



(a) Initial BP algorithm gradient descent diagram



(b) Gradient descent diagram of BP algorithm with momentum

Figure 3: Schematic diagram of introducing momentum gradient descent to find the optimal directional contrast

Figure 3 (a) shows the optimization direction of gradient update in the original state, which is perpendicular to the previous direction. Figure 3 (b) shows the optimization direction after introducing momentum. The red and blue lines represent the gradient direction, the green line signifies the velocity direction, and the red line signifies the optimized optimization direction based on the vector operation criterion. Therefore, the main idea is to use the direction updated last time as the basis, and combine it with the gradient as the update direction, which can make the network structure faster. The correction of relevant parameters is shown in formula (19).

$$\begin{cases} v_t = d\Phi_t + qv_{t-1} \\ \eta_{t+1} = \eta_t - v_t \end{cases} \tag{19}$$

In formula (19), v_t and v_{t-1} represent the current and previous updates, respectively. d represents the efficiency of gradient learning. Φ_t represents the gradient. q represents the momentum factor. The PSO algorithm is further used to learn parameters. The research makes the following improvements based on the basic PSO. A compression factor is applied to accelerate the convergence speed of the PSO and optimize the computational efficiency. The compression factor is introduced, which can dynamically adjust the speed of the particles to avoid premature convergence of the particles to the local optimal solution. The mathematical expression based on the compression factor is shown in formula (20).

$$\begin{cases} \lambda = \dfrac{2}{|2-\varphi-\sqrt{\varphi^2-4\varphi}|} \\ \varphi = c_1 + c_2 \end{cases} \tag{20}$$

In formula (20), λ represents the compression factor. c_1 and c_2 represent learning factors. In the PSO algorithm, the momentum and compression factors can significantly improve the convergence speed and global search ability of the algorithm. Specifically, first, the position k and velocity k of the particles are initialized, while the individual optimal position k and the global optimal position k are initialized. Then, the position of the particle is updated based on formula (21).

$$\begin{aligned} & v_i(t+1) = q \cdot v_i(t-1) + \lambda \cdot (\omega \cdot v_i(t)) \\ & + c_1 \cdot r_1 \cdot (gbest_i - x_i(t)) + c_2 \cdot r_2 \cdot (gbest - x_i(t))) \cdot x_i(t) \\ & = x_i(t) + v_t(t+1) \end{aligned} \tag{21}$$

After iterative updates based on formula (21), the fitness of each particle is evaluated, and the individual optimal position and the global optimal position are updated. The iteration stops when the maximum number of iterations is reached or the convergence conditions are met. Based on the optimized BP and PSO discussed in the above research, a new PSO-BP is formed by combining the two to complete the parameter learning. The steps based on the PSO-BP are as follows. Firstly, the parameters of PSO and BP are initialized, and the selection probability p is defined. A number between [0,1] is randomly generated and compared it with the selection probability. If the random number is less than the random probability, PSO is applied to update the parameters. Otherwise, BP is applied to update the parameter values. Whether the maximum iteration is reached or the optimal value has been found is determined. If it meets the requirement, it ends. Otherwise, it continues to generate random numbers and repeats the above steps. Finally, the pseudo-code of PSO-BP is shown in Table 2.

Table 2: PSO-BP pseudo-code

```
Algorithm: PSO-BP
Input: Training dataset D, number of particles p_size, maximum iterations max_iter
Output: Optimized neural network weights W

Initialize:
    Initialize the neural network weights W randomly
    Initialize the particle swarm with p_size particles
    For each particle i:
        Initialize the position x_i and velocity v_i randomly
        Set the personal best position pbest_i to x_i
        Evaluate the fitness of x_i using the training dataset D
    Set the global best position gbest to the best pbest_i

Main Loop:
    For iteration = 1 to max_iter:
        For each particle i:
            Update the velocity v_i using:
                v_i = w * v_i + c1 * rand() * (pbest_i - x_i) + c2 * rand() * (gbest - x_i)
            Update the position x_i using:
                x_i = x_i + v_i
            Evaluate the fitness of x_i using the training dataset D
            If the fitness of x_i is better than pbest_i:
                Update pbest_i to x_i
            If the fitness of x_i is better than gbest:
                Update gbest to x_i
        Update the neural network weights W using the global best position gbest
        Perform backpropagation to fine-tune the weights W
        If the stopping criterion is met (e.g., convergence or max_iter reached):
            Break

Return the optimized neural network weights W
```

3 Results

3.1 Performance analysis based on PSO-BP

The experimental environment parameters and PSO-BP algorithm parameter settings are displayed in Table 3.

Table 3: Experimental environment parameters

Parameter name	Parameter model	Parameter name	Parameter model
CPU	Intel® Core™ i5-12500H	Inertial weighting	0.5
RAM	64GB	c1	2
GPU	NVIDIA GeForce RTX 3050	c2	2
Kernel	14 nuclei	BP algorithm learning rate	0.1
Display memory	4GB	Gradient learning rate	
Development language	Python3.8	Random numbera a	0.5
Related libraries	Keras 、 Slearn	Maximum iterations	100
Number of particles p_size	50	Center Particle range	[0,1]
Delta particle range	[0,0.2]	P particle value range	[-50,50]
PSO algorithm learning rate	0.1	Penalty factor	0.3

Batch size	32	Epochs	50

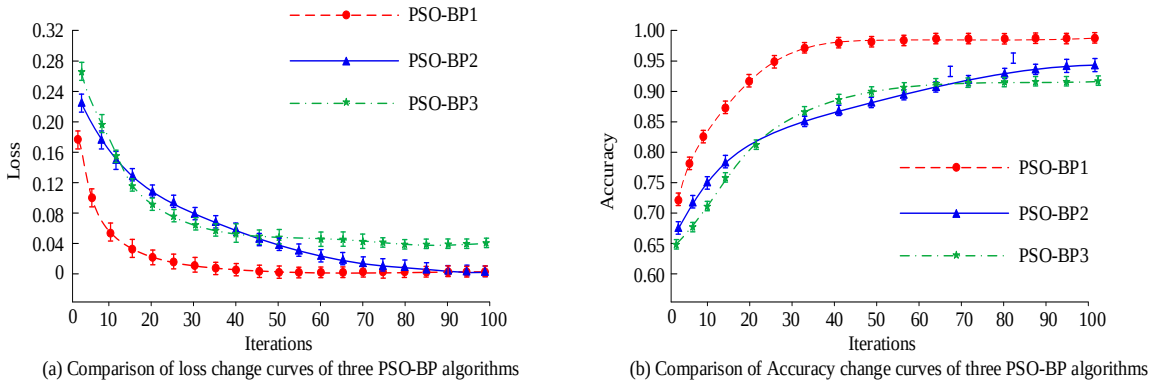
After setting the parameters in Table 3, the UNSW-NB15 dataset is selected to assess the performance of the PSO-BP. The dataset is a large dataset developed by the University of New South Wales in Australia for research on network intrusion detection systems, capable of simulating normal and abnormal network traffic in real network environments. Subsequently, the data in the dataset is preprocessed, dividing into training and testing sets in a 6:4. The data set is randomly divided into K subsets, ensuring that each subset is equal in size. In each iteration, a subset is selected as the validation set, and the remaining K-1 subsets are merged into the training set. It is repeated K times, ensuring that each subset is used as a validation set once. Meanwhile, SMOTE oversampling is adopted. By inserting new synthetic samples between the minority class samples, the number of minority class samples is increased to balance the dataset. Simulation experiments are performed on six different types of network attack types in the dataset, including DoS, Fuzzers, and Shellcode. The sample distribution and attack types of the dataset are displayed in Table 4.

Table 4: Sample distribution and attack types in the dataset

Attack type number	Training set	Testing set
A1	220	147
A2	4,333	2,888
A3	28,456	18,970
A4	5,098	3,398
A5	754	503
A6	109	72

After completing the classification of attack types in the dataset. The performance of the research method is assessed using the training set. The index includes loss function, accuracy, recall, and F1-score. The PSO-BP algorithm discussed in references [17] and [18] is selected for comparative experiments [17-18]. In each iteration of PSO-BP1, a number between [0,1] is randomly generated and compared with the selection probability. If the random number is less than the selection probability, the PSO algorithm is used to update the parameters. Conversely, the parameter values are updated based on the BP algorithm. PSO-BP2 only uses the PSO algorithm for parameter update and does not combine the BP algorithm. PSO-BP3 only uses the BP algorithm for parameter update and does not combine the PSO algorithm. The loss function and accuracy variation curve on the training set are shown in Figure 4.

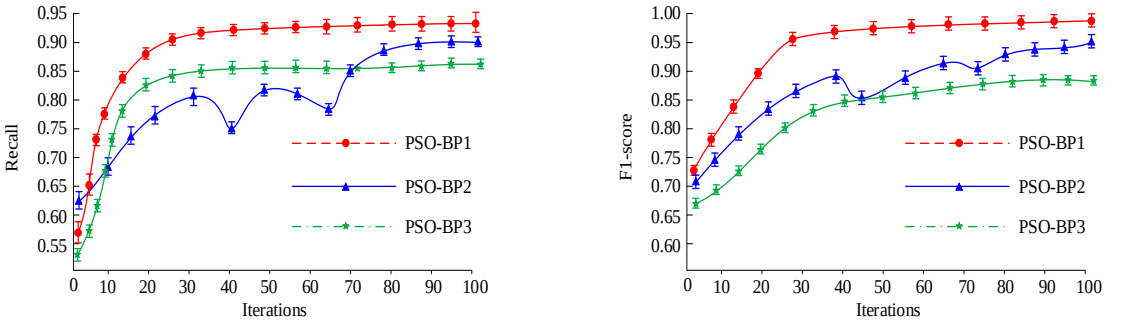
Figure 4 (a) shows the loss function curves of three PSO-BP algorithms. The loss function usually represents the difference between the output value and the true value of the network model. The loss function values of all three algorithms decreased with the increase of iterations. The research method (PSO-BP1) tended to flatten the loss function values when the iteration exceeded 30, and the loss function value eventually converged to around 0.01. The comparison algorithm PSO-BP2 had the same loss function convergence value as the PSO-BP1, but its convergence speed was slow. The loss function value only flattened at around 90 iterations. Finally, the PSO-BP3 algorithm had the maximum loss function convergence value, which was around 0.04. Figure 4 (b) displays the accuracy curves. The accuracy of all three methods increased with the increase of iterations. The accuracy of the PSO-BP1 outperformed the other two algorithms, and its accuracy eventually converged to around 0.99. The recall rate and F1-score are further compared, as displayed in Figure 5.



(a) Comparison of loss change curves of three PSO-BP algorithms

(b) Comparison of Accuracy change curves of three PSO-BP algorithms

Figure 4: Comparison of loss function and accuracy curve of three PSO-BP algorithms



(a) Comparison of Recall change curves of three PSO-BP algorithms

(b) Comparison of F1-score change curves of three PSO-BP algorithms

Figure 5: Comparison of recall and F1-score curves of three algorithms

Figure 5 (a) displays the recall curves. The recall of the PSO-BP1 and PSO-BP3 increased with the increase of iterations. However, PSO-BP2 exhibited oscillations between 30-70 iterations, which may be due to over-fitting of the model, leading to fluctuations in the recall within the iteration region. The final recall of the PSO-BP1 converged to around 0.94, which outperformed other algorithms. Figure 5 (b) shows the F1-score curve for three algorithms. The PSO-BP1 had the highest F1-score of 0.73 in the initial state, followed by PSO-BP2 and finally PSO-BP3. After convergence of three methods, the F1-score of the PSO-BP1 exceeded the other two algorithms, and the final F1-score converged to around 0.98. A K-fold cross-validation instrument is introduced to verify the performance of the module under different datasets, where the K is set to 5. Therefore, the performance indicators based on K-fold cross-validation are shown in Table 5.

In Table 5, the PSO-BP1 is superior to the other two PSO-BP models in various performance indexes. In terms of accuracy, the average accuracy of the PSO-BP1 reached 98.1%, the average recall reached 96.5%, the average F1-score was 97.3%, and the FPR, FNR, and FAR were 1.8%, 3.5% and 3.3%, respectively. Through K-fold cross-validation, the stability and generalization ability of the model under different data partition are verified. The experimental results show that the performance indexes of the PSO-BP1 fluctuate less and the standard deviation is low under different fold numbers, which indicates that the model has good universality and robustness. The research further introduces Rule-Based System (RBS), traditional BP neural network, and Support Vector Machine (SVM) for comparative analysis. The results are shown in Table 6.

Table 5: Cross-validation results

Number of folds	Algorithm name	Accuracy	Recall	F1-score	FPR	FNR	FAR
1	PSO-BP1	0.982	0.965	0.973	0.018	0.035	0.032
2	PSO-BP1	0.978	0.959	0.968	0.021	0.041	0.038
3	PSO-BP1	0.985	0.972	0.978	0.015	0.028	0.029
4	PSO-BP1	0.981	0.960	0.971	0.019	0.040	0.035
5	PSO-BP1	0.983	0.968	0.975	0.017	0.032	0.031
1	PSO-BP2	0.965	0.94	0.952	0.025	0.060	0.045
2	PSO-BP2	0.96	0.935	0.947	0.028	0.065	0.05
3	PSO-BP2	0.968	0.945	0.956	0.023	0.055	0.048
4	PSO-BP2	0.962	0.938	0.95	0.026	0.062	0.052
5	PSO-BP2	0.967	0.942	0.954	0.024	0.058	0.049
1	PSO-BP3	0.958	0.93	0.943	0.030	0.070	0.055
2	PSO-BP3	0.955	0.925	0.939	0.032	0.075	0.060
3	PSO-BP3	0.96	0.932	0.946	0.029	0.068	0.057
4	PSO-BP3	0.957	0.928	0.942	0.031	0.072	0.059
5	PSO-BP3	0.962	0.935	0.948	0.028	0.065	0.054

Table 6: Performance analysis of PSO-BP algorithm and control algorithm

Algorithm name	PSO-BP1	RBS	BP	SVM
Accuracy	0.973	0.885	0.924	0.946
Recall	0.941	0.852	0.897	0.913
F1-score	0.982	0.876	0.919	0.938
FPR	0.02	0.063	0.048	0.039
FNR	0.035	0.102	0.076	0.068
FAR	0.036	0.095	0.062	0.054
Training time	128.4s	20.3s	150.6s	142.7s

In Table 6, the accuracy, recall, and F1-score of the PSO-BP1 were higher than those of the other three algorithms, indicating its high accuracy in detecting network attacks. Its FPR, FNR, and FAR were lower than those of other algorithms, indicating its excellent performance in reducing misjudgment. Finally, the training time was only 128.4s, which was moderate and suitable for real-time network security situation awareness. The accuracy and recall of RBS were low, and the FPR and FNR were high, indicating that RBS relies on predefined rules and is difficult to adapt to new attacks. The training time was short, but the flexibility was poor. The research compares the resource utilization rates of three algorithms, as shown in Table 7.

Table 7: Analysis of resource utilization of three algorithms

Algorithm name	PSO-BP1	PSO-BP2	PSO-BP3
CPU usage (%)	44.7	50.3	47.9
Memory usage (MB)	256.3	280.7	265.4
Processing time (s)	0.22	0.31	0.28
R^2	0.982	0.965	0.958
MSE	0.0031	0.0052	0.0063
RMSE	0.0548	0.0721	0.0792
MAPE	0.0117	0.0149	0.0167
MAD	0.0347	0.0413	0.0446

Table 7 shows the resource utilization comparison of the three algorithms. The CPU usage of PSO-BP1 was 44.7%, which was lower than that of PSO-BP2 (50.3%) and PSO-BP3 (47.9%), indicating that it puts less pressure on system resources. The memory usage of PSO-BP1 was 256.3MB, which was lower than that of PSO-BP2 (280.7 MB) and PSO-BP3 (265.4 MB), indicating that it is more efficient in memory utilization. The processing time of PSO-BP1 was 0.22s, which was lower than that of PSO-BP2 (0.31s) and PSO-BP (0.28s), indicating that PSO-BP1 is more efficient in single data processing. Meanwhile, the PSO-BP1 has better prediction performance index values than those of the control algorithm, indicating that its prediction accuracy as well as error are better than those of the control algorithm. The improvement of model performance by online learning mechanism, incremental learning mechanism, and transfer learning mechanism is verified through ablation experiments. The results of the ablation experiment are shown in Table 8.

Table 8: Results of the ablation experiment

Experimental setup	Accuracy	Recall	FPR	FNR
Baseline model (without dynamic learning mechanism	0.887	0.858	0.053	0.103
Online learning mechanism	0.918	0.894	0.042	0.083
Incremental learning mechanism	0.929	0.902	0.035	0.071
Transfer learning mechanism	0.915	0.895	0.045	0.085
Online learning + incremental learning	0.934	0.915	0.033	0.065
Online learning + transfer learning	0.925	0.905	0.035	0.075
Incremental learning + transfer learning	0.935	0.928	0.030	0.062
Complete model	0.945	0.936	0.025	0.052

In Table 8, the complete model demonstrated significant performance advantages. Specifically, the accuracy of the complete model reached 0.945, which was a significant improvement compared to the baseline model. This indicates that the model is more precise in identifying network attacks and normal traffic. The recall also reached 0.936, meaning that the model can detect

actual attack events more effectively and reduce the possibility of missed reports. Compared with the baseline model of 0.050 and 0.100, the FPR and FNR were respectively reduced to 0.025 and 0.052, demonstrating the outstanding ability of the complete model in reducing false positives. It can more accurately distinguish attacks from normal behaviors, avoid unnecessary alerts, and ignore real threats. These improvements not only enhance the model's detection capabilities, but also increase its adaptability and stability in dynamic network environments, enabling it to better cope with emerging attack types and constantly changing network traffic. The research further verifies the influence of the momentum factor and the compression factor on the model performance through ablation experiments. The results are shown in Table 9.

Table 9: Ablation study results

Experimental Setup	Accuracy	Recall	FPR	FNR	FAR
Baseline (No Momentum, No Compression)	0.887	0.858	0.053	0.103	0.05
With Momentum	0.918	0.894	0.042	0.083	0.045
With Compression	0.929	0.902	0.035	0.071	0.04
With Momentum and Compression	0.945	0.936	0.025	0.052	0.035

In Table 9, the performance indicators of the baseline model were relatively low, with an accuracy of 0.887, a recall of 0.858, an FPR of 0.053, an FNR of 0.103, and a FAR of 0.050. When the momentum factor and the compression factor were introduced, the performance of the model was improved. After introducing both the momentum factor and the compression factor simultaneously, the performance of the complete model was the best, with its accuracy increasing to 0.945 and the recall increasing to 0.936. Finally, the performance indicators of PSO-BP and adaptive fuzzy systems, reinforcement learning, CNN, LSTM, and Transformer models are compared in different dynamic scenarios. The results are shown in Figure 6.



Figure 6: Comparison of model performance indicators in dynamic scenarios

In Figure 6, the two-color bands of the model respectively represent dynamic scenario 1 (where new attack types gradually emerge) and dynamic scenario 2 (where attack features change dynamically). The results show that the PSO-BP model demonstrates significant advantages. In scenario 1 where new attack types gradually emerged, the accuracy of the PSO-BP model reached 0.942, the recall was 0.925, the F1-score was

0.933, and the FPR and FNR were 0.024 and 0.052, respectively. This indicates that the PSO-BP model can efficiently identify new attack types. In scenario 2 where attack features change dynamically, the accuracy of the PSO-BP model was 0.940, the recall was 0.920, the F1-score was 0.930, and the FPR and FNR were 0.025 and 0.055, respectively. This further proves the excellent adaptability and stability of the PSO-BP model in attack features. In conclusion, the PSO-BP model significantly outperforms other models in dynamic attack scenarios, such as adaptive fuzzy systems, reinforcement learning-based methods, CNN, LSTM, and Transformer, fully demonstrating its superior performance in NSSA.

3.2 Analysis of network security situation awareness based on PSO-BP algorithm

In the above research, it is explored that the proposed PSO-BP has better performance indicators on the testing set. Furthermore, the PSO-BP is characterized on the testing set to evaluate its perception performance for six types of attacks. In addition to accuracy, the research introduces FPR, FNR, and FAR. The accuracy of network attack situation recognition based on the three algorithms is shown in Figure 7.

In Figure 7, the recognition accuracy of the proposed PSO-BP for all attack types was significantly higher than that of the other algorithms. The accuracy of A6 was as high as 98.9%, which represents an attack type like worms, referring to the self-replicating malicious software that destroys the system by occupying network bandwidth. Worm network attacks mainly spread by scanning for vulnerabilities in the network, such as operating system vulnerabilities and software vulnerabilities. After infecting the target device, they automatically replicate and continue to search for new targets. The type of network attack with the lowest recognition rate was A2, which stands for denial-of-service attack. It refers to the attack that limits the availability of the service by overloading the target server or network, ultimately making it unusable. In addition, DoS attacks may mask attack characteristics through distributed attack sources. The first mode of DOS attack is SYN Flood, that is, sending a large number of ping command packets to the target host to consume the resources of the target host. Specifically, the statistical analysis results of the accuracy for different attack types are shown in Table 10.

Table 10: Statistical analysis results of model accuracy

Attack type	PSO-BP 1	PSO-BP 2	PSO-BP 3	*p* (PSO-BP1 vs PSO-BP2)	*p* (PSO-BP1 vs PSO-BP3)
A1	0.988	0.95	0.955	0.023	0.031
A2	0.956	0.92	0.925	0.045	0.038
A3	0.986	0.955	0.96	0.018	0.027
A4	0.959	0.93	0.935	0.029	0.039
A5	0.958	0.932	0.938	0.033	0.041
A6	0.989	0.965	0.97	0.012	0.02
Average accuracy	0.973	0.942	0.948	0.025	0.035

In Table 10, the accuracy of the PSO-BP1 for different types of attacks was 98.8%, 95.6%, 98.6%, 95.9%, 95.8%, and 98.9%, respectively. Its average accuracy was 97.3%, which was an average improvement of 8.8% and 4.7% compared with the other two algorithms. Meanwhile, on all attack types, the *p* values were all less than 0.05, indicating that the accuracy of PSO-BP1 against different attack types was significantly better than that of PSO-BP2 and PSO-BP3. The FPR and FNR values of three algorithms for different types of attacks are shown in Figure 8.

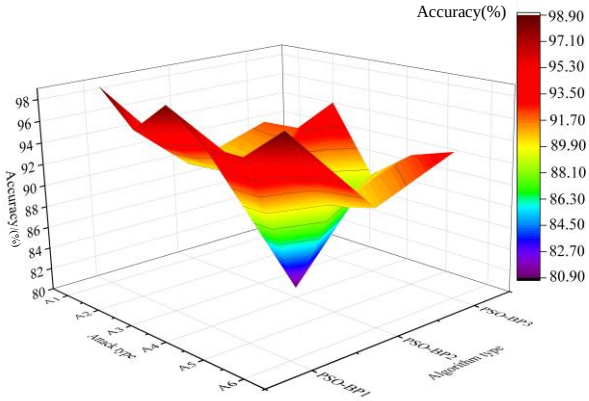
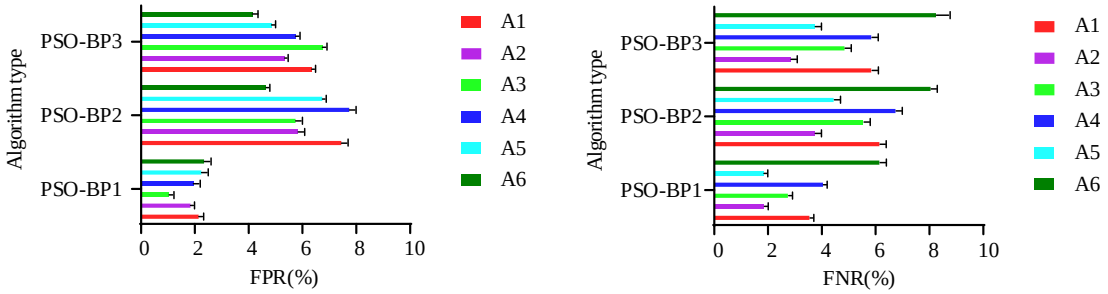


Figure 7: The accuracy of three algorithms based on various attack types



(a) Comparison of FPR values of three algorithms

(b) Comparison of FNR values of three algorithms

Figure 8: Comparison of FPR and FNR values of three algorithms based on different attack types

In Figure 8 (a), the overall FPR value of the PSO-BP1 was the lowest, and the FPR value for A3 was also the lowest. This represents vulnerability exploitation, mainly referring to conducting cyber attacks by taking advantage of software vulnerabilities. The FPR values of the PSO-BP1 were 2.1%, 1.9%, 1.1%, 2.0%, 2.3% and 2.4% respectively. The average FPR value was 2.0%, which was on average 4.5% and 3.5% lower than that of the other two algorithms. In Figure 8 (b), the PSO-BP1 has the lowest FNR value, among which A5 had the lowest FPR value, which represents the vulnerability code. By exploiting the vulnerability in the system, it enables intruders to run arbitrary code on the system. The FNR values of the PSO-BP1 were 4.1%, 1.9%, 2.8%, 4.1%, 1.9% and 6.2%, respectively, with an average FNR value of 3.5%. The FAR values of the three algorithms against different attack types are shown in Figure 10.

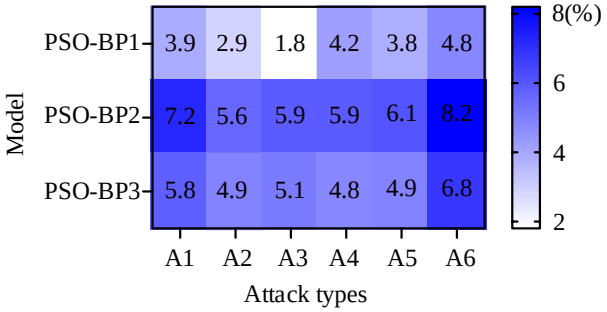


Figure 9: FAR values of three algorithms based on different attack types

Figure 9 compares the FAR values of three algorithms based on different types of network attacks. The PSO-BP1 had significantly lower FAR values than the other two algorithms in all attack types, with A3 having the lowest FAR value. Meanwhile, the PSO-BP2 algorithm had the highest FAR value for all types of network attacks, followed by PSO-BP3. The algorithm had FAR values of 3.9%, 2.9%, 1.8%, 4.2%, 3.8%, and 4.8 for all types of attacks, with an average FAR value of 3.6%, which was 2.9% and 1.8% lower than the average FAR values of the other two algorithms. The processing time, resource occupation, detection accuracy, and computational cost of the PSO-BP1 model in large-scale network environments are compared with those of the intelligent adaptive controller and the fuzzy-neural synchronization method. The results are shown in Table 11.

Table 11: Comparison results of performance indicators

Model	Processing time (s)	CPU usage (%)	Memory usage (MB)	Computational cost (s)
PSO-BP1	0.21	45.5	260.3	130.6
Intelligent adaptive controller	0.32	55.9	302.9	152.6
Fuzzy-neural synchronization method	0.28	50.2	281.3	139.8

In a large-scale real-time network environment, the PSO-BP1 model demonstrated significant performance advantages. Specifically, the processing time of the PSO-BP1 model was 0.21 seconds, while the intelligent adaptive controller was 0.32 seconds, and the fuzzy-neural synchronization method was 0.28 seconds, indicating that the PSO-BP model could more effectively handle high-dimensional data and massive network traffic. The CPU usage of the PSO-BP1 model was 45.5% and the memory usage was 260.3MB, both of which were lower than those of the intelligent adaptive controller (CPU 55.9%, memory 302.9MB) and the fuzzy-neural synchronization method (CPU 50.2%, memory 281.3MB). It is indicated that the PSO-BP1 model is more efficient in resource utilization, which is suitable for deployment in resource-constrained environments. Furthermore, the computational cost of the PSO-BP1 model was 130.6 seconds, further demonstrating its high efficiency in the training and inference processes. The research further compares the delay indicators of the three methods, among which the real-time threshold for processing delay was 0.5 seconds, the real-time threshold for system delay was 1.0 second,

the real-time threshold for transmission delay was 0.2 seconds, and the real-time threshold for queuing delay was 0.3 seconds. The comparison of delay indicators is shown in Figure 10.

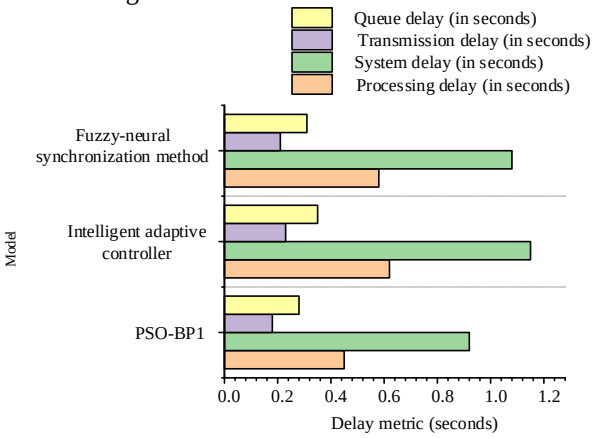


Figure 10: Comparison of delay indicators

In Figure 10, the PSO-BP1 model outperformed the other two methods in all delay indicators and was all below the real-time threshold, indicating its good performance in real-time. Specifically, its processing delay was 0.45 seconds, the system delay was 0.92 seconds, and the transmission delay and queuing delay were 0.18 seconds and 0.27 seconds, respectively. The intelligent adaptive controller has processing delay and system delay slightly higher than the real-time threshold, indicating that its performance in real-time is slightly poor. The fuzzy neural synchronization method has a system delay slightly higher than the real-time threshold, indicating that it performs moderately on real-time performance.

4 Discussion

In the research results, the loss function value of the PSO-BP1 model basically tends to level off when the number of iterations exceeds 30, and eventually converges to around 0.01. In contrast, other models such as the BP algorithm and the traditional PSO-BP algorithm have higher loss function values and slower convergence speeds under the same number of iterations. Meanwhile, the PSO-BP1 model can receive new data in real-time and update parameters through the online learning mechanism, enabling it to quickly adapt to the dynamically changing network environment. In scenarios where new attack types gradually emerge, the accuracy of the PSO-BP1 model reaches 0.942, the recall is 0.925, and the F1-score is 0.933. This indicates that the model can effectively identify new attack types while maintaining a low FPR and FNR. To evaluate the performance of the PSO-BP1 model more comprehensively, the research conducts a quantitative comparison between the PSO-BP1 and the baseline model. Specifically, the accuracy of PSO-BP1 is superior to 0.92000 of the GWO + CW-RNN model, 0.648 of the quadratic SVM model, and 0.94 of the QUEST model, but slightly lower than 0.98 of the hierarchical model + grey relational analysis. Faced with different attack types, the performance of the PSO-BP1 model in DoS is relatively lower than that of other attack types. The possible reason is that the sample size may be much larger than that of other attack types, leading to over-fitting of A2 attack type features during the training process. This data imbalance may cause the model to have deviations when distinguishing normal traffic from DoS attack traffic, thereby affecting the detection accuracy. Attackers may use camouflage techniques to hide the features of DoS attacks, making them appear as normal traffic. Currently, Mulumba D M et al. proposed a coal mine safety risk assessment model based on the PSO-BP algorithm to address certain risks in coal mining. Momentum was introduced to optimize gradient descent. Compared with other models, the neural network model had a smaller error value, with a mean square error of only 0.0002 and an average absolute percentage error of only 4.3. This research demonstrates the feasibility of the momentum-based optimization gradient descent [19]. Liu X et al. proposed a perceptual model of variational mode decomposition-PSO-BP applied to predict the permeability index. A compression factor was introduced into the particle swarm algorithm for optimization. The composite prediction model improved the accuracy by 3% compared with traditional prediction methods. The research demonstrated the feasibility of introducing a compression factor into the PSO [20]. Compared with references [19] and [20], although the PSO-BP algorithm combined with momentum and compression factor has been explored in previous literature, the research has made significant contributions in the following aspects. Firstly, a hybrid update strategy is proposed. By randomly choosing the PSO or BP algorithm for parameter update, the global search and local search capabilities have been effectively balanced, avoiding the problem that the traditional BP algorithm is prone to fall into local optimum. Secondly, an online learning mechanism and transfer learning strategy are introduced, enabling the model to adapt to new attack types in real-time and quickly adjust parameters. Finally, the data features are preprocessed and evaluated through PCA and fuzzy evaluation methods, and the weights and thresholds of the network are optimized by combining momentum and compression factors. Finally, an update mechanism can be established to enable the model to continuously learn and adapt to new attack patterns. Based on the model proposed in the research, integrating it into existing network security frameworks may enhance the overall network defense effectiveness. For example, embedding the model into existing intrusion detection systems or security information and event management systems enables them to perceive network situations and provide real-time warnings. In addition, by integrating with existing security frameworks, the model can utilize existing security data and logs to further enhance its detection capability and accuracy. In summary, the research proposes a fuzzy neural network model based

on particle swarm optimization and error back-propagation algorithm, and introduces the momentum mechanism and compression factor. This combination is highly innovative in the field of fuzzy neural networks, mainly reflected in the following aspects. Firstly, in the traditional BP algorithm, the gradient descent method is prone to getting stuck in local optimum and may experience oscillations during the convergence process. By introducing the momentum mechanism, the model can combine the velocity at the previous moment with the the current gradient direction to optimize the direction and velocity of parameter updates. Secondly, by introducing a compression factor, the model can dynamically adjust the velocity of the particles, preventing them from converging prematurely to the local optimal solution. The accuracy of the specific PSO-BP1 model reaches 0.942, the recall is 0.925, and the F1-score is 0.933. Finally, the fuzzy neural network combines the reasoning ability of fuzzy logic and the self-learning ability of neural networks, which can effectively handle the fuzziness and uncertainty of data. By introducing the momentum mechanism and compression factor into the fuzzy neural network, the adaptability and detection accuracy of the model when dealing with complex network attack data have been further enhanced.

Although the PSO-BP1 model shows certain robustness under adversarial attacks, adversarial attacks remain a serious challenge. Adversarial attacks not only can deceive the model into generating incorrect outputs, but also may lead to a decline in the model's performance on normal data. Subsequently, more advanced adversarial training methods can be explored, such as the combination of adversarial training and transfer learning, to further enhance the robustness of the model.

5 Conclusion

In the computer NSSA, the research first constructed an NSSA model, and then introduced BP algorithm and PSO algorithm. Momentum was introduced to optimize gradient descent for BP algorithm, and PSO algorithm was optimized with compression factor. Finally, they were combined to obtain PSO-BP. The relevant results showed that the proposed PSO-BP had good performance, with faster convergence speed and higher final convergence value compared with other PSO-BP algorithms. The algorithm had higher numerical accuracy and lower FAR, FPR, and FNR for different types of computer network attacks. Although significant progress has been made in the field of network security situation awareness, there are still some limitations. In the actual network environment, the types and characteristics of attacks are diverse and may occur simultaneously. A single optimization method may not be able to effectively identify and respond to all types of attacks when facing complex multi-attack scenarios, resulting in limited generalization ability and adaptability. Meanwhile, deep learning techniques can be integrated into existing PSO-BP models to further enhance their adaptability and decision-making capabilities in dynamic network environments. In addition, it can be considered to extend the PSO-BP model to the multi-agent framework to achieve distributed situation awareness. Each agent can be responsible for a sub-region in the network and improve overall situational awareness through collaboration.

References

[1] Shanmugham E K, Dhatchnamurthy S, Pakkiri P S. Adaptive activation functions with deep kronecker neural network optimized with bear smell search algorithm for preventing MANET cyber security attacks. Network: Computation in Neural Systems, 2025, 36(3): 426-450. https://doi.org/10.1080/0954898X.2024.2321391

[2] Diaba S Y, Shafie-Khah M, Elmusrati M. Cyber security in power systems using meta-heuristic and deep learning algorithms. IEEe Access, 2023, 11: 18660-18672. https://doi.org/10.1109/ACCESS.2023.3247193

[3] Badescu R. Phishing Threats in the Age of social media: A User-Centric Approach. Informatica Economica, 2024, 28(4): 83-101. https://doi.org/10.24818/issn14531305/28.4.2024.07

[4] Okeoma O, Ayoola MA, Chiedozie MO Anwuli NO, Tochukwu O, Chibuike D. Cybersecurity in U.S. and Nigeria Banking and Financial Institutions: Review and Assessing Risks and Economic Impacts. Acta Informatica Malaysia. 2023; 7(1): 54-62. https://doi.org/10.26480/aim.01.2023.54.62

[5] Du X, Ding X, Tao F. Network security situation prediction based on optimized clock-cycle recurrent neural network for sensor-enabled networks. Sensors, 2023, 23(13): 6087-6087. https://doi.org/10.3390/s23136087

[6] Alqudhaibi A, Albarrak M, Aloseel A, Jagtap S, Salonitis K. Predicting cybersecurity threats in critical infrastructure for industry 4.0: a proactive approach based on attacker motivations. Sensors, 2023, 23(9): 4539. https://doi.org/10.3390/s23094539

[7] Dalal S, Manoharan P, Lilhore U K, Seth B, Alsekait D M, Simaiya S, et al. Extremely boosted neural network for more accurate multi-stage Cyber attack prediction in cloud computing environment. Journal of Cloud Computing, 2023, 12(1): 14-26. https://doi.org/10.1186/s13677-022-00356-9

[8] Zhang H, Kang K, Bai W. Hierarchical network security situation awareness data fusion method in cloud computing environment. Journal of

computational methods in sciences and engineering, 2023, 23(1): 237-251. https://doi.org/10.3233/JCM-226542

[9] Fan Z, Zhao P, Jin B, Tang Q J, Zheng C S, Li X. Research on Key Method of Cyber Security Situation Awareness Based on ResMLP and LSTM Network. IETE Journal of Research, 2024, 70(3): 2716-2730. https://doi.org/10.1080/03772063.2023.2176365

[10] Sarjakivi P, Ihanus J, Moilanen P. Using Wargaming to Model Cyber Defense Decision-Making: Observation-Based Research in Locked Shields. European Conference on Cyber Warfare and Security. 2024, 23(1): 457-464. https://doi.org/10.34190/eccws.23.1.2270

[11] De Keersmaeker F, Cao Y, Ndonda G K, Sadre R. A survey of public IoT datasets for network security research. IEEE Communications Surveys & Tutorials, 2023, 25(3): 1808-1840. https://doi.org/10.1109/COMST.2023.3288942

[12] Guo X, Yang J, Gang Z, et al. Research on network security situation awareness and dynamic game based on deep Q learning network. Journal of Internet Technology, 2023, 24(2): 549-563. https://doi.org/10.53106/160792642023032402030

[13] Hwang Y S, Shin J. Deep Learning Assisted Intelligent Human Computer Interaction for Next Generation Internet Applications. Informatica, 2024, 48(2): 145-146. https://doi.org/10.31449/inf.v48i2.6286

[14] Saminathan K, Mulka S T R, Damodharan S. An artificial neural network autoencoder for insider cyber security threat detection. Future Internet, 2023, 15(12): 373. https://doi.org/10.3390/fi15120373

[15] Nguyen C M, Nguyen A T, Delprat S. Neural-network-based fuzzy observer with data-driven uncertainty identification for vehicle dynamics estimation under extreme driving conditions: Theory and experimental results. IEEE Transactions on Vehicular Technology, 2023, 72(7): 8686-8696. https://doi.org/10.1109/TVT.2023.3249832

[16] Gaba S, Budhiraja I, Kumar V. A systematic analysis of enhancing cyber security using deep learning for cyber physical systems. IEEE Access, 2024, 12: 6017-6035. https://doi.org/10.1109/ACCESS.2023.3349022

[17] Xu M, Zheng Y, Li Y, Wu W H. Prediction of properties of anti-breast cancer drugs based on PSO-BP neural network and PSO-SVM. Nanjing Xinxi Gongcheng Daxue Xuebao, 2023, 15(1): 51-65. https://doi.org/10.4208/JICS-2023-003

[18] Sumathi S, Rajesh R. A Dynamic BPN-MLP Neural Network DDoS Detection Model Using Hybrid Swarm Intelligent Framework. Indian Journal of Science and Technology, 2023, 16(43): 3890-3904. https://doi.org/10.17485/IJST/v16i43.1718

[19] Mulumba D M, Liu J, Hao J, Zheng Y N, Liu H Q. Application of an optimized PSO-BP neural network to the assessment and prediction of underground coal mine safety risk factors. Applied Sciences, 2023, 13(9): 5317-5317. https://doi.org/10.3390/app13095317

[20] Liu X, Zhang Y, Li X, Zhang Z F, Li H Y, Liu R, et al. Prediction for permeability index of blast furnace based on VMD-PSO-BP model. Journal of Iron and Steel Research International, 2024, 31(3): 573-583. https://doi.org/10.1007/s42243-023-01097-y

Temporal Heterogeneous Graph Neural Network for User Influence Prediction in Social E-Commerce

Kexin Yin, Jinjin Wu*
Hebei University of Engineering Science, Shijiazhuang 050091, Hebei, China
E-mail: 15203213962@163.com
*Corresponding author

Keywords: social e-commerce, dynamic heterogeneous graph, graph neural network, user influence communication, time-aware modeling

Received: July 15, 2025

This study proposes a user influence propagation model based on dynamic heterogeneous network representation learning. The model combines multi-type nodes and semantic edge types, and adopts a learnable temporal embedding strategy to construct a dynamic heterogeneous graph, so as to realize the time-aware representation of nodes. In the representation learning stage, a multi-head attention mechanism is introduced to enhance context modeling and propagation path awareness. Meanwhile, structural contrastive learning is used as an auxiliary task to improve the discriminability of node representation, and a joint training strategy of node classification and edge prediction is adopted to enhance the generalization ability of the model. In the experimental evaluation, the proposed model is compared with several representative Graph Neural Network (GNN) models, including Graph Convolutional Network (GCN), Temporal Graph Attention Network (TGAT), and Heterogeneous Graph Transformer (HGT). In the overall prediction task, the proposed model achieves a Mean Squared Error (MSE) of 0.1214, a Mean Absolute Error (MAE) of 0.2398, and an R^2 of 0.701. All these metrics are significantly better than those of TGAT (MSE: 0.1597, MAE: 0.2785, R^2: 0.603) and HGT (MSE: 0.1429, MAE: 0.2652, R^2: 0.642). Within a 14-day prediction window, the model maintains an error rate of 0.1390, demonstrating superior temporal generalization capability. For identifying high-influence user groups, the model achieves an MSE of 0.1268, significantly better than HGT's 0.1456, indicating higher sensitivity in modeling strong propagation nodes. This shows that the combination of these methods enables the proposed model to demonstrate stronger temporal modeling capability and heterogeneous relationship understanding capability compared with TGAT and HGT, in terms of capturing users' historical behaviors and predicting future influence.

Povzetek: Študija predstavi model za napovedovanje vpliva uporabnika, ki z večglavo pozornostjo in kontrastnim učenjem doseže nižje napake ter boljšo časovno učinkovitost kot obstoječi GNN-ji.

1 Introduction

With the rapid development of mobile internet and the widespread adoption of content-based social platforms, social commerce has emerged as a key component of the digital economy, integrating "people, products, and scenarios" [1,2]. Within this ecosystem, users actively or passively participate in the dissemination of product information through behaviors such as posting images and videos, commenting, liking, and sharing, thereby significantly enhancing information flow and product recommendation efficiency on platforms [3]. User influence, as a critical factor driving content dissemination and consumer decision-making, has attracted increasing attention from both platform operators and academic researchers [4,5]. Accurately modeling user influence not only facilitates the identification of potential Key Opinion Consumer (KOC) and Key Opinion Leader (KOL), but also serves as a fundamental support for improving personalized recommendations, content distribution efficiency, and marketing return on investment (ROI) [6].

However, existing research on user influence modeling still faces three key limitations. First, most approaches rely on static graph structures to model user relationships, which fail to capture the dynamic evolution of user behaviors and social structures, making them inadequate for real-time information propagation scenarios [7,8]. Second, the majority of models employ homogeneous graphs, which cannot represent the heterogeneous semantic relationships among different types of nodes such as users, products, and behaviors [9]. In social commerce, influence is rarely a single-dimensional phenomenon; rather, it often arises from complex behavioral sequences. For example, "User A purchases a product and writes a review, which is liked by User B and later shared by User C". Such behavioral chains involve highly heterogeneous nodes and relations [10]. Third, traditional rule-based influence evaluation methods lack flexibility and struggle to generalize in the

data-rich, behaviorally diverse contexts of social commerce platforms [11].

To address the aforementioned challenges, this study proposes a dynamic heterogeneous network -based representation learning model for modeling user influence propagation in social e-commerce. To clarify the research direction, this study proposes the following research questions: Can the introduction of heterogeneous edge types improve the accuracy of influence prediction? Can the multi-head attention mechanism enhance the propagation awareness ability of node representation in dynamic heterogeneous networks? How does the proposed model perform in identifying high-influence users compared with existing graph neural network (GNN) models? The main innovations and contributions of this study are as follows:

(1) The dynamic heterogeneous network structure is introduced into the modeling of user influence propagation in social e-commerce, realizing the joint modeling of multi-type nodes (including users, commodities, behaviors, and social relationships) and their temporal evolution.

(2) A representation learning framework integrating heterogeneous GNN and time-aware mechanism is designed, which effectively improves the deep semantic extraction ability of dynamic user representation.

(3) A propagation prediction model based on representation learning is proposed. Without relying on manual feature construction, the model can accurately estimate the propagation range and influence intensity of users, significantly enhancing the scalability and application value of the model.

2 Literature review

In recent years, numerous studies have attempted to characterize user influence in social networks by constructing propagation models [12]. Traditional models such as the Independent Cascade (IC) model and the Linear Threshold (LT) model have been widely adopted for modeling information diffusion [13]. For instance, Zhou et al. [14] suggested that both IC and LT models described the process of information spreading from one node to its neighbors through probabilistic mechanisms, offering strong theoretical interpretability. However, these models heavily relied on manually predefined propagation probability parameters, making them poorly suited to the dynamic nature of social behaviors. With the advancement of machine learning techniques, some studies have shifted toward using user behavior data for influence prediction. For example, Shao et al. [15] developed a user propagation probability model based on historical behavioral logs and improved prediction performance through supervised learning. Similarly, Sunarya et al. [16] proposed a "social influence factor" model that incorporated variables such as social structure and behavioral patterns to predict influence relationships among users. Nevertheless, these approaches are typically based on homogeneous network structures and often overlook temporal dynamics and the structural heterogeneity of multi-type interactive behaviors, limiting their applicability to the heterogeneous propagation mechanisms found in modern social e-commerce platforms.

Heterogeneous information network has been widely applied in user behavior modeling and recommendation systems due to their capability to represent multi-type entities and relations. Chen et al. [17] defined the multi-type characteristics of nodes and edges in heterogeneous graph structures and introduced meta-paths as an essential tool for capturing semantic relationships. Kumar and Krishna [18] further proposed the Metapath2Vec algorithm, which generated context sequences through meta-path-guided random walks to enhance the semantic richness of node representations. In recent years, deep HGNN has become increasingly prominent. For instance, the heterogeneous graph attention networks proposed by Mei et al. [19] and Wang et al. [20] leveraged attention mechanisms at both the node type and meta-path levels to improve recommendation accuracy and behavior prediction performance. Jia et al. [21] introduced the Heterogeneous Graph Transformer (HGT), which employed a multi-head attention mechanism aware of node and edge types to further enhance the quality of node representations in heterogeneous graphs. These methods provide a solid technical foundation for modeling the diverse entity relationships among users, products, content, and behaviors in social e-commerce scenarios. However, it is important to note that most of these approaches are based on static graph structures and fail to capture the temporal evolution of user behaviors.

For another example, Gao et al. [22] proposed a social influence prediction method based on Heterogeneous Graph Neural Network (HGNN). By utilizing historical event topic features and user interest features, they constructed an end-to-end heterogeneous neural network model to predict users' behaviors in social networks more accurately. This method provided a reference technical foundation for modeling different types of nodes (users, commodities, behaviors) and their interaction relationships in social e-commerce platforms. Zhong & Huang [23] proposed a dynamic graph representation learning method based on temporal graph transformer. This method could effectively retain high-order neighborhood information and capture the temporal evolution characteristics of user behaviors through a time-aware mechanism, providing a new idea for the modeling of user influence propagation in social e-commerce.

However, the multi-scale attention graph transformer framework proposed by Qiu [24], although applied to computed tomography (CT) image analysis, combined the ideas of graph structure modeling and multi-head attention mechanism. It provided a technical reference for processing complex structures and multi-level contextual information, and could be used for reference in the fine-grained modeling of node features and behavioral interactions in dynamic social e-commerce networks. Zhang [25] constructed a user preference model based on GNN and multi-layer attention mechanism, which was applied to social access control management. By learning users' social behaviors and preferences to optimize system strategies, the ideas of this model for user behavior representation and influence analysis could directly

inspire the identification of high-influence users and propagation prediction in social e-commerce platforms. Table 1 summarizes these related works:

Table 1: Summary of related work on social network user influence modeling

Literature	Method	Graph Type	Dynamic/Static	Limitations
Zhou et al. [14]	IC / LT	Homogeneous	Static	Relies on manually set propagation probability and cannot handle dynamic behaviors
Shao et al. [15]	User propagation probability model	Homogeneous	Static	Ignores multi-type behaviors and temporal evolution
Sunarya et al. [16]	Social Influence Factor	Homogeneous	Static	Lacks heterogeneity and time awareness
Chen et al. [17]	Heterogeneous information network+metapath	Heterogeneous	Static	Static graph, without considering temporal evolution
Kumar & Krishna [18]	Metapath2Vec	Heterogeneous	Static	Static graph, without capturing dynamic behaviors
Mei et al. [19]/ Wang et al. [20]	Heterogeneous graph attention network	Heterogeneous	Static	Static graph, lacking time awareness
Jia et al. [21]	HGT	Heterogeneous	Static	Static graph, difficult to handle dynamic evolution
Gao et al. [22]	HGNN	Heterogeneous	Static	Static graph, without modeling temporal evolution
Zhong & Huang [23]	Time diagram converter	Heterogeneous	Dynamic	Limited handling of node type heterogeneity
Qiu [24]	Multiscale attentional diagram converter	Heterogeneous	Dynamic	Original application is image analysis, requiring migration to social e-commerce scenarios
Zhang [25]	GNN+multilayer attention mechanism	Heterogeneous	Static	Static graph, needs improvement for dynamic user influence modeling

Although existing studies have made progress in user influence modeling, heterogeneous graph representation learning, and dynamic graph analysis, several research gaps and technical challenges remain. First, there is a lack of a unified framework that integrates dynamics, heterogeneity, and behavioral diversity to comprehensively model user influence propagation. Second, most current dynamic graph models are not structurally optimized for the unique characteristics of product propagation in social e-commerce scenarios, making them inadequate for explaining user influence differences under complex behavioral paths. Third, existing models still face limitations in interpretability and deploy ability, lacking efficient mechanisms to support high-frequency propagation prediction and recommendation decision-making. To address these limitations, this study aims to construct a representation learning framework that integrates both dynamic and heterogeneous features, enabling accurate capture of influence propagation characteristics within complex and multi-faceted behavior networks. This approach seeks to bridge the gap in comprehensively modeling user influence mechanisms in social e-commerce environments and to provide more robust technical support for precision marketing and intelligent recommendation.

3 Research methodology

3.1 Overall framework of model

To address the challenge of simultaneously modeling structural heterogeneity and temporal dynamics in user influence propagation within social e-commerce scenarios, this study proposes an end-to-end framework based on dynamic heterogeneous network representation learning. The goal is to effectively capture the multi-type relationships among users, content, and products, while modeling users' propagation potential throughout temporal evolution. The proposed model takes as input multi-source behavioral logs collected from social e-commerce platforms, including user-to-user interactions (e.g., likes, comments, shares), user-product interactions

(e.g., views, purchases, favorites), and the corresponding timestamps of these actions. During the preprocessing stage, these behaviors are abstracted into a heterogeneous graph consisting of multiple types of nodes (e.g., users, products, behavioral events) and multiple types of edges (e.g., social relations, product interactions, content propagation paths). The graph is then temporally segmented or constructed as an event stream based on behavioral timestamps, resulting in a sequence of dynamic heterogeneous graphs that preserve temporal evolution characteristics.

For example, within a specific time window, suppose that user A performs the following operations during this period: liking a product X (U-I edge), commenting on a product Y (U-I edge), and sharing information about a product (U-U edge). The timestamp order of these behaviors determines how to construct graph nodes and edges within this time window. Each node (e.g., user, product, behavior) is assigned a timestamp, which serves as a snapshot of the graph at that specific time point. Subsequently, the system splits the behavior data into multiple time periods (such as daily or hourly intervals), where each time period corresponds to one graph snapshot. This enables the dynamic construction of the graph from the temporal dimension.

In the representation learning module, a HGNN framework is introduced and integrated with temporal awareness mechanisms to capture both structural and behavioral representations of each node across different time periods. Specifically, the model employs type-specific transformation functions to handle semantic differences among various types of nodes and edges. Meanwhile, a temporal encoding module (such as positional encoding or time-gating mechanisms) is incorporated to model the sequential characteristics of nodes within behavioral trajectories. As a result, each user is represented by a low-dimensional embedding in the dynamic network that jointly captures their current semantic state and historical behavioral patterns. This representation not only reflects the user's interest preferences and social position, but also encodes their potential influence on information propagation within a given time window.

In the propagation modeling stage, this study constructs a prediction network based on node embeddings to regress and forecast users' influence levels within a future time window. This prediction module takes as input the users' time-aware embeddings and integrates graph structural features of their upstream and downstream propagation paths. Influence intensity is then estimated via attention-weighted aggregation or a Multi-Layer Perceptron (MLP). The output may represent the user's propagation scope, the number of influenced individuals, or the volume of conversion actions triggered, serving downstream tasks such as key node identification, personalized recommendation, or marketing optimization. The entire framework operates as a unified end-to-end architecture, enabling joint optimization of graph representation learning and propagation prediction during training to enhance embedding discriminability and forecasting accuracy. Additionally, the model offers strong scalability and interpretability, allowing for insight into influence sources through analysis of node attention weights and propagation path structures. Figure 1 illustrates the overall architecture.

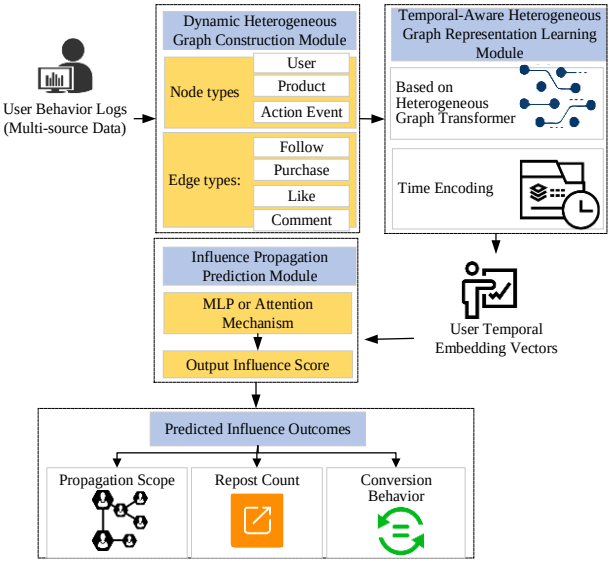


Figure 1: Model overall architecture diagram

Figure 1 shows that the first key step is to convert behavior logs into heterogeneous graphs through the data preprocessing process. Then, these graphs are divided into multiple snapshots via time slicing to form a sequence of dynamic heterogeneous graphs. Each snapshot represents the manifestation of user behaviors within a specific time window. Subsequently, the snapshots are fed into the representation learning module, which combines a time-aware mechanism to generate low-dimensional embeddings of nodes. Finally, these embeddings are used in the propagation prediction module. Figure 2 shows the algorithm pseudocode for the entire process.

```
Input:  - Multi-source user behavior logs (user-user interactions, user-item interactions, timestamps)  - Time window size Δ
Output:  - Predicted user influence scores for future time windows
# 1. Data Preprocessing
Categorize nodes as users, items, and behavior events
Construct edges: U-U, U-I, U-Action, each with timestamp
Slice behavior logs into time windows to form dynamic graphs {G_1, G_2, ..., G_T}
# 2. Dynamic Heterogeneous Graph Construction
For each time window G_t in {G_1, ..., G_T}:
    Build heterogeneous graph with multiple node and edge types
    Encode temporal information for each node using learnable position embeddings
# 3. Time-aware Representation Learning
For each node v in G_t:
    Apply heterogeneous graph neural network (HGNN)
    Integrate temporal encoding to generate time-aware node embedding h_v^t
# 4. User Influence Propagation Prediction
For each user node embedding h_v^t:
    Aggregate neighbor information via attention mechanism
    Feed aggregated representation into Multi-Layer Perceptron (MLP)
    Output predicted influence score ŷ_v^(t+Δ)
# 5. Model Training
Optimize prediction with Mean Squared Error (MSE) loss
Optionally include auxiliary tasks (node classification, edge prediction) for joint learning
Return: Predicted user influence scores for each future time window
```

Figure 2: Pseudo-code of overall algorithm flow

3.2 Dynamic heterogeneous graph construction and representation learning

Before constructing the dynamic heterogeneous graph, time slicing is first performed on user behavior logs. Based on the timestamp information of user behaviors, the behavior data is divided into multiple time windows. For example, each time window is set to 7 days. Within each time window, user behaviors are mapped to nodes and edges in the graph. Each graph slice contains the user behavior data in this time period, as well as social relationships between users, and relationships between users and products. Each edge is also attached with a timestamp to capture the temporal characteristics of information propagation. In this way, a series of dynamic graphs are constructed, where each graph represents the network state within time window t.

In social e-commerce platforms, user behaviors exhibit significant heterogeneity and temporal dynamics. Users interact with products and others through diverse actions such as likes, comments, and purchases, which evolve distinctly over different time points. To comprehensively model this structure, we construct dynamic heterogeneous graphs integrating multiple types of nodes including users, items, and action events, as well as their heterogeneous interactions. Each edge in the graph is labeled not only with semantic types (e.g., "purchase," "like," "follow") but also with precise timestamps, capturing the temporal characteristics of information propagation. The entire graph is represented as a time-evolving sequence of graphs $\{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_T\}$, where each graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$ corresponds to a snapshot of the network structure within a specific time window.

To learn node representations with rich semantic expressiveness and temporal awareness from such graphs, this study employs HGNN as the foundational modeling approach and incorporates a temporal encoding mechanism to construct time-aware embeddings. The message passing mechanism is based on HGT. At each layer, the update of node representations considers the structural information of neighbor nodes, and performs different weighting processes according to edge types. For example, the "purchase" relationship may be more important than the "like" relationship for the spread of user influence.

Each edge type is associated with a corresponding weight matrix to handle information transmission. In addition, an attention mechanism is introduced, enabling the model to automatically learn the importance of different neighbor nodes, thereby effectively improving the accuracy of propagation prediction. The representation of each node is updated by weighted aggregation of its neighbors' representations, where the weights are dynamically calculated by the attention mechanism. By introducing time-aware encoding and the time difference of neighbors' behaviors on the basis of HGT, the attention parameters consider node types and edge types, and adjust the propagation characteristics of time series. This enhances the dynamic adaptability of node representations. Additionally, a temporal difference function is introduced to encode the temporal features of neighboring nodes' behaviors. At each network layer, the node representation update can be formally defined as:

$$\mathrm{h}_v^{(l+1)} = \sigma\left(\sum_{r\in\mathcal{R}} \sum_{u\in\mathcal{N}_r(v)} \alpha_{uv}^{(r)} \cdot W_r^{(l)} \mathrm{h}_u^{(l)}\right) \quad (1)$$

$\mathcal{R}$ represents the set of all edge types. $\mathcal{N}_r(v)$ represents the set of neighbor nodes adjacent to node v through relationship type r. $W_r^{(l)}$ is the weight matrix of the corresponding edge type at the l-th layer. σ is the activation function. $\alpha_{uv}^{(r)}$ is the attention weight of edge (u,v), which combines the neighbor representation and edge type.

To improve the time modeling ability of the model, the timestamp difference Δt_{uv} is introduced into the attention weight calculation. This study adopts a learnable positional encoding technique. For each discrete timestamp t, a vector with a fixed dimension is learned as the embedding representation of the temporal position. For each time slice, the corresponding temporal vector is a trainable parameter of the model, and its dimension is consistent with that of the node embedding. Different from the fixed sine/cosine positional encoding in Transformer, this vector is optimized through backpropagation during the training process, thereby enabling adaptive capture of the patterns of user behaviors changing over time.

In the calculation of node attention, the node's own representation h_v, the neighbor node's representation h_u, and the temporal encoding vector $\phi(\Delta t_{uv})$ are concatenated first. The concatenated result is then input into the linear transformation matrices W_q and W_k. Combined with the attention parameter a_r, the attention score $\alpha_{uv}^{(r)}$ is calculated. Through this process, structural semantic information and temporal information are integrated into the node representation simultaneously, enabling time-aware modeling of neighbor relationships in the dynamic heterogeneous graph. The multi-head attention mechanism acts on both node types and edge types simultaneously, performing independent calculations for different types of neighbors within each graph layer. Time awareness is integrated into the attention calculation of each edge through learnable positional encoding, rather than using multi-head attention independently directly on time slices.

The learnable positional encoding can flexibly reflect behavior patterns under non-uniform time intervals. For scenarios where user behaviors on social e-commerce platforms are sparse or high-frequency, it provides more accurate temporal feature representation, thereby improving the discriminative ability of node representation and the accuracy of subsequent propagation prediction tasks. The time-aware attention mechanism is expressed as:

$$\alpha_{uv}^{(r)} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\mathbf{a}_r^{\top}\left[W_q\mathbf{h}_v \| W_k\mathbf{h}_u \| \mathrm{LearnablePE}(\Delta t_{uv})\right]\right)\right)}{\sum_{u'\in\mathcal{N}_r(v)} \exp(\cdot)} \quad (2)$$

$\mathrm{LearnablePE}(\Delta t_{uv})$ represents a learnable time position coding vector, $\|\cdot\|$ indicates the vector stitching operation, and a_r is the attention parameter under the edge type r. The above mechanism makes the representation of nodes at each time point not only integrate the structural semantics of multi-type neighbors, but also include the

time influence of propagation behavior. It provides more discriminating input features for subsequent propagation prediction tasks.

During training, structural contrastive learning (SCL) is employed as an auxiliary objective to enhance the stability and discriminative power of representation learning. Specifically, positive neighbor pairs are sampled from the graph, representing nodes connected by real edges, while negative pairs are randomly selected node pairs with no direct connection. The goal is to maximize the similarity between positive pairs and minimize the overlap between negative pairs. The objective function is defined as:

$$\mathcal{L}_{\text{contrast}} = -\log \frac{\exp\left(\frac{\sin(h_v, h_u^+)}{\tau}\right)}{\exp\left(\frac{\sin(h_v, h_u^+)}{\tau}\right) + \sum_{u^-} \exp\left(\frac{\sin(h_v, h_{u^-})}{\tau}\right)} \quad (3)$$

sim (·) represents cosine similarity, τ is temperature coefficient, and u^+ and u^- are positive sample and negative sample neighbors respectively. Finally, the time-aware embedded representation of nodes will be used as the input of the communication prediction module to realize the quantitative modeling of users' future influence. Figure 3 shows the whole dynamic graph representation learning process.

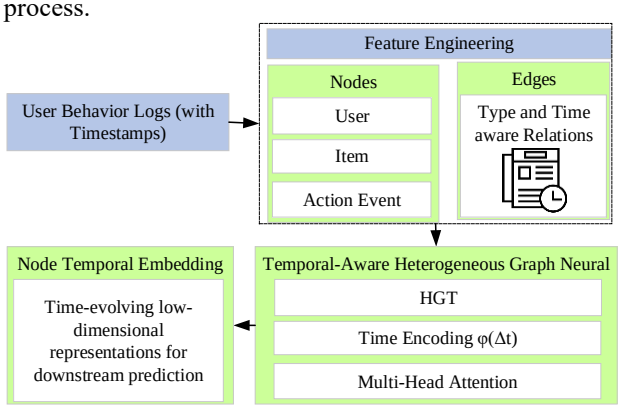


Figure 3: Construction of dynamic heterogeneous graph and learning flow chart of time-aware representation

3.3 Modeling of user influence communication

After obtaining the time-aware representation of user nodes in the dynamic heterogeneous graph, this study further constructs a user influence propagation modeling module to estimate information propagation potential in the future time window. User influence is defined as the sum of the number or weight of secondary communication behaviors (such as likes, comments, forwarding, etc.) caused by their behaviors or contents within a specific time range. The modeling task can be reduced to a time-sensitive regression prediction problem, and the goal is to estimate the possible propagation influence score $\hat{y}_v^{t+\Delta}$ in the future time period [t, t+Δ] based on the embedded vector h_v^t of the user node at a given time point.

To achieve this goal, this study designs a communication prediction module that integrates structural perception and historical behavior aggregation. Firstly, the module takes h_v^t as the basic input, and combines its neighbor node representation on the propagation path in the previous time period, and integrates the neighbor propagation contribution by weight through the attention mechanism to generate a representation vector with contextual propagation awareness. Then, the vector is input to an MLP, and the estimated influence of users in the future time period is output, as Figure 4 illustrates. MLP consists of two layers. Each layer uses the ReLU activation function, and the dropout rate is set to 0.2.

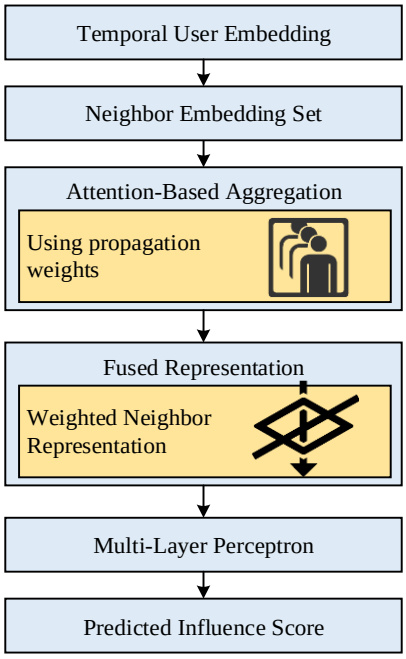


Figure 4: Structure diagram of user influence propagation modeling

The whole process can be formally expressed as follows:

$$\hat{y}_v^{t+\Delta} = \text{MLP}\left(h_v^t + \sum_{u \in \mathcal{N}_v^t} \beta_{uv} \cdot h_u^t\right) \quad (4)$$

$\mathcal{N}_v^t$ represents the neighbor set connected with the user v by the propagation path at time t. β_{uv} is the importance weight of the neighbor node u for the propagation contribution, which is given by the following equation:

$$\beta_{uv} = \frac{\exp\left(q^\top \cdot \tanh\left(W_1 h_v^t + W_2 h_u^t\right)\right)}{\sum_{u' \in \mathcal{N}_v^t} \exp(\cdot)} \quad (5)$$

W_1 and W_2 are the learnable weight matrices, and q is the attention projection vector. The dimension of q is 128. This mechanism not only allows the model to dynamically pay attention to the key propagation paths in the prediction, but also has certain interpretability, that is, it can trace back to which neighbor behaviors the influence comes from. In the regression task, the Mean Squared Error (MSE) is defined as the average of the squared differences between the predicted values and the true values, which is shown in equation (6).

$$\mathcal{L}_{\text{pred}} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} (\hat{y}_v^{t+\Delta} - y_v^{t+\Delta})^2 \quad (6)$$

$\mathcal{L}_{\text{pred}}$ represents the loss function value of the model in the prediction task, i.e., the mean square of prediction errors, which is used to optimize model parameters. V denotes the set of nodes, representing all user nodes in the graph. $|\mathcal{V}|$ indicates the size of the node set, i.e., the total number of users, which is used to normalize the error. v ∈ V represents the current user node being indexed. $\hat{y}_v^{t+\Delta}$ denotes the propagation influence score of user v in the future time window [t, t+Δ] predicted by the model. $y_v^{t+\Delta}$ represents the actual observed propagation influence value of user v in the same time window. $\hat{y}_v^{t+\Delta} - y_v^{t+\Delta}$ indicates the error between the predicted value and the true value. $(\hat{y}_v^{t+\Delta} - y_v^{t+\Delta})^2$ represents the squared error, which is used to penalize predictions with larger deviations and ensure that the loss function is sensitive to large errors. $\frac{1}{|\mathcal{V}|}\sum_{v\in\mathcal{V}}(\cdot)$ denotes the average of squared errors over all nodes, obtaining the mean squared error of the entire network.

In addition, to further improve generalization ability, the model incorporates auxiliary tasks (such as node classification or edge prediction) during the propagation prediction training process as joint learning objectives. This enables the user embeddings to retain propagation semantics while gaining broader task adaptability. Through this modeling approach, the framework effectively integrates structural semantics, behavioral history, and temporal information into a unified prediction architecture, providing a reliable technical foundation for subsequent user selection, targeted marketing, and propagation path control.

3.4 Experimental design

The data used in this study is derived from user behavior logs of a large-scale social e-commerce platform, including user-user interactions (such as likes, comments, and follows), user-product behaviors (such as views, purchases, and favorites), and corresponding timestamp information. To protect privacy, all users and products have undergone anonymization processing. The data covers the time range from January 2024 to June 2024, including approximately 500,000 active users, 300,000 products, and 12 million behavior records. For the construction of dynamic heterogeneous graphs, the data preprocessing process includes the following steps:

1) Node type division: Map users, commodities, and behavior events to different types of nodes.

2) Edge type division: Establish edges according to behavior types, including user-user (U–U), user-commodity (U–I), and user-behavior event (U–Action), with timestamps of behaviors attached. Edge weights are assigned based on interaction types and frequencies to reflect the contribution of each behavior to propagation. For example, likes, comments, shares, and purchases are weighted differently to capture their respective influence intensities.

3) Time slicing: The behavior logs are segmented according to a fixed time window (7 days) to generate a time-series graph $\{G_1, G_2, \ldots, G_T\}$, where each graph G_t corresponds to a network snapshot of a time period.

4) Graph construction: Within each time window, nodes and edges are combined to form a heterogeneous graph, with node types, edge types, and temporal attributes retained. This graph serves as input for subsequent time-aware heterogeneous graph representation learning.

For missing or noisy behavior events, smoothing processing is performed through time window aggregation and neighbor weighted average strategies to ensure the stability and integrity of input node features. Figure 5 shows the pseudocode for this process:

```
# Input: user behavior logs (logs), time window length (window_size)
# Output: dynamic heterogeneous graph sequence (graphs)

graphs = []  # initialize list to store graph sequence

# Map logs to node types: user, item, action event
nodes = map_nodes(logs)

# Map logs to edge types: U-U, U-I, U-Action, with timestamp
edges = map_edges(logs)

# Slice logs into time windows
for t in range(start_time, end_time, window_size):
    G_t = build_heterogeneous_graph(nodes, edges, time_window=t)
    graphs.append(G_t)

return graphs
```

Figure 5: Pseudo-codes of data preprocessing process

For model comparison, three representative baseline models are selected: Graph Convolutional Network (GCN), Temporal Graph Attention Network (TGAT), and HGT. In terms of evaluation metrics, to quantify the model's ability to predict users' future influence in information propagation, MSE, Mean Absolute Error (MAE), and the coefficient of determination (R^2) are adopted as the primary regression performance indicators. In addition, for the auxiliary evaluation of influence estimation, continuous propagation scores are divided into three intensity categories (high, medium, and low) according to preset thresholds. Then, F1-score and AUC are calculated on these categories to evaluate the model's discriminative ability at different propagation intensity levels. Meanwhile, the category distribution and threshold division strategy are explicitly reported to ensure the transparency and reproducibility of the results.

In terms of training parameter settings, all models are evaluated under the same data partition and graph structure. The time window is set to seven days, and a sliding window approach is used to generate training and testing graph sequences. The length of the time window Δ is set to 7 days, mainly based on statistical analysis of user behavior activity. Through the analysis of user behavior logs in the dataset, it is found that the vast majority of users have at least one interaction behavior (such as liking, commenting, purchasing, etc.) within 7 consecutive days. This setting can ensure the relative completeness of node and edge information in graph snapshots, while also balancing computational efficiency. Specifically, the first 70% of the time slices are used for training, while the remaining 30% are used for validation and testing. Details

are shown in Table 2.

Table 2: Model parameter settings

Parameter category	Parameter name	Parameter value
Training parameter	Optimizer	Adam (β_1=0.9, β_2=0.999, ϵ=10^{-8})
	Initial learning rate	0.001
	Batch size	512
	Maximum number of training rounds	100
Model parameter	Embedded dimension	128
	Time window length	7 days
	Time coding mode	Learnable position embedding
Attention mechanism parameter	Number of multi-headed attention heads	4
	Attention vector dimension q	128
	Attention matrix	128×128
MLP parameter	Hidden layer number	2
	Activation function	ReLU
	Dropout rate	0.2

4 Result and discussion

4.1 Comparison of overall prediction performance of models

First, the overall regression performance of each model in predicting users' future influence scores is evaluated, using MSE, MAE, and R^2 as the primary evaluation metrics. Figure 6 presents the results.

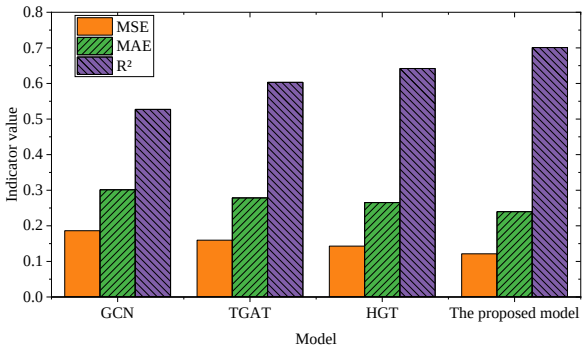


Figure 6: Overall regression performance of different models in influence prediction task

Figure 6 shows that the proposed model achieves the best performance across all evaluation metrics in the user influence propagation prediction task. Compared to the traditional static GNN model GCN, the proposed model reduces MSE and MAE by 34.8% and 20.4%, respectively, while increasing R^2 by approximately 17.4 percentage points. This indicates that the model not only provides more accurate numerical predictions but also demonstrates stronger fitting capability. Compared with the time-aware GNN TGAT, the proposed model still shows significant improvements in both MSE and MAE, suggesting that incorporating temporal factors alone is insufficient to fully capture the dynamic evolution of user influence. Although HGT, with its strong heterogeneous graph modeling capability, improves prediction performance to some extent, its disregard for temporal dynamics results in a delayed response to propagation trends. Consequently, its performance in terms of R^2 and error metrics lags behind the proposed approach. Overall, by integrating both heterogeneous structure and temporal evolution, the proposed model effectively captures complex propagation relationships between users and between users and items. This leads to more expressive node embeddings and significantly improves the accuracy of future influence prediction.

To verify the statistical significance of these performance improvements, 5 repeated experiments are conducted for each model, and a paired t-test is used to examine the differences in MSE, MAE, and R^2 among different models. The results show that the performance differences between the proposed model and GCN, TGAT, and HGT are all significant at the level of $p < 0.05$, which supports the conclusion of "significant improvement".

4.2 Prediction robustness evaluation under different time spans

To further evaluate the model's robustness across prediction tasks of varying durations, this study sets three prediction windows (Δ = 3 days, 7 days, and 14 days) and recorded the performance of each model in terms of MSE. Figure 7 presents the results.

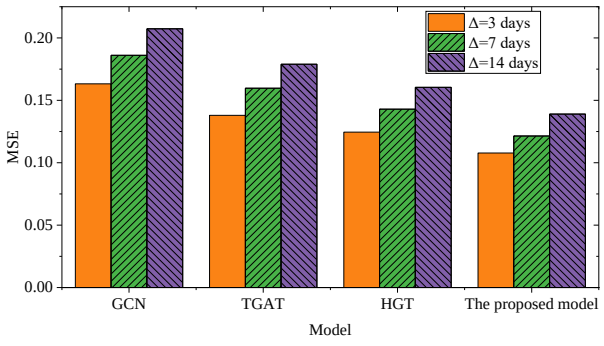


Figure 7: MSE performance under different prediction time spans

The data in Figure 7 further validate the model's robustness across different prediction horizons. As the prediction window extends from Δ = 3 days to Δ = 14 days, all models experience an increase in error, reflecting the growing uncertainty and modeling difficulty of propagation behaviors over longer time spans. However, the proposed model consistently achieves the lowest MSE across all time intervals, with the smallest error increase, demonstrating superior temporal generalization and trend extrapolation capabilities. Compared to GCN, the proposed model reduces error by 33% in the long-term prediction scenario (Δ = 14 days), and outperforms TGAT and HGT with error reductions of 22.3% and 13.3%, respectively. This advantage stems primarily from the model's dynamic temporal modeling and precise capture of user behavior patterns within heterogeneous structures, enabling user representations to reflect both current behavior states and the cumulative effect of historical actions on future propagation potential. Therefore, in evolving social e-commerce propagation scenarios, the proposed model exhibits stronger robustness and generalization in prediction stability and temporal adaptability, making it well-suited for deployment in real-world marketing monitoring and propagation effectiveness analysis systems.

Similarly, by repeating the experiment 5 times and conducting paired t-tests, it is verified that the MSE improvements of the proposed model under different prediction windows are all statistically significant ($p < 0.05$). This further demonstrates the model's robustness and temporal adaptability in dynamic heterogeneous scenarios, making it suitable for marketing monitoring and propagation effect analysis in real e-commerce environments.

4.3 The analysis of the influence of heterogeneous structure on prediction performance

To further investigate the impact of heterogeneous structural information on model performance, this study designs four different edge-type configurations. These configurations progressively incorporate user social edges (U–U), behavior event edges (U–Action), and item similarity edges (I–I) to construct dynamic heterogeneous graphs. Under a consistent structural design, only the number and semantic complexity of heterogeneous edge types are varied. The model's performance across multiple evaluation metrics under different configurations is compared. Figure 8 presents the specific results.

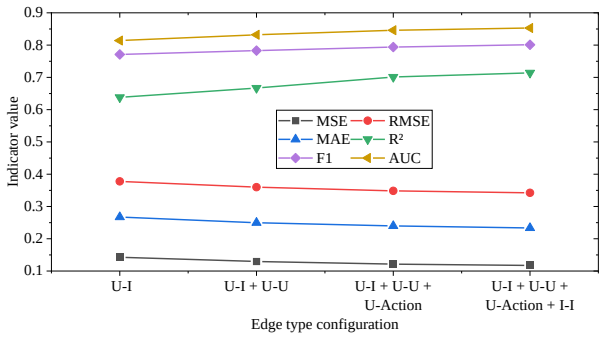


Figure 8: Influence of heterogeneous edge type configuration on model performance

Figure 8 demonstrates that incorporating heterogeneous structures significantly enhances the model's multi-dimensional performance. In the regression task, as the semantic richness of edge types increases, the model's errors measured by MSE and MAE steadily decrease, while R^2 correspondingly improves, indicating stronger fitting capability. For the auxiliary binary classification task, both the F1 score and AUC metrics for propagation event recognition show notable increases with enhanced heterogeneous structures. This suggests that the model not only achieves more accurate regression of user propagation intensity but also exhibits strong binary classification ability. Particularly, after introducing item-to-item (I–I) edges, the model captures latent user interest migration trends, resulting in higher accuracy in identifying propagation potential. These findings further confirm the importance and applicability of dynamic heterogeneous graph modeling in social e-commerce propagation scenarios.

4.4 Hierarchical analysis of user influence distribution and behavior intensity

All users in the test set are divided into three tiers based on their actual influence scores. The k-means clustering algorithm is adopted to partition the influence scores to determine the boundaries between users with high, medium, and low influence. This ensures that the tiered results can truly reflect the distribution of user influence. The MSE of each model is then calculated for these user tiers to analyze their sensitivity in recognizing varying levels of user behavior. The results are shown in Figure 9. As illustrated, the proposed model achieves the best performance across all user tiers, with particularly notable error reduction for high-influence users, where the MSE reaches 0.1268. This indicates that the model has a stronger ability to differentiate behavior intensity levels, accurately capturing propagation trends of highly influential users while reducing the risk of misclassifying extreme individual behaviors. Such capability is of significant practical importance for applications like precision marketing and user segmentation-based recommendation.

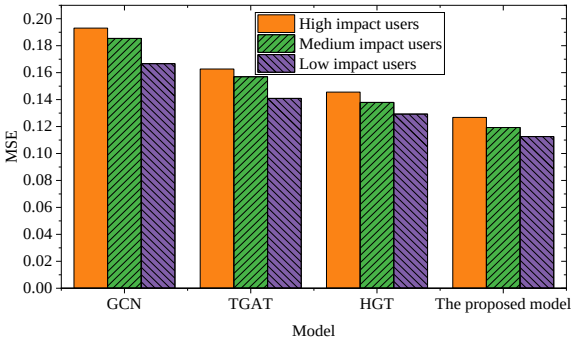


Figure 9: Prediction error under different influence levels

To enhance the model interpretability, this study further analyzes the attention weights in the propagation prediction module. For an intuitive demonstration of the mechanism's effect, the attention weights of some neighbor nodes of two high-influence users are selected as examples (Table 3).

Table 3: An example of neighbor attention weight of high-impact users

User ID	Neighbor node ID	Types of behaviors	Attention weight
U102	U87	Like	0.32
U102	U45	Comment	0.27
U102	I23	Purchase	0.21
U102	U76	Share	0.20
U215	U103	Comment	0.35
U215	I87	Purchase	0.30
U215	U112	Like	0.20
U215	U90	Share	0.15

Table 3 shows the model can dynamically adjust attention weights based on the behavior types of user neighbors and their historical propagation contributions. For instance, the neighbor nodes of the high-influence user U102 who have performed "like" and "comment" behaviors have relatively higher weights, indicating that these neighbors exert a greater impact on U102's future propagation potential. Similarly, the key neighbor nodes of user U215 mainly contribute to influence through "comment" and "purchase" behaviors.

4.5 Analysis of ablation experiment

To verify the proposed effectiveness of each module in the dynamic heterogeneous graph representation learning framework, ablation experiments are conducted on key components in this section, including the SCL module and the joint auxiliary task module (i.e., node classification and edge prediction). By comparing the performance of the complete model with the models from which each module is removed in the propagation prediction task, the contribution of each module to the model performance can be quantified, and the innovation of the method design can be verified. The definitions of different model variants are as follows: Full Model: Includes the SCL module and joint auxiliary tasks. w/o SCL (without SCL): Retains the time-aware heterogeneous graph representation learning and propagation prediction modules unchanged, but removes the auxiliary objective of SCL. w/o Auxiliary Tasks (without Auxiliary Tasks): Keeps the SCL module and propagation prediction module unchanged, but removes the node classification and edge prediction tasks. w/o SCL & Auxiliary Tasks (baseline version): Only retains the time-aware heterogeneous graph representation learning and propagation prediction modules. The experimental results are shown in Table 4:

Table 4: Ablation experimental results

Model Version	MSE	MAE	R^2
Full Model	0.1268	0.2103	0.892
w/o SCL	0.1425	0.2271	0.871
w/o Auxiliary Tasks	0.1352	0.2184	0.882
w/o SCL & Auxiliary Tasks	0.1548	0.2397	0.861

Table 4 shows that after removing the SCL module, both MSE and MAE increase significantly, indicating that structural contrastive learning helps enhance the discriminative power and stability of node representations. When the auxiliary task module is removed, MSE and MAE also rise, suggesting that joint learning enables node representations to simultaneously consider propagation prediction and structural semantics, which is beneficial for improving generalization ability. The performance of the full model is significantly better than other ablated versions, with an R^2 of 0.892, demonstrating the obvious advantages of the multi-module joint design. It is evident that both structural contrastive learning and auxiliary tasks play a significant role in improving dynamic heterogeneous graph representation learning and user influence prediction. Specifically, SCL strengthens the distinguishability of node representations, while auxiliary tasks provide additional semantic supervision, making node embeddings more robust and generalizable in the propagation prediction task. Through this ablation analysis, the contribution of each module in the overall framework can be clearly identified, verifying the innovativeness and technical effectiveness of the model design.

4.6 Computational complexity analysis

To evaluate the feasibility of the proposed model in practical deployment, this study analyzes the model's time complexity and memory consumption, and compares it with existing heterogeneous graph models, namely HGT and TGAT. The analysis focuses on both the training and inference phases. In the training phase, the proposed model, like TGAT, has a time complexity of $O(h \cdot N \cdot E \cdot d)$. The computational load mainly comes from the aggregation operation of multi-head attention on neighbor nodes, thus maintaining good scalability on large-scale graph data. In contrast, HGT needs to model multiple node types and edge types, leading to an increased complexity of $O(h \cdot N \cdot E \cdot d \cdot |R| \cdot |T|)$. When the number of edge

types and node types is large, the training efficiency of HGT decreases significantly. In the inference phase, the proposed model also maintains a comparable complexity to TGAT, i.e., $O(h \cdot N \cdot E \cdot d)$, ensuring efficient prediction capability. However, HGT involves an additional matrix transformation overhead of $O(N \cdot d^2)$ besides the conventional neighbor aggregation computation, which further reduces the inference efficiency when the node scale is large. Therefore, it can be concluded that the proposed model, while integrating temporal information and heterogeneous structural information, avoids excessive complexity growth, making it more suitable for the requirements of efficient inference and rapid updates in practical applications.

The results are shown in Table 5. Where: h denotes the number of heads in multi-head attention. N represents the number of nodes. E indicates the number of edges in the graph (i.e., the total number of neighbors across all nodes). d stands for the dimension of node embeddings. |R| is the number of edge types. |T| is the number of node types.

Table 5: Complexity comparison

Model	Training time complexity	Reasoning time complexity
The proposed model	$O(h \cdot N \cdot E \cdot d)$	$O(h \cdot N \cdot E \cdot d)$
HGT	$O(h \cdot N \cdot E \cdot d \cdot \lvert R \rvert \cdot \lvert T \rvert)$	$O(h \cdot E \cdot d + N \cdot d^2)$
TGAT	$O(h \cdot N \cdot E \cdot d)$	$O(h \cdot N \cdot E \cdot d)$

Table 5 shows that the proposed model has a complexity similar to that of TGAT. However, when dealing with multi-type nodes and edges, it is more efficient than HGT. At the same time, on the premise of ensuring prediction accuracy, its memory usage remains at a reasonable level, making it suitable for online deployment and analysis in real social e-commerce scenarios.

4.7 Discussion

In the previous experiments, the proposed user influence propagation model based on dynamic heterogeneous graph representation learning exhibited excellent performance in the overall regression task. For example, in the general prediction task, compared with TGAT's results (MSE = 0.1597, MAE = 0.2785, R^2 = 0.603), the proposed model achieved an MSE of 0.1214, an MAE of 0.2398, and an R^2 of 0.701. This represents a performance improvement of approximately 22.3% to 17.4%. The proposed model showed the smallest error growth across all time horizons. Especially in the long-term prediction scenario with Δ = 14 days, it reduced errors by 33%, 22.3%, and 13.3% compared with GCN, TGAT, and HGT respectively. This demonstrates its excellent temporal generalization ability and trend extrapolation ability. This improvement is mainly attributed to the model's simultaneous consideration of temporal dynamic evolution and heterogeneous structural information. This enables node embeddings to comprehensively reflect the current behavioral state and historical cumulative effects, thereby predicting future propagation potential more accurately. Compared with schemes in the literature that only use single temporal encoding or heterogeneous modeling (e.g., Wang et al. [26], Zhou et al. [27], and Chen et al. [28]), this study integrates neighbor node information through a multi-head attention mechanism and introduces learnable temporal encoding. This significantly enhances the model's ability to capture key nodes in propagation paths and improves its interpretability. Despite the above achievements, this study still has certain limitations. First, the model has limited ability to model multi-hop propagation paths and multi-level influence cascades, and there is room for improvement in the prediction accuracy for low-activity users. Second, due to the complex structure of the model, the computational cost of training and inference is relatively high, which may pose challenges for the deployment of large-scale online systems. Finally, user behaviors have high semantic dependence. In the future, joint modeling of text information and graph structure data can be further explored to enhance the model's interpretability and generalization ability.

5 Conclusion

5.1 Research contribution

This study addresses the dynamic propagation mechanism of user influence in social e-commerce environments by proposing a user influence propagation modeling method based on dynamic heterogeneous network representation learning. The method constructs a dynamic heterogeneous graph with a time-aware mechanism, comprehensively modeling the semantic relationships among multiple entity types such as users, products, and behavioral events. By jointly capturing structural heterogeneity and behavioral evolution, the study achieves precise characterization of users' future propagation potential. In terms of model design, this study integrates multi-head attention mechanisms with learnable temporal embedding strategies, enabling node representations to maintain structural diversity while enhancing temporal trend awareness.

5.2 Limitations and future research directions

Although preliminary achievements have been made in method development and experimental verification, there are still several limitations. The current model mainly focuses on influence prediction between static node pairs, and its ability to model multi-hop paths and multi-level influence cascades is limited. Future work can expand this by introducing a path attention mechanism or causal GNN. In addition, data sparsity may lead to insufficient information for some user or item nodes, thereby affecting prediction accuracy. The high complexity of the model may cause overfitting risks, which need to be mitigated through regularization or

larger-scale data. The high computational cost during training and inference also restricts the model's real-time application on ultra-large-scale platforms. To address these limitations, future research will focus on optimizing the model's generalization ability, reducing computational costs, and exploring richer data fusion strategies to improve the model's practical usability and adaptability.

References

[1] Lin X, Wang X. Towards a model of social commerce: improving the effectiveness of e-commerce through leveraging social media tools based on consumers' dual roles[J]. European journal of information systems, 2023, 32(5): 782-799. https://doi.org/10.1080/0960085X.2022.2057363

[2] He P, Shang Q, Pedrycz W, et al. short video creation and traffic investment decision in social e-commerce platforms[J]. Omega, 2024, 128: 103129. https://doi.org/10.1016/j.omega.2024.103129

[3] Kala Kamdjoug J R. The influence of social network communication on the buying behavior of Cameroonian consumers on social e-commerce platforms[J]. Journal of Enterprise Information Management, 2023, 36(5): 1319-1348. https://doi.org/10.1108/JEIM-09-2022-0329

[4] Sanbella L, Van Versie I, Audiah S. Online marketing strategy optimization to increase sales and e-commerce development: An integrated approach in the digital age[J]. Startupreneur Business Digital (SABDA Journal), 2024, 3(1): 54-66. https://doi.org/10.33050/sabda.v3i1.492

[5] Mrisha S H, Xixiang S. The power of influence: How social media influencers are shaping consumer decision making in the digital age[J]. Journal of Consumer Behaviour, 2024, 23(4): 1844-1853. https://doi.org/10.1002/cb.2308

[6] Yang B, Zhang R, Cheng X, et al. Exploring information dissemination effect on social media: an empirical investigation[J]. Personal and Ubiquitous Computing, 2023, 27(4): 1469-1482. https://doi.org/10.1007/s00779-023-01710-7

[7] Zhou S, Hudin N S. Advancing e-commerce user purchase prediction: Integration of time-series attention with event-based timestamp encoding and Graph Neural Network-Enhanced user profiling [J]. Plos one, 2024, 19(4): e0299087. https://doi.org/10.1371/journal.pone.0299087

[8] Yu Z, Das S, Xiong C. Mates: Model-aware data selection for efficient pretraining with data influence models[J]. Advances in Neural Information Processing Systems, 2024, 37: 108735-108759. https://doi.org/10.52202/079017-3453

[9] Hasselwander M, Weiss D. Consumer preferences for super app services: E-commerce, social media, and banking dominate[J]. European Research on Management and Business Economics, 2025, 31(2): 100284. https://doi.org/10.1016/j.iedeen.2025.100284

[10] Wu M, Long R. How does green communication promote the green consumption intention of social media users?[J]. Environmental Impact Assessment Review, 2024, 106: 107481. https://doi.org/10.1016/j.eiar.2024.107481

[11] Yang G, Chaiyasoonthorn W, Chaveesuk S. Exploring the influence of live streaming on consumer purchase intention: A structural equation modeling approach in the Chinese E-commerce sector[J]. Acta Psychologica, 2024, 249: 104415. https://doi.org/10.1016/j.actpsy.2024.104415

[12] Yang L, Yuan X, Yang X. Study of the influencing mechanism of user interaction behavior of short video e-commerce live-streaming from the perspective of SOR theory and interactive ritual chains[J]. Current Psychology, 2024, 43(35): 28403-28415. https://doi.org/10.1007/s12144-024-06494-4

[13] Mir I A, Salo J. Analyzing the influence of social media influencer's attributes and content esthetics on endorsed brand attitude and brand-link click behavior: The mediating role of brand content engagement [J]. Journal of Promotion Management, 2024, 30(1): 1-28. https://doi.org/10.1080/10496491.2023.2251461

[14] Zhou F, Xu X, Trajcevski G, et al. A survey of information cascade analysis: Models, predictions, and recent advances[J]. ACM Computing Surveys (CSUR), 2021, 54(2): 1-36. https://doi.org/10.1145/3433000

[15] Shao C, Shao F, Liu X, et al. A multi-information dissemination model based on cellular automata[J]. Mathematics, 2024, 12(6): 914. https://doi.org/10.3390/math12060914

[16] Sunarya P A, Rahardja U, Chen S C, et al. Decipheri ng digital social dynamics: A comparative study of logistic regression and random forest in predicting e-commerce customer behavior[J]. Journal of Applied Data Sciences, 2024, 5(1): 100-113. https://doi.org/10.47738/jads.v5i1.155

[17] Chen Y, Chen F, Wu Z, et al. Heterogeneous Graph Embedding with Dual Edge Differentiation[J]. Neural Networks, 2025, 183: 106965. https://doi.org/10.1016/j.neunet.2024.106965

[18] Kumar V, Krishna P R. An effective representation learning model for link prediction in heterogeneous information networks[J]. Computing, 2024, 106(7): 2185-2210. https://doi.org/10.1007/s00607-023-01238-x

[19] Mei G, Pan L, Liu S. Heterogeneous graph embedding by aggregating meta-path and meta-structure through attention mechanism[J]. Neurocomputing, 2022, 468: 276-285. https://doi.org/10.1016/j.neucom.2021.10.001

[20] Wang Z, Liang F, Li C, et al. Does China's low-carbon city pilot policy promote green development? Evidence from the digital industry[J]. Journal of Innovation & Knowledge, 2023, 8(2): 100339. https://doi.org/10.1016/j.jik.2023.100339

[21] Jia X, Wu P, Chen L, et al. Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory pre

diction via scene encoding[J]. IEEE transactions on pattern analysis and machine intelligence, 2023, 45 (11): 13860-13875. https://doi.org/10.1109/TPAMI.2023.3298301

[22] Gao L, Wang H, Zhang Z, et al. HetInf: Social influence prediction with heterogeneous graph neural network[J]. Frontiers in Physics, 2022, 9: 787185. https://doi.org/10.3389/fphy.2021.787185

[23] Zhong Y, Huang C. A dynamic graph representation learning based on temporal graph transformer[J]. Alexandria Engineering Journal, 2023, 63: 359-369. https://doi.org/10.1016/j.aej.2022.08.010

[24] Qiu Y. MAGT: Multi-scale Attention Graph Transformer with Local Context Enhancement for CT Image Analysis[J]. Informatica, 2025, 49(31). https://doi.org/10.31449/inf.v49i31.8111

[25] Zhang Y. Graph Neural Network-Based User Preference Model for Social Network Access Control[J]. Informatica, 2025, 49(16). https://doi.org/10.31449/inf.v49i16.7705

[26] Wang C, Wu X, Chen T. Neighborhood effect, virtual social networks, and farmers' social e-commerce participation behavior[J]. Agribusiness, 2025, 41(1): 238-259. https://doi.org/10.1002/agr.21885

[27] Zhou Y, Wang S, Hu Y. Manufacturers' social e-commerce channel selection strategy with social popularity concern[J]. Electronic Commerce Research, 2024, 24(2): 1119-1151. https://doi.org/10.1007/s10660-022-09601-4

[28] Chen X, Zhang X, Chen J. TAM-Based study of Farmers live streaming E-Commerce adoption intentions[J]. Agriculture, 2024, 14(4): 518. https://doi.org/10.3390/agriculture14040518