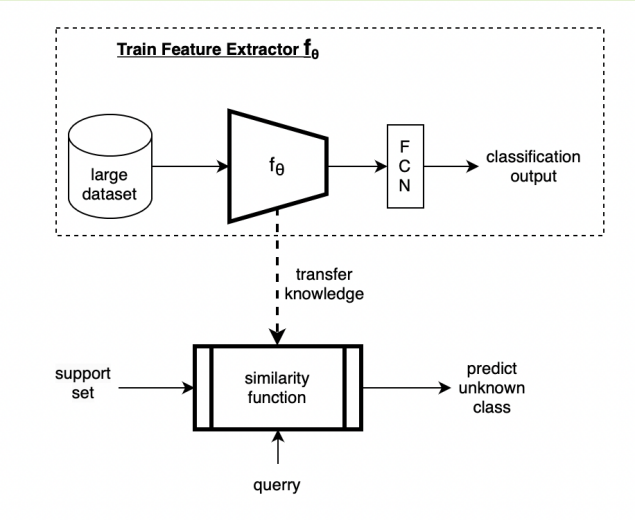
sEMG Motion Classification Via Few-Shot Learning With Applications To Sports Science

Mert Ergeneci, Erkan Bayram, Ayberk Yarkın Yıldız, Daryl Carter, and Panagiotis Kosmas, *Member, IEEE*

***Abstract*— Motion classification with surface electromyography (sEMG) has been studied for practical applications in prosthesis limb control and human-machine interaction. Recent studies have shown that feature learning with deep neural networks (DNN) reaches considerable accuracy in motion classification tasks. However, DNNs require large datasets for acceptable performance and fail for tasks with few data samples available for training. Professional athlete training includes hundreds of exercises, and coupled with privacy and confidentiality issues acquiring a large dataset for all the exercises is not feasible. As a result, state-of-the-art DNN architectures are unsuitable for real-life sports applications. We utilise few-shot learning (FSL) techniques to overcome the small dataset problem of sports-related motion classification tasks. The employed methodology uses the knowledge gathered from a large set of tasks to classify unseen tasks with a few data samples. The FSL approach with a siamese network and triplet loss reached the best performance with a median F1-score of 72.01%, 76%, and 79% for 1, 5 and 10 shot datasets that include an unseen set of tasks, respectively. In contrast, DNN with transfer learning (TF) reached 49.27%, 51.58%, and 67.66% for the same set of tasks, respectively.**



***Index Terms*— Surface Electromyography; Sports Science; Deep Neural Networks; Motion Classification; Meta Learning; Few-shot Learning**

I. Introduction

In professional sports muscle injury creates a burden on financials and a barrier to sporting success [1], [2]. To mitigate the risks of injury, the trainers should observe the physiological changes of the athletes regularly [3]. Surface electromyography (sEMG), a non-invasive medical technique for measuring the electrical activity on muscle tissue, can be utilised to monitor such physiological shifts of athletes during training [4]. The sEMG signal maintains information required to detect the movements and monitor the muscle-related physiological risk factors [5]–[8]. The movement type determines the contraction technique (i.e., isotonic or isometric), the muscle activation intensity, and the muscles to be recruited. To this end, studies have shown that the estimation of injury risk factors should be calculated separately for each movement type [3], [9], [10].

Motion classification via sEMG signal is a supervised machine learning (ML) task for determining the movements realised by individuals [11]. The literature on sEMG-based motion classification mostly focuses on feature engineering to represent sEMG signals in a discriminative way [12]. Feature engineering utilises signal processing algorithms, which can be based on time-domain, frequency-domain or time-frequency-domain, to extract a feature vector [11]. Recent studies show that deep neural networks (DNN) could be used to learn efficient feature vector representations of sEMG data among various motion classes [13], [14]. Some publications further reveal that DNNs achieve superior performance compared to ML methods based on predefined feature engineering models [13]–[16]. Recent studies have also used domain adaptation techniques via transfer learning (TL) or meta-learning to enhance DNN performance against sEMG data variability [18]. sEMG data is open to variability due to electrode shift, sensor positioning, muscular fatigue, and human physiological diversity [19]–[23]. The variability of the sEMG signal can potentially create unseen data samples to a trained model, which may distort the extracted features and result in a drop in classification accuracy [24]–[26]. In [15], the authors adopt

M. E. is with School of Natural and Mathematical Sciences, King's College London, Strand, London, WC2R 2LS, UK (e-mail: mert.1.ergeneci@kcl.ac.uk).

E. B. is with Neurocess Limited, 7 Bell Yard, Strand, London, United Kingdom, WC2A 2JR, UK (e-mail: erkan@neurocess.co).

A. Y. Y. is with Neurocess Limited, 7 Bell Yard, Strand, London, United Kingdom, WC2A 2JR, UK (e-mail: yarkin@neurocess.co).

D. C. is with Leeds United Football Club Elland Road, Leeds, LS11 0ES, UK (e-mail: daryl.carter@leedsunited.com).

P. K. is with School of Natural and Mathematical Sciences, King's College London, Strand, London, WC2R 2LS (e-mail: panagiotis.kosmas@kcl.ac.uk).

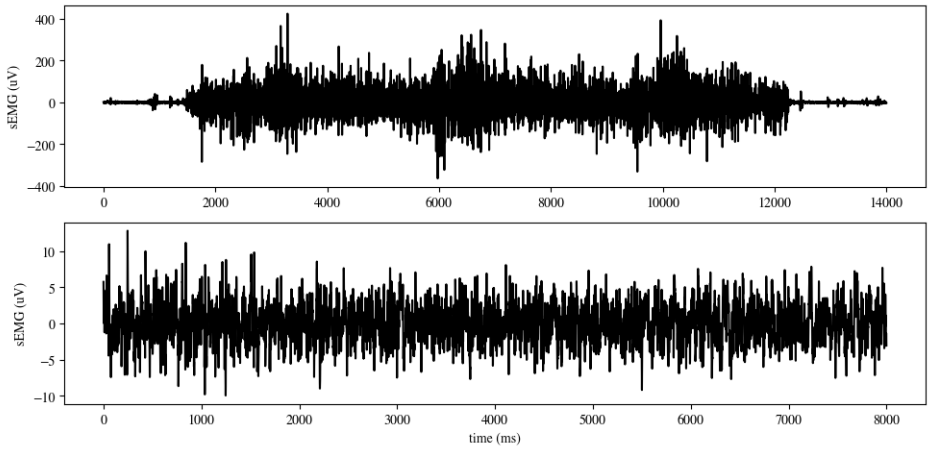


Fig. 1. The top plot displays the sEMG recording of the biceps femoris muscle during single leg elevated glute bridge exercise, which is a lower extremity activity recorded from a professional athlete. The bottom plot shows the sEMG recording of the hand gesture collected from biceps brachii during a hand squeeze. The data of the bottom plot is acquired from a publicly available dataset called NinaPro [17]. The lower extremity recording (top) includes transient sEMG amplitude, while the hand gesture is steady.

a TL strategy for a convolutional neural network (CNN) to generalise the hand gesture data of unseen subjects using the knowledge of available datasets. Works [27], [28] employ a domain adaptation method to improve deep learning performance for the same motion labels recorded at different sessions, i.e., inter-session. The inter-session data of known labels may vary between different recordings of the same person due to changes in sensor positioning and fatigue [27], [28]. Finally, a few-shot learning (FSL) technique is used by [14] to adapt both intra-session (i.e., different repetitions of a movement within the same recording) variability and subject dependency of the sEMG data. All these studies commonly work with the Ninapro dataset, a publicly available dataset for hand gestures.

In sports applications, there are hundreds of different exercises, i.e., motion classes, and obtaining a large dataset for each class is not feasible. However, gathering a small dataset (e.g., 5 or 10 instances) from numerous motion classes is achievable. In this case, the standard supervised ML models fail because of the small dataset size, which limits the predictive power of DNNs [11]. This work addresses this conceptual and technological gap by proposing a DNN employing a FSL solution that leverages generalisation across tasks, i.e., motion classes, rather than across observations in one dataset. The contributions of this paper can be listed as follows:

- A few-shot meta-learning is employed for the first time to classify unseen labels of motion classes, given a few samples of a training dataset, i.e., 1, 5, 10, and 20 observations, respectively.
- A real-life application is tested in a sports setting with the first-team players of a professional football club (Leeds United Football Club).
- An FSL approach is tested on a dataset gathered from lower extremity muscle groups and sports exercises for the first time. The lower extremity motion types used in this work take longer than hand gestures and require long-term correlations of the sEMG signal. As seen in Fig. 1, most hand gestures include steady-state, i.e., isometric, contractions with a very slight transient-state at the beginning [11]. On the other hand, the dataset used in this paper contains a continuous dynamic sEMG signal, transitioning between eccentric and concentric contractions [29]–[31].

The paper is organised as follows. We comprehensively describe the dataset preparation and methodology in Section II. Then, experiments and data collection are detailed in Section III. The results are evaluated in Section IV. Discussion and conclusion on the results are provided in Section V.

II. Methodology

A. Dataset

In this work, we obtain the dataset $\mathcal{D}$ from sEMG recordings of 15 Leeds United Football Club (LUFC) male first-team athletes while performing 7 different motion types. The Neurocess dry, active, bipolar sEMG system is used for recording data [32]. $\mathcal{D}$ includes a total of $4,193$ set of pairs (x^i, y^i), where y^i is the true label, x^i is the recorded sEMG signal, and i is the index of the pair. The recorded sEMG vector x^i includes a two-channel sEMG signal of length $N = 8000$, approximating 8 seconds per channel as the sampling frequency is $f_s = 1000$ Hz. More details for the experiment protocol and dataset are documented in Section III.

B. Preprocessing

Data preprocessing involves three steps: segmentation, sample entropy (SampEn), and downsampling.

In segmentation, we split up the sEMG signal in each data instance $x^i[n]$ into frames of length $N = 128$ with an overlap $L = 96$, which correspond to a total of 248 frames for each sEMG channel. The frame and overlap size is selected according to the best-performing parameters in the test in Section III. We define the resulting column vector, x^i_f, as

$$x^i_f = [x^i[(f-1)(N-L)+1]\ldots, x^i[(f-1)(N-L)+N]]^T, \quad (1)$$

where f represent the index of frames.

After the segmentation, we calculate the sample entropy (SampEn) of each frame and then sequentially merge each result as a vector. We use the SampEn transformation as a measure of complexity, because its proven to be an effective sEMG signal representation for amplifying muscle activation and suppressing noise. For the SampEn step, we first seperate each frame x_f^i into sub-frames of size s using a sliding window technique such that the jth sub-frame is given as

$$X_{(f,s)}^i[j] = \left[x_f^i[j+k]\right]_{k=0}^{s-1}. \tag{2}$$

Then, we define the two parameters N^{s+1} and N^s as

- $N_{(f,s+1)}^i$ is the number of sub-frames of length $s+1$ in frame x_f^i having $d[X_{(f,s+1)}^i[l], X_{(f,s+1)}^i[j]] < r$,
- $N_{(f,s)}^i$ is the number of sub-frames of length s in frame x_f^i having $d[X_{(f,s)}^i[l], X_{(f,s)}^i[j]] < r$,

where $s = 2$, r is the 0.25 times the standard deviation of x_f^i, and d is the Chebyshev distance [33]. We determine the parameters r and s according to the work [4].The output SampEn vector of the ith observation $S^i[n]$ is finally calculated as

$$S^i[f] = \left[-\log\left(\frac{N_{(f,s+1)}^i}{N_{(f,s)}^i}\right)\right]_{f=0}^{M-1}, \tag{3}$$

where the total frame size $M = 248$.

A larger value of SampEn indicates that there are fewer similar sequences of length $s+1$ than those of length s, which suggests a higher level of randomness and unpredictability [33]. We apply the SampEn transformation to each frame x_f^i and combine the outputs sequentially to get $S^i[f]$.

In the final pre-processing stage, we downsample the SampEn vector, $S^i[f]$. To reduce the high-frequency components of the signal, we use the Savitzky–Golay filter with window size $W_n = 51$ and the polynomial order $P_o = 3$. The filter is a finite-impulse response low-pass filter based on local least-squares polynomial approximation [34]. Let C be the coefficient matrix for the local least square estimation such that

$$C = (J^\top J)^{-1} J^T, \tag{4}$$

where J is Vandermonde matrix of size $51x4$. The output of the Savitzky-Golay filter, $\tilde{S}^i$, is calculated as

$$\tilde{S}^i[j] = (C * S^i)_j = \sum_{k=-\frac{P_o-1}{2}}^{\frac{P_o-1}{2}} C_k S^i[j+k], \tag{5}$$

such that

$$\frac{P_o+1}{2} \leq j \leq |S| - \frac{P_o-1}{2}. \tag{6}$$

Next, we decimate the filtered signal $\tilde{S}^i$ by 4 as

$$\bar{x}^i[k] = \left[\tilde{S}^i[4(k-1)+1]\right]_{k=1}^{\frac{M}{4}}, \tag{7}$$

where $\bar{x}^i$ is the decimated signal.

At the end of pre-processing, we obtain $\bar{x}^i$ of size $2x61$ from the recorded sEMG signal of size $2x8000$. In the rest of the paper, we use the pre-processed dataset $\bar{\mathcal{D}}$, which include pairs of $(\bar{x}^i, y^i)$.

C. Base Deep Learning Model

In both meta-learning and TL applications, there is a need for a deep learning (DL) structure that extracts sEMG features efficiently in the given task set. This section details the DL structure used as the base model in the rest of the paper.

The tasks in our dataset include sEMG signal with sequential patterns and temporal dependencies, and we choose bidirectional long short-term memory (BiLSTM) recurrent neural network (RNN) to learn those temporal patterns [4], [35]. Conventional LSTM RNNs are used in sequence classification due to their ability to extract temporal attributes [36]–[38]. BiLSTM RNNs process data in both directions of time with two hidden layers without bringing any implementation complexity to the model [39].

As seen in Table I, the overall model consists of two cascaded BiLSTM-RNN layers, two fully connected layers, and an output layer. The BiLSTM RNN layers and the fully connected layers extract a feature vector representing the discriminative features of the sEMG sequence. The final output layer is the classifier that predicts the label, $\tilde{y}^i$, of any given input $\bar{x}^i$. To tune the model, we test varying numbers of hidden layers on a test dataset. The model with the highest median F-1 score is adopted as the base model. The details are provided in Section III.

Let $\mathcal{C}$ be a subset of $\bar{\mathcal{C}}$ in $\bar{\mathcal{D}}$. Then, the DL model f_ϕ with trainable parameters ϕ can be represented as

$$f_\phi : \mathbb{R}^{2x61} \mapsto \mathcal{C}. \tag{8}$$

In conventional supervised learning, the optimum trainable parameters ϕ^* is calculated as

$$\phi^* = \arg\min_\phi \mathcal{L}(\bar{\mathcal{D}}; \phi, \omega), \tag{9}$$

where $\mathcal{L}$ is the binary cross entropy loss function, ω is the Adam optimizer.

TABLE I
THE DETAILS OF THE DEEP LEARNING MODEL ARCHITECTURE WITH TYPE OF LAYER, LAYER OUTPUT VECTOR SHAPE, AND TOTAL NUMBER OF TRAINABLE PARAMETERS.

Layers	Type	Output Shape
0-1	Input (2)*	(61,2)
1-2	BiLSTM (2)*	(61, 256)
2-3	BiLSTM (1)*	(61,128)
3-4	BiLSTM (1)*	(128)
4-5	FC (1)*	64
5-6	FC (1)*	32
6-7	FC (1)**	$\lvert\mathcal{C}\rvert$

∗ *The number inside the parenthesis next to the layer type represent the number of parallel layers.*
∗∗ *The output shape of the model is selected as the number of unique labels in the training set, denoted by* $|\mathcal{C}|$.

D. Transfer Learning (TL)

In this work, we aim to adopt TL to use the acquired knowledge from a set of tasks $\mathcal{C}^1$ such that $\mathcal{C}^1 \subseteq \bar{\mathcal{C}}$ over a set of unseen tasks $\mathcal{C}^2$ such that $\mathcal{C}^2 \subseteq \bar{\mathcal{C}}$ and $\mathcal{C}^2 \cap \mathcal{C}^1 = \emptyset$. TL allows us to use the same model structure and generalised knowledge for different tasks by only changing the classifier

layer. Assume $f_{\phi^*}(\cdot)$ is the trained model for the set $\mathcal{C}^1$ and $\bar{f}_{\phi^*}(\cdot)$ is the feature extractor, i.e., same model with classifier layer removed, such that

$$\bar{f}_{\phi^*} : \mathbb{R}^{2x61} \mapsto \mathbb{R}^{32}, \tag{10}$$

where the output vector $\mathbb{R}^{32}$ is the feature embedding. To utilise the trained feature extractor $\bar{f}_{\phi^*}(\cdot)$ of the set $\mathcal{C}^1$ to the set $\mathcal{C}^2$, we add a classifier layer that has the same output shape with the size of $\mathcal{C}^2$ as

$$f^{\mathcal{C}^2}_{\phi^*} : \mathbb{R}^{2x61} \mapsto \mathcal{C}^2. \tag{11}$$

In (11), the optimized parameters for the task $\mathcal{C}^1$ is transferred to the feature extractor of $f^{\mathcal{C}^2}_{\phi^*}(\cdot)$. For a better performance, the model parameters, ϕ^* can be retrained and calibrated for the task C^2 as

$$\phi^{**} = \arg\min_{\phi \in \Phi^*} \mathcal{L}(\bar{\mathcal{D}}; \phi, \omega), \tag{12}$$

where ϕ^{**} is the retrained model parameters and Φ^* is the subspace for optimization which is restricted by the transferred model. This study employs two TL techniques, fine-tuning and adaptive batch normalisation, for the retraining procedure in (21). Each of these techniques restricts the optimization space in different ways which are clear in the content.

1) Fine-Tuning: In fine-tuning, some layers of the feature extractor $\bar{f}_{\phi^*}(\cdot)$ in (10) are kept frozen, i.e., untrainable, and some other layers are updated in accordance with Equation (21). In a DNN, the first layers detect simpler and more general patterns, while the deeper layers learn more complicated and specific dataset features [40], [41]. In this work, freezing the first BiLSTM RNN layer and keeping the other layers trainable reach the best performance for fine-tuning, which is detailed in Section III.

2) Adaptive Batch Normalisation (AdaBatch): To apply AdaBatch, $f_{\phi}(\cdot)$ is trained for the task $\mathcal{C}^1$ with batch normalization (BN) technique as in [42]. Then, all the layers except the BN layers of $\bar{f}_{\phi^*}(\cdot)$ are frozen prior to retraining for the task $\mathcal{C}^2$. The hypothesis behind this method suggests that the domain-related information is stored in BN layers, and updating the BN weights for the new task enhances domain adaptation [15].

E. Few-Shot Learning

This paper employs metric-based FSL, formulated as N-way, K-shot. FSL is part of a meta-learning framework such that a base learner solves any given task, and a meta-learner learns the aggregated across-task knowledge from existing tasks to classify unseen labels with a small dataset [43].

In metric-based FSL, the model predicts the label of any given query, i.e., $\tilde{x}$, by comparing the similarity of the query with the other classes in the support set $\mathcal{S}^{sup}_c$, which is defined as

$$\mathcal{S}^{sup}_c := [(\bar{x}_1, y_1), \cdots, (\bar{x}_K, y_K)], \tag{13}$$

where c is a class in the set $\mathcal{C}^2$, which is the set of unseen tasks. For the sake of simplicity, we denote it as $\mathcal{C}$. The pairs in (13) are selected randomly from the set $\{(\bar{x}, y) \in \mathcal{D}^{train} | y \in \mathcal{C}\}$ with replacement. N denotes the number of distinct c, and K represents the number of pairs in $\mathcal{S}^{sup}_c$. A query sample $\tilde{x}$ is randomly selected from the set $\{(\tilde{x}, y) \in \mathcal{D}^{test} | y \in \mathcal{C}\}$. It is associated with support set $\mathcal{S}^{sup}_c$. This relation is called an episode. To make a prediction, the model makes a pairwise comparison of the query $\tilde{x}$ with all the samples of the support set in an episode and selects the c with the highest similarity or lowest distance among the classes in $\mathcal{C}$.

In this paper, we adopt metric-based FSL. Thus all the following methods are functions to estimate the distance or similarity between the query-support pair. We use the base model feature extractor $\bar{f}_{\phi^*}(\cdot)$ in (10) to extract the feature embedding.

1) L^p Norm: Let $CL^p(\cdot)$ be a classification function that maps a query sample $\tilde{x}$ to a task in the set $\mathcal{C}^2$ based on a support set S^{sup}. We define $CL^p(\cdot)$ as

$$CL^p(\tilde{x}) := \arg\min_{c \in \mathcal{C}^2} \frac{1}{|S^{sup}_c|} \sum_{(\bar{x}_i, y_i) \in S^{sup}_c} \|\bar{f}_{\phi^*}(\bar{x}_i) - \bar{f}_{\phi^*}(\tilde{x})\|_p. \tag{14}$$

The L^p norm, i.e., $\|\cdot\|$, of a sequence $x[n]$ is calculated as

$$\|x\|_p = (\sum_{i=1}^{i=n} |x[i]|^p)^{\frac{1}{p}} \tag{15}$$

In this work, we use $CL^p(\cdot)$ for three different p values: $p = 1$, i.e., Manhattan Distance, $p = 2$ i.e., Euclidean Distance, and $p = 20$.

2) Matching Network with Fixed Attention: First, we recall the cosine similarity $cos(\cdot)$ of two sequence $x_a[n]$ and $x_b[n]$ as

$$cos(x_a, x_b) = \frac{\langle x_a, x_b \rangle}{\|x_a\|_2 \|x_b\|_2} = \frac{\sum_{k=1}^{n} x_a[k] x_b[k]}{\|x_a\|_2 \|x_b\|_2} \tag{16}$$

Next, we calculate the cosine similarity between the feature embedding of query $\tilde{x}^i$ and the pairs of support set $\mathcal{S}^{sup}_c$ as

$$\hat{x}_c = \frac{1}{|S^{sup}_c|} \sum_{(\bar{x}_i, y_i) \in S^{sup}_c} cos(\bar{f}_{\phi^*}(\bar{x}_i), \bar{f}_{\phi^*}(\tilde{x})) \tag{17}$$

Let $CM(\cdot)$ be a classification function that maps a query sample $\tilde{x}$ to $\mathcal{C}^2$ based on a support set S^{sup} by utilising the softmax function for majority voting.

$$CM(\tilde{x}) = \arg\max_{c \in \mathcal{C}^2} \sigma(-\hat{x}_c) = \arg\max_{c \in \mathcal{C}^2} \frac{\exp(-\hat{x}_c)}{\sum_{\tilde{c} \in \mathcal{C}} \exp(-\hat{x}_{\tilde{c}})}. \tag{18}$$

3) Prototype Network with L^2 Norm: The average embedded vector $\hat{x}_c$ for each class in the support set is calculated as

$$\hat{x}_c := \frac{1}{|S^{sup}_c|} \sum_{(\bar{x}_i, y_i) \in S^{sup}_c} \bar{f}_{\phi^*}(\bar{x}_i). \tag{19}$$

Then the classification function $CP(\cdot)$ is defined as

$$CP(\tilde{x}) := \arg\max_{c \in \mathcal{C}^2} \sigma(-\|\hat{x}_c - \bar{f}_{\phi^*}(\tilde{x})\|_2). \tag{20}$$

4) Siamese Network: In this approach, we employ a deep-learning architecture to determine if both samples in a pair belong to the same class. As seen in Table II, the overall model consists of three fundamental parts: a siamese (twin) network architecture with shared weights, a lambda layer

which calculates the absolute difference of two feature vectors, two fully connected layers, and one output layer which returns a similarity score. The twin network consists of a duplicate of the base DL model $\bar{f}_{\phi*}$ in Section II. We apply TL to use the acquired knowledge of the other set of tasks. We use two different loss functions: contrastive loss and triplet loss. Both loss functions aim to minimise the distance metric for the same class samples while maximising the distance for distinct ones.

The training batch for contrastive loss consists of $(\bar{x}_i, x^-, 0)$ and $(\bar{x}_i, x^+, 1)$ where x^+ denotes a sample with the same label with $\bar{x}_i$, and x^- denotes a sample with a different label from $\bar{x}_i$. Note that any pair in the training batch is randomly selected from the samples of the tasks in $\mathcal{C}^1$.

The training batch for triplet loss consists of $(\bar{x}_i, x^-, x^+)$. Therefore, the contrastive loss and triplet loss are denoted by $\mathcal{L}^2$ and $\mathcal{L}^3$, respectively. Then, the siamese model $\mathcal{SL}^p_\phi$ with trainable parameters ϕ and the selected loss $\mathcal{L}^p$ can be represented as:

$$\mathcal{SL}^p_\phi : \mathbb{R}^{2\times 2\times 32} \mapsto [0, 1]$$

For better performance, the model parameters, ϕ, can be retrained and calibrated for the similarity of a given pair,

$$\phi^{**} = \arg\min_{\phi\in\Phi^*} \mathcal{L}^p(\bar{\mathcal{D}}; \phi, \omega), \tag{21}$$

where ϕ^{**} is the retrained model parameters, $\mathcal{L}^p$ is the selected loss, and Φ^* is the subspace for optimisation which the transferred model restricts. It is clear that both the transferred model $\bar{f}_{\phi^*}$ and the siamese model $\mathcal{SL}_{\phi^{**}}$ encounters only the samples labelled as one of the tasks in the set $\mathcal{C}^1$. Let $CS(.)$ be a classification function that maps a query sample $\tilde{x}$ to the set $\mathcal{C}^2$ based on a support set S^{sup}. It is defined for the selected loss $\mathcal{L}^p$ as:

$$CS^p(\tilde{x}, S^{sup}) = \arg\max_{c\in\mathcal{C}^2} \frac{1}{|S^{sup}_c|} \sum_{(\bar{x}_i, y_i)\in S^{sup}_c} \mathcal{SL}^p_{\phi^{**}}(\tilde{x}, \bar{x}_i) \tag{22}$$

TABLE II
THE DETAILS OF THE SIAMESE NETWORK ARCHITECTURE WITH TYPE OF LAYER AND LAYER OUTPUT VECTOR SHAPE

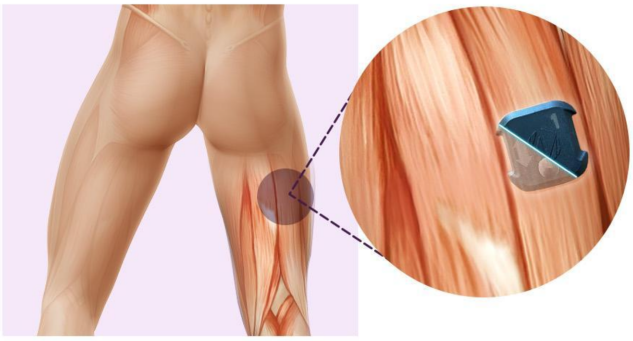
Layers	Type	Output Shape
0-1	Input (2)*	(61,2)
1-6	f_ϕ (2)**	(32,1)
6-7	Absolute Difference (1)	32
7-8	FC (1)*	32
8-9	FC (1)*	16
9-10	FC (1)	1

∗ The number inside the parenthesis next to the layer type represent the number of parallel layers.
∗∗ It is the first 6 layers in the transferred model $\bar{f}_\phi$ (see Table I), used as feature embedding function.

III. EXPERIMENTS

The sEMG data was recorded from 15 Leeds United Football Club (LUFC) male first-team athletes via the Neurocess system, which consists of wireless, dry, active, bipolar sEMG sensors [32]. During the experiments, two sEMG sensors were positioned on each subject's left and right biceps femoris long head muscles. The SENIAM recommendations suggest that the electrodes need to be placed at 50% on the line between the ischial tuberosity and the lateral epicondyle of the tibia [44]. For the replication of the experiment process, we provide an illustration of the electrode placement in Fig. 2. The

Fig. 2. The illustration of electrode placement location and orientation for the biceps femoris long head muscle group.



experiment protocols and sensor positioning were supervised and maintained by the first-team physiotherapist of LUFC. We completed all the sEMG recordings in the LUFC training ground under ethical clearance guidelines granted by King's College London, referencing MRSP-20/21-20996. We did not keep any personally identifiable info during the experiments and anonymised all the documented data.

The athletes were requested to realise the following hamstring drills:

- ($C1$) 60° adductor squeeze: The athlete lies in a supine position with 60° hip flexion and squeezes the examiner's fist, which is located between the athlete's knees.
- ($C2$) Glute bridge: The athlete lies down in a supine position, knees in full flexion and feet flat on the floor and close to the hip. Then the athlete lifts his hip off the floor as high as possible.
- ($C3$) single leg (SL) elevated long-lever bridge: The athlete lies down in a supine position, with his feet flat on a box of 45cm height and with knee flexion angle 45°. Then, the athlete lifts his hip off as high as possible using only one leg.
- ($C4$) Isometric prone contraction: The athlete lies down prone with his knee flexed at 45°. The examiner locates his hands on the athlete's ankle giving full resistance, and the athlete tries to contract against that resistance.
- ($C5$) Elevated glute bridge: The athlete lies down in a supine position, knees in 90° flexion and feet flat on a 45cm box. Then the athlete lifts his hip off the floor as high as possible.
- ($C6$) SL prone curls: The athlete lies down prone with his knee at 0°. Then, he curls his knee up to 90° and back to 0° with his hip at a stable position.
- ($C7$) Running: The athlete runs on a treadmill with 3 kph speed.

The dataset $\bar{\mathcal{D}}$ is split into training ($\bar{\mathcal{D}}^{train}$), validation ($\bar{\mathcal{D}}^{val}$), and test ($\bar{\mathcal{D}}^{test}$) sets. The training set is used to train the base model $f_\phi(\cdot)$, the validation set is used to prevent overfitting during the training, and the test set is used to test and compare all the methods. In Table III, the training,

validation, and test set sizes are provided.

TABLE III
MOTION TYPES AND CORRESPONDING SAMPLE SIZE OF TRAINING, VALIDATION, AND TEST SETS

Motion type	Training Set	Validation Set	Test Set
C1	389	111	120
C2	487	111	120
C3	540	153	120
C4	301	76	120
C5	244	62	120
C6	318	65	120
C7	403	93	120
total	**2682**	**671**	**840**

To simulate a real-life setting for the motion classification, we create multiple tasks using the classes in $\bar{\mathcal{D}}$. The set of tasks $\bar{\mathcal{C}}$ includes all the 7 motion classes in $\bar{\mathcal{D}}$, and it is used to test the performance of the base DL model. The set of tasks $\mathcal{C}^1$ consists of the classes $C1$ to $C4$ and The set of tasks $\mathcal{C}^2$ consists of the classes $C5$ to $C7$ in $\bar{\mathcal{D}}$. In this study, we repeated all the experiments 10 times with different random seeds for splitting the dataset and determining initial model weights. To compare the performance of the methods, we employed the F1-score. Since each method was tested with 10 iterations of different random seeds, we used the median and the interquartile range (IQR) to summarise the results. The IQR is the difference between the 3^{rd} quartile and 1^{st} quartile of a distribution, contributing to the range of the central half of the distribution. We picked median and IQR over mean and standard deviation to represent the results since median and IQR are more robust when the distribution includes outliers. The high IQR represent that the data is more spread out, while the small IQR value means that the data points are more crowded around the median. For our case, the small IQR with high median values of F1-scores are preferred.

A. Selection of Model Parameters and Structure

We first selected the optimum number of BiLSTM layers for the DL model, f_ϕ, by testing the model performance with various numbers of layers for the set of tasks $\bar{\mathcal{C}}$.

Then, we determined the best-performing parameters for the pre-processing step by testing the ideal DL model f_ϕ with various values of frame size, overlap, and downsampling rate for the set of tasks $\bar{\mathcal{C}}$.

To pick the ideal number of frozen layers for the TL technique, described in II-D.1., we first trained the base model f_ϕ for the set of tasks $\mathcal{C}^1$ and then used the dataset of the set of tasks $\mathcal{C}^2$ with a various number of frozen layers.

B. New Task with Few Samples

To simulate a real-life sports setting, we aim to classify a set of tasks, $\mathcal{C}^2$, with a few samples using the knowledge aggregated from another set of tasks $\mathcal{C}^1$.

Let ν be the sample per class in the training set of the model DL classfier. Precisely, the training set in this experiment consists of $|\mathcal{C}^2|\nu$ sample. We ran this experiment by following ν values 1, 5, 10, 20, and 50. To compare the performance of the proposed methodology, we used the following tests.

1) Base DL model: To see how the DL model performs with a small dataset, we trained the model with different numbers of samples for $\mathcal{C}^2$ without using any data of $\mathcal{C}^1$.

2) TL: We first trained the base DL model with data of $\mathcal{C}^1$. Then, we applied TL techniques using a few samples of $\mathcal{C}^2$.

3) Metric-Based Few-Shot Learning: Same as III-B.2, we train the base model with $\mathcal{C}^1$ and then apply the FSL techniques in II-E using $\mathcal{C}^2$ as $\mathcal{S}_c^{sup}$.

IV. RESULTS

As detailed in Section III-A, the first part of the experiments includes the selection of ideal parameters and model structure.

We tested various hidden BiLSTM layers for the first validation step to select the ideal number of layers for a set of tasks $\bar{\mathcal{C}}$. As seen in Table IV, a total number of 3 BiLSTM layers result in the best performance with 79.26% median F1-score. Consequently, we employed the base DL model structure with 3 cascaded BiLSTM layers.

TABLE IV
MEDIAN F1 SCORE AND IQR FOR BASE DL MODEL TESTED FOR A SET OF TASKS WITH **1**, **2**, **3**, **4** NUMBER OF BILSTM LAYERS RESPECTIVELY

	1 Layer	2 Layer	3 Layer	4 Layer
F1 Score	78.08	74.94	79.26	74.69
IQR	2.11	2.78	4.99	4.38

TABLE V
MEDIAN F1 SCORE AND IQR OF THE BASE DL MODEL f_ϕ FOR EACH CLASS IN THE SET OF TASKS $\mathcal{C}^2$

	Precision	IQR	Recall	IQR	F1	IQR
C1	91,81	2,65	90,00	5,55	90,32	2,96
C2	97,72	2,60	89,16	3,26	93,85	1,84
C3	92,24	2,89	96,66	3,08	95,55	2,45
C4	89,51	3,64	93,44	1,40	90,98	2,45
Av	92,70		92,31		92,68	

TABLE VI
MEDIAN F1 SCORE AND IQR FOR DIFFERENT NUMBER OF FROZEN LAYERS IN TL FINE-TUNING FOR THE SET OF TASKS $\mathcal{C}^2$

	1 Layer	1-2 Layer	1-2-3 Layer
Median F1 Score	93.04	89.50	87.64
IQR	2.24	0.45	3.01

TABLE VII
MEDIAN F1 SCORE AND IQR FOR DEEP-LEARNING VS SAMPLE PER CLASS

Sample Per Class ν	$\nu = 1$	$\nu = 5$	$\nu = 10$	$\nu = 20$	$\nu = 50$
Median F1 Score	18.53	41.01	59.54	57.47	68.66
IQR	2.27	20.58	8.69	5.44	12.04

After optimising the ideal number of hidden layers of the base DL model, we selected the best-performing parameters for the pre-processing step in Section II-B. As seen in Fig. 3, the frame size $N = 128$, overlap $L = 96$, and downsampling rate of 4 resulted in the highest weighted F1-score of 83%.

The last test of the Section III-A includes the choice of the ideal number of frozen layers in the TL fine-tuning. First,

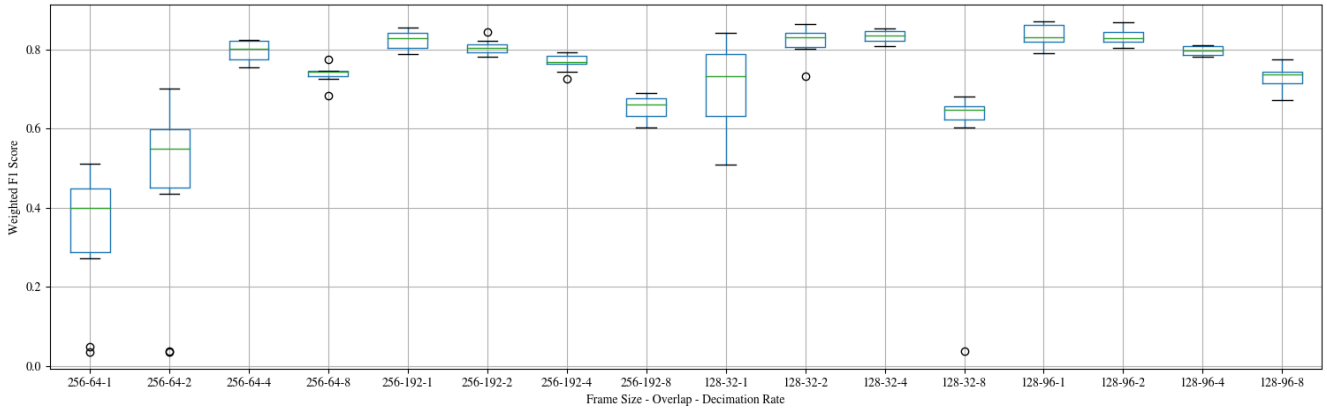


Fig. 3. The plot of weighted F1-scores of various pre-processing parameters, which are frame size, overlap, and downsampling rate respectively.

TABLE VIII
MEDIAN F1 SCORE AND IQR IN TRANSFER LEARNING VS SAMPLE PER CLASS

Sample Per Class ν	$\nu = 1$	$\nu = 5$	$\nu = 10$	$\nu = 20$	$\nu = 50$
Median F1 Score	49.27	51.58	67.66	63.67	72.97
IQR	11.79	9.71	9.45	5.11	7.53

TABLE IX
MEDIAN F1 SCORE AND IQR FOR DIFFERENT FSL ALGORITHMS

		k=1	**k=5**	**k=10**	**k=20**	**k=50**
CL^1	**F1**	63.73	73.7	71.68	74.1	72.56
	IQR	5.23	2.85	3.56	4.41	6.33
CL^2	**F1**	64.02	73.24	72.42	73.94	72.32
	IQR	3.36	1.74	3.34	4.0	6.06
CL^{20}	**F1**	62.3	72.45	72.09	74.69	74.26
	IQR	4.37	1.41	2.0	2.53	1.88
CM	**F1**	65.01	71.63	71.53	73.31	71.25
	IQR	3.04	0.7	4.63	4.11	7.49
CP	**F1**	65.01	72.29	73.96	76.31	74.92
	IQR	3.04	2.97	2.75	2.07	3.38
CS^2	**F1**	70.14	72.57	75.32	74.74	**76.44**
	IQR	1.14	1.52	0.72	0.98	0.65
CS^3	**F1**	65.41	75.55	76.74	78.10	73.90
	IQR	1.86	1.11	3.00	3.30	0.38
CS^3_{AB}	**F1**	**72.01**	**76.32**	**79.16**	**78.80**	75.37
	IQR	1.10	0.69	0.55	0.41	0.70

we trained the base DL model f_ϕ for the set of tasks $\mathcal{C}^1$. The test results of the trained base DL model f_{ϕ^*} can be seen in Table V. We used TL fine-tuning with various frozen layers, and as seen in Table VI, the first layer frozen resulted in the finest performance for the unseen set of tasks $\mathcal{C}^2$. In the second part of the experiments, we compared the performance of the methods described in Section II. In Table VII, the results for the experiment B.1 are provided. The base deep learning model fails to distinguish labels when few data is provided for the training. As seen in Table VIII, The TL approach, detailed in the experiment B.2, reaches a superior performance compared to the DL model without TL. The TL technique improves the median F1 score from 18.53% to 49.27% for $\nu = 1$ dataset, and from 41.01% to 67.66% for $\nu = 5$ dataset.

The results of the proposed FSL methods, as detailed in the experiment B.3, are provided in Table VI. All the FSL based approaches outperform both TL and DL techniques. The siamese network with AdaBatch and triplet loss, CS^3_{AB}, has the best performance, where it reaches median F1-score of 72.01% for 1-shot, 76% for 5-shot, and 79% for 10-shot. The FSL methods have lower F1-score when the dataset gets larger than 10-shot due to the variance bias trade off.

V. DISCUSSION & CONCLUSION

sEMG-based motion classification with DNNs has been investigated mostly for hand-gesture recognition [45], [46]. The existing techniques, including feature engineering and feature learning, are optimised for large datasets aggregated under controlled lab settings. However, gathering large data samples per label in real-life applications is unfeasible. To this end, we utilised several FSL-based approaches to improve the DNN performance for unseen tasks with few training data samples. For the first time, this study applies a few-shot learning approach for 7 hamstring activities used in sports training of 15 professional soccer players. The meta-learning approach with siamese network and triplet loss reached the best performance with a median F1-score of 72.01%, 76%, and 79% for 1, 5 and 10 shot datasets, respectively. We further tested DNN with and without TF as reference methods to compare the meta-learning performance on the same set of tasks. The DNN without TF reached 18.53%, 41.01%, and 59.54% and DNN with TF reached 49.27%, 51.58%, and 67.66% for 1, 5 and 10 shot datasets, respectively. As expected, training a DNN solely with a small dataset resulted in the lowest median F1-score with a high IQR, which means that the model performance depends highly on the training dataset and initial model weights. The TF improved performance of DNN by using the retained information from the previous set of tasks to adapt to the new set of tasks with a small dataset. We tested 8 different FSL techniques, and all the FSL approaches outperformed the reference methods, which are DNN with and without TF. It can be inferred that FSL is effective in sEMG-based motion classification with a small dataset.

The FSL resulted in the best performance given a new set of tasks with a small dataset. The model performs a classification

F1-score above 70% for 1-shot tests, which indicates it can successfully classify an unseen class of motion even if a single sample of that label is provided for training. However, the proposed model must first be trained with a relatively large dataset to classify a new set of tasks with a small dataset. This study used a set of tasks, $\mathcal{C}^1$, to train the model, and then the model can classify any new set of tasks $\mathcal{C}^2$ with a small dataset. The initial training is required to optimise the base deep learning model, which outputs feature vectors for similarity or distance tests in the meta-learning method. After the initial training, the meta-learning model can classify new tasks without training or large dataset requirements. However, the proposed methodology cannot be used in certain cases where no preliminary dataset, such as $\mathcal{C}^1$, is available. In future work, we will attempt to train the base deep learning model of the proposed meta-learning approach with the publicly available NinaPro database, which includes a large dataset of hand gestures [17]. When trained with a NinaPro dataset, suppose the meta-learning model can classify few-shot lower extremity motion classes with similar performance, i.e., median F1-score higher than 70%. In that case, our model becomes generic and reproducible for all sports-related motion classification applications.

REFERENCES

[1] Martin Hägglund, Markus Waldén, Henrik Magnusson, Karolina Kristenson, Håkan Bengtsson, and Jan Ekstrand. Injuries affect team performance negatively in professional football: an 11-year follow-up of the UEFA champions league injury study. *British Journal of Sports Medicine*, 47(12):738–742, May 2013.

[2] https://www.premierinjuries.com/betting.php?Date=all$_2$019.

[3] Brady Green, Matthew N Bourne, Nicol van Dyk, and Tania Pizzari. Recalibrating the risk of hamstring strain injury (HSI): A 2020 systematic review and meta-analysis of risk factors for index and recurrent hamstring strain injury in sport. *British Journal of Sports Medicine*, pages bjsports–2019–100983, April 2020.

[4] Mert Ergeneci, Daryl Carter, and Panagiotis Kosmas. semg onset detection via bidirectional recurrent neural networks with applications to sports science. *IEEE Sensors Journal*, 22(19):18751–18761, 2022.

[5] Tahereh Kamali, Reza Boostani, and Hossein Parsaei. A multi-classifier approach to muap classification for diagnosis of neuromuscular disorders. *IEEE transactions on neural systems and rehabilitation engineering*, 22(1):191–200, 2013.

[6] Sumitra S Nair, Robert M French, Davy Laroche, and Elizabeth Thomas. The application of machine learning algorithms to the analysis of electromyographic patterns from arthritic patients. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 18(2):174–184, 2009.

[7] Barbro Larsson, Stefan Karlsson, Magnus Eriksson, and Björn Gerdle. Test–retest reliability of emg and peak torque during repetitive maximum concentric knee extensions. *Journal of Electromyography and Kinesiology*, 13(3):281–287, 2003.

[8] Kaan Gokcesu, Mert Ergeneci, Erhan Ertan, Abdallah Zaid Alkilani, and Panagiotis Kosmas. An semg-based method to adaptively reject the effect of contraction on spectral analysis for fatigue tracking. In *Proceedings of the 2018 ACM International Symposium on Wearable Computers*, pages 80–87, 2018.

[9] K. Bennell, H. Wajswelner, P. Lew, A. Schall-Riaucour, S. Leslie, D. Plant, and J. Cirone. Isokinetic strength testing does not predict hamstring injury in australian rules footballers. *British Journal of Sports Medicine*, 32(4):309–314, December 1998.

[10] Matthew Buckthorpe, Steve Wright, Stewart Bruce-Low, Gianni Nanni, Thomas Sturdy, Aleksander Stephan Gross, Laura Bowen, Bill Styles, Stefano Della Villa, Michael Davison, and Mo Gimpel. Recommendations for hamstring injury prevention in elite football: translating research into practice. *British Journal of Sports Medicine*, 53(7):449–456, November 2018.

[11] Andrés Jaramillo-Yánez, Marco E Benalcázar, and Elisa Mena-Maldonado. Real-time hand gesture recognition using surface electromyography and machine learning: A systematic literature review. *Sensors*, 20(9):2467, 2020.

[12] Dezhen Xiong, Daohui Zhang, Xingang Zhao, and Yiwen Zhao. Deep learning for emg-based human-machine interaction: A review. *IEEE/-CAA Journal of Automatica Sinica*, 8(3):512–533, 2021.

[13] Dario Farina, Ning Jiang, Hubertus Rehbaum, Aleš Holobar, Bernhard Graimann, Hans Dietl, and Oskar C Aszmann. The extraction of neural information from the surface emg for the control of upper-limb prostheses: emerging avenues and challenges. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(4):797–809, 2014.

[14] Elahe Rahimian, Soheil Zabihi, Amir Asif, Dario Farina, Seyed Farokh Atashzar, and Arash Mohammadi. Fs-hgr: Few-shot learning for hand gesture recognition via electromyography. *IEEE transactions on neural systems and rehabilitation engineering*, 29:1004–1015, 2021.

[15] 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.

[16] Stefan Karlsson and Björn Gerdle. Mean frequency and signal amplitude of the surface emg of the quadriceps muscles increase with increasing torque—a study using the continuous wavelet transform. *Journal of electromyography and kinesiology*, 11(2):131–140, 2001.

[17] Manfredo Atzori, Arjan Gijsberts, Ilja Kuzborskij, Simone Elsig, Anne-Gabrielle Mittaz Hager, Olivier Deriaz, Claudio Castellini, Henning Müller, and Barbara Caputo. Characterization of a benchmark database for myoelectric movement classification. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 23(1):73–83, 2015.

[18] Rita Chattopadhyay, Narayanan C Krishnan, and Sethuraman Panchanathan. Topology preserving domain adaptation for addressing subject based variability in semg signal. In *2011 AAAI Spring Symposium*, pages 4–9. AI Access Foundation, 2011.

[19] A. J. Young, L. J. Hargrove, and T. A. Kuiken. Improving myoelectric pattern recognition robustness to electrode shift by changing interelectrode distance and electrode configuration. *IEEE Transactions on Biomedical Engineering*, 59(3):645–652, 2012.

[20] Claudio Castellini, Angelo Emanuele Fiorilla, and Giulio Sandini. Multi-subject/daily-life activity EMG-based control of mechanical hands. *Journal of NeuroEngineering and Rehabilitation*, 6(1), November 2009.

[21] N. Jiang, S. Dosen, K. Muller, and D. Farina. Myoelectric control of artificial limbs—is there a need to change focus? [in the spotlight]. *IEEE Signal Processing Magazine*, 29(5):152–150, 2012.

[22] D. Farina, N. Jiang, H. Rehbaum, A. Holobar, B. Graimann, H. Dietl, and O. C. Aszmann. The extraction of neural information from the surface emg for the control of upper-limb prostheses: Emerging avenues and challenges. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(4):797–809, 2014.

[23] Claudio Castellini, Panagiotis Artemiadis, Michael Wininger, Arash Ajoudani, Merkur Alimusaj, Antonio Bicchi, Barbara Caputo, William Craelius, Strahinja Dosen, Kevin Englehart, and et al. Proceedings of the first workshop on peripheral machine interfaces: going beyond traditional surface electromyography. *Frontiers in Neurorobotics*, 8, Aug 2014.

[24] C. R. Steinhardt, J. Bettthauser, C. Hunt, and N. Thakor. Registration of emg electrodes to reduce classification errors due to electrode shift. In *2018 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–4, 2018.

[25] A. J. Young, L. J. Hargrove, and T. A. Kuiken. The effects of electrode size and orientation on the sensitivity of myoelectric pattern recognition systems to electrode shift. *IEEE Transactions on Biomedical Engineering*, 58(9):2537–2544, 2011.

[26] R. N. Khushaba. Correlation analysis of electromyogram signals for multiuser myoelectric interfaces. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(4):745–755, 2014.

[27] Muhammad Zia ur Rehman, Asim Waris, Syed Omer Gilani, Mads Jochumsen, Imran Khan Niazi, Mohsin Jamil, Dario Farina, and Ernest Nlandu Kamavuako. Multiday emg-based classification of hand motions with deep learning techniques. *Sensors*, 18(8):2497, 2018.

[28] Yu Du, Wenguang Jin, Wentao Wei, Yu Hu, and Weidong Geng. Surface emg-based inter-session gesture recognition enhanced by deep domain adaptation. *Sensors*, 17(3):458, 2017.

[29] Kelcy Tobey and Jonathan Mike. Single-leg glute bridge. *Strength & Conditioning Journal*, 40(2):110–114, April 2018.

[30] Anthony G. Schache, Tim W. Dorn, Gavin P. Williams, Nicholas A.T. Brown, and Marcus G. Pandy. Lower-limb muscular strategies for

increasing running speed. *Journal of Orthopaedic & Sports Physical Therapy*, 44(10):813–824, 2014. PMID: 25103134.

[31] ANTHONY G. SCHACHE, PETER D. BLANCH, TIM W. DORN, NICHOLAS A. T. BROWN, DOUG ROSEMOND, and MARCUS G. PANDY. Effect of running speed on lower limb joint kinetics. *Medicine & Science in Sports & Exercise*, 43(7):1260–1271, July 2011.

[32] https://www.neurocess.co.

[33] Xu Zhang and Ping Zhou. Sample entropy analysis of surface emg for improved muscle activity onset detection against spurious background spikes. *Journal of Electromyography and Kinesiology*, 22(6):901–907, 2012.

[34] Ronald W. Schafer. What is a savitzky-golay filter? [lecture notes]. *IEEE Signal Processing Magazine*, 28(4):111–117, 2011.

[35] Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

[36] Sepp Hochreiter. Ja1 4 rgen schmidhuber (1997).“long short-term memory”. *Neural Computation*, 9(8).

[37] Alex Graves, Santiago Fernández, Marcus Liwicki, Horst Bunke, and Jürgen Schmidhuber. Unconstrained online handwriting recognition with recurrent neural networks. In *Advances in Neural Information Processing Systems 20, NIPS 2008*, 2008.

[38] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. Bidirectional lstm networks for improved phoneme classification and recognition. In *International conference on artificial neural networks*, pages 799–804. Springer, 2005.

[39] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[40] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[41] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36. JMLR Workshop and Conference Proceedings, 2012.

[42] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[43] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

[44] Hermie J Hermens, Bart Freriks, Catherine Disselhorst-Klug, and Günter Rau. Development of recommendations for semg sensors and sensor placement procedures. *Journal of electromyography and Kinesiology*, 10(5):361–374, 2000.

[45] DS Dorcas and RN Scott. A three-state myo-electric control. *Medical and biological engineering*, 4(4):367–370, 1966.

[46] Laura Dipietro, Mark Ferraro, Jerome Joseph Palazzolo, Hermano Igo Krebs, Bruce T Volpe, and Neville Hogan. Customized interactive robotic treatment for stroke: Emg-triggered therapy. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 13(3):325–334, 2005.