Online testing of myoelectric pattern recognition with unsupervised domain adaptation

A potential method for addressing edge cases in myoelectric prosthesis control

Master's thesis in Complex Adaptive Systems

Alexander Hannius

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Online testing of myoelectric pattern recognition with unsupervised domain adaptation

A potential method for addressing edge cases in myoelectric prosthesis control

Alexander Hannius

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2023

Online testing of myoelectric pattern recognition with unsupervised domain adaptation
A potential method for addressing edge cases in myoelectric prosthesis control
Alexander Hannius

Supervisor: Jan Zbinden, Center for Bionics and Pain Research, Department of Electrical Engineering
Supervisor: Rita Laezza, Division of Systems and Control, Department of Electrical Engineering
Examiner: Petter Falkman, Division of Systems and Control, Department of Electrical Engineering

Master's Thesis 2023
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2023

Online testing of myoelectric pattern recognition with unsupervised domain adaptation
A potential method for addressing edge cases in myoelectric prosthesis control
Alexander Hannius
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Wide-scale adoption of deep myoelectric pattern recognition (MPR) for decoding motor intent is hindered by the lengthy data collection sessions required to train classifiers tailored to each individual user. Due to this already cumbersome process, it is infeasible to manually record even larger data sets that cover the wide variety of conditions encountered in real-world application of an MPR-powered device such as a robotic prosthesis. Other areas of deep learning often employ domain adaptation to cope with restrictions on data collection. Through online tests with able-bodied subjects, we show that deep MPR can be improved with an unsupervised domain adaptation method to make more accurate hand gesture classifications on data domains that are not represented in the labelled subset of training data.

Keywords: Machine Learning, Domain adaptation, Prosthetics, Bionic Limbs.

Acknowledgements

First and foremost I would like to thank my family and friends for acting as a pillar of support in an otherwise turbulent year of exploring my own ability to plan and conduct research. I would also like to thank my supervisors Jan Zbinden and Rita Laezza for placing the trust in me to propose and pursue a project within this field. Myoelectric control and by extension human-machine interfaces represent a very exciting trajectory in current technological development and I'm glad to have been able to dip my toes in some of the work that enables it. All in all the research and writing of this thesis has made me certain that I will return to the field at some point in the future.

The computations for this work were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at https://www.c3se.chalmers.se partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

Alexander Hannius, Gothenburg, August 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AdaBN	Adaptive Batch Normalization
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
CNNSE	Convolutional Neural Network with Squeeze and Excitation
DANN	Domain Adversarial Neural Networks
EMG	Electromyography
FFNN	Feed-Forward Neural Network
GRL	Gradient Reversal Layer
HD-sEMG	High-Definition Surface Electromyography
LDA	Linear Discriminant Analysis
MCD	Maximum Classifier Discrepancy
MPR	Myoelectric Pattern Recognition
sEMG	Surface Electromyography
SVM	Support Vector Machine
SWD	Sliced Wasserstein Discrepancy
TCN	Temporal Convolutional Networks
UDA	Unsupervised Domain Adaptation
VR	Virtual Reality

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

k	Index for class label

Parameters

γ	Batch normalization learnable scale parameter
β	Batch normalization learnable shift parameter
K	Number of distinct class labels
θ_f	Model parameters for the encoder in DANN
θ_y	Model parameters for the label classifier in DANN
θ_d	Model parameters for the domain classifier in DANN
θ_g	Model parameters for the encoder in MCD
θ_{f1}	Model parameters for classifier F_1 in MCD
θ_{f2}	Model parameters for classifier F_2 in MCD
n	Hyperparameter determining the number of training loop repeats per batch in MCD

Variables

x	EMG input sample
y	Predicted class label
d	Predicted domain label in DANN
$\mathbf{f}$	Feature space vector in DANN and MCD

L_y	Label classifier cross entropy loss in DANN
L_d	Domain classifier cross entropy loss in DANN

Functions

$d(p_1, p_2)$	MCD distance function
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2
$W_1(\mu_1, \mu_2)$	Wasserstein-1 distance function

Contents

List of Figures

1
Introduction

Myoelectric pattern recognition (MPR) can restore some functionality lost due to limb amputation. MPR enables the development of robotic prostheses that can replicate the intended motion of a user. In order to reach the full utility of the biological limb it intends to replace, an electromyography (EMG) -based robotic prosthesis must correctly classify intended motion regardless of the signal's recording environment.

Studies on non-amputee subjects that incorporate a multitude of dynamic factors (such as limb position) have demonstrated that domain adaptation can improve performance over conventional training in cases where a domain shift in the data is present [1]. Additionally, patients who are active users of implanted electrodes have reported a sensation of decreased accuracy when performing motions outside the scope of a typical lab-recorded training dataset (e.g. twitching when raising one's shoulder to reach a shelf). While in no way conclusive, this suggests that some causes for domain shift persist in users with implanted electrodes, motivating further study of domain adaptation methods for MPR and it's possible clinical application.

In this work, we evaluate different pairings of *unsupervised domain adaptation* methods with neural network architectures from MPR and time-series classification literature to create a domain adaptation training pipeline suitable for electromyography data. We also demonstrate how a classifier trained with this pipeline improves accuracy in the presence of domain shift through real-time testing with 10 able-bodied subjects.

1.1 Robotic prosthetic hands in the real world

As of writing, advances in the mechanical capabilities of robotic hands far outpace the human-machine interfaces that are built to control them. Telerobotics applications often employ some type of sensor glove or camera tracking to control robotic manipulator arms, these approaches are inaccessible to people with hand amputation. Myoelectric control provides a workaround to this issue by being able to pick up on any arbitrary muscle activation with the appropriate placement of sensing electrodes, allowing the use of residual muscles to control a robotic prosthesis hand.

Direct control is arguably the most common way to decode motor intention from myoelectric signals (judging from its ubiquity in commercially available systems)

but is limited in practice. It works by placing a threshold on each input electrode channel that determines the activation of a corresponding prosthesis movement, difficulties arise when trying to isolate individual muscle activations and it can often only reliably control a single degree of freedom (e.g. hand opening/closing).

Myoelectric control that uses *pattern recognition* differentiates itself by considering the entire width of electrode input channels across a time window, exposing contextual cues that can be used to separate individual muscle activations. Typically, a set of features are extracted from the EMG window (e.g. the Hudgins set [2]) and these features are in turn fed into machine learning models such as linear discriminant analysis (LDA) [3] and support vector machines (SVM) [4]. Similarly to recent developments in many other research fields, deep learning methods (such as CNNs [5] and transformers [6] trained with gradient descent algorithms) have largely superseded the use of the more classical machine learning approaches, opting for learned features as opposed to a predefined set of transformations.

1.2 Deep domain adaptation

Within deep learning research, methods from a sub-field known as *domain adaptation* are often employed to deal with discrepancies in data samples seen during training versus those in real-world application (referred to as a *domain shift* in the data). Critically, domain adaptation differentiates itself from the "usual" learning setting by *excluding* the assumption that both source and target datasets are samples from the *same distribution*. Domain adaptation methods work under the assumption that classification performance on a source set is a good indicator of performance on a target set if the two distributions are similar. A neural network-based feature encoder that receives data with distinct distributions in the input layer can be trained to make the two distributions appear similar in the resulting output feature layer. Going by this assumption, a classifier trained on these adapted output features should see better knowledge transfer between domains. This makes domain adaptation a potentially useful tool in situations where labeled data for a particular task is gathered from a distribution (the source domain) that is different from the one the final application makes predictions on (the target domain). This setup, in which unlabeled target data is available, is typically referred to as unsupervised domain adaptation (UDA).

A testament to the usefulness of domain adaptation can be found in research on semantic image segmentation, which refers to the problem of assigning a class to every single pixel in an image. Manually assigning class labels to each pixel in an image presents a far more laborious process than labeling training data for regular image classification (in which an entire image corresponds to a single class label). A proposed solution is to use simulated data, where pixel labels can be acquired "for free" by rendering objects with known ground truth labels to an image [7]. The trade-off is that the training now relies on the simulation's capability of generating images that are faithful recreations of real-world examples. In this case, domain adaptation can use real-world target data to calibrate a network that only has ac-

cess to simulated samples during training [8], bridging the gap between simulation and the real world.

In the context of this work, the distinction between source and target domains are recordings that are from the same subject but with different dynamic factors (limb pose in the case of the online test). Both supervised and unsupervised domain adaptation are often considered to be a type of transfer learning [9].

1.3 Previous work

Research on supervised/unsupervised domain adaptation for MPR typically focus on domain shifts that are linked to the use of surface electrodes [10], issues that are largely absent in users with implanted electrodes [11].

Supervised domain adaptation for surface electromyography (sEMG) was explored by Ketyko et al. [12]. Pairings of source and target datasets yielding either *intra-session*, *inter-session*, or *inter-subject* domain shifts were considered through offline evaluation on existing benchmark datasets. Domain adaptation was achieved by pre-training on samples from the source set followed by fine tuning on samples from the target set. While it is observed that fine tuning results in improved performance, labels from the target set must be used and the method is thus limited to cases where these are readily available. Similarly, concepts from unsupervised domain adaptation have been applied in a calibration process used to reduce the influence of inter-session domain shift caused by variations in sEMG sensor placement [13].

Du et al. presented CapgMyo, a dataset created for offline evaluation of high-density surface electromyography (HD-sEMG) and to study domain shift caused by jumps in time between subsequent recordings for the same subject as well as inter-subject shifts [14]. In addition, it was demonstrated that a simple unsupervised domain adaptation method (AdaBN[15]) can improve MPR performance in tasks with inter-session and inter-subject pairings of source and target data domains.

More recently, Côté-Allard et al. used virtual reality (VR) and a hand tracking camera to provide real-time feedback for participants in order to create a dataset that incorporates dynamic factors such as gesture intensity, limb position, electrode shift, and transient signal states. By decoupling hand position feedback from the recorded surface electromyography (sEMG) signals, the dataset can be used as a benchmark without biasing it towards any particular classifier. Datasets recorded in this manner were consequently used to investigate the effects of unsupervised domain adaptation [1] and transfer learning [16] on the mentioned factors. No online experiment with unsupervised domain adaptation is performed due to the participants feedback coming solely from a hand tracking camera. The authors distinguish it from true online testing by referring to the method as training with *dynamic* datasets.

2
Theory

This section provides a contextual overview of unsupervised domain adaptation by covering a subset of methods, roughly following a progression from basic to more recent developments.

- **Section 2.1** covers Adaptive Batch Normalization (AdaBN), which is primarily relevant for being previously used with CapgMyo.
- **Section 2.2** describes a more recent take on UDA in the form of Domain-Adversarial training of Neural Networks (DANN).
- **Section 2.3** describes an alternative domain adaptation method known as Maximum Classifier Discrepancy (MCD).
- **Section 2.4** extends on the concepts introduced by MCD with Sliced Wasserstein Discrepancy (SWD).

Section 2.2 and Section 2.4 corresponds to the methods used in the actual offline and online experiments (described further in Section 3).

PyTorch Adapt is a python library that contains reference implementations of various unsupervised domain adaptation methods compatible with the PyTorch machine learning framework [17]. It includes all of the methods that have been described in this section.

2.1 Adaptive Batch Normalization

Batch normalization was originally proposed by Ioffe and Szegedy [18] as a means to minimize the *internal* domain shift between neural network layers in a bid to improve training of deeper neural networks. While batch normalization evidently helps with training, the exact mechanism by which it happens has been questioned since its initial introduction [19].

Batch normalization adds a running mean and variance that are used to normalize the layer inputs. However, with neuron outputs constrained to normalized values, parts of the network's nonlinear activation functions become unavailable and representational power is compromised. To compensate for this, two additional learnable parameters γ and β are added. They allow the optimization algorithm to respectively scale and shift the normalized inputs to each neuron, restoring the network's capability to use any arbitrary window of the activation function.

Adaptive batch normalization (AdaBN) [15] is largely identical to regular batch normalization with the exception that it continues to update parameters beyond the training phase, allowing the the method to adapt to new data.
In the case of CapgMyo, AdaBN was chosen as a domain adaptation method under the hypothesis that the knowledge allowing the classifier to discriminate between classes is stored in the network weights, whereas knowledge about the domain is contained within the batch normalization parameters.

2.2 Domain-adversarial training

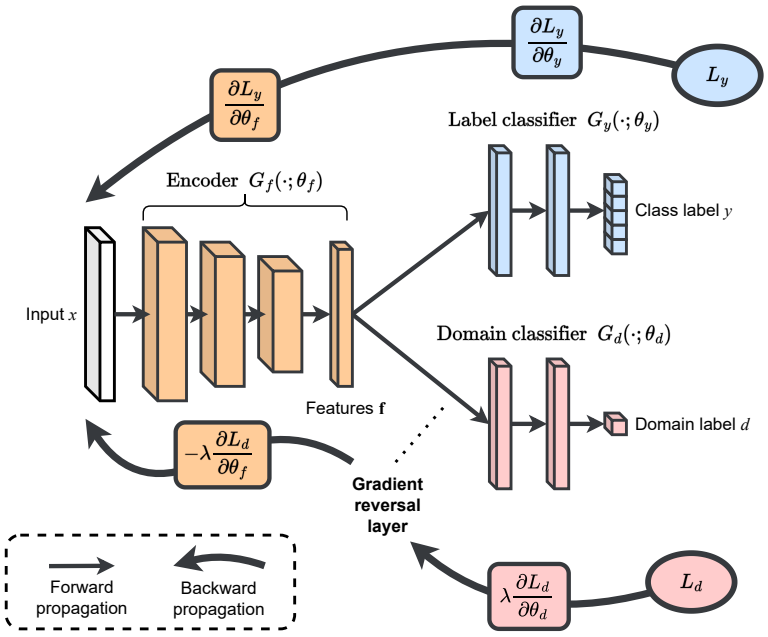


Figure 2.1: A diagram illustrating the structure of a domain adversarial neural network (DANN) with its encoder and two classifier heads. The primary mechanism that allows DANN to function as a domain adaptation method is the gradient reversal layer (GRL) found between the encoder output layer and domain classifier input layer. During backpropagation the GRL flips the sign of the loss derivatives coming from the domain classifier branch (lowermost in the figure), causing all of the layers to the left of it (i.e. the encoder) to be updated to generate features **f** that maximize the domain prediction error. The label classifier branch (uppermost in the figure) makes sure that **f** remains descriptive w.r.t. class labels by propagating updates that minimize the class prediction error. The simultaneous training of the encoder and domain classifier to respectively *maximize* and *minimize* the domain prediction error is hypothesized to make **f** tend towards containing no information about the domain origin of input x. Figure adapted from [20].

Domain adversarial neural networks (DANN [20]) learn to produce domain-invariant representations by training two classifier heads on the output of a feature extractor

(i.e. an encoder). The training procedure uses labeled data from a source set and unlabeled data from a target set, each sample is given an additional "domain label" that corresponds to its domain of origin.

The first head is an ordinary label classifier, which minimizes its loss by correctly predicting the class labels of incoming samples. The other, a domain classifier, attempts to predict which input domain the output representation belongs to. For samples where only the domain label is present, forward propagation of the label classifier is skipped. By negating the backwards propagated loss derivatives in what the authors call a "gradient reversal layer", the weights of the encoder are updated to learn representations that effectively fool the domain classifier into producing outputs that maximize the Shannon entropy (i.e. no information about true domain labels).

Côté-Allard et al. [1] proposed a variation of DANN to perform domain adaptation for MPR.

2.3 Maximum Classifier Discrepancy

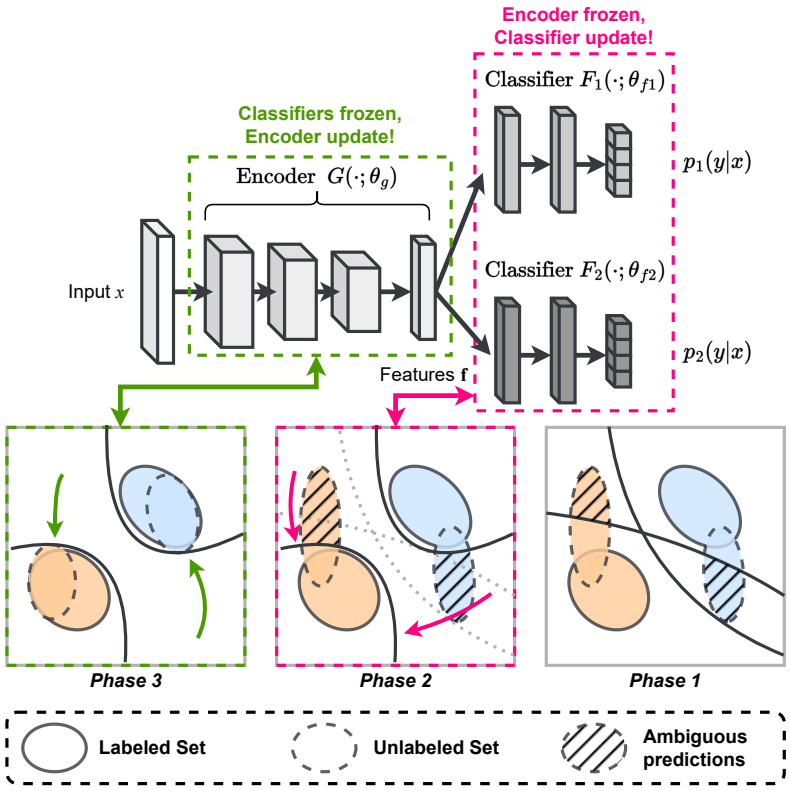


Figure 2.2: A diagram illustrating the structure of a neural network using maximum classifier discrepancy for unsupervised domain adaptation. The upper- and lowermost branches in the figure are identical in terms of neural network architecture but are initialized with different parameters. When training both branches on the same label prediction task, the different starting parameters will yield slightly different decision boundaries. By using a so-called discrepancy loss function to measure the extent at which the two decision boundaries differ, parameters can be updated to shape said boundaries. By iteratively alternating between freezing the encoder and training to maximize discrepancy (causing decision boundaries to diverge) and freezing the classifiers while minimizing discrepancy (causing the now unfrozen encoder to generate features $\mathbf{f}$ that satisfy the new, "stricter", decision boundaries), encoder features generated from unlabeled data are encouraged to align with labeled features since only the latter have an effect on decision boundaries. Figure adapted from [8].

The key idea behind *maximum classifier discrepancy* (MCD) [8] stems from the observation that identically training two classifiers with different weight initialisations will result in a region of the input space in which the two classifiers disagree on their predictions, due to slight variations in decision boundaries. By defining a measure of how much the two classifiers disagree (L1-distance in the case of MCD, see Equation 2.1) and combining it with the notion of source and target data domains, one can create a training pipeline that detects which target samples falls outside the support

of the source set (and are thus likely to be misclassified) without ever having access to target labels.

$$d(p_1, p_2) = \frac{1}{K} \sum_{k=1}^{K} |p_{1k} - p_{2k}| \tag{2.1}$$

By default, the region in which two identically trained classifiers disagree will be relatively small. In order to remedy this, MCD temporarily stops updating parameters in the encoder network that produces the classifiers input embeddings, and instead updates the classifiers to *maximize* the measure of how much their predictions differ (i.e. the discrepancy loss). The decision boundaries are moved so that the classifiers disagree on the largest possible portion of target set predictions. Importantly, the classification loss on the source set is still used in this step to ensure that the classifiers decision boundaries can move without compromising the correctness of the labeled source set predictions.

After having refined the decision boundaries to more tightly enclose the source set, the next step of the training procedure is to update the encoder (and temporarily stop updating the classifiers) to bring sample embeddings from the disagreeing regions into the regions where classifiers agree (i.e. the training now *minimizes* the discrepancy loss). No classification loss is computed for this step since we are only updating the encoder.

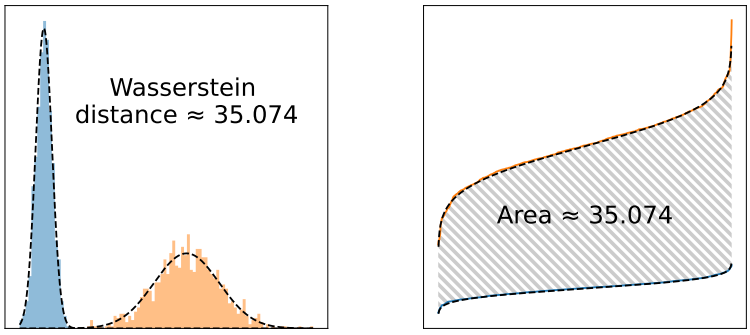
The amount of times these two steps are repeated per batch is decided by a hyperparameter n that is introduced for the method. For each batch, regular training with only the classifier loss is performed, followed by n back and forth repetitions of the *maximizing* and *minimizing* steps. The resulting method outperforms DANN on most standard domain adaptation benchmarks [8].

2.4 Sliced Wasserstein Discrepancy

Inspired by the contemporary success of applying the Wasserstein distance to generative adversarial networks [21], *sliced Wasserstein discrepancy* (SWD) expands on MCD by introducing a new loss function to measure discrepancy between the two classifiers [22].

The Wasserstein distance has its origins in transportation theory. Within the field of computer science it is often referred to as the earth mover's distance, making reference to an informal analogy that considers the problem of finding the minimum distance an earth mover would have to travel in order to transform a starting pile of dirt into a destination pile. More generally, the measure is not restricted to problems going from a single "pile" to another, it can be computed for any two probability distributions.

While computing the true Wasserstein distance is intractable for practical use in



(a) blue: $\mathcal{N}(-10, 2^2)$, orange: $\mathcal{N}(25, 8^2)$

(b) Inverse CDFs of the two normal distributions.

Figure 2.3: A toy example illustrating how the one-dimensional Wasserstein-1 distance can be computed. The blue and orange normalized histograms in 2.3(a) represent 1000 samples from two normal distributions. In 2.3(b), an approximation of the two inverse cumulative distribution functions is retrieved by sorting each set of samples. In accordance with 2.2, the area between the two curves corresponds to the Wasserstein-1 distance. The dashed curves indicate each respective ideal function.

a loss function, *sliced Wasserstein discrepancy* substitutes it with an approximation that computes the Wasserstein distance for multiple one-dimensional problems that are created by projecting down all sample points to a set of stochastically chosen lines. These projections are what is referred to as *slices* by the original authors [23], giving the method its name. Critically, computing the one-dimensional Wasserstein distance for two distributions with N samples each is a special case that can be solved in $O(N \log(N))$ operations by using any ubiquitous sorting algorithm, such as Quicksort. This improves on the more general linear programming formulation which typically require $O(N^{2.5} \log(N))$ operations to reach a solution [24].

The equation for computing this one-dimensional Wasserstein-1 distance is as follows:

$$W_1(\mu_1, \mu_2) = \int_{\mathbb{R}} |F_1^{-1}(x) - F_2^{-1}(x)| \, dx \tag{2.2}$$

The terms $F_1^{-1}(x)$ and $F_2^{-1}(x)$ refer to the inverse cumulative distribution functions (CDF) that are retrieved from the respective sorting process of the source and target samples. A toy example illustrating the inverse CDFs connection to Wasserstein distance is shown in figure 2.3.

By building on these concepts, *sliced Wasserstein discrepancy* defines a loss function that provides a more nuanced measure of how two output distributions differ from one another, yielding a slight improvement in performance on domain adaptation benchmarks when compared to standard MCD [22].

3
Methods

3.1 Overview

In order to keep the online experiments within a reasonable time duration, we need to narrow down which neural network/domain adaptation combination is of particular interest. Once a suitable combination has been selected, we can compare it with baselines through an online experiment with participants to verify that the hypothesized improvements that domain adaptation provide carry over to a real-world system. In Section 3.2, we present the candidates and procedures that led to the final selection of neural network architecture and domain adaptation method. Following this in Section 3.3, we show how the selected combination is used in an online experiment along with a description of the setup used to collect data from participants.

3.2 Offline experiments

Research on applied deep learning has seen an surge in popularity as of late, generating a wide variety of material that describes techniques for improving the performance of neural network models on various tasks across multiple fields. When looking for an appropriate neural network model, we recognise that it would be unfeasible to exhaustively test every imaginable option and choose to instead focus on a subset of models that have seen previous success when applied to MPR. Naturally one could optimize further, but for the purpose of evaluating whether or not domain adaptation helps in an online test scenario we expect this simplified set of candidates to suffice.

This leaves us with models taken from previous MPR research; *Feed-Forward Neural Networks* (FFNN), *Convolutional Neural Networks with Squeeze and Excitation* (CNNSE), and *Temporal Convolutional Networks* (TCN) along with a model from deep learning research on generic time-series that appeared promising: the TSEncoder from *Ts2Vec* [25].

For optimization, Adam was used across all offline evaluation runs with a constant learning rate of 0.001 and no weight decay. All "source only" models were trained for 50 epochs and the domain adaptation methods were trained for 5000 epochs, both trained without any form of early stopping. We suspect the domain adaptation models could be trained an order of magnitude faster with some more

hyperparameter tuning, however the methods do optimize for more objectives than minimization of a single classification loss, and this is also a suspected reason for them requiring more epochs to show any resemblance of convergence.

3.2.1 Preparation of offline data

The chosen development data set CapgMyo DBB records data with a 128-channel HD-sEMG setup. Because of this, the channels in the dataset are downsampled to match the 8-channel electrode setup illustrated in Figure 3.2. For each of the eight 8x16 electrode arrays used in CapgMyo, data from only a single longitudinal pair of the centermost electrodes are used (the rest is discarded). In all training procedures, the data is normalized with a mean and standard deviation computed from the *source data only*, mimicking the "typical" procedure in which normalization parameters acquired during training are reused at test time.

The authors of CapgMyo provide raw and pre-processed variants of the data sets, the latter are used for all offline experiments in this work. The pre-processing consists of cropping out the transients during the start and end of movements in order to focus on the part in which a static hand pose is held (each sample lasting a second, or 1,000 timestamps). A band-stop filter (45-55 Hz, second-order Butterworth) is also used to remove power-line interference.

3.2.2 Selection of neural network architecture

By training all of the candidate neural network models on the development dataset, we retrieve baseline accuracy results of the models when presented with a domain adaptation task. An accuracy score is computed by averaging runs across ten different source/target combinations and ten different pseudo-random number generation seeds, resulting in a total of 100 samples for each network architecture. These results allow us to make a more informed decision on which models are likely to perform well when paired with a domain adaptation method.

With the neural network models selected, we pair them with DANN and SWD. The pairings are trained to classify target domain samples with access to labeled source data and unlabeled target data. The same 100 dataset and seed combinations used for network selection are reused for this step.

TSEncoder is subsequently used for the online experiment, a detailed view of the layers that make up its structure can be seen in Figure 3.1. Diagrams of the rest of the evaluated models can be found in Appendix A.

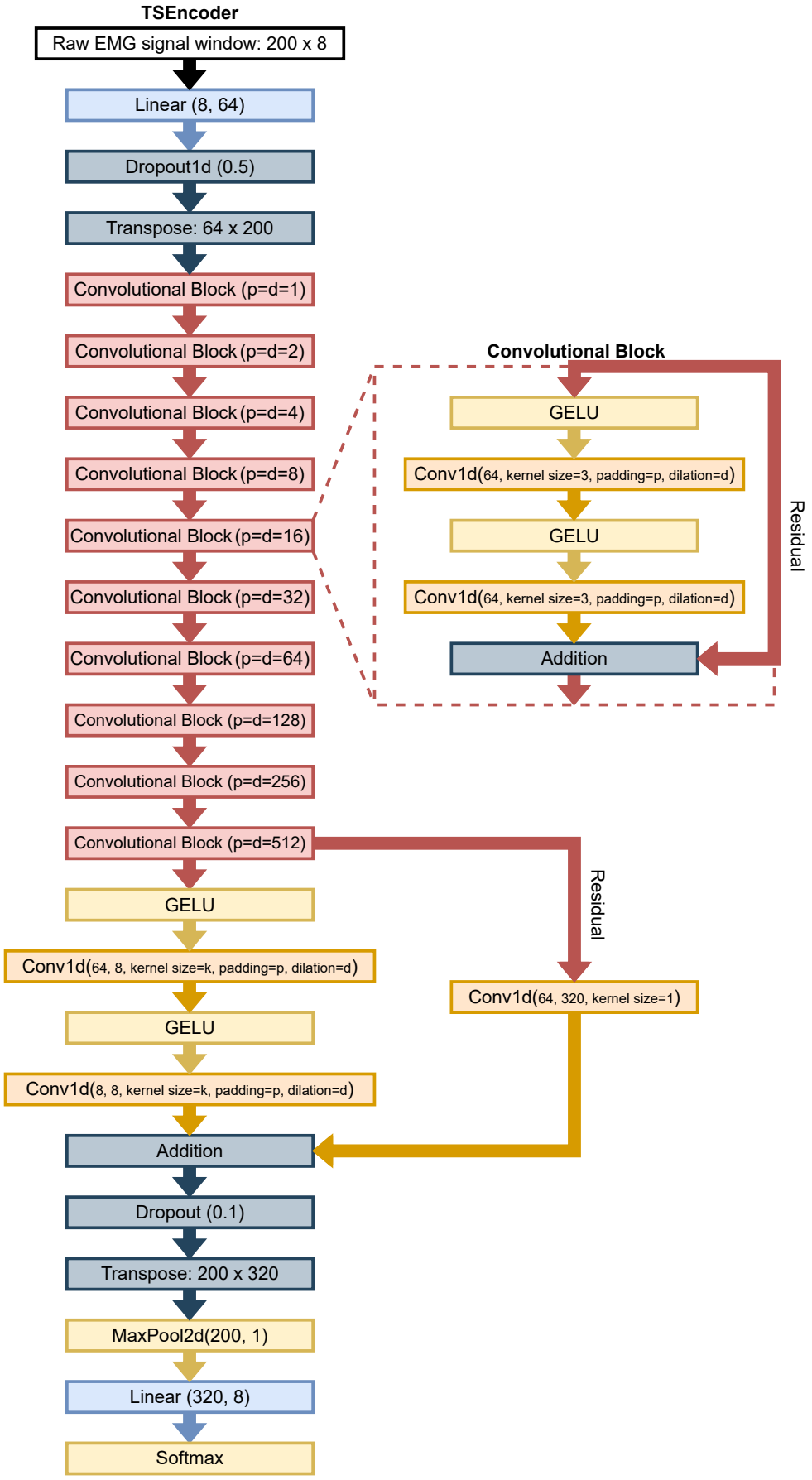


Figure 3.1: Diagram showing the structure of TSEncoder.

3.3 Online experiments

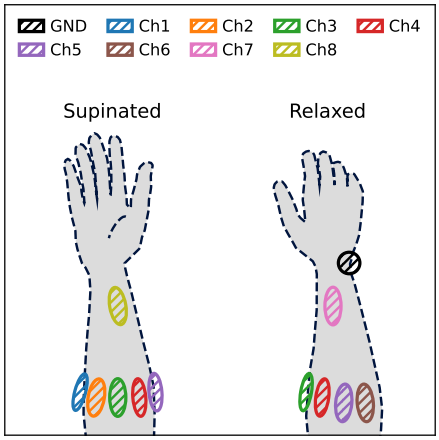


Figure 3.2: Illustration of dipole electrode placement and the poses that make up the two domains during online motion tests. Note that in some cases the same electrode site appears in both views.

For online testing, participants are fitted with two dipole electrode pairs on the front and back of the wrist along with six pairs equally distanced in a semi-circle across the upper forearm and a single electrode ground reference attached to the skin above the wrist bone (illustrated in Figure 3.2). If the electrodes are to come loose at any point during the experiment the participants entire session is invalidated and thus not considered in any further analysis.

Unlike the inter-session domain shift in CapgMyo DBB that is caused by a jump in time between subsequent recordings from a single subject, we induce a domain shift by comparing a source set of sEMG data recorded with a relaxed pose (palm down) versus a target set of sEMG data in a supinated pose (palm up). The idea is that this domain shift will be more similar to the "awkward" limb positions in which an EMG controlled prosthesis begins to twitch or otherwise misbehave (e.g. when reaching for a high shelf as mentioned in the introduction). Similar signs of misclassification in extreme limb positions were observed by Côté-Allard et al. [16] when using a virtual reality (VR) setup to simultaneously track the position of the hand when doing EMG gesture classification.

Prior to the motion tests, participants are asked to perform gestures for the recording of two training data sets; the first with a relaxed wrist position (palm facing down) and the second with a supinated wrist (palm facing up, see Figure 3.2). The gestures performed are as follows: thumb extend, thumb flex, index extend, index flex, middle/ring/little extend, middle/ring/little flex, and rest, making up a total of 7 gestures representing 3 degrees of freedom.

The recorded source and target datasets are used to train three classifiers. The first classifier only uses labeled source dataset for training, acting as a reference

to what performance may typically look like. The second classifier trains on both source and target datasets to act as a baseline for a best case scenario in which labeled data from the target domain is available. The third classifier uses labeled source data and unlabeled target data together with our domain adaptation method to compare its online performance to the other two cases.

Each participant performs motion tests (as described by Kuiken et al. [26]) in the relaxed and supinated wrist position for the three different classifiers, the order of which is randomized (and remains undisclosed to the participant) to remove any possible bias stemming from the participants getting accustomed to the system and to subvert the expectation that latter models will bring improvements.

3.3.1 Participant information

The motion test procedure was recorded for 11 able-bodied subjects with one participant being excluded from analysis due to gradual loosening of surface electrode contact throughout the test.

The study was carried out in accordance with the declaration of Helsinki. Signed informed consent was obtained from each participant before conducting the experiments. The study was approved by the Regional Ethical Review Board in Gothenburg (Dnr. 2022-06513-01).

4

Results

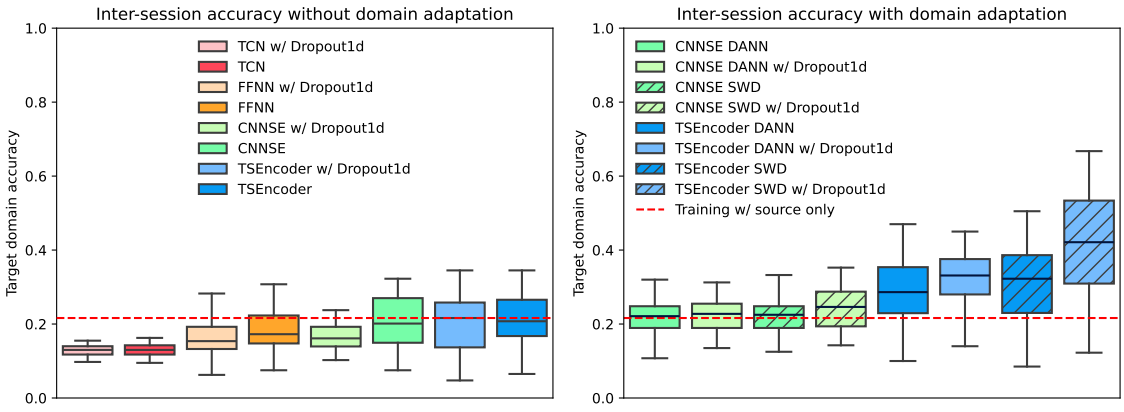
4.1 Offline experimental results

Throughout the offline experiments it was found that the inclusion of a binomial timestamp masking layer in TSEncoder (analogous to applying PyTorch's Dropout1d across all input channels with a probability of 0.5) made a significant impact on its domain adaptation performance. For this reason, all models were retroactively evaluated with and without timestamp masking layers to see if the effect carried over to models other than TSEncoder. Curiously enough, the timestamp masking layer only appears to have a significant impact in the final combination of TSEncoder/SWD where its effect was first discovered (as can be seen in the right panel of Figure 4.1).

For the rest of the models, baseline accuracy on the CapgMyo DBB inter-session domain adaptation task appears to rank them in an order that roughly follows the size and depth of the underlying neural network (left panel in Figure 4.1).

Once CNNSE and TSEncoder were picked out as the most promising candidates for further study, the possible combinations of the models together with the domain adaptation methods DANN and SWD were evaluated. The combinations showed a result on par or better than the best accuracy baseline observed in the previous experiment (right panel in Figure 4.1).

The top scoring combination of base model and domain adaptation method (TSEncoder/SWD) displays an apparent jump in accuracy within the context of this work, but it falls within the range one may expect from the CapgMyo DBB offline performance reported in previous research [14]. It is also worth noting that we only use one eighth of the available channels (due to the down-sampling) when comparing to other works.



(a) Training various models with source domain samples only

(b) Training selected models with domain adaptation and unlabeled target samples

Figure 4.1: Left: experimental results showing target domain accuracy when training all the considered neural network architectures on source domain samples only. Source and target domains correspond to different recording sessions from the same subject in CapgMyo DBB (adjacent pairs 001/002, 003/004, etc. up until 019/020). **Right:** Target domain accuracy comparison for the different pairings of neural networks (with and without timestamp masking) and the two evaluated domain adaptation methods, DANN and SWD. The dashed line represents the highest median accuracy achieved with "regular" source only training (corresponding to TSEncoder with "Dropout1d" i.e. timestamp masking).

4.2 Experimental results from motion tests

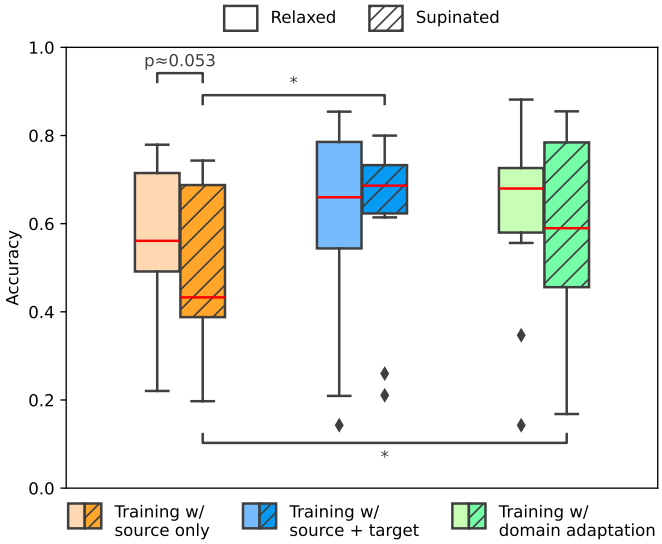


Figure 4.2: Motion test accuracy on both the source (**relaxed**, blank boxes) and target (**supinated**, striped boxes). Statistical significance is marked with a single star (*), indicating a p-value below 0.05. In the supinated case, training with domain adaptation (**green** boxplot) shows a significant improvement over training with source samples only (**orange** boxplot) without compromising performance in the source domain. Training with labels from both source and target domains (shown in **blue**) is used as a baseline to illustrate a best case scenario.

For the online tests we found that the classifier trained only on source (relaxed) domain samples (highlighted with shades of orange in Figure 4.2) performed as expected when participants did motion tests in the relaxed domain (palm facing down). However, a near significant degradation ($p \approx 0.053$) in accuracy occurs when participants switch to doing the motion test in the supinated domain (palm up).

This decrease in accuracy can be mitigated through the use of labeled samples originating from both domains during training (i.e. all of the data recorded from the patient before motion tests). Accordingly, the classifier that trained using all of the available data (highlighted with shades of blue in Figure 4.2) show improvements in accuracy to the point where there no longer is any hint of significant difference between the two domains. While it would be ideal to cover all imaginable scenarios in the training data, recording such datasets quickly becomes infeasible for practical use cases, as illustrated by the doubled amount of labeled data required to improve performance in this relatively contrived experimental setup. The third and final classifier that was trained using domain adaptation (highlighted with shades of green in Figure 4.2) appeared to strike a middle-ground between the other two classifiers, showing a less pronounced increase in performance on the supinated domain when compared with the classifier using all of the labeled data (blue). However, the classifier using domain adaptation significantly improves accuracy over the "source

only" training case (orange) by using unlabeled target data which is far cheaper to collect for practical scenarios than a fully labeled dataset.

All reported p-values were based on the Wilcoxon signed-rank test, which tests the null hypothesis that the two sets of samples originate from the same underlying distribution. Tests resulting in p-values below 0.05 are marked as significant with a star (*) in Figure 4.2.

4.3 Discussion

This study originally posed the question whether or not UDA could be used to improve the online performance of MPR and by extension its application to robotic prostheses. The results show that the reported benefits of using UDA for MPR, previously only explored through offline experiments, also carry over to online use.

The online results also appear to hint at a positive relationship between model/-dataset size and performance for deep MPR, which is in line with contemporary deep learning research. A reason for MPR research being relatively slow at acknowledging the need for larger scale when using deep learning methods may be that the benefits are not as apparent when changes are applied in isolation, for instance when using a larger network without also increasing the size of the dataset. In this sense, UDA may serve as a catalyst for further improvement by providing a realistic path towards larger datasets. These larger datasets could then justify another round of evaluation of MPR with larger networks than what is used in clinical application today, similar to what has been done for CNNSE and TCN.

Regarding the offline results, the discovered combination of TSEncoder and SWD domain adaptation does not show any improvement over the state of the art, suggesting that many other previously proposed methods could lead to similar results if they were evaluated through online experiments. However, with the tools used during the creation of this work we were unable to reproduce the original results reported for UDA on CapgMyo DBB [14]. On the topic of limitations, this study is limited to an online experiment that only tests a single cause of domain shift. It is still an open question to see if UDA can be equally effective when presented with other causes of domain shift, such as EMG data domains stemming from different people. It is also unknown whether or not an identical experiment that uses simultaneous classification (multilabel classification) would yield positive results. Limiting ourselves to multiclass classification for this work was a simplification made in order to not deviate too far from already established UDA works.

In future work, it could be interesting to evaluate other methods alongside the one proposed in this work in a context where they all can be faithfully reproduced and tested on identical data. It could also be a good idea to proceed with experiments where the participants are active users of robotic prostheses, in order to make the study more closely align with the clinical applications we are aiming to improve with this research.

Despite only being a proof of concept, this study provides further confirmation that UDA for MPR may be a promising direction for future research. We anticipate that what has been shown here only represents a brief look at the improvements one may reasonably expect from effectively applying UDA to MPR.

5
Conclusion

Through online testing with the motion test protocol, this work has shown that the use of domain adaptation leads to a significant increase in online performance when a domain shift is caused by wrist supination. Despite being a somewhat contrived case of domain shift, the results show promise for the future application of domain adaptation to train an MPR system that better handles limb positions not present in a labelled EMG recording session. By relaxing the specific conditions at which an MPR system functions correctly, the burden of replicating precise EMG signals is decreased. For a patient using an MPR-controlled robotic prosthesis, this may unlock a whole new set of movements that previously weren't comfortable to perform due to unwanted reactions in the prosthesis.

Bibliography

[1] Ulysse Cote-Allard, Gabriel Gagnon-Turcotte, Angkoon Phinyomark, Kyrre Glette, Erik J. Scheme, Francois Laviolette, and Benoit Gosselin. Unsupervised Domain Adversarial Self-Calibration for Electromyography-Based Gesture Recognition. *IEEE Access*, 8:177941–177955, 2020.

[2] B. Hudgins, P. Parker, and R.N. Scott. A new strategy for multifunction myoelectric control. *IEEE Transactions on Biomedical Engineering*, 40(1):82–94, 1993.

[3] K. Englehart and B. Hudgins. A robust, real-time control scheme for multifunction myoelectric control. *IEEE Transactions on Biomedical Engineering*, 50(7):848–854, 7 2003.

[4] M.A. Oskoei and Huosheng Hu. Support Vector Machine-Based Classification Scheme for Myoelectric Control Applied to Upper Limb. *IEEE Transactions on Biomedical Engineering*, 55(8):1956–1965, 8 2008.

[5] Wei Li, Ping Shi, and Hongliu Yu. Gesture Recognition Using Surface Electromyography and Deep Learning for Prostheses Hand: State-of-the-Art, Challenges, and Future. *Frontiers in Neuroscience*, 15:259, 4 2021.

[6] Ricardo V. Godoy, Anany Dwivedi, and Minas Liarokapis. Electromyography Based Decoding of Dexterous, In-Hand Manipulation Motions With Temporal Multichannel Vision Transformers. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 30:2207–2216, 2022.

[7] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for Data: Ground Truth from Computer Games. 8 2016.

[8] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum Classifier Discrepancy for Unsupervised Domain Adaptation. 12 2017.

[9] Xiaofeng Liu, Chaehwa Yoo, Fangxu Xing, Hyejin Oh, Georges El Fakhri, Je-Won Kang, and Jonghye Woo. Deep Unsupervised Domain Adaptation: A Review of Recent Advances and Perspectives. 8 2022.

[10] Jianwei Liu, Xinjun Sheng, Dingguo Zhang, Jiayuan He, and Xiangyang Zhu. Reduced Daily Recalibration of Myoelectric Prosthesis Classifiers Based on Domain Adaptation. *IEEE Journal of Biomedical and Health Informatics*, 20(1):166–176, 1 2016.

[11] Max Ortiz-Catalan, Bo Håkansson, and Rickard Brånemark. An osseointegrated human-machine gateway for long-term sensory feedback and motor control of artificial limbs. *Science Translational Medicine*, 6(257), 10 2014.

[12] Istvan Ketyko, Ferenc Kovacs, and Krisztian Zsolt Varga. Domain Adaptation for sEMG-based Gesture Recognition with Recurrent Neural Networks. *Pro-*

ceedings of the International Joint Conference on Neural Networks, 2019-July, 1 2019.

[13] Patrick P. K. Chan, Qiuxia Li, Yinfeng Fang, Linyi Xu, Kairu Li, Honghai Liu, and Daniel S. Yeung. Unsupervised Domain Adaptation for Gesture Identification Against Electrode Shift. *IEEE Transactions on Human-Machine Systems*, 52(6):1271–1280, 12 2022.

[14] Yu Du, Wenguang Jin, Wentao Wei, Yu Hu, and Weidong Geng. Surface EMG-Based Inter-Session Gesture Recognition Enhanced by Deep Domain Adaptation. *Sensors 2017, Vol. 17, Page 458*, 17(3):458, 2 2017.

[15] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting Batch Normalization For Practical Domain Adaptation. 3 2016.

[16] Ulysse Cote-Allard, Gabriel Gagnon-Turcotte, Angkoon Phinyomark, Kyrre Glette, Erik Scheme, Francois Laviolette, and Benoit Gosselin. A Transferable Adaptive Domain Adversarial Neural Network for Virtual Reality Augmented EMG-Based Gesture Recognition. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 29:546–555, 2021.

[17] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. PyTorch Adapt. 11 2022.

[18] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2 2015.

[19] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How Does Batch Normalization Help Optimization? 5 2018.

[20] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. 5 2015.

[21] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. 1 2017.

[22] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced Wasserstein Discrepancy for Unsupervised Domain Adaptation. 3 2019.

[23] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein Barycenter and Its Application to Texture Mixing. pages 435–446. 2012.

[24] Rainer Burkard, Mauro Dell'Amico, and Silvano Martello. *Assignment Problems*. Society for Industrial and Applied Mathematics, 1 2012.

[25] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. TS2Vec: Towards Universal Representation of Time Series. *Proceedings of the 36th AAAI Conference on Artificial Intelligence, AAAI 2022*, 36:8980–8987, 6 2021.

[26] Todd A. Kuiken, Guanglin Li, Blair A. Lock, Robert D. Lipschutz, Laura A. Miller, Kathy A. Stubblefield, and Kevin B. Englehart. Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms. *JAMA*, 301(6):619–628, 2 2009.

A

Appendix 1

A.1 Neural network diagrams

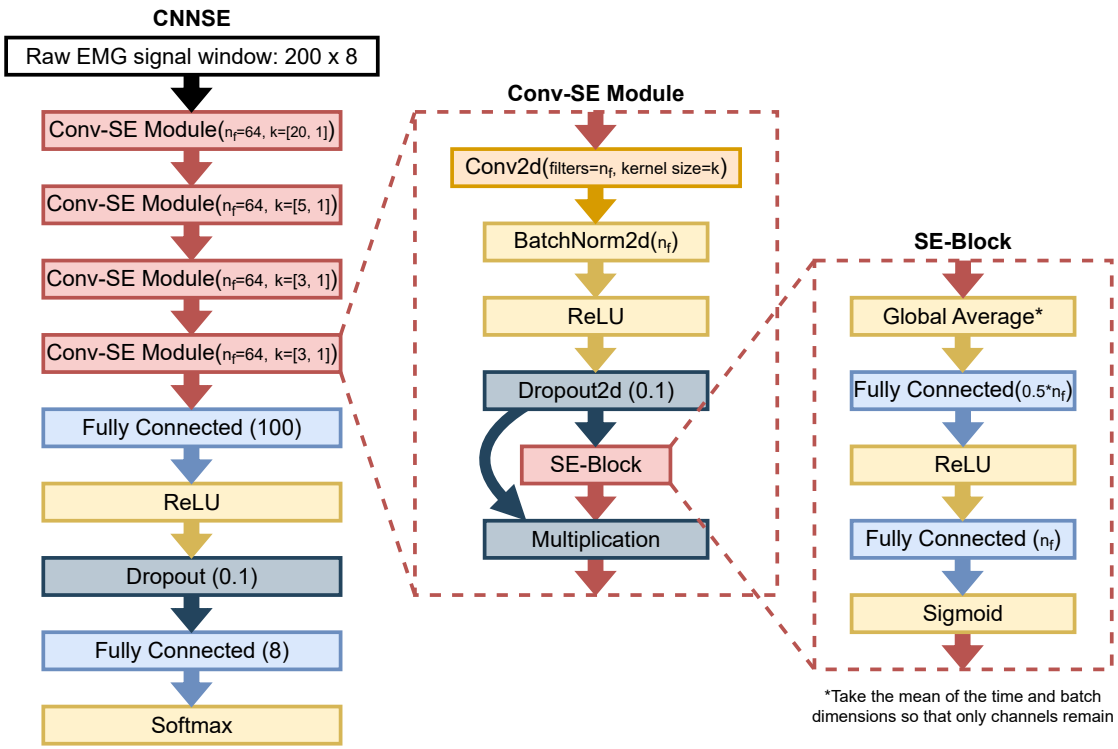


Figure A.1: Diagram showing the structure and various parameters used for CNNSE in the experiments.

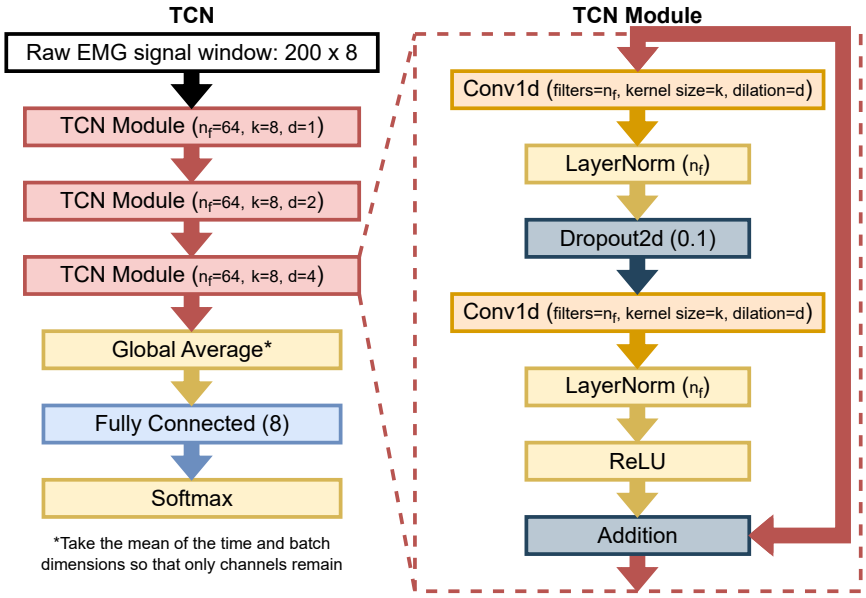


Figure A.2: Diagram showing the structure and various parameters used for TCN in the experiments.

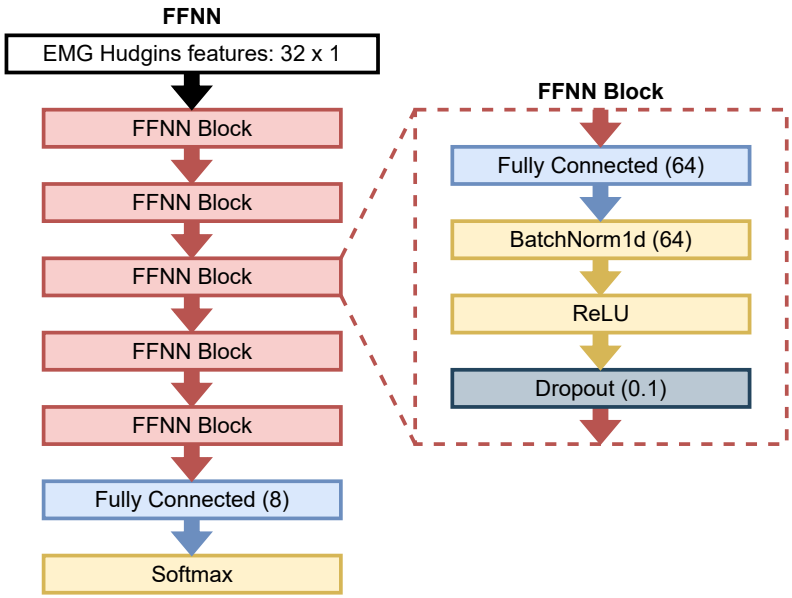


Figure A.3: Diagram showing the structure and various parameters used for FFNN in the experiments.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY