SEMESTER PROJECT

ANALYSIS OF TIME DATA TO DECODE MOTOR INTENTION

JUNE 2020

Student:
Nikolina Tomić
nikolina.tomic@epfl.ch

Supervisor:
Vincent Mendez

Laboratory:
Translational Neural Engineering Lab (TNE)

1 Introduction

Neural signals from peripheral nervous system (PNS) are interesting for both control and sensory feedback of upper limb neuroprosthesis. As opposed to traditional methods, such as electromyography (EMG) and Targeted Muscle Reinnervation (TMR) they enable more physiological control of neuroprosthesis [1],[2]. Thus, decoding motor intention and providing sensory information from electroneurographic (ENG) signals is a challenge worth attention in order to restore motor hand functions, i.e. to control the prosthesis.

This project aims at investigating ENG signals recorded using TIME electrodes. More precisely, at testing performance of Deep Learning techniques, such as Artificial Neural Networks (NN) and Convolutional Neural Networks (CNN). These signals were subject of previous study from [1], where potential of TIME electrodes to describe motor intention is demonstrated by applying Support Vector Machines (SVM) method. Therefore, it represents reasonable basis for further research and this project focuses on providing motor control, i.e. decoding motor intention of amputee by applying Deep Learning algorithms.

Primarily, we strive to get insight in the possibility of extracting the pattern being stored in the signals. In [1], this is already proven to be the case, and here it is investigated from Deep Learning perspective. Artificial Neural Networks models were created to fit the preprocessed data and additionally Kolmogorov-Smirnov test was applied to describe the data discriminatory properties.

Nevertheless, we seek to decode the pattern from as less preprocessed data as possible. Feature engineering, process of designing the features and extracting useful information from the data is a long-used approach. Nevertheless, so called feature learning technique is novel approach, considered as a more convenient and useful. This implies taking raw signals, and letting the algorithm learns the features on its own. As Deep Learning methods are suitable for such an approach, raw signals are tackled in the final part of the project. Furthermore, preprocessing takes time and as decoding system should be fast, it may be an obstacle.

Yet another mission is regression rather than classification. This is because regression provides more flexibility in terms of activity of actuators and their Degree of Freedom (DoF). As opposed to classification, regression allows for more DoF being active, with different activity level.

Finally, generalizing the model to the case of more than one session is desirable in real-world applications. Namely, two ways of tackling the data are possible: intra-session and inter-session. The former, meaning fitting the model to only single experimental session data (while validating and testing on the same session as well). The letter, taking all provided sessions together and creating a single model accordingly. Ideally, in case of 5 experimental sessions, we would use 1^{st},2^{nd}, and 3^{rd} session for training, then successfully validate and test on 4^{th} and 5^{th}. This means algorithm is able to learn on signals from the experiment and predict on unseen signals from the following days. Therefore, besides intra-session, inter-session approach is used, as final goal is to enable the application in the real-life situation.

Additionally, the idea here was also to examine and possibly adopt the pipeline widely used in EMG data processing, i.e. creating the images from the signals and using CNN architecture. While EMG signals processing already benefit from CNN based classification and regression, it is undiscovered in field of ENG signals processing. It is examined here whether eventually it has a potential to decode ENG data. It is considered in both, preprocessed and raw signals cases.

2 Brief literature overview

Electroneurographic (ENG) signals are more challenging and less discovered than electromyography (EMG) signals due to their complexity. Starting from possible solutions for control of artificial upper-limbs, in [3] two main approaches are summarized. These are control provided from acquisition by Surface Electrodes (SE) or Implantable Electrodes (IE). The former, being deeply studied and already experienced their use in practice, in particular giving EMG signals. The letter, oppositely being less studied, but considered as powerful. The main advantage of using IE that has been proven is they do solve practical issues associated with SE. The examples are consistency of the signal over time, with less impact of surrounding noise and less fatigue patients face with while using SE interface. The better quality of signals recorded by IE has been suggested and it is believed IE provide more useful and valuable information than SE. Furthermore, they allow for sensory feedback. However, machine learning performance improvement is yet unanswered question. Therefore, nowadays researchers are putting more effort to discover this fact.

Four main families of IE electrodes are given by literature, based on target and location within the target: extra-muscular, intra-muscular, extra-neural and intra-neural [3]. Among them, intra-neural electrodes are the most invasive, yet most selective ones. Basically, they provide neuronal firing patterns due to direct contact with afferent and efferent fibers.

Next, brief overview of research done on signals recorded by mentioned sub-type of electrodes will be given. This contains:

- Micro-Electrode Arrays - MEAs, with Utah Electrode Array (UEA) and Utah Slanted Electrode Array (USEA) being an example
- Longitudinal Intrafascicular Electrode - LIFE

• Transverse Intrafascicular Multichannel Electrode - TIME

Research done one signals recorded by using Utah Slanted Electrode Arrays (USEAs) is presented in [4]. Note that USEAs are modification of UEAs with electrodes of different length, to reduce number of redundant electrodes being present in UEAs. Firing rate is extracted from 33.3 ms windows and modified Kalman filter is used to decode the signals. The subject was able to reach 5 DoF movement via a virtual prosthetic hand, and up to 131 sensory percepts across phantom hands at the testing phase. This outcome shows the large potential of neural signals decoding in purpose of motor and sensory control of prosthetic device.

The work presented in [4] investigates LIFEs (4 thin film LIFE - tf-LIFE4) electrodes to provide both motor control and sensory-feedback. Three hand movements were defined and final classification accuracy up to 85 % is achieved. Furthermore, accuracy is shown to be larger in case of recording signal from several sites rather than single location and by training the patient before final recording of the signals. Yet another study from [2] shows the potential of tf-LIFEs to provide motor control. Spike sorting based decomposition is applied to classify the signals in 4 classes. Support Vector Machine (SVM) classification in binary-fashion is conducted and accuracy reached optimistic value of 85 % for different grip types. Again, there is an indication of improving the accuracy by increasing the number of channels up to some degree.

Concerning TIME electrodes, while the extraction of sensory-feedback is proven to be possible, motor control potential is less discovered. It is only studied in [1], where classification in 11 classes is conducted, with encouraging accuracy - up to 82 %. Binned -Root Mean Square is calculated to create feature-representation of signals, Principal Component Analysis (PCA) is applied to select the most discriminative channels and SVM is used to classify. Note that, in this project the same signals are used and preprocessing pipeline repeated.

Finally, brief focus on EMG signals, as part of this project is inspired by framework created for EMG signal processing. Namely, the state-of-the art approach is presenetd in [6]. Particularly, CNN architecture is proposed, what requires image representation of one dimensional signals. This is an example of feature-learning method, where raw signals are input in CNNs, and also example of regression approach, rather that classification. The same architecture was then used in [7], where again low regression loss is achieved.

3 Project pipeline

As already mentioned in the introduction, few perspectives are combined in this project, as aim was to get insight in qualitative properties of TIMEs signals. In Fig. 1, structure of this project is visualized. 4 movements, followed by rest phase were defined and corresponding ENG signals provided. Signals are first cleaned from artifacts and downsampled, as it was done in [1]. Classification and regression targets are created and this project is then driven by two main ideas: 1. preprocess the signals and decode features representation; 2. decode the original, raw signals. In the former case, decode stands for classification, while in the latter for regression.

Preprocessing is done similarly as in [1], by calculating Root Mean Square (RMS) of the signals. The NN classification approach is then performed for intra-session and inter-session data split.

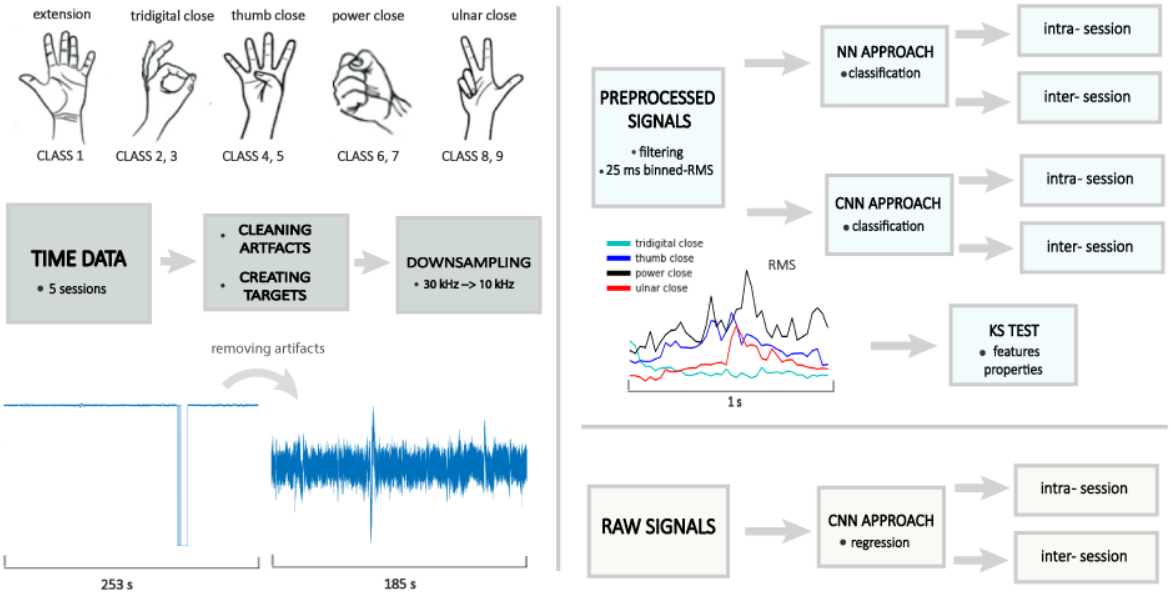

Figure 1: The structure of this project

Besides NN, CNN on preprocessed-data is also tested, again in both intra-session and inter-session fashion.

Kolomgorov-Smirnov test is applied in order to analyse features properties, and to compare intra-session and inter-session capabilities.

Last step was to test performance of CNN-regression approach on raw signals. To recall, final goal was to test the same pipeline widely used in EMG signal processing, i.e. creating images from RAW data and performing CNN, while doing regression rather than classification. This is, again done in intra-session and inter-session fashion.

4 Data and targets

Here, data structure needed to familiarize the reader with concepts used in this project will be explained. For more information regarding acquisition procedure please refer to [1].

The data for this study were collected on 4 different days and 5 experimental sessions are on our disposal (last day 2 sessions were recorded). Five different movements of phantom hand were defined: Tridigital Pinch, Thumb opposition, Power grasp, Ulnar finger movement, and Finger abduction and adduction. In this project last movement is excluded, as it is not achievable with robotic hand (Fig. 1). Corresponding signals were recorded, with each movement formulated as follows: flexion, followed by extension of phantom fingers, ending with rest phase. Duration of each flexion and extension was 1 s, while rest phase was performed for 3 seconds. In total, one repetition lasts 5 seconds. Each of 5 different hand gestures were repeated 10 times resulting in 250 s (5x10x5s) of data per session, with sampling frequency of 30 kHz. TIMEs are 56-channels electrodes - thus providing 56-dimensional signals.

4.1 Cleaning and downsampling signals

Due to the data acquisition imperfection, mainly displacement of the electrodes, signals were superimposed by the artifacts. Therefore cleaning is performed, i.e. repetitions containing artifacts are removed. This is simply done by visual review of the signals, as artifacts are clearly evident from jump in signal amplitude (illustrated in Fig. 1). After cleaning, final data structure was the same as in [1]. Nevertheless, for the purpose of NN, 3rd session was additionally cleaned, i.e. one repetition was removed. Reason is a peak, not that large to be rejected previously, but remarkably impacting NN performance (shown in red in Fig. 2).

Finally, signals were down-sampled to 10 kHz, as it was done in [1]. It is shown in [8] that 10 kHz is the optimal sampling frequency for ENG signals and moreover it reduces the computational power needed to process the data. The example of original and down-sampled signal is shown in Fig. 2, as a visual demonstration that down-sampling do not result in loss of the information.

4.2 Creating targets

Classification Classification targets are straight forward ones. In summary, there are 9 classes on our disposal. The rest phase is chosen to be class 1. Flexion and extension of each movement are represented with 2 different classes; flexion with classes 2, 4, 6, 8 and extension with classes 3, 5, 7, 9 (see Fig. 1); thus providing 8 classes in total (4 movements).

Regression Regression targets are created as 5-channel signals, where each channel represents single finger angle. The principle is as follows: linearly increasing signal from 0 to 1 when flexion of finger, linearly decreasing signal from 1 to 0 when extension, and 0 when rest.

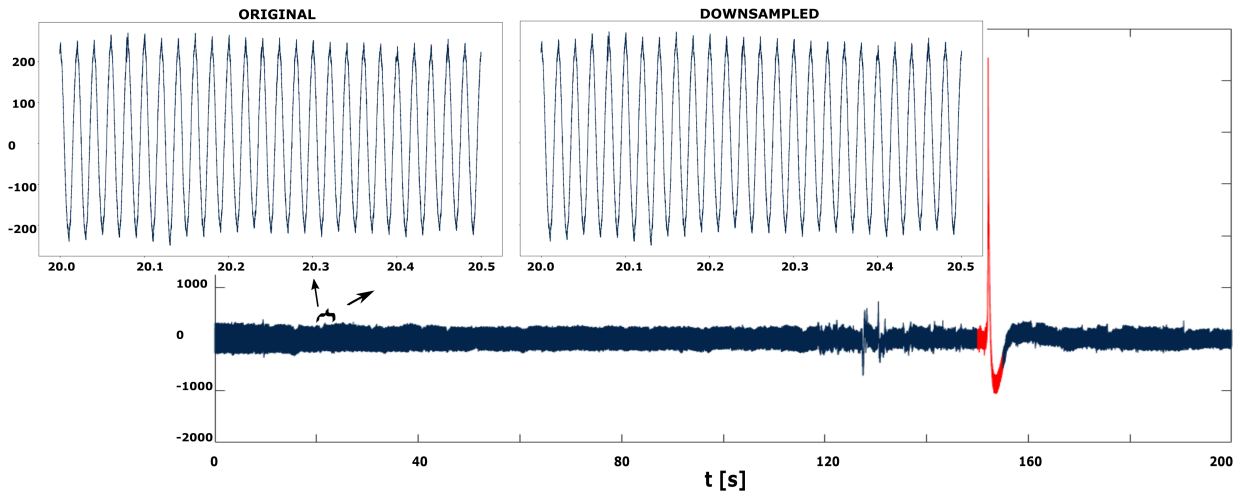

Figure 2: Illustration of original, full signal and original versus down-sampled 500 ms long segment. 1st channel of session 3 is shown. Red part of the signal represents the peak that was cleaned in favor of Neural Networks performance

In Fig. 3 representatives of regression targets are shown, each demonstrating 5 seconds (corresponding to single repetition).

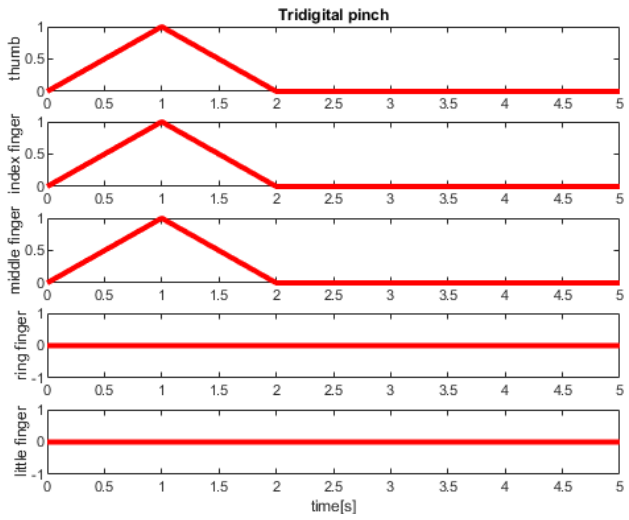

Figure 3: Regression targets for Tridigital Pinch: 3 active fingers, with 1st s representing flexion, 2nd s representing extension, followed by the rest to 5th s. Unactive fingers are at rest during entire repetition. Regression targets are actually angles of fingers.

4.3 Training-test-validation split

Data split is created such that 60% of data makes training set, while rest is taken for validation (20%) and test (20%).

Intra-session Intra session sets are always chosen randomly, using *scipy* library. Note that, depending on duration of the session after cleaning, total data size may vary.

Inter-session Here, 1st, 2nd, and 3rd sessions are used for training, 4th for validation, and 5th for testing. Yet another data split is used in inter-session approach (section 5.4.2). Namely, data from all the sessions are merged and random split applied as for intra-session. This was done to test the gap in performance between regularly (previous case) and randomly split data, as poor performance on validation and testing set is observed for regular split.

5 Methods

5.1 Technical requirements

To clean the signals and create targets, Matlab (version 2019) is used, no special toolbox required. For the rest of the project, Python with packages listed in Table 1 is used.

List of python packages used	
library	version
h5py	2.10.0
keras	2.3.1
matplotlib	3.2.1
numpy	1.18.5
pandas	1.0.4
scipy	1.4.1
talos	1.0.0
tensorflow	2.1.0
tqdm	4.41.1

Table 1: The list of specific packages used for this project development

5.2 Choice of the models

As there is no baseline provided for TIMEs data processing, the choice of the models is diverse yet uncertain. Thus, short explanation of general approach used for model development is given.

Neural Network Starting point for was very simple model, which was increased to improve both training and validation performance. While training loss was increasing with model complexity, validation loss was monitored and simply, at the level when it becomes to drop complexity is no longer increased. The idea is to make a balance between good training performance and generalization, i.e. to prevent overfitting. Fixed model and training parameters were then chosen experimentally and are given for each model.

Convolutional Neural Network Due to the inspiration from EMG signals processing, Ameri-like model from [6] and [7], is used as a framework. It is then increased or decreased following the same idea of balancing training and validation performance.

Loss and activation in output layer For classification activation function is set to be *softmax*, while loss is *categorical cross-entropy* as they are considered as most suitable. In case of regression, *mean squared error* is used as loss function, and activation in output layer is *linear* (*relu* was also considered, but it did not change the result).

5.3 Preprocessing

Preprocessing is done similarly as in [1], i.e. signals are filtered and then 25 ms binned-RMS (Root Mean Square) is calculated. The difference here is that, only low-pass filtering is performed as a first step (before calculating binned-RMS), instead of band-pass one. Corresponding steps are demonstrated in Fig. 4, along with plots showing results after each step.

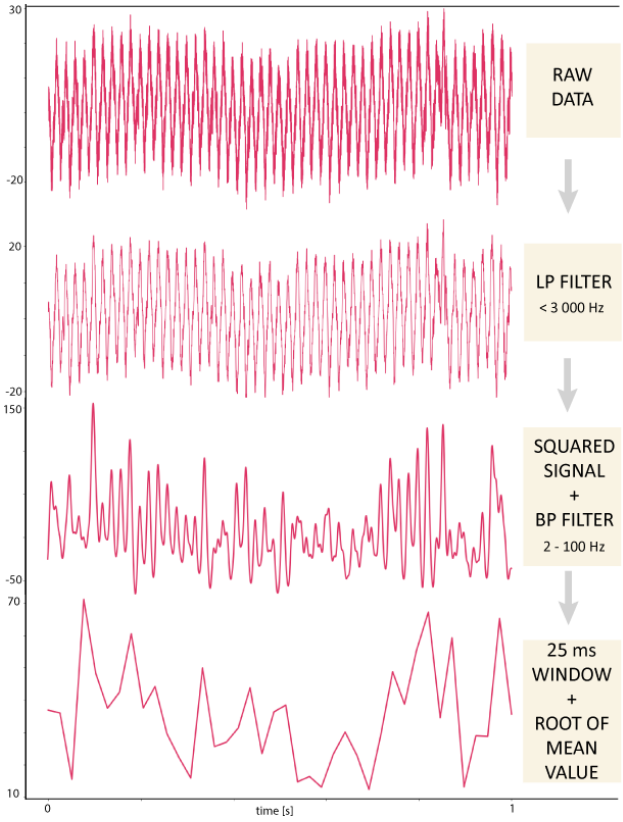

Figure 4: From raw signal to RMS representation: steps and results after each step shown in 1 s long segment of the signal. More precisely, 1st second of 13th channel from session 1 is shown.

In Fig. 5 potential of binned-RMS signals to separate 4 flexion intentions is shown. One can observe various amplitude levels in distinct signals, where each of them represents flexion from one of the 4 phantom-hand movements. Note that, after calculating binned-RMS, sampling frequency becomes 40 Hz.

5.4 NN classification

As in [1], classification futures are firstly defined such that they contain only first 200 ms of each repetition, where the rest of 800 ms is rejected. Then, NN model is augmented and performance is tested on full 1-s signal, to check whether the pattern formation exists during entire movement. 1-s signals are then used to perform inter-session classification.

As NN are considered being sensitive to different scales, standardization is performed on RMS data. This is done by extracting mean value and standard deviation from training set, and performing Z-normalization on training, validation and testing data accordingly. Additionally, in all cases of NN approach, size of class rest is reduced to the size of other 8 classes. Namely, it is known that NN are prone to errors due to the unbalanced classes, and in this experiment rest phase took 3 sec while others took only 1. Therefore, it makes sense to balance all the classes

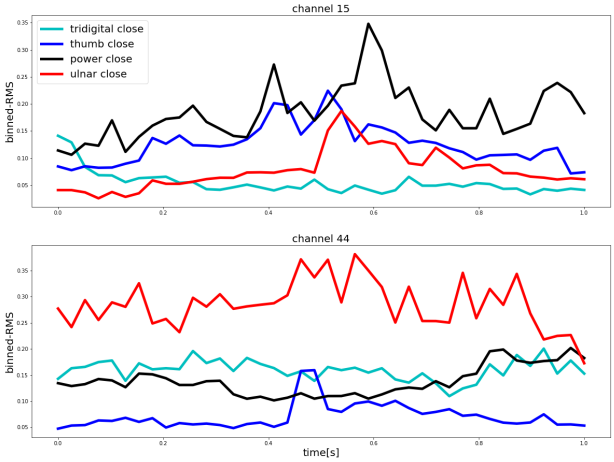

Figure 5: Binned-RMS signals. More precisely, 1st second of each of 4 gestures from Session 1 is shown. Note that channels 15 and 44 are represented, but in total there are 56 channels.

before applying NN framework. Note: input size is 56 due to the 56 channels given by TIME electrodes, and output size is 9 (9 classes).

5.4.1 Intra-session

200 ms segment Experimentally, the following model architecture is chosen: Feed-forward neural network containing 3 hidden layers and dropout layer in front of the output one (Fig. 6). The following training parameters were fixed:

- optimizer - *Adam*
- initial learning rate - 0.01
- exponential decay of learning rate
- number of epochs - 100
- early stopping while monitoring validation loss and patience of 40 epochs.

Then, grid search is performed to find optimal model hyper-parameters shown in Table 2.

1 s - entire repetition To check existence of pattern formation during entire movement, 1 s of the

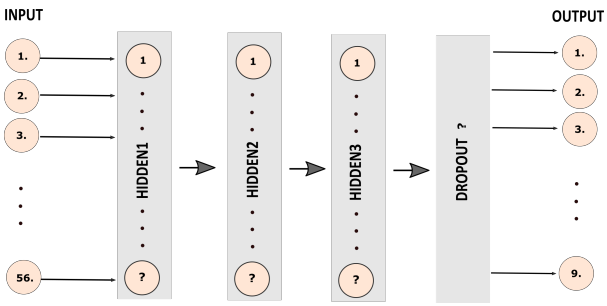

Figure 6: NN arhitecture used for 200 ms RMS data fitting. Note that parameters marked with **?** are to be defined in grid search.

signal is taken instead of 200 ms segment. Experimentally, previous model is augmented by one hidden layer (more data usually requires more complex architecture), fixed parameters are kept, and grid search is performed for the model hyper-parameters shown in Table 3.

5.4.2 Inter-session

Again experimentally, and due to the complexity of the data, previous model is enriched by two new hidden layers, while fixed parameters remained the same. Grid search is performed to optimize parameters shown in Table 4.

As already stated in Section. 4.3., inter-session training is performed twice, on regularly created sets and randomly created ones. Random split makes sense in terms of comparing performance reached from regular inter-session versus randomly created training, validation and test sets. In other words if the results from intra-session and random-fashion inter-session are satisfactory, that means data hold discriminatory properties. At the same time, if performance is poor on regular-fashion inter-session, this reveals divergence between sessions, which prevents the algorithm to generalize properly.

5.5 CNN classification

Starting from RMS data, 200 ms windows with 25 ms overlap are created. The window length is adopted from EMG literature [6],[7],[9],[10]. Namely, 150-200 ms windows are considered as optimal, as the criterion is the trade off between delay in real-time processing (the smaller the window-the smaller the delay) and taking the stationary part of signal (note that second property is not equal for EMG and ENG signals).
We have 56 channels and 8 samples representing 200 ms of RMS, giving us 56x8 images. They are then reshaped to 16x28, to obtain more square-like shape, driven by the idea from [6]. Note that images here are low resolution, thus blurred display in Fig. 7.

5.5.1 Intra-session

To perform intra-session classification simple CNN architecture inspired by [6] is created, shown in Fig. 8.
The following training and model parameters were fixed:

- optimizer - *Adam*
- initial learning rate - 0.01
- step decay of learning rate
- number of epochs - 50
- early stopping while monitoring validation loss and patience of 15 epochs.
- activation in fully connected layers - *relu*
- kernel initializer - *glorot uniform*

Grid search is performed to optimize hyper-parameters listed in Table 5. Note that here data set is little, therefore it is expected inter-session approach would enable better learning.

200 ms intra-session	
hyper-parameter	possible values
activation function	[relu, elu]
drop rate	[0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3]
hidden layer 1	[32, 40, 64 , 80]
hidden layer 2	[40, 64, 80 , 128]
hidden layer 3	[64, 80, 128, 256]

Table 2: Grid search parameters used in the case of intra-session 200 ms RMS signals

1 s intra-session	
hyper-parameter	possible values
activation function	[relu, elu]
drop rate	[0, 0.05, 0.1, 0.15, 0.2]
hidden layer 1	[32, 40, 64]
hidden layer 2	[40, 64, 80, 128, 256]
hidden layer 3	[64, 80, 128, 256]
hidden layer 4	[64, 80, 128, 256]

Table 3: Grid search parameters used in the case of intra-session 1 s RMS signals.

inter-session	
hyper-parameter	possible values
activation function	[relu, elu]
drop rate	[0, 0.1, 0.2]
hidden layer 1	[32, 64]
hidden layer 2	[64, 80]
hidden layer 3	[64, 80]
hidden layer 4	[64, 80, 128, 256]
hidden layer 5	[64, 80, 128, 256]
hidden layer 6	[64, 80, 128, 256]

Table 4: Grid search parameters used in the case of inter-session binned-RMS signals

RMS images intra-session	
hyper-parameter	possible values
drop rate	[0, 0.1, 0.2, 0.3, 0.4, 0.5]
hidden layer 1	[64]
hidden layer 2	[24, 32, 64, 128]
lambda	[1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 0]

Table 5: Grid search parameters used in the case of intra-session RMS images

Figure 7: Example of Binned-RMS images. Image size is 16x28 pixels

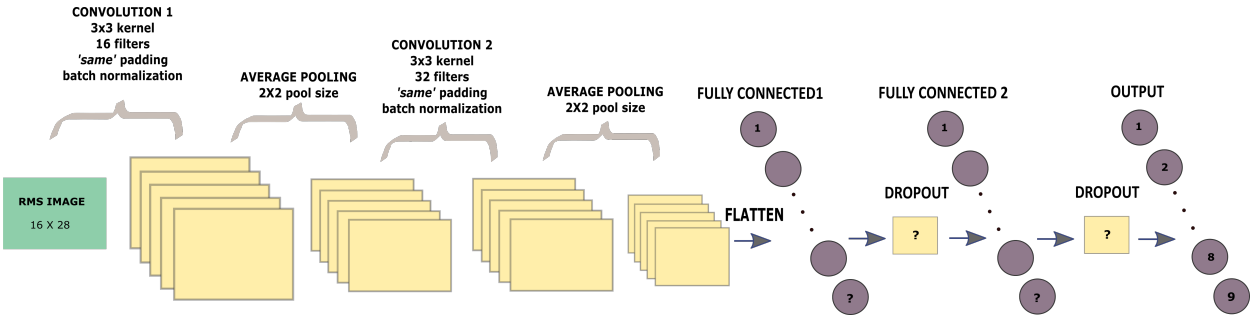

Figure 8: CNN architecture used to fit intra-session RMS images: 2 convolutional blocks and 2 fully connected layers, each followed by dropout. Note that parameters marked with **?** are to be found in grid search.

5.5.2 Inter-session

To perform inter-session, again the complexity of model from intra-session is increased. Instead of two convolutional blocks, seven of them are defined, and three additional fully-connected layers, each followed by dropout are implemented. While fixed parameters are kept, grid search is used to find optimal parameters listed in Table 6.

RMS images inter-session	
hyper-parameter	possible values
drop rate	[0, 0.3]
hidden layer 1	[64, 128]
hidden layer 2	[64, 128]
hidden layer 3	[64, 128]
hidden layer 4	[64, 128]
hidden layer 5	[64, 128]
lambda	[1e-5, 1e-1, 0]

Table 6: Grid search parameters used in the case of inter-session RMS-images

5.6 Kolmogorov–Smirnov Test

Kolmogorov–Smirnov (KS) test is an useful tool if we want to check to what extent distributions of two one-dimensional variables match, i.e. it is non-parametric measure of similarity between two one-dimensional distributions. In this particular case, it is used to test significance of certain channel (feature) to separate certain class. Particularly, data are split in 9 sets: each of them containing the RMS signals of the single class. Then, we can take a look from binary-classification perspective. Starting from 9 classes, there are 37 possible combinations of binary-classification representations. Taking one of them, KS test is applied 56 times, as data contains 56 channels and KS test can only be used on 1D data.
For each of 37 combinations, mean and maximum value of KS is calculated. This gives an insight in the degree of similarity between two classes. If they are the same, KS score is 0. The larger the score - the more descriptive the features. Ideally, KS score is equal to 1, meaning two classes are perfectly separated.

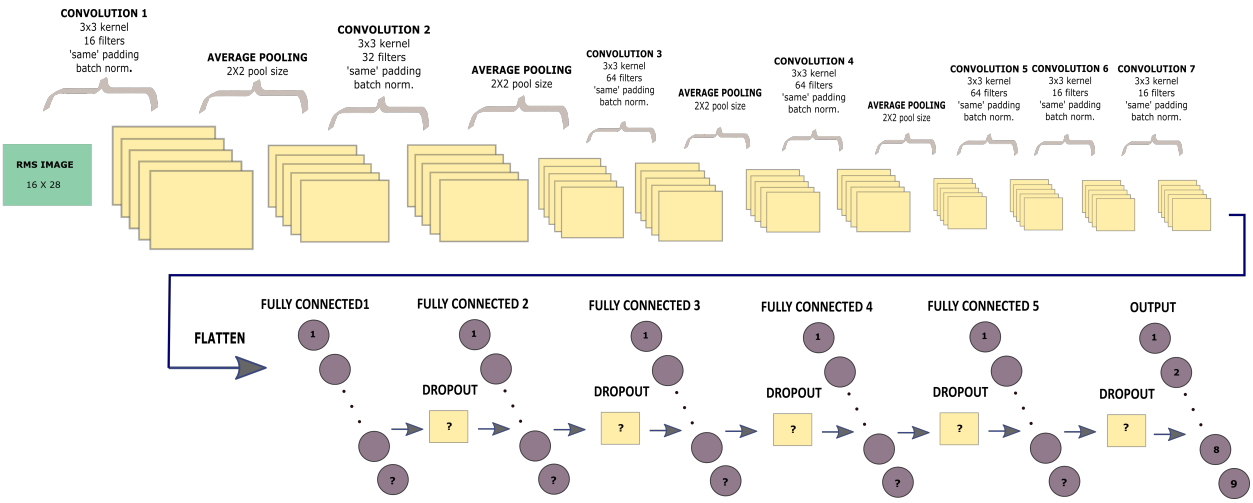

Figure 9: CNN architecture used to fit inter-session RMS images: 7 convolutional blocks and 5 fully-connected layers, each followed by dropout.

Nevertheless, considering solely a channel resulting in maximum value of KS score can overshadow global picture. More precisely, maybe we get high maximum score, but the rest channels result in low scores, what should also be taken in account. On the other hand, mean value gives that global overview, but it can be lower due to the noisy channels. In this project channel selection was not performed, therefore mentioned limitation may potentially exist. Thus, to cope with limitations explained above, both mean and maximum KS score are reported.

KS test is applied on sets containing signals originating from only single session, and sets containing mixed signals, from all sessions. Idea behind is to compare discriminatory properties of binned-RMS intra-session and inter-session features.

5.7 CNN regression

As already mentioned, the final goal of this project is to examine regression performed on raw (previously cleaned and down-sampled) signals. Windows length used for EMG signals is again adopted ([6],[7],[9],[10]), here being 150 ms with overlap of 40 ms. As sampling frequency of raw signals is 10 kHz, size of 150 ms-long images is 56x1500. In Fig. 10 raw image is illustrated. Note that, for this part data were not balanced.

Figure 10: Representation of raw image. Note that height of image is scaled (increased) in the sake of representation.

5.7.1 Intra-session

To fit raw images CNN architecture shown in Fig. 11 is used. Kernel shape here is not usual squared-version, due to the shape of the images. The same holds for the pool size. As previously, model is experimentally created starting from [6], and the following parameters are fixed:

- optimizer - *Stohastic Gradient Descent*
- initial learning rate - 0.001
- step decay of learning rate
- number of epochs - 50
- early stopping while monitoring validation loss and patience of 15 epochs.
- activation in fully-connected layers - *relu*
- kernel initializer - *glorot uniform*

Note that although *Adam* optimizer is the most general, yet optimal choice, it is not used here. The reason is that it requires more memory for computation and it resulted in memory error in case of raw images.

Targets are multiplied by 10 to avoid problem of vanishing gradient, and grid search is performed to optimize model parameters shown in Table 7.

Raw images intra-session	
hyper-parameter	possible values
drop rate	[0, 0.1, 0.2, 0.3, 0.4, 0.5]
hidden layer 1	[32, 64]
hidden layer 2	[16, 24, 32]
lambda	[1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 0]

Table 7: Grid search parameters used in the case of intra-session raw-images

5.7.2 Inter-session

Note that, for both intra-session and inter-session the same architecture is used but hyper-parameters optimized are different (Table 8).

Raw images inter-session	
hyper-parameter	possible values
drop rate	[0, 0.1, 0.2, 0.3, 0.4, 0.5]
hidden layer 1	[32, 64, 128, 256]
hidden layer 2	[16, 24, 32, 64]
lambda	[1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 0]

Table 8: Grid search parameters used in the case of inter-session raw-images

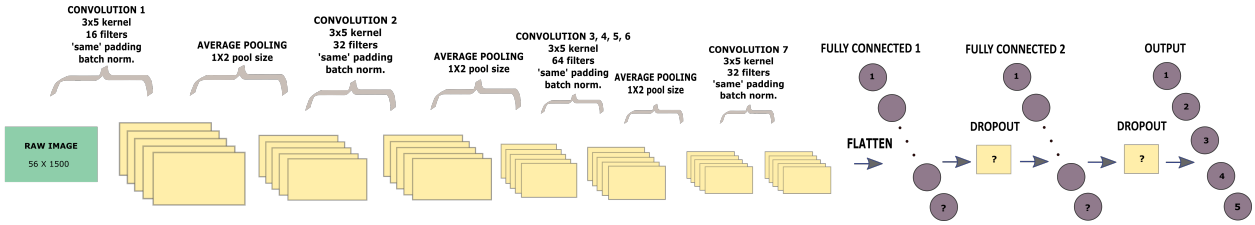

Figure 11: CNN architecture used to fit raw images: 7 convolution blocks (3, 4, 5, and 6 are the same) and 2 fully-connected layers, each followed by dropout

6 Results

After performing grid search, model showing the best validation performance (highest validation accuracy in case of classification or lowest validation loss in case of regression) is taken as a best. Here evaluation results of the best models on training, validation and testing sets will be presented and visualized. Mean and standard deviation of performance are calculated from 100 runs for NN models, and from 20 runs for CNN models.

RMS signals classification In Table 9 performance of best model in case of RMS classification (both intra- and inter- session) is shown, while corresponding hyper-parameters are represented in Appendix. In Fig. 12 and Fig. 13 predicted labels vs ground truth on testing set (part of the testing set - 60 samples) are visualized.

Evaluation of RMS data fitting		
model	session	accuracy [%], $(\mu \pm \sigma)$
RMS 200 ms	session 1	train $= 100 \pm 0$ val $= 84 \pm 2$ test $= 82 \pm 4$
	session 2	train $= 100 \pm 0$ val $= 75 \pm 3$ test $= 73 \pm 5$
	session 3	train $= 99 \pm 0.3$ val $= 75 \pm 2$ test $= 72 \pm 4$
	session 4	train $= 95 \pm 1$ val $= 79 \pm 2$ test $= 72 \pm 5$
	session 5	train $= 99 \pm 0.3$ val $= 79 \pm 3$ test $= 68 \pm 6$
RMS 1 s	session 1	train $= 100 \pm 0$ val $= 88 \pm 1$ test $= 81 \pm 2$
	session 2	train $= 100 \pm 0$ val $= 87 \pm 0.9$ test $= 83 \pm 2$
	session 3	train $= 99 \pm 0.2$ val $= 79 \pm 1$ test $= 72 \pm 1$
	session 4	train $= 97 \pm 0.4$ val $= 76 \pm 1$ test $= 68 \pm 2$
	session 5	train $= 98 \pm 0.3$ val $= 83 \pm 0.9$ test $= 76 \pm 2$
inter-session	regular split	train $= 73 \pm 1$ val $= 11 \pm 2$ test $= 11 \pm 1$
	random split	train $= 83 \pm 1$ val $= 65 \pm 0.9$ test $= 62 \pm 1.5$

Table 9: Results obtained on the best models in case of RMS signals classification

One can see high performance on testing set in intra-session case; up to 82 % on 200 ms and 83 % on 1 s RMS signals. Validation loss is even larger, in every case. Training loss usually goes up to 100 %, which means early stopping did not take place, and validation loss was increasing all the time. We can predict that validation and testing performance would improve if data set is larger, as usually more data is needed for better generalization.

In inter-session case, there is an significant gap between regular-split and random-split approach, with testing accuracy on regular-split sets being very low (11 %), and on randomly-created sets being promising (62 %). Similar holds for validation loss. Moreover, training accuracy can reach 73 % and 83 % in regular- and random-split case, respectively. Training accuracy would probably be even larger if early stopping was not implemented, because training is stopped due to the saturation of validation loss. Nevertheless, decreasing validation performance in favor of very good training is not the objective.

RMS images classification In Table 10 performance of best models in case of RMS images classification is shown, and corresponding hyper-parameters are listed in Appendix. Again, results for single session and inter-session approach are visualized here (Fig. 14), for rest please refer to Appendix.

Observation: in some cases training accuracy is encouraging but standard deviation is large, so it seems to happen just by chance. Moreover, validation and testing accuracy are reaching prior probability at most.

Evaluation of RMS images	
session	accuracy [%], $(\mu \pm \sigma)$
session 1	train $= 52 \pm 19$ val $= 17 \pm 4$ test $= 14 \pm 5$
session 2	train $= 45 \pm 23$ val $= 11 \pm 2$ test $= 13 \pm 5$
session 3	train $= 34 \pm 19$ val $= 9 \pm 2$ test $= 8 \pm 3$
session 4	train $= 34 \pm 24$ val $= 8 \pm 2$ test $= 10 \pm 2$
session 5	train $= 21 \pm 22$ val $= 7 \pm 3$ test $= 10 \pm 2$
inter-session	train $= 18 \pm 3$ val $= 17 \pm 2$ test $= 11 \pm 0.7$

Table 10: Results obtained on the best models in case of RMS images classification

In this case a prior (as a percentage) is given by $\frac{100}{9} = 11.11\%$, where 9 is number of classes. Taking a look at Fig. 14 , one can see the predicted class is almost always the same: so single class is always predicted correctly, but everything else is predicted as that class as well. This basically means optimization algorithm is not able to learn neither on training set. Moreover, it is observed that training ends up soon after 1^{st} epoch (around 17^{th}-20^{th} epoch). To recall, the patience is set to 15 epochs, which means early stopping criterion takes place soon after initialization. In other words, weights are not modified much after initialization and optimal value can't be calculated. The same conclusion holds in both intra-session and inter session. Note that model used in inter-session case is already very complex (7 convolutional layers followed by 5 fully-connected ones), so we can't argue the poor performance is due to the simple model.

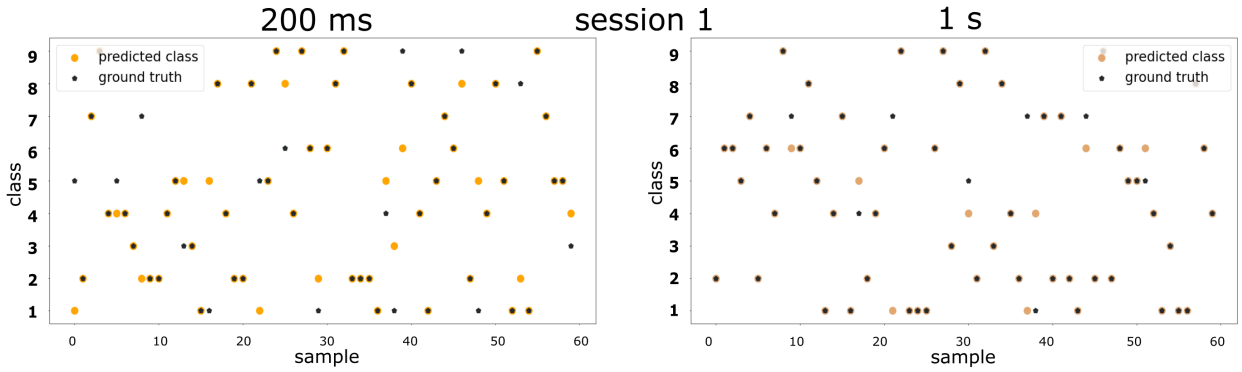

Figure 12: Illustration of results obtained in RMS intra-session case on testing set; 200 ms versus 1 s (session 1 here, find session 2-5 in the Appendix)

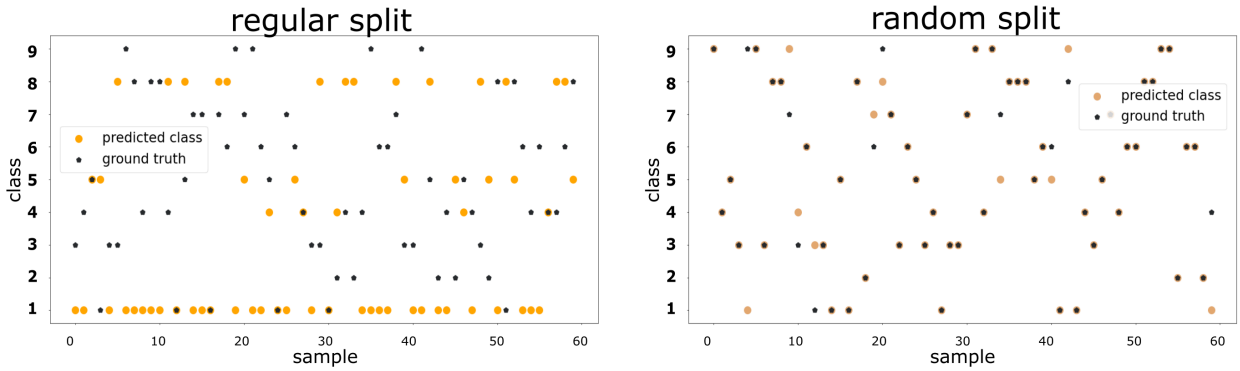

Figure 13: Illustration of results obtained in RMS inter-session case on testing set; regular split versus random split

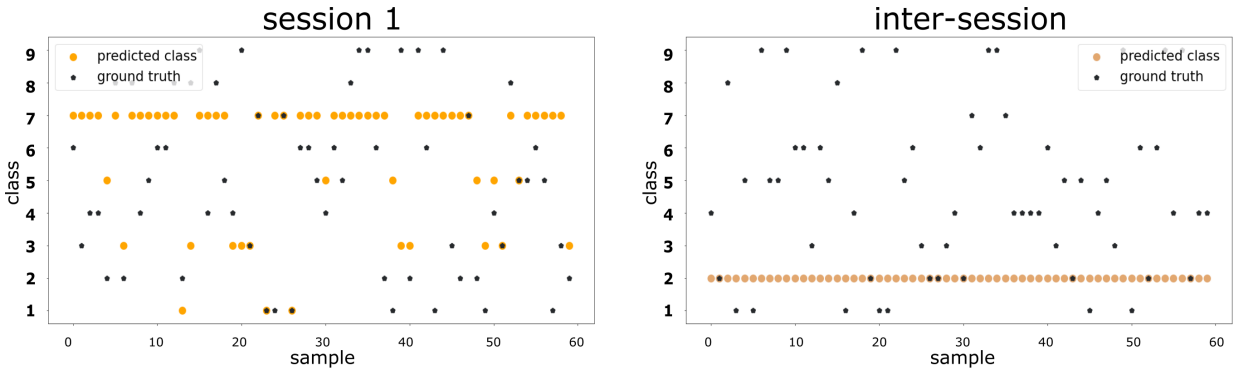

Figure 14: Illustration of results obtained in RMS images on testing set; intra-session and inter-session case (session 1 here, find session 2-5 in the Appendix)

Kolmogorov-Smirnov test Results of KS test are shown in the following figures (Fig. 15, Fig. 16, and 17). Recall: the larger the KS score, the more descriptive the features. The idea is to compare representation capability of binned-RMS signals in two cases: set containing RMS signals from single session (session 4) and all sessions (mixed-data). Here, session 4 is randomly taken, but the same holds for any other experimental session.

One can observe way larger KS score in case of single-session data set, for both maximum and mean value. This outcome refers to blurring the pattern prominence by mixing the sessions. In other words, patterns describing single movement seem to vary from one experimental session to another. That is why in inter-session regular split validation and testing performance is poor, while in random-split case it significantly improves.

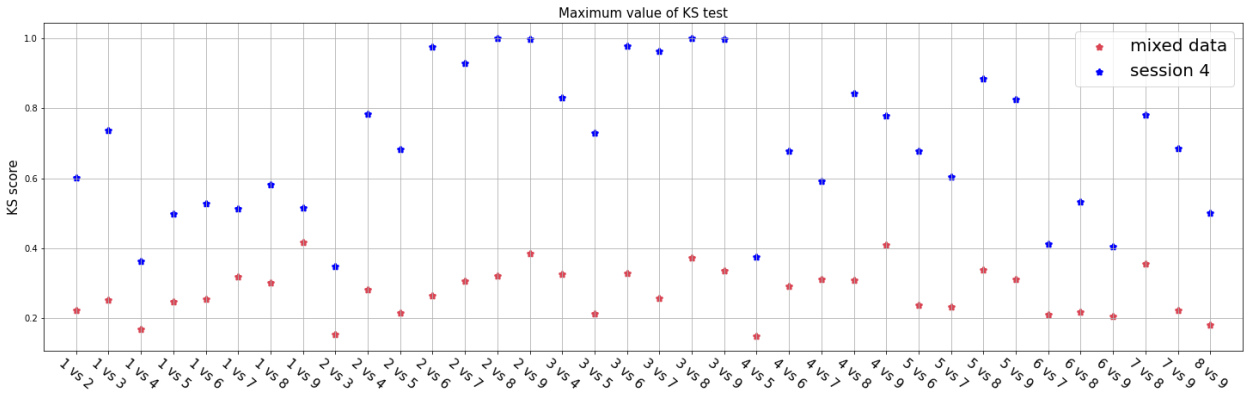

Figure 15: Maximum value of KS test

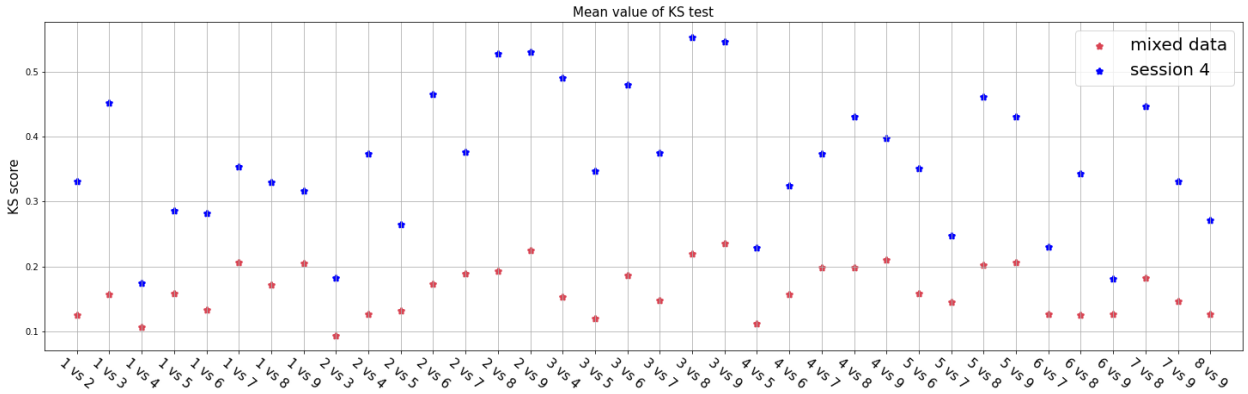

Figure 16: Mean value of KS test

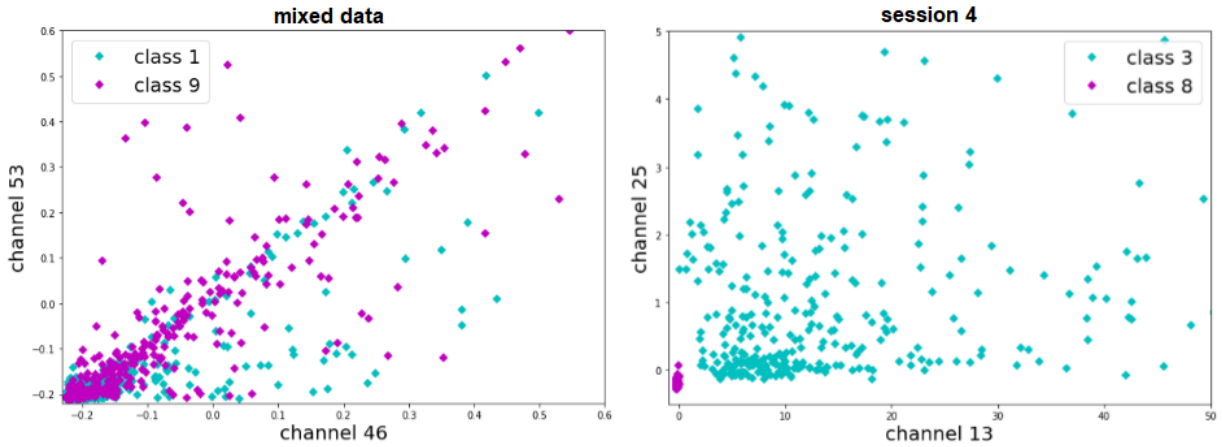

Figure 17: 2D illustration of best separation obtained in case of mixed data versus single session (session 4 is shown, but the same holds for any other session)

Finally, in Fig. 17, best obtained 2-dimensional separations in both cases are visualized. The most we can get in mixed-data case is separation between class 1 (rest) and class 9 (ulnar close-extension) in channels 46 and 53. Although this two channels' activities tend to cluster, they cluster similarly, and there is an apparent correlation between them. This makes an obvious contrast with case of a single session. In the right plot of the figure, we clearly see the inactivity of channels 13 and 25 in case of class 8 (ulnar close-flexion) what enables clear distinction from class 3 (tridigital close-extension), even its activity is sparse (not clustered). Consequently, the similar picture is observed as a general-trend, i.e. in case of single session clear separation, while in case of mixed data clear correlation.

Raw images regression Raw images regression results are shown in Table 11 and illustrated in the following figures. Corresponding model-hyper parameters and session 3-5 are again represented in Appendix. In intra-session case, valuable performance is reached on training set, but again CNN architecture fails to generalize on validation and testing sets. Not only that validation and testing losses are around 6 (recall that targets were multiplied by 10, thus loss of 6 does not sound as little), but also no activity is captured (Fig. 18). The predicted value oscillates in range 1-5 with no indication of describing any peaks (recall that peak represents finger activity). For instance, there is an manifestation of fitting an activity of fingers in session 2, (Fig. 19), but some peaks are not on the right place. Particularly, activity shows up in all the channels when only some of them should be active according to ground truth targets. Thus, different movements are not distinguished.

In Fig. 20, performance on testing set in inter-session case is shown. Again, predicted targets look like noise with no evidence of capturing activity. Testing loss is again low, which means that model was able to fit training data very accurately.

Note that, overfitting prevention is introduced in form of regularization (*lambda*) and drop-out (*drop rate*), for both intra- and inter-session approach. Nonetheless, it did not help to create the model able to generalize.

Evaluation of raw images	
session	loss, $(\mu \pm \sigma)$
session 1	train $= 0.94 \pm 0.16$ val $= 6.69 \pm 0.18$ test $= 6.10 \pm 0.16$
session 2	train $= 1.23 \pm 0.29$ val $= 6.14 \pm 0.31$ test $= 5.9 \pm 0.27$
session 3	train $= 0.91 \pm 0.14$ val $= 5.9 \pm 0.19$ test $= 5.5 \pm 0.30$
session 4	train $= 0.84 \pm 0.13$ val $= 5.44 \pm 0.11$ test $= 5.61 \pm 0.14$
session 5	train $= 0.94 \pm 0.12$ val $= 5.74 \pm 0.14$ test $= 5.45 \pm 0.12$
inter-session	train $= 0.75 \pm 0.09$ val $= 7.38 \pm 0.81$ test $= 7.35 \pm 0.70$

Table 11: Results obtained on the best models in case of raw images regression

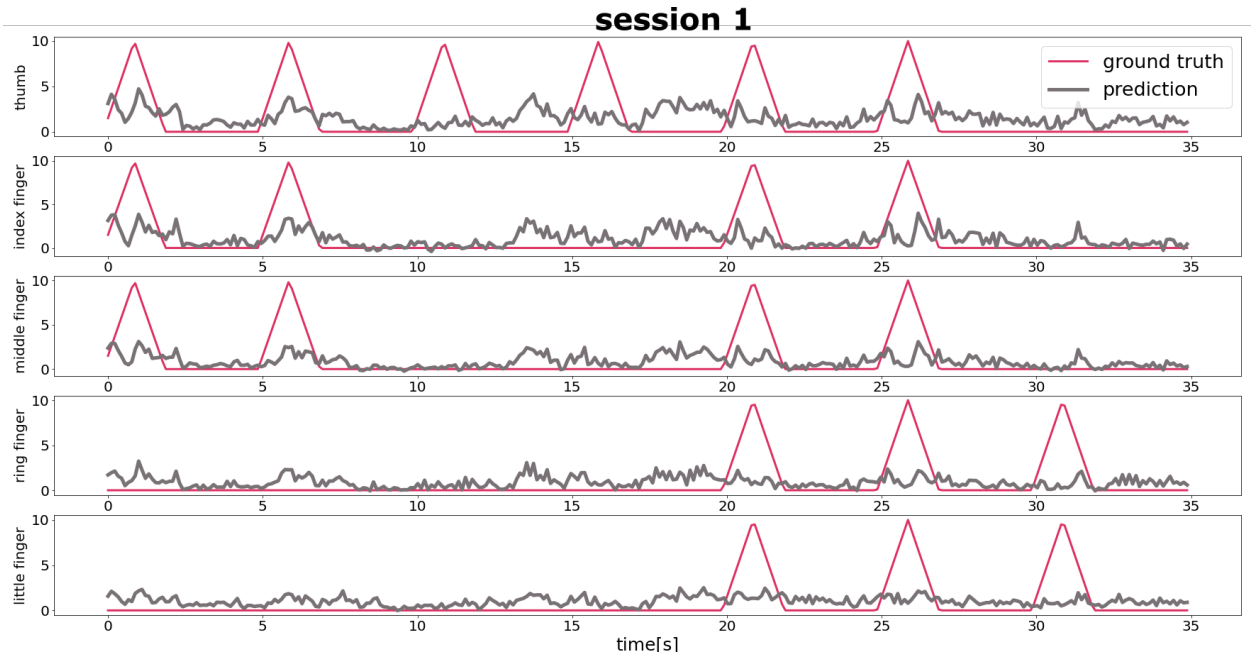

Figure 18: Regression results in case of raw images; intra-session (session 1 here, for session 3-5 please refer to Appendix); performance on testing set is shown

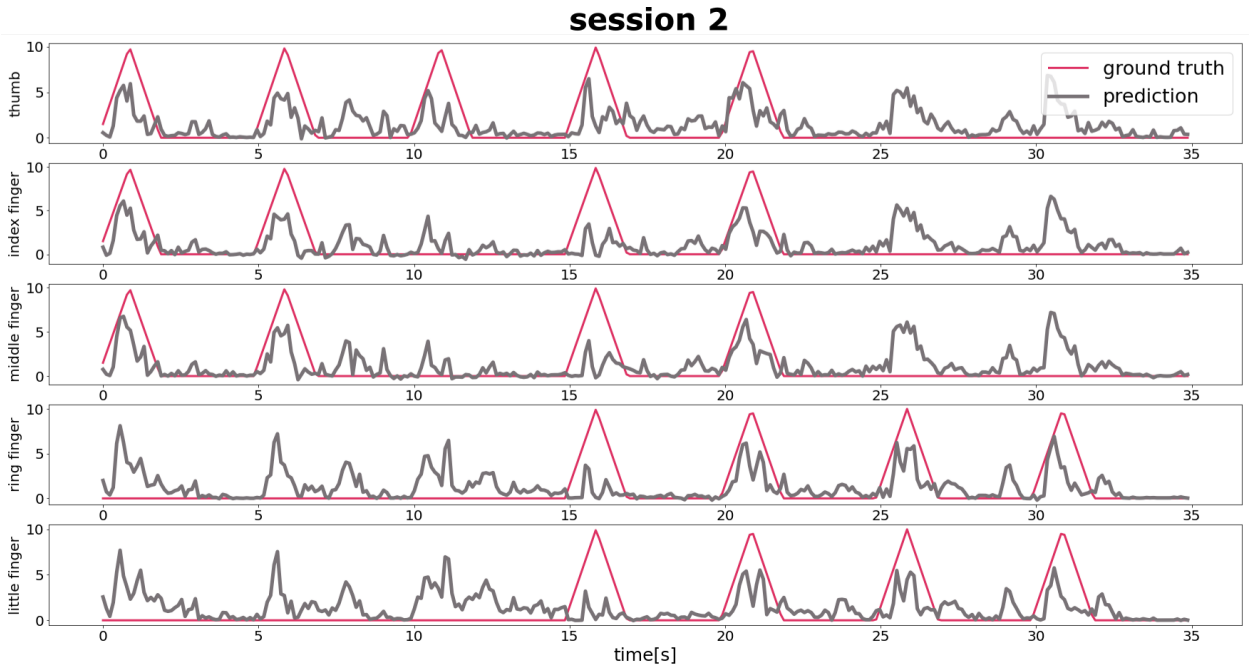

Figure 19: Regression results in case of raw images; intra-session (session 2 here, for session 3-5 please refer to Appendix); performance on testing set is shown

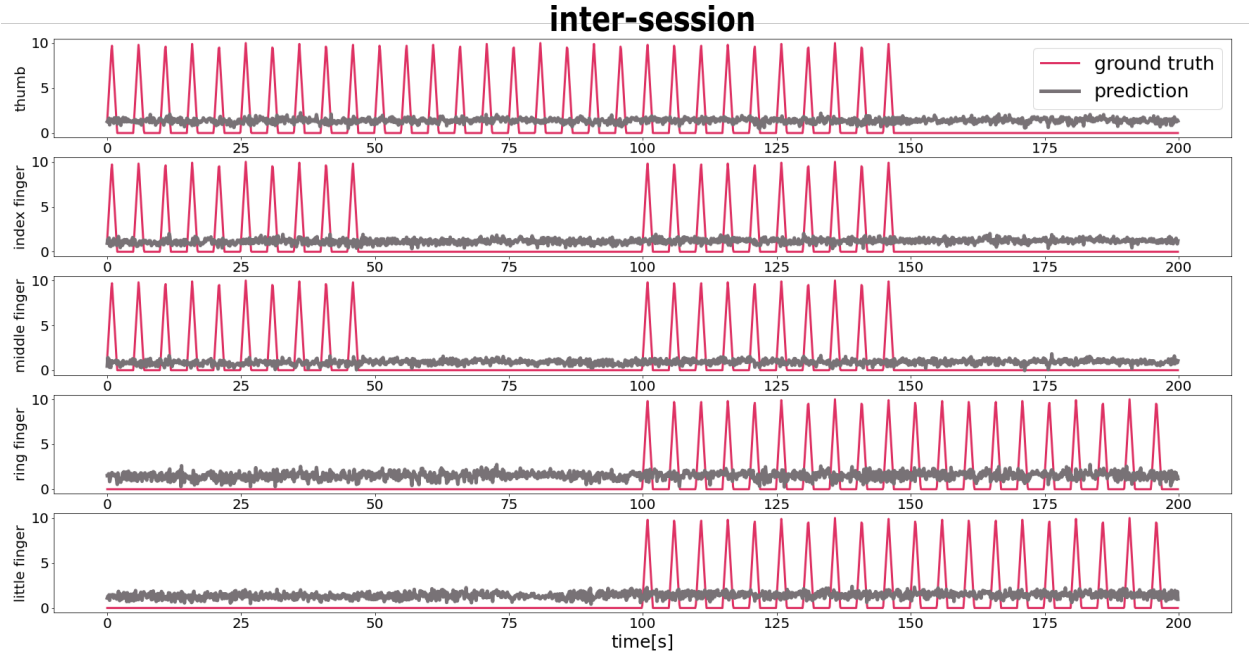

Figure 20: Regression results in case of raw images, performance on testing set is shown; inter-session

7 Discussion

Starting from 200 ms binned-RMS results, we can confirm results obtained [1] from Deep Learning perspective. Extending it to 1 s signals, we conclude that pattern formation exists in entire repetition of phantom hand movement. Therefore, it makes sense to use entire repetition and have more samples on disposal.

Moving further to inter-session framework, in this project differences in binned-RMS patterns between distinct experimental sessions are discovered. If regular split, performance on validation and testing sets is poor, while training is satisfying. This means model is able to learn on training set but it is not able to generalize the knowledge to the following experimental session. On the other side, if random-split, performance is promising even on validation and testing

sets, and it would probably become better if more complex model is used. However, the goal here was not to reach as high accuracy as possible, but to test whether the difference in two approaches exists. In real world application we would like to split the data regularly, as explained in the Introduction. In summary, this inconsistency in performance confirms that manifestation of motor intention on ENG signal exists, but it varies from one session to the next. We can argue this is the case at least in the binned-RMS signals case, and it is even more approved by the outcome of Kolmogorov-Smirnov test.

The KS test clearly points out to obstacle in terms of diversity of motor intention manifestation among distinct experimental sessions. Both mean and maximum value of KS score are significantly larger in case of single-session features for all binary combinations. Thus, when mixed from 5 sessions, features become less descriptive.

CNN classification on RMS images resulted in accuracy equal to prior. This means, the class is chosen by chance (in this case most often single class is predicted no matter the input) and no optimal weights of fully-connected layer are found. This may be due to the low resolution of images (16x28), and eventually no existence of sharp transitions and patterns. If this is the case, there are no features to be extracted by kernels, and no enough information is given to CNNs. Solution would be to create larger images, but that would induce longer delay in processing. To recall, in EMG literature optimal window length is considered to be 200 ms at most, because larger windows make noticeable delay in real time processing. Therefore, taking larger windows is not desirable.

CNN pipeline taken from EMG signals literature demonstrated low performance here, in both intra-session and inter-session approach. It is already shown signals do hold descriptive information, and high performance in at least intra-session case is expected. However, this is not the case. There are some indications of activity being described on testing set in intra-session case, but predicted output is always very similar in all 5 channels, independently of movement. As training loss is low (usually below 1), it possibly means CNN fits noise rather than useful information. This is probably due to the following fact: while EMG characterizes muscle activity, ENG does so with neuronal activity, i.e. it represents spikes. These signal are not considered as equivalent, from both physiological and signal processing point of view. Therefore, it make sense the pipeline created specifically for EMG signals is not able to address ENG signals processing. Note that, *MaxPooling* instead of *AveragePooling*, and normalization of data before creating images, and taking 90 % of data to train were tested, but none of those modifications improved the performance.

8 Conclusion

The goal of this project was to incorporate few aspects in order to explore TIMEs data potential. The binned-Root Mean Square of signals is calculated and used as input to Neural Networks. The classification in 9 classes is conducted and performance up to 84 % on testing set is reached while performing intra-session fashion classification. However, regular inter-session classification resulted in poor performance. By performing inter-session in random-fashion and Kolmogorov-Smirnov test, variability among distinct sessions is discovered. This means, for instance, RMS features representing specific class in session 1 and any other session are divergent. This is probably an answer on low performance in case of regular inter-session classification.

200 ms-long images are then created from 1D-RMS signals and Convolutional Neural Networks are used to perform classification. It turns out this approach is not able to adapt model parameters neither to fit the training set, probably due to low-resolution and low-size images.

Lastly, CNN feature-learning framework is adopted from EMG signals processing and applied to perform regression. The final outcome is low training loss but high validation and training losses, in both intra-session and inter-session case. The reason may be that too much noise is present on top of the useful information in raw signals, which makes CNNs fit noise.

In summary, we conclude that some kind of preprocessing is desirable, because CNN outcome points out the raw signals are superimposed by noise and NN approach on preprocessed signals results in high accuracy. Next, CNN pipeline applied on raw signals is not encouraging neither applied on RMS signals nor on raw signals. It may make sense to test its performance on less preprocessed data than RMS. For instance, just by applying filtering before giving an input to CNNs. RMS comes after few preprocessing and averaging steps, and therefore it may remove the information necessary to extract the patterns in CNN method. Consequently, one solution could also be to apply NN performance in feature learning-fashion.

Finally, to improve inter-session approach, fine-training can be applied in the case where intra-session works well. In real application, this would mean user need to calibrate the system before usage. Due to the time limit, it is not investigated in this project, but it make sense. Notably, taking binned-RMS signals and performing fine-tuning is encouraging, as high performance is reached in intra-session and randomly created inter-session case.

9 Literature

[1] M. Cracchiolo *et al.*, "Decoding of grasping tasks from intraneural recordings in trans-radial amputee, " *J. Neural Eng.*, vol. 17, no. 2, p. 026034, Apr. 2020, doi: 10.1088/1741-2552/ab8277.

[2] S. Micera *et al.*, "Decoding of grasping information from neural signals recorded using peripheral intrafascicular interfaces," *J NeuroEngineering Rehabil*, vol. 8, no. 1, p. 53, 2011, doi: 10.1186/1743-0003-8-53.

[3] M. Ortiz-Catalan, R. Brånemark, B. Håkansson, and J. Delbeke, "On the viability of implantable electrodes for the natural control of artificial limbs: Review and discussion," *BioMed Eng OnLine*, vol. 11, no. 1, p. 33, 2012, doi: 10.1186/1475-925X-11-33.

[4] S. Wendelken *et al.*, "Restoration of motor control and proprioceptive and cutaneous sensation in humans with prior upper-limb amputation via multiple Utah Slanted Electrode Arrays (USEAs) implanted in residual peripheral arm nerves," *J NeuroEngineering Rehabil*, vol. 14, no. 1, p. 121, Dec. 2017, doi: 10.1186/s12984-017-0320-4.

[5] P. M. Rossini *et al.*, "Double nerve intraneural interface implant on a human amputee for robotic hand control," *Clinical Neurophysiology*, vol. 121, no. 5, pp. 777–783, May 2010,
doi: 10.1016/j.clinph.2010.01.001.

[6] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "Regression convolutional neural network for improved simultaneous EMG control," *J. Neural Eng.*, vol. 16, no. 3, p. 036015, Jun. 2019, doi: 10.1088/1741-2552/ab0e2e.

[7] Leonardo Pollina, Lab Immersion I – project report, Translational Neural Engineering (TNE) lab.

[8] A. Diedrich, W. Charoensuk, R. J. Brychta, A. C. Ertl, and R. Shiavi, "Analysis of raw microneurographic recordings based on wavelet de-noising technique and classification algorithm: wavelet analysis in microneurography," *IEEE Trans. Biomed. Eng.*, vol. 50, no. 1, pp. 41–50, Jan. 2003,
doi: 10.1109/TBME.2002.807323.

[9] L. H. Smith, L. J. Hargrove, B. A. Lock, and T. A. Kuiken, "Determining the Optimal Window Length for Pattern Recognition-Based Myoelectric Control: Balancing the Competing Effects of Classification Error and Controller Delay," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 19, no. 2, pp. 186–192, Apr. 2011, doi: 10.1109/TNSRE.2010.2100828.

[10] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multifunction myoelectric control," *IEEE Trans. Biomed. Eng.*, vol. 50, no. 7, pp. 848–854, Jul. 2003, doi: 10.1109/TBME.2003.813539.

10 Appendix

10.1 Best-model parameters

Grid-search results, binned-RMS		
model	session	optimal parameters
RMS 200 ms	session 1	activation - relu drop rate = 0.1 hidden layer 1 - 64 hidden layer 2 - 128 hidden layer 3 - 256
	session 2	activation - relu drop rate = 0.15 hidden layer 1 - 80 hidden layer 2 - 80 hidden layer 3 - 128
	session 3	activation - relu drop rate = 0.05 hidden layer 1 - 80 hidden layer 2 - 128 hidden layer 3 - 128
	session 4	activation - relu drop rate = 0.2 hidden layer 1 - 40 hidden layer 2 - 128 hidden layer 3 - 256
	session 5	activation - relu drop rate = 0.25 hidden layer 1 - 64 hidden layer 2 - 80 hidden layer 3 - 128
RMS 1 s	session 1	activation - relu drop rate = 0.2 hidden layer 1 - 64 hidden layer 2 - 256 hidden layer 3 - 128 hidden layer 4 - 128
	session 2	activation - elu drop rate = 0.1 hidden layer 1 - 64 hidden layer 2 - 256 hidden layer 3 - 128 hidden layer 4 - 80
	session 3	activation - elu drop rate = 0.2 hidden layer 1 - 64 hidden layer 2 - 256 hidden layer 3 - 256 hidden layer 4 - 256
	session 4	activation - elu drop rate = 0 hidden layer 1 - 64 hidden layer 2 - 256 hidden layer 3 - 64 hidden layer 4 - 80
	session 5	activation - elu drop rate = 0.2 hidden layer 1 - 64 hidden layer 2 - 256 hidden layer 3 - 256 hidden layer 4 - 64
inter-session	regular split	activation - relu drop rate = 0.2 hidden layer 1 - 32 hidden layer 2 - 80 hidden layer 3 - 80 hidden layer 4 - 64 hidden layer 5 - 80 hidden layer 6 - 128
	random split	activation - elu drop rate = 0.2 hidden layer 1 - 64 hidden layer 2 - 80 hidden layer 3 - 80 hidden layer 4 - 128 hidden layer 5 - 128 hidden layer 6 - 256

Table 12: Parameters of best models in case of RMS-signals classification

Grid search results, RMS images	
session	optimal parameters
session 1	lambda - 0 drop rate = 0.4 hidden layer 1 - 64 hidden layer 2 - 32
session 2	lambda - 1e-2 drop rate = 0.1 hidden layer 1 - 64 hidden layer 2 - 24
session 3	lambda - 1e-5 drop rate = 0.1 hidden layer 1 - 64 hidden layer 2 - 32
session 4	lambda - 1e-1 drop rate = 0.3 hidden layer 1 - 64 hidden layer 2 - 64
session 5	lambda - 1e-1 drop rate = 0.2 hidden layer 1 - 64 hidden layer 2 - 24
inter-session	lambda - 1e-1 drop rate = 0.3 hidden layer 1 - 64 hidden layer 3 - 64 hidden layer 3 - 64 hidden layer 4 - 128 hidden layer 5 - 64

Table 13: Parameters of best models in case of RMS-images classification

Grid search results, raw images	
session	optimal parameters
session 1	lambda - 1e-1 drop rate = 0.5 hidden layer 1 - 32 hidden layer 2 - 32
session 2	lambda - 1e-5 drop rate = 0.1 hidden layer 1 - 32 hidden layer 2 - 16
session 3	lambda - 1e-1 drop rate = 0.3 hidden layer 1 - 32 hidden layer 2 - 32
session 4	lambda - 1e-4 drop rate = 0.2 hidden layer 1 - 64 hidden layer 2 - 24
session 5	lambda - 1e-1 drop rate = 0.5 hidden layer 1 - 32 hidden layer 2 - 32
inter-session	lambda - 1e-4 drop rate = 0.2 hidden layer 1 - 256 hidden layer 2 - 32

Table 14: Parameters of best models in case of raw-images classification

10.2 Best models illustration on test set

10.2.1 RMS signals intra-session

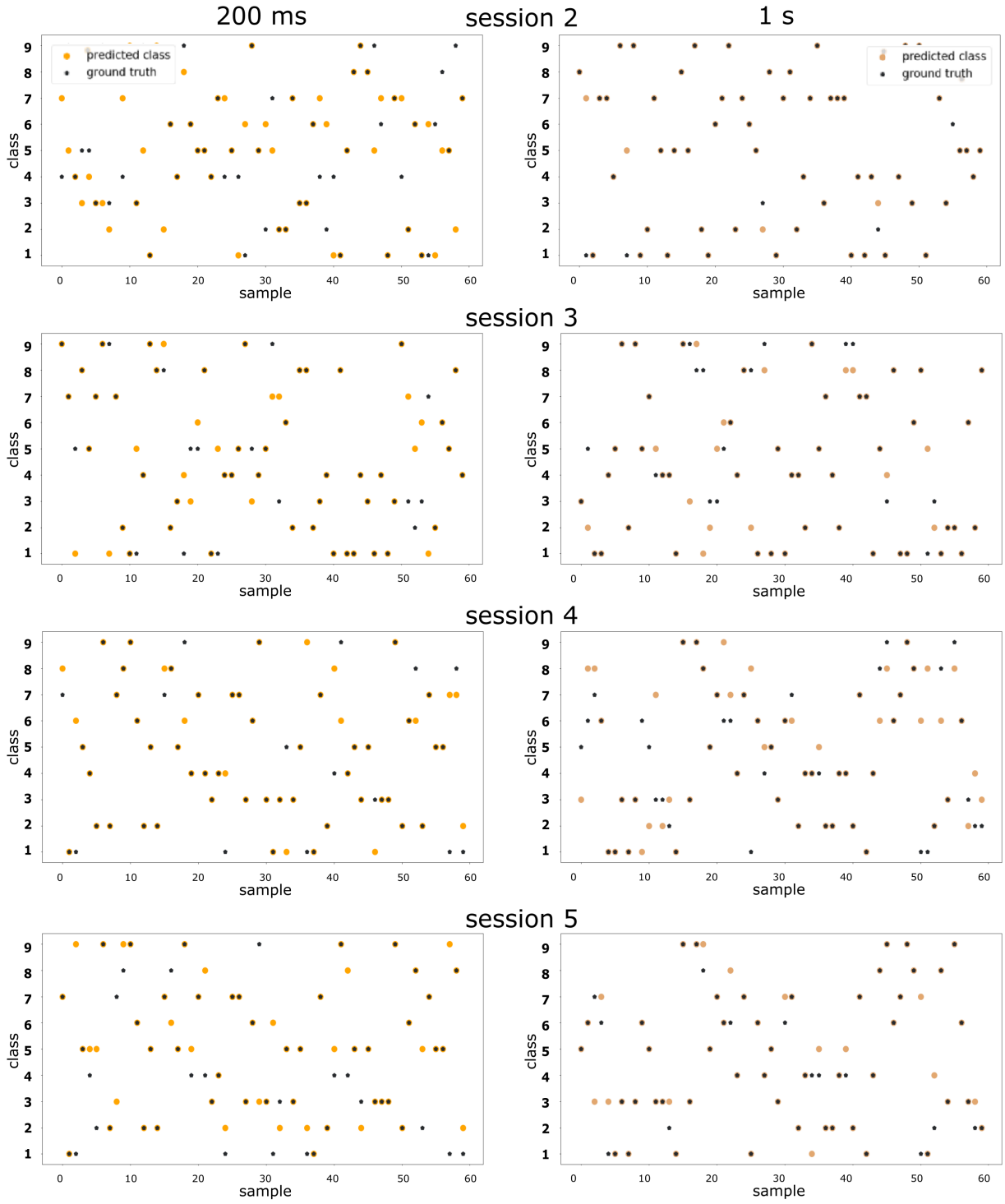

Figure 21: Illustration of results obtained in RMS intra-session case

10.2.2 RMS images intra-session

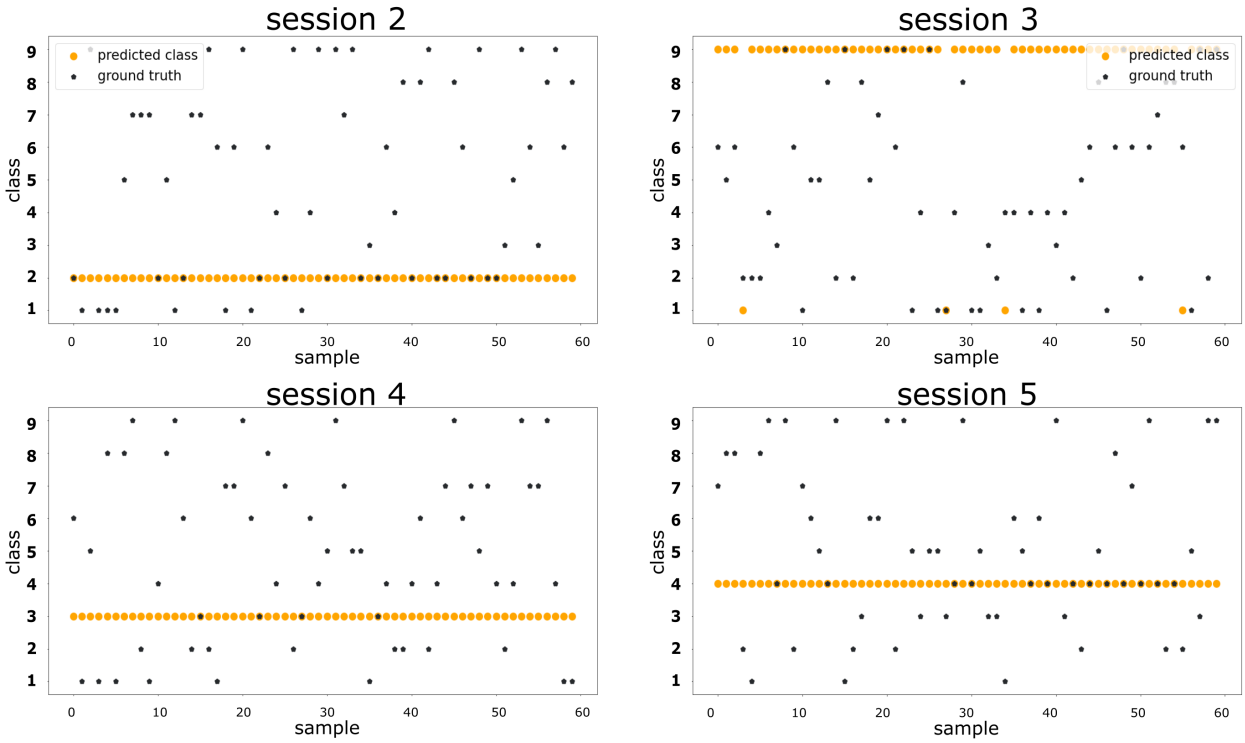

Figure 22: Illustration of results obtained in RMS-images intra-session case

10.2.3 Raw images intra-session

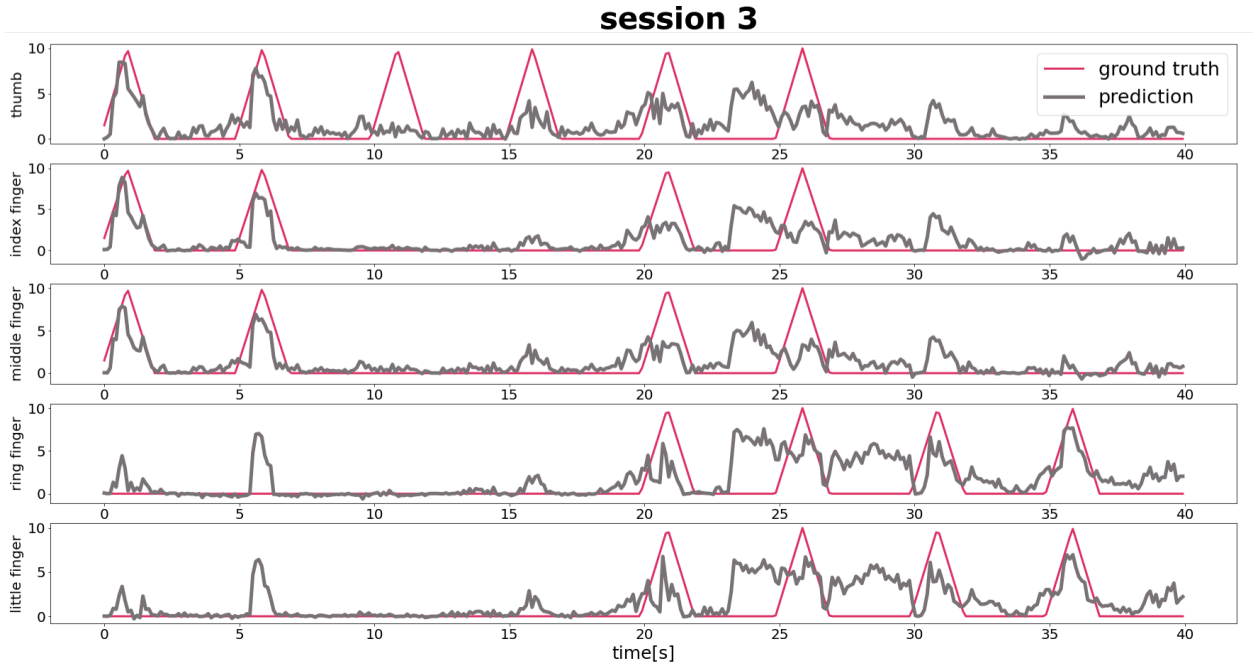

Figure 23: Regression results in case of raw images, performance on testing set is shown; session 3

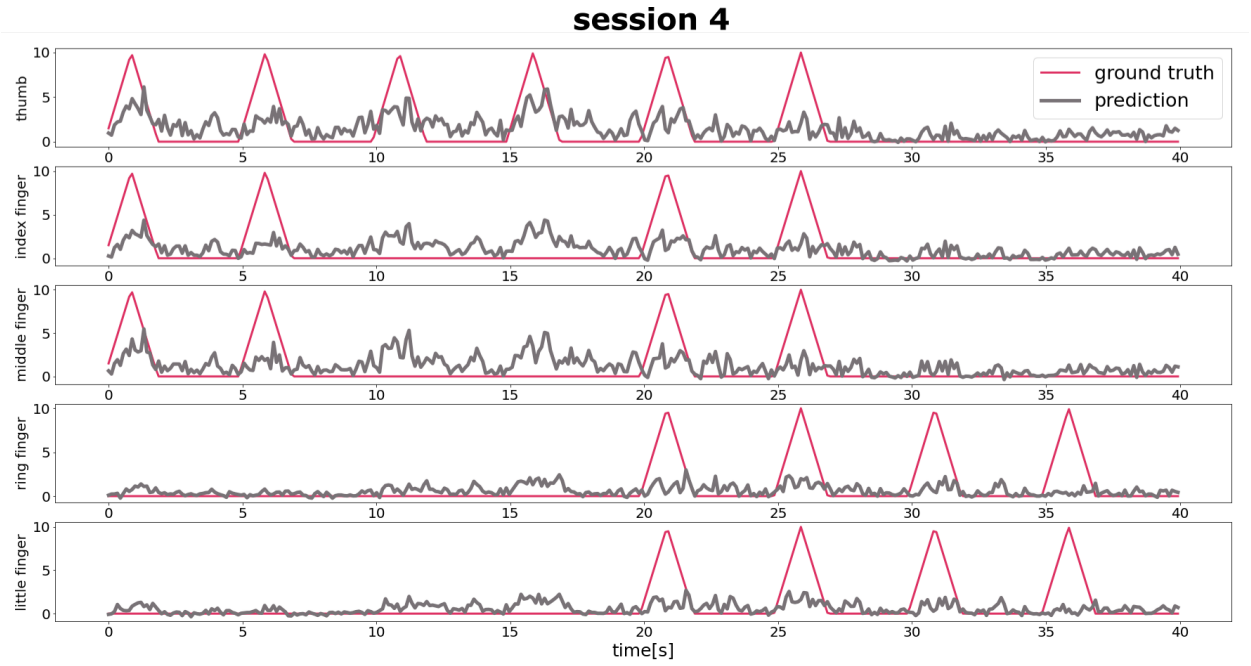

Figure 24: Regression results in case of raw images, performance on testing set is shown; session 4

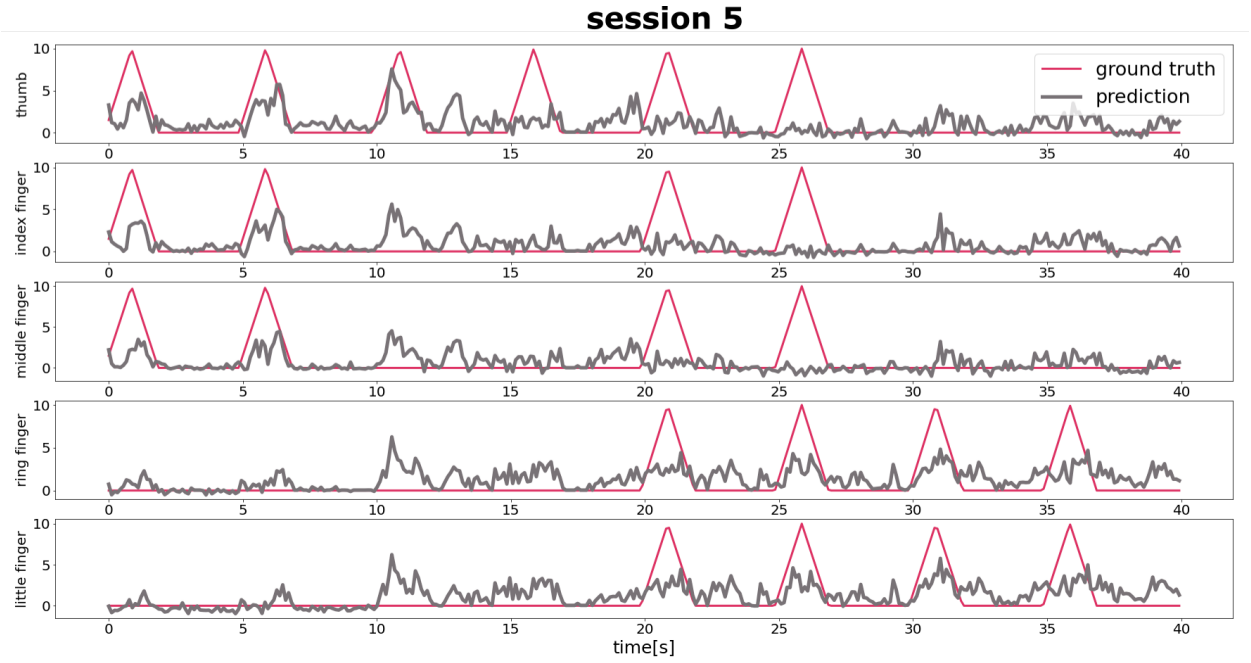

Figure 25: Regression results in case of raw images, performance on testing set is shown; session 5