Generative sEMG Deep Learning for Early Prediction of Locomotion with Small Training Datasets

Zhanfeng HUANG, Zhiping LIN
School of Electrical and Electronic Engineering,
Nanyang Technological University, Singapore
{huan0343@e. , ezplin@}ntu.edu.sg

Haihong ZHANG, Chuanchu WANG,
Soon Huat NG, Christina Ka Yin TANG
*Institute for Infocomm Research, A*STAR, Singapore*
{hhzhang,ccwang,ngsh,kytang}@i2r.a-star.edu.sg

Kai Keng ANG
*Institute for Infocomm Research, A*STAR, Singapore*
School of Computer Science and Engineering,
Nanyang Technological University, Singapore
kkang@i2r.a-star.edu.sg

***Abstract*—Early detection of locomotion intention is highly relevant to the development of intelligent rehabilitation/assistive robotics. While surface electromyography(sEMG) has been a promising tool, it is often challenged by the shear variability of sEMG patterns in contrast to only a handful of sEMG training samples per discrete motion intention class for each individual user to begin with. To address this issue, we introduce a deep convolutional generative adversarial networks (DCGANs), including dynamic time warping (DTW) and fast Fourier transform mean square error (FFT MSE) for artificial signal quality assessment. On a preliminary sEMG data set of 3-class directional lower-limb movement, the proposed method yielded an average accuracy rate of 89.31% ± 6.52. While this is a feasibility study using healthy human subjects, the result warrants extended study to further establish the generative adversarial network learning for EMG intention detection in real-world rehabilitation/assistive system applications.**

***Index Terms*—rehabilitation, electromyography, generative adversarial networks**

I. Introduction

Surface electromyography (sEMG) has been among major sensor modalities in the field of motion intention detection. By capturing intrinsic neuromuscular electrical activities, it can offer robots with the ability to understand human motion intentions and facilitate safe interactions between humans and robots [1]. Most of the EMG studies in this field thus far are focused on either the estimation of kinematic parameters of the classification or prediction of human motion state transition. Various machine learning tools have been proposed. For instance, an Encoder-Decoder Temporal Convolutional Network (ED-TCN) Betthauser et al. [2] was proposed for sEMG motion prediction with a relatively lower prediction latency. Two streams of convolutional networks were employed in Hajian and Morin [3] to learn informative features from raw sEMG data using different scales and estimate the motion generated during elbow flexion and extension. In Pew and Klute [4], the authors used support vector machine (SVM), K nearest neighbor (KNN) and bagged decision tree ensemble (Ensemble) as classifiers, integrated inertial motion unit (IMU) and sEMG signals, and predicted the subject's turning intention 400ms in advance. In Côté-Allard et al. [5], a migration learning method was introduced to learn general and information-rich features from a large amount of data generated by aggregating signals from multiple users.

Applications of EMG for intention detection usually employ the subject-dependent model training approach to resolve the issue of sheer data variability due to various intrinsic (e.g. disability condition of the exoskeleton user) or extrinsic factors (such as the sensor array configuration). The collection of individual motion intention samples (especially when using real assistive devices like an exoskeleton) can be expensive.

On the other hand, data augmentation has been widely used to address various small training data problems, especially in the fields of computer vision (CV) and natural language processing (NLP) and has achieved remarkable results [6]. For the biomedical signal of electroencephalography (EEG), Lashgari et al. [7] compared different data augmentation . For sEMG, Anicet Zanini and Luna Colombini [8] proposed a generative adversarial network using multiple features in sEMG to simulate Parkinson's disease (PD) sEMG signals. Gunasar et al. [9] proposed a "find valid augmented samples" algorithm, which considers the validity of the generated sEMG signal and significantly improves the accuracy of using sEMG for gesture classification.

In the work, we propose a deep convolutional generative adversarial networks (DCGANs) to learn and classify EMG data from 3-class directional lower-limb movement. The training method uses dynamic time warping (DTW) and fast Fourier transform mean square error (FFT MSE) for artificial signal quality assessment. the proposed method is tested on a data set of 5 human subjects, and yielded an average accuracy rate of 89.31% ± 6.52. While this is a feasibility study using healthy

human subjects, the result warrants extended study to further establish the generative adversarial network learning for EMG intention detection in real-world rehabilitation/assistive system applications.

II. Materials And Methods

In this section, we show the sEMG signal processing process and a data augmentation method based on Deep Convolutional Generative Adversarial Networks (DCGANs) and introduce the signal quality evaluation method, and finally use the discriminator in the model to predict the signal motion intention. The architecture of the model is shown in Fig. 1. In the first step, the input sEMG signal is high-pass filtered to remove noise and artifacts in the signal. The second step is to divide the data into two categories according to the different classification labels in the data set. The first category is input into DCGANs to update the network parameters, and the second category does not participate in updating parameters but is directly input into the discriminator to judge whether it is similar to the first category. Meanwhile, a set of one-dimensional random signals is input to DCGANs to generate a set of generated signals. Finally, the prediction results of the discriminator are analyzed, and the quality of the generated signal is analyzed using dynamic time warping (DTW) and fast Fourier transform mean square error (FFT MSE).

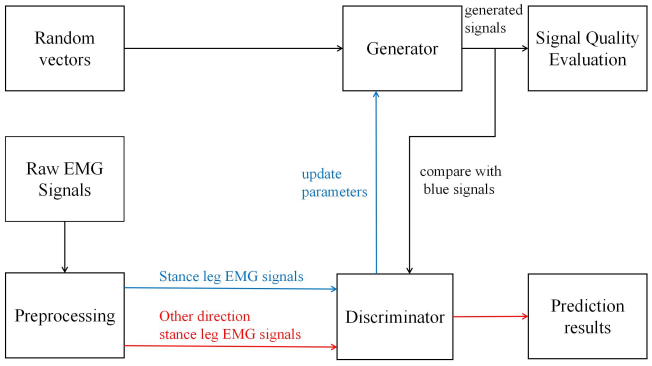


Fig. 1: Architecture of the data augmentation and motion prediction model

A. Data Collection and Pre-processing

We designed a VR game to obtain data when the subjects were in motion. Here we provide a concise description of the data collection process because it is a complex process and will be reported in a separate full paper. Five healthy male individuals were enlisted as volunteers for the study. A wearable sensor array system consisting of three parts including sEMG, kinematics, and insole pressures was equipped in each experiment subject during data collection. Among them, for the sEMG part, four muscles that included Biceps Femoris (BF), Vastus Medialis (VM), Tibialis Anterior (TA), and Gastrocnemius Medialis (GM) were selected. For each subject, the process was divided into a total of 3 phases all on the same day. Each session had subjects perform right kicks in three designated directions: left, center, and right. The process is controlled by a computer that pseudo-randomly chooses the direction of the next kick and then informs the subjects with audiovisual cues. When prompted, the subjects were asked to quickly perform a specific directional kick. On the other hand, the subjects did not know the specific direction before the cue, but only started to move when the cue appeared. The study was conducted in compliance with the guidelines outlined in the research protocol, which was approved by the ethics review board of the Agency for Science, Technology and Research, Singapore, with reference to IRB Ref. No. 2020-006.

Since we want to study the movement of the legs of the disabled, especially the other healthy leg when the movement of one leg is limited, in this experiment, the right leg is uniformly used to simulate the limited movement of the disabled leg The sEMG movement and intention generated during the body movement, the left leg is used as the standing leg, and the sEMG signal of the healthy leg of the disabled is simulated. The aim of this study is to perform motion prediction on the kicking leg (right leg), specifically, the kicking direction based on stance leg sEMG signals. At the same time, in order to test whether the difference in the movement direction of the subject's right leg can be predicted and accurately described, in the signal generation, all the data of the subject kicking to the left are used for sEMG signal generation, and the data of kicking to the right are used for prediction verification. In order to make the collected data more consistent, we define two-time nodes t_{cue} and t_{onset}. Where t_{cue} refers to the time when the target appeared on the subject's screen, and t_{onset} refers to the time when the subject began to attack the target, which was determined by the pressure sensor in the leg. Since we are interested in the sEMG signal when the movement has not started, the first 500ms before t_{onset} is intercepted as the signal to be learned during preprocessing. We define this time point as $t_{before\_onset\_500ms}$.

The amplitude of the acquired sEMG signal is very weak (0-10mV), the noise will be introduced during the acquisition process. To fit the data requirements of machine learning, the filtered signal needs to be normalized. Fig. 2 shows the preprocessing of a four-channel sEMG signal, where each row represents one of the channels in the signal.

B. The Deep Convolutional Generative Adversarial Networks Model

The architectures of the generator and discriminator in DCGANs are shown in Fig. 3 and Fig. 4 respectively. For the generator input, it is a set of random one-dimensional vectors. for the discriminator, it is a set of signals generated by the generator. The output of the discriminator is a probability in the range [0,1]. When the probability is 0, it means that the discriminator believes that the generated signal is very different from the real signal in the training set. When the probability is 1, it means that the discriminator believes that the generated signal is consistent with the real signal. For DCGANs, we hope that this probability can approach 0.5 because it means that the

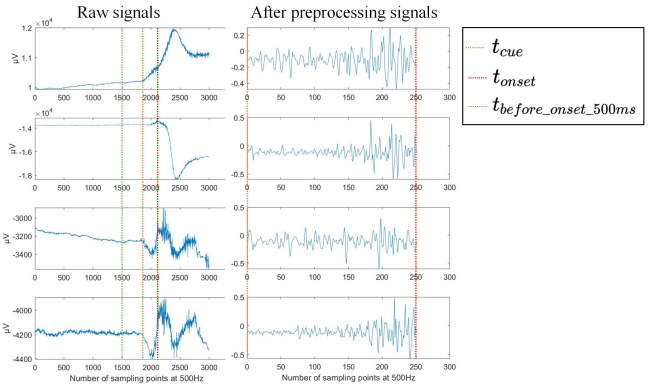


Fig. 2: sEMG signal preprocessing flow

discriminator has half the probability that the generated signal is real, which means that the generated signal is enough to "fool" the discriminator.

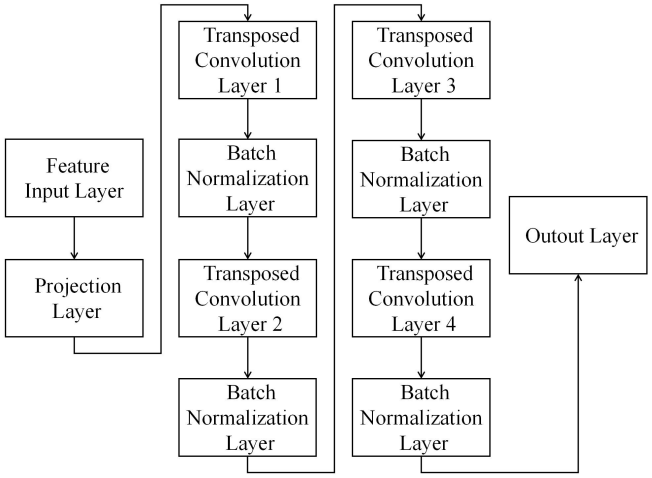


Fig. 3: Generator structure in DCGANs

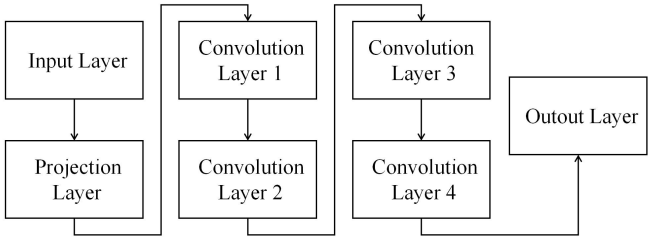


Fig. 4: Discriminator structure in DCGANs

For the input of the generator, the length of the random vector determines that the generator can generate diverse outputs. In this paper, the length of this vector is determined to be 8, which is determined by the input sample size, network structure, and sEMG signal characteristics. After generating random noise in the input layer, it is fed into the projection layer. The projection layer refers to a fully connected layer, and its function is to map the random noise input of the generator to a latent space similar to the real data distribution. After the transformation of the projection layer, the input vector is converted into an (8*1024) matrix for subsequent processing.

Transposed convolution is originally an operation in convolutional neural networks to upsample the feature map output by the convolutional layer for processing in subsequent layers. In the task of DCGANs generating signals, we need to restore the full-scale data after the projection layer to the size of the original signal before further calculation. The size calculation formula of transposed convolution is shown in (1). Among them, I is the size of the input signal, O is the size of the output signal, s is the convolution straddle size, k is the size of the convolution kernel, and p is the padding of the input signal to align the size of the output signal.

$$O = s(I-1) + k - 2p \tag{1}$$

For a one-dimensional signal, the calculation process of the transposed convolution is: 1. First calculate the size of the output signal according to (1) and initialize it. 2. Padding the input vector, that is, filling the corresponding number of zeros on both sides of the vector according to the specific value of padding to align it. 3. Perform convolution calculation, Let the input be $X = [X_1, X_2, ...X_N]$, the convolution kernel be $K = [K_1, K_2, ...K_M]$,the output be

$$Y = \begin{bmatrix} P & Q \end{bmatrix}^T \tag{2}$$

.At the same time, the convolution kernel is expanded into a sparse matrix according to the step size, assuming a straddle size of 2, a sparse matrix is

$$C^T = \begin{bmatrix} K_1 & 0 \\ . & 0 \\ . & K_1 \\ K_N & . \\ 0 & . \\ 0 & K_N \end{bmatrix} \tag{3}$$

The result of the transposed convolution is

$$\begin{bmatrix} PK_1 \\ PK_2 \\ PK_3 + QK_1 \\ PK_4 + QK_2 \\ . \\ . \\ QK_{N-1} \\ QK_N \end{bmatrix} \tag{4}$$

another way of expressing is shown in formula (5).

$$X = C^T Y' \tag{5}$$

There are four transposed convolution layers in Fig. 3, where the convolution kernel size of the first layer is 13, and the convolution kernel size of the last three layers is 9. The number of convolution kernels decreases in each layer, which is 256, 128, 64, and 4 respectively.

The architecture of the discriminator is similar to that of the generator, the difference is that the transposed convolution layer in the generator is replaced by a convolution layer, batch regularization is removed, the dropout layer is used instead, and the activation function is replaced.

The transposed convolution corresponds to upsampling the input to generate a higher-dimensional signal, and the convolutional layer is equivalent to downsampling the signal to obtain signal features. For the convolutional layers, in layer 1, we use convolution kernels with a length of 9 and a number of 256 in order to obtain a larger field of view. In layers 2 and 3, the length of kernels is 5, and the number is 128 and 64 respectively. In layer 4, we use a convolution kernel with a length of 8 to reduce the feature signal dimension to 1 to obtain the final classification output.

The specific network parameters are as follows: a total of 5000 epochs are trained, the mini-batch size is set to 16, the learning rate is 0.0002, and the Adaptive Moment Estimation (Adam) optimizer is used to update the network parameters.

C. Model Evaluation

In order to quantitatively evaluate the quality of the signal generated by the generator, we analyze the signal from two perspectives of time domain and frequency domain. In the time domain, we use Dynamic Time Warping (DTW) to calculate the distance between generated and real signals. In the frequency domain, we propose to use Fast Fourier Transform Mean Square Error(FFT MSE), which performs a fast Fourier transform on the generated signal and the real signal to obtain the spectrum, and calculate the mean square error between them.

DTW is a method for comparing the similarity of two-time series. In practical applications, because time series data may have problems such as time axis offset, noise, missing, etc., common similarity measurement methods such as traditional Euclidean distance or correlation coefficient are not applicable, but DTW method can effectively deal with these questions. DTW was originally applied to measure the similarity of speech signals. Specifically, the sound length of the same syllable uttered by the same person may be different, and DTW can be used to measure the similarity of speech signals very well. Similarly, in the sEMG signal, the same subject's movement in the same direction may also take a different time to be reflected on the sEMG signal. Therefore, we use DTW in the sEMG signal to measure their similarity.

The calculation process of DTW can be described as: 1. Suppose the two signals to be compared are $P = \{P_1, P_2, ...P_N\}$, $Q = \{Q_1, Q_2, ...Q_M\}$, initialize an M*N matrix dp, for each element in the matrix$dp(i, j)$, calculate the length of the euclidean distance between them to initialize the matrix, that is

$$dp(i,j) = \sqrt{(P_i^2 - Q_j^2)} \tag{6}$$

2. Using the idea of dynamic programming, traverse from the upper left corner of the matrix to the lower right corner, for each $dp(i, j)$, it represents the minimum distance of the (i, j) position in the matrix. Therefore, it needs to be compared with the elements in the right, bottom, and lower right positions of the matrix, select the minimum distance among them, and accumulate them. That is,

$$dp(i,j) = min(dp(i-1,j), dp(i,j-1), dp(i-1,j-1)) + dp(i,j) \tag{7}$$

3. Traverse each element in P, Q to get the element in the lower right corner of the final matrix, which is the distance between the two signals after DTW calculation.

Since sEMG signals also contain rich information in the frequency domain, we need to measure their similarity again from the perspective of the frequency domain. We use FFT MSE to measure this similarity, and the specific calculation process is as follows:

1. Perform FFT on the two signals separately to obtain their representation in the frequency domain.

2. Perform a difference operation on each point represented by the two frequency domains to obtain their difference.

3. Square the difference to get the mean square error.

4. The mean square error is weighted and averaged to obtain the mean square error of the entire frequency domain.

Finally, what we want to compare is the similarity between the generated signal and the original signal. We use a ratio to measure this similarity, that is, the value of the DTW and FFT MSE of the generated signal is divided by the original signal. The closer the ratio is to 1, It means that the generated signal is more similar to the original signal.

III. Results And Discussions

We evaluate the models from two perspectives of classification accuracy and signal quality. We use the sEMG signal of the subject's standce leg in the dataset for experiments. We train the model with signals of subjects kicking in a certain direction (left or right), causing the model to generate some generated signals. Then, we test the module with these generated signals versus the real signal of the subject kicking in different directions. At the same time, we also compare the accuracy of the model and the long short-term memory network (LSTM) under the original data set. Finally, we quantitatively evaluate the quality of the generated signal with two indicators.

In this model, we don't have a classifier to classify each class of the data. But notice that in the DCGANs network, the output of the discriminator already contains classification information. Specifically, for a binary classification problem, the DCGANs network only learns the features of a certain class and generates signals of this class, where the output of the discriminator is a probability value representing the probability that the discriminator thinks the signal is generated or fake. Therefore, ideally, for another type of signal, the output given by the discriminator should approach 1, which means that the discriminator has learned the difference between the two types of data. For the class used for training, the output of the discriminator should tend to be 0.5, which means that the discriminator cannot distinguish the difference between the

signal generated by the generator and the original signal of this class. We combine these two probabilities as an indicator of the model's predictive accuracy and use the test set mentioned above for evaluation.

We use the signal kicked in the same direction for training, use the same direction signal generated by the generator, and the real signal of the user kicking in a different direction for testing. These two types of signals are fake signals for the model, and we hope to observe the accuracy of the model to identify these two types of signals. Table 2 shows the accuracy of the model to identify these two types of signals, where the first column is the same direction signal generated by the generator, and the second column is the real signal from a different direction. The results show that the accuracy of the model for the first type of signal is in the interval of 0.5351-0.6835, and the accuracy of the second type of signal is in the interval of 0.8184-0.9832.

TABLE I: Accuracy of the model identifying fake signals

	same direction	different direction
subject 1	0.6835	0.8641
subject 2	0.5351	0.9832
subject 3	0.5911	0.9575
subject 4	0.6489	0.8423
subject 5	0.6322	0.8184

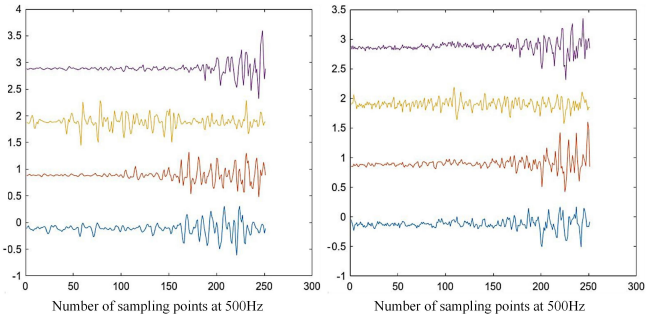
At the same time, we compared the effect of traditional deep learning methods on this binary classification problem. We trained an LSTM network with 20 hidden neurons, input the data of kicks in the same direction and different directions into the network as the training set, and obtained the classification accuracy results as shown in Table 2 Due to the very small size of the data set, the LSTM network has an overfitting phenomenon, and the accuracy on the test set is between 0.6-0.8. Since our method does not directly classify the data, we compare the accuracy of the model's output when correctly identifying kicks in different directions as a measure of accuracy.

TABLE II: Accuracy of Motion Prediction

	LSTM	Our method
subject 1	0.7079	0.8641
subject 2	0.7229	0.9832
subject 3	0.8242	0.9575
subject 4	0.7614	0.8423
subject 5	0.6437	0.8184

In general, the model can well identify sEMG signals in different directions on each subject. For signals in the same direction, due to the small amount of data, a slight overfitting is caused, but the probability is generally maintained at 0.5 Left and right, it can be considered that the discriminator cannot distinguish between the real signal and the generated signal.

Next, we evaluate the signal quality generated by the generator. Figure 4.5 shows the process of the generator iteratively generating a signal from random noise, and Fig. 5 visually compares the final generated signal with the real signal. Each sub-figure is a complete 4-channel signal, and different colors distinguish different channels. In order to distinguish each channel conveniently, the ordinate in the figure has no actual physical meaning.



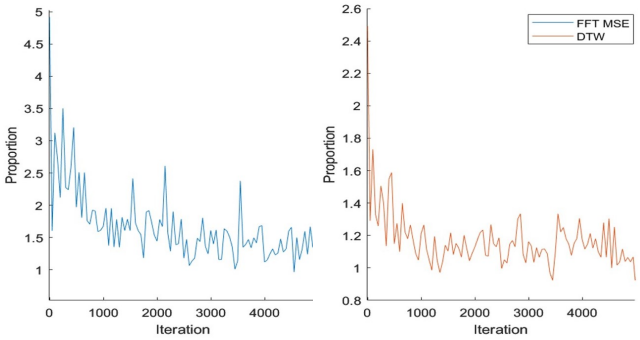
(a) real signal (b) generated signal

Fig. 5: Comparison of real and generated signals

Since the two methods of DTW and FFT MSE are to compare the similarity between two signals, it is impossible to compare two sets of signals with multiple channels. Therefore, our approach is to verify after every 50 iterations and randomly select two from the training set. Signals, calculate the DTW and FFT MSE between each of their channels, and let the generator generate a generated signal, and calculate the DTW and FFT MSE between them.

For comparison, we draw the ratio between the generated signal and the original signal DTW and FFT MSE, as shown in Fig. 6. It can be seen that at the beginning of the training, the generator generates random noise, and the two ratios are relatively large, which are 4.92 and 2.48 respectively. As the training progresses, the two ratios converge to close to 1 respectively and reach respectively at the end of the training 1.04 and 0.87. During the training process, the two ratios fluctuate to a certain extent, because the parameters in the network learning process are constantly updated, and at the same time, the signal extracted by each epoch does not understand the vibration. In general, the two ratios are very close to 1, indicating that the time domain and frequency domain characteristics of the generated signal are very close to the original signal.

In addition, we found that the generated signals are prone to mode collapse during training, that is, the samples generated by the generator are mostly very similar. Although the originally generated signals all belong to the same category, considering the diversity of generated samples, it is necessary to avoid mode collapse as much as possible. In this experiment, we avoid mode collapse by randomly inverting sample labels between each mini-batch. Specifically, we invert the sample labels with a probability of 0.2, i.e. swap labels of a part of the real signal with the generated signal. This inversion does not affect the forward propagation of the gradient, that is, the output of the discriminator for the classification probability of the sample, and only updates the parameters when the net loss is backward propagated. Figure 4.8 shows the advantages



(a) FFT MSE (b) DTW

Fig. 6: The Proportion of DTW and FFT MSE between generated and raw signals, proportion represents the ratio of DTW and FFT MSE of the generated signal to the original signal

of doing this, where the picture on the left of 4.8 is the four sample outputs of the generator before inverting the labels, and the right is the sample output after inversion. It can be seen that the data on the right is obviously messier than the data on the left, which means that the diversity of generated signals is greatly increased. However, reversing labels will also increase the learning difficulty of the discriminator, so it is not advisable to set the reversing probability too high, but to adjust it according to the size and category of the dataset.

IV. Conclusion

In this paper, a novel sEMG signal data-augmented model for motion prediction has been proposed to address the small training data set problem in individual data-driven machine learning of predictive EMG models. We have shown that the deep convolutional generative adversarial networks can learn to generate EMG motion initiation process samples of a given motion class, and this can translate to improved prediction performance compared with LSTM. On the other hand, this preliminary study is limited in sample size (only 5 subjects) and in the characteristics of the data (only healthy adults and simple leg kick tasks). Future studies in real-world use scenarios (involving real patient users while considering multiple variables including gender and disability condition) are needed to further develop and establish the proposed methodology.

Acknowledgment

The research is funded by National Robotics Programme, Singapore under Grant No. M22NBK0074.

References

[1] QL Li, Yu Song, and ZG Hou. Estimation of lower limb periodic motions from semg using least squares support vector regression. *Neural Processing Letters*, 41:371–388, 2015.

[2] Joseph L Betthauser, John T Krall, Shain G Bannowsky, György Lévay, Rahul R Kaliki, Matthew S Fifer, and Nitish V Thakor. Stable responsive emg sequence prediction and adaptive reinforcement with temporal convolutional networks. *IEEE Transactions on Biomedical Engineering*, 67(6):1707–1717, 2019.

[3] Gelareh Hajian and Evelyn Morin. Deep multi-scale fusion of convolutional neural networks for emg-based movement estimation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 30:486–495, 2022.

[4] Corey Pew and Glenn K Klute. Turn intent detection for control of a lower limb prosthesis. *IEEE Transactions on Biomedical Engineering*, 65(4):789–796, 2017.

[5] Ulysse Côté-Allard, Cheikh Latyr Fall, Alexandre Drouin, Alexandre Campeau-Lecours, Clément Gosselin, Kyrre Glette, François Laviolette, and Benoit Gosselin. Deep learning for electromyographic hand gesture signal classification using transfer learning. *IEEE transactions on neural systems and rehabilitation engineering*, 27(4):760–771, 2019.

[6] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478*, 2020.

[7] Elnaz Lashgari, Dehua Liang, and Uri Maoz. Data augmentation for deep-learning-based electroencephalography. *Journal of Neuroscience Methods*, 346:108885, 2020.

[8] Rafael Anicet Zanini and Esther Luna Colombini. Parkinson's disease emg data augmentation and simulation with dcgans and style transfer. *Sensors*, 20(9):2605, 2020.

[9] Tekin Gunasar, Alexandra Rekesh, Atul Nair, Penelope King, Anastasiya Markova, Jiaqi Zhang, and Isabel Tate. Decision forest based emg signal classification with low volume dataset augmented with random variance gaussian noise. *arXiv preprint arXiv:2206.14947*, 2022.