EEG Brain Signal Dataset for Emotion Prediction

Deep Learning Project Report

CS 5413-WA: Manik Dhingra(1116823)

CS 5413-WB: Prajwal Dhatwalia(1112718)

Winter 2020, Lakehead University

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

Table of Contents

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

Introduction

Electroencephalography (EGG) is an electrophysiological monitoring method to record the electrical activity of the brain. Clinically, EEG refers to the recording of the brain's spontaneous electrical activity over a period of time, as recorded from multiple electrodes placed on the scalp[1].

15 Chinese film clips (positive, neutral and negative emotions) were chosen from the pool of materials as stimuli used in the experiments.

The selection criteria for film clips are as follows:

a. the length of the whole experiment should not be too long in case it will make subjects fatigue
b. the videos should be understood without explanation
c. the videos should elicit a single desired target emotion.

The duration of each film clip is about 4 minutes. Each film clip is well-edited to create coherent emotion eliciting and maximize emotional meanings[2].

There is a total of 15 trials for each experiment. There is a 15s hint before each clip and 10s feedback after each clip. The order of presentation is arranged so that two film clips targeting the same emotion are not shown consecutively. For the feedback, participants are told to report their emotional reactions to each film clip by completing the questionnaire immediately after watching each clip[1].

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

Dataset Specification

The dataset used in this project contains a total of 310 attributes, each attribute is the reading of a particular sensor to detect human response to some stimuli. Based on these attributes, a particular instance in the dataset can be categorized into three emotions - positive, negative and neutral. There exist a total of 84,420 training samples and 58,128 testing samples.

Feature Extraction

The experimental analysis in this project summarizes the performance of the model over the original set of attributes, as well as extracted features. Moreover, a grid-search was also applied to examine if feature-scaling and/or normalization would improve the performance. For feature extraction, different methods were used, namely principal component analysis (PCA), linear discriminant analysis (LDA), and quadratic discriminant analysis (QDA). Among the three mentioned, features extracted by PCA yielded the best overall performance, irrespective of the model used for further classification. Therefore, features have been extracted using PCA. Different versions of the dataset have been created with different numbers of features extracted using PCA, as well as by scaling the values using a StandardScaler from scikit-learn library. Therefore, overall, we have ten versions of the dataset - one original, and one scaled using original, and four with different PCA extracted features, and another four by scaling these PCA features.

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

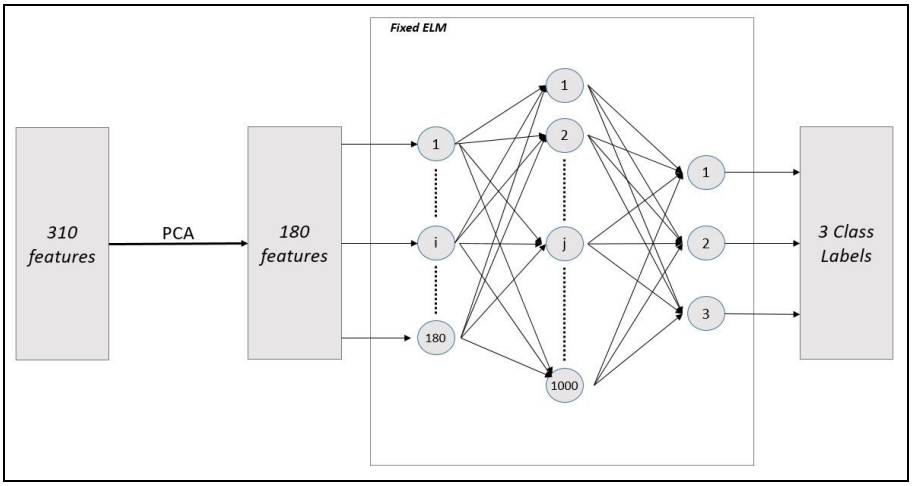
Feature Scaling

After feature extraction, with the help of a grid-search over performances of a given model over-scaled and unscaled features, it was indicated that feature-scaling can be useful, depending on the classification model used. The table below shows the comparison of performances (for the ELM model) for unscaled features, as well as scaled features using StdScaler and MinMaxScaler from sklearn. A similar comparison has been done for each other model, and the best possible combination of scaling-methods and extraction-methods has been used for classification.

Classification Models

Extreme Learning Machine (ELM):

The fixed ELM model used in this project contains 3 layers, an input layer, a hidden layer, and an output layer. The number of input nodes (features) for this model is 180 which are obtained by applying PCA on the original 310 features. The hidden layer of ELM contains 1000 hidden neurons, which finally transform to the output, i.e. 3 classes. The architecture of the ELM-model is shown below.



mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

Random Forests Classifier (RFC):

The Random Forest model used here is called from the scikit-learn library in python. There are two hyperparameters for random-forests - the number of trees in the forest, and the number of features selected for each tree in the forest. The number of trees in the RF model used in this project is 200, whereas all of the 180 features selected after PCA are used by each tree. Moreover, these 180 selected features are further scaled using the StdScaler method in sklearn, which are then used for classification. As described before, all ten versions of the data-set are fed to the same RandomForest classifier (with 200 decision trees each) and majority voting is performed.

Support Vector Machine (SVM):

The support vector machine model used is also from the sklearn library in python and the only hyperparameter tuning done for this model is on the value of C (penalty factor), which gives the best results at C=0.2. No feature scaling is done for the SVM classifier used. The overall method of SVM takes as input the set of PCA-selected features (180, 220, 225 and 250 features) and transforms them into the output class-labels, i.e. positive, negative or neutral.

Neural Network:

A simple neural network is used, for which the class-labels are encoded in one-hot notation. This neural network contains an input layer of *N* nodes (PCA selected features), four hidden dense layers of 512, 256, 128 and 64 nodes respectively, and finally the output layer with 3 nodes for classification. Between each layer, 10% of all nodes have been dropped for the training process.

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

Results

The experiment used four different classification models for comparison. These include - Fixed ELM, Neural Network, SVM, and Random Forests. Analysis of these models, given the feature extraction and feature-scaling methods for each, yield average testing accuracies as mentioned below (the best of average results are highlighted in **bold**).

Model(s)	**Average Training Accuracy**	**Average Testing Accuracy**
Random Forest with 10 estimators	93.32%	62.31%
Random Forest with 100 estimators	100%	68.82%
Voting 10 Random Forests with 200 estimators each	100%	81.21%
Voting 10 Random Forests with 10 estimators each	100%	78.30%
1 SVC with Linear Kernel	100%	69.38%
1 SVC with RBF Kernel	100%	81.55%
1 Neural Network	100%	81.70%
Voting 5 Neural Networks	100%	78.84%
Voting 10 Random Forests and 1 Neural Network	100%	80.81%
Voting 4 SVCs with RBF Kernel	100%	81.06%
Voting 4 SVCs and 10 Random Forests	100%	82.34%
Voting 4 SVCs, 10 Random Forest and 1 Neural Network	100%	82.59%
Voting 4 SVCs, 10 Random Forests and 5 Neural Network	100%	82.59%
Voting 4 SVCs and 5 Neural Network	***100%***	***82.61%***

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

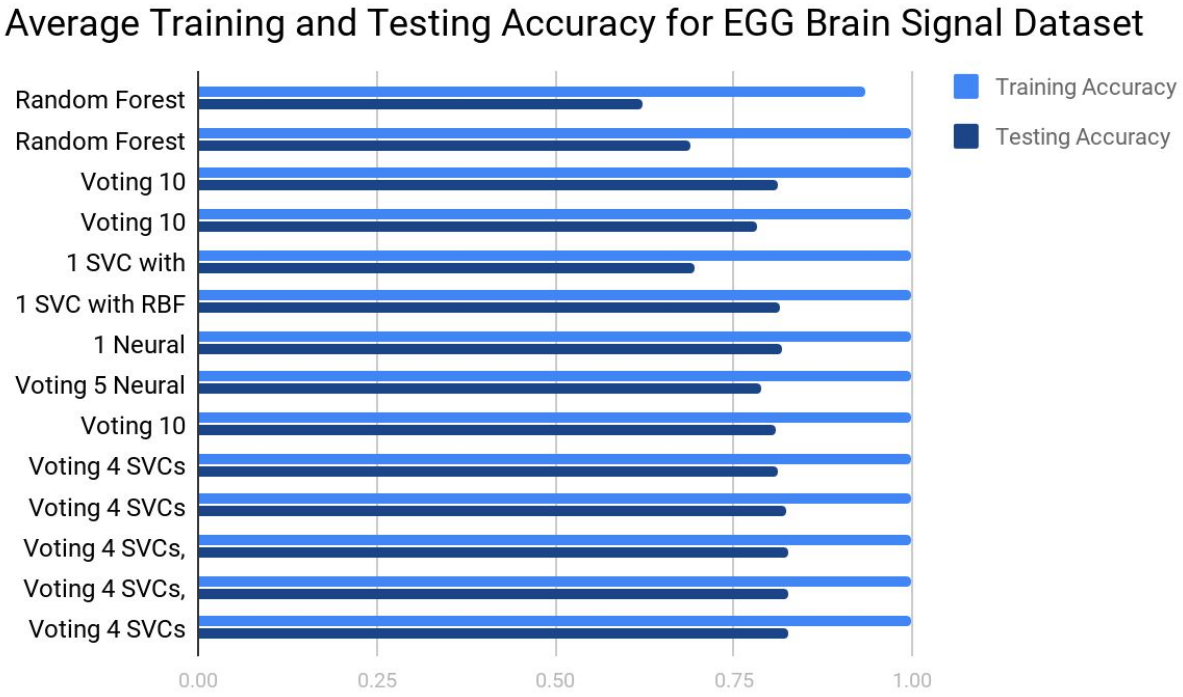


Figure: Testing accuracy and testing time comparison for different models

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

Conclusion

The best testing accuracy was achieved by doing max-voting with 4 Support Vector Classifiers (SVCs) and 5 neural network models trained independently, with feature extraction using Principal Component Analysis (PCA) and with feature-scaling, i.e. an average testing accuracy of 82.61%, and average testing time of 4185.71 seconds (1 hour 9 minutes and 45 seconds) for complete iteration over one loop.

mdhingra@lakeheadu.ca | pdhatwal@lakeheadu.ca

References

1. https://en.wikipedia.org/wiki/Electroencephalography
2. https://github.com/dhatwalia/EEG_Dataset