

Transfer Learning in sEMG-based Gesture Recognition

Panagiotis Tsinganos^{*†}, Jan Cornelis[†], Bruno Cornelis[†], Bart Jansen^{†‡}, Athanassios Skodras^{*}

^{*} Dept. of Electrical and Computer Engineering, University of Patras, Patras, Greece

{panagiotis.tsinganos, skodras}@ece.upatras.gr

[†] Dept. of Electronics and Informatics, Vrije Universiteit Brussel, Brussels, Belgium

{ptsingan, jpcornel, bcorneli, bjansen}@etrovub.be

[‡] IMEC, 3001 Leuven, Belgium

Abstract—The latest advancements in the field of deep learning and biomedical engineering have allowed for the development of myoelectric interfaces based on deep neural networks. A longstanding problem of these interfaces is that the models cannot easily be applied to new users due to the high variability and stochastic nature of the electromyography signals. Further training a new model for every new subject requires the collection of large volumes of data. Therefore, this work proposes a transfer learning (TL) scheme which allows reusing the knowledge of a pre-existing model for a new user. Firstly, a convolutional neural network (CNN) is trained on an initial dataset using the data of multiple subjects. Then, the weights of this model are fine-tuned for a new target subject. The approach is evaluated on the Ninapro datasets DB2 and DB7. The experimentation included three different CNN models and eight preprocessing alternatives. The results showed that the success of the TL method depends on how the data are preprocessed. Specifically, the biggest accuracy improvement (+5.14%) is achieved when only the first 20% of the signal duration is used.

Index Terms—gesture recognition, electromyography, convolutional neural network, transfer learning

I. INTRODUCTION

Gesture recognition has been extensively studied for applications such as human computer interaction [1], [2], prosthetics control [3], sign language recognition [4] and serious gaming [5]. The electrical activity of the muscles can be recorded from the surface of the skin through surface electromyography (sEMG). Machine learning techniques can be used to analyze the sEMG signal and perform gesture recognition.

Over the past years, deep learning (DL) models have been applied with great success in sEMG-based gesture recognition. In these approaches, the sEMG signals are directly fed as input to a convolutional neural network (CNN) which is used to infer the gesture label. Although transfer learning (TL) is not a new concept, it has seen widespread use with DL models.

The definition of TL involves the clarification of two terms, the domain and the task. A domain \mathcal{D} consists of a feature space \mathcal{X} and a marginal probability distribution $P(X)$, where $X = \{x_1, \dots, x_n\} \in \mathcal{X}$. Given a specific domain, $\mathcal{D} = \{\mathcal{X}, P(X)\}$, a task consists of two components: a label space \mathcal{Y} and an objective predictive function $f : \mathcal{X} \rightarrow \mathcal{Y}$. The function f is used to predict the corresponding label $y = f(x)$ of a new instance x . This task, denoted by $\mathcal{T} = \{\mathcal{Y}, f(X)\}$,

is learned from the training data consisting of pairs $\{x_i, y_i\}$ [6].

Given source and target domains $\mathcal{D}_S, \mathcal{D}_T$ and the corresponding learning tasks $\mathcal{T}_S, \mathcal{T}_T$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or/and $\mathcal{T}_S \neq \mathcal{T}_T$, TL aims to help improve the learning of the target predictive function $f_T(\cdot)$ in \mathcal{T}_T using the knowledge in \mathcal{D}_S and \mathcal{T}_S .

For the domain of sEMG-based gesture recognition TL methods are very relevant. The biggest problem is the high variability of the sEMG signals not only between the subjects (inter-subject) but also between different recordings of the same subject (intra-subject) [7]. In the first case, these discrepancies are due to the different physiology of the forearm muscles across the subjects as well as due to the possible ways one can perform the same gesture. Differences in the recordings of the same subject are due to variations in gesture duration, applied force (intra-session) and sensor placement (inter-session). All these variations in the input signal can result in a poor performance of a trained classifier when tested on a different dataset.

Similarly to data augmentation, TL can also be used when we do not have enough training data. Recording a sufficient amount of sEMG signals from populations like the elderly or patients is not always feasible or good practice. In that case, one could transfer a model trained on an abundant dataset of healthy subjects using only a small dataset recorded from the target domain. Of particular interest in prosthetics is utilising the information learned from healthy subjects to improve the control of prosthetic hands used by amputees.

In this work, we investigate the application of TL and specifically domain adaptation to the problem of gesture recognition based on sEMG. Similarly to previous works we fine-tune a pretrained model using the sEMG signals collected from a subject not present in the source dataset [15], [19]. The difference is that the source and the target datasets come from different databases thus the recording protocol/hardware and the number of gestures are different. Further, we investigate the effect of various preprocessing steps and CNN models.

The rest of the paper is organized as follows. Section II provides a literature review on the use of TL for gesture recognition. In Section III, the details of the proposed method and of the CNN architectures used for experimentation are given.

Section IV describes the results followed by a discussion. Finally, conclusions are given in Section V.

II. RELATED WORK

TL for sEMG data is an important research topic. In [8], the recorded sEMGs of each subject in the source domain are considered to have a different conditional probability distribution $P(y|x)$. Then, two weighting schemes are applied that evaluate the similarities in distributions between the source and target domains. The optimal weights minimize the differences in the predictions of the source models on the target domain. A new classifier is learned using the re-weighted source data and a few target data. To address electrode shift that may happen due to donning/doffing, the authors of [9] transform the disturbed target data to the original source space. Assuming that the extracted features change linearly between the neighboring electrodes, the corrected feature values of the disturbed data can be estimated. Finally, the model trained on the source domain provides label predictions for the corrected target data. Based on the bilinear model of [10], the authors of [11] developed a framework for decomposing the myoelectric signals into subject specific and motion specific components based on the spectral coefficients of the wavelet transform. Then, a transformation is sought that minimizes the distance between the coefficients of different users or different sessions of the same user. In [12], the sEMG features from two datasets are embedded in a low dimensional space and then a linear transformation estimated by the least squares approach is applied to minimize their distance. Then, a k-nearest neighbors classifier was used for label prediction.

In [13], an adaptation algorithm based on linear discriminant analysis (LDA) is proposed to overcome the accuracy decrease over multiple days. Their algorithm adapts the mean and the covariance matrix of the LDA using a weighted average of their values calculated on the source and calibration sets. In [14], the batch normalization layers of a CNN model are trained for each subject independently, while the remaining parameters are shared between the subjects. Then, for a new subject a small calibration set is used to train only the batch normalization layers. In [15], a model trained on sEMG data is transferred to a new multi-modal classifier. The pretrained sEMG model weights are used to initialize the corresponding module in the multi-modal classifier, while the IMU module is trained from scratch. In [16], Progressive Neural Networks (PNN) are employed, where the source domain model is connected in parallel to the target domain model. The connection is made through layer-wise connections and the target domain model is trained to capture only the difference between the source and the new target space. In [7], a two-stage RNN is proposed. The model consists of a dense layer called adaptation matrix initialized with the identity matrix, which is followed by an LSTM layer. In the first step, only the LSTM is trained using the source domain data. Next, a small set of calibration data from the target domain is used to train the adaptation layer. In [17], an approach similar to [18] is followed. An ensemble of CNN models is

Algorithm 1: Transfer learning steps

Input: $\mathcal{D}_S, \mathcal{D}_T$: Source and target domain data

Input: $lr_1 \gg lr_2$: Learning rates for transfer learning steps

Output: Subject-specific models M_t

Base model M_S with L layers;

Pretrain base model M_S with \mathcal{D}_S ;

for subject data d_t in \mathcal{D}_T **do**

$M_t = M_S$;

$Update(M_{t:L}, lr_1, d_t)$; // Train last layer of M_t

$Update(M_t, lr_2, d_t)$; // Optimize M_t with a smaller learning rate

end

trained using the data in the source domain. These models are ranked according to the prediction accuracy on the target domain subject. Then, the top-k models are fine-tuned with the calibration data and the prediction for new inputs is computed as the majority voting of these models. In [19], LDA- and SVM-based TL approaches are compared to a typical CNN approach. It consists of pretraining the CNN classifier on the source domain, and then transferring all the layers except the last one to the target domain where the model weights are fine-tuned with the data of the target subject. The results showed that the transferred CNN models performed better than the SVM and LDA as well as the models trained from scratch on the target domain.

III. METHODS

In this study, we assume that $\mathcal{D}_S \neq \mathcal{D}_T$ meaning $\mathcal{X}_S = \mathcal{X}_T$ and $P_S(X) \neq P_T(X)$, i.e. the feature space is the same but the marginal distributions are different in the two domains. Further, $\mathcal{T}_S \neq \mathcal{T}_T$ is analyzed into $\mathcal{Y}_S \neq \mathcal{Y}_T$ and $f_S() \neq f_T()$, i.e. the target label space is different than the source and the models in the two domains are different.

TL is applied in three steps as shown in Algorithm 1. As the source domain we consider the data from all the subjects in a given dataset. This can be also considered as a multi-source domain due to the differences between the subjects [8]. In the first step, this data is used to pretrain the weights of the base model. Next, the entire model, except the last layer, is transferred to the target subject-specific model and the data from this target subject are used twice. Firstly, for training only a new randomly initialized last layer of the model, while keeping the transferred weights frozen, and secondly to slightly adapt the weights of the entire model.

The difference between $P_S(X)$ and $P_T(X)$ is due to the different way each subject performs a gesture. The reason why we assume $\mathcal{Y}_S \neq \mathcal{Y}_T$ is because in the target dataset we are interested in a different set of gestures. Specifically, the target domain consists of a subset of the available gestures in the source domain, $\mathcal{Y}_T \subset \mathcal{Y}_S$. The difference between $f_S()$ and

$f_T()$ lies on the model parameters of the last layers although the architecture remains the same.

A. Datasets and preprocessing

In this study, we evaluate the performance of the TL approach using two datasets. Specifically, the Ninapro DB2 [20] is used as the source domain in all the experiments. It consists of 40 subjects performing 40 gestures each one repeated 6 times. The target dataset is the Ninapro DB7 [21] containing 20 subjects that perform 40 gestures repeated 6 times. We have used only the first group of gestures in DB7 that consists of 17 basic movements of the fingers and the wrist. Data are recorded using 12 Trigno Delsys electrodes.

The preprocessing steps are as follows. The RMS envelope of each repetition signal is calculated followed by a Butterworth low-pass filter as in [22]. Next, the signals are normalized to $[0, 1]$ range and resampled at 100Hz to reduce processing time.

A set of preprocessing steps found in previous TL studies of sEMG datasets are investigated. These include (i) alignment of the electrodes across the subjects of source and target domains [16], (ii) training only with the middle section of the signal [16], [23], (iii) training only with the first samples of the signal, (iv) dividing the entire repetition into three parts with different labels [24].

The alignment of the electrodes (*EA*) for each subject consists of identifying the active channel for a specific set of gestures on the first subject [16]. This is done by calculating the iEMG feature, $iEMG(x) = \sum_i |x_i|$, for each gesture on the first subject. On subsequent subjects of the source and target datasets and for the same gestures, the iEMG feature is also calculated. The most active channel is the one that maximizes the iEMG feature. Then, the channels from every subject are circularly shifted accordingly to match the active channel of the first subject.

Training only with the middle steady section of the signal (*PLAT*) is done by removing the transient phases at both ends. In [16], this is achieved during the recording of the dataset. In contrast, the work of [23] considers the first and the last 1.5s of the signal (the average duration of the gestures is 5s) in the Ninapro dataset as transients which are removed. We have followed the latter approach, while we additionally varied the size of the segment that is kept as a proportion of the total duration of the repetition.

In many practical use cases, a correct gesture prediction is expected before the subject completes the gesture. Therefore, we investigated an approach where the models are trained only with the first few seconds of the gesture repetition (*ASC*). The segment of the signal that is kept is defined as a proportion of the total duration of the repetition.

In [24], the authors predict simultaneously the type of gesture and the gesture phase of the input. Each repetition is divided into the ascending, plateau and descending stages (*APD*). As a result the classifier needs to predict 3x more gesture labels. The subjects are instructed to remain in each stage for a specific amount of time, thus the ground truth

TABLE I
ARCHITECTURE DETAILS OF THE CNN MODELS

Model	AtzoriNet* [25]	TCNet [26]	TSNet
			ChCCConv1D(C, 3)
Details	Conv2D(32, (1,C))		Conv2D(32, (1,C))
	Conv2D(32, (3,3))	CConv1D(32, 3, 1)	MaxPool((2,2))
	MaxPool((3,3))	CConv1D(64, 3, 2)	Conv2D(64, (3,3))
	Conv2D(64, (5,5))	CConv1D(64, 3, 4)	MaxPool((2,2))
	MaxPool((3,3))	CConv1D(128, 3, 8)	Conv2D(64, (5,5))
	Conv2D(64, (5,1))	Attention()	MaxPool((2,2))
	Conv2D(G, (1,1))	FC(G)	Conv2D(G, (1,1))
	Softmax	Softmax	Softmax
Params	84K	75K	120K

C : number of sEMG channels, G : number of gestures, $Conv2D(k, (f_1, f_2))$: a 2D Convolutional layer of k filters with size (f_1, f_2) , $MaxPool((f_1, f_2))$: a max pooling layer of size (f_1, f_2) , $CConv1D(k, f, d)$: a 1D Causal Convolutional layer of k filters with size f and dilation ratio d , $ChCCConv1D(k, f)$: a 1D Channel-wise Causal Convolutional layer of k filters with size f , $Attention()$: an Attention layer, $FC(k)$: a fully connected layer of k units, $Softmax$: a softmax distribution layer

labelling is done automatically during the recording. In our case, we followed two approaches which are depicted in Fig. 1. In the first case, we arbitrarily divide the repetitions as a proportion of the total duration of the repetition. In the second case, a threshold set to 70% of the maximum value is used to define the borders of the plateau region.

The final preprocessing step is the use of a sliding window. To be valid in terms of real-time constraints, a window size of 200ms and a step of 10ms have been used. The input to the CNN models consists of matrices of size $N \times C$, where N is the window size and C the number of electrodes. Therefore, the input array has a size of 20×12 .

B. CNN models

As learner models $f()$ we investigate different CNN architectures that have been applied to sEMG-based gesture recognition. These are the modified AtzoriNet [25] denoted below as AtzoriNet*, the TCNet (Temporal Causal Network) model [26] and a new CNN that combines temporal and spatial convolutions called TSNet (Temporal Spatial Network). A brief description of the former two follows while more details are provided for the latter.

The version of AtzoriNet presented in [22] consists of blocks of convolutional and average pooling layers that end in a single softmax layer. The first convolution is applied across the electrodes dimension, while subsequent layers use square kernels. The outcome of our experimentation in [25] is a model with improved accuracy that includes max pooling and dropout layers.

The TCNet model [26] employs only convolutions along the time dimension. The difference to the regular convolution operation is that the input signal of each layer is zero-padded from the left to enforce causality. Thus, the output y_n at timestep n is computed using the inputs up to x_n , i.e. $y_n = f(x_{0..n})$. To obtain the output label for the input sequence an attention layer is appended after the last convolution. The

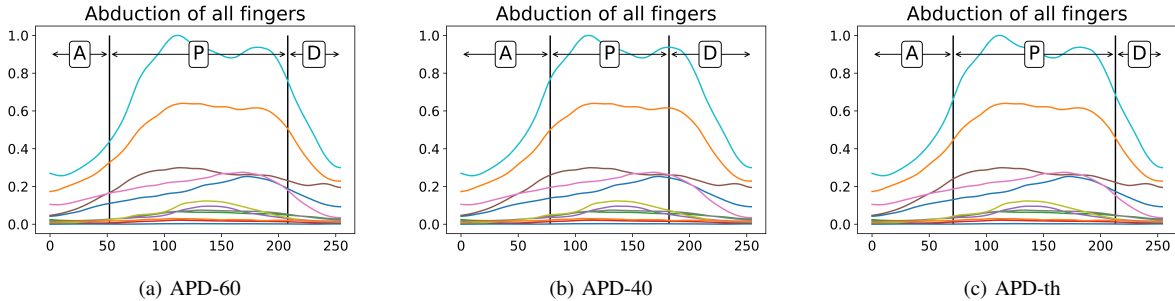


Fig. 1. Division of 'Abduction of fingers' into ascending (A), plateau (P) and descending (D) phases using (a) 60% for the plateau, (b) 40% for the plateau, (c) threshold-based algorithm.

TCNet architecture is built using residual blocks [27] and dilated convolutions with an exponential factor [26], [28].

Since the input signal is two dimensional (time and space), an appropriate design that exploits the information in both dimensions should be used. In typical feature extraction methods, the features are computed channel-wise and then they are combined by the classifier. Therefore, contrary to the above-mentioned models that either apply an initial spatial convolution [25] or use only temporal ones [26], the proposed TSNet starts with channel-wise causal convolutions whose output features are combined in later layers using regular 2D convolutions.

Given an input sEMG segment x of dimensions $N \times C$, where N is the duration of the window segment and C the number of electrodes, the output y^1 of the first channel-wise temporal causal convolution has the same dimensions and is calculated as:

$$\begin{aligned}
 y_{n,c}^1 &= (x_{:,c} * h_{:,c}^1)_n \\
 &= \sum_{m=0}^K x_{n-m,c} h_{m,c}
 \end{aligned} \tag{1}$$

where n and c are the indices for the time and electrode dimensions, K is the size of the convolutional kernel h . More than one such layers can be used in a sequential fashion, e.g. $y_{n,c}^2 = (y_{:,c} * h_{:,c}^2)_n$. Then, normal 2D convolutions are applied for subsequent layers l . The kernel sizes follow the models of AtzoriNet* [25]. A small grid search was performed to find the number of filters and the number of convolutional layers.

An overview of the architectures described above is provided in Table I.

Data preparation and training of the models are carried out on a workstation with an Intel i9-7920X 2.90 GHz processor, 128 GB RAM and an Nvidia GeForce RTX 2080Ti 11GB GPU. The implementation is based on the Tensorflow and Keras Python libraries.

C. Experiments

The experimentation consists of different types of training the models, namely AtzoriNet*, TCNet, and TSNet, with only target domain data (i.e. the *Baseline* approach), the pretraining on the source dataset (*Pretrain*), the first (*TL1*) and

TABLE II
LIST OF SELECTED HYPER-PARAMETERS VALUES

Type	Optimizer	Learning rate	Epochs	Batch size
Baseline	Adam	0.001	20	128
Pretrain	SGD	0.001	20	128
TL1,2	Adam	0.001	10+10	256

the second (*TL2*) steps of transfer learning. In our notation, each iteration is characterized by the dataset, the model, the preprocessing steps and the type of experiment. For example, *DB7_AtzoriNet_PLAT-60_Baseline* corresponds to the baseline experiment for the DB7 target dataset preprocessed with the plateau extraction function where the middle 60% of the signal is used for training the AtzoriNet* classifier. Similarly, *DB7_TSNet_APD-th_TL2* corresponds to the second step of transfer learning on the DB7 target dataset for the TSNet model and the threshold-based division of the gesture phases. To allow a fair comparison with the methods using the APD preprocessing, the predictions on the different phases of a gesture are summarized and presented along the per class results. From the available gesture repetitions in each dataset, one repetition is used as the test set and the rest are utilized for training the model.

1) *Hyper-parameters*: For the baseline experiments, the hyper-parameters for the optimization of the CNNs were chosen based on previous knowledge. However, for the pretraining and transfer learning steps a small grid search was performed. The search included the optimizer, learning rate, epochs and batch size. The selected values are shown in Table II. The hyper-parameters that define the CNN architectures, i.e. number of layers, size of convolutions, were selected based on previous knowledge [25], [26].

2) *Analysis*: The results are analysed under the statistical framework of repeated measures Analysis of Variance (rmANOVA) followed by post-hoc pairwise tests at a significance level of $\alpha = 0.05$. This is used in the case of studying the differences in mean scores of the same population under three or more different conditions. In our case, these conditions refer to the various preprocessing steps, the different CNN classifiers and the evaluation type, e.g. Baseline or TL2.

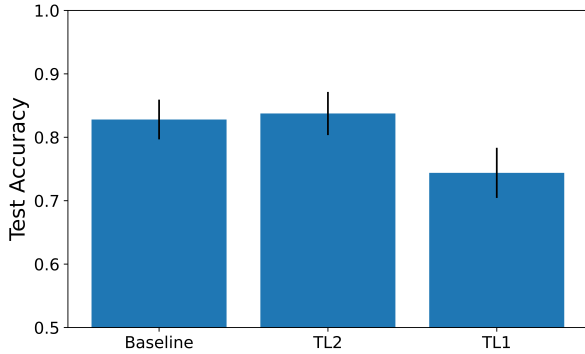


Fig. 2. Difference in classification accuracies between Baseline, TL2 and TL1 for the case of AtzoriNet* without preprocessing. The vertical black bars correspond to standard deviations.

The statistical analysis was performed using the IBM SPSS software.

IV. RESULTS AND DISCUSSION

In Table III the results of the Baseline and TL2 experiments for the target dataset Ninapro DB7 are shown. Results on TL1 were always worse than the corresponding Baseline therefore they are omitted. Fig. 2 shows the difference in accuracy between Baseline, TL1 and TL2. The results of the Pretrain step on Ninapro DB2 are shown in Table IV. Results are reported as the average and standard deviation of the test accuracy across the target subjects. The values in the APD methods correspond to the test accuracy summarized over the results of the same gesture, i.e. accuracy on the 17 gesture labels, while in parentheses are the accuracies calculated on the models' output, i.e. accuracy on the 51 labels of the APD method.

The results of the repeated measures ANOVA are given below. On the target dataset, the Shapiro-Wilk test of normality showed that the data were normally distributed ($p > 0.05$). Mauchly's Test of Sphericity indicated that the assumption of sphericity had been violated, $\chi^2(104) = 202.804, p \ll 0.05$, and therefore, a Greenhouse-Geisser correction was used. The three-way interactions between CNN model, preprocessing and evaluation type were not significant, $F(5.214, 99.070) = 1.966, p = 0.087$. However, there were significant two-way interactions between CNN model and preprocessing, $F(5.674, 107.806) = 3.426, p = 0.005$, as well as between preprocessing and evaluation type, $F(4.096, 77.822) = 5.941, p \ll 0.05$. A further analysis of the interaction between preprocessing and evaluation type indicates whether the different preprocessing steps had any influence on the differences between the Baseline and the TL2 evaluations. Specifically, after applying the Bonferonni adjustment, a significant difference for the preprocessing step ASC-20, $p = 0.001$, and ASC-40, $p = 0.027$, was found.

An analysis of the interaction between preprocessing and CNN model can show whether the preprocessing steps had

any effect on the performance of the networks. The differences between the models were not significant in the cases of EA, $p = 0.993$, and ASC-20, $p = 0.061$. After applying the Bonferonni adjustment, there were significant differences between the TSNet and the other two models for the case of no preprocessing, $p \ll 0.05$ with AtzoriNet* and $p = 0.004$ with TCNet, and for the APD-60, $p = 0.006$ with AtzoriNet* and $p = 0.029$ with TCNet. Further, the difference between TSNet and AtzoriNet* was significant for the ASC-40, $p = 0.02$, APD-40, $p = 0.016$, and APD-th, $p = 0.003$. Lastly, the difference between TSNet and TCNet were also significant for the PLAT-60, $p = 0.019$.

The low accuracy of TL1 shown in Fig. 2 could be attributed to two factors. Firstly, the differences between the two datasets and specifically the different number of gestures, i.e. 40 gestures in the source DB2 and 17 gestures in the target DB7. Secondly, during the TL1 step only a small fraction of the total parameters of the model is trained, e.g the last layer in AtzoriNet* consists of 2600 weights while total number of parameters is 84K. Therefore, a final fine-tuning of the entire model is required to adapt to a new subject.

The above significance tests show that the type of preprocessing plays a significant role in the training of the transferred models. In the majority of the cases, the improvement gained by the TL approach was significant only with the ASC-20 preprocessing step. This finding is very important because it means that the classification accuracy can be improved with a TL approach when the prediction is made with respect to the first 20% of the signal. Thus, the use of ASC-20 preprocessing in prosthetic control should certainly be investigated. However, when the entire signal is available during training, the model performance can be higher without a TL approach.

Regarding the TSNet model, the results in Table III show that the classification at TL2 is always higher than the Baseline. Further, between the evaluated models, TSNet had generally a better performance but it was dependent on the preprocessing step.

V. CONCLUSIONS

Our analysis demonstrated that any improvement caused by the use of TL is subject to the preprocessing steps. The experimentation including three different CNN models and eight preprocessing alternatives showed that the ASC-20 preprocessing resulted in a significant improvement of the TL approach compared to the baseline. This translates to +5.14% higher accuracy for the Ninapro DB7 with the TCNet model. The TSNet model had generally a better performance but it was dependent on the preprocessing step. In all the cases, the TL2 accuracy was higher than Baseline. Further, both TL steps are required to successfully transfer the model to a new subject.

Overall, it is evident that under certain circumstances transfer learning can be a useful tool for improving the performance of deep learning classifiers for the problem of gesture recognition.

TABLE III

RESULTS IN AVERAGE \pm STD FOR NINAPRO DB7 TARGET DATASET. THE VALUES IN PARENTHESES CORRESPOND TO THE TEST ACCURACY CALCULATED ON THE MODELS' OUTPUT. I.E. ACCURACY ON THE 51 LABELS OF THE APD METHODS

Model	Preprocessing	Baseline	TL2
AtzoriNet*	None	0.8640 \pm 0.0372	0.8523 \pm 0.0351
	EA	-	0.8773 \pm 0.0480
	PLAT-60	0.9033 \pm 0.0405	0.9057 \pm 0.0443
	ASC-20	0.7755 \pm 0.0792	0.7953 \pm 0.0783
	ASC-40	0.8360 \pm 0.0462	0.8502 \pm 0.0612
	APD-40	0.8569 \pm 0.0403 (0.7259 \pm 0.0733)	0.8578 \pm 0.0380 (0.7447 \pm 0.0698)
	APD-60	0.8577 \pm 0.0416 (0.7557 \pm 0.0625)	0.8613 \pm 0.0385 (0.7639 \pm 0.0600)
	APD-th	0.8569 \pm 0.0394 (0.7719 \pm 0.0611)	0.8628 \pm 0.0410 (0.7844 \pm 0.0633)
TCNet	None	0.8709 \pm 0.0405	0.8634 \pm 0.0425
	EA	-	0.8804 \pm 0.0394
	PLAT-60	0.9140 \pm 0.0387	0.8938 \pm 0.0501
	ASC-20	0.7777 \pm 0.0507	0.8291 \pm 0.0860
	ASC-40	0.8402 \pm 0.0621	0.8641 \pm 0.0538
	APD-40	0.8704 \pm 0.0375 (0.7586 \pm 0.0699)	0.8805 \pm 0.0440 (0.7692 \pm 0.0726)
	APD-60	0.8776 \pm 0.0396 (0.7845 \pm 0.0644)	0.8827 \pm 0.0422 (0.7905 \pm 0.0695)
	APD-th	0.8753 \pm 0.0474 (0.8044 \pm 0.0644)	0.8767 \pm 0.0478 (0.8030 \pm 0.0735)
TSNet	None	0.8743 \pm 0.0369	0.8833 \pm 0.0452
	EA	-	0.8808 \pm 0.0401
	PLAT-60	0.9170 \pm 0.0429	0.9193 \pm 0.0423
	ASC-20	0.7717 \pm 0.0743	0.7867 \pm 0.0723
	ASC-40	0.8493 \pm 0.0498	0.8673 \pm 0.0436
	APD-40	0.8666 \pm 0.0321 (0.7329 \pm 0.0726)	0.8706 \pm 0.0400 (0.7525 \pm 0.0688)
	APD-60	0.8705 \pm 0.0353 (0.7631 \pm 0.0693)	0.8773 \pm 0.0434 (0.7782 \pm 0.0674)
	APD-th	0.8694 \pm 0.0367 (0.7798 \pm 0.0697)	0.8791 \pm 0.0391 (0.7994 \pm 0.0679)

TABLE IV

RESULTS FOR PRETRAIN ON NINAPRO DB2 SOURCE DATASET

Preprocessing	AtzoriNet*	TCNet	TSNet
None	0.3246	0.4285	0.3562
EA	0.3392	0.4701	0.3916
PLAT-60	0.3866	0.5035	0.4907
ASC-20	0.2979	0.4041	0.2117
ASC-40	0.3657	0.5025	0.4213
APD-40	0.2748 (0.2128)	0.4470 (0.3721)	0.3539 (0.2950)
APD-60	0.2834 (0.2028)	0.4388 (0.3829)	0.3858 (0.3292)
APD-th	0.2955 (0.2357)	0.4507 (0.3920)	0.3652 (0.3303)

ACKNOWLEDGMENT

The work is supported by the 'Andreas Mentzelopoulos Scholarships for the University of Patras' and the VUB-UPatras International Joint Research Group on ICT (JICT).

REFERENCES

- [1] J. Qi, G. Jiang, G. Li, Y. Sun and B. Tao, "Intelligent Human-Computer Interaction Based on Surface EMG Gesture Recognition," in *IEEE Access*, vol. 7, pp. 61378–61387, 2019, doi: 10.1109/ACCESS.2019.2914728.
- [2] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 1–54, 2015, doi: 10.1007/s10462-012-9356-9.
- [3] N. Parajuli et al., "Real-time EMG based pattern recognition control for hand prostheses: A review on existing methods, challenges and future implementation," *Sensors (Basel)*, vol. 19, no. 20, p. 4596, 2019, doi: 10.3390/s19204596.
- [4] M. J. Cheok, Z. Omar, and M. H. Jaward, "A review of hand gesture and sign language recognition techniques," *Int. j. mach. learn. cybern.*, vol. 10, no. 1, pp. 131–153, 2019, doi: 10.1007/s13042-017-0705-5.
- [5] N. Nasri, S. Orts-Escolano, and M. Cazorla, "An sEMG-controlled 3D game for rehabilitation therapies: Real-time time hand gesture recognition using deep learning techniques," *Sensors (Basel)*, vol. 20, no. 22, p. 6451, 2020, doi: 10.3390/s20226451.
- [6] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010, doi: 10.1109/tkde.2009.191.
- [7] I. Ketyko, F. Kovacs, and K. Z. Varga, "Domain adaptation for sEMG-based gesture recognition with recurrent neural networks," in 2019 International Joint Conference on Neural Networks (IJCNN), 2019, doi: 10.1109/ijcnn.2019.8852018.
- [8] R. Chattopadhyay, Q. Sun, W. Fan, I. Davidson, S. Panchanathan, and J. Ye, "Multisource domain adaptation and its application to early detection of fatigue," *ACM Trans. Knowl. Discov. Data*, vol. 6, no. 4, pp. 1–26, Dec. 2012, doi: 10.1145/2382577.2382582.

- [9] C. Prahm et al., "Counteracting Electrode Shifts in Upper-Limb Prosthesis Control via Transfer Learning," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 27, no. 5, pp. 956–962, May 2019, doi: 10.1109/TNSRE.2019.2907200.
- [10] T. Matsubara and J. Morimoto, "Bilinear Modeling of EMG Signals to Extract User-Independent Features for Multiuser Myoelectric Interface," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 8, pp. 2205–2213, Aug. 2013, doi: 10.1109/TBME.2013.2250502.
- [11] X. Sheng, B. Lv, W. Guo, and X. Zhu, "Common spatial-spectral analysis of EMG signals for multiday and multiuser myoelectric interface," *Biomed. Signal Process. Control*, vol. 53, p. 101572, Aug. 2019, doi: 10.1016/j.bspc.2019.101572.
- [12] N. Rabin, M. Kahlon, SaritMalayev, and A. Ratnovsky, "Classification of human hand movements based on EMG signals using nonlinear dimensionality reduction and data fusion techniques," *Expert Syst. Appl.*, p. 113281, Feb. 2020, doi: 10.1016/j.eswa.2020.113281.
- [13] M. M. C. Vidovic, H.-J. Hwang, S. Amsuss, J. M. Hahne, D. Farina, and K.-R. Muller, "Improving the Robustness of Myoelectric Pattern Recognition for Upper Limb Prostheses by Covariate Shift Adaptation," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 24, no. 9, pp. 961–970, Sep. 2016, doi: 10.1109/TNSRE.2015.2492619.
- [14] 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, doi: 10.3390/s17030458.
- [15] W. Wang, B. Chen, P. Xia, J. Hu, and Y. Peng, "Sensor Fusion for Myoelectric Control Based on Deep Learning With Recurrent Convolutional Neural Networks," *Artif. Organs*, vol. 42, no. 9, pp. E272–E282, Sep. 2018, doi: 10.1111/aor.13153.
- [16] U. Côté-Allard et al., "Deep Learning for Electromyographic Hand Gesture Signal Classification Using Transfer Learning," *ArXiv e-prints*, Jan. 2018, Accessed: Mar. 08, 2018. [Online]. Available: <http://arxiv.org/abs/1801.07756>.
- [17] K.-T. Kim, C. Guan, and S.-W. Lee, "A subject-transfer framework based on single-trial EMG analysis using convolutional neural networks," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 1, pp. 94–103, 2020, doi: 10.1109/TNSRE.2019.2946625.
- [18] S. Kanoga, T. Hoshino, and H. Asoh, "Subject Transfer Framework Based on Source Selection and Semi-Supervised Style Transfer Mapping for Semp Pattern Recognition," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 1349–1353, doi: 10.1109/ICASSP40776.2020.9054070.
- [19] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "A Deep Transfer Learning Approach to Reducing the Effect of Electrode Shift in EMG Pattern Recognition-Based Control," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 2, pp. 370–379, Feb. 2020, doi: 10.1109/TNSRE.2019.2962189.
- [20] M. Atzori et al., "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Sci. Data*, vol. 1, no. 1, p. 140053, 2014.
- [21] A. Krasoulis, I. Kyranou, M. S. Erden, K. Nazarpour, and S. Vijayakumar, "Improved prosthetic hand control with concurrent use of myoelectric and inertial measurements," *J. Neuroeng. Rehabil.*, vol. 14, no. 1, p. 71, Dec. 2017, doi: 10.1186/s12984-017-0284-4.
- [22] 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," *Front. Neuro-robot.*, vol. 10, p. 9, 2016.
- [23] M. Zanghieri, S. Benatti, A. Burrello, V. Kartsch, F. Conti, and L. Benini, "Robust real-time embedded EMG recognition framework using Temporal Convolutional Networks on a multicore IoT processor," *IEEE Trans. Biomed. Circuits Syst.*, vol. 14, no. 2, pp. 244–256, 2020, doi: 10.1109/TBCAS.2019.2959160.
- [24] M. Stachaczyk, S. F. Atashzar, S. Dupan, I. Vujaklija, and D. Farina, "Toward universal neural interfaces for daily use: Decoding the neural drive to muscles generalises highly accurate finger task identification across humans," *IEEE Access*, vol. 8, pp. 149025–149035, 2020, doi: 10.1109/access.2020.3015761.
- [25] P. Tsinganos, B. Cornelis, J. Cornelis, B. Jansen, and A. Skodras, "Deep learning in EMG-based gesture recognition," in *Proceedings of the 5th International Conference on Physiological Computing Systems*, 2018, doi: 10.5220/0006960201070114.
- [26] P. Tsinganos, B. Cornelis, J. Cornelis, B. Jansen, and A. Skodras, "Improved gesture recognition based on sEMG signals and TCN," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, doi: 10.1109/icassp.2019.8683239.
- [27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *ArXiv e-prints*, Dec. 2015, Accessed: May 03, 2018. [Online]. Available: <http://arxiv.org/abs/1512.03385>.
- [28] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," *ArXiv e-prints*, Nov. 2015, Accessed: Oct. 20, 2018. [Online]. Available: <http://arxiv.org/abs/1511.07122>.