UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

”PROCESSING AND CLASSIFICATION OF EVENT-RELATED POTENTIALS WITH SUPPPORT-VECTOR MACHINES”

TRABAJO DE FIN DE GRADO
Junio - 2021

AUTOR: Xavier-Alexy Moreno-Vassart Martinez
DIRECTOR: Vicente Galiano Ibarra

Contents

1 INTRODUCTION

1.1 INTRODUCTION AND OBJECTIVES

With data from electroencephalography, the goal is to classify and predict which event caused a person's evoked-potential response from a list of candidates.

Different hybrid approaches will be tested and compared. These will include different types of signal processing and learning algorithms such as support-vector machines.

Additionally, we'll also classify mental diseases using individuals' ERPs and validating support-vector machines.

1.1.1 INTRODUCTION TO SOME CONCEPTS

1.1.1.1 ELECTROENCEPHALOGRAPHY

The experimental setup that furnished the data for this study uses 19 sensors as shown in figure 1. This follows the standard `1020` distribution of sensors. These are worn by the test subject (fig. 2) for the length of the experiment.

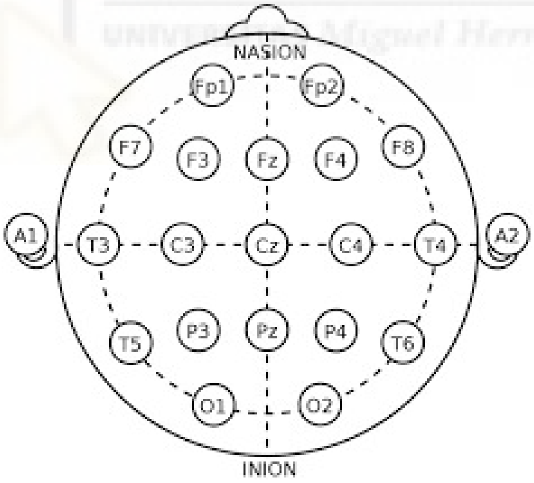

Figure 1: Electrode placement on the surface of the scalp

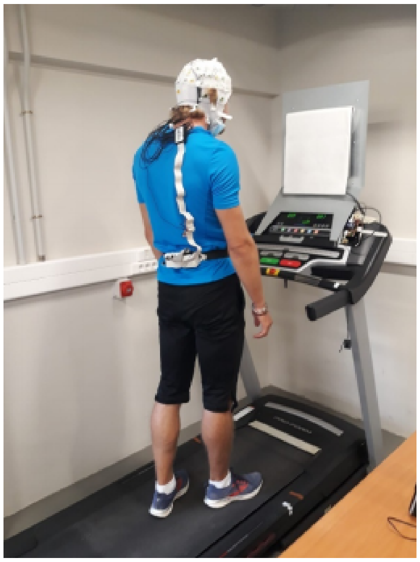

Figure 2: Experimental setup for electroencephalography

Electroencephalography (EEG) records electrical activity on the surface of the scalp. While it doesn't have a great spatial resolution, it has been proven enough to provide an estimation of the activity of the superficial cortex (1).

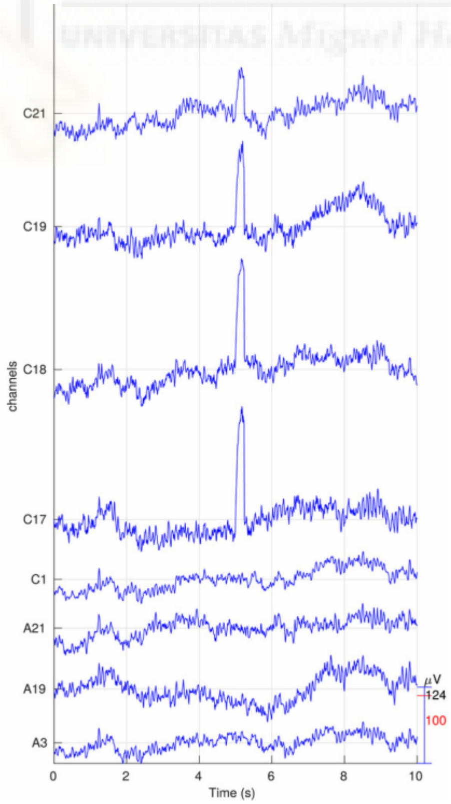

Figure 3: Raw EEG data before eye blink artifact removal

EEG is very sensitive to external noise and has a bad signal to noise ratio. The simplest technique to solve this issue is to average (2) multiple experiments.

Signals therefore need to be preprocessed to clean them of all these types of noises that include bio-signals and electromagnetic interference. The most common interference is the brain activity responsible for eye blinks and eye movement (3).

Independent component analysis (ICA) processing was used to clean the blink artefacts as visible in figure 3.

ICA processing is a method that separates the 19 signals in another set of subcomponent that are statically independent from each other. Once the new set of components is obtained, the components responsible for blinks, vertical and horizontal movement of eyes are eliminated (4). Finally the original signals are rebuilt from the set of independent components, stripped of the artefacts.

This is not the only way to remove glitches and other noises but it was the chosen technique for this study. It is often mixed with other approaches (3).

The results can be seen in figure 4.

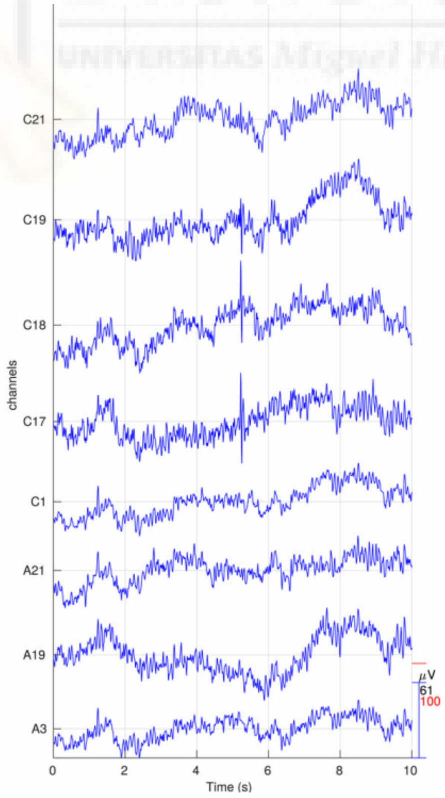

Figure 4: Raw EEG data after ICA processing

1.1.1.2 EVENT-RELATED POTENTIALS

An event-related potential (ERP) is the electric potential measured on the scalp that is a direct result of a specific stimulus or event. They are measured by electroencephalography (EEG).

The following figure (fig. 5) shows the ERPs of one of the individuals from four different events. They are overlapped on the same axis before any additional processing for machine learning.

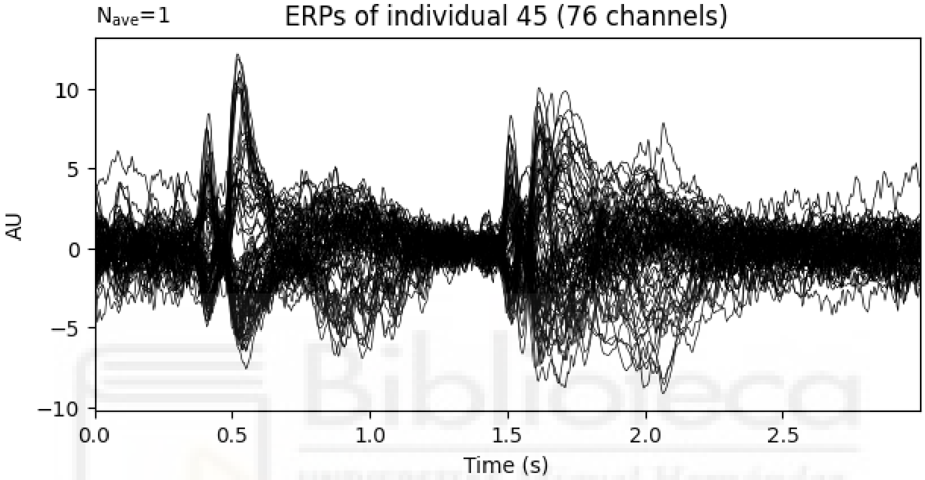

Figure 5: ERPs of individual 45

Electric potential is measured for 3 seconds and a sampling rate of 250Hz. We therefore have 750 samples per sensor, 4ms apart.

1.1.1.3 SUPPORT-VECTOR MACHINES

A Support-vector machine (SVM) is a supervised learning model that analyses data for classification and regression analysis (5). Given some data points in n-dimension, an SVM tries to find a hyperplane (fig. 6) of dimension n-1 that separates the data points. The support vectors are the data points of different class that are closer to the hyperplane.

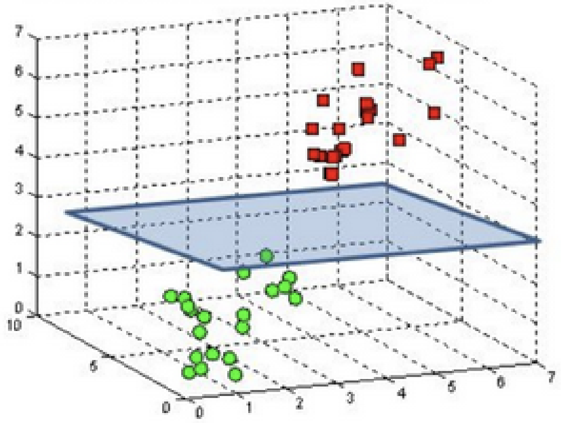

Figure 6: A hyperplane in 3D space is a plane

SVMs maximize the margin between the hyperplane and the support vectors of different classes.

To further improve the results, a *kernel function* (6) can be used. This allows the hyperplane to take new non-linear shapes and improve significantly the classification of non-linear distributed data.

1.1.1.4 ARTIFICIAL NEURAL NETWORKS

Artificial neural networks (ANNs, NNs) are a collections of nodes called neurons (7). They are connected by edges that have associated weights (fig. 7). These nodes are aggregated by layers, two of which are of particular importance, the input and output layers. Signals traverse from the input layer towards the output layer and get affected by the weights of every edge on the way. By adjusting the output layers we can make use of the ANN as a classifier (8).

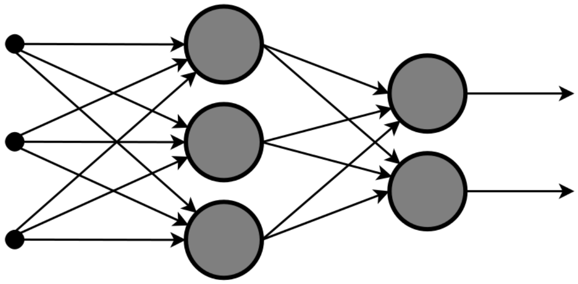

Figure 7: A neural network representation

To improve the predicted result we'll alter the weights of every edge using backpropagation (9), this is another method of supervised learning that we will compare to SVMs.

1.1.2 OBJECTIVES

This study has two main objectives. Both involve classifying ERP signals.

Firstly, we will classify different stimuli that provoked different responses on a group on individuals. The goal being to identify which stimulus was responsible for a given response.

Lastly, we will look into the differences between the ERPs of individuals with different mental disorders and compare them to the ERPs of a control group. Ideally we will be able to identity which group is the patient from, with nothing more but the data from EEG.

1.1.2.1 EVENT CLASSIFICATION

Our starting ground is EEG data corresponding to 85 individuals and 4 different events. These stimuli will be described as ***GO***, ***NO-GO***, ***no-answer*** and ***unexpected***.

As visible in fig. 8, the evoked response is similar for all individuals.

We can also visually compare it to the mean of all individuals (fig. 9). It looks like classifying events for unrelated individuals is feasible. Experimental results will confirm or deny this hypothesis.

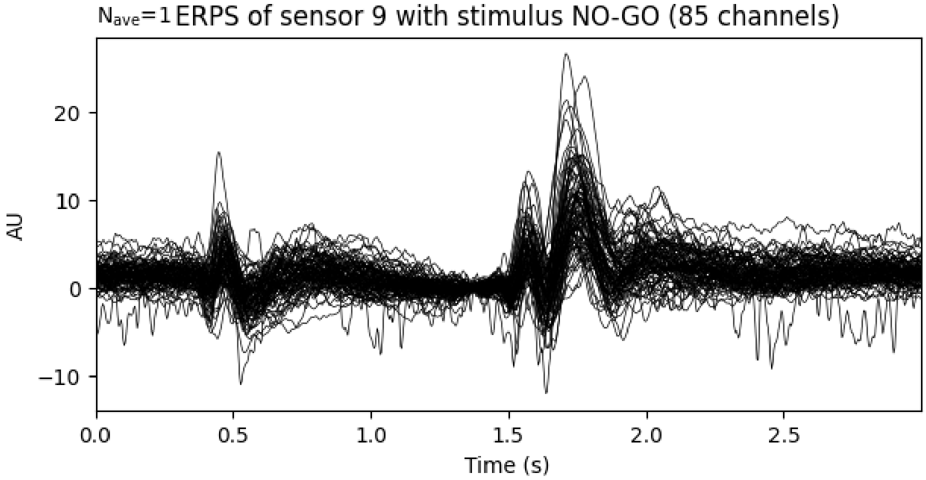

Figure 8: Overlapped ERPs of stimulus NO-GO at sensor 9 for 85 individuals

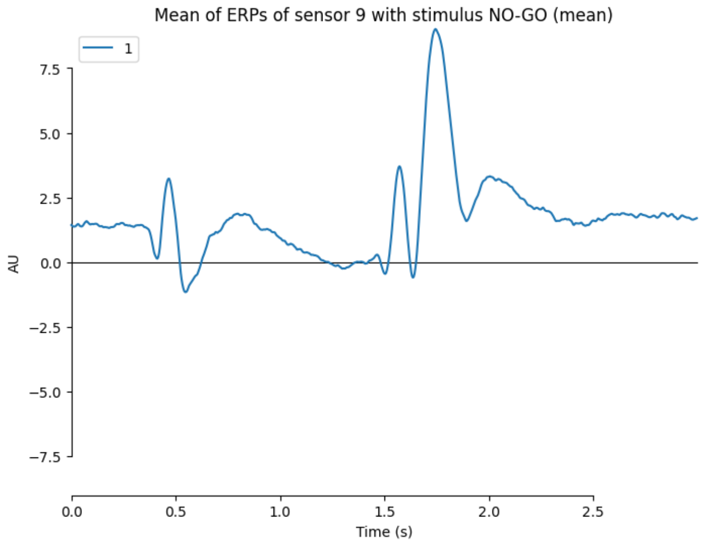

Figure 9: Mean ERP of stimulus NO-GO at sensor 9

We'll have to pre-process the data to normalise and standardise it (10) but also to reduce it's dimensionality (11).

The main method to reduce dimensionality is the use of root-mean-square error (RMSE). We will measure the difference of the signals of an individual with the mean signals corresponding to each stimulus.

Then we will compare the different processing techniques and different classification algorithms to come up with the most accurate one.

1.1.2.2 PATHOLOGY CLASSIFICATION

What if instead of grouping up ERP signals by the event that caused them, we compared all evoked-responses (the 4 previously mentioned) of individuals with different mental disorders?

We we'll use the same processing and dimensionality reduction techniques to compare the EEG data of healthy individuals and people with different neurological pathologies. We will call the group healthy individuals ***group N***, while the other will take letters starting from A.

2 MATERIALS AND METHODS

2.1 MATERIALS

For this research, we made use of a virtual private server (VPS) over secure the secure shell protocol (SSH). This allowed us to have an always ready online environement to work.

We made use of the Python programing language for all calculations (12). This includes all processing and machine learning. In particular we made use of the following external libraries: `mne`, `numpy`, `matplotlib`, `scikit-learn` and `scipy`.

2.2 METHODS FOR EVENT CLASSIFICATION

2.2.1 PRE-PROCESSING ERPS

The first step towards building a successful classifier is to normalise the data (10). To achieve this goal we designed the following process (fig. 10).

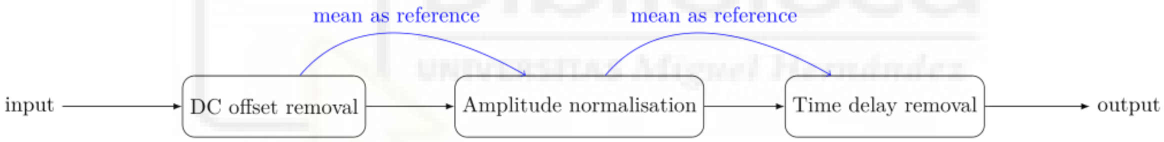

Figure 10: Initial pre-processing

The first step is to remove any DC offset the data might have. This might be due to different factors external or internal to the EEG measurements. Removing it entirely works as a necessary baseline for the following pre-processing algorithms.

The following step is to normalise the data. Since we have no prior reference for normalisation, the mean of all individuals used for this model serves this purpose well. The mean signal is then stored for processing of future data (as shown by fig. 11).

The last step is not required but will be shown (sec. 3.1.3) to improve results significantly. We will correct the time delay some signals might have with the use of cross-correlation (13). To calculate the cross-correlation in a time-effective manner we will make use of a fast-fourier transform (14).

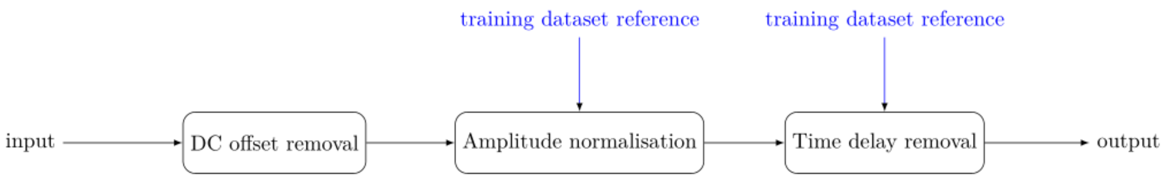

Figure 11: Pre-processing for new datasets

This step also requires a reference. The mean of all individuals after the previous processing steps (dc-offset removal and normalisation) was used. It will also be kept for processing of future data (fig. 11).

It might be controversial since data is already timed during the experimental setup for data gathering. And yet results (sec. 3.1.3) show it is beneficial.

As a first example let's have a look at a particular signal from a single sensor (fig. 12). The blue signal is the original. The orange one is the smoothed out version of the original data for ease of view and to be used as reference. The green one is the result of all stages of pre-processing. It is quite easy to tell it is centred around the x axes (after removing the DC offset); the maximum value it reaches is 1 (it has been normalised) and some time delay has been corrected. The red cross marks are local extrema that will be discussed at section 2.2.4.

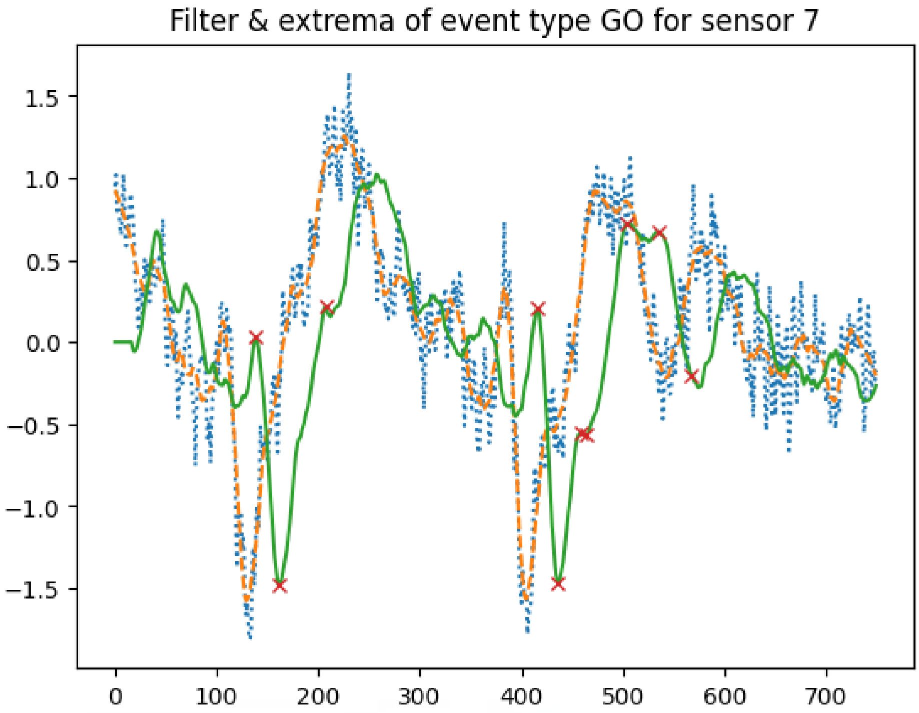

Figure 12: Processing stages of signal from sensor 7, event type *GO* and individual0

After all this signal processing we are left with a more meaningful dataset. The following figure shows the signals for a particular individual compared to the mean (fig. 13). Signals from all 19 sensors are shown topographically, in a manner that represents the physical placement of the sensors on the scalp.

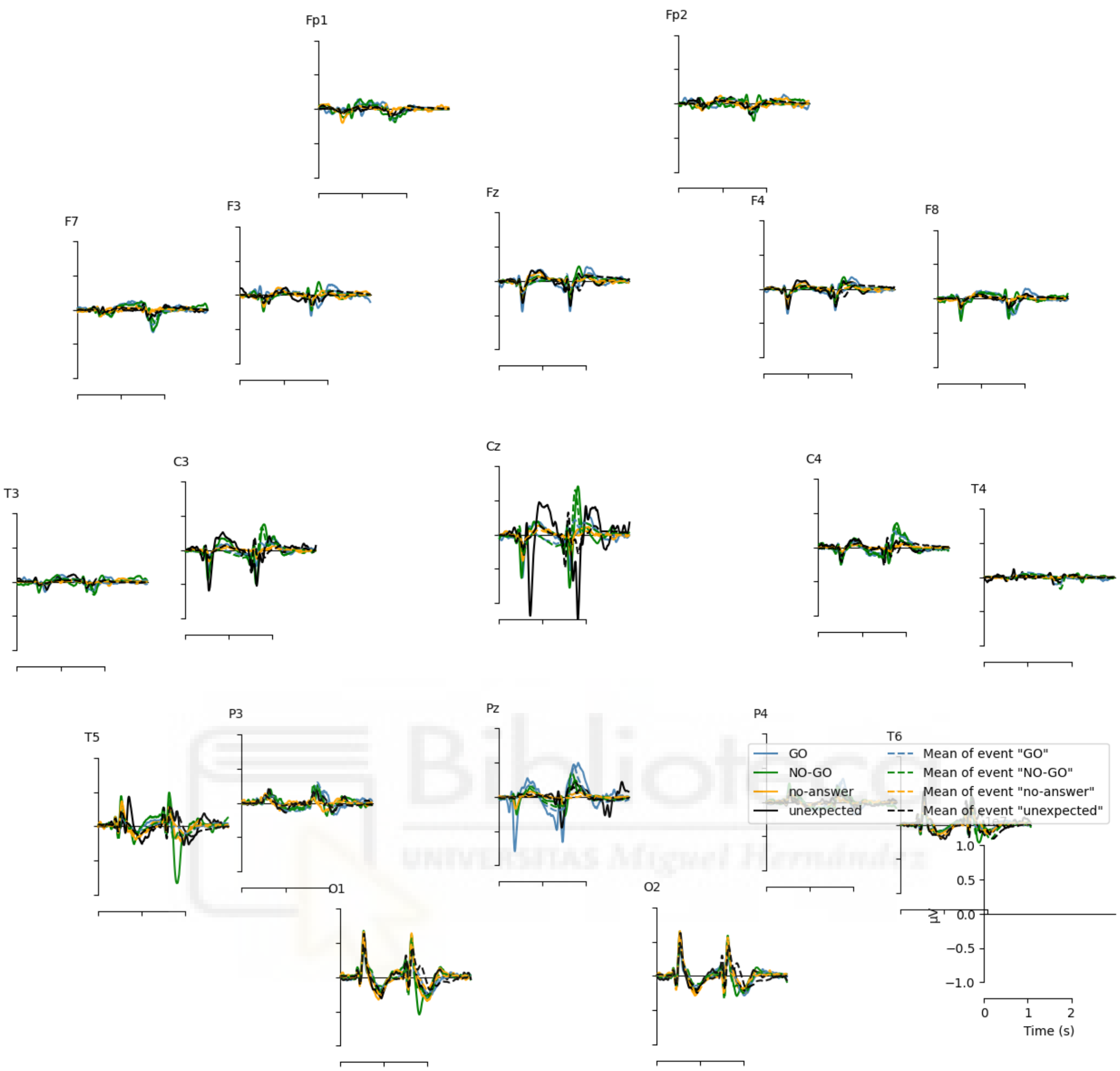

Figure 13: 2D topography of individual 0

Continuous lines represent the processed responses of said individual, while dashed lines represent the mean evoked responses.

This data, as shown, will be used for classification and compared to data of lower dimensionality.

As a better example, figure 14 shows the 2D topography of the processed data of event *GO* for a particular individual. This graph shows that despite being compared to the same mean event, there is some disparity. If we compare it to the previous 2D topography (fig. 13) we can observe the differences between the evoked response of all 4 events. Those differences are visible in this scenario, but comparing by hand a new signal would take time and wouldn't be a trivial task. This is exactly the process we want to automate with the proposed classifiers (sec. 2.2.5).

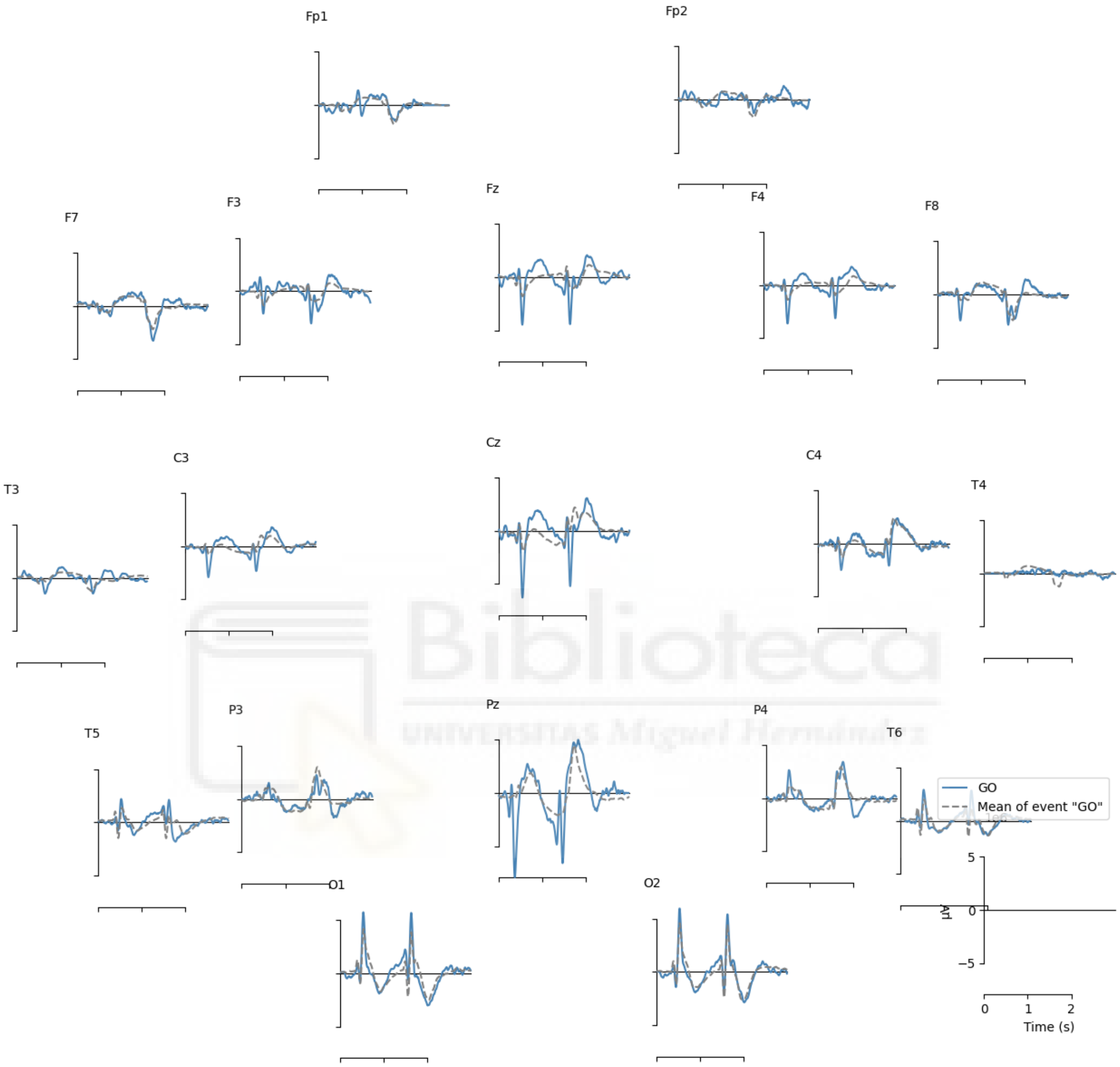

Figure 14: 2D topography of event *GO* for individual 0 compared to the mean

2.2.2 ROOT-MEAN-SQUARE ERROR AND MEAN ABSOLUTE ERROR

Root-mean-square error (RMSE) and Mean Absolute Error (15) (MAE) will be compared as means of reducing the dimensionality of the data at hand. Their definitions are available below (equations 1 and 2 respectively).

$$RMSE = \sqrt{\frac{\sum (y_i - y_p)^2}{n}} \tag{1}$$

$$MAE = \frac{\sum |y_i - y_p|}{n} \tag{2}$$

The RMSE and MAE are calculated individually for all 19 sensors. They consider the difference between each sample with the mean sample of all individuals for that given time period and then divided by the number of samples. This is evaluated for the mean of every response separately and gives us 4 positive floating values that characterise a person's evoked response.

To have a better understanding of how measuring error works with RMSE, fig. 15 shows the 4 RMSE values for all 4 of his evoked responses. As expected, the smallest value (closest to 0) is always the RMSE value that corresponds with the event that caused that evoked response.

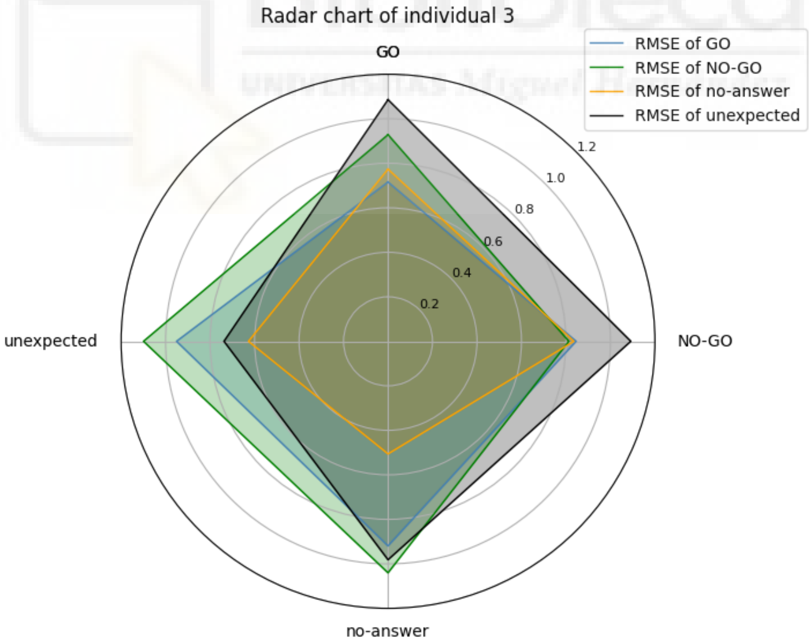

Figure 15: Radar chart of RMSE values for individual 3

If we have a look at the set of RMSE, neither a box-plot (fig. 16) nor a violin-plot (fig. 17) show clear results.

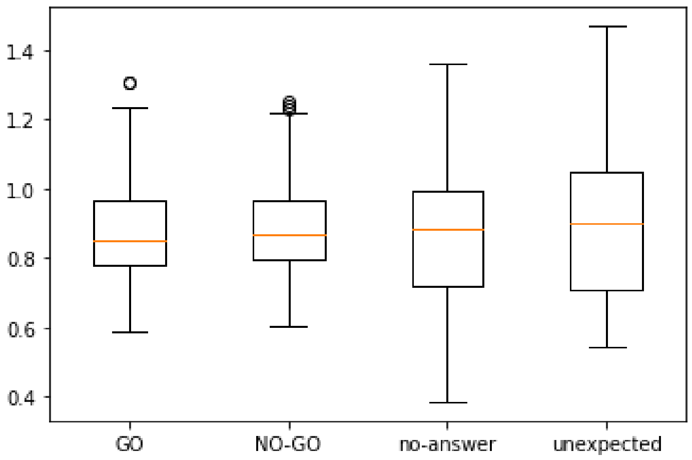

Figure 16: Box-plot of all RMSE of all 85 individuals and 4 events sorted by stimulus

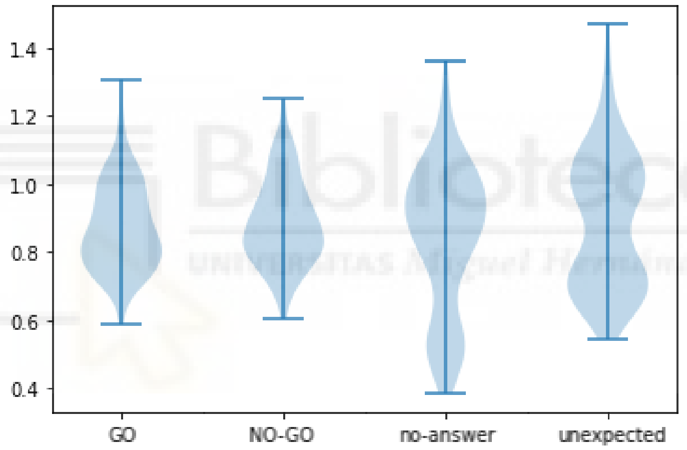

Figure 17: Violin-plot of all RMSE of all 85 individuals and 4 events sorted by stimulus

The violin-plot only shows an ideal distribution of values for the event *no-answer*. In this case there is a small group of RMSE values that group up closer to a value of 0 and the rest are generally higher. This is most likely because one forth of the RMSE values correspond to that type of event and are expected to result in lower values.

A possible explanation for why this behaviour isn't shared for all four groups, is that for a given person, all signals might look more alike and give 4 smaller RMSE values for all 4 events compared to a different individual. What really matters is (as seen in fig. 15) that there is one RMSE value of the quartet (corresponding to a single event) that is significantly smaller than the others, disregarding their relationship with the quartets obtained for the other individuals or events.

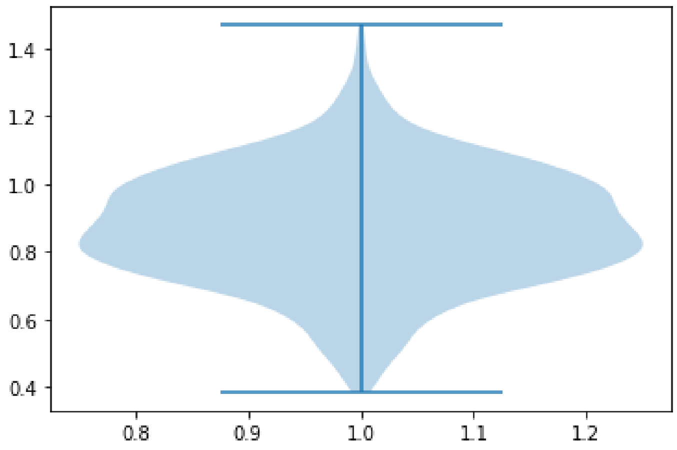

Figure 18: Violin-plot of all RMSE

All of these figures (figs. 16, 17, 18) show that no RMSE value has a preposterous value. They all are smaller than 1.5, most of also them being smaller than 1.2.

2.2.3 VISUALISING THE DATASET

To get a better grasp on the data we will represented in 2D. However we have 4 values of the sort: $RMSE \in [0; \infty[$ to represent a single point in the 2D plane.

To solve this, let's build a function F in a such a way that:

- proximity to $\begin{bmatrix}1\\1\end{bmatrix}$ shows how close a signal is to the average *GO* event
- proximity to $\begin{bmatrix}-1\\1\end{bmatrix}$ shows how close a signal is to the average *NO-GO* event
- proximity to $\begin{bmatrix}-1\\-1\end{bmatrix}$ shows how close a signal is to the average *no-answer* event
- proximity to $\begin{bmatrix}1\\-1\end{bmatrix}$ shows how close a signal is to the average *unexpected* event

As such this is function F is a pseudo-classifier that should return the smallest RMSE value as a prediction:

$$\begin{aligned}\vec{F}(\vec{RMSEs}) = \begin{bmatrix}1\\1\end{bmatrix} f(RMSEgo) + \begin{bmatrix}-1\\1\end{bmatrix} f(RMSEnogo) \\ + \begin{bmatrix}-1\\-1\end{bmatrix} f(RMSEnoans) + \begin{bmatrix}1\\-1\end{bmatrix} f(RMSEunexp)\end{aligned} \tag{3}$$

It can be rewritten as such:

$$\vec{F}(\vec{RMSEs}) = \begin{bmatrix}\begin{bmatrix}1\\1\end{bmatrix} & \begin{bmatrix}-1\\1\end{bmatrix} & \begin{bmatrix}-1\\-1\end{bmatrix} & \begin{bmatrix}1\\-1\end{bmatrix}\end{bmatrix} * f(\begin{bmatrix}RMSEgo\\RMSEnogo\\RMSEnoans\\RMSEunexp\end{bmatrix}) \tag{4}$$

To work as inteted, it needs to get converge to one of those points faster than it goes away from the others. This is achievable if:

$$\lim_{x \to +\infty} f(x) = 0 \tag{5}$$

$$\lim_{x \to 0} f(x) = 1 \tag{6}$$

Equation 5 can also be written as:

$$\lim_{x \to k} f(x) = 0 \tag{7}$$

Where k is the expected RMSE for a noise signal or, in a more practical approach, a high enough value that it's beyond any expected result (for example 1.7 with our previous data).

Additionally $f(x)$ is only defined for positive real numbers, and should be derivable have no local extrema.

The distance to each of the 4 points gives us the proximity to said event and the one with the minimum distance is the best guess.

$$\lim_{\vec{RMSEs} \to \begin{bmatrix}0 & k & k & k\end{bmatrix}} \vec{F} \approx \begin{bmatrix}1\\1\end{bmatrix} \tag{8}$$

$$\lim_{\vec{RMSEs} \to \begin{bmatrix} k & 0 & k & k \end{bmatrix}} \vec{F} \approx \begin{bmatrix} -1 \\ 1 \end{bmatrix} \tag{9}$$

$$\lim_{\vec{RMSEs} \to \begin{bmatrix} k & k & 0 & k \end{bmatrix}} \vec{F} \approx \begin{bmatrix} -1 \\ -1 \end{bmatrix} \tag{10}$$

$$\lim_{\vec{RMSEs} \to \begin{bmatrix} k & k & k & 0 \end{bmatrix}} \vec{F} \approx \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{11}$$

This is not a full proof. It only proves convergence for optimal values of $\vec{RMSEs}$.

Ideally all 4 vectors should be separated by the same angle.

We could have chosen $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\begin{bmatrix} -1 \\ 0 \end{bmatrix}$, $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $\begin{bmatrix} 0 \\ -1 \end{bmatrix}$ as position vectors and should expect similar precision-recall. The advantage of the chosen vectors is that we don't need to compute the distance. The quadrant the point is a part of already shows the prediction.

Let's try a couple of functions:

$$f(x) = \frac{\frac{1}{x}}{max(\frac{1}{x})} \tag{12}$$

$$f(x) = \frac{k - x}{k} \tag{13}$$

Using equation 12 we can build a 2D representation (fig. 19) shows how the points from event *no-answer* move towards $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$ more than the others. This goes according to what we observed in chapter 2.2.2.

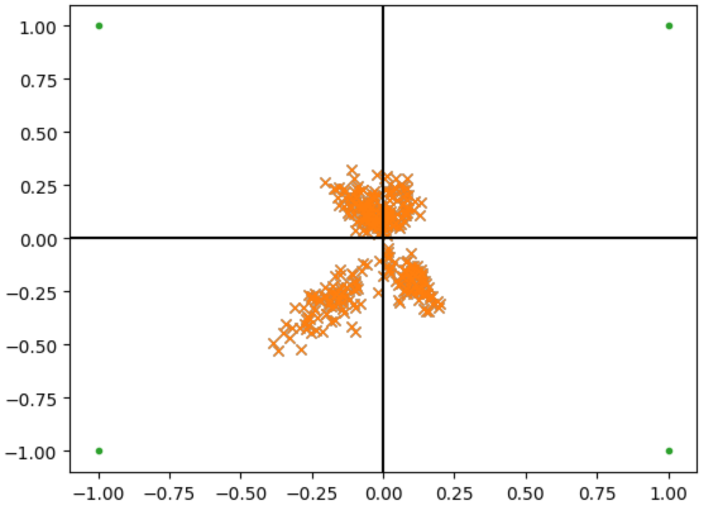

Figure 19: 2D representation of all RMSE values with f(x) from eq. 12

The 2D representation with equation 13 (fig. 20) shows even better results grouping up points more than before (fig. 19). This is a sign that our data might be well suited for machine learning classifiers.

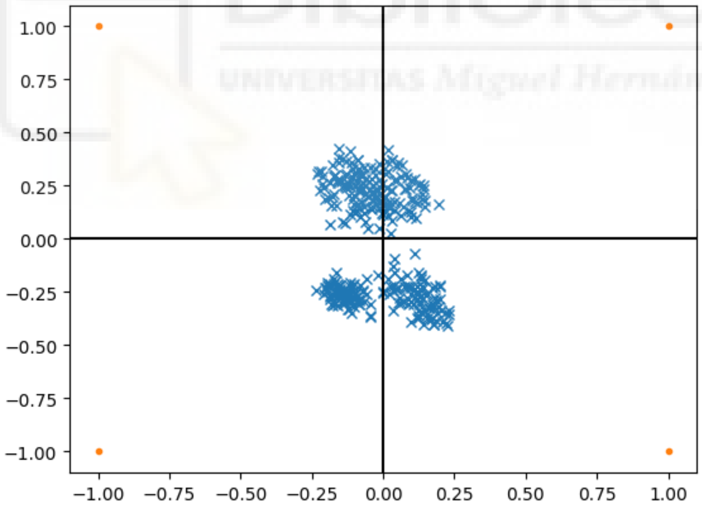

Figure 20: 2D representation of all RMSE values with f(x) from eq. 13

2.2.4 ALTERNATIVE DIMENSIONALITY REDUCTION WITH CHARACTERISTIC EXTREMA

As an alternative to RMSE and MAE, we could take a different approach inspired by neuroscience instead of signal processing.

This time we'll identify evoked potentials (16). These are the extrema of the ERP signals we have been working with. Both the time and amplitude of the extrema are characteristic. Some examples are the P1 visual stimulus or the P3 oddball response. This common notation uses P for positive extrema and N for negative ones.

To make use of these for classification, we will determine the most deviant extrema. The removal of DC offset is critical for this application. Normalisation shouldn't change the results much considering it is done on the overall signal. However removing any time-delay is potentially a very bad idea for this method of characterisation since it alters one of the 2 values that conform an extrema pair. It effectively alters the time values of all extrema.

To obtain a list of more relevant extrema we'll first smooth out the signal with a Saviztky-Golay filter (17) since averaging multiple experiments (2) is not an option.

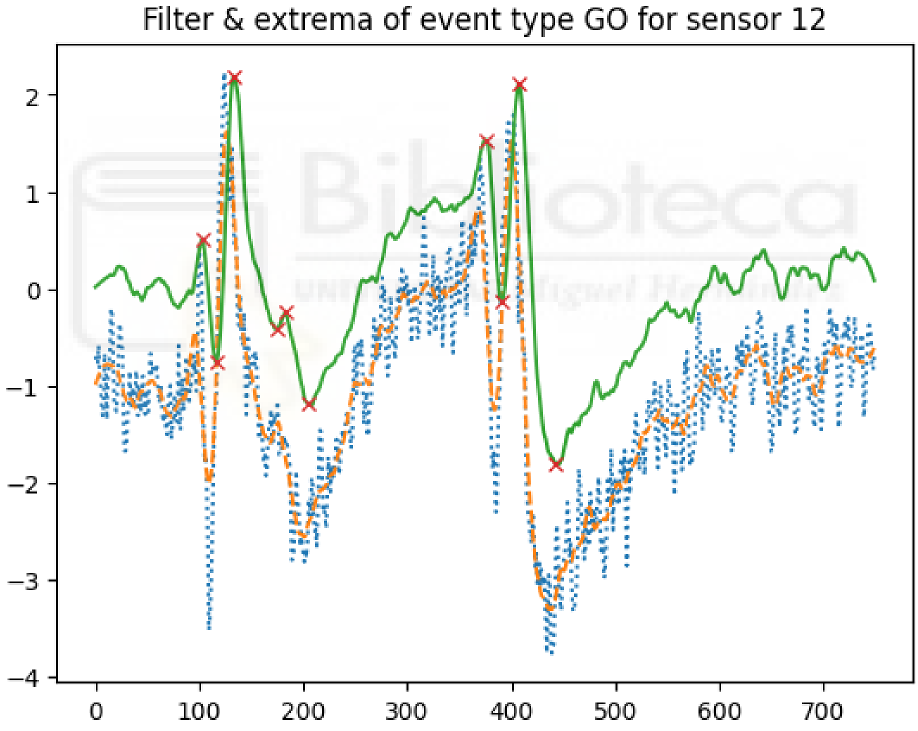

Figure 21: Characteristic extrema for sensor 12 with event *Go* and individual0

Once filtered obtaining all local extrema is relatively simple. As a final step, we need to sort the extrema by relevance. In this case we chose the extrema that had a bigger y-coordinate variation to the next or previous extrema. This method provided the best classifying results of all extrema-oriented methods.

The best amount of extrema coordinates to keep for classification was found experimentally to be 14 (sec. 3.1.2).

2.2.5 CLASSIFICATION METHODS

Before any machine learning we need to prepare the training and testing datasets. Since we plan on using different data (preprocessed data as is, ERP quartets and extrema values), we will generate training sets based on indices and not on the data itself. This ensure the splitting is the same for all the classifiers we build. It is a fairer way to compare them than splitting the data again for each one.

The data for both training and testing set is randomised before the split (figures 22, fig. 23). This random split is a safety precaution to prevent overfitting our predictors.

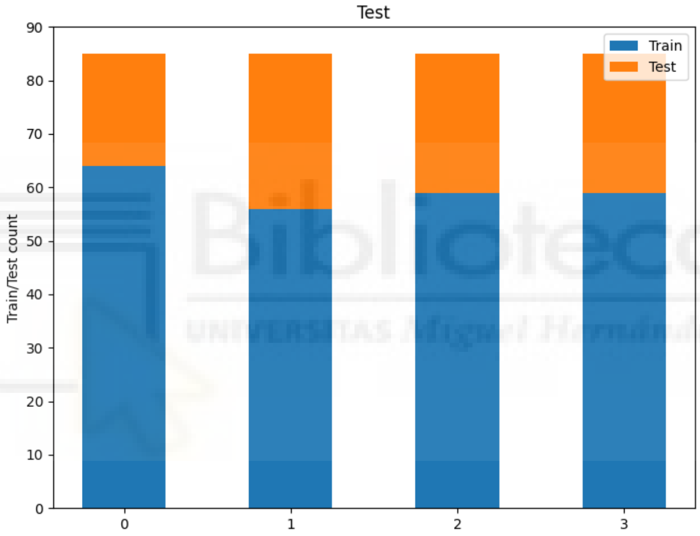

Figure 22: Training dataset distribution 1

We reserved 30% of the data for testing. This leaves us with 238 evoked-responses for training and 102 for testing, considering the 85 individuals and the 4 evoked-responses.

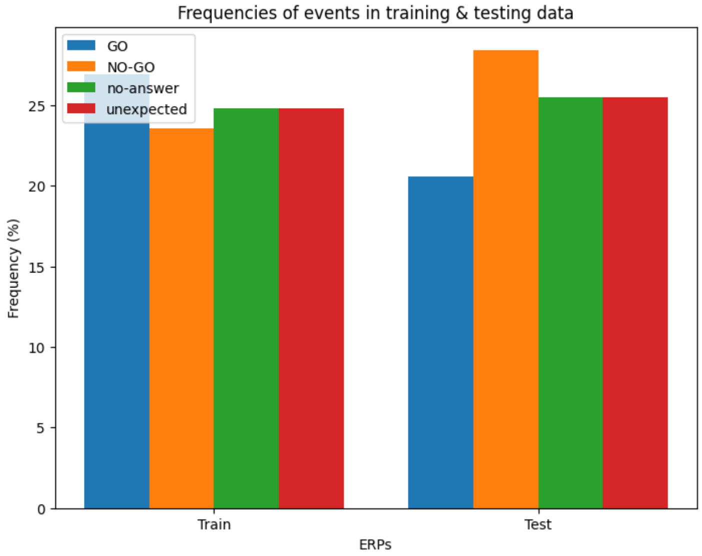

Figure 23: Training dataset distribution 2

Now that we have the data prepared for machine learning we have to asks ourselves what is the best approach for this classification task. We will compare different sets where:

1. RMSE are calculated on the average of all 19 sensors (4 values per evoked-response, 0.028% of total)
2. RMSE and MAE are calculated on the average of all sensors (8 values per evoked-response, 0.056% of total)
3. RMSE calculated for all 19 sensors separately (19x4 values per evoked-response 0.53% of total)
4. MAE calculated for all 19 sensors separately (19x4 values per evoked-response 0.53% of total)
5. averaged RMSE calculated for all 19 sensors (4 values per evoked-response 0.028% of total)
6. averaged RMSE and averaged MAE calculated for all 19 sensors (8 values per evoked-response, 0.056% of total)
7. extrema coordinates for all 19 sensors (19x14 values per evoked-response 0.53% of total)
8. processed signals averaged for all 19 sensors (750 values per evoked-response 5.26% of total)
9. processed signals for all 19 sensors (19x750 values per evoked-response 100% of total)

All of these will be tested with SVMs following a *One vs Rest* approach where one SVM classifier is built for every class (every event in our case) and compared to each other as a final step. Neural nets (also called multi-layer perceptrons - MLP) will also be compared to the SVM for cases 2 and 8. Lastly a second hidden layer architecture is tried for the last predictor (processed signals for all 19 sensors).

Once these models are fitted and tested, the best ones will be tried against new independent data from a different set of individuals.

The independent data was measured every 8ms (125Hz sampling rate), which ends up being 375 samples instead of 750 over 3s. To circumvent this issue these signals are all interpolated to fit the training dataset (750 samples over the same length of time).

2.3 METHODS FOR PATHOLOGY CLASSIFICATION

In a first attempt, we will build a classifier that will pick the right group from the total of 3. The pre-processing for ERPs from individuals grouped up by pathologies (including the normality group) is the same as before (sec. 2.2.1). As an alternative, we will consider the data of each pathology separate from each other. We will process each of them together with the normality group; to build smaller classifiers that only attempt to predict if a person has a given pathology or not.

Processing is therefore done twice and references kept, for the first batch and also for each pathology on the second batch.

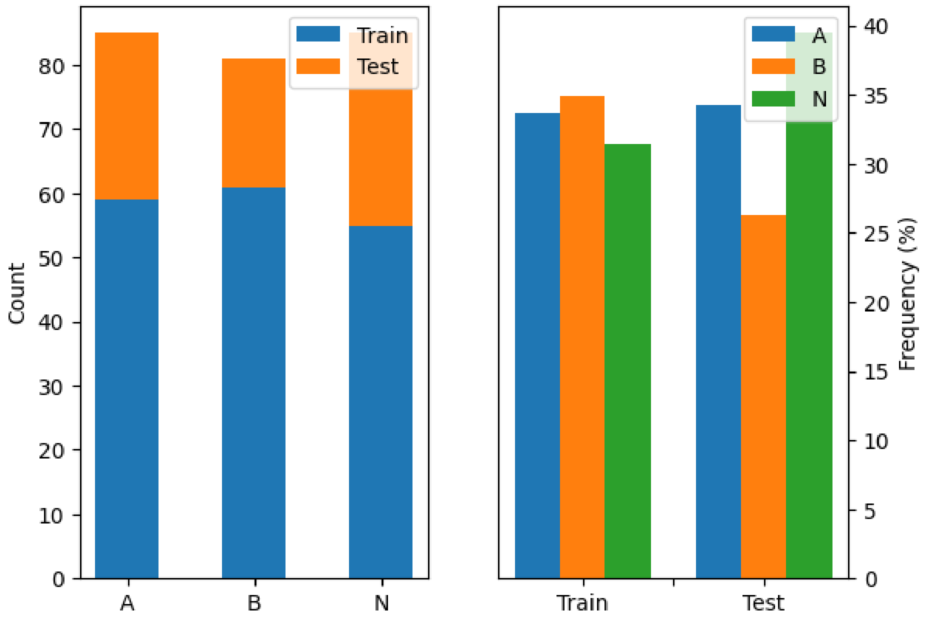

Figure 24: Frequency analysis of pathologies dataset

For both systems we will calculate the RMSE compared to their respective mean. In the first case, we will take the mean of all groups (including the normality group) while in the second case we will keep the mean of each pathology paired with the normality group. There is therefore a different mean for every sensor and for every event. Before we only had to keep track a mean (and thus RMSE value) for each stimulus.

This is the extent of the computations before applying machine learning models.

The data is split following the same ideas exposed in section 2.2.5.

This means a split for the first model (fig. 24) and a split per pathology for the second model (figures 25, fig. 26).

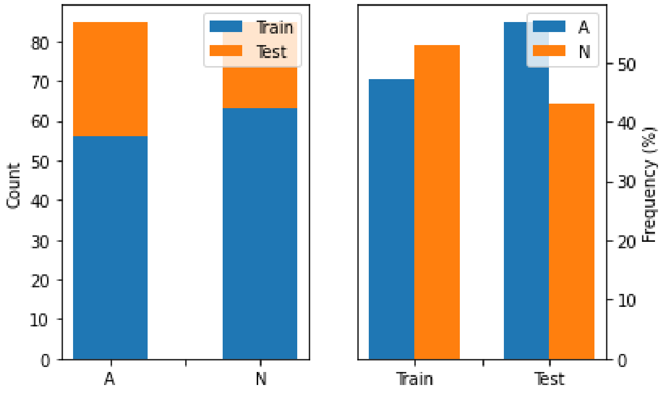

Figure 25: Frequency analysis of dataset for pathology A

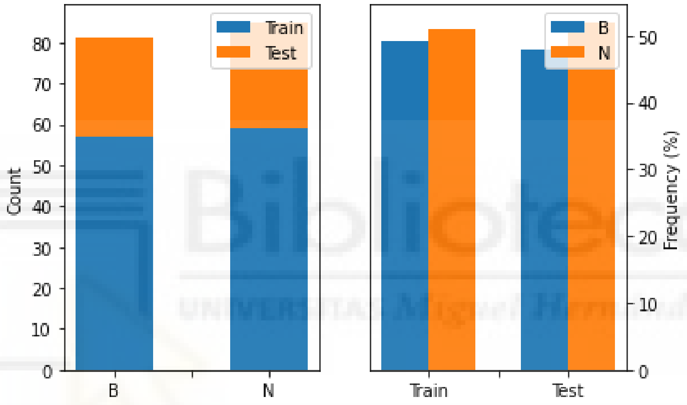

Figure 26: Frequency analysis of dataset for pathology B

We will build training and testing sets with preprocessed data as is (4x19x750 floating numbers per individual) and also RMSE based (4x19x3 considering we have groups for 2 pathologies and the normality)). Both will be modelled with SVMs and MLPs.

The same two sets of input data will be tried for the mentioned pairs of a pathology and the normality group. They will also be tried with SVMs and MLPs.

Neural nets have the added complexity of requiring to pick a structure of hidden layers that works efficiently for the data at hand. For the best of those classifiers, multiple hidden layers will be tested, to potentially find a model that surpasses the results of all others, including the SVMs.

These individual classifiers per pathology will be tried beforehand with the data preprocessed in bulk (the same data as for the multi-group predictor).

3 RESULTS AND DISCUSSION

3.1 CLASSIFYING EVENTS

This section will discuss the results of all the tested classifiers that aim to predict which event caused a given evoked-potential. For all SVMs the *rbf* kernel is used as it consistently provides much better results than *gaussian* or *linear* kernels.

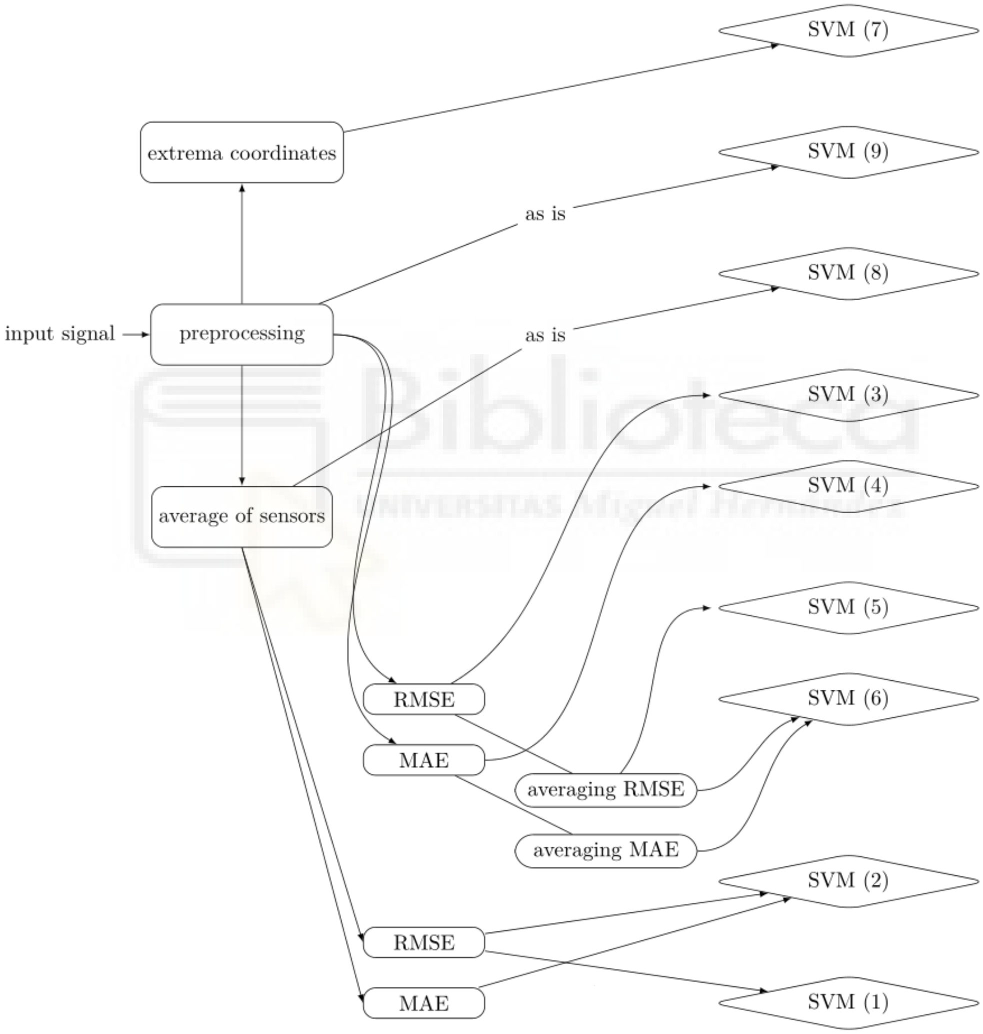

Figure 27: Classifier input as listed in sec. 2.2.5

Figure 27 displays all tested classifier inputs as they were listed in secion 2.2.5. We will first have a look at all the classifiers that take the average of sensors as inputs (as is but also with RMSE and MAE). Secondly, we will check the results from the classifier with extrema coordinates. Thirdly we will be comparing how a classifier (number 5 on the graph) behaves when we skip preprocessing steps. And finally we will compare the best classifiers from this graph. This final comparison will also include a variant of 2 of these (3 and 9) but making use of neural networks instead of SVMs.

Several confusion matrices will be displayed. These represent the predicted event in the vertical axes and the correct event in the horizontal one. It is an effective way of showing not only the probability of each prediction given a true event, but also the probability of a prediction to really be of a wanted class or another.

3.1.1 CLASSIFIERS ON AVERAGED SENSOR DATA

Let's check the confusion matrices of the SVM classifiers that work on averaged sensor data.

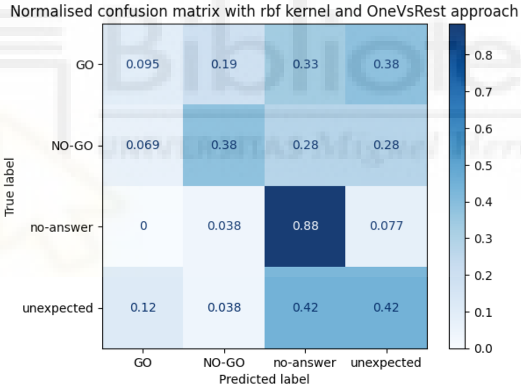

Figure 28: Confusion matrix of SVM with RMSE calculated on averaged sensor data

As shown on figure 28, the results are lacklustre. F1 score is only of 0.45 and the classifier only seems to have a grasp on the class *no-answer*. Figures 39 & 41 as seen in the appendices (sec. 5.1.1) show additional results based on averaged sensor data.

Either with RMSE and MAE data together or with the averaged values from the preprocessed signals, all those SVM classifiers give poor results.

3.1.2 CLASSIFIERS WITH EXTREMA COORDINATES

Classification with extrema coordinates was not very successful as shown on figure 29. The best F1 score achieved was 0.69 without normalisation. This is far from the results with RMSE values or the preprocessed signal (as shown in the following section 3.1.4).

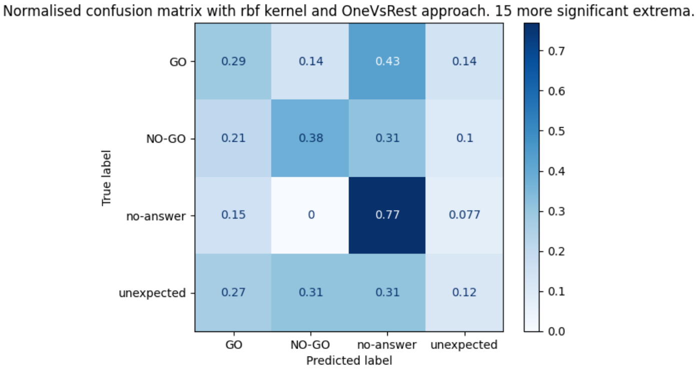

Figure 29: Confusion matrix of SVM on extrema coordinates with delay removal

The best results were achieved with 14 extrema values for non-normalised data while it was 15 for normalised data. Multiple SVMs were automated and the best was kept.

3.1.3 COMPARING PRE-PROCESSING METHODS

We will now compare the same SVM with different pre-processing. The goal is to determine if correcting time-delays is helpful.

This comparison will be done with a classifier that takes averaged RMSE values for every sensor. In other words, 19x4 RMSE are calculated and these RMSE values are averaged for each event. This effectively reduces the data per class to 4 floating values (0.028% of total values per evoked-response) as with the RMSE over averaged sensors. This method however, doesn't remove that much information and gives satisfying results.

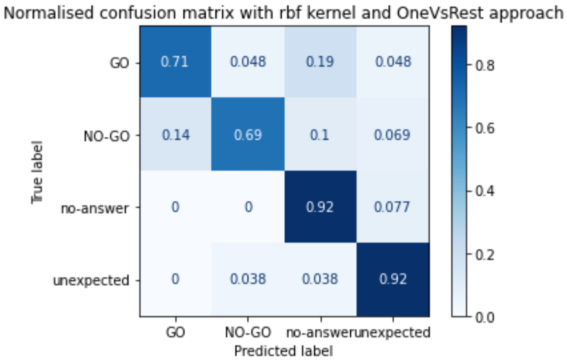

Figure 30: Confusion matrix of SVM with averaged RMSE with only DC removal

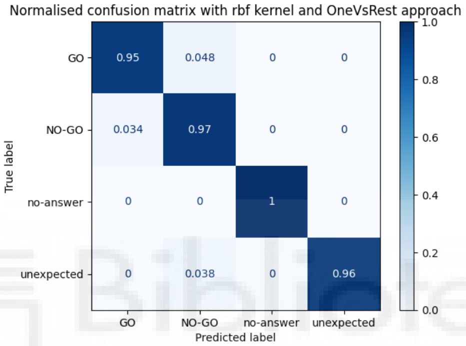

Figure 31: Confusion matrix of SVM with averaged RMSE with DC removal and normalisation

As seen on figures 30, 31 & 32, normalisation is critical. It improves predictions very significantly.

Time-delay correction doesn't offer such a dramatic increase but is nonetheless a net benefice. It allows us to reach an F1 score of 1.0 with only 4 values per evoked-response.

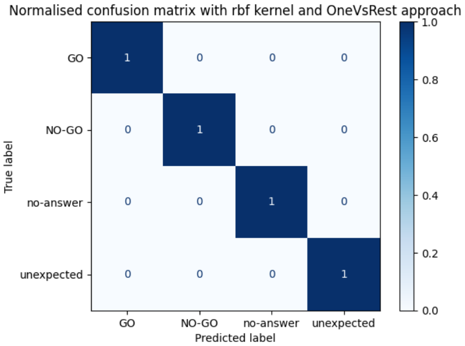

Figure 32: Confusion matrix of SVM with averaged RMSE with DC removal, normalisation and time-delay removal

3.1.4 COMPARING THE BEST CLASSIFIERS

Now that we have achieved good results it is time to compare the best SVM classifiers and their neural net counterparts. We will measure the F1 score but also the fitting and prediction time.

All the confusion matrices can be found in section 5.1.2.

They all seem to provide decent results. Some details tell them apart, not all of them achieve an F1 score of 1.0 and some take much longer to train and predict. It is very likely that with better selected hidden layer structures neural nets could improve their results and potentially their fitting time.

Table 1: Comparison of tested classifiers

Classifier	Fitting (ms)	Prediction (ms)	F1	F1 (new data)
SVM Average RMSE (4)	5.44 ± 0.362	1.28 ± 0.107	1.00	0.99
SVM RMSE (4x19)	10.7 ± 0.891	3.37 ± 0.083	1.00	0.99
SVM RMSE&MAE (8x19)	12.9 ± 0.719	5.37 ± 0.106	1.00	-
SVM MAE (4x19)	10.2 ± 0.517	3.13 ± 0.040	0.99	-
SVM RAW (19x750)	1580 ± 101	1690 ± 75.1	0.98	0.99
MPL RMSE (4x19)	653 ± 25.9	1.43 ± 0.051	1.00	1.00
MPL RAW (19x750)	2210 ± 35.2	120 ± 1.1	0.97	0.96
MPL RAW 2 (19x750)	2210 ± 65.8	117 ± 0.454	0.99	0.98

On first approach we can see that multilayer perceptrons have on average a higher fitting time that grows with the dimensionality of the data to classify.

Simple classifiers (smaller dimensionality) shows the best improvement in fitting time with SVMs as opposed to neural networks (1.6% of the time for RMSE values for all sensors). With higher dimensionality, the fitting time is still longer but is of the same magnitude (71% of the time). For complex models, neural networks are no longer at such a big disadvantage.

They provide, however, a much lower prediction time that is also proportional to the dimension than SVMs (since prediction with neural networks is mostly matrix multiplication). For the lower dimensionality the MLP takes 65% of the time of the SVM while for higher dimensionality this ratio goes down to 6.9%. Prediction time scales much better with complexity for neural networks.

The last column of the table refers to the next section 3.1.5.

3.1.5 TESTING CLASSIFIERS ON NEW DATA

Prediction results with new and independent data from a different set of 82 individuals are very close, and almost as good as they were with the original testing dataset. As mentioned before (see fig. 11), the reference signals used for these new individuals are the same that were computed with the training data.

The F1 scores are shown in table 1 and the confusion matrices are available in section 5.1.3.

The neural net that uses the RMSE values for all 19 sensors achieved the ideal F1 score of 1.0. This is particularly promising considering the data was interpolated beforehand to adjust it to the training data. We should expect similar results with correctly measured ERPs.

3.2 CLASSIFYING PATHOLOGIES FROM A SET OF ERPS

3.2.1 MULTIPLE GROUPS IN A SINGLE CLASSIFIER

This first approach shows slightly better results with SVM than neural networks. This is true both for predictors that use all the data points of the processed signals and for RMSE. Results are lacklustre but better on processed signals. RMSE classifiers are too close to a random pick to be of any use.

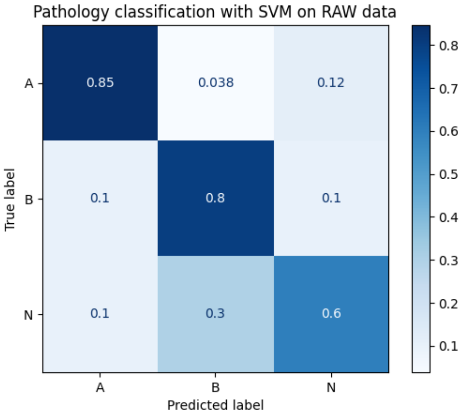

Figure 33: Confusion matrix of pathology classification with SVM on all preprocessed signals

All 4 confusion matrices are available at sec. 5.1.4.

The best F1 score achieved with this line of work is 0.74. It shows results are not random and could potentially be improved, but this method doesn't scale well. Training would have to start from scratch for every new pathology to test for. Optimizing the neural network layers would also have to be done again.

3.2.2 CLASSIFIERS FOR EVERY PATHOLOGY (GROUPED UP WITH NORMALITY)

3.2.2.1 PRE-PROCESSING IN BULK

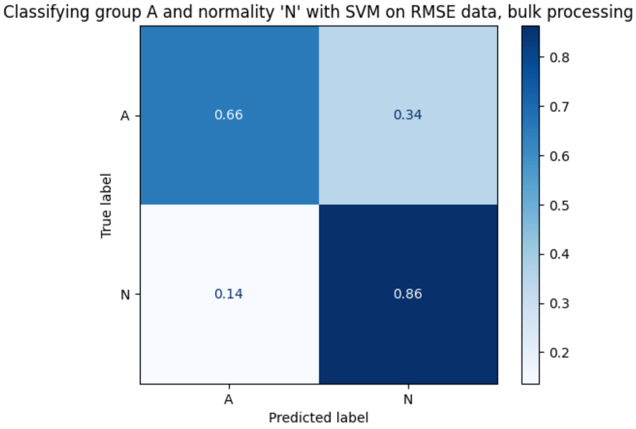

Figure 34: Confusion matrix of SVM classifier for group A and bulk preprocessing on RMSE

This figure (fig. 34) shows how hard it is to identify groups when data has been preprocessed in bulk. The rest of the confusion matrices are avaible at section 5.1.5.

Group B is being troublesome to classify, all methods show results worse than random, they are in fact picking wrong options more often than not.

Overall, predictors on full preprocessed data behave better than on RMSE data while still being unacceptable.

3.2.2.2 INDIVIDUAL PRE-PROCESSING

sec. 5.1.6

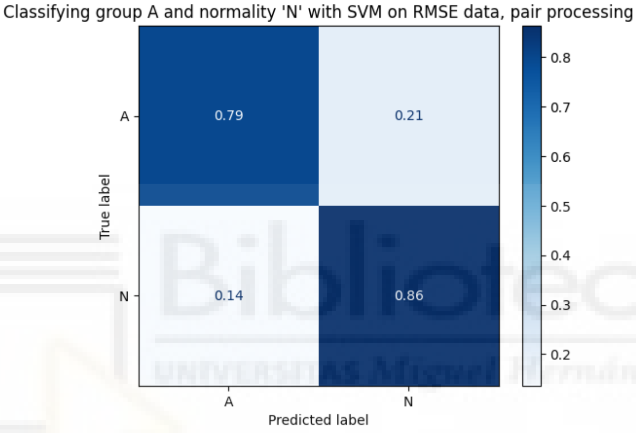

Figure 35: Confusion matrix of SVM classifier for group A and individual preprocessing on RMSE

These new batch of classifiers achieves much more interesting results. SVM results are around 0.8 F1 score for both groups while neural nets reach 0.9 for group A but struggle a bit with group B (fig. 35 shows one of them, the rest are at section 5.1.6.

And while 0.8 shows groups can be identified, it can be improved. The following two figures show the classification matrices after optimising the hidden layer structure (figs. 36, 37).

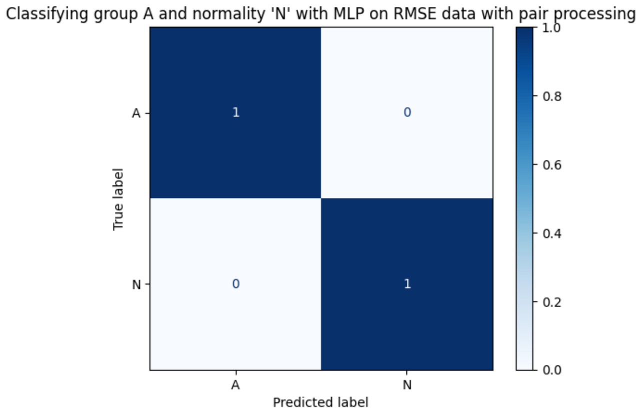

Figure 36: Optimised confusion matrix of MLP classifier for group A and individual prepossessing on RMSE

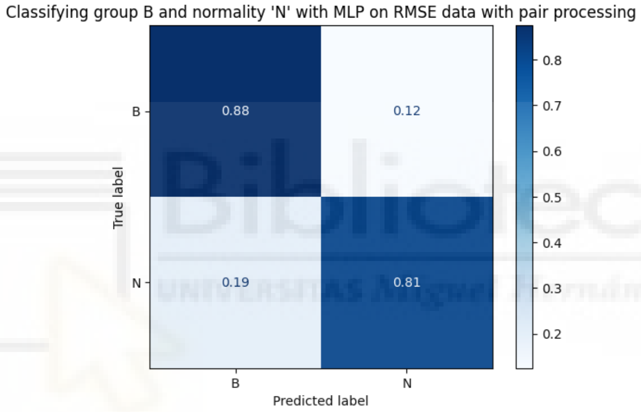

Figure 37: Optimised confusion matrix of MLP classifier for group B and individual prepossessing on RMSE

We have achieved 100% accuracy for group A. Group B is now at an F1 score of 0.84. This is the best that was achieved with the 85, 81 and 85 individuals of group A, B and N respectively.

It is likely that with more training data and further optimisation, group B prediction could be improved.

3.2.2.3 NEURAL NET OF NEURAL NETS

A first attempt was made to utilise the previous classifiers (sec. 3.2.2.2) as inputs for a final neural net, to classify all groups together like in section 3.2.1. It is unrefined and could use several improvements but it provides a basis for future work.

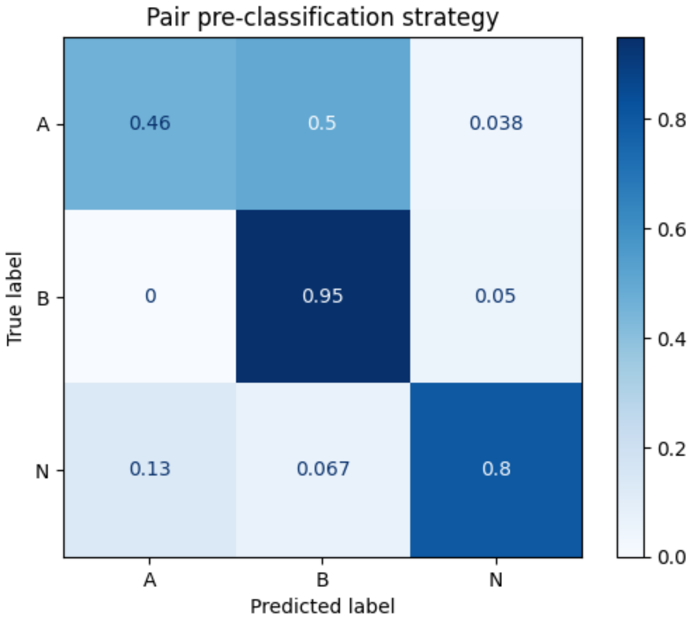

Figure 38: MLP from MLP probabilities

It takes the prediction probabilities of the previous neural nets (build to identify group A and group B) as inputs (fig. 38).

4 CONCLUSIONS

4.1 CLASSIFYING EVOKED-RESPONSES

RMSE and MAE give the same info, while RMSE has been more accurate in our use case. It is counterproductive to use them both, since they both measure the error compared to a reference. It doesn't add any meaningful data to the classifier. That's why, after the previous results, we recommend the exclusive use of RMSE.

In general SVMs approaches are faster to iterate for lower dimensions. However, neural networks are more suited for multiple predictions and are especially time-effective with higher dimensionality.

Neural networks also have the added benefit of potentially doing a better job than an SVM with *rbf* kernel, if the hidden layers are specifically selected for the task at hand.

For evoked-response classification, neural nets - operating on RMSE values for all 19 sensors - gave the best results. They have been tested on independent data and maintain a 100% accuracy score. While more testing is required, it shows that 85 individuals (70% of that for fitting) might be sufficient training data to build a very successful predictor.

4.2 CLASSIFYING PATHOLOGIES

Classifying all pathology groups at once is not scalable. We recommend sticking to the second proposed method, building classifiers for each pathology.

With this design, we can identify pathology A with very satisfying results (100% accuracy with our testing data). For pathology B, results could most likely be improved with a larger training dataset, with more individuals. And while we can't say anything about other diseases, that might not affect the brain in different ways that are detectable from EEG data; the results are promising enough to encourage future work. This line of research could provide with an additional tool to assist neurologists in faster and more precise diagnosis.

It is probably best to use neural nets that only compare: a group of individuals with a single disease, with a group of healthy individuals. Not only it gives the best results, but it makes more sense to check for different pathologies separately.

5 APPENDIX

5.1 ADDITIONAL FIGURES

5.1.1 CONFUSION MATRICES OF SVM ON AVERAGED SENSOR DATA

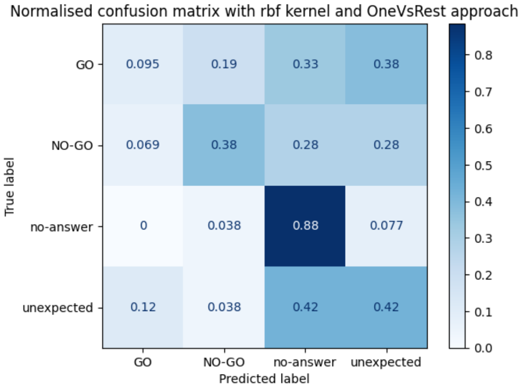

Figure 39: Confusion matrix of SVM with RMSE calculated on averaged sensor data

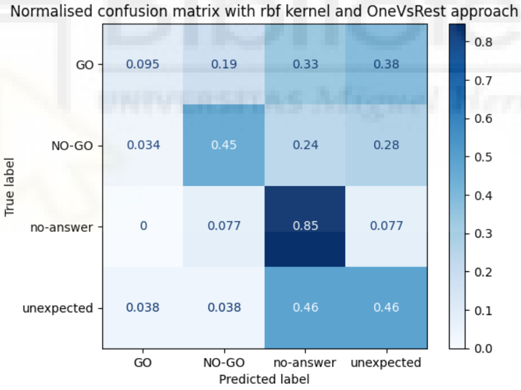

Figure 40: Confusion matrix of SVM with RMSE & MAE calculated on averaged sensor data

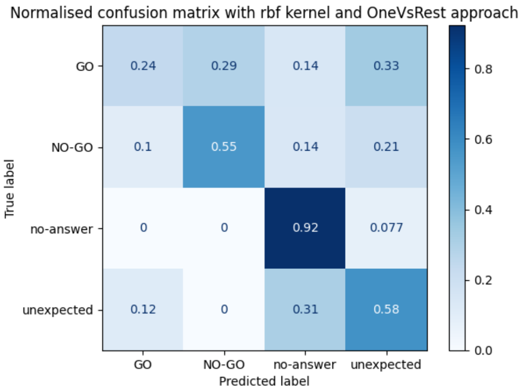

Figure 41: Confusion matrix of SVM with processed signals on averaged sensor data

5.1.2 CONFUSION MATRICES OF COMPARED EVOKED-RESPONSE CLASSIFIERS (SECTION 3.1.4)

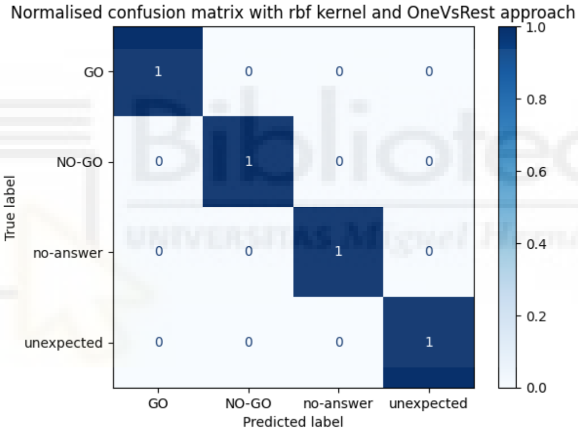

Figure 42: Confusion matrix of SVM with averaged RMSE

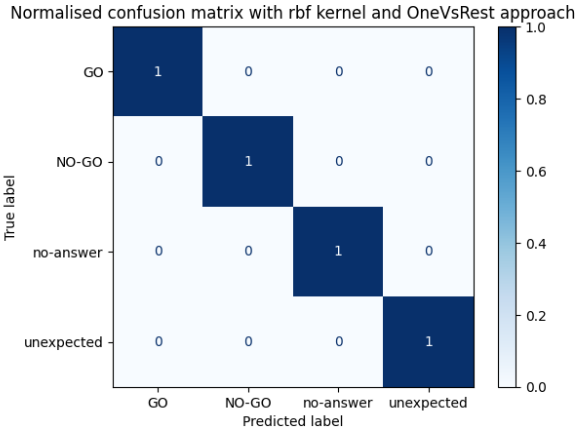

Figure 43: Confusion matrix of SVM with RMSE of all sensors

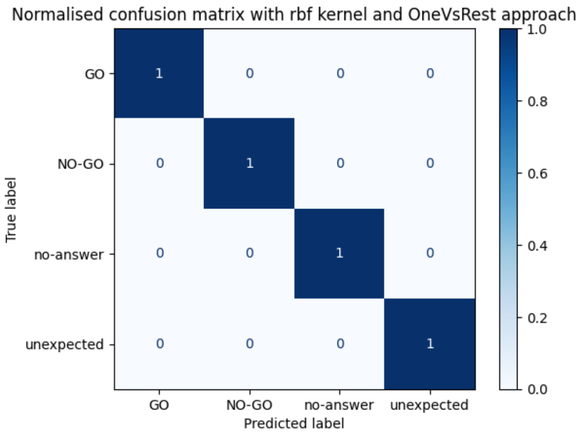

Figure 44: Confusion matrix of SVM with RMSE & MAE of all sensors

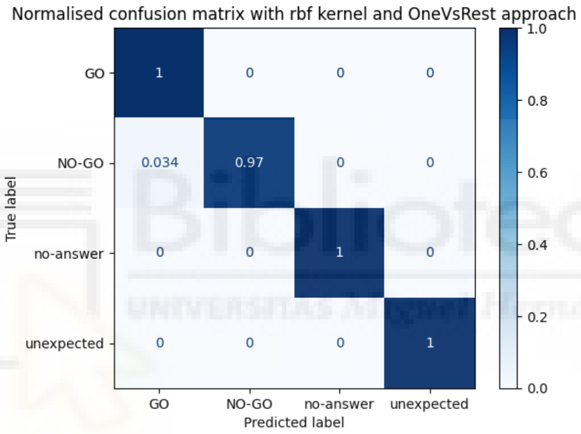

Figure 45: Confusion matrix of SVM with MAE of all sensors

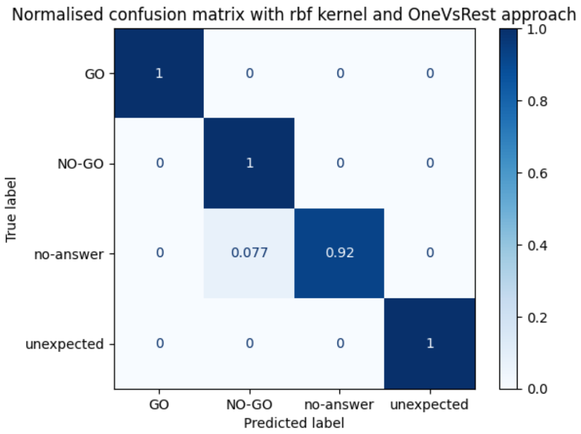

Figure 46: Confusion matrix of SVM on preprocessed signals of all sensors

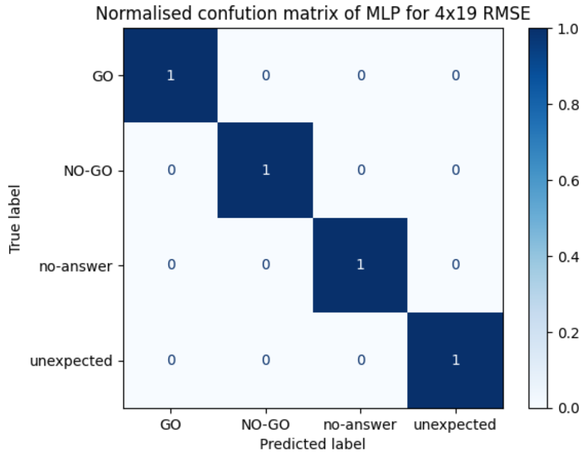

Figure 47: Confusion matrix of neural network on RMSE of all sensors

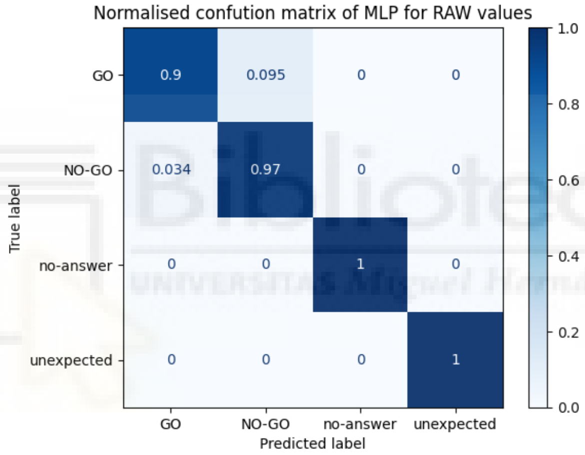

Figure 48: Confusion matrix of neural network on preprocessed signals of all sensors and a single hidden layer

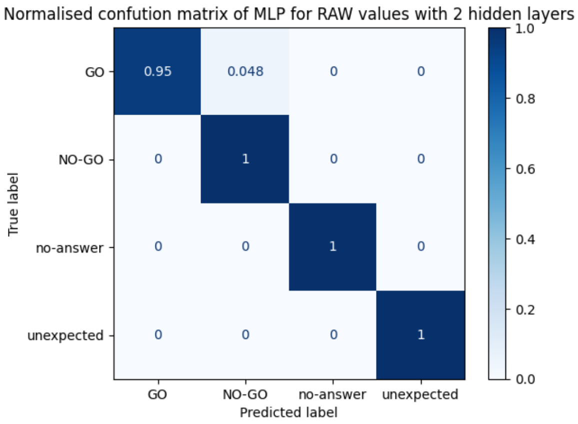

Figure 49: Confusion matrix of neural network on preprocessed signals of all sensors and two hidden layers

5.1.3 CONFUSION MATRICES OF EVOKED-RESPONSE CLASSIFIERS TESTED ON INDEPENDENT DATA (SECTION 3.1.5)

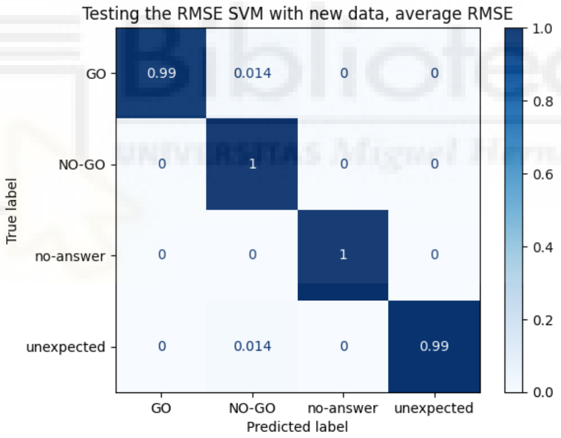

Figure 50: Confusion matrix of SVM on independent data with averaged RMSE

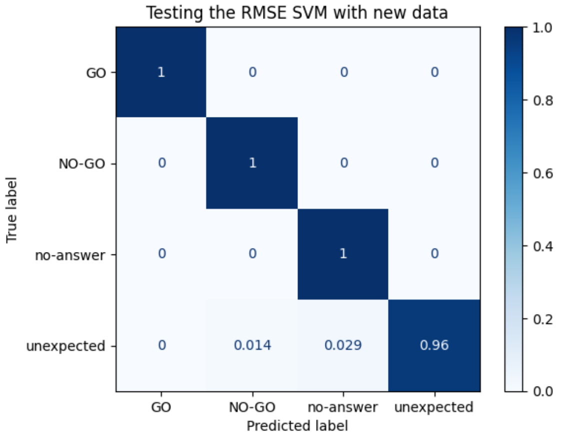

Figure 51: Confusion matrix of SVM on independent data with RMSE of all sensors

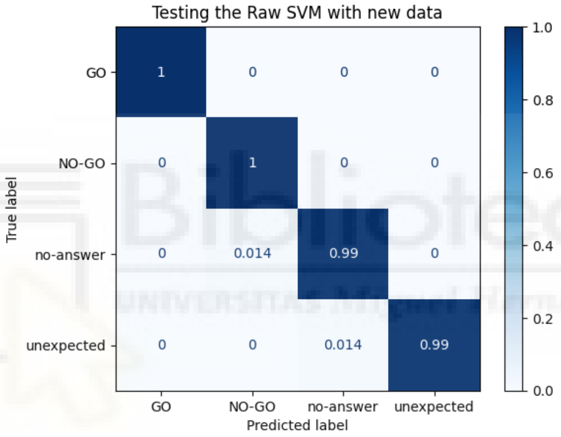

Figure 52: Confusion matrix of SVM on independent preprocessed signals of all sensors

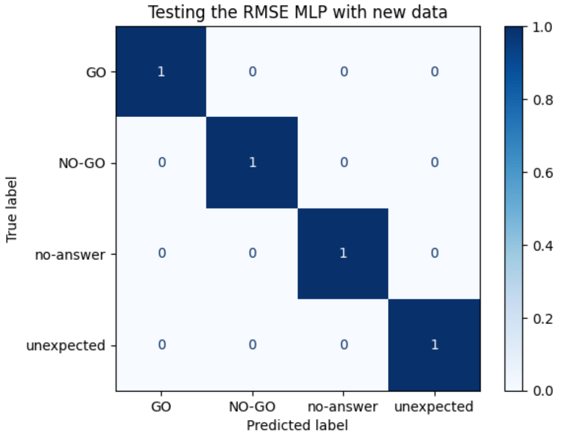

Figure 53: Confusion matrix of MLP on independent data with RMSE of all sensors

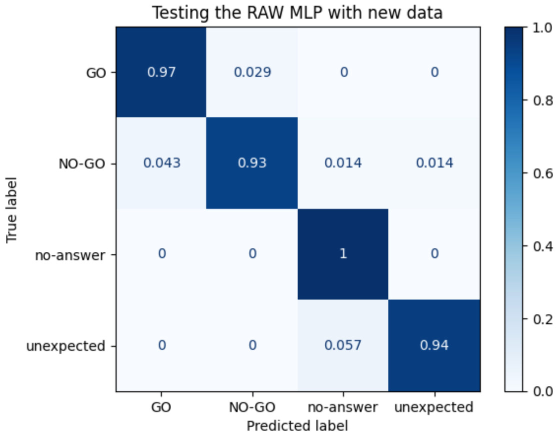

Figure 54: Confusion matrix of MLP on independent preprocessed signals of all sensors and a single hidden layer

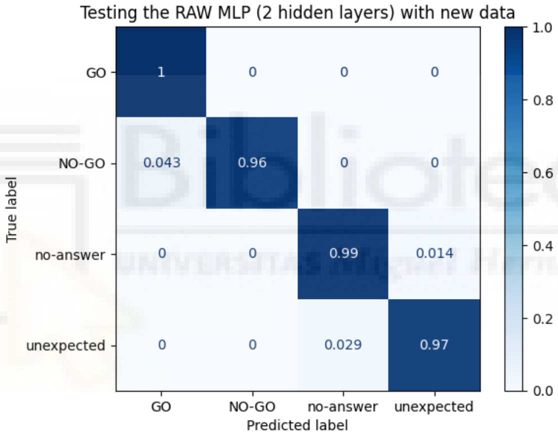

Figure 55: Confusion matrix of MLP on independent preprocessed signals of all sensors and a two hidden layers

5.1.4 CONFUSION MATRICES OF PATHOLOGY CLASSIFIERS WORKING ON ALL GROUPS

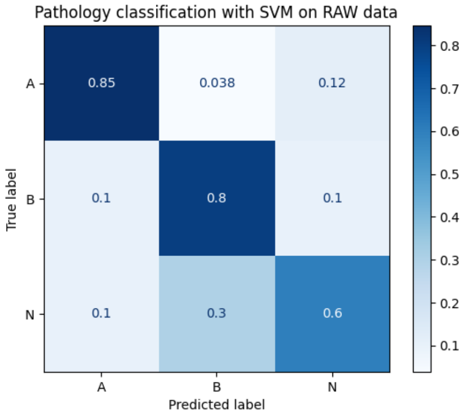

Figure 56: Confusion matrix of pathology classification with SVM on all preprocessed signals

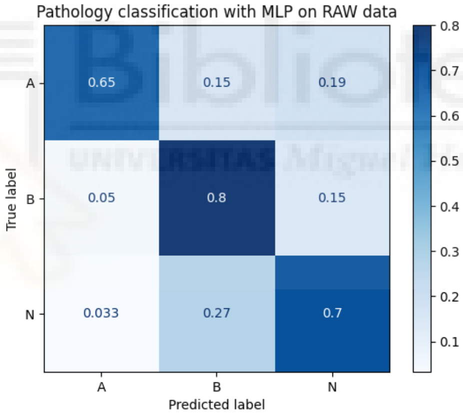

Figure 57: Confusion matrix of pathology classification with mlp on all preprocessed signals

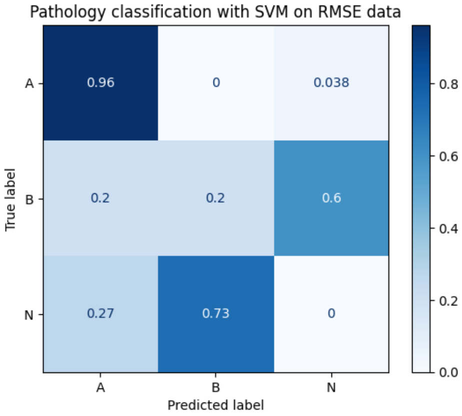

Figure 58: Confusion matrix of pathology classification with svm on all preprocessed signals

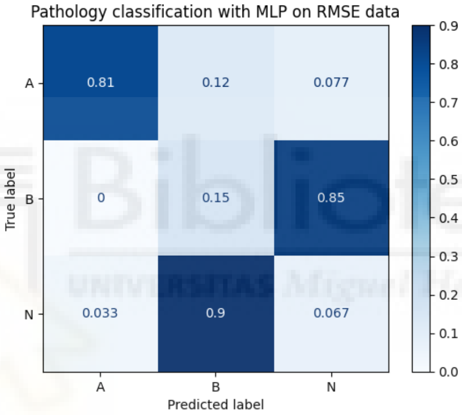

Figure 59: Confusion matrix of pathology classification with mlp on all preprocessed signals

5.1.5 CONFUSION MATRICES OF INDIVIDUAL PATHOLOGY CLASSIFIERS WITH BULK PREPROCESSING

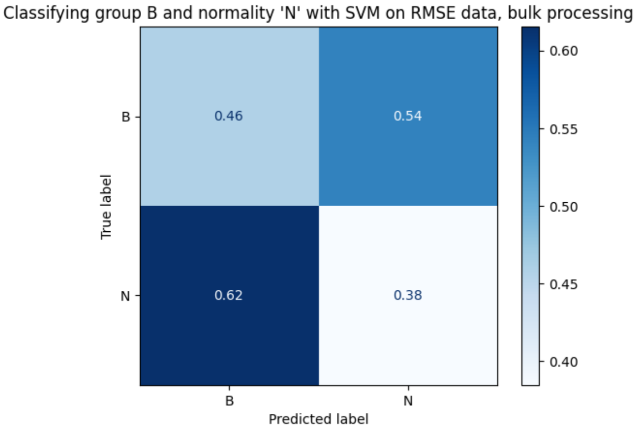

Figure 60: Confusion matrix of SVM classifier for group B and bulk preprocessing on RMSE

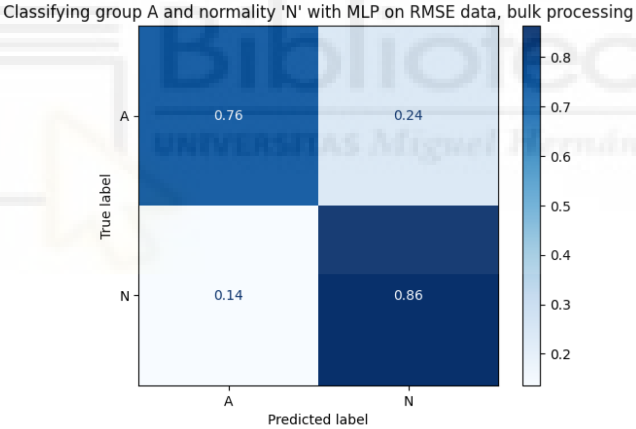

Figure 61: Confusion matrix of MLP classifier for group A and bulk preprocessing on RMSE

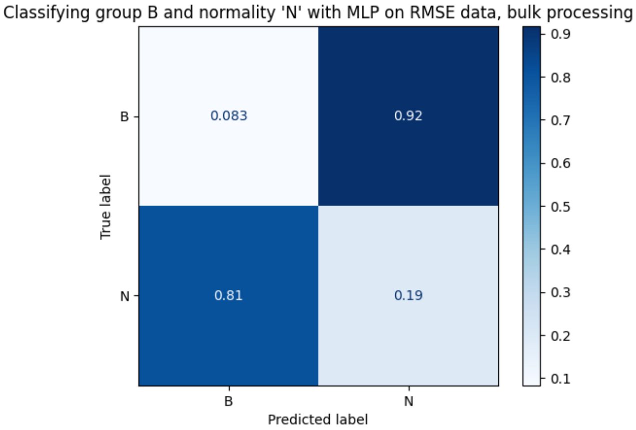

Figure 62: Confusion matrix of MLP classifier for group B and bulk preprocessing on RMSE

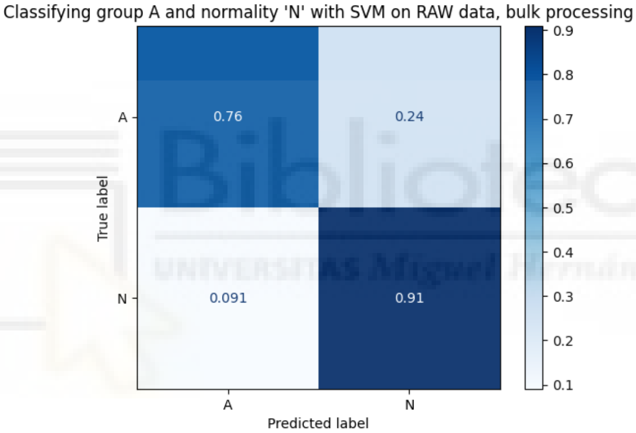

Figure 63: Confusion matrix of SVM classifier for group A and bulk preprocessing on signals

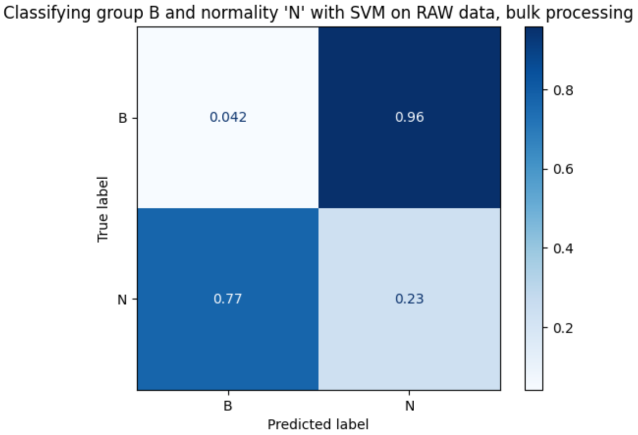

Figure 64: Confusion matrix of SVM classifier for group B and bulk preprocessing on signals

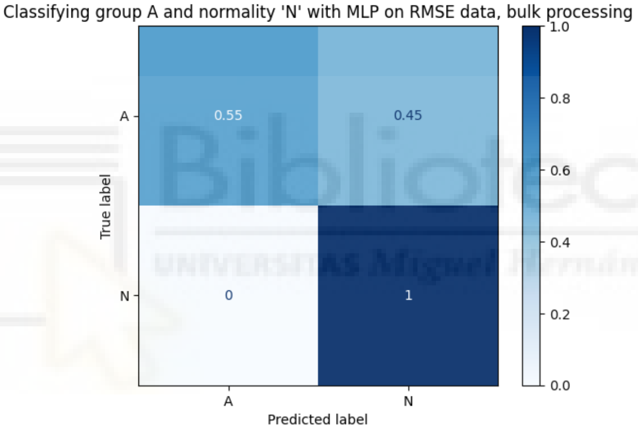

Figure 65: Confusion matrix of MLP classifier for group A and bulk preprocessing on signals

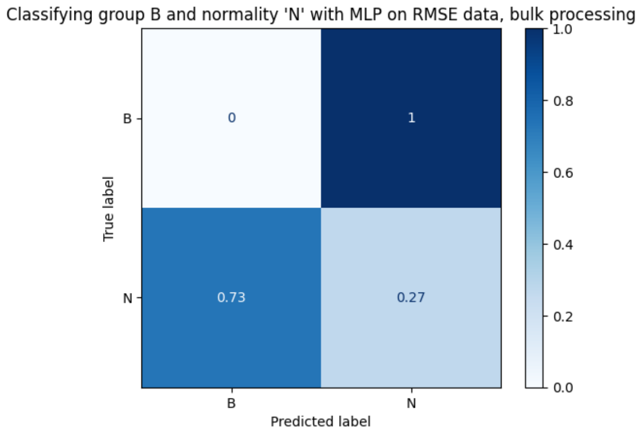

Figure 66: Confusion matrix of MLP classifier for group B and bulk preprocessing on signals

5.1.6 CONFUSION MATRICES OF INDIVIDUAL PATHOLOGY CLASSIFIERS WITH INDIVIDUAL PREPROCESSING

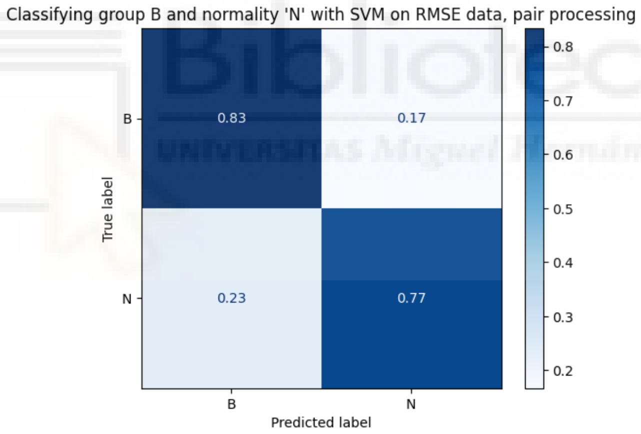

Figure 67: Confusion matrix of SVM classifier for group B and individual preprocessing on RMSE

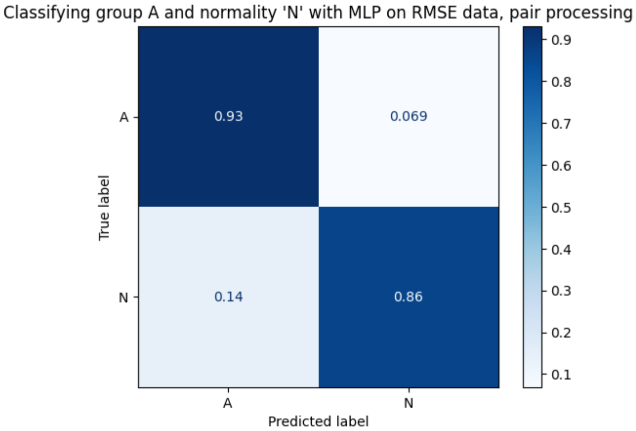

Figure 68: Confusion matrix of SVM classifier for group A and individual preprocessing on RMSE

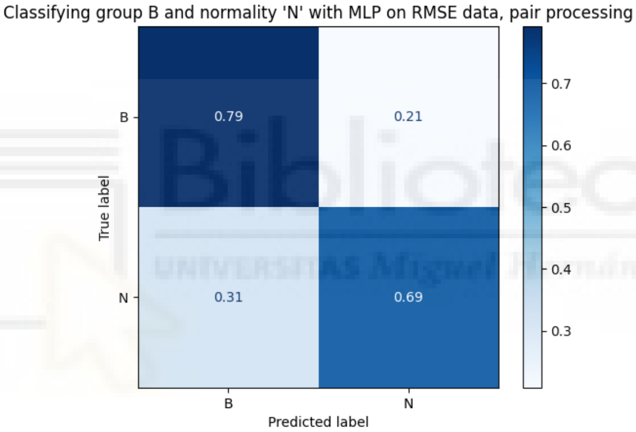

Figure 69: Confusion matrix of SVM classifier for group B and individual preprocessing on RMSE

5.2 LIST OF FIGURES

6 REFERENCES

1. Niedermeyer E, Silva FHL da. 2005. Electroencephalography: Basic principles, clinical applications, and related fields. Lippincott Williams & Wilkins.

2. Regan D. 1966. Some characteristics of average steady-state and transient responses evoked by modulated light. *Electroencephalography and Clinical Neurophysiology* 20:238–248. doi: https://doi.org/10.1016/0013-4694(66)90088-5.

3. Jiang X, Bian G-B, Tian Z. 2019. Removal of artifacts from EEG signals: A review. *Sensors* 19. doi: 10.3390/s19050987.

4. Hyvärinen A, Oja E. 2000. Independent component analysis: Algorithms and applications. *Neural Networks* 13:411–430. doi: https://doi.org/10.1016/S0893-6080(00)00026-5.

5. Cortes C, Vapnik V. 1995. Support-vector networks. *Machine Learning*. doi: 10.1007/BF00994018.

6. Hofmann T, Schölkopf B, Smola AJ. 2008. Kernel methods in machine learning. *The Annals of Statistics* 36:1171–1220. doi: 10.1214/009053607000000677.

7. Rosenblatt F. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review* 65–386.

8. Brodley CE, Friedl MA. 1996. Identifying and eliminating mislabeled training instances, p. 799–805. *In* In AAAI/IAAI. AAAI Press.

9. Ian Goodfellow AC Yosua Bengio. 2016. Back-propagation and other differentiation algorithms, p. 200–220. *In* Deep learning. MIT Press.

10. Mohamad I, Usman D. 2013. Standardization and its effects on k-means clustering algorithm. *Research Journal of Applied Sciences, Engineering and Technology* 6:3299–3303.

11. Duda RO, Hart PE, Stork DG. 2001. Pattern classification, 2nd ed. Wiley-Interscience.

12. Oliphant TE. 2007. Python for Scientific Computing. *Computing in Science and Engineering* 9:10–20. doi: 10.1109/MCSE.2007.58.

13. Papoulis A. 1962. The fourier integral and its application. New York: McGraw-Hill.

14. Cooley JW, Turkey JW. 1965. An algorithm for the machine calculation of complex fourier series. *Math Comp*. doi: https://doi.org/10.1090/S0025-5718-1965-0178586-1.

15. Willmott C, Matsuura K. 2005. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research* 30:79. doi: 10.3354/cr030079.

16. Galloway NR. 1990. Human brain electrophysiology: Evoked potentials and evoked magnetic fields in science and medicine. *British Journal of Ophthalmology* 74:255–255. doi: 10.1136/bjo.74.4.255-a.

17. Steinier Jean, Termonia Yves, Deltour Jules. 1972. Smoothing and differentiation of data by simplified least square procedure. *Analytical Chemistry* 44:1906–1909. doi: 10.1021/ac60319a045.