GAN-Based Semi-Supervised Training of LSTM Nets for Intention Recognition in Cooperative Tasks

Matija Mavsar, Jun Morimoto, *Member, IEEE*, and Aleš Ude, *Member, IEEE*

***Abstract*—The accumulation of a sufficient amount of data for training deep neural networks is a major hindrance in the application of deep learning in robotics. Acquiring real-world data requires considerable time and effort, yet it might still not capture the full range of potential environmental variations. The generation of new synthetic data based on existing training data has been enabled with the development of generative adversarial networks (GANs). In this paper, we introduce a training methodology based on GANs that utilizes a recurrent, LSTM-based architecture for intention recognition in robotics. The resulting networks predict the intention of the observed human or robot based on input RGB videos. They are trained in a semi-supervised manner, with the output classification networks predicting one of possible labels for the observed motion, while the recurrent generator networks produce fake RGB videos that are leveraged in the training process. We show that utilization of the generated data during the network training process increases the accuracy and generality of motion classification compared to using only real training data. The proposed method can be applied to a variety of dynamic tasks and different LSTM-based classification networks to supplement real data.**

***Index Terms*—Deep learning methods, real-time action recognition from video, human-robot collaboration.**

I. Introduction

COOPERATION between humans and robots enables the realization of complex tasks and at the same time relieves human workers of stressful and demanding labor. To ensure safe and efficient cooperation, autonomous systems for supervision and control of collaborative workspaces are crucial. Detecting and predicting human and robot motion during task performance can provide vital information for optimizing collaborative behaviors. Furthermore, it is important to develop solutions that can generalize across various applications, allowing for rapid adaptation to specific tasks.

Recurrent neural networks (RNNs), specifically long short-term memory (LSTM) networks [1], have proven useful in predicting future states in dynamic processes, as they can process sequential inputs by utilizing memory cells. Several methods for predicting human and robot motion based on position measurements or captured RGB(D) images have been proposed. They enable quick robot motion adaptation and thus a more efficient execution of collaborative tasks [2], [3]. However, ensuring the robustness of neural networks in robotics is often challenging due to the limited availability of high-quality training data.

In the field of machine vision, state-of-the-art networks for object classification are typically trained on millions of diverse samples. In robotics, data collection often needs to occur in specific environments where robots perform their tasks, which can be time-consuming and may require human intervention. Additionally, these environments can change rapidly, necessitating the gathering of additional training data. While simulation technologies and domain randomization can increase the amount of data and improve performance to some extent [4], significant differences between real and simulated data may persist. Given these challenges, a method that can operate with smaller amounts of data and still achieves successful motion prediction is needed.

In this paper, we propose a method to maximize the utilization of existing training data, consisting of input RGB videos and the corresponding labels. In our previous work, we developed an approach for generating object handover behaviors using recurrent neural networks [5], [6], where videos of the giver's motion are used as input to an LSTM network, which computes the necessary receiver's motion for a successful handover. The proposed network can predict either the handover location [6] or complete receiver trajectories [5]. To enhance this approach, we introduce a semi-supervised recurrent neural network training technique that employs a *Generative Adversarial Network* (GAN) with LSTM layers. This architecture comprises a generator and a discriminator, where the generator attempts to create fake RGB videos with a distribution similar to the real data, in an attempt to fool the discriminator. The discriminator can be any LSTM classification network that takes sequences of RGB images as input and categorizes the observed motion if the input is real, or labels it as fake if its input is a generated video. The resulting predictions can be used to control the motion of a robot in a collaborative environment.

Manuscript received 16 June 2023; accepted 25 October 2023. Date of publication 15 November 2023; date of current version 23 November 2023. This letter was recommended for publication by Associate Editor V. V. Unhelkar and Editor G. Venture upon evaluation of the reviewers' comments. This work was supported by program group Automation, robotics, and biocybernetics under Grant P2-0076, in part by Young Researcher Grant under Grant PR-09781, in part by the Slovenian Research and Innovation Agency, in part by EU's Horizon Europe grant euRobin under Grant 101070596, in part by project JPNP20006, commissioned by NEDO, in part by JSPS KAKENHI under Grants JP22H03669 and JP22H04998, in part by JST Mirai Program under Grant JPMJMI21B1, in part by JST Moonshot R&D Program under Grant JPMJMS223B-3, and in part by Tateishi Science and Technology Foundation. *(Corresponding author: Matija Mavsar.)*

Matija Mavsar and Aleš Ude are with the Department of Automatics, Biocybernetics, and Robotics, Jožef Stefan Institute, 1000 Ljubljana, Slovenia, also with the Department of Brain-Robot Interface, ATR Computational Neuroscience Laboratories, Kyoto 619-0288, Japan, and also with the Faculty of Electrical Engineering, University of Ljubljana, 1000 Ljubljana, Slovenia (e-mail: matija.mavsar@ijs.si; ales.ude@ijs.si).

Jun Morimoto is with the Department of Brain-Robot Interface, ATR Computational Neuroscience Laboratories, Kyoto 619-0288, Japan, and also with the Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan (e-mail: xmorimo@atr.jp).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3333231, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3333231

A. Contributions

Our main contributions can be summarized as follows:

- A semi-supervised methodology for the augmentation of training data that constructs synthetic videos and leverages them in the training process of classification networks, enabled by the use of a recurrent LSTM generator network. The proposed generator network can be combined with LSTM-based neural networks for intention prediction and motion classification.
- Extensive experiments demonstrate that the proposed GAN-based training approach enhances the performance of different LSTM-based motion classification networks compared to the same networks trained without the generative component.

The key benefit of our approach is that the networks trained with it generalize significantly better to different variations in the data than networks trained using conventional methods. Therefore, our approach is apt for real-world applications where data exhibits substantial variability, as demonstrated by realizing a practical human-robot collaboration task.

II. Related Work

Human-robot collaboration (HRC) has garnered extensive attention in recent years due to the demands of service robot applications in both home and industrial settings [7], where robots must cooperate with humans to perform different tasks. The key research goals include enhancing task performance, facilitating robot learning through physical interaction, and ensuring task fluency [8]. To improve cooperation and increase control over the collaborative environment, interfaces for better perception and motion prediction are required. For this reason, intention recognition is an important aspect of HRC, enabling the robot to recognize and predict human actions. Towards this end, the estimation of human motion from video sources has been investigated for many years [9]. Callens et al. [10] present an approach that learns motion models to detect motion onset and estimate intent. Some other approaches employ convolutional neural networks [11]. While wearables and motion capture techniques are increasingly common for activity recognition [12], [13], they do require additional hardware. Recurrent neural networks, especially LSTM nets, have found widespread use in HRC for predicting future outcomes based on sequences of past inputs. Methods for RNN-based activity recognition from input videos have been developed [14], [15], and some approaches use human skeleton motions as input to predict future poses [2], [16].

Generative adversarial networks (GANs) were initially designed for unsupervised learning, e.g. generation of photorealistic images as well as other types of data. In recent years their use has extended to other areas, such as reinforcement learning [17] and semi-supervised learning [18]. GANs in various forms have also been used in the context of robotics and human-robot collaboration, e.g. to predict optimal grasping strategy for a receiver during a human-to-human object handover using unlabeled data [19]. The generation of additional data is a common use of GANs; some existing methods employ GANs to create synthetic training data for detection and classification tasks [20], while others generate unlabeled data and use it together with real data in a semi-supervised learning pipeline [18], [21]. Exploiting GANs for sequence generation has been addressed in [22], where authors utilize LSTM-based generator and discriminator to obtain synthetic energy consumption training data, while a number of regularization techniques for better training on non-image data were presented in [23].

In our work, we aim to enhance an RGB video dataset by applying GANs for the generation of additional synthetic data in the form of image sequences. Several approaches were developed in the past for unsupervised video generation with GANs. In [24], videos of human faces are generated based on desired condition vectors. In [25] and [26], recurrent layers are added for unconditional video generation. These methods train two discriminators that process individual frames and entire videos, aiding the generator in creating both realistic and temporally consistent images. Notably, the discriminator in these approaches outputs either fake or real labels, whereas our task requires classifying input observations into a limited set of classes. For classification tasks, Dai et al. [27] have shown that employing GANs in semi-supervised training leads to higher accuracy. Interestingly, the generator tends to create unrealistic images, which helps refine the decision boundary between classes. Madani et al. [28] implement this approach for X-ray image classification by modifying the discriminator output layer to classify real data into corresponding real classes and fake data into a separate class.

From our analysis of the state-of-the-art, we conclude that RNNs and GANs have indeed been utilized for motion recognition. However, they have not been combined for semi-supervised training of motion classification networks. Drawing inspiration from these methods, we propose a GAN-based architecture for semi-supervised training of LSTM-based classification networks to predict human or robot intentions in the context of human-robot collaboration.

III. Semi-Supervised Intention Recognition

A. Generative Adversarial Networks

The concept of a generative adversarial network was first introduced by Goodfellow et al. [29]. A GAN consists of two networks that compete against each other: a generative network G that attempts to generate data similar to the data from a real dataset, and a discriminative network D that computes the probability of a specific sample coming from the real data. In probabilistic terms, a generative network should learn how to map random noise vectors $\mathbf{z} \in \mathbb{R}^M$ from a fixed *a priori* distribution $p_z(\mathbf{z})$ to the data vectors $\mathbf{x}$, $\mathbf{x} = G(\mathbf{z}, \boldsymbol{\theta}_g)$, where $\boldsymbol{\theta}_g$ are the parameters of the generative network. On the other hand, a discriminative network with parameters $\boldsymbol{\theta}_D$ computes the probability $D(\mathbf{z}, \boldsymbol{\theta}_D)$ that a sample $\mathbf{x}$ comes from the real data distribution $p_{\text{data}}(\mathbf{x})$.

To train a GAN, we simultaneously optimize the parameters of both networks: the generator should try to generate data that the discriminator network recognizes as real data, while the discriminator should correctly classify real and generated data. The optimum is obtained when the discriminator network cannot distinguish between the real and generated data [29], i.e. $p_G = p_{\text{data}}$ and $D(G(\mathbf{z}, \boldsymbol{\theta}_G), \boldsymbol{\theta}_D) = 0.5$. The training of the GAN can therefore be described as the following minimax game with value function $V(G, D)$:

$$\min_{\boldsymbol{\theta}_G} \max_{\boldsymbol{\theta}_D} \; V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\log(1 - D(G(\mathbf{z})))]. \quad (1)$$

In our architecture, both the generator and the discriminator network contain recurrent layers, while the discriminator output layer is adapted to predict $K+1$ classes. One of the classes corresponds to the generated data, while the other K classes correspond to a set of possible intentions of the observed human or robot.

B. Recurrent Neural Networks

Recurrent neural networks can process sequential input data and are therefore especially suitable for analysis of dynamic processes, including robot and human motion. They consist of memory cells that can store information dependant on inputs from previous time steps. They are composed of a cell and several gates, regulating the flow of information in and out of the cell. In each time step, the current cell state $\mathbf{c}(t)$ and hidden state $\mathbf{h}(t)$ are fed back to the LSTM unit together with the next input. In this way, each new network output depends on the results from previous time steps.

C. Recurrent Generative Adversarial Networks

A typical classification network categorizes input data into one of K classes by generating a probability distribution across these classes. In supervised learning, the model is trained to minimize cross-entropy between the actual and predicted labels. In the context of GANs, this approach can be extended to semi-supervised learning by designating the generated synthetic data as an extra class [27], [30]. We introduce a methodology named *RSS-GAN* that employs Recurrent GANs to train LSTM-based classification networks in a Semi-Supervised manner. The proposed method features an LSTM generator network, enabling generation of image sequences, and an LSTM discriminator network, designed for intention classification from input videos.

The structure of the architecture is shown in Fig. 2. The input to the recurrent generator G is a random noise vector $\mathbf{z} \in \mathbb{R}^M$ from *a priori* noise distribution $p_z(\mathbf{z})$, which is first passed through an LSTM layer. To generate a sequential output, the input vector $\mathbf{z}$ is repeatedly passed through the LSTM layer along with the cell state $\mathbf{c}(t)$ and hidden state $\mathbf{h}(t)$ from the previous time step, with the initial states set to zero. If the total length of the sequence is N, the resulting sequence of hidden states after N time steps is then $\{\mathbf{h}_i\}_{i=1}^N$, which is passed through deconvolutional, normalization and nonlinear layers. This results in a sequence of RGB images $\mathbf{I}(t) \in \mathbb{R}^{W\times H\times 3}$ of width W and height H, i.e. $\{\mathbf{I}_i\}_{i=1}^N$.

The recurrent discriminator D takes a sequence of RGB images $\{\mathbf{I}_i\}_{i=1}^N$ of observed human or robot motion as input, which can come either from a real distribution p_{data}, i.e. from the training dataset, or from the generator's distribution p_G, representing generated "fake" data. The images are passed through convolutional, LSTM and fully connected layers. The discriminator output vector $\mathbf{l} \in \mathbb{R}^{K+1}$, $\mathbf{l} = \{l_1, l_2, \ldots, l_{K+1}\}$ is additionally processed by a softmax layer to obtain a probability distribution p_D over $K+1$ possible motion classes:

$$p_D(y=i|\{\mathbf{I}(t)\}_t) = \frac{e^{l_i}}{\sum_{k=1}^{K+1} e^{l_k}},\ i = 1, \ldots, K+1, \quad (2)$$

where y represents a motion class, i.e. $y \in \{1, 2, \ldots, K+1\}$. As mentioned in Section III-A, class $K+1$ denotes videos presumed to be generated or fake, while other classes denote possible intentions of the observed agent.

While the discriminator network in Fig. 2 is custom-designed, we could replace it with any other LSTM recurrent neural network designed for motion classification, enhanced with a class that classifies fake videos. Besides the CNN-LSTM discriminator network from Fig. 2, we evaluated our approach also on an unidirectional LSTM network for action recognition [15] that consists of pretrained ResNet-18 convolutional layers, fully connected layers, and LSTM layers, with the output layer extended by the fake video class.

The following set of pairs containing image sequences and the corresponding labels is needed for training the networks:

$$\{\{\mathbf{I}_{i,j}\}_{i=1}^{N_j},\ y_j\}_{j=1}^{NumEx}. \quad (3)$$

$\mathbf{I}_{i,j}$ is the i-th image in the j-th training image sequence, N_j is the number of images in this training pair, y_j is the motion class label for the j-th sequence and $NumEx$ is the number of training pairs in the dataset.

D. RSS-GAN Training Method

In each training step, the discriminator and generator losses are calculated and used to update the weights of the networks. Note that we calculate the losses in each time step, i.e. every time a new image in the image sequence is processed by the networks. This is because in a real setting, we want to predict the motion class of the observed agent as soon as possible, after only a few images of motion are available. The prediction typically becomes more accurate as a larger part of the executed motion is processed.

The discriminator is trained to improve the classification accuracy of motion videos by minimizing the cross-entropy between the predicted class distribution and the actual class label. We use both real videos and the image sequences generated by network G to train the discriminator. The loss function for the discriminator is given by [21]:

$$\begin{aligned} L_{D,n} = &-\mathbb{E}_{\{\mathbf{I}_i\}_{i=1}^n, y\sim p_{\text{data}}}[\log p_D(y|\{\mathbf{I}_i\}_{i=1}^n)] \\ &-\mathbb{E}_{\{\mathbf{I}_i\}_{i=1}^n \sim p_G}[\log p_D(y=K+1|\{\mathbf{I}_i\}_{i=1}^n)] \\ &= L_{D,n,\text{sup}} + L_{D,n,\text{unsup}}, \end{aligned} \quad (4)$$

where $L_{D,n,\text{sup}}$ and $L_{D,n,\text{unsup}}$ are respectively the supervised

$$L_{D,n,\text{sup}} = -\mathbb{E}_{\{\mathbf{I}_i\}_{i=1}^n, y\sim p_{\text{data}}}[\log p_D(y|\{\mathbf{I}_i\}_{i=1}^n, y<K+1)] \quad (5)$$

and unsupervised loss

$$\begin{aligned} L_{D,n,\text{unsup}} = &-\mathbb{E}_{\{\mathbf{I}_i\}_{i=1}^n \sim p_{\text{data}}}[\log[1-p_D(y|\{\mathbf{I}_i\}_{i=1}^n, y=K+1)]] \\ &-\mathbb{E}_{\{\mathbf{I}_i\}_{i=1}^n \sim p_G}[\log p_D(y=K+1|\{\mathbf{I}_i\}_{i=1}^n)]. \end{aligned} \quad (6)$$

In the supervised loss function $L_{D,n,\text{sup}}$ we evaluate the classification of real videos, while in the unsupervised loss function $L_{D,n,\text{unsup}}$ we evaluate whether the generated videos were correctly labeled as fake and the real videos as not fake.

To train the generator, instead of directly maximizing the output of the discriminator, we define a loss function that matches the statistics of real and generated data [21]. Specifically, we train the generator to match the expected values of real image features and generated image features:

$$\begin{aligned} L_{G,n} = &\|\mathbb{E}_{\{\mathbf{I}_i\}_{i=1}^n \sim p_{\text{data}}} \boldsymbol{f}(\{\mathbf{I}_i\}_{i=1}^n) \\ &-\mathbb{E}_{\{\mathbf{I}_i\}_{i=1}^n \sim p_G}(\boldsymbol{f}(\{\mathbf{I}_i\}_{i=1}^n)\|^2, \end{aligned} \quad (7)$$

Algorithm 1: RSS-GAN Training Procedure.

Initialization: Create a recurrent generator G with parameters $\boldsymbol{\theta}_G$ and a recurrent discriminator D with parameters $\boldsymbol{\theta}_D$. Set the number of epochs to E. Set number of generator update steps to n_G.

for epoch $= 1$ to E **do**
  Sample noise $\mathbf{z}$ from p_z.
  Calculate a batch of image sequences from the generator's distribution, $\{\mathbf{I}_i\}_{i=1}^{n} \sim p_G$.
  Sample a batch of image sequences and labels from the real training data, $\{\mathbf{I}_i\}_{i=1}^{n}, y \sim p_{\text{data}}$.
  Update discriminator weights $\boldsymbol{\theta}_D$ by backpropagating loss L_D given by (8).
  for step $= 1$ to n_G **do**
    Perform operations 2 and 3 to obtain $\{\mathbf{I}_i\}_{i=1}^{n} \sim p_G$.
    Update generator weights $\boldsymbol{\theta}_G$ by backpropagating loss L_G, given by (8).
  end for
end for
return $\boldsymbol{\theta}_G$, $\boldsymbol{\theta}_D$

where $\boldsymbol{f}(\{\mathbf{I}_i\}_{i=1}^{n})$ denotes the output of an intermediate discriminator layer (shown in Fig. 2). Note that the generation of fake samples, calculation of loss and its backpropagation can be repeated any number of times to optimize the performance of the generator network. We denote the number of generator update steps (before the discriminator is updated again) as n_G and describe its use in more detail in Algorithm 1.

Since the discriminator and generator loss are calculated for each time step n, the total loss for an image sequence of length N is given by

$$L_D = \frac{1}{N}\sum_{n=1}^{N} \gamma_n L_{D,n}, \quad L_G = \frac{1}{N}\sum_{n=1}^{N} \gamma_n L_{G,n}, \tag{8}$$

where $L_{D,n}$ and $L_{G,n}$ are calculated using (5)–(7) for a time step n and are weighted with γ_n,

$$\gamma_n = \frac{1}{1 + e^{-\alpha \frac{n-1}{N-1} + 0.5}}. \tag{9}$$

Here, parameter α adjusts weights at each time step, emphasizing later images where more information is available.

The RSS-GAN models and their training were implemented using PyTorch library [31]. The learning rate was set to 10^{-4}. The training was stopped after 60 consecutive epochs of no loss reduction on the validation dataset. The model weights with the highest classification accuracy on the validation dataset were used for evaluation on the test dataset. The training procedure is summarized in Algorithm 1.

IV. Experiments

We conducted a series of experiments to evaluate the performance of the RSS-GAN training method. We assessed motion prediction accuracy using two distinct datasets. In the first scenario, a robot executed motions towards random goal poses. The goal of the task was to predict the robot's target area based on its observed motions. In the second scenario, a human worker placed a workpiece into one of the four designated slots.

Fig. 1. Example of an experimental setup for a human-robot collaboration task. The proposed approach can be used to supplement real training data for a more dynamic and efficient performance during the assembly process.

Here, the objective was to predict the targeted slot based on the observed human arm movement.

Examples of both robot and human input videos, along with the goal areas for each motion class, can be seen in Fig. 3. We compared the accuracy of classification networks trained using our proposed approach – employing both real and generated data – to the accuracy of the same networks trained solely with real data, i.e., without the use of the GAN-based generator. We evaluated the classification performance on robot data using our custom-designed CNN-LSTM discriminator network (see Fig. 2). Meanwhile, for the human videos, we tested both the custom CNN-LSTM and the LSTM-based network with ResNet-18 convolutional layers (ResNet-LSTM) proposed in [15].

A. Motion Data Acquisition

Robot motion data was acquired by executing 7198 motions, defined as minimum jerk trajectories [32] with randomly selected initial position, fixed initial orientation and randomly selected end pose. The initial and final positions as well as final orientations were confined to a fixed range of values. The motions were recorded using an RGB camera. To distribute motions into a discrete set of classes, the motion end area was split into $K = 4$ subareas in the x-z plane and the videos were labelled with $y \in \{1, 2, 3, 4\}$, based on the subarea where the robot motion has finished. This way we obtained 7198 video-label data pairs, which were split into training, validation and test subsets of sizes 5879, 654 and 665, respectively. We applied small random rotations to training images to imitate variations of the camera view directions. Brightness, contrast and saturation were randomly adjusted and Gaussian noise was added to the acquired frames. All RGB frames were resized to 128×128 pixels and the video streams were subsampled to contain 8 frames.

We gathered human motion data in a human-robot collaboration setting, as illustrated in Fig. 1. In this setting, both the human and the robot share a workspace to place workpieces into $K = 4$ slots. We recorded the human using an RGB camera, while videos were labeled with $y \in \{1, 2, 3, 4\}$, depending on the goal slot of each human motion. This way we obtained a total of 1200 video-label data pairs. The gathered human data was split into training, validation, and test subsets in two distinct

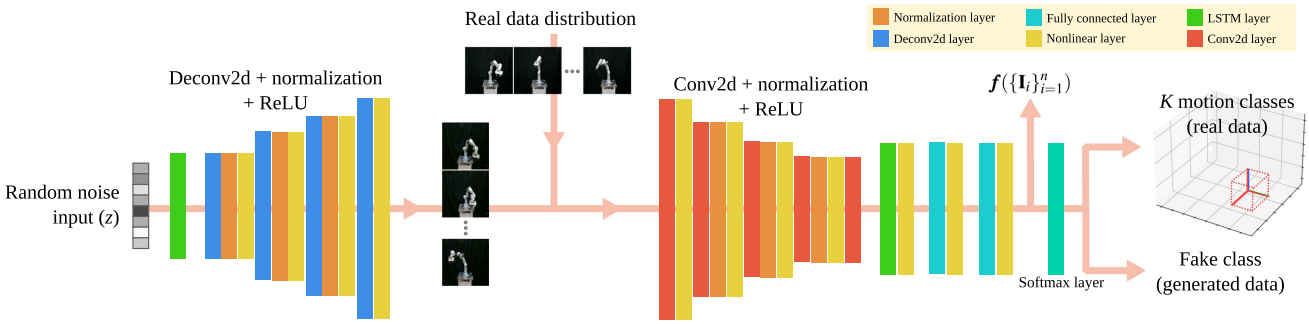

Fig. 2. Proposed system for semi-supervised learning of networks for intention recognition from image sequences. The generator network creates synthetic videos of robot or human motion, while the discriminator network attempts to correctly classify videos from the real data distribution and label videos from the generator's distribution as fake. The discriminator therefore predicts a probability distribution across K real motion classes and one fake class, while the generator attempts to generate images with features similar to real image features.

ways in order to statistically evaluate the neural network training under varied conditions:

- Firstly, we created five separate sets of test data. We achieved this by selecting five non-overlapping test subsets, each containing 80 consecutive samples, from the last 400 data pairs. The remaining data was randomly distributed between training and validation data.
- Secondly, 10 different train-validation-test splits were made by designating the last 200 samples as the test subset. The remaining data underwent 10 random train-validation divisions.

Our rationale behind this regime was to assess the trained networks using *testing data* from distinct recording sessions than those from which the *training/validation data* originated. This was crucial as we aimed to understand how well the networks generalize to new data, especially given that human motion exhibits significantly more variability than robot motion. Another layer of variability arose from changes in clothing across recording sessions. All training sets underwent same randomization as the robot sets. Image sizes were adjusted depending on the architecture employed.

B. Results With Robot Motion Data

To assess the performance of the training process using our proposed RSS-GAN methodology, we first applied it to train the custom-designed CNN-LSTM network on the robot motion dataset. Throughout our experiments, we varied several parameters, including the percentage of training data used, the dimension M of the random noise vector $\mathbf{z}$ used as input to the generator network, and the number of consecutive updates to the generator's weights, denoted as parameter n_G in Algorithm 1.

We conducted some initial tests by varying values of n_G and M to evaluate their impact. A higher number of loops n_G improved the accuracy, likely because the generator produced higher quality training images. Similarly, the value of M significantly influenced the accuracy, prompting us to test two distinctly different values. Based on these initial findings, we opted to train the proposed networks using combinations of $n_G \in \{2, 5\}$ and $M \in \{8, 100\}$. We then evaluated these architectures using the test dataset. We set the parameter α from (9) to 7, ensuring the appropriately balanced weight distribution between early and late time steps.

Classification accuracy was determined after processing each frame in the input image sequences (example predictions are illustrated in Fig. 3). Fig. 4 displays the average accuracy of the networks in relation to the number of processed frames and different hyperparameters. Training was conducted with varying sizes of training datasets. The accuracy noticeably improves as more frames are processed; with the RSS-GAN approach, when utilizing 100% of the training data, it peaks at 85.7%. In contrast, when trained solely on real data—without using the proposed GAN-based training methodology—the accuracy reaches only up to 82%.

The presented results show that the motion classification accuracy improves when the generated data is used in the training process along with the real data. The average accuracy was higher in all cases, regardless of the percentage of data and hyperparameters used for training. The differences were, however, not very large. This is probably because the variability in the input images is low, which reduces the effect of additionally generated data on generalization.

C. Results With Human Motion Data

In the experiment with human motion data, the variability of the gathered data was much higher than in robot experiments, both due to the variability of motion and variability of clothing. As explained in Section IV-A, the testing sets were formed in such a way to test the generalization power of the networks, i.e. the test data was collected in different data collection sessions than the data used for training/validation. We used two different series of datasets to statistically evaluate the performance of the trained networks. Based on robot data results, we selected the combination $n_G = 2$ and $M = 100$ to train our custom CNN-LSTM network, since it has shown consistent performance. When training the ResNet-LSTM, however, $n_G = 1$ was used to reduce the training time. The results are shown in Fig. 5.

Similarly to the robot data experiment, the performance of LSTM-based networks increases as more data frames are processed, either with or without employing the proposed RSS-GAN training method. However, overall accuracy of the networks trained with RSS-GAN methodology is significantly higher that the accuracy of networks trained without the generated data. The reason for this performance difference is that

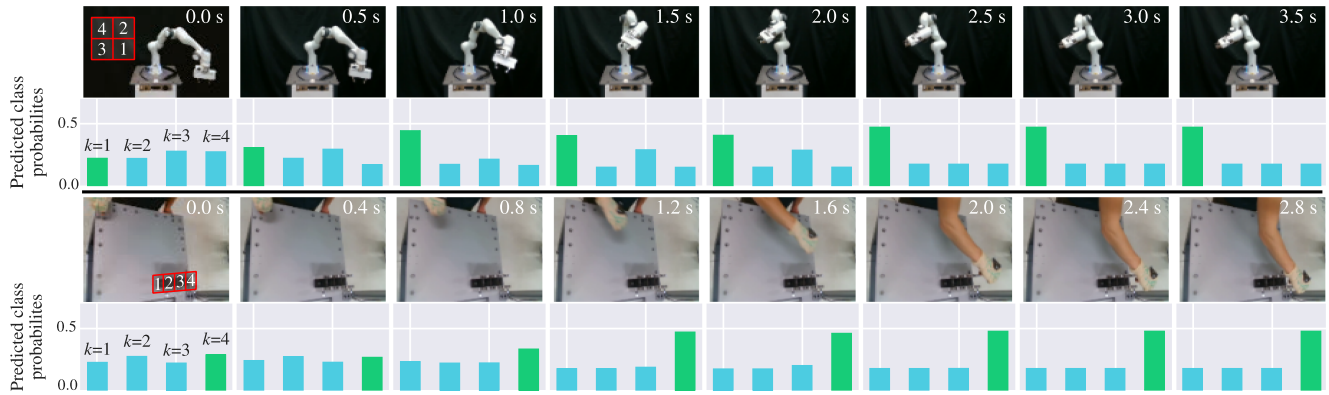

Fig. 3. Predicted probability distributions across motion classes for example robot and human input videos. Red rectangles represent the goal areas for each class of motion that the network must predict. As more camera frames are processed, the predicted probability of the correct class typically increases. Robot motion prediction model was trained using parameters $n_G = 5,\ M = 8$, while values $n_G = 2,\ M = 100$ were used in the case of human data.

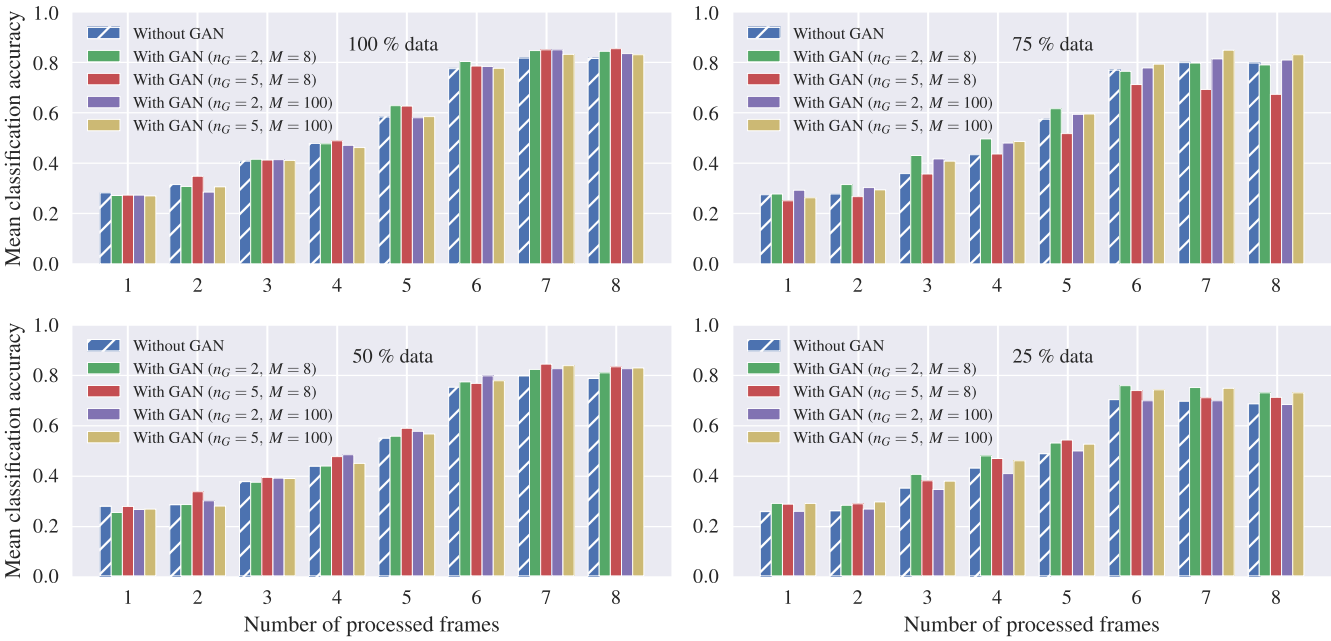

Fig. 4. Accuracy comparison on robot video dataset for different combinations of random noise vector size M and number of generator updates n_G. Networks were trained using 100%, 50%, 75%, and 25% of training data. Additionally, accuracy for models trained using real data only (i.e. without utilization of synthetic generated data) is shown. A clear increase in classification accuracy can be observed with more processed frames for all models.

the GAN-generated data prepare the networks for variations not present in the training data. Thus the networks trained with RSS-GAN methodology generalize better.

The standard deviations plotted atop the accuracy bars indicate that the results are statistically significant. Notably, once all data frames are processed, networks trained with GANs exhibit significantly greater accuracy compared to those trained without GANs. It is worth noting that the mean accuracy of networks trained using the RSS-GAN methodology surpasses the sum of the mean accuracy and standard deviation of networks trained without GANs.

D. Observations

We observe that the increase in classification accuracy is more pronounced with human data than robot data. The main reason for this is that robot data is less diverse, with minimal background changes and with motions following consistent trajectories. Consequently, the data generated by GAN may not provide as much novel information and has a smaller impact on network performance. Additionally, we observed that accuracy was lower for robot data compared to human data, possibly because robot goal areas were less clearly defined, making it easier for the network to misclassify motion, particularly with low-resolution images.

Fig. 6 displays example synthetic videos generated during training with the RSS-GAN approach. The generators for both types of data output image sequences that resemble real data. However, the images can clearly be recognized as fake. This is in line with the findings in [27], where the authors show that the generator distribution should not match the true data distribution for good semi-supervised learning.

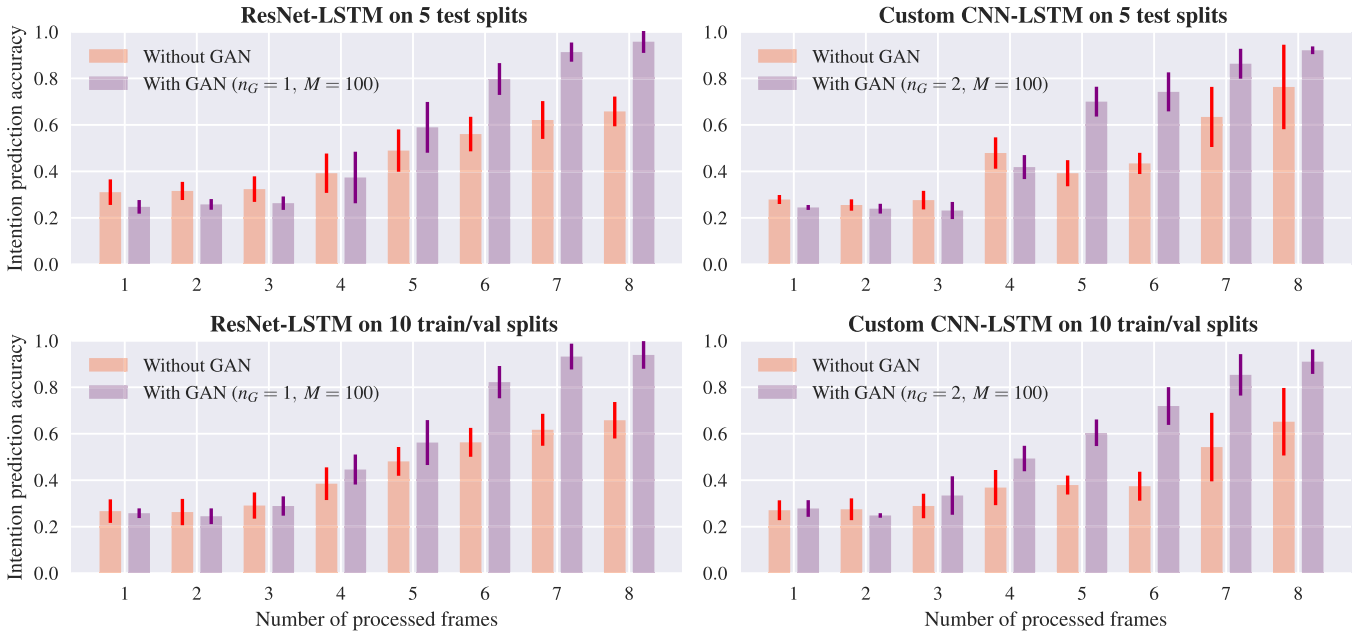

Fig. 5. Comparison of classification accuracy on human motion dataset with and without GAN utilization. We evaluated the performance of both our custom discriminator network and of a previously proposed action recognition network ResNet-LSTM. The top row shows performance of the networks trained on 5 different sets of test data and randomly selected training/validation data. The bottom networks were trained on a single test set, while randomly varying the training/validation data. The error intervals at the top of each bar show the standard deviation of the classification accuracy across the datasets. Image input sizes were 128×128 when using our custom CNN-LSTM and 224×224 when using ResNet-LSTM.

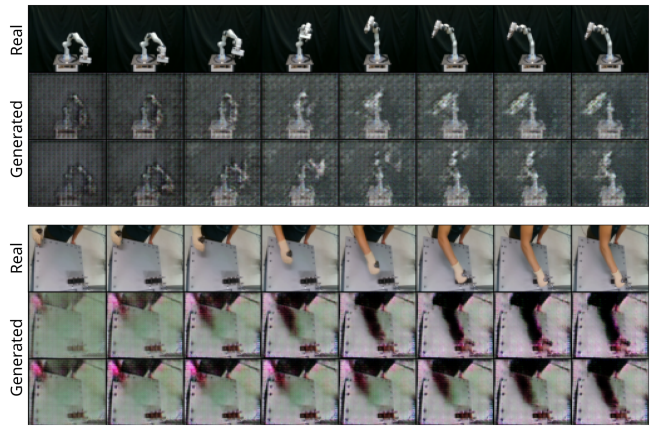

Fig. 6. Example of generated robot (top) and human (bottom) images, compared to real images. It can be easily seen that they do not come from the real data distribution. However the generated features still have the ability to positively contribute to the training process. The presented images were generated using models trained with $n_G = 5$, $M = 8$ and $n_G = 2$, $M = 100$ on robot and human data, respectively.

The ResNet-LSTM architecture trained on 5 splits of human motion data was implemented in a real human-robot collaborative assembly experiment and is presented in a video uploaded as supplementary material. This experiment shows that the network can predict the goal slot of human motion in a timely manner and can be used to adapt the robot motion to prevent interference with a human worker.

E. Benefits and Limitations

Our experiments with robot and human motion data are complementary and show that the impact of the RSS-GAN training approach on performance varies depending on the dataset. Based on the results, we find that when the training dataset already covers most of the potential variations, the expected performance increases are relatively small. This is noticeable with the results on robot data, where training videos have very little variations and the robot motions are relatively consistent. On the other hand, networks trained using the RSS-GAN approach exhibit significantly improved generalization when faced with larger differences between testing and training data, as is the case with human data. This is supported by a notable increase in classification accuracy observed in our experiments.

There are some inherent limitations to the generalization capabilities of the trained networks. Our proposed method exhibits best performance when the training and testing data are coherently connected. When the variability of input videos is extremely high, the variations become too broad, overwhelming the capacity of the RSS-GAN method to generate useful synthetic data.

While the proposed approach can address some classification performance issues, it is important to note that it cannot fully resolve all of them. As discussed in Section IV-D, classifying our robot motion dataset presents a challenge due to continuous transitions between motion classes, making precise classification at the class boundaries difficult. Unlike when dealing with large variations in motion and lighting conditions, the RSS-GAN training method is unable to produce data that effectively address this specific issue.

V. Conclusion

In this letter, we introduce a recurrent generative adversarial approach, termed RSS-GAN, to train intention prediction networks through semi-supervised learning. Leveraging recurrent layers enables us to generate and process sequences of images,

allowing for the prediction of an observed agent's intention even before the motion finishes. We demonstrated the applicability of the proposed training approach on different LSTM-based discriminator networks and evaluated it using two distinct types of training data: robot motion and human motion videos. Our findings illustrate that incorporating synthetically generated data alongside real data enhances motion classification accuracy of the LSTM-based networks, which makes their application suitable for real-time human-robot collaborative tasks.

In our current training methodology, we have experimented with various types of LSTM-based discriminators, maintaining the generator's design as depicted in Fig. 2. However, it is worth noting that the generator network can also be modified, as demonstrated in [26]. In our future work, we aim to explore different LSTM-based generator networks to further improve performance. Another potential research direction involve moving towards regression problems, where the goal is to predict the final continuous pose of human or robot movement. This task goes beyond merely classifying the motion into a predefined set of goals.

REFERENCES

[1] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[2] J. Zhang, H. Liu, Q. Chang, L. Wang, and R. X. Gao, "Recurrent neural network for motion trajectory prediction in human-robot collaborative assembly," *CIRP Ann.*, vol. 69, no. 1, pp. 9–12, 2020.

[3] W. Yang, C. Paxton, A. Mousavian, Y.-W. Chao, M. Cakmak, and D. Fox, "Reactive human-to-robot handovers of arbitrary objects," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 3118–3124.

[4] J. Tobin et al., "Domain randomization and generative models for robotic grasping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 3482–3489.

[5] M. Mavsar, B. Ridge, R. Pahić, J. Morimoto, and A. Ude, "Simulation-aided handover prediction from video using recurrent image-to-motion networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 30, 2022, doi: 10.1109/TNNLS.2022.3175720.

[6] M. Mavsar and A. Ude, "RoverNet: Vision-based adaptive human-to-robot object handovers," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2022, pp. 858–864.

[7] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. Albu-Schäffer, K. Kosuge, and O. Khatib, "Progress and prospects of the human–robot collaboration," *Auton. Robots*, vol. 42, no. 5, pp. 957–975, 2018.

[8] G. Hoffman, "Evaluating fluency in human–robot collaboration," *IEEE Trans. Hum.- Mach. Syst.*, vol. 49, no. 3, pp. 209–218, Jun. 2019.

[9] A. Ude, "Robust estimation of human body kinematics from video," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1999, pp. 1489–1494.

[10] T. Callens, T. v. d. Have, S. V. Rossom, J. D. Schutter, and E. Aertbeliën, "A framework for recognition and prediction of human motions in human-robot collaboration using probabilistic motion models," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5151–5158, Oct. 2020.

[11] Y. Li et al., "Efficient convolutional hierarchical autoencoder for human motion prediction," *Vis. Comput.*, vol. 35, pp. 1143–1156, 2019.

[12] A. Kubota, T. Iqbal, J. A. Shah, and L. D. Riek, "Activity recognition in manufacturing: The roles of motion capture and sEMG+ inertial wearables in detecting fine vs. gross motion," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 6533–6539.

[13] W. Wang, R. Li, Z. M. Diekel, Y. Chen, Z. Zhang, and Y. Jia, "Controlling object hand-over in human–robot collaboration via natural wearable sensing," *IEEE Trans. Hum.- Mach. Syst.*, vol. 49, no. 1, pp. 59–71, Feb. 2019.

[14] Z. Wang, B. Wang, H. Liu, and Z. Kong, "Recurrent convolutional networks based intention recognition for human-robot collaboration tasks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2017, pp. 1675–1680.

[15] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, "Action recognition in video sequences using deep bi-directional LSTM with CNN features," *IEEE Access*, vol. 6, pp. 1155–1166, 2018.

[16] M. S. Yasar and T. Iqbal, "A scalable approach to predict multi-agent motion for human-robot collaboration," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1686–1693, Apr. 2021.

[17] C. Wang, C. Pérez-D'Arpino, D. Xu, L. Fei-Fei, K. Liu, and S. Savarese, "Co-GAIL: Learning diverse strategies for human-robot collaboration," in *Proc. Conf. Robot Learn.*, 2022, pp. 1279–1290.

[18] Z. Zheng, L. Zheng, and Y. Yang, "Unlabeled samples generated by GAN improve the person re-identification baseline in vitro," in *Proc. IEEE Int. Conf. Comp. Vis.*, 2017, pp. 3754–3762.

[19] R. Ye, W. Xu, Z. Xue, T. Tang, Y. Wang, and C. Lu, "H2O: A benchmark for visual human-human object handover analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15762–15771.

[20] H. Lu, M. Du, K. Qian, X. He, and K. Wang, "GAN-based data augmentation strategy for sensor anomaly detection in industrial robots," *IEEE Sensors J.*, vol. 22, no. 18, pp. 17464–17474, Sep. 2022.

[21] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2016.

[22] M. N. Fekri, A. M. Ghosh, and K. Grolinger, "Generating energy data for machine learning with recurrent generative adversarial networks," *Energies*, vol. 13, 2020, Art. no. 130.

[23] M. Lee, D. Tae, J. H. Choi, H.-Y. Jung, and J. Seok, "Improved recurrent generative adversarial networks with regularization techniques and a controllable framework," *Inf. Sci.*, vol. 538, pp. 428–443, 2020.

[24] Y. Wang, P. Bilinski, F. Bremond, and A. Dantcheva, "ImaGINator: Conditional spatio-temporal GAN for video generation," in *Proc. IEEE/CVF Winter Conf. Appl. Comp. Vis.*, 2020, pp. 1160–1169.

[25] S. Gupta, A. Keshari, and S. Das, "RV-GAN: Recurrent GAN for unconditional video generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2022, pp. 2024–2033.

[26] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz, "MoCoGAN: Decomposing motion and content for video generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1526–1535.

[27] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. Salakhutdinov, "Good semi-supervised learning that requires a bad GAN," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2017.

[28] A. Madani, M. Moradi, A. Karargyris, and T. Syeda-Mahmood, "Semi-supervised learning with generative adversarial networks for chest X-ray classification with ability of data domain adaptation," in *Proc. 15th Int. Symp. Biomed. Imag.*, 2018, pp. 1038–1042.

[29] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2014.

[30] A. Odena, "Semi-supervised learning with generative adversarial networks," in *Proc. ICML Workshop Data-Efficient Mach. Learn.*, 2016.

[31] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

[32] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. New York, NY, USA: Wiley, 2006.