Emotion classification using electroencephalography statistical features

Pamela Reyna-Gauna ✉ **and José Garcia-Higuera** ✉

Department of Electrical and Computer Engineering, Technical University of Munich

✉ pamela.reyna-gauna@tum.de
✉ arturo.garcia@tum.de

March 16, 2023

1 Introduction

Human-machine interfaces (HMI) have been created and studied for several years now for different applications, one of them being understanding human behavior. Through the use of machines, we are able to understand the subject's emotions and classifying them into different states and comprehend how are we interact with the environment and other humans. Some applications of the classification of emotions based on brain signals are detection of brain abnormalities, mental workload analysis and person identification models [1].

Other studies [2, 3, 4] have done this before, proving that it is possible to group electroencephalography (EEG) patterns into either positive, negative or neutral emotions.

Emotion classification is divided into two main parts: feature extraction, and classifier design. The present study makes use of a dataset composed of statistical features from EEG signals (already extracted features) recorded with a Muse headset (A commercial device for EEG recording via 4 EEG sensors) and seeks to implement different classifiers to detect emotions.

2 Background

2.1 Emotion classification

Emotions are defined as mental states that are induced by physiological changes [5]. These modifications can be influenced by thoughts, feelings and behavioral changes that eventually have an impact on the decisions we make. Emotions play a very important role in the evolution of consciousness and the operations of mental processes. Among several methods, EEG has gained much popularity and use.

EEG is a well-known recording method that measures the electrical activity in the skull. It is able to create specific patterns for each time step, translating what the subject is thinking, feeling or doing into electrical current. Either dry or wet electrodes are placed on the patient's head and although this device cannot acquire brain signals from deeper tissues, it is mostly used to see brain activity in the cortex. An important advantage of using EEG as a recording device is that it is economical, portable and easy to use [6]. Once the data from the brain's activity is extracted, it is furthered filtered and converted into the frequency domain to see the power spectrum. This way, it is easier to understand the phenomena and the exact moment at which this happens.

2.2 Dataset

Signals from 4 electrodes of a Muse EEG headband were recorded from 2 subjects while they watched videos reflecting 3 states: positive, negative, and neutral. Each trial, video, lasted 3 minutes. A total of 2549 statistical characteristics per trial were extracted from the recorded EEG signals. For feature extraction, the features are classified depending on the analysis window. For the analysis of a complete window (3 minutes of signals), values such as mean, maximum, minimum, covariance, eigenvalues of the covariance matrix, and FFT magnitudes, among others, were acquired. On a sliding window of 1s the mean, maximum, and minimum values are extracted [7]. Finally the dataset to be process has 2133 trials labeled as positive, negative or neutral with 2549 statistical characteristics each one. In the Figure 1 the distribution of the data based on the classes is shown. It can be seen that it is a balanced dataset.

3 Theoretical Foundations

In order to predict whether the class of the data is positive, negative or neutral based on its specific features, we used various classifiers and evaluated

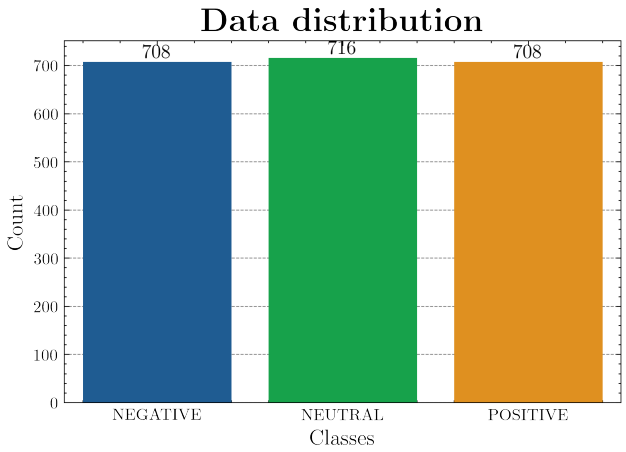

Figure 1 Distribution of trial per class in the EEG statistical features dataset.

each of their accuracy.

The Multi-Layer Perceptron model is a classifier that is used for classification and regression tasks. In general terms, the model has an input layer, one or more hidden layers and an output layer [8]. Nodes (neurons) are found in the hidden layers and with their weighted sum in the inputs, pass the result through an activation function. The weight of the nodes is determined during training with backpropagation. The output of each node represents the probability of the input data belonging to certain class. At last, the class with the highest probability is chosen as the predicted one.
Gaussian Naive Bayes (GNB) is a classifier that uses Baye's theorem. This model assumes two things: first, that the features in the input data are independent of each other and second, that their values have a Gaussian distribution [9]. Also using probability, GNB calculates the conditional of each class given the features and selects the class with the highest probability as the prediction.

A binary classifier used to find the best separating hyperplane over two classes is the Support Vector Machine (SVM). In our case, we used a kernel function to separate the data into three dimensions, since we have three classes of emotions [10].

The logistic regression classifier is yet another binary classifier. The difference from the latter, is that this model evaluates the relationship between the input features and the probability of the positive class [11]. In other words, it constrains the result to the range of [0,1] and it represents the probability of belonging to the positive class.

A different type of classifier is called Decision Trees, since it divides the features into several subsets that increase in purity with respect to the target label. The general idea is that its internal nodes represent a decision based on a single feature and a threshold, whereas each leaf node represents a class label [12]. This kind of classifier handles categorical and continuous features and it is usually preferred over other classifiers because of its visual simplicity.

Lastly, we also classified our data using the Random Forests model. This classifier is an ensemble of decision trees that, to prevent overfitting, are trained on a randomly selected subset of the data. Additionally, they use a random selection of features to increase diversity and make the classifier more robust [13]. Random forests are more accurate than any single decision tree by itself.

4 Methods

4.1 Pre-Processing

The first step was to verify if all features were complete or if there was NaN data. For this, the functions: $isnull()$ and $isna()$ from pandas() were used.
Next, we checked the data type of the dataset, especially we were looking for possible categorical variables within the data. The only categorical variable found was the label i.e. emotion. For this case the feature 'label' is coded, replacing the values as follows: replace('NEUTRAL': 0, 'POSITIVE': 1, 'NEGATIVE': 2)
To check whether it was necessary to normalize the data, the ranges of several variables were analyzed. In Figure 2 it is seen how for 5 variables the ranges vary drastically. In the case of the frequency values, the ranges varied in the thousands, while the standard deviation values were values that did not exceed the hundreds. The $StandardScaler$ function was used to perform a standardization, assuming that the features have a Gaussian distribution. A normalization was also used, with the $MinMaxScaler$ function, which normalizes taking into account the maximum and minimum of the data. However, the best results were obtained with the standardization, which is why the following results have this type of transformation.

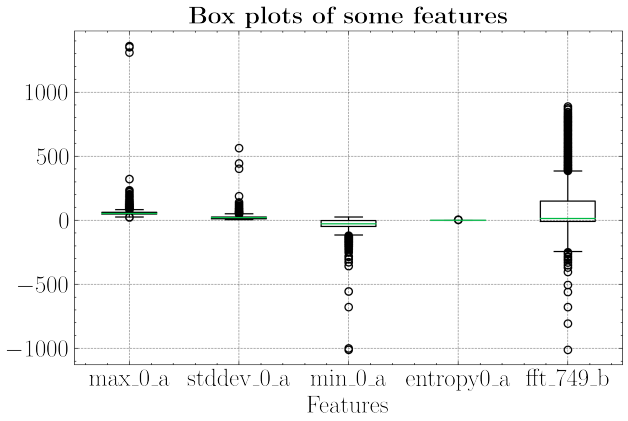

Figure 2 Box plot of 5 features from the dataset to exemplify the variability in the ranges of the features.

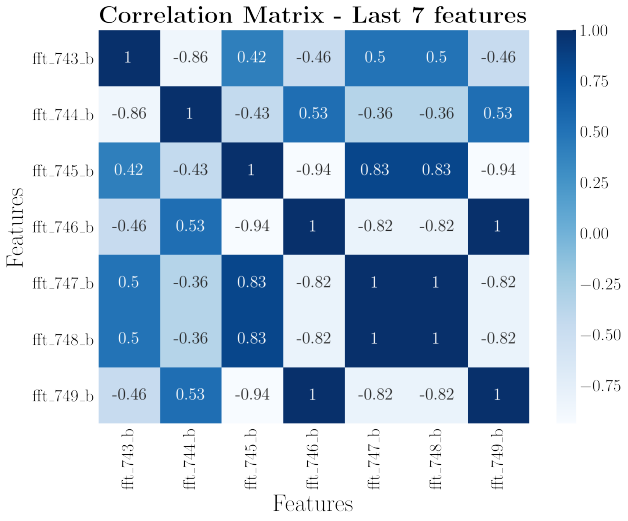

Figure 3 Correlation Matrix of the last 7 features of the dataset. Used to exemplify the high correlation between some features.

Finally, in order to reduce the dimensionality of the classification problem, we sought to reduce the number of features to optimize the implementation of the classifiers. Since much of the data computed was performed on sliding windows that overlapped by 0.5s we assumed that there were possibly highly correlated features. A correlation matrix was calculated in order to see what correlation (Pearson) coefficient there was between the features. This can be seen with the last 7 features of the dataset in Figure **??** where some features have even 100% of correlation (outside the diagonal which must always be 100%.). We took the features that have a correlation below 95%.The resulting filtered dataset has 1292 features, almost 50% of the features were eliminated.

4.2 Training a Model

We divided the data into three subsets, with 80% of the data used for training the model. To prevent overfitting and evaluate the model's performance, we further split the remaining 20% of the data into two equal parts - 10% for validation and 10% for testing. This allowed us to fine-tune the model's hyperparameters using the validation set and then obtain an unbiased estimate of the model's performance on unseen data using the testing set, for the case of the MLP model.
For the GNB, SVM, Logistic Regression, and Random Forest we used *sklearn* library and follow in most of the cases the default configuration. For the case of the MLP model we desinged form scratch using $PyTorch$.

For the GNB model does not have any specific configuration. For the case of SVM model we used a linear kernel. For the Logistic Regression we used a maximum number of iterations taken for the solvers to converge of 1000. In the case of the Random Forest we used 100 as the number of trees.

The case of the MLP required more design. We created a model of a 2-layer Multi-Layer Perceptron (MLP) neural network for classification tasks with feature size as inputs (2549 for the original dataset and 1292 for the case of the filtered dataset)and three outputs. The inputs go through the first linear layer and then through a ReLU activation function. This is commonly used in neural networks since its thresholding operations makes it computationally efficient and it also helps diminish the vanishing gradient problem. After they have gone through the ReLU, they go through the second linear layer and produce the output. To train the neural network we used an Adaptive Moment Estimation (ADAM) as an optimizer, which adapts to the learning rate for each parameter during training. We also used the Cross-entropy loss function to calculate the difference between the true values and the predicted ones. The function penalizes the model when it predicts a low probability for the true class and vice versa, a high probability for the incorrect class. It also rewards the model when it's correct, meaning a high probability for a true class and a low probability for the incorrect class. The advantage this function has over other loss functions, is its sensitivity and fast convergence during training. The training loop iterates over a given number of epochs (20), shuffles the training data and eventually iterates over batches of samples (64). After each batch is done, the optimizer is zeroed out, input and output tensors are retrieved and the loss is calculated. We used the backward function to compute the gradients of the loss with respect to

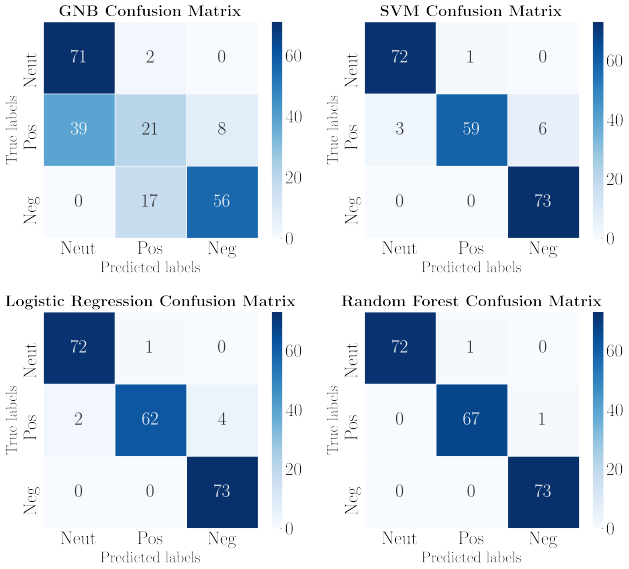

Figure 4 Confusion Matrix for the GNB, SVM, Logistic Regression and Random Forest methods. Dataset filtered and standardized.

the model's parameters. Additionally, we determined the accuracy on the validation set by comparing the predicted classes to the ground-truth labels and calculating the mean over all epochs.

5 Results and discussion

Confusion matrices were generated in order to determine which classifiers had the most errors. In Figure 4 it can be seen how the GNB model is the one with more errors, primarily in the Positive and Negative classes. For the case of Logistic Regression and Random Forest the errors are minimal and for the case of the SVM the Positive class is the one with most errors.

The MLP was used to generate confusion matrices for the filtered and original datasets. We wanted to see if there was an impact on the accuracy of the model with a 50% reduction of the features. As it can be seen in the Figure 5, the confusion matrix are the same with the exception of an additional error in the case of the filtered dataset.

In the Table 1 there is a complete overview of all possibilities tested with the classifiers. Classifiers were tested with and without standardization, and with and without feature filtering. As a final result the Random Forest and MLP models gave the best results. It can also be seen how the standardization transformation considerably improves most of the models. Also, very close accuracies are achieved between the filtered and unfiltered dataset cases, which concludes that the filtering of the featuers could lead us to similar results,

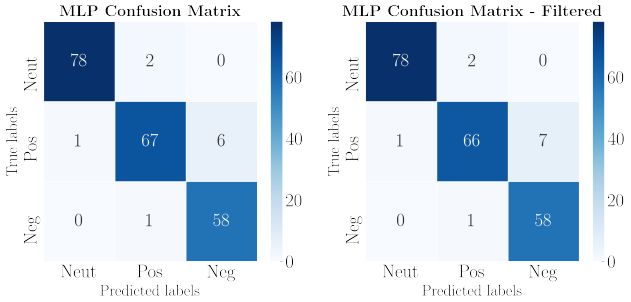

Figure 5 Confusion matrix for the MLP classifier. The right on is using the all the features for training and the left one is using a filtered set of features. Dataset standardized.

thus saving a lot of computational power. The GNB classifier was the worst performer in all cases. The GNB and Logistic Regression models are the most sensitive to transformation. The SVM model failed to be trained with the unscaled data, took a long time, and finally did not finish.

6 Conclusion

Five different classifiers were implemented in order to classify emotions from a dataset of statistical metrics. Results of 99.07% and 96.73% were achieved for the random forest and MLP cases. It was proved that the standardization transformation is fundamental for the improvement of the algorithm performance and additionally the filter of characteristics by means of the correlation matrix allows us to reach similar models optimizing computation.

The design of an MLP network with two fully connected layers and an activation function was sufficient to achieve high performance. The validation and test results prove that the model does not present overfitting.

EEG signals present many interferences or artifacts such as flickering, and heart rate, among others. The extraction of these statistical characteristics considerably facilitates the use of these signals. However, it should be emphasized that these are features that must be tested in online processing or for more precise applications to really see their value.

References

[1] B. Kaur, D. Singh, and P. P. Roy, "Eeg based emotion classification mechanism in bci," vol. 132, 2018.

Classifier	No Filtered (Stand)[%]	Filtered (Stand)[%]	No Filtered (NoStand)[%]	Filtered (NoStand)[%]
GNB	64.95	69.16	40.19	34.58
SVM	95.79	95.33	Hardware	Hardware
LogisticR	96.73	96.33	34.11	34.11
RandomForest	99.07	99.07	98.16	99.07
MLP	96.73	96.26	88.32	89.25

Table 1 Percentage of accuracy over validation dataset for all classifiers. No Filtered refers to the original dataset. Filtered refers to the dataset after pruning the features. (Stad) refers to the dataset after being standardized. (NoStand) refers to the dataset without transformation. Hardware means that because of hardware limitations the algorithm could not finish to be processed.

[2] J. Ashford, J. J. Bird, F. Campelo, and D. R. Faria, "Classification of eeg signals based on image representation of statistical features," in *Advances in Computational Intelligence Systems: Contributions Presented at the 19th UK Workshop on Computational Intelligence, September 4-6, 2019, Portsmouth, UK 19*, pp. 449–460, Springer, 2020.

[3] M. M. Javaid, M. A. Yousaf, Q. Z. Sheikh, M. M. Awais, S. Saleem, and M. Khalid, "Real-time eeg-based human emotion recognition," in *Neural Information Processing: 22nd International Conference, ICONIP 2015, November 9-12, 2015, Proceedings, Part IV 22*, pp. 182–190, Springer, 2015.

[4] R. W. Picard, E. Vyzas, and J. Healey, "Toward machine emotional intelligence: Analysis of affective physiological state," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 10, pp. 1175–1191, 2001.

[5] D. Purves, G. J. Augustine, D. Fitzpatrick, L. C. Katz, A.-S. LaMantia, J. O. McNamara, and S. M. Williams, "Physiological changes associated with emotion," 2001.

[6] M. Teplan *et al.*, "Fundamentals of eeg measurement," *Measurement science review*, vol. 2, no. 2, pp. 1–11, 2002.

[7] J. J. Bird, D. R. Faria, L. J. Manso, A. Ekárt, and C. D. Buckingham, "A deep evolutionary approach to bioinspired classifier optimisation for brain-machine interaction," *Complexity*, vol. 2019, 2019.

[8] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit, *et al.*, "Mlp-mixer: An all-mlp architecture for vision," *Advances in neural information processing systems*, vol. 34, pp. 24261–24272, 2021.

[9] Z.-j. Bi, Y.-q. Han, C.-q. Huang, and M. Wang, "Gaussian naive bayesian data classification model based on clustering algorithm," in *2019 International Conference on Modeling, Analysis, Simulation Technologies and Applications (MASTA 2019)*, pp. 396–400, Atlantis Press, 2019.

[10] W. S. Noble, "What is a support vector machine?," *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.

[11] A. I. Schein and L. H. Ungar, "Active learning for logistic regression: an evaluation," 2007.

[12] C. Kingsford and S. L. Salzberg, "What are decision trees?," *Nature biotechnology*, vol. 26, no. 9, pp. 1011–1013, 2008.

[13] G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, pp. 197–227, 2016.