EEG Signal Processing and Classification of brain intentions

Sofiia Petryshyn

Signal Processing Course

CS UCU 2021

Introduction

1. What is EEG?

An electroencephalogram (EEG) is a test used to evaluate the electrical activity in the brain. Brain cells communicate with each other through electrical impulses. An EEG can be used to help detect potential problems associated with this activity.
An EEG tracks and records brain wave patterns. Small flat metal discs called electrodes are attached to the scalp with wires. The electrodes analyze the electrical impulses in the brain and send signals to a computer that records the results.
The electrical impulses in an EEG recording look like wavy lines with peaks and valleys. These lines allow doctors to quickly assess whether there are abnormal patterns. Any irregularities may be a sign of seizures or other brain disorders.
In Figure 1 we can see the basic electrode placement of 22 electrodes on the right hand, and on the left hand, we can see EEG signal recorded.

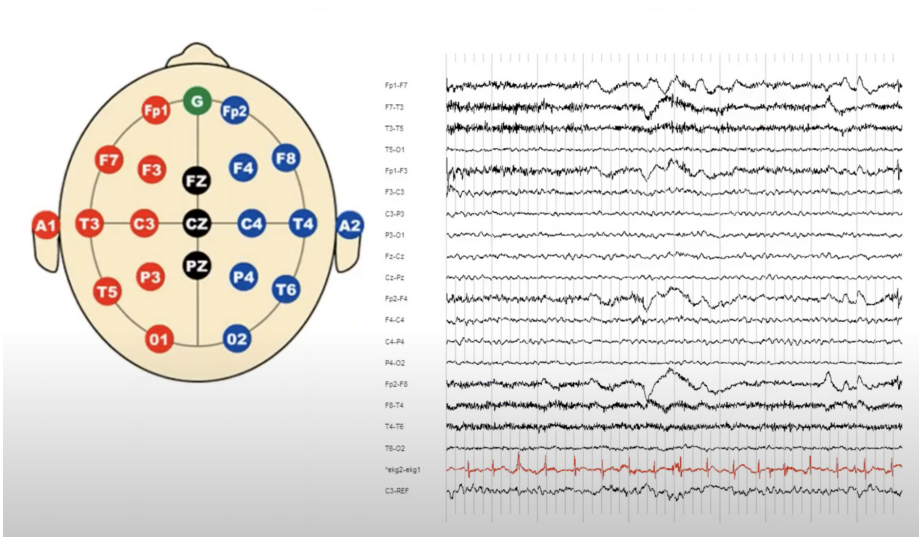


Fig. 1. Electrode placement

2. Why EEG?

An EEG signal is used to detect any activity that is happening in the human brain. Scientists can detect some brain disorders, emotions, or movements of particular parts of the body. The measurements given by an EEG are used to confirm or rule out various conditions.

3. Related works:

Effective automated method for detection and suppression of muscle artefacts from single-channel EEG signal

This paper involves three major steps: decomposition of the input EEG signal into two modes using VMD; detection of MAs based on zero crossings count thresholding in the second mode; retention of the first mode as MAs-free EEG signal only after detection of MAs in the second mode. The authors evaluate the robustness of the proposed method on a variety of EEG and EMG signals (EEG during mental arithmetic tasks database (EEGMAT)). Evaluation results using different objective performance metrics depict the superiority of the proposed method as compared to existing methods while preserving the clinical features of the reconstructed EEG signal.
Results show that our proposed method achieves a Se of 100% in detecting all the MAs-corrupted EEG signals taken from all three databases. To present the efficacy of the proposed method, we implement some of the existing denoising methods such as EEMD, EEMD-CCA, EMD-CCA, EEMD-MCCA, and wavelet denoising for suppressing MAs from the EEG signal.

Outlier detection for single-trial EEG signal analysis

The performance of a brain-computer interface (BCI) system is usually degraded due to the outliers in electroencephalography (EEG) samples. This paper presents a novel outlier detection method based on robust learning of Gaussian mixture models (GMMs). The proposed method was applied to the single-trial EEG classification task. After trial-pruning, feature extraction and classification are performed on the subset of training data, and experimental results demonstrate that the proposed method can successfully detect the outliers and therefore achieve more reliable results.

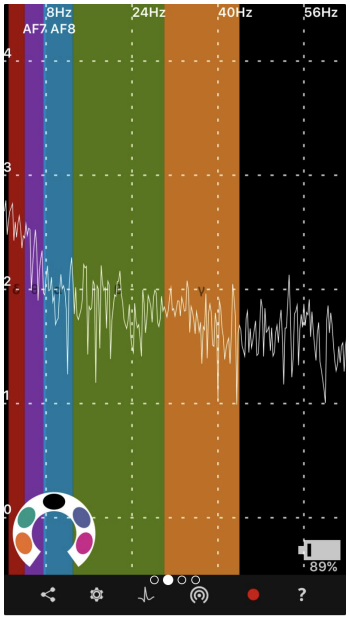
Project description

1. Dataset description

The first idea was to collect the dataset on my own using the Muse 2 device. Muse is a brain-sensing headband that uses real-time biofeedback in form of 14 EEG channel signals.
However, I had some trouble collecting the data with Muse, as the device was discharging really fast while using it. However, I decided to use Muse 2 data collected by the other developer, Tim de Boer, from Amsterdam.

Muse 2 device

To collect the data from Muse I had to use third-party software called Mind Monitor, the app cost $15 and the signal recorded can be sent to the PC by Dropbox or by email. By pressing the record button, a CSV file is made. Data were collected at 256 Hz for the raw EEG signals, and 10 Hz for the brain wave data, accelerometer, gyroscope, Headband On or Off, HSI (Horse Shoe Indicator), and the marker button presses.



Mind monitor recording screen example

The setup was as follows (the following is copied from the paper):

1. The participant is sitting down on a chair with arms extended in parallel, resting on a table.
2. Two bottles of water are on the table. One of them is approximately 5 cm to the left of the left hand and the other bottle is 5 cm to the right of the right hand.
3. The participant's head faces forward, while the eyes rotate to the left, looking to the bottle.
4. We asked the participant to imagine picking up the bottle with the left hand, but without moving the hand; only thinking about it for 6 s.
5. Then, we asked the participant to look at the bottle on the right and imagine picking up the bottle with the right hand, but without moving the hand, only thinking about it for 6 s.
6. We repeated steps 4–5 for 3 times for each participant: 1 min in total.
7. Then, we repeated steps 1–6 for 5 times.

This resulted in 3 x 5 = 15 minutes of data. But because of zero values due to bad contact with the skin, there remained just 10 minutes of data. Using the brain wave data for all individual sensors, we have 4 x 5 = 20 features, which are the signal levels of the 5 different brain waves (gamma, beta, alpha, theta, delta), for each of the 4 sensors.

Important to notice, that the Muse device already filters the signal into the specific brain wave signals (gamma, beta, alpha, theta, delta) rather than raw EEG data.

The link to the database is provided here.

An example of data collected can be seen in Figure 2 below. In the screenshot, we can see the original signal collected. Each plot represents a group of EEG signals (Alpha, Beta, Theta, Gamma, Delta) and the last plot represents labels distribution over time the signal was collected. The collected EEG signal is processed for a little bit by the program MindMonitor, here is the description of their processing. However, we can see that still data processing is still required.

Fig 2. Example of EEG recorded.

2. Data pre-processing and dataset formation

For this project the data the raw collected data looks as follows:

	TimeStamp	Delta_TP9	Delta_AF7	Delta_AF8	Delta_TP10	Theta_TP9	Theta_AF7	Theta_AF8	Theta_TP10
0	2021-08-26 18:45:49.518	0.529532	1.062243	0.672268	0.406634	0.563284	0.510825	0.157816	0.257729
1	2021-08-26 18:45:49.519	0.529532	1.062243	0.672268	0.406634	0.563284	0.510825	0.157816	0.257729
2	2021-08-26 18:45:49.520	0.529532	1.062243	0.672268	0.406634	0.563284	0.510825	0.157816	0.257729
3	2021-08-26 18:45:49.520	0.529532	1.062243	0.672268	0.406634	0.563284	0.510825	0.157816	0.257729
4	2021-08-26 18:45:49.521	0.529532	1.062243	0.672268	0.406634	0.563284	0.510825	0.157816	0.257729

Fig 3. Raw EEG collected data example

3. Steps of data preparation:

1. Set labels

There is one row that indicates the marker that we set during the signal collection. Using this column we are going to set labels for our data: columns 'left' and 'right', where we set binary values depending on the process that the person wearing the EEG helmet was thinking about.

2. Aggregate values of each EEG channel for each 100 milliseconds

For each channel, we are going to take the 'mean' statistic value for each sliding window. Whereas, for the 'label' column we take the 'mode' for each window.

3. Clean the data: Outlier detection and removal with GMM

a. Butterworth Filter Frequency Response

As we can see the data is quite noisy, we are going to use Butterworth Filter. The frequency response of the Butterworth filter is maximally flat in the passband and rolls off towards zero in the stopband.

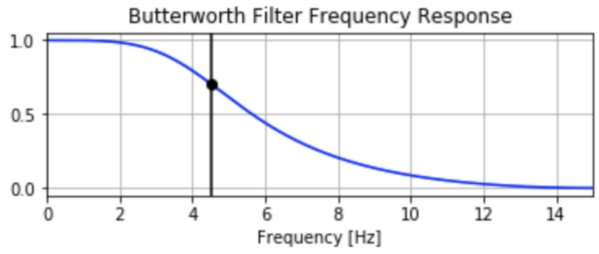


Fig 4. Butterworth Filter Frequency Response

b. Using Nyquist Frequency

The Nyquist rate or frequency is the minimum rate at which a finite bandwidth signal needs to be sampled to retain all of the information. If a time series is sampled at regular time intervals dt, then the Nyquist rate is just 1/(2 dt).

c. Gaussian Mixture Model

Building Gaussian Mixture Model we set a probability to each row value. The probability value tells us of how much the value can be described by the distribution or is it an outlier.

Fig. 5 Example of pre-processed EEG signal.

Here we can see the values to be less noisy after filtering and outlier removal steps.

4. Feature engineering

a. Dimensionality reduction with PCA

Principal component analysis (PCA). Linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space. The input data is centered but not scaled for each feature before applying the SVD. We added 4 new columns of PCA values to a feature list.

Gamma_AF8	Gamma_TP10	Elements	left	right	pca_1	pca_2	pca_3	pca_4
-0.462037	0.013447	NaN	0.0	0.0	0.782521	-0.217533	-0.006769	-0.196357
-0.504300	0.012351	NaN	0.0	0.0	0.770773	-0.222912	-0.008712	-0.235120
-0.544276	0.011485	NaN	0.0	0.0	0.760095	-0.228673	-0.011038	-0.273098
-0.579986	0.011038	NaN	0.0	0.0	0.751187	-0.235187	-0.013986	-0.309252
-0.609838	0.011161	NaN	0.0	0.0	0.744234	-0.242446	-0.017924	-0.342443

Fig 6. Example of added features after principal component analysis

b. Dimensionality reduction with ICA

Independent component analysis (ICA) is a recently developed method in which the goal is to find a linear representation of non-Gaussian data so that the components are statistically independent, or as independent as possible. Such a representation seems to capture the essential structure of the data in many applications, including feature extraction and signal separation. We added 20 more features that represent our data.

FastICA_13	FastICA_14	FastICA_15	FastICA_16	FastICA_17	FastICA_18	FastICA_19	FastICA_20
0.016544	0.012692	-0.028136	-0.006015	0.009738	-0.011257	0.010143	-0.001879
0.013355	0.008746	-0.023223	-0.003354	0.009689	-0.014978	0.007659	-0.002654
0.010281	0.004959	-0.018601	-0.000962	0.009722	-0.018247	0.005431	-0.003239
0.007398	0.001484	-0.014579	0.000900	0.009854	-0.020652	0.003669	-0.003469
0.004752	-0.001559	-0.011391	0.002022	0.010093	-0.021882	0.002527	-0.003229

Fig 7. Example of added features after independent component analysis

c. Add moving windows statistics values

Then for each EEG channel, we applied a sliding window of 1-second size. For each window, we calculated the statistics (mean, median, std, slope, min, max).

d. Apply Fourier transform

The Fourier Transform is a tool that breaks a waveform (a function or signal) into an alternate representation, characterized by the sine and cosine funcitons of

varying frequencies. For our signal, we took a sampling frequency of 100 Hz and set the following list of frequencies to build sine & cosine functions on [0., 10., 20., 30., 40., 50.]. Then for each channel we added time based channel value describing a signal through a particular frequency function.

Gamma_TP10_freq_30.0_Hz_ws_10	Gamma_TP10_freq_40.0_Hz_ws_10	Gamma_TP10_freq_50.0_Hz_ws_10
-0.024844	-0.024917	-0.024945
0.028541	0.024630	0.023281
-0.142190	-0.139012	-0.137927
-0.075756	-0.074854	-0.074542
0.063069	0.064181	0.064565

Fig 8. Example of added columns after FFT

4. Classification of brain process based on the EEG signal

The data collected from Muse 2 device had columns describing 14 EEG channel signals and the marker what the person was thinking about. Sometimes, the participant's head faces forward, while the eyes rotate to the left, looking to the bottle, and sometimes they were looking to the right. As we have labeled data with 3 possible labels we are going to train a classifier model that looks at the features that we have in the dataset and learns the labels.

In order to build a classifier, we used Random Forest Classifier. It is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and predicts the final output.

Technical requirements:

We used scikit-learn library that optimizes and automates processes related to Machine Learning programming in Python. Use cases of scikit-learn: split the dataset into train and test parts, feature ranking with recursive feature elimination, and Random Forest Classifier model.

The data of shape ((1950, 344), (1950,)) was divided into train test sets in proportion 0.67 and 0.33 respectively.

Experiments:

We run three experiments with all the features generated. First of all we trained a RFC on all the 344 features and got a model accuracy to be 100% on train set and 87% on the test set.

Then, in order to make a smaller model, we reduced the number of features used to 20. Feature selection was done in two steps:

- RFE model was fitted to our train set containing 344 features and then the features were ranked due to importance (Fig 9).

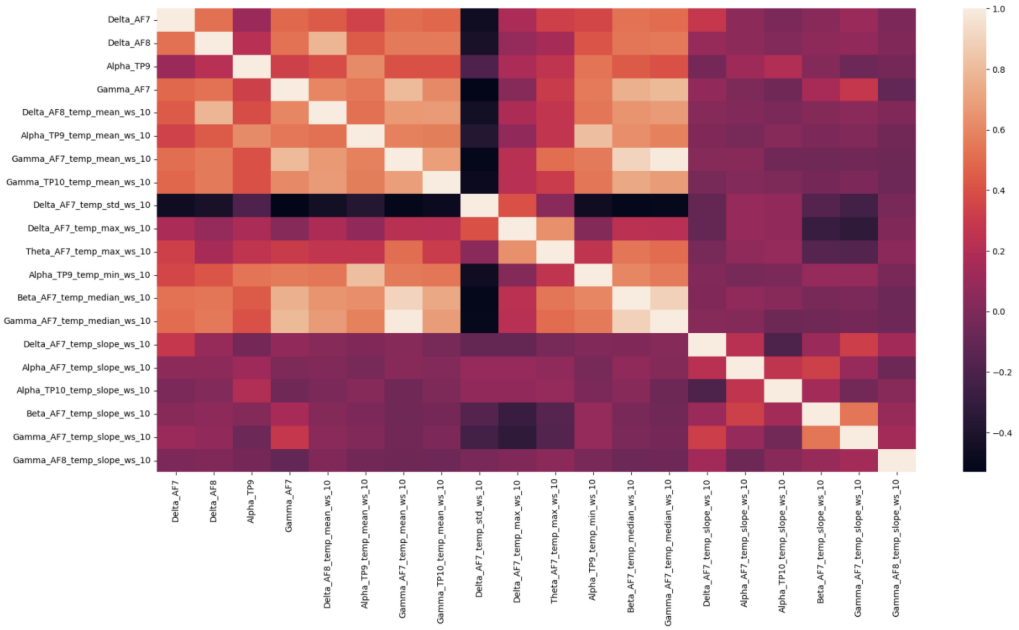


Fig 9. Correlation matrix among 20 top features selected by RFE model

- after RFC was fitted to the data we took feature importance key value to take top 20 features. (Fig 10)

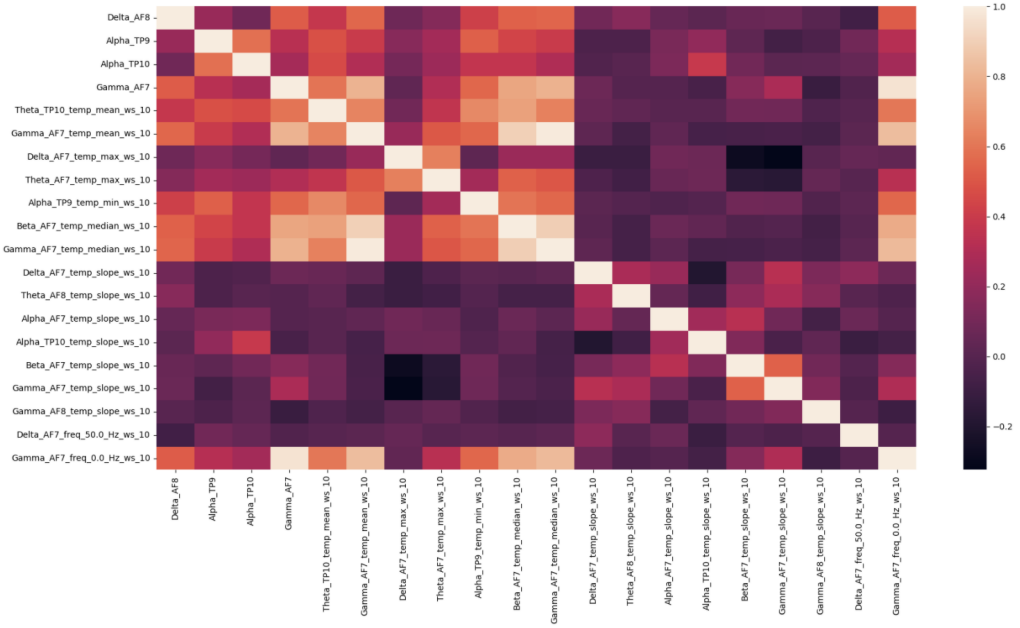


Fig 10. Correlation matrix among 20 top features selected by RFC model

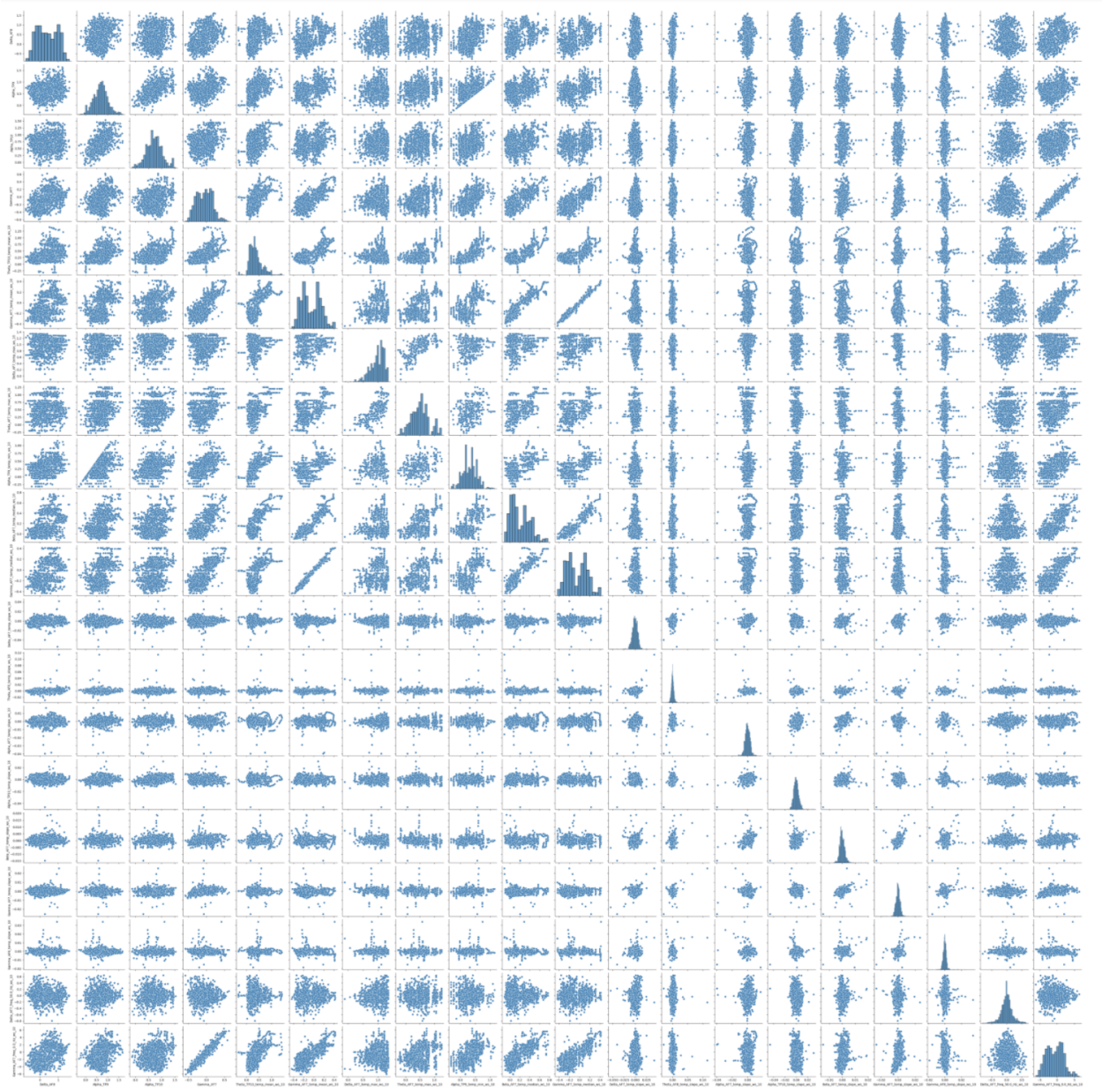


Fig 11. A pairplot of features pair distribution

As this approach of feature selection was better, what we could observe from the correlation matrix above. We still decided to prove this hypothesis with the results of the RFC model. We observed that RCF model with RCE model features selected gave 85% accuracy on the test set and RCF feature selected the classifier performed with 86% of accuracy.

Conclusion:

In this work, we did all the processes from scratch. This paper contains data collection, dataset creation, data pre-processing, data cleaning, and modeling the data. We have seen that it is possible to collect the data with EEG channels doing a set of experiments, we also cleaned the data in the way to build a Random Forest Classifier on top of the data collected.
Pre-processing of the data needed a lot of different techniques: we used the sliding window approach, statistics aggregation, PCA & ICA dimensionality reduction, and FFT to represent the signal with a wider feature range. The whole path lead us to the classifier model, that we successfully trained on 344 features, getting accuracy 87% and using 20 features to get 85% and 86% accuracy values.