FS-HGR: Few-Shot Learning for Hand Gesture Recognition via Electromyography

Elahe Rahimian, Soheil Zabihi, Amir Asif, *Senior Member, IEEE*, Dario Farina, *Fellow, IEEE*, Seyed Farokh Atashzar, *Member, IEEE*, and Arash Mohammadi, *Senior Member, IEEE*

***Abstract*— This work is motivated by the recent advances in Deep Neural Networks (DNNs) and their widespread applications in human-machine interfaces. DNNs have been recently used for detecting the intended hand gesture through the processing of surface electromyogram (sEMG) signals. *Objective*: Although DNNs have shown superior accuracy compared to conventional methods when large amounts of data are available for training, their performance substantially decreases when data are limited. Collecting large datasets for training may be feasible in research laboratories, but it is not a practical approach for real-life applications. The main objective of this work is to design a modern DNN-based gesture detection model that relies on minimal training data while providing high accuracy. *Methods*: We propose the novel Few-Shot learning- Hand Gesture Recognition (FS-HGR) architecture. Few-shot learning is a variant of domain adaptation with the goal of inferring the required output based on just one or a few training observations. The proposed FS-HGR generalizes after seeing very few observations from each class by combining temporal convolutions with attention mechanisms. This allows the meta-learner to aggregate contextual information from experience and to pinpoint specific pieces of information within its available set of inputs. *Data Source & Summary of Results*: The performance of FS-HGR was tested on the second and fifth Ninapro databases, referred to as the DB2 and DB5, respectively. The DB2 consists of 50 gestures (rest included) from 40 healthy subjects. The Ninapro DB5 contains data from 10 healthy participants performing a total of 53 different gestures (rest included). The proposed approach for the Ninapro DB2 led to 85.94% classification accuracy on *new repetitions with few-shot observation (5-way 5-shot)*, 81.29% accuracy on *new subjects with few-shot observation (5-way 5-shot)*, and 73.36% accuracy on *new gestures with few-shot observation (5-way 5-shot)*. Moreover, the proposed approach for the Ninapro DB5 led to 64.65% classification accuracy on *new subjects with few-shot observation (5-way 5-shot)*.**

***Index Terms*— Myoelectric control, electromyogram (EMG), meta-learning, few-shot learning (FSL).**

Manuscript received November 3, 2020; revised March 30, 2021; accepted April 21, 2021. Date of publication May 4, 2021; date of current version June 7, 2021. This work was supported in part by the Department of National Defence's Innovation for Defence Excellence and Security (IDEaS), Canada, and in part by the Borealis AI through the Borealis AI Global Fellowship Award. The work of Seyed Farokh Atashzar was supported by the U.S. National Science Foundation under Award 2037878. *(Corresponding author: Arash Mohammadi.)*

Elahe Rahimian and Arash Mohammadi are with the Concordia Institute for Information System Engineering (CIISE), Concordia University, Montreal, QC H3G 2W1, Canada (e-mail: e_ahimia@encs.concordia.ca; arash.mohammadi@concordia.ca).

Soheil Zabihi is with the Department of Electrical and Computer Engineering, Concordia University, Montreal, QC H3G 2W1, Canada (e-mail: s_zab@encs.concordia.ca).

Amir Asif is with the Department of Electrical Engineering and Computer Science, York University, Toronto, ON M3J 1P3, Canada (e-mail: asif@eecs.yorku.ca).

Dario Farina is with the Department of Bioengineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: d.farina@imperial.ac.uk).

Seyed Farokh Atashzar is with the Department of Electrical and Computer Engineering, the Department of Mechanical and Aerospace Engineering, the NYU WIRELESS Center, and the NYU Center for Urban Science and Progress (CUSP), New York University (NYU), Brooklyn, NY 11201 USA (e-mail: sfa7@nyu.edu).

Digital Object Identifier 10.1109/TNSRE.2021.3077413

I. Introduction

THE recent advances in Machine Learning (ML) and Deep Neural Networks (DNNs) coupled with innovations in rehabilitation technologies have resulted in a surge of significant interest in the development of advanced myoelectric prosthesis control systems. Hand motion recognition via surface Electromyogram (sEMG) signals [1], [2] is considered as a central approach in the literature. Conventional ML techniques, such as Linear Discriminant Analysis (LDA) [3]–[5] and Support Vector Machines (SVMs) [3], [4], [6], have been used for detecting the intended hand gesture through processing of sEMG signals. Although classical pattern-recognition-based myoelectric control has been widely studied in academic settings over the last decades, the advanced methodologies have not been used in many commercial examples. This is due to a noticeable gap [7], [8] between real-world challenges and existing methodologies. Among the reasons for this gap are:

(i) *Training Time:* The first problem is the extended training time required by the end-user to mitigate the differences between the desired and performed movements. Such a training process, which is time consuming, tedious and unpleasant, can take up to several days in practice.

(ii) *Variability in the Characteristics of sEMG Signals:* The second issue is the variability in nature of the sEMG signals. This variability is caused by (a) Time-dependent and stochastic nature of the neural drive to muscles; (b) Dependency of the neural drive to the dynamic and kinematics of tasks, and; (c) Variability in neural control strategies between different users and the changes caused by amputations. In addition, sEMG recording could vary based on electrode location. Given such variations, therefore, the probability distributions of sEMG signals may be different over time. Consequently, models trained based on some specific observations may not consistently and directly be reused over time. This would require retraining and recalibration, which cannot be done often in real-life applications.

Recently, DNNs have been designed and used by our team [9]–[12] and other research groups [13]–[19], for myocontrol, achieving superior classification performance than conventional approaches. For example, in [19], which is among the first DNN-based methods developed for the analysis of sEMG data, it was shown that results of a DNN with a very simple architecture are comparable to the average result of classical methods. More specifically, the average classification accuracy obtained using a simple CNN architecture on Ninapro DB2 was reported as $60.27 \pm 7.7\%$. The average classification accuracy obtained using all the classical methods on this dataset is $60.28 \pm 6.51\%$. The best classical classification method (Random Forests (RF) with all features) obtained an average classification accuracy of $75.27 \pm 7.89\%$. It is worth noting that the DNN performance depends on several factors such as pre-processing and the designed architecture. The optimization of parameters is, therefore, fundamental for the performance. Consequently, there have been several efforts to design advanced DNN architectures. For instance, the average classification accuracy obtained based on recent works [17], [18] is 82.95%, which shows their superior performance compared to classical methods. However, DNNs need large training data to achieve high performance. This may be feasible in laboratory conditions but poses constraints in the practical use of prostheses in real-life applications. There is an unmet need for the design of a modern gesture detection technique that relies on minimal training data while achieving high performance.

In this paper we introduce, for the first time, the concept of few-shot training for myoelectric systems. Few-shot learning minimizes the need for recalibration and would allow the user to retrain the ML core of control, by only few basic exercises instead of extensive recalibration procedures. For this purpose, here we propose an innovative *"Few-Shot learning-Hand Gesture Recognition", referred to as the FS-HGR*. The proposed meta-learning FS-HGR architecture takes advantage of domain knowledge and requires a small amount of training data (when compared with traditional counterparts) to decode new gestures of the same or new users. The paper makes the following contributions:

- A class of architectures is introduced for sEMG meta-learning, where the meta-learner, via adaptation, quickly incorporates and refers to the experience based on just few training observations.
- By proposing the FS-HGR framework, which utilizes a combination of temporal convolutions and attention mechanisms, we provide a novel venue for adopting few-shot learning, to not only reduce the training time, but also to eventually mitigate the significant challenge of variability in the characteristics of sEMG signals. In other words, the proposed FS-HGR framework allows a myoelectric controller, that has been built based on background data, to adapt to the changes in the stochastic characteristics of sEMG signals using a small number of new observations.

The paper is organized as follows: Section II provides a brief overview of relevant literature. In Section III, we present the dataset used in development of the proposed FS-HGR framework together with the pre-processing step. The proposed FS-HGR architecture is developed in Section IV. Experimental results and different evaluation scenarios are presented in Section V. Finally, Section VI concludes the paper.

II. Related Works

A common strategy used for hand gesture recognition in recent works is applying DNN with the focus on improving hand gestures classification performance on "*never-seen-before repetitions*". Along this line of research, several state-of-the-art works [10]–[12], [14], [16]–[22] mainly used the Ninapro database [23]–[25], which is a public dataset providing kinematic and sEMG signals from 52 finger, hand, and wrist movements. The Ninapro database is similar to data obtained in real-world conditions, and as such it allows development of advanced DNN-based recognition frameworks.

The common approach in recent studies [10]–[12], [14], [16]–[22], following the recommendations provided by the Ninapro database, is to train DNN-based models on a training set consisting of approximately $2/3$ of the gesture trials of each subject. The evaluation is then performed on the remaining trials constituting the test set. Although existing DNN techniques achieve promising performance on never-seen-before repetitions, they fail to function properly if the repetition is not extensively explored [26]–[28]. For example, in [29], the authors reported the average accuracy over the 10 participants of the Ninapro DB5 for one to four training repetitions (one repetition is equal to 5 seconds of data). The accuracy decreases and the model fails to function properly if the repetition is not extensively explored. For example, for one to four training repetitions the accuracies are equal to 49.41 ± 5.82; 60.12 ± 4.79; 65.16 ± 4.46, and; 68.98 ± 4.46, respectively. Thus, for a new user or a new gesture, a significant amount of training should be conducted and the whole learning process should be redone, assuming a small variation between the new class and the previous classes. If the aforementioned change is more than minimal, there may be the need to recalibrate the whole process for all classes. In addition, existing DNN-based methodologies require large training datasets and perform poorly on tasks with only a few observations being available for training purposes.

In [30], the authors proposed a domain adaptation method that maps both the original and target data into a common domain, while keeping the topology of the input data probability distributions. For this purpose, the authors used a local dataset, where the sEMG data was acquired by repetitive gripping tasks while data was collected from 8 subjects. In addition to the above, Transfer Learning (TL) was also used to adopt a pre-trained model and leverage the knowledge acquired from multiple subjects and speed up the training process for the new users. In [29], [31], the authors proposed a TL-based algorithm adopting CNN to transfer knowledge across multiple subjects for sEMG-based hand gesture recognition. The authors in [29], [31], applied the Myo armband to collect sEMG signals and used the fifth Ninapro database, which contains data from 10 intact-limb subjects. The pre-training for each participant was done employing the training sets of the remaining nine participants and the

average accuracy was obtained over the 10 participants of the Ninapro DB5 [6]. Finally, [32], [33] applied deep learning along with domain adaptation techniques for inter-session classification to improve the robustness for the long-term uses. Due to the variability of the signal space, the generalizability of existing techniques is questionable and it is not clear how they would perform in real-life scenarios when the training data is limited. It is not clear how these models would perform on scenarios with larger number of subjects and postures. For example, in [32], 7 subjects participated in the experiment and 7 movements were classified. As another example, in [33], the authors separately used three datasets to train and evaluate their model based on High-density surface electromyogram (HD-sEMG) signals. The first dataset referred to as the CSL-HDEMG consists of 5 subjects performing 27 gestures. The second and third datasets referred to as CapgMyo DB-b and DB-c, respectively, consist of 23 subjects performing 8 gestures for DB-b and 12 gestures for DB-c. In summary, the number of subjects and movements in previous studies were relatively small - particularly in comparison with the Ninapro database, therefore, making it difficult to explore the generalizability of the existing techniques and has motivated us to focus on this relatively large-scale sEMG database.

In summary, there is an urgent need to develop adaptive learning methods with the focus on designing a classifier which can be adopted for new subjects based on only a few observations through a fast learning approach. This is a challenging task since many factors, such as electrode location and muscle fiber lengthening/shortening, can affect the collected sEMG signals. Moreover, the differences between users and the changes caused by amputations result in discrepancies between different conditions [2], [8]. To the best of our knowledge, this is the first time that *Few-shot Learning* is adopted in the literature to classify 49 hand gestures on *new* subjects using a small (one to five) number of training observations.

III. Material and Methods

A. Database

The proposed FS-HGR architecture was evaluated on the Ninapro [23]–[25] benchmark database, which is a publicly available dataset for hand gesture recognition tasks. Ninapro is a widely used benchmark for evaluation of different models developed using sparse multichannel sEMG signals.

In this work, the second Ninapro database [23] referred to as the DB2 was utilized. Delsys Trigno Wireless EMG system with 12 wireless electrodes (channels) was used in the DB2 dataset to collect electrical activities of muscles at a rate of 2 kHz. The dataset consists of signals collected from 28 men and 12 women with age 29.9±3.9 years, among whom 34 are right-handed and 6 are left-handed. The DB2 consists of 50 gestures including wrist, hand, grasping, and functional movements along with force patterns from 40 healthy (intact-limb) subjects. The subjects repeated each movement 6 times, each time lasted for 5 seconds followed by 3 seconds of rest. More detail on the Ninapro database are described in [23].

B. Pre-processing Step

Following the pre-processing procedure established in previous studies [16], [19], [22], [23], we used a 1st order low-pass Butterworth filter to smooth the electrical activities of muscles. Moreover, we applied μ-*law* transformation to magnify the output of sensors with small magnitude (in a logarithmic fashion), while keeping the scale of those sensors having larger values over time. This transformation approach has been used traditionally in speech and communication domains for quantization purposes. We propose to use it for scaling the sEMG signals as a pre-processing approach. The μ-*law* transformation was performed based on the following formulation

$$F(x_t) = \text{sign}(x_t)\frac{\ln\left(1+\mu|x_t|\right)}{\ln\left(1+\mu\right)}, \tag{1}$$

where $t \geq 1$ is the time index; x_t denotes the input to be scaled, and the parameter μ defines the new range. Here $\mu = 2,048$ was utilized, i.e., the scaled data points were distributed between 0 and 2,048. Afterwards, we fed the scaled sEMG signals to Minmax normalization. In our previous work [11], we empirically observed that the normalization of the scaled sEMG signals is better than non-scaled sEMG signals. For example, the results obtained without scaling for a window of length 50 ms was 71.49%, while normalization of scaled sEMG signals has improved the results to 81.71%. In this work, we noticed a similar trend and as such continued using normalization of scaled sEMG signals. This completes a brief introduction of the utilized dataset and the introduced pre-processing step. Next, we develop the proposed Meta Learning-based FS-HGR framework.

IV. The FS-HGR Framework

Meta-learning can be formalized as a sequence-to-sequence learning problem. The bottleneck is in the meta-learner's ability to internalize and refer to experience. To address this shortcoming for the gesture recognition task based on sparse multichannel sEMG, inspired by [26], we proposed a class of model architectures by combining temporal convolutions with attention mechanisms to enable the meta-learner to aggregate contextual information from experience. This integrated architecture allows the meta-learner to pinpoint specific pieces of information within its available set of inputs. Our main goal is to construct and train a hand gesture recognition model that can achieve rapid adaptation. Next, we first elaborate on the meta-learning concept.

A. The Meta-Learning Problem

A supervised learning task starts with a given dataset $\mathcal{D} = \{(\boldsymbol{X}_i, y_i)\}_{i=1}^{M}$, consisting of M observations, where the i^{th} observation is denoted by $\boldsymbol{X}_i$, for $(1 \leq i \leq M)$, with its associated label denoted by y_i. The main objective is to learn a (possibly non-linear) function $f(\cdot)$ defined based on its underlying parameters θ that maps each observation $\boldsymbol{X}_i$ to its corresponding label, $y_i = f(\boldsymbol{X}_i; \theta)$. In a supervised learning approach, the dataset is divided into: (a) The training data $\mathcal{D}_{train}$ used for learning the parameters θ of the model; (b) The validation data $\mathcal{D}_{val}$ utilized for tuning the hyper-parameters of the model, and; (c) The test data $\mathcal{D}_{test}$ for model evaluation.

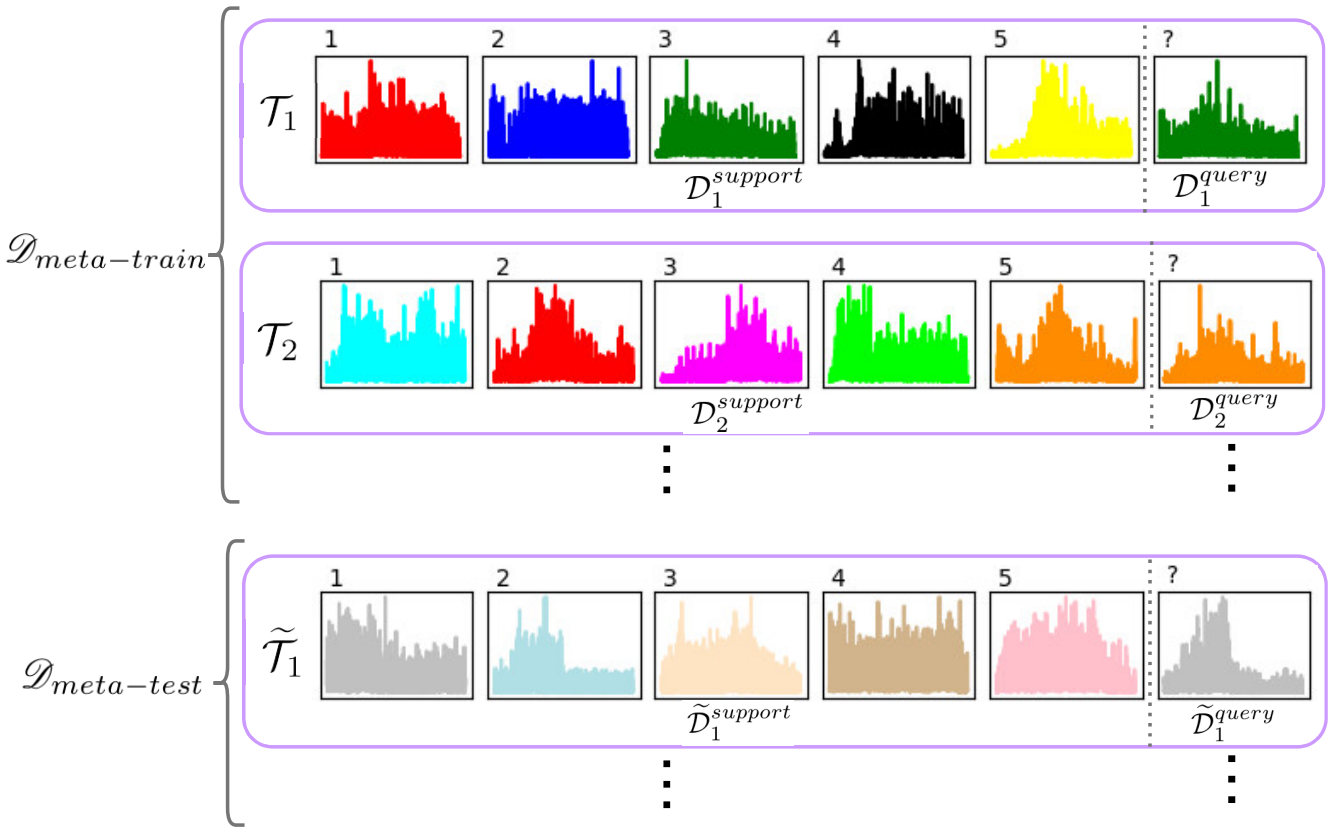


Fig. 1. 5-way 1-shot classification: Each task $\mathcal{T}_j$, represented in a purple box, is associated with a support-set $\mathcal{D}^{support}$ and a query-set $\mathcal{D}^{query}$. Here, for constructing $\mathcal{D}^{support}$, first, 5 classes are selected from the $\mathscr{D}_{meta\text{-}train}$, and then one observation from each of these 5 classes are selected (each class is represented with a different colour and is associated with a label 1-5). $\mathcal{D}^{query}$ consists of 1 observation selected from one of those 5 classes. The $\mathscr{D}_{meta\text{-}test}$ is represented in the same approach, covering a different set of datasets, which do not include any classes presented in any of the datasets in $\mathscr{D}_{meta\text{-}train}$. Finally, $\mathscr{D}_{meta\text{-}val}$ is defined in the same way to determine the hyper-parameters of the model.

In this context, we focused on meta-supervised learning, where the goal is generalization across tasks rather than across data points. Therefore, instead of using the aforementioned conventional data subsets (Items (a)–(c) above), we have a meta-set denoted by $\mathscr{D}$, which in turn splits into meta-train $\mathscr{D}_{meta-train}$, meta-validation $\mathscr{D}_{meta-val}$, and meta-test $\mathscr{D}_{meta-test}$ sub-datasets. Furthermore, one needs to construct different tasks (as shown in Fig. 1) within each meta-dataset. Task $\mathcal{T}_j \in \mathscr{D}$ is episodic and is defined by two components, a support-set $\mathcal{D}_j^{support}$ and a query-set $\mathcal{D}_j^{query}$, i.e., $\mathcal{T}_j = (\mathcal{D}_j^{support}, \mathcal{D}_j^{query})$.

Within the context of meta-learning, our focus is specifically on few-shot learning (typically referred to as k-shot learning with k being a small integer), which is briefly described next. In a N-way k-shot classification, our goal is training on $\mathscr{D}_{meta-train}$, where the input is the support-set $\mathcal{D}_j^{support}$ and, a query instance $\boldsymbol{X}_j^{query} \in \mathcal{D}_j^{query}$. To be more precise, $\mathcal{D}_j^{support} = \{(\boldsymbol{X}_i, y_i)\}_{i=1}^{k\times N}$, where N classes are selected from the meta-train set, and then k observations are selected from each of these classes. To make predictions about a new test data point, $\boldsymbol{X}_j^{query} \in \mathcal{D}_j^{query}$, we produce a mapping function $f(\cdot)$ that takes as input $\mathcal{D}_j^{support}$ and $\boldsymbol{X}_j^{query}$ to produce the label $\hat{\mathrm{y}}_j^{query} = f(\mathcal{D}^{support}, \boldsymbol{X}_j^{query}; \theta)$. Hyper-parameter selection is performed by using $\mathscr{D}_{meta-val}$. Generalization performance of the meta-learner is then evaluated on the $\mathscr{D}_{meta-test}$ [27].

Fig. 1 shows a $N = 5$-way $k = 1$-shot classification task, where inside each purple box is a separate dataset $\mathcal{T}_j$ consisting of the support-set $\mathcal{D}_j^{support}$ (on the Left-Hand Side (LHS) of the dashed line) and the query-set $\mathcal{D}_j^{query}$ (on the Right-Hand Side (RHS) of the dashed line). In the illustrative example of Fig. 1, we are considering a 5-way 1-shot classification task where for each dataset, we have one observation from each of the 5 classes (each given a label 1 to 5) in the support-set and 1 observation for evaluation from the query-set of that specific task. For training the model $\mathscr{D}_{meta-train}$ is used, where each Task $\mathcal{T}_j$ is drawn from $p(\mathcal{T})$ distribution, while during the test procedure $\mathscr{D}_{meta-test}$ is used, which consists of unseen tasks randomly sampled from a different distribution (i.e., $\widetilde{\mathcal{T}}_j \sim p(\widetilde{\mathcal{T}})$), where $p(\widetilde{\mathcal{T}})$ is similar in nature to $p(\mathcal{T})$. As shown in Fig. 1, task $\widetilde{\mathcal{T}}_j$ is associated with a dataset $\widetilde{\mathcal{D}}_j$ splitting into two parts, i.e., the support-set $\widetilde{\mathcal{D}}_j^{support}$ and the query-set $\widetilde{\mathcal{D}}_j^{query}$ (Fig. 1). For each task, we measure performance on the $\widetilde{\mathcal{D}}_j^{query}$ based on the knowledge of its cosponsoring $\widetilde{\mathcal{D}}_j^{support}$.

B. Description of the FS-HGRModel

In few-shot classification, the goal is to reduce the prediction error on data observations with unknown labels given a small training set. Inspired by [26], the proposed FS-HGR network receives as input a sequence of observation-label pairs $\mathcal{D}_j^{support} = \{(\boldsymbol{X}_i, y_i)\}_{i=1}^{k\times N}$, followed by $\mathcal{D}_j^{query}$, which consists

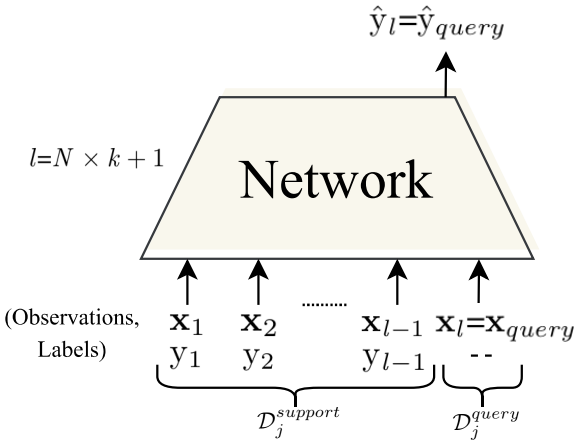


Fig. 2. For each task $\mathcal{T}_j$, the set of observations and labels are concatenated together and sequentially fed to the model. The final observation is concatenated with a null label instead of True label. The network is supposed to predict the missing label of final observation based the previous labels that it has seen. In *N*-way *k*-shot classification, *N* shows the number of classes which are selected from whole set of labels, and *k* shows the observations that are sampled from each of those *N* classes.

Algorithm 1 The Training Procedure

Input: $\mathscr{D}_{meta-train}$, and; mapping function $f(\cdot)$ with parameters θ.

Require. $p(\mathcal{T})$: distribution over tasks

1: **while** not done **do**
2: Sample batch of tasks $\mathcal{T}_j \sim p(\mathcal{T})$
3: **for all** $\mathcal{T}_j$ **do**
4: Split $\mathcal{T}_j$ into $\mathcal{D}_j^{support}$ and $\mathcal{D}_j^{query}$
5: Predict the missing label of final observation of $\mathcal{T}_j$: $\hat{y}^{query} = f(\mathcal{D}_j^{support}, \mathcal{D}_j^{query}; \theta)$
6: **end for**
7: Update θ using $\Sigma_{\mathcal{T}_j \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_j}(\hat{y}^{query}, y^{query})$
8: **end while**

of an unlabelled observation. The meta-learning model predicts the label of the final observation based on the previous labels that it has seen. During the training phase, first, we select N classes, with k observations per $\mathcal{D}_j^{support}$ (in terms of our running illustrative example, for each task, we have $k = 1$ observation from each of the underlying $N = 5$ classes). For constructing the $\mathcal{D}_j^{query}$, we select an extra observation from one of those selected classes. Afterwards, each set of the observations and labels are concatenated together (the final observation is concatenated with a null label instead of the ground truth label as it is used for evaluation purposes), and then all $(N \times k + 1)$ are sequentially fed to the network. Finally, the loss $\mathcal{L}_j$ is computed between the predicted and ground truth label of the $(N \times k + 1)^{th}$ observation. During such a training mechanism, the network learns how to encode the first $N \times k$ observations to make a prediction about the final observation [26]. The training procedure is described in Algorithm 1 and the schematic of the model is shown in Fig. 2.

C. The Building Modules of the FS-HGR Framework

After completion of the pre-processing step, sEMG signals acquired from N_S number of sensors are segmented by a window of length of $W = 200$ ms selected to satisfy the acceptable delay time [34], i.e., the window length W is required to be under 300 ms. With a larger window of 300 ms, the results would likely improve. However, the use of shorter windows (e.g., 200 ms or 260 ms) provides an extra time (100 ms and 40 ms, respectively) to perform the pre-processing and classification tasks, which allows staying within the target 300 ms. A second reason for using a window of duration 200 ms is to perform a fair comparison with prior works [17]–[22] reported in Table I. Finally, sliding window with steps of 50 ms is considered for segmentation of the sEMG signals. By using overlapping, there are more observations for training the underlying architecture. In [18], [21], a sliding window with steps of 100 ms was considered for segmentation of the sEMG signals. On the other hands, in [17], [20] a sliding window with steps of 10 ms was used. In all these previous studies, the window size was 200 ms. The larger the overlapping size, the more training data are available, i.e., the more augmentation. However, extended augmentation increases the training time. In our work, for providing a fair comparison and to keep a reasonable training time, we considered a sliding window with steps of 50 ms, which is approximately between the values considered in the prior works.

1) The Embedding Module: To develop the FS-HGR for few-shot learning, we aimed to first extract a 128-dimensional feature vector from each observation with size of $(W = 200 \times N_S = 12)$. The "Embedding Module" is, therefore, used to extract a 128-dimensional feature vector, which is then provided as input to the proceeding modules within the proposed architecture.

Adopting a proper Embedding Module has a significant effect on the results. For validating our claim, therefore, we utilized four different Embedding Modules:

(i) The first Embedding Module, referred to as the *FC Embedding*, consists of three Fully Connected (FC) layers to output a 128-dimensional feature vector from each observation. The first FC layer in the FC Embedding Module is used to increase the input dimensional to $(W \times 128)$. Subsequently, the second (which is followed by ReLU activation function) and third FC layers with output size of 100 and 1, respectively, are adopted to reduce the sequence length of each observation to (1×128) (Fig. 3(a));

(ii) *LSTM Embedding:* Fig. 3(b) illustrates the second Embedding Module, referred to as the *LSTM Embedding*, which utilizes a Long Short-Term Memory (LSTM) layer as its first block followed by two FC layers. The LSTM layer takes the observation with input size 12 and converts it to an output with 128 features. Then, the two FC layers are adopted to reduce the observation's sequence length to 1;

(iii) *T-Block Embedding I:* This third Embedding Module utilizes the *TemporalBlock Module* (which will be described in next sub-section) consisting of $f = 128$ 1D-Convolutions with kernel size $k_S = 2$, and dilation factor $d = 1$ as its first block. The *TemporalBlock Module* is followed by two FC layers to decrease the input's sequence length to 1 as shown in Fig. 3(c), and;

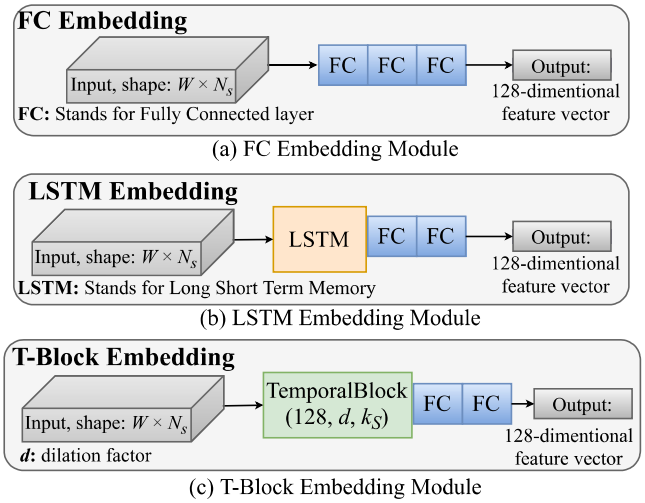


Fig. 3. **The Embedding Module**, which converts an input with size ($W \times N_S$), W stands the window length and N_S shows the number of sensors (input features), to a 128-dimensional feature vector. (a) **FC Embedding Module**, which uses three FC layers to outputs a 128-dimensional feature vector. (b) **LSTM Embedding Module**, which adopts a LSTM layer followed by two FC layers. (c) **T-Block Embedding Module**, which consists of a Temporal Block with number of filters $f = 128$, kernel size $k_S = 2$, and dilation factor d, followed by two FC layers.

(iv) *T-Block Embedding II:* This embedding is similar in nature to the one described above in Item (iii), however, here the goal is to examine the effect of increasing the size of the receptive field. As such, the fourth Embedding Module utilizes two *TemporalBlock Modules* with $d = 1$ and $d = 2$. It is noteworthy to mention that the first FC layer in both LSTM and T-Block Embedding modules are followed by ReLU activation function.

2) The TemporalBlock Module: Inspired by [10], [12], [26], [35], [36], the proposed FS-HGR few-shot learning architecture utilizes *Dilated Causal 1D-Convolutions* over the temporal dimension. The proposed architecture, therefore, provides several advantages over Recurrent Neural Networks (RNNs) such as low memory requirement and faster training. In addition, and unlike conventional CNNs, by incorporation of dilated causal convolutions, we increased the receptive field of the network and as such benefit from the time-series nature of the input.

As shown in Fig. 4(a), each *TemporalBlock* consists of two dilated causal 1D-convolutions, each with dilation factor d, kernel size k_S, and f number of filters. To learn the complex structure of the underlying data, each dilated causal 1D-convolutions is followed by a ReLU activation function. Finally, by concatenating the results and the input, the training speed can be considerably improved. This module takes an input with size ($C_{in} \times l$) and output a tensor with size ($C_{out} \times l$). Here, l denotes the sequence length and is equal to ($N \times k + 1$).

3) The TemporalConvNet Module: The benefit that comes with the designed "TemporalConvNet" module is that its training procedure is much faster and efficient compared to LSTM or Gated Recurrent Unit (GRU) architectures. In other words, through this approach one complete sequence can be processed through only one forward pass, while in RNN-based models this, typically, needs several passes due to temporally linear hidden state dependency. The TemporalConvNet module consists of a series of TemporalBlock modules with exponentially growing dilation factors d. More specifically, as shown in Fig. 4(b), for an input with sequence length $l = (N \times k + 1)$, the TemporalConvNet consists of $Z = \lceil \log_2 l \rceil$ number of TemporalBlock modules. The dilation factors d for the TemporalBlock modules are equal to $[1, 2, 4, \ldots, 2^{Z-1}]$, respectively.

4) The Attention Module: The final constituent module within the proposed FS-HGR architecture is referred to as the "Attention Module," included with the objective of pinpointing a specific type of information within the available (possibly significantly large) context [37]. Attention mechanism has been recently utilized [13] within the context of sEMG-based hand gesture recognition, where the experiments showed attention's capability to learn a time-domain representation of multichannel sEMG data. By integrating the TemporalConvNet, described above, and the Attention Module, essentially we provided the FS-HGR architecture with the capability to access the past experience without any limitations on the size of experience that can be used effectively. Furthermore, in the FS-HGR framework we used the Attention Module at different stages to provide the model with the ability to learn how to identify and select pieces of useful information and its appropriate representation from its experience.

As shown in Fig.4(c), to get queries, keys, and values, three linear transformations are applied to the input. The attention mechanism then compares queries to each of the key values with a dot-product, scaled by $\sqrt{d_k}$, which results compatibility scores. To obtain attention distribution over the values, softmax function is applied to the scores. Then, we computed the weighted average of the values, weighted by the attention distribution. In practice, the keys, values, and queries are packed together into matrices $\boldsymbol{K}$, $\boldsymbol{V}$, and $\boldsymbol{Q}$, respectively. The matrix of outputs is obtained as follows:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}})\boldsymbol{V}, \tag{2}$$

where d_k stands for length of the key vector in matrix $\boldsymbol{K}$. Then, the results and inputs are concatenated together. This completes description of the modules incorporated to construct the proposed FS-HGR framework.

D. The Architecture

The overall structure of the proposed FS-HGR architecture consists of four Attention modules, where the first three ones are followed by a TemporalConvNet module. The final Attention module is followed by a FC layer to produce the label of the final observation in each task $\mathcal{T}_j$. More specifically, after feeding each observation with size ($W \times N_S$) to an Embedding Module, we obtained a 128-dimensional feature vector (Fig. 3). Then, for constructing each task $\mathcal{T}_j$ with sequence length l (Fig. 2), the set of observations (each observation is converted to a 128-dimensional feature vector) and labels are concatenated. The final observation in the sequence is concatenated with a null label instead of a True label. The network is supposed to predict the missing label of the final observation based on the previous labels that it

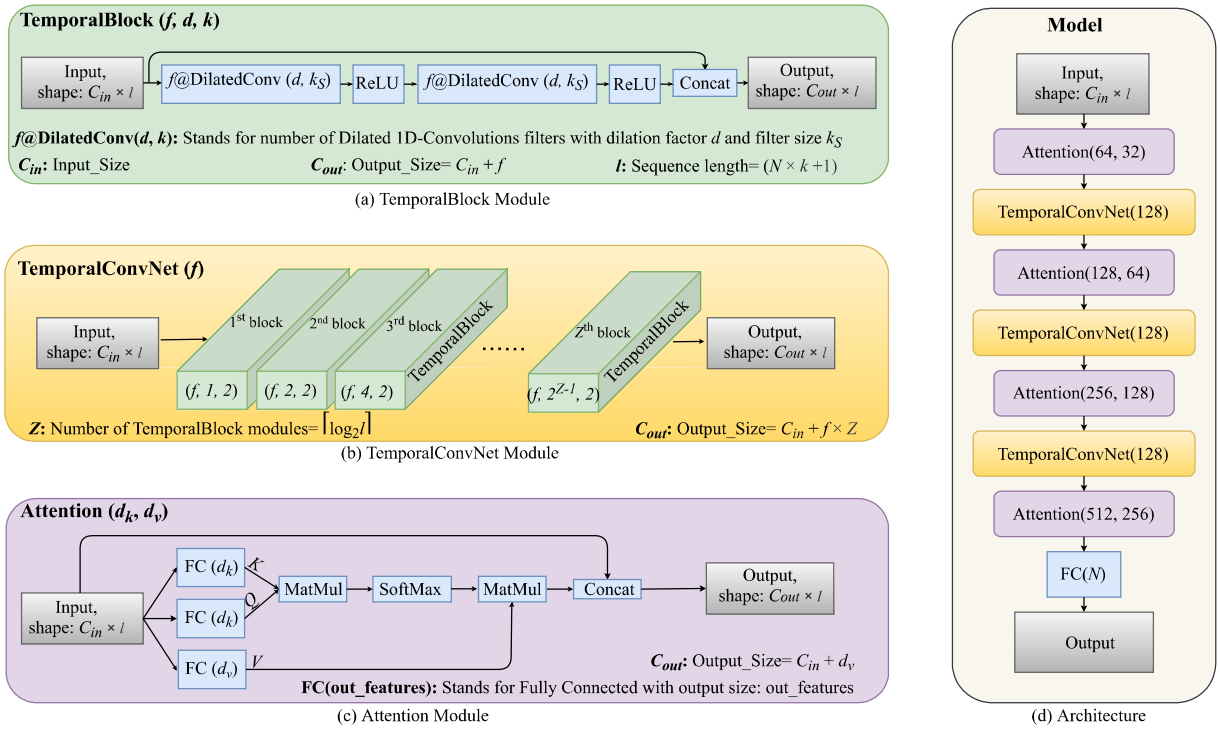


Fig. 4. (a) **The TemporalBlock Module**, which consists of f Dilated Causal 1D-Convolutions with dilation factor d and kernel size k_S. This module converts an input with C_{in} features to an output with C_{out} features. The sequence length of the input l is equal to $(N \times k + 1)$, which N shows the number of classes and k denotes the number of observations of each class. (b) **The TemporalConvNet Module**, which consists of a series of TemporalBlock modules (green ones). The kernel size of each TemporalBlock Module is equal to 2; however, their dilation factor d increases exponentially. (c) **The Attention Module**, which consists of three FC layers with output size d_k, d_k, and d_v, respectively, to produce matrix $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$. (d) **The Architecture**, consisting of three TemporalConvNet modules (yellow ones), and four Attention modules (purple ones). Here, 128 denotes the number of filters f in Dilated 1D-Convolutions. The architecture is supposed to predict the missing label of the $(N \times k + 1)^{th}$ observation in each task $\mathcal{T}_j$.

has seen. In summary, to perform the hand gesture recognition task, the FS-HGR framework is constructed based on different modules as shown in Fig. 4(d).

V. Experiments and Results

In this section, we describe a comprehensive set of experiments to analyse and evaluate the proposed FS-HGR framework. At stated previously, in few-shot classification, we would like to classify inputs in N classes when we have just k observations per class. More specifically, for N-way k-shot classification, to construct each Task $\mathcal{T}_j = (\mathcal{D}_j^{support}, \mathcal{D}_j^{query})$, for $(1 \leq j \leq N_{\text{Tasks}})$, first, we randomly select N classes from the total number of $\mathcal{N}$ available classes in the meta-set $(N \ll \mathcal{N})$. Then, we select k observations from each of those selected N classes. These k observations together constitute the support-set $\mathcal{D}_j^{support}$. An additional observation is randomly selected to form the query-set $\mathcal{D}_j^{query}$. In the experiments, there are a total of $\mathcal{N} = 49$ classes in the meta-set, each class corresponding to a specific hand gesture. For example, consider the $N = 10$-way $k = 5$-shot classification scenario. To construct each Task, we randomly select $N = 10$ out of the $\mathcal{N} = 49$ available classes and then from each class randomly select $k = 5$ observations to form the support-set $(\mathcal{D}_j^{support})$ for the j^{th} Task. Additionally, one extra observation is randomly selected from one of the $N = 10$ classes to form the query-set $(\mathcal{D}_j^{query})$ for the j^{th} Task. In the $N = 10$-way $k = 5$-shot classification scenario, therefore, each task consists of $N \times k + 1 = 51$ number of observations selected randomly. In a training experiment, we consider $10,000$ iterations per epochs. Therefore, with a batch-size of 64, we create $640,000$ tasks per epoch. We consider 25 epochs for training. As a final note, within the few-shot learning context, it is common to report the results based on different values of N and k. In the experiments, we followed the common practice of using $N = 5$ and $N = 10$ together with $k = 1$ and $k = 5$. By increasing the number of classes (N) in each Task $\mathcal{T}_j$, the classification accuracy will decrease. At the same time, by increasing the number of observations per class (k), the classification accuracy is expected to improve as there are more observations from each of the underlying N classes.

In the following, we present three evaluation scenarios. In all experiments, Adam optimizer was used for training purposes with learning rate of 0.0001. Different models were trained with a mini-batch size of 64 except in 10-way 5-shot classification where mini-batch size of 32 was used. For measuring the classification performance, the loss $\mathcal{L}_j$ was computed between the predicted and ground truth label of $(N \times k + 1)^{th}$ observation in each task $\mathcal{T}_j$. The average loss was computed using Cross-entropy loss. Finally, the average accuracy is reported on the $(N \times k + 1)^{th}$ observation.

Experiment 1: Classification on New-Repetitions With Few-Shot Observation: The first experiment shows that our proposed network is applicable when we had new repetitions with few-shot observation on the target. We evaluated our proposed architecture when $\mathcal{D}_{meta-train}$ consisted of the 2/3

TABLE I

EXPERIMENT 1: 5-WAY, 1-SHOT, 5-SHOT, AND 10-SHOT CLASSIFICATION ACCURACIES ON *new repetitions with few-shot observation*. THE CLASSIFICATION ON NEW REPETITIONS WITH FEW-SHOT OBSERVATION ARE PERFORMED BY USING META-SUPERVISED LEARNING APPROACH. THIS TABLE ALSO SHOWS A COMPARISON BETWEEN OUR METHODOLOGY (META-SUPERVISED) LEARNING AND PREVIOUS WORKS WHERE SUPERVISED LEARNING METHODOLOGY, i.e., DNN AND CLASSICAL ML METHODS ARE USED

Proposed Method (FS-HGR)		**5-way Accuracy (%) ± STD**		
	Embedding Module	**1-shot**	**5-shot**	**10-shot**
Meta-Supervised Learning	FC Embedding	72.59 ± 0.15	85.13 ± 0.15	89.26 ± 0.14
	LSTM Embedding	**75.03** ± 0.15	84.06 ± 0.14	88.45 ± 0.12
	T-Block Embedding I	73.46 ± 0.15	**85.94** ± 0.13	89.40 ± 0.14
	T-Block Embedding II	74.89 ± 0.15	85.88 ± 0.14	**89.70** ± 0.15
Previous Works		**Method**	**Accuracy (%)**	
DNN-based Methods	Wei *et al.* [17]	CNN	**83.70**	
	Hu et al. [18]	Hybrid CNN-RNN	82.20	
	Ding *et al.* [20]	CNN	78.86	
	Zhai *et al.* [21]	CNN	78.71	
	Geng *et al.* [22]	CNN	77.80	
Classical Methods	Zhai *et al.* [21]	SVM	**77.44**	
	Atzori *et al.* [19]	RF	75.27	
	Pizzolato *et al.* [6]	RF	72.25	

TABLE II

TRAIN AND TEST TIME FOR ONE TASK $\mathcal{T}_j$ USING 5-WAY 1-SHOT, 5-WAY 5-SHOT, AND 5-WAY 10-SHOT FOR EXPERIMENT 1

The Embedding Module	**5-way Accuracy**					
	1-shot		**5-shot**		**10-shot**	
	train time	**test time**	**train time**	**test time**	**train time**	**test time**
FC Embedding	0.736 ms	0.175 ms	3.034 ms	0.856 ms	6.378 ms	1.428 ms
LSTM Embedding	1.242 ms	0.537 ms	4.930 ms	1.598 ms	10.417 ms	2.815 ms
T-Block Embedding I	1.41 ms	0.382 ms	6.46 ms	1.352 ms	13.59 ms	2.396 ms
T-Block Embedding II	4.27 ms	0.488 ms	19.05 ms	1.841 ms	38.43 ms	3.613 ms

of the gesture trials of each subject (following [19], repetitions 1, 3, 4, and 6 repetitions were used for training purposes), and $\mathcal{D}_{meta-test}$ consisted of the remaining repetitions. Table I shows our results when using few-shot classification as well as previous works which used supervised learning. From Table I, it can be observed that the proposed FS-HGR architecture outperformed existing methodologies when evaluated based on the same setting, i.e., 89.70% best accuracy with the FS-HGR compared to 83.70% best accuracy achieved by the state-of-the-art.

Reference [19], used different classifiers such as K-Nearest Neighbors (K-NN), SVM, RF, and LDA. The average classification accuracy obtained using all the classical methods on the DB2 dataset is $60.28 \pm 6.51\%$. They show that the highest average classification accuracy is $75.27 \pm 7.89\%$, obtained with RF. Reference [21] showed that the average accuracy of SVM on all movement types is 77.44%. Finally, in Reference [6], the best accuracy is reported with RF classifier, which is $72.25 \pm 7.13\%$. It can be seen from Table I that DNN-based methods provide improved performance.

The average of training and testing times for one Task $\mathcal{T}_j$ using 1-shot, 5-shot, and 10-shot for Experiment 1 are summarized in Table II. It is noteworthy to say that the time of processing depends on the hardware. In this work, we used a "NVIDIA's GeForce GTX 1080 Ti Graphic Cards".

Experiments 2: Classification on New-Subject With Few-Shot Observation: In this scenario, like the previous experiment, the second Ninapro database DB2 was utilized. It consists of 49 gestures plus rest from 40 intact-limb subjects.

TABLE III

EXPERIMENT 2(a): 5-WAY AND 10-WAY, 1-SHOT AND 5-SHOT CLASSIFICATION ACCURACIES BASED ON *new subjects with few-shot observation*. IN THIS EXPERIMENT, WE ADOPTED FOUR DIFFERENT EMBEDDING MODULES: (i) FC EMBEDDING; (ii) LSTM EMBEDDING; (iii) T-BLOCK EMBEDDING I, AND; (iv) T-BLOCK EMBEDDING II

The Embedding Module	**5-way Accuracy (%) ± STD**		**10-way Accuracy (%) ± STD**	
	1-shot	**5-shot**	**1-shot**	**5-shot**
FC Embedding	62.87 ± 0.13	78.90 ± 0.15	43.47 ± 0.15	68.59 ± 0.14
LSTM Embedding	64.46 ± 0.16	79.82 ± 0.14	49.58 ± 0.17	69.93 ± 0.19
T-Block Embedding I	**67.81** ± 0.14	81.08 ± 0.14	50.31 ± 0.14	69.94 ± 0.18
T-Block Embedding II	66.98 ± 0.15	**81.29** ± 0.17	**52.05** ± 0.15	**70.71** ± 0.18

TABLE IV

COMPARISON OF 5-WAY, 1-SHOT AND 5-SHOT CLASSIFICATION ACCURACIES BETWEEN THE EXPERIMENT 2(a) AND 2(b) BASED ON *new subjects with few-shot observation*

The Embedding Module	**Experiment 2(a)**		**Experiment 2(b)**	
	5-way Accuracy (%) ± STD			
	1-shot	**5-shot**	**1-shot**	**5-shot**
FC Embedding	62.87 ± 0.13	78.90 ± 0.15	72.69 ± 0.15	86.08 ± 0.14
LSTM Embedding	64.46 ± 0.16	79.82 ± 0.14	75.56 ± 0.17	89.14 ± 0.12
T-Block Embedding I	**67.81** ± 0.14	81.08 ± 0.14	75.11 ± 0.14	89.66 ± 0.13
T-Block Embedding II	66.98 ± 0.15	**81.29** ± 0.17	**77.08** ± 0.15	**90.47** ± 0.12

In this experiment, to validate our claim that the proposed FS-HGR architecture can classify hand gestures of new subjects just by training with a few observations, we split the DB2 database into $\mathcal{D}_{meta-train}$, $\mathcal{D}_{meta-val}$, and $\mathcal{D}_{meta-test}$ such that the subjects in these meta-sets are completely different (i.e., there is no overlap between the meta-sets). In other words, $\mathcal{D}_{meta-train}$ consists of the first 27 subjects, while $\mathcal{D}_{meta-val}$ includes the sEMG signals from the 28^{th} subject to 32^{ed} subject (5 subjects). Finally, we evaluated our model on the remaining subjects, i.e., $\mathcal{D}_{meta-test}$ consists of the final 8 subjects in the DB2 database.

It is noteworthy to mention that the proposed network is trained once and shared across all participants (which is different from previous works that trained the model separately for each participant). For constructing task $\mathcal{T}_j$, however, we can feed data in two different approaches:

- *Experiment 2(a):* In the first approach, for constructing $\mathcal{D}_j^{support}$ for each task $\mathcal{T}_j$, we selected all of the N classes from a specific user, which was randomly selected from the existing participants. This is the more realistic and practical scenario.
- *Experiment 2(b):* In the second approach, for constructing $\mathcal{D}_j^{support}$, N classes were selected from different participants.

Table III shows few-shot classification accuracies for Experiment 2(a) based on four different embedding modules. The adaptive learning method of the proposed FS-HGR focuses on transfer learning information between a source and a target domain despite the existence of a distribution mismatch between $\mathcal{D}_{meta-train}$ and $\mathcal{D}_{meta-test}$. The results reported in Table III show that the proposed mechanism achieves acceptable results despite the fact that the sEMG signals are user-dependent. Table IV shows a comparison of 5-way classification accuracies between Experiments 2(a) and 2(b). As was it expected, Experiment 2(b) achieved better results, which is due to the presence of variations among the probability distribution of sEMG signals obtained from different subjects.

TABLE V

5-WAY, 1-SHOT AND 5-SHOT CLASSIFICATION ACCURACIES BASED ON *new subjects with few-shot observation* (EXPERIMENT 2(a)). IN THIS EXPERIMENT, WE OBTAINED THE ACCURACY FOR EACH SUBJECT IN THE TEST SET USING T-BLOCK EMBEDDING II. THE WILCOXON SIGNED RANK TEST IS APPLIED TO COMPARE THE DIFFERENT SHOTS (e.g., 1-SHOT AND 5-SHOT). NULL HYPOTHESIS IS REJECTED WHEN $H_0 = 0$ ($p < 0.05$)

The Subject	5-way Accuracy (%)	
	1-shot	5-shot
Subject 33	71.94	85.62
Subject 34	73.48	85.49
Subject 35	64.84	77.68
Subject 36	64.58	78.53
Subject 37	64.25	82.52
Subject 38	63.75	77.81
Subject 39	61.82	76.50
Subject 40	71.29	86.30
$H_0 = 0$ ($p < 0.05$)	0 (0.01172)	-

TABLE VI

EXPERIMENT 3: 5-WAY, 1-SHOT, 5-SHOT, AND 10-SHOT CLASSIFICATION ACCURACIES BASED ON *new gesture with few-shot observation*

The Embedding Module	5-way Accuracy (%) ± STD		
	1-shot	5-shot	10-shot
FC Embedding	45.94 ± 0.16	67.20 ± 0.14	79.87 ± 0.15
LSTM Embedding	46.05 ± 0.16	71.76 ± 0.14	81.58 ± 0.16
T-Block Embedding I	**49.78** ± 0.15	71.57 ± 0.13	83.41 ± 0.14
T-Block Embedding II	45.48 ± 0.17	**73.36** ± 0.17	**83.97** ± 0.17

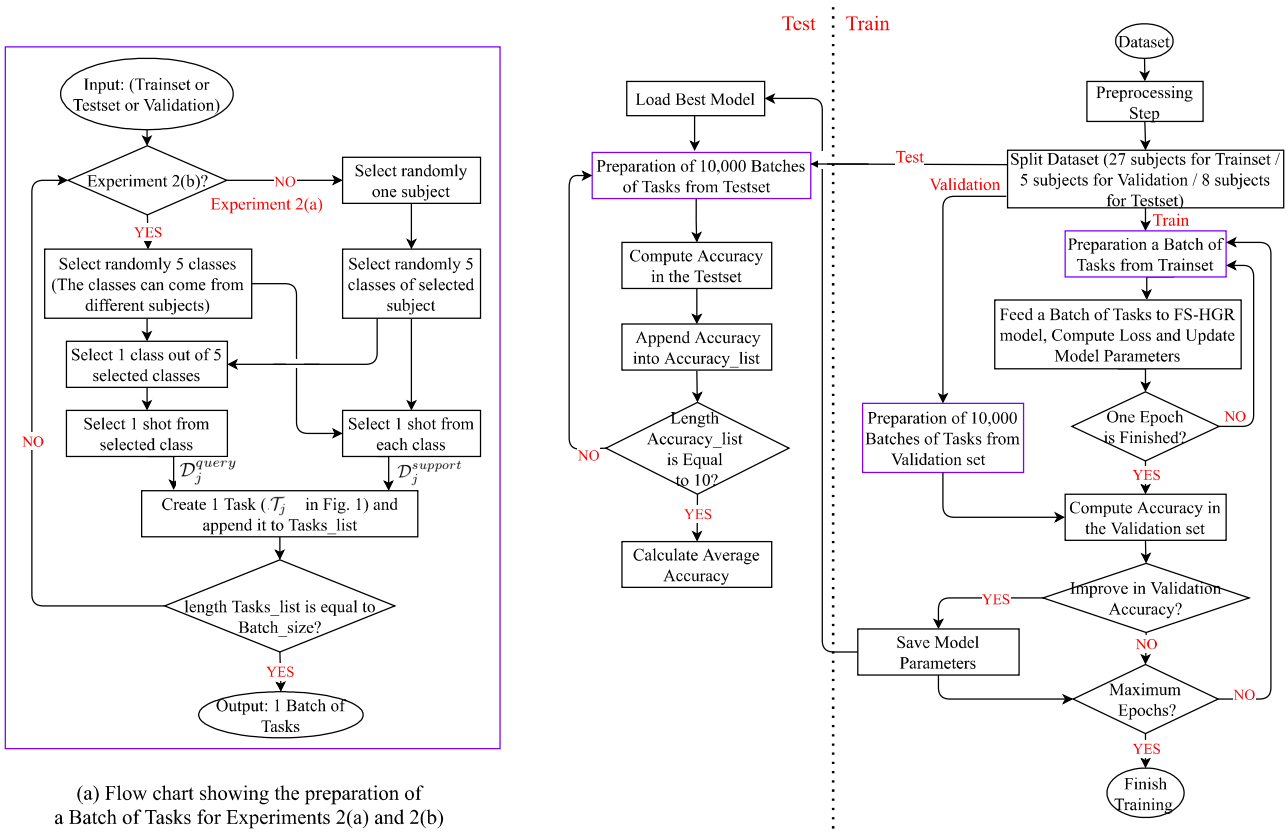
However, this is not a practical setting as in practice all of the N classes in $\mathcal{D}_j^{support}$ comes from the same user (i.e., Experiment 2(a)).

It is worth mentioning that Experiment 2(a) is the more realistic and challenging one, while Experiment 2(b) is included for completeness and comparison purposes. The rational behind Experiment 2(a) is that a prosthesis hand will be utilized by a user, therefore, the model should be able to distinguish different hand movements of this specific person. In this sense, Experiment 2(a) is more realistic than Experiment 2(b). In Experiment 2(a), for constructing $\mathcal{D}_j^{support}$, we selected all the N classes from a specific user, which was randomly selected from the participants. However, in Experiment 2(b), for constructing $\mathcal{D}_j^{support}$, N classes were selected from different participants, which implies that observations were obtained from different distributions/people, so the model would more easily discriminate these observations. For Experiment 2(a), we have conducted a statistical hypothesis test to evaluate if there is significant evidence to reject the hypothesis that lower and higher shots have similar accuracies. We followed [29] and used the Wilcoxon signed-rank test [38] considering each participant as a separate dataset. Table V compares accuracy for each subject in the test set in Experiment 2(a) for 5-way 1-shot and 5-way 5-shot. The difference in accuracy between 1 and 5 shots was considered statistically significant by the Wilcoxon signed rank test as the ($p < 0.05$).

Experiment 3: Classification on New-Gestures With Few-Shot Observations: In this scenario, the goal is evaluating the capability of the proposed FS-HGR architecture when the target consists of solely out-of-sample gestures (i.e., new gestures with few-shot observation). Performing well in this task allows the model to evaluate new observations, exactly one per novel hand gesture class. In this experiment, the Ninapro database DB2 was used. The DB2 dataset includes three sets of exercises denoted by Exercise B, C, and D. Exercise B includes 8 isometric and isotonic hand configurations and 9 basic movements of the wrist; Exercise C consists of 23 grasping and functional movements; and finally, Exercise D consists of 9 force patterns. For training purposes, $\mathcal{D}_{meta-train}$ consisted of the first 34 gestures of each user, which is equal to approximately 68% of the total gestures. $\mathcal{D}_{meta-val}$ included 6 gestures or 12% of the total gestures. The remaining gestures (9 gestures), were used in $\mathcal{D}_{meta-test}$ for evaluation purposes. Exercises B and C were, therefore, used for training and validation, and Exercises D, with different gestures, were used for test purposes. Table VI shows the efficiency of the proposed model when we had out-of-sample gestures in the target. The model predicted unknown class distributions in scenarios where few observations from the target distribution were available.

VI. CONCLUSION

We proposed a novel few-shot learning recognition approach for the task of hand gesture recognition via sEMG signals. The proposed FS-HGR framework could quickly generalize after seeing very few observations from each class. This is achieved by exploiting the knowledge gathered from previous experiences to accelerate the learning process performed by a new subject. The experience gained over several source subjects is leveraged to reduce the training time of a new target user. In this way the learning process does not start every time from the beginning, and instead refines. The ability to learn quickly based on a few observations is a key characteristic of the proposed FS-HGR framework that distinguishes this novel architecture from its previous counterparts. A second contribution of the paper is its capability to address the user-dependent nature of the sEMG signals. The proposed FS-HGR framework transfers information between a source and a target domain despite the existence of a distribution mismatch among them. This would dramatically reduce the number of required cumbersome training sessions leading to a drastic reduction in functional prosthesis abandonment. In the paper, we have shown that for a new user/gesture when the distributions of sEMG signals is different from that of the training data, the model can still classify the hand movements. This is because the knowledge gained during the training phase is leveraged for the new users/gestures. Other factors such as time variability between days, and type of amputations affect the distributions of sEMG signals. These factors were not investigated in the proposed architecture while are relevant to study in future research. A third factor that can affect distributions of sEMG signals is misplacement or displacement of sensors (electrode locations/shift), which is an open topic of research that has not been addressed in this paper. This can be applied to any existing research focusing on the processing of sEMG. We believe that the proposed approach has the potential to also address this problem, however, we have not completed our experiments and thus cannot strongly mention



(a) Flow chart showing the preparation of a Batch of Tasks for Experiments 2(a) and 2(b)

(b) Flow chart for training, validation, and test steps for Experiment 2

Fig. 5. (a) Flowchart for preparation of a batch of tasks for Experiment 2. (b) Flowchart for training, validation, and test steps of Experiment 2. Each purple box in (b) represents one or more repetitions of (a).

that. We consider this as a limitation for our current study and a future research direction. As a final note, we would like to mention that multi-channel EMG recording has become a common trend and there are commercialized wearable sensors. It is correct that the higher number of sensors results in higher complexity of electronics, but thanks to recent advances in the area of wearable sensors, this has been achieved and is progressing.

APPENDIX

A. Flowchart of the FS-HGR Framework

We have included a flowchart (Fig. 5) for the proposed method applied to Experiment 2. Experiments 1 and 3 are similar in nature to Experiment 2. Fig. 5(a) shows the preparation of a batch of tasks for Experiment 2(a) and 2(b). Fig. 5(b) shows the training, validation, and test steps.

B. Comparison with State-of-Art TL-based Model

Moreover, we have compared the proposed FS-HGR framework with the TL technique of Reference [29] for cross-user scenarios. To provide a fair comparison, we have used the same public dataset utilized in [29], i.e., Ninapro DB5. The Ninapro DB5 [6] was recorded with the Myo Armband, and contains data from 10 healthy participants performing a total of 53 movements (rest included) divided into three exercise sets. The performance of their proposed TL-based architecture is investigated based on the second exercise set of DB5, which contains 18 (rest included) number of gestures. More specifically, in their proposed approach, the pre-training for each participant was performed by employing the training sets of the remaining nine participants. Finally, the average accuracy over the 10 participants was reported. Furthermore, the data is first separated by applying sliding windows of 52 samples (260 ms) with an overlap of 235 ms. We followed the same criteria and obtained the average accuracy over the 10 participants for cross-user model (Experiment 2(a)) for 5-way 5-shot and 5-way 10-shot classifications. In addition to the results of Reference [29], we have included results of cross-user models based on classical hand-crafted features developed in References [6], [23], [39]–[41], coupled with traditional classifiers (RF, and LDA). Results are reported in Table VII. It is observed that the proposed FS-HGR method outperforms classical and state-of-the-art TL-based models over this cross-user scenario.

C. Discussions on the Training Time of the FS-HGR Framework

Finally, it is worth noting that adoption of few-shot learning within the FS-HGR framework has resulted in reduction in the required training time for users. In previous studies, such as [17]–[22], for each user's gesture, 4 repetitions, each one lasting 5 seconds, were required for calibrating (fine-tuning)

TABLE VII
COMPARISON OF EXPERIMENT 2(A) ON NINAPRODB5 DATABASE BETWEEN CLASSICAL METHODS [6], [23], [39]–[41], DNN METHODS [29], AND OUR PROPOSED FEW-SHOT LEARNING APPROACH (FS-HGR)

Duration	**Our Proposed Few-shot Learning Method**	
	T-Block Embedding I	T-Block Embedding II
5-way 5-shot (1.3 seconds)	61.63 ± 5.09	64.65 ± 4.88
5-way 10-shot (2.6 seconds)	65.71 ± 4.79	**67.69 ± 4.58**

Duration	**Classical Methods**				**DNN Methods**	
	[39] (RF)	[40] (LDA)	[6], [23] (LDA)	[41] (RF/LDA)	[29]	[29] + TL
10 seconds	50.85 ± 4.29	50.08 ± 4.63	46.85 ± 4.81	53.00 ± 3.85 (RF)	55.65 ± 4.38	60.12 ± 4.79
5 seconds	40.70 ± 5.84	40.86 ± 6.91	37.60 ± 6.67	42.26 ± 5.78 (LDA)	46.06 ± 6.09	49.41 ± 5.82

TABLE VIII
AVERAGE PREDICTION ACCURACY BY THE NETWORK FOR EACH CLASS FOR 5-WAY, 1-SHOT, 5-SHOT, AND 10-SHOT CLASSIFICATION ACCURACIES ON *new repetitions with few-shot observation*. IN THIS EXPERIMENT, WE ADOPTED T-BLOCK EMBEDDING II

	5-way k-shot Accuracy ± STD %										
Movement	**1-shot**	**5-shot**	**10-shot**	**Movement**	**1-shot**	**5-shot**	**10-shot**	**Movement**	**1-shot**	**5-shot**	**10-shot**
1	85.81 ± 0.17	93.04 ± 0.23	95.59 ± 0.26	18	67.30 ± 0.37	77.80 ± 0.45	85.05 ± 0.34	35	64.80 ± 0.44	81.15 ± 0.32	84.73 ± 0.34
2	83.15 ± 0.20	90.99 ± 0.33	94.61 ± 0.40	19	71.91 ± 0.32	80.78 ± 0.44	86.71 ± 0.37	36	69.13 ± 0.51	84.40 ± 0.44	87.83 ± 0.54
3	85.43 ± 0.17	91.59 ± 0.22	93.86 ± 0.17	20	68.74 ± 0.34	80.44 ± 0.46	85.55 ± 0.36	37	69.27 ± 0.23	85.37 ± 0.54	88.03 ± 0.36
4	79.84 ± 0.30	89.79 ± 0.56	92.63 ± 0.23	21	68.60 ± 0.29	81.46 ± 0.32	86.65 ± 0.38	38	64.11 ± 0.51	81.42 ± 0.40	86.11 ± 0.37
5	75.56 ± 0.20	86.31 ± 0.46	91.03 ± 0.35	22	68.20 ± 0.34	78.90 ± 0.62	84.73 ± 0.39	39	76.72 ± 0.22	89.06 ± 0.37	90.18 ± 0.25
6	80.26 ± 0.33	88.60 ± 0.40	92.96 ± 0.35	23	61.34 ± 0.26	74.96 ± 0.34	81.84 ± 0.45	40	76.35 ± 0.31	88.28 ± 0.32	90.66 ± 0.27
7	81.90 ± 0.22	89.96 ± 0.27	92.47 ± 0.29	24	66.87 ± 0.31	79.99 ± 0.34	84.63 ± 0.28	41	85.58 ± 0.28	94.95 ± 0.14	96.32 ± 0.21
8	79.35 ± 0.27	89.28 ± 0.23	92.41 ± 0.29	25	67.06 ± 0.49	81.08 ± 0.47	87.34 ± 0.32	42	86.35 ± 0.31	94.89 ± 0.31	95.79 ± 0.17
9	73.27 ± 0.38	83.83 ± 0.41	88.87 ± 0.17	26	65.36 ± 0.51	78.96 ± 0.71	84.24 ± 0.33	43	86.47 ± 0.18	95.11 ± 0.25	95.66 ± 0.27
10	69.24 ± 0.28	80.71 ± 0.33	88.19 ± 0.29	27	63.34 ± 0.38	77.53 ± 0.37	83.42 ± 0.31	44	89.12 ± 0.33	96.27 ± 0.15	96.99 ± 0.17
11	63.16 ± 0.42	75.48 ± 0.28	83.37 ± 0.39	28	64.82 ± 0.51	78.46 ± 0.44	83.73 ± 0.27	45	88.32 ± 0.28	95.18 ± 0.20	96.97 ± 0.26
12	71.62 ± 0.32	84.35 ± 0.36	89.23 ± 0.37	29	63.20 ± 0.60	76.74 ± 0.57	81.61 ± 0.40	46	86.28 ± 0.24	93.20 ± 0.26	96.18 ± 0.22
13	80.10 ± 0.33	90.46 ± 0.29	93.09 ± 0.30	30	64.28 ± 0.47	78.63 ± 0.63	83.83 ± 0.35	47	89.76 ± 0.16	94.11 ± 0.18	96.65 ± 0.13
14	78.00 ± 0.33	88.76 ± 0.45	91.99 ± 0.21	31	70.89 ± 0.43	83.26 ± 0.57	86.78 ± 0.33	48	91.89 ± 0.18	95.17 ± 0.28	97.01 ± 0.27
15	78.26 ± 0.28	87.87 ± 0.26	92.10 ± 0.25	32	73.32 ± 0.40	85.88 ± 0.29	87.97 ± 0.26	49	90.64 ± 0.22	93.50 ± 0.26	97.59 ± 0.13
16	75.32 ± 0.40	85.38 ± 0.50	88.12 ± 0.41	33	63.61 ± 0.51	79.58 ± 0.61	83.15 ± 0.61				
17	76.72 ± 0.25	88.90 ± 0.23	91.35 ± 0.26	34	68.91 ± 0.30	83.07 ± 0.44	87.71 ± 0.48				

the model for a new user. Therefore, for a dataset consisting of 49 gestures, $49\times4\times5 = 980$ seconds of data must be collected to calibrate (fine-tune) the model for a new user. On the other hand, traditional hand-crafted methods, such as the Canonical Correlation Analysis (CCA)-based approach proposed in [42], require 3-to-5 seconds of EMG data per movement class (determined empirically) for calibration. While this calibration set is much smaller than the training set previously used [17]–[22], it is developed for a smaller number of subjects. In the proposed FS-HGR approach, however, the model is tuned to a new user by seeing a small number of observations (each of duration 200 ms). For a new user in a N-way k-shot classification problem, we just need to see k-shot from each gesture. Each shot is a window of size 200 ms. For example, in 5-way 1-shot scenario, the duration of the required data from a new user for calibration (fine-tuning) is 49×200 ms $= 9.8$ seconds (which is $980/9.8 = 100$ times less than in previous methods). It is worth noting that increasing the number of shots (while improving the accuracy) increases the required number of training observations. For Instance, in a 5-way 5-shot problem, the 5- shots (windows of length 200 ms) come from one repetition, i.e., for each new user only one repetition from each class, lasting 1 second, is required. In other words, we do not need to collect data for the same number of repetitions as in previous works. Therefore, the total duration of the required data from a new user would be $49 \times 5 \times 200$ ms $= 49$ seconds, which is still much lower than conventional deep learning-based approaches. In a practical setting, the number of utilized shots can be adjusted based on the required level of accuracy and training time, which provides flexibility for practical use.

D. The Average Prediction Accuracy for Each Class

Moreover, Table VIII shows the average prediction accuracy by the network for each class for 5-way, 1-shot, 5-shot, and 10-shot classification accuracies on *new repetitions with few-shot observation* for T-Block Embedding II.

REFERENCES

[1] N. Jiang, S. Dosen, K. R. Muller, and D. Farina, "Myoelectric control of artificial limbs—Is there a need to change focus," *IEEE Signal Process. Mag.*, vol. 29, no. 5, pp. 150–152, May 2012.

[2] D. Farina, R. Merletti, and R. M. Enoka, "The extraction of neural strategies from the surface EMG," *J. Appl. Physiol.*, vol. 96, pp. 1486–1495, Oct. 2004.

[3] D. Esposito *et al.*, "A piezoresistive array armband with reduced number of sensors for hand gesture recognition," *Frontiers Neurorobotics*, vol. 13, p. 114, Jan. 2020.

[4] M. Tavakoli, C. Benussi, P. A. Lopes, L. B. Osorio, and A. T. de Almeida, "Robust hand gesture recognition with a double channel surface EMG wearable armband and SVM classifier," *Biomed. Signal Process. Control*, vol. 46, pp. 121–130, Sep. 2018.

[5] G. R. Naik, A. H. Al-Timemy, and H. T. Nguyen, "Transradial amputee gesture classification using an optimal number of sEMG sensors: An approach using ICA clustering," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 24, no. 8, p. 837–846, Oct. 2015.

[6] S. Pizzolato, L. Tagliapietra, M. Cognolato, M. Reggiani, H. Müller, and M. Atzori, "Comparison of six electromyography acquisition setups on hand movement classification tasks," *PLoS ONE*, vol. 12, no. 10, pp. 1–7, 2017.

[7] C. Castellini *et al.*, "Proceedings of the first workshop on peripheral machine interfaces: Going beyond traditional surface electromyography," *Frontiers Neurorobotics*, vol. 8, p. 22, Aug. 2014.

[8] D. Farina *et al.*, "The extraction of neural information from the surface EMG for the control of upper-limb prostheses: Emerging avenues and challenges," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 4, pp. 797–809, Jul. 2014.

[9] A. K. Clarke, "Deep learning for robust decomposition of high-density surface EMG signals," *IEEE Trans. Biomed. Eng.*, vol. 38, no. 2, pp. 526–534, Feb. 2020.

[10] E. Rahimian, S. Zabihi, S. F. Atashzar, A. Asif, and A. Mohammadi, "Surface EMG-based hand gesture recognition via hybrid and dilated deep neural network architectures for neurorobotic prostheses," *J. Med. Robot. Res.*, vol. 23, no. 5, 2020, Art. no. 2041001.

[11] E. Rahimian, S. Zabihi, S. F. Atashzar, A. Asif, and A. Mohammadi, "XceptionTime: Independent time-window xceptiontime architecture for hand gesture classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 1304–1308.

[12] E. Rahimian, S. Zabihi, S. F. Atashzar, A. Asif, and A. Mohammadi, "Semg-based hand gesture recognition via dilated convolutional neural networks," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Nov. 2019, pp. 1–5.

[13] D. Josephs, C. Drake, A. Heroy, and J. Santerre, "SEMG gesture recognition with a simple model of attention," 2020, *arXiv:2006.03645*. [Online]. Available: http://arxiv.org/abs/2006.03645

[14] L. Chen, J. Fu, Y. Wu, H. Li, and B. Zheng, "Hand gesture recognition using compact CNN via surface electromyography signals," *Sensors*, vol. 20, no. 3, p. 672, 2020.

[15] Y. Peng, H. Tao, W. Li, H. Yuan, and T. Li, "Dynamic gesture recognition based on feature fusion network and variant convLSTM," *IET Image Process.*, vol. 14, no. 11, pp. 2480–2486, Sep. 2020.

[16] W. Wei, Y. Wong, Y. Du, Y. Hu, M. Kankanhalli, and W. Geng, "A multi-stream convolutional neural network for sEMG-based gesture recognition in muscle-computer interface," *Pattern Recognit. Lett.*, vol. 119, pp. 131–138, Mar. 2019.

[17] W. Wei, Q. Dai, Y. Wong, Y. Hu, M. Kankanhalli, and W. Geng, "Surface-electromyography-based gesture recognition by multi-view deep learning," *IEEE Trans. Biomed. Eng.*, vol. 66, no. 10, pp. 2964–2973, Oct. 2019.

[18] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, "A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition," *PLoS ONE*, vol. 13, no. 10, Oct. 2018, Art. no. e0206049.

[19] M. Atzori, M. Cognolato, and H. Müller, "Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands," *Frontiers Neurorobot.*, vol. 10, p. 9, Sep. 2016.

[20] Z. Ding, C. Yang, Z. Tian, C. Yi, Y. Fu, and F. Jiang, "SEMG-based gesture recognition with convolution neural networks," *Sustainability*, vol. 10, no. 6, p. 1865, Jun. 2018.

[21] X. Zhai, B. Jelfs, R. H. M. Chan, and C. Tin, "Self-recalibrating surface EMG pattern recognition for neuroprosthesis control based on convolutional neural network," *Frontiers Neurosci.*, vol. 11, p. 379, Jul. 2017.

[22] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, and J. Li, "Gesture recognition by instantaneous surface EMG images," *Sci. Rep.*, vol. 6, no. 1, Dec. 2016, Art. no. 36571.

[23] M. Atzori *et al.*, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Sci. Data*, vol. 1, no. 1, pp. 1–13, Dec. 2014.

[24] A. Gijsberts, M. Atzori, C. Castellini, H. Muller, and B. Caputo, "Movement error rate for evaluation of machine learning methods for sEMG-based hand movement classification," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 4, pp. 735–744, Jul. 2014.

[25] M. Atzori *et al.*, "Characterization of a benchmark database for myoelectric movement classification," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, no. 1, pp. 73–83, Jan. 2015.

[26] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," 2017, *arXiv:1707.03141*. [Online]. Available: http://arxiv.org/abs/1707.03141

[27] A. Srinivasan, A. Bharadwaj, M. Sathyan, and S. Natarajan, "Optimization of image embeddings for few shot learning," in *Proc. 10th Int. Conf. Pattern Recognit. Appl. Methods*, 2021, pp. 1–4.

[28] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 100, 2017, pp. 1126–1135.

[29] U. Cātālard *et al.*, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 27, no. 4, pp. 760–771, Oct. 2019.

[30] R. Chattopadhyay, N. C. Krishnan, and S. Panchanathan, "Topology preserving domain adaptation for addressing subject based variability in SEMG signal," in *Proc. AAAI Spring Symp., Comput. Physiol.*, 2011, p. 4–9.

[31] U. Cote-Allard, C. L. Fall, A. Campeau-Lecours, C. Gosselin, F. Laviolette, and B. Gosselin, "Transfer learning for sEMG hand gestures recognition using convolutional neural networks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, p. 1663.

[32] M. Zia ur Rehman *et al.*, "Multiday EMG-based classification of hand motions with deep learning techniques," *Sensors*, vol. 18, no. 8, p. 2497, 2018.

[33] Y. Du, W. Jin, W. Wei, Y. Hu, and W. Geng, "Surface EMG-based inter-session gesture recognition enhanced by deep domain adaptation," *Sensors*, vol. 17, no. 3, p. 458, Feb. 2017.

[34] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE Trans. Biomed. Eng.*, vol. 40, no. 1, pp. 82–94, 1993.

[35] S. Bai, J. Zico Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*. [Online]. Available: http://arxiv.org/abs/1803.01271

[36] A. van den Oord *et al.*, "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*. [Online]. Available: http://arxiv.org/abs/1609.03499

[37] A. Vaswani *et al.*, "Attention is all you need," 2017, *arXiv:1706.03762*. [Online]. Available: http://arxiv.org/abs/1706.03762

[38] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, Dec. 1945.

[39] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multifunction myoelectric control," *IEEE Trans. Biomed. Eng.*, vol. 50, no. 7, p. 848–854, Jun. 2003.

[40] R. N. Khushaba and S. Kodagoda, "Electromyogram (EMG) feature reduction using mutual components analysis for multifunction prosthetic fingers control," in *Proc. 12th Int. Conf. Control Autom. Robot. Vis. (ICARCV)*, Dec. 2012, pp. 1534–1539.

[41] A. Phinyomark *et al.*, "EMG feature evaluation for improving myoelectric pattern recognition robustness," *Expert Syst. Appl.*, vol. 40, no. 12, p. 4832–4840, 2013.

[42] R. N. Khushaba, "Correlation analysis of electromyogram signals for multiuser myoelectric interfaces," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 4, pp. 745–755, Jul. 2014.