Semester Project

Deep learning for inter-hardware EMG compatibility

Niederhauser Thibault
Professor: Micera Silvestro
Supervisor: Mendez Vincent

École Polytechnique Fédérale de Lausanne
Translational Neural Engineering Lab

January 2021

Contents

1. Introduction

The growing development of neuroprosthetics and robotics has allowed great progress in the design of hand prosthesis for amputees. A key aspect that determines the ease of use of such devices is the control method, which need to allow the users to easily and accurately control the robotic hand. A promising method for this purpose is myoelectric control [1]. This technique consists of recording surface electromyography (EMG) signals on the patients' forearm and using this data to decode the intended hand gesture. As the natural muscular activity related to hand motion is recorded by the EMG-electrodes, intuitive control of the prosthesis is enabled.

A widely used technique for EMG recording is the placement of 6 to 8 electrodes on the forearm of amputee patients. The state-of-the-art decoding approach applies pattern recognition algorithms on extracted features from EMG signals to retrieve patients' motor intentions. Such methods can be used for grasp classification, single-finger classification or wrist motion regression [1].

However, the recent tendency in EMG decoding is to move from feature engineering to feature learning using deep-learning (DL) models, that learn and extract the relevant features from raw data and then perform classification or regression tasks. Recent studies suggest that DL-based methods for EMG decoding outperform the conventional pattern recognition approaches [2], [3].

Nonetheless, the major drawback of DL-based approaches is that they require a large amount of training data. In the case of myoelectric control of hand prosthesis, the EMG data needs to be generated by the patients, therefore the recording of large single-user datasets are made impossible as they would consume to much time. A way to tackle this problem is to use transfer learning algorithms on data acquired by different subjects. Indeed, recordings of several users can be exploited to improve the classification or regression performances on an other patient's recording [4] [5]. However, even though transfer-learning has been proven useful, research in this field typically makes use of different hardware with varying numbers of electrodes, from 6 bipolar channels to 128 mono-polar electrode grids. Since the number of electrodes define the input shape, data from different hardware can not be used by the same DL regressor or classifier, which limits the potential of transfer-learning on EMG data.

The aim of this work is to show that the information learnt on 6-electrode data can be leveraged to improve the performance on 8-electrode data in a regression task. The proposed approach consists of two main steps: first simple dimension reduction and augmentation techniques are implemented in order to obtain same-size data from both hardware. Then the resized data is used as input into a convolutional neural network (CNN) that performs single-finger angle regression. Two different CNN architectures are developed and evaluated. The data from 6-electrode hardware is used to pre-train the CNN, with the aim of achieving better performance on 8-electrode data in a second time.

2. Background

2.1 EMG decoding

Conventional machine learning methods are the state-of-the-art approach to decode recorded EMG signal of the forearm [6]. Such techniques imply that the features are extracted manually (feature engineering) from the EMG. The extracted features are then fed into machine learning models to perform grasp classification (K-Nearest Neighbors [7], Support Vector Machine [8], Linear Discriminant Analysis [9], Multilayer Perceptrons [10], etc.) or single finger angle regression (using non-negative matrix factorization [11], multilayer perceptrons, non-parametric Gaussian process [12], etc.).

In recent years, a paradigm shift from feature engineering to feature learning using DL-based approaches has been observed, as DL enable better classification (respectively regression) performances. Indeed, recent studies showed that deep CNNs [3] and hybrid architectures involving recurrent neural networks (RNN) and convolutional neural networks [13] were able to outperform conventional methods for grasp classification based on EMG data. Furthermore, Ameri et al. developed a CNN architecture for single-finger regression that showed better results than state-of-the-art approaches [2].

Consequent difficulties emerge from *inter-session* (the model is trained on one/some recording session-s and tested on another session) and inter-user (the model is trained on one/some user-s sessions and tested on another user) EMG recordings. Indeed, EMG signals are affected by many factors such as user's physiology, electrode shifts or muscle fatigue. Consequently, the recorded EMG might differ a lot from one session to another, which complicates the decoding. Transfer-learning approaches using convolutional neural networks were proven useful to tackle this issue. As a matter of fact, transfer-learning architectures showed satisfying performances on *inter-session* and *inter-user* trials for grasp classification [4], as well as for single-finger angle regression [5]. Transfer-learning methods consist of using a large amount of other-session data to pre-trained a model and, in a second time, to fine-train this model on the desired session. This way, the model does not need to be trained from zero on the desired session, but benefits from the learned weights during the pre-training phase.

2.2 Siamese Convolutional Neural Networks

Siamese Convolutional Neural Networks (SCNNs) is a specific type of CNN architecture that finds applications in signature verification and face identification [14]. A siamese network is a set of two sub-networks that share some common weights. In the specific case of face identification, during training, the two sub-nets take two face images as input, map the images into a low-dimensional space where the distance between different faces is maximized using a so-called constrastive loss. This concept generalizes to other kind of input and siamese neural networks can be seen as networks trained to map inputs into a vector space where the distance between similar classes is minimized and the distance between different classes is maximized. After training, such networks can be used either for identification (feed the new input into one sub-network and compute to which

class it belongs based on the distance to known classes in the low-dimensional space) or verification (feed a known input into one sub-network and the input to be verified into the second sub-network. Whether the two inputs belong to the same class can be extracted from the distance between the two output vectors).
A great advantage of siamese neural networks is that they enable highly efficient transfer learning. Indeed, a Siamese Network is trained on data to output similarities or differences between two inputs and does not need to be retrained from scratch to integrate a new class (for example, a face belonging to new unknown person). To this extent, Siamese Networks were successfully used as a tool for transfer learning for face recognition on small datasets [15] and speech emotion recognition [16]. Furthermore, SCNNs were shown to enable one-shot image recognition [17] and speaker identification [18].

However, note that the previously mentioned SCNNs address classification problems, whereas this work deals with a regression problem. Doumanoglou et al. [19] developed a regression Siamese architecture for object pose estimation. In this case, the loss has no longer the purpose of maximizing the distance between different classes, but is used to associate the distances in the feature spaces with the distances in the 3D-pose output space. This approach was shown to elicit faster learning and better pose estimation performances.

3. Methods

3.1 EMG recording protocol

The EMG data is recorded by bipolar surface EMG electrodes with a sampling frequency of 2000 Hz. During acquisition, the signal is analogically notch filtered at 50 Hz and band pass filtered between 5 and 450 Hz. The two tested hardware are 6-channel and 8-channel EMG, therefore 6, respectively 8, electrodes are positioned randomly around the user's forearm in a ring configuration. The random placement of the electrodes highly simplifies the procedure and increases the potential usability of the device, but increases the difficulty of decoding the signal.
During the recording process, the user replicates a sequence of movement performed by a virtual hand displayed on a screen. The movement sequence consist of 6 repetitions of the following hand gestures:

[Flex. Pinky – Rest – Flex. Ring – Rest – Flex. Middle – Rest – Flex. Index – Rest – Thumb opposition – Rest – Ulnar Grasp – Rest – Tripod Grasp – Rest – Pinch Grasp – Rest – Flex. P,R,M,I (thumb up) – Key Grasp (subject closes thumb on index) – Rest – Power Grasp – Rest – Open – Rest]

The obtained data is organised in a "channel xx temporal" form; it consists of c vectors of size t, where c is the number of EMG electrodes and t the recording time. An example of the recorded signal during a 6-channel recording procedure can be seen in Figure 3.1.

Moreover, during the recording, the hand pose is labelled (according to the virtual hand position) with a 6-D vector, whose elements correspond to the 5 fingers' angles and the apposition-opposition of the thumb (see Figure 3.2).

3.2 Transfer learning pipeline

As the aim of this work is to enable inter-hardware EMG compatibility using deep-learning, the transfer-learning pipeline is inspired from the works of Côté Allard et al. [4] and Ameri et al. [5], which propose transfer-learning approaches for inter-session EMG compatibility based on pre-training and fine-training of a model (see section 2.1). Similarly, in this work, convolutional neural networks are pre-trained on 8 different 6-channel recordings and then fine-trained on two separate 8-channel recordings.
Since the data comes from 2 different hardware (6-electrode vs 8-electrode EMG), it has different dimensions. However, the deep-learning models need to be pre-trained and fine-trained with data of the same dimension. Therefore input resizing (dimension reduction or augmentation) is performed as a pre-processing step. Those allow to obtain "6 channels x temporal" or "8 channels x temporal" input data for both hardware (see section 3.3).

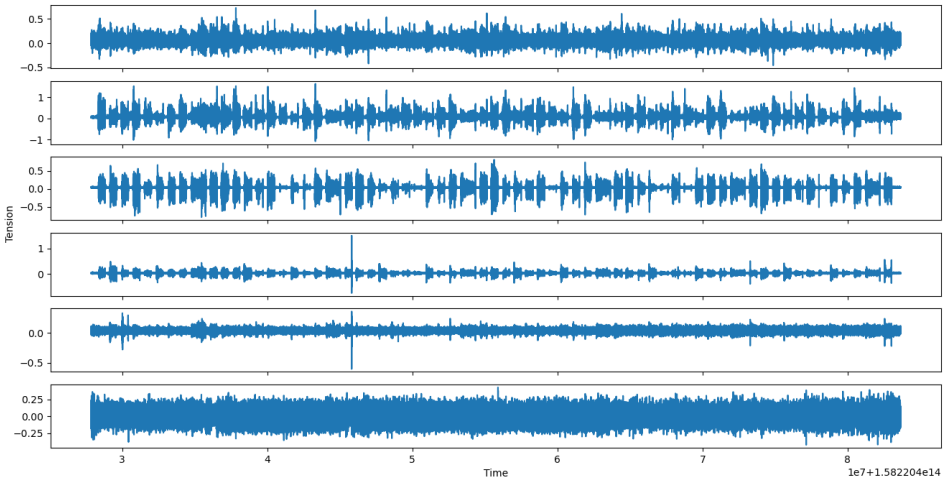


Figure 3.1: 6-channel raw EMG signal

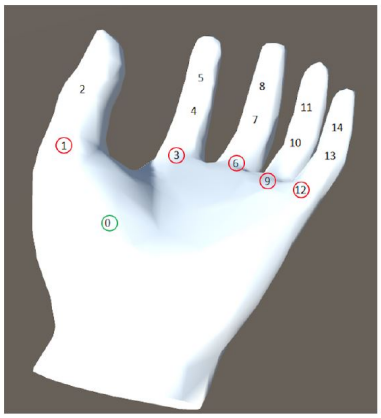


Figure 3.2: Hand pose labels: in red, the 5 angles corresponding to the flexion/extension of the fingers and, in green, the angle corresponding to the opposition-apposition of the thumb.

The input data of the CNNs is are simply reshaped time windows cut from the "6 channels x temporal" or "8 channels x temporal" data obtained after the input resizing steps. In this work two different CNN architectures are tested (see section 3.4). An overview of the transfer learning pipeline is depicted in Figure 3.3. Note that each time, the models are pre-trained on the data of eight 6-channel sessions, using $\frac{1}{6}$ of the data as validation set and $\frac{5}{6}$ as training set. The fine-training is performed using 8-channel data of two separate recordings. For each 8-channel recording $\frac{1}{6}$ of the data is saved as a test set and the three following fine-training approaches are tested:

- Complete fine-training, using the whole session data: $\frac{4}{6}$ of the data as training set, $\frac{1}{6}$ as validation test
- Small data fine-training, using a reduced subset of the session data: $\frac{1}{6} * 0.8$ of the data as training set, $\frac{1}{6} * 0.2$ as validation test. Note that only $\frac{1}{6}$ of the available data is used for the fine-training step.

- No fine-training: $\frac{1}{6}$ of the data is used as test set and directly evaluated with the pre-trained model.

This allows to observe the effect of the amount of fine-training data on the regression performances.

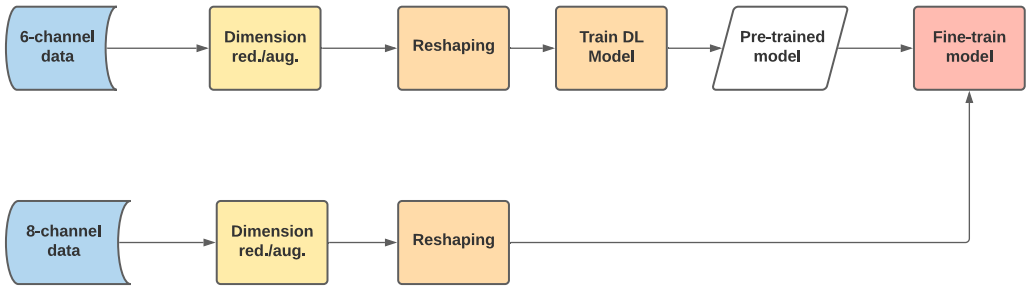


Figure 3.3: Transfer learning pipeline

3.3 Input dimensions resizing

Regardless of which deep-learning model is used for the single finger regression, the sizes of the input fed into the model need to be the same. Therefore, pre-processing steps to bring 6-channel and 8-channel data to the same size are implemented.

A first naive approach to resize the inputs is to duplicate or drop some channels from the recorded data. The two following methods are tested: dropping two randomly chosen channels from the 8-channel data or augmenting the 6-channel data by duplicating 2 channels. The duplication is done as follows: the 6th channel is added in first position and the 1st channel is added in the last position. This augmentation step allows to take into account the ring-like disposition of the EMG-electrodes. However, in order to obtain same input sizes, the augmentation can only be performed on the 6ch-data, which is a conceptual draw-back.

Principal Components Analysis (PCA) is another to reduce the data dimensions and obtain inputs of the same size. In this work, PCA is implemented using the as follows: the channels are considered as the dimensions and the points of the time series are the data points. Hence, for each recording, there exist n data points (n corresponding to the number of recorded time samples) of dimension 6, respectively 8. For both 6-channel and 8-channel data, the 6 first principal components are computed. Finally, the data is projected on the principal components. This allows to obtain a $6 \times n$ vector for both recording hardware and minimizes the amount of lost variance. The PCA algorithm is implemented using the `PCA` function of `sklearn`.

Another candidate to extract same-size inputs from 6-channel and 8-channel recordings, is Independent Component Analysis (ICA) for Blind Source Separation (BSS) [20][21]. BSS through ICA attempts to extract independent non-Gaussian components from a mixed signal perceived by a set of sensors. A state-of-the-art BSS problem tackled by ICA is the "cocktail party problem" [22]. This problem consists of retrieving the individual speeches of speakers talking simultaneously in the same room by using the mixed signal recorded by microphones (senors) in the room.
By analogy, considering the forearm muscular activity as the mixed signal and the EMG-electrodes as the sensors, one can apply ICA to try and extract the independent component of signal and project the mixed signal on them. The obtained signals ideally ideally correspond to single muscle activity signals. ICA can only extract as many independent components as the number of sensors. In this case, since in the worst case,

the number of sensor is 6, 6 independent component are extracted from the time-series. The projection on those vectors result in $6 \times n$ vectors that can be used as input to the neural network. The ICA algorithm is implemented using the `FastICA` function from `sklearn`.

All consistent combinations of input dimensions resizing for pre-training and fine-training are tested and schematized in Figure 3.4.

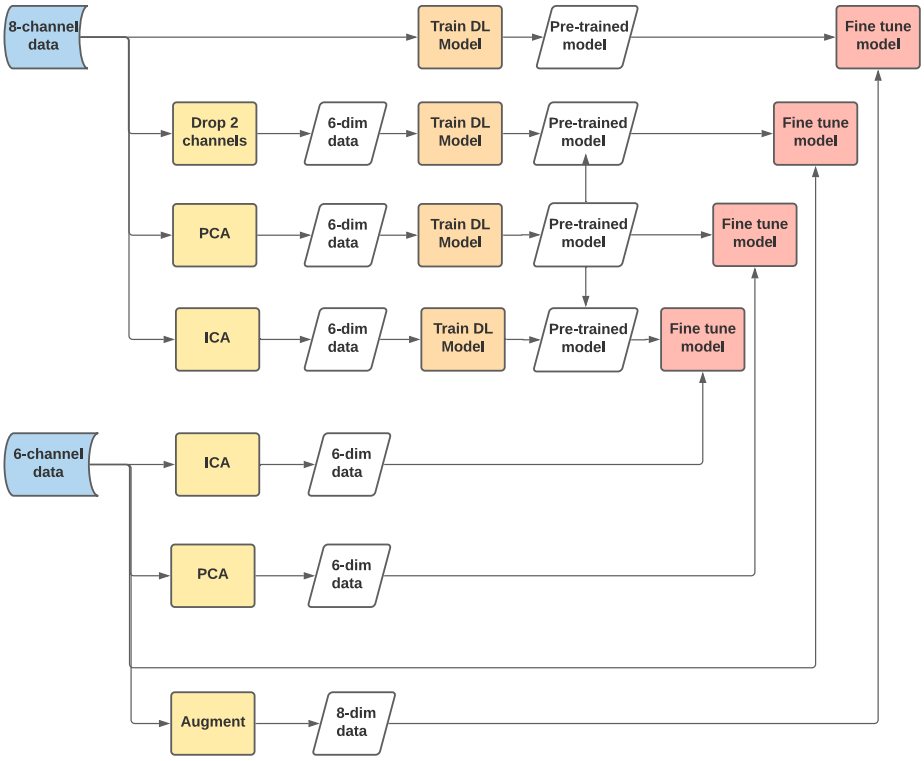


Figure 3.4: Transfer learning pipeline: from 6-channel to 8-channel data

3.4 Deep Learning Models

The purpose of deep-learning models is to map a time-window from the "channel x temporal" EMG data data to single-finger angles. Two different deep learning models are implemented: an Ameri-like convolutional Neural Network and a Siamese Convolutional Neural Network.

3.4.1 Simple Ameri-like CNN

The designed Ameri-like model is a regression convolutional neural network for EMG control inspired by the work of Ameri et al. [2]. The Ameri-like model uses a 3D squared image as input. It is first fed into convolutional layers that are meant to extract the important features from the input. Then a few fully convolutional layers perform regression, i.e map the learnt features to single-finger angles.

The exact architecture used in this work is shown in Figure 3.5 ads is the same as the one propose in the work of last year's student L. Pollina. The optimal parameters of the model were as well found by Pollina using a grid search and are reused in this work (see Figure 3.6).

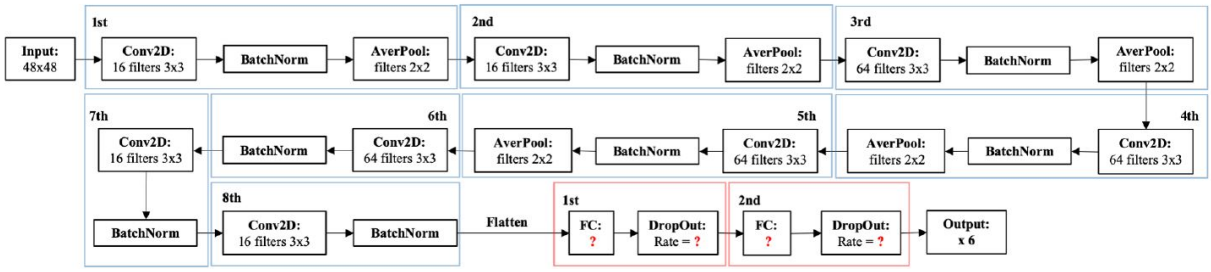


Figure 3.5: Ameri-like model architecture: the input is a squared image of size 48x48. 8 convolutional layers are used for feature extraction and 2 fully convolutional hidden layers are used for regression. The output is a 6-dimensional vector corresponding to 6 joint angles of the hand.

CNN Ameri-like	
Hyper-parameter	**Possible values**
nbr_filters_hid_1	[24,32,48,64,80]
nbr_filters_hid_2	[8,12,16,24,32]
drop_rate	linspace(0,0.2,9)
λ	linspace(1e-7,1e-4,7)

Figure 3.6: Ameri-like model parameters

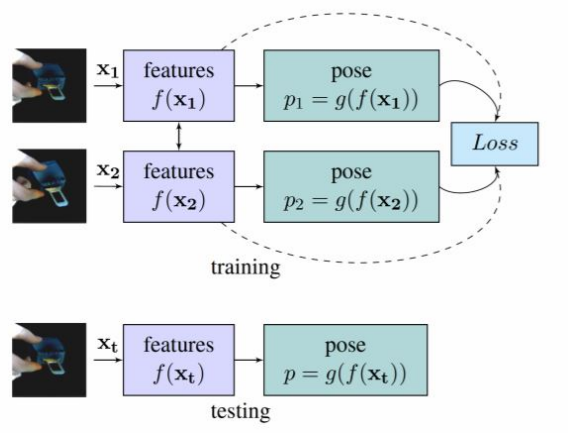
3.4.2 Siamese Ameri-like Network

A regression siamese CNN architecture is developed based one the work of Doumanoglou et al. [19]. The siamese network consists of two identical Ameri-like CNN sharing all their weights. Note that the same pre-processing and input resizing steps as for the simple Ameri-like CNN are applied.
During the training phase, two inputs are fed into the siamese network and both the outputs of the feature extraction layers and the regression layers are used to compute the loss (see Figure 3.7a). A regression loss l_R and a feature loss l_F are defined as follows:

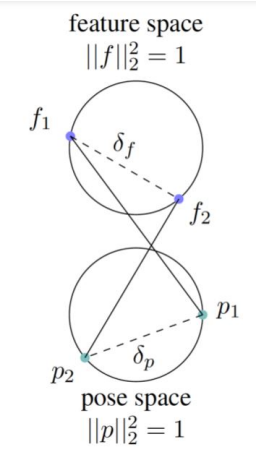
$$l_R = \sum_{n=1}^{K} ||(g(f(x_n)) - y_n||_2^2$$

$$l_F = \sum_{n=1}^{K} \left| ||f(x_{n,1}) - f(x_{n,1})||_2^2 - ||y_{n,1} - y_{n,2}||_2^2 \right|$$

where K is the number of samples in each mini-batch, f is the feature extractor, g is the regressor, $x_{n,i}$ is the input corresponding to the n^{th} sample fed into the i^{th} sub-network and $y_{n,i}$ is the true label of the n^{th} sample of the i^{th} sub-network. Note that $f(x_n)$ corresponds to the extracted features and $g(f(x_n))$ is the predicted label.
l_R is a conventional mean square error and l_F is a custom loss that enforces the distances between the features of two samples to be close the distances between the corresponding ground truth vectors in the pose space.



(a)



(b)

Figure 3.7: Regression Siamese Network: the network is trained using the siamese architecture (a) in order to enforces an mapping between the distances in the feature space and the pose space (b). Finally, one branch of the siamese network is extracted for the testing phase (a). [19]

This results in a correspondence between the distances in the feature space and the pose space (see Figure 3.7b).
Importantly, during training the weights of the feature extraction layers are updated using the loss $l_E = l_R + l_F$, whereas the weights of the regression layers are updated using only the regression loss l_R. Hence at each epoch and for each mini-batch the weight updates are computed as follows:

$$\nabla w_F = \frac{\partial l_R}{\partial w_R} + \frac{\partial l_F}{\partial w_F}$$
$$\nabla w_R = \frac{\partial l_R}{\partial w_R}$$

where w_F and w_R are the weights of the feature extraction and regression layers respectively.

Importantly, note that for a data-set of M samples, $\binom{M}{2}$ possible different pairs can be generated, which would result in an extremely large amount of training data. To avoid dealing with non-viable amount of data, each recorded data point is paired with two other data point chosen randomly. This way the training data size is $2 * M$ and memory or training time issues are avoided.

Finally, note that the Siamese architecture is only used during training and its only purpose is to associate the feature space with the output pose space. After the training phase, one branch of the Siamese Network is extracted and used for testing. The conceptual idea behind this approach is that the mapping between the feature and the pose space learnt during pre-training on the 6-channel hardware data could enable faster learning with fewer data during fine-training on 8-channel recordings.

3.4.3 Input Reshaping

Time-windows cut from "channel x temporal" data are used as input for both the simple and siamese Amer-like models. However, the input of the Ameri-like model is a squared image, therefore the time-windows need

to be reshape as squared images before being sent into the model. To achieve squared dimensions, the time window length needs to be chosen carefully, so that the number of data-points in each window is a perfect square. Knowing from the work of Côte-Allard et al. [4] that the optimal window length should lay between 100 and 250ms and that the sampling frequency is 2000 Hz, time windows of 0.144s and 0.192s were chosen for 8-channel and 6-channel inputs respectively. Indeed:

$$0.192s \times 2000Hz \times 6 = 48^2$$

$$0.144s \times 2000Hz \times 8 = 48^2$$

Once the time-window of the relevant size is cut, the data-points are reorganised in 48x48x1 dimensions (see Figure 3.8) and can then be fed into the Ameri-like model (simple or siamese).

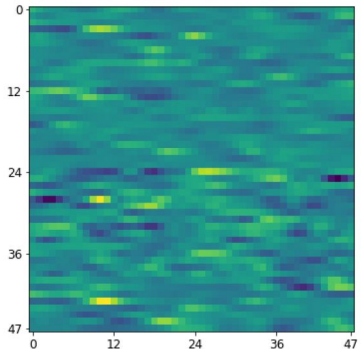


Figure 3.8: Example of an 48x48x1 input

4. Results

4.1 Control Test

In order to assess the potential benefits of the transfer learning methods, one first need to have a sense of the performance that can be reached without transfer learning, only using the data of one session. For this purpose, a control test is performed. For both 8-channel recordings, an Ameri-like model is trained and tested on the same recording data using $\frac{4}{6}$ of the data as the training set, $\frac{1}{6}$ as the validation set and $\frac{1}{6}$ as the test set. The obtained results are reported in Table 4.1

Table 4.1: Control test: same-user losses on the Ameri-like model

User	Hardware	Loss
1	8-channel	**6.53**
2	8-channel	**4.21**

4.2 Simple Ameri-like Model

The test losses using the simple Ameri-like model combined with different input resizing methods are reported in Table 4.2. Figure 4.1 shows the same results in a graphical way and compares the result to the control test. Examples of the obtained regression on the test set can be found in Appendix B.

Table 4.2: Simple Ameri-like model: test losses

Pre-training data	Fine-training/testing data	**No fine-training**		**Small-data fine-training**		**Complete fine training**	
		Loss u1	**Loss u2**	**Loss u1**	**Loss u2**	**Loss u1**	**Loss u2**
6-ch	8-ch + Drop	**15.76**	**15.50**	**13.77**	**9.92**	**6.02**	**5.49**
6ch + Augmentation	8-ch	**18.71**	**18.51**	**16.47**	**7.78**	**5.84**	**5.14**
6ch + PCA	8-ch + PCA	**25.93**	**25.02**	**15.71**	**25.48**	**6.92**	**6.79**
6ch + ICA	8-ch + ICA	**16.02**	**16.10**	**22.5**	**16.15**	**16.25**	**17.85**

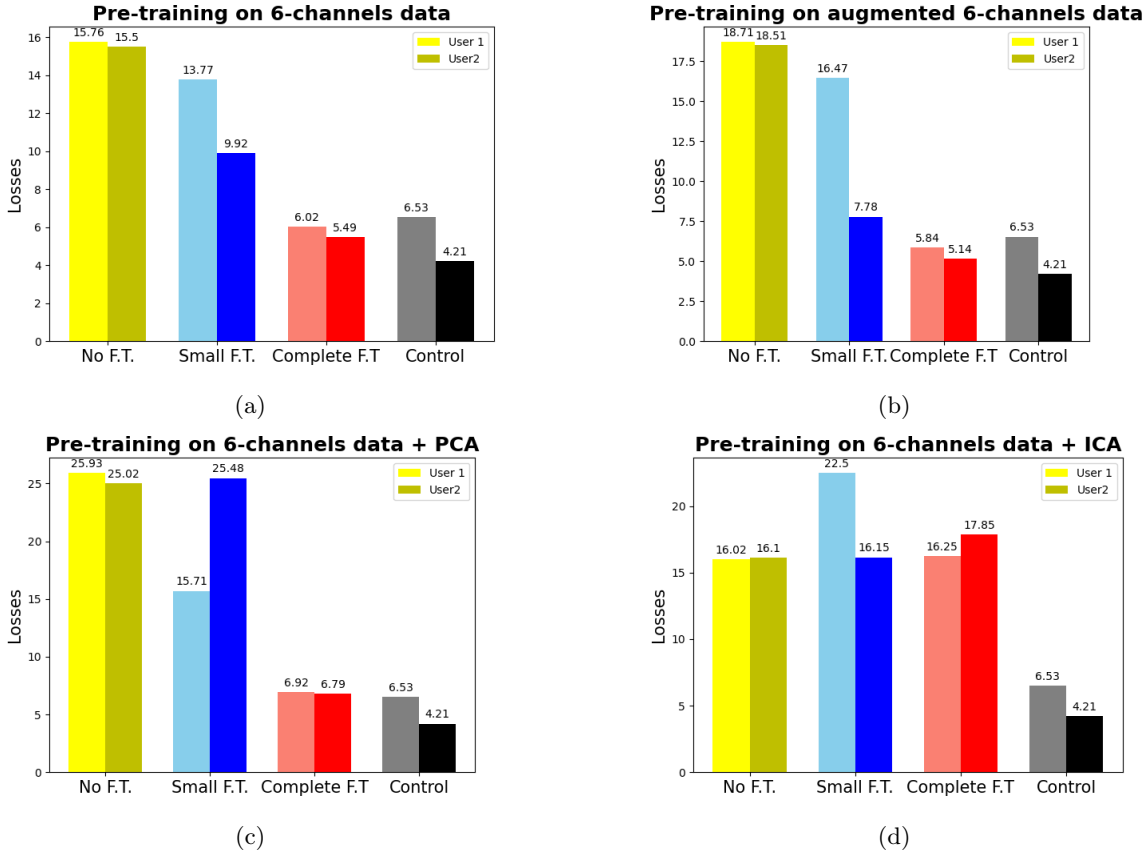


Figure 4.1: Obtained Losses with the simple Ameri-like model

4.3 Siamese Ameri-like Model

The test losses using the siamese Ameri-like model combined with different input resizing methods are reported in Table 4.3. Figure 4.2 shows the same results in a graphical way and compares the result to the control test. Example of the obtained regression on the test set can be found in Appendix B.

Table 4.3: Siamese Ameri-like model: test losses

Pre-training data	Fine-training/testing data	**No fine-training**		**Small-data fine-training**		**Complete fine training**	
		Loss u1	**Loss u2**	**Loss u1**	**Loss u2**	**Loss u1**	**Loss u2**
6-ch	8-ch + Drop	**16.0**	**15.56**	**12.88**	**11.08**	**5.56**	**6.28**
6ch + Augmentation	8-ch	**22.36**	**23.00**	**15.9302**	**7.81**	**5.32**	**5.66**
6ch + PCA	8-ch + PCA	**25.28**	**22.99**	**17.88**	**28.72**	**7.31**	**6.13**
6ch + ICA	8-ch + ICA	**17.13**	**16.65**	**16.38**	**18.04**	**15.90**	**14.44**

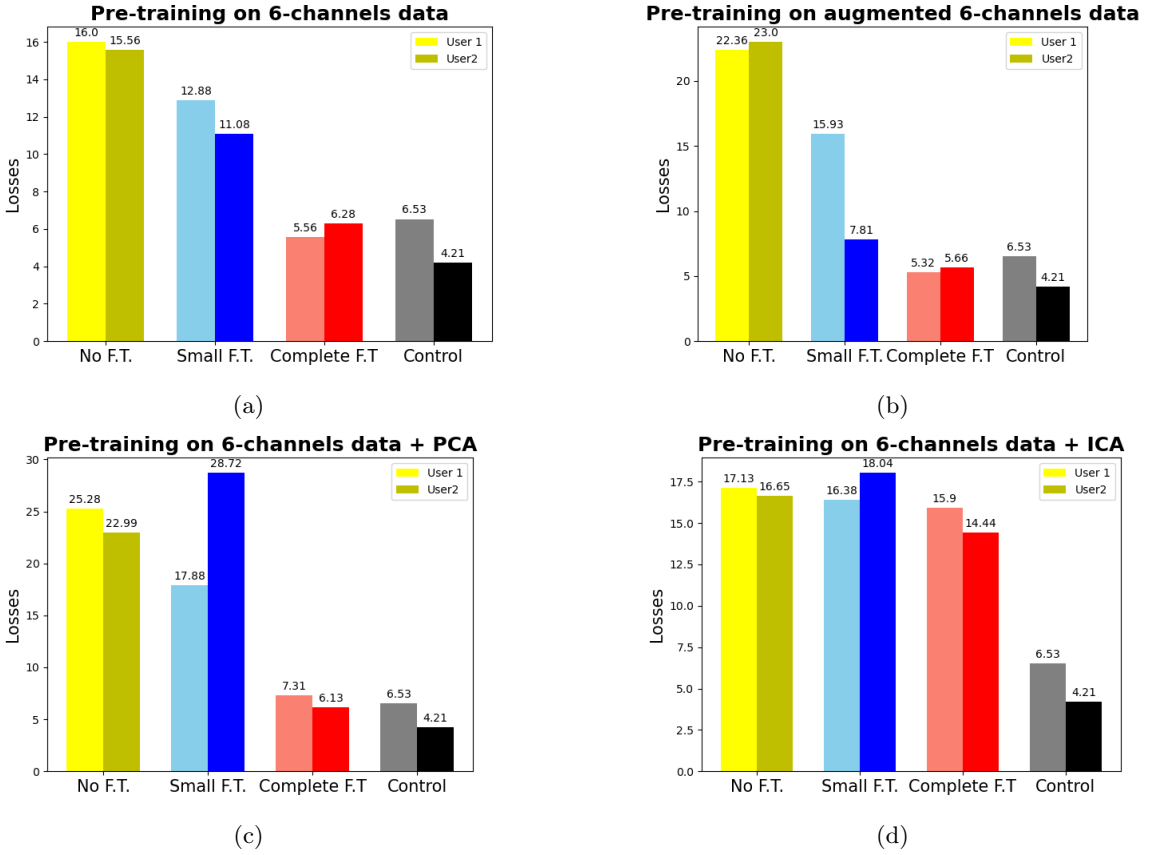


Figure 4.2: Obtained Losses with the siamese Ameri-like model

5. Discussion

5.1 Result interpretation

First of all, one can observe that, except for the ICA approach, all tested methods follow the same tendencies:

- As expected, usng more data for fine-training results in better performances.
- The pre-trained models with no further fine-training show very poor performances. This was to be expected, since the model has absolutely no knowledge of the test-session data and, as explained in section 2.1, the EMG recorded data tend to vary significantly between sessions.
- The performance of all the models fine-trained on the complete available data show similar performances as the control model (trained on complete available data, but without the pre-training steps), which implies that the pre-training did not improve the performances. **Hence, the attempt to implement trans-hardware transfer-learning shows no satisfying results**.

Interestingly, both deep-learning models coupled with ICA as pre-processing step yield poor results, no matter how much data is for the fine-training step. This can be explained for the following reasons:

- ICA for blind source separation (BSS) relies on assumption on the signal that are not true in this specific case. Indeed, for this approach to work the sub-components of the recorded mixed signal need to be non-Gaussian and statistically independent, however it is unsure if the forearm muscular activity fulfills those requirements.
- ICA only allow to extract as many independent components as the number of sensors. Therefore, in this work only 6 independent components are extracted (since the smallest hardware has 6 recording electrodes). However, the number of existing independent components is unknown. Assuming that the independent components correspond to the activities of individual muscles of the forearm, there might exist way more than 6 independent components.

For those reasons, the considered problem is probably not a case where Blind Source Separation can be performed easily and the extracted vectors most likely do not correspond to independent components. Hence, the input data has no physical nor conceptual meaning, consequently the test data has no or only few similitude with the training data and the deep-learning model fails.

No significant differences can be observed when comparing the simple and siamese Ameri-like architectures. A potential explanation can be found by looking at the training losses of the siamese model. Indeed, while the regression loss l_R decreases a lot during training (reaches values smaller than 5), the feature loss l_F stays much higher (around 20; note that both loss are normalized and are therefore comparable). This suggests that the siamese model fails to learn the desired mapping between the feature and pose spaces, and has therefore no significant advantage in comparison to the simple Ameri-like model.

5.2 Further work and improvements

An interesting extension to this work would be to apply the same transfer-learning pipeline on high-density (HD) EMG electrodes instead of the used low-density (LD) 6-channel and 8-channel electrodes. Indeed, the basic channel duplication and channel dropping techniques for input resizing might work better with HD-electrodes, since losing or duplicating a few recording channels would probably have less relative impact if a high number of channels are used.

Furthermore, the siamese network training could potentially be improved by using more clever input pairs. Indeed, former studies on siamese networks propose to implement hard negative pair mining [23] [24]. This consists of generating pair of inputs that either belong to the same class or to very different classes. Hard negative pair mining enable better and faster learning. In the case of regression tasks for pose estimation, the concept of similar or different classes does not exist, nonetheless Doumanoglou et al. showed that the method can be adapted by generating that are pairs that are both close in the pose space, or have very large pose differences [19]. This method could potentially improve the learning of the siamese model, but comes with a much higher computational cost.

Finally, in this work reshaped time-windows from the "channel x temporal" EMG data are used as input into the CNNs, as this was by Ameri et al. [2]. However, other input forms could be tested such as raw-time windows, spectograms or wavelet transformed signal. Such inputs are used in some CNN models for audio or speech processing [25] [26] and could potentially yield better result on EMG data.

6. Conclusion

To conclude, the goal of this work in this work was to explore algorithms for EMG inter-hardware compatibility in single-finger regression tasks. Therefore, transfer-learning methods were designed and tested. The conceptual idea was to use 6-electrode EMG data to pre-train CNN-models and, in a second time, to fine-train and test the models on 8-electrode data.
The first step of the process was to resize the data recorded by the two different hardware, so that it can be used by the same deep-learning model. Simple augmentation and reduction methods involving channel dropping and duplicating as well as state-of-the-art dimensionality reduction algorithms (ICA and PCA) were tested: a simple CNN and a siamese CNN. On top of that, two different CNN architectures were implemented and different combinations of input resizing and CNN architectures were evaluated.
A striking observation is that the use of ICA in this context is counter-productive and achieves worse performances than single-session training. This suggest that blind source separation can not be performed straightforwardly on low-density EMG recordings.
All-in-all, no improvement could be found from inter-hardware pre-training, as the best obtained regressions with the transfer-learning methods are similar to the performances reached with simple single-user training (without pre-training). Possibly, the use of HD-electrode hardware or improved pair generation in the siamese model could yield better results.

A. Bibliography

[1] Geethanjali Purushothaman. "Myoelectric control of prosthetic hands: State-of-the-art review". In: *Medical Devices: Evidence and Research* Volume 9 (July 2016), pp. 247–255. DOI: 10.2147/MDER.S91102.

[2] A. Ameri et al. "Regression convolutional neural network for improved simultaneous EMG control." In: *Journal of neural engineering* 16 3 (2019), p. 036015.

[3] Weidong Geng et al. "Gesture recognition by instantaneous surface EMG images". In: *Scientific Reports* 6 (Nov. 2016), p. 36571. DOI: 10.1038/srep36571.

[4] Ulysse Côté Allard et al. "Deep Learning for Electromyographic Hand Gesture Signal Classification Using Transfer Learning". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* PP (June 2018). DOI: 10.1109/TNSRE.2019.2896269.

[5] Ali Ameri et al. "A Deep Transfer Learning Approach to Reducing the Effect of Electrode Shift in EMG Pattern Recognition-Based Control". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* PP (Dec. 2019), pp. 1–1. DOI: 10.1109/TNSRE.2019.2962189.

[6] Sara Abbaspour et al. "Evaluation of surface EMG-based recognition algorithms for decoding hand movements". In: *Medical Biological Engineering Computing* 58 (Nov. 2019). DOI: 10.1007/s11517-019-02073-z.

[7] Jangwoo Kwon et al. "EMG signals recognition for continuous prosthetic arm control purpose". In: Dec. 1996, pp. 365–368. ISBN: 0-7803-3702-6. DOI: 10.1109/APCAS.1996.569291.

[8] Marie-Françoise Lucas et al. "Multichannel Surface EMG Classification Using Support Vector machines and Signal-based Wavelet Optimisation". In: *Biomedical Signal Processing and Control* 3 (Apr. 2008), pp. 169–174. DOI: 10.1016/j.bspc.2007.09.002.

[9] Tadahiro Oyama et al. "Wrist EMG signals identification using neural network". In: Dec. 2009, pp. 4286–4290. DOI: 10.1109/IECON.2009.5415065.

[10] Francesco Tenore et al. "Decoding of Individuated Finger Movements Using Surface Electromyography". In: *IEEE transactions on bio-medical engineering* 56 (June 2009), pp. 1427–34. DOI: 10.1109/TBME.2008.2005485.

[11] Silvia Muceli, Ning Jiang, and Dario Farina. "Extracting Signals Robust to Electrode Number and Shift for Online Simultaneous and Proportional Myoelectric Control by Factorization Algorithms". In: *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society* 22 (Oct. 2013). DOI: 10.1109/TNSRE.2013.2282898.

[12] Jimson Ngeo, Tomoya Tamei, and Tomohiro Shibata. "Continuous and simultaneous estimation of finger kinematics using inputs from an EMG-to-muscle activation model". In: *Journal of neuroengineering and rehabilitation* 11 (Aug. 2014), p. 122. DOI: 10.1186/1743-0003-11-122.

[13] Yu Hu et al. "A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition". In: *PLOS ONE* 13 (Oct. 2018), e0206049. DOI: 10.1371/journal.pone.0206049.

[14] H. Wu et al. "Face recognition based on convolution siamese networks". In: *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. 2017, pp. 1–5. DOI: 10.1109/CISP-BMEI.2017.8302003.

[15] M. Heidari and K. Fouladi-Ghaleh. "Using Siamese Networks with Transfer Learning for Face Recognition on Small-Samples Datasets". In: *2020 International Conference on Machine Vision and Image Processing (MVIP)*. 2020, pp. 1–4. DOI: 10.1109/MVIP49855.2020.9116915.

[16] Kexin Feng and Theodora Chaspari. "A Siamese Neural Network with Modified Distance Loss For Transfer Learning in Speech Emotion Recognition". In: (June 2020).

[17] Gregory R. Koch. "Siamese Neural Networks for One-Shot Image Recognition". In: 2015.

[18] Ivette Vélez, Caleb Rascon, and Gibran Fuentes-Pineda. "One-Shot Speaker Identification for a Service Robot using a CNN-based Generic Verifier". In: (Sept. 2018).

[19] Andreas Doumanoglou et al. "Siamese Regression Networks with Efficient mid-level Feature Extraction for 3D Object Pose Estimation". In: (July 2016).

[20] A. Hyvärinen and E. Oja. "Independent component analysis: algorithms and applications". In: *Neural Networks* 13.4 (2000), pp. 411–430. ISSN: 0893-6080. DOI: https://doi.org/10.1016/S0893-6080(00)00026-5. URL: http://www.sciencedirect.com/science/article/pii/S0893608000000265.

[21] Sanjeev Jain and Dr Rai. "Blind source separation and ICA techniques: a review". In: *IJEST* 4 (Apr. 2012).

[22] Simon Haykin and Zhe Chen. "The Cocktail Party Problem". In: *Neural computation* 17 (Oct. 2005), pp. 1875–902. DOI: 10.1162/0899766054322964.

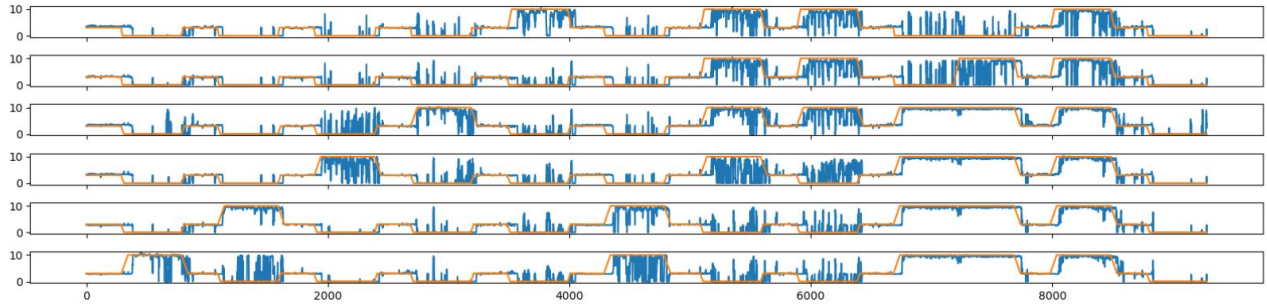
[23] Jiang Wang et al. "Learning Fine-Grained Image Similarity with Deep Ranking". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Apr. 2014). DOI: 10.1109/CVPR.2014.180.

[24] Edgar Simo-Serra et al. "Discriminative Learning of Deep Convolutional Feature Point Descriptors". In: (Dec. 2015). DOI: 10.1109/ICCV.2015.22.

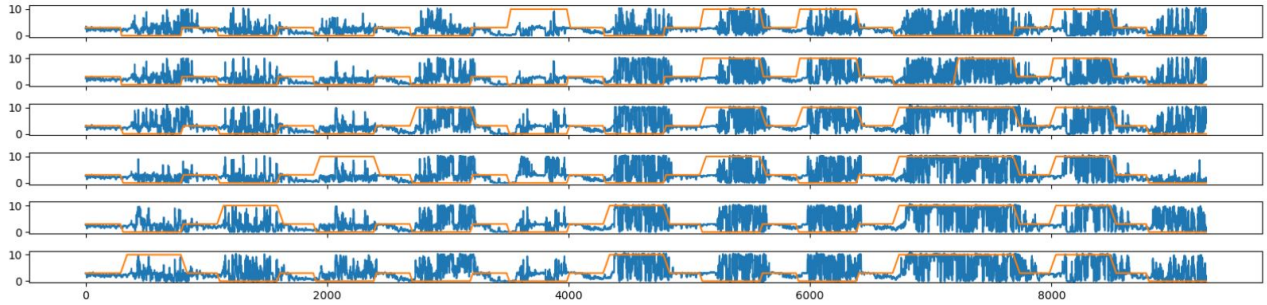
[25] Ossama Abdel-Hamid et al. "Convolutional Neural Networks for Speech Recognition". In: *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* 22 (Oct. 2014), pp. 1533–1545. DOI: 10.1109/TASLP.2014.2339736.

[26] Yanick Lukic et al. "Speaker identification and clustering using convolutional neural networks". In: Sept. 2016, pp. 1–6. DOI: 10.1109/MLSP.2016.7738816.

B. Examples of obtained regressions

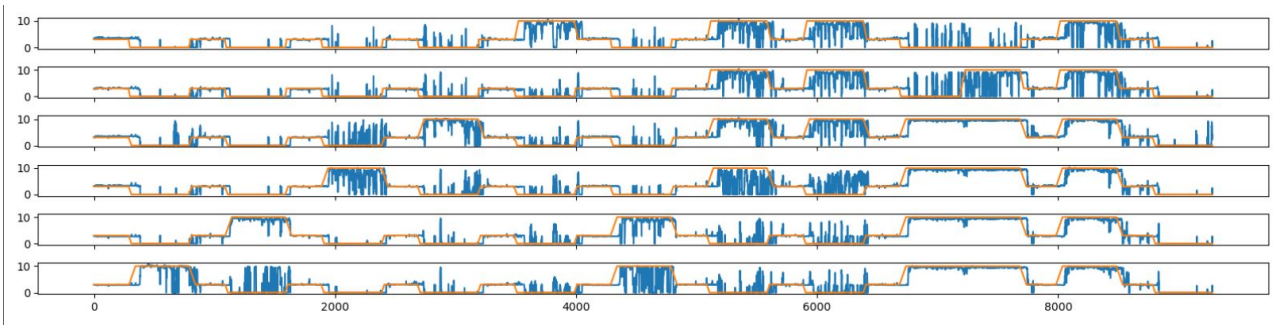


(a) Good regression: Data pre-trained on augmented 6-channel data and fine-trained on complete 8-channel data

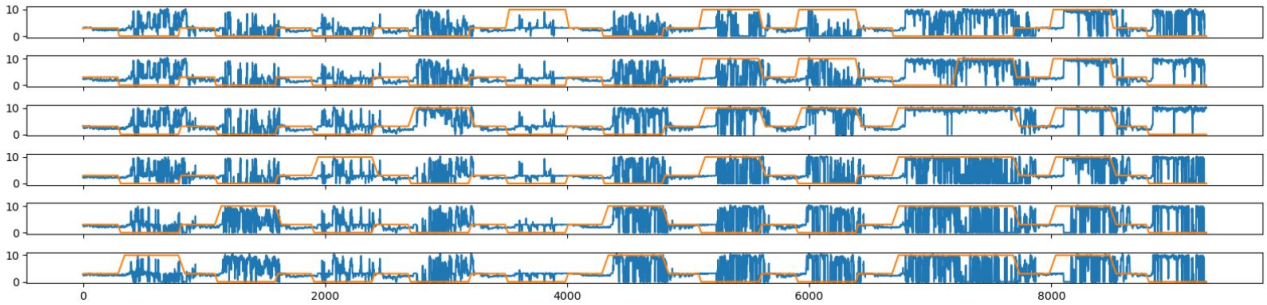


(b) Poor regression: Data pre-trained on augmented 6-channel data and tested on 8-channel data without fine-training

Figure B.1: Examples of regressions using the simple Ameri-like model



(a) Good regression: Data pre-trained on augmented 6-channel data and fine-trained on complete 8-channel data



(b) Poor regression: Data pre-trained on augmented 6-channel data and tested on 8-channel data without fine-training

Figure B.2: Examples of regressions using the siamese Ameri-like model