**EECS199 Fall quarter Report**

Development of Biometric Verification Algorithm using Elctronencephalogram(EEG)

By 94825298

Yinfei Wang

Dec 13 2020

**List**

1. Introduction
2. Work(trials) and Approach
3. EEG ML Code Understanding
4. EEG new features extraction
5. Summary and Future Plan
6. **Introduction**

This report introduced the summary of the work completed by I, Yinfei Wang, as one of the members in Professor Cao H.’s lab under Manoj Vishwanath in the Development of Biometric Verification Algorithm using Elctronencephalogram(EEG) project, based on the previous work of Spencer Kam. My goal for this project is that given the six channels of 20 people’s EEG data for a whole night, provide an algorithm that can train the data and identify the belongings of one small section of EEG data, chosen from the whole EEG data set. In the final stage of my work, the input will be a 5 seconds EEG data set, the output will be the number labeled on the experientors and the correctness will be the evaluation of the algorithm.

This report will include the different stage of parameters that I picked every time I calculated the data for better correctness, the understanding of the ML code that I implemented for my algorithm, the effort of feature extraction for better correctness, my work summary for this quarter and future plan for my incomplete work.

1. **Work and Approach**

I tried to understand the paper and the original feature.py file(without correct input and algorithm) and started to calculate the data using create\_data.ipynb file. First, I continued the setting for calculation for the last calculation of the data of Spencer. I used 1000 EEG signals for calculating each training data, used 500 overlap of EEG signals for each training data, started from 0 EEG signal, without using the correct channels(It was corrected after several stage of trials) and 6 features including rms, ApprEnt, LZComp, mpf and sef for each training data. It was improved compared to the last trail implemented by Spencer by increasing the number of training from 10 to 1000 because ML algorithm need more than a thousand data to approach a good correctness.

Then, I started working on train.ipynb file to implement ML algorithms. The algorithm is based on Scikit-learn tool[[1]](#footnote-1) including 10 algorithms(KNN, Pipeline, SVC, DecisionTree, RandomForest, Neural Network, Adaboost, GaussianNB, QDA and VotingCLF).

After implementing all the ML tools, I started training training data that I get from the previous step. Because the time to calculate one training data is about 15 seconds and more than 4 hours for calculating 1000 data for each people’s EEG signals, I started my training based on 5 people’s EEG signals and one channel, and I got 92% correctness. Then, I increase to 20 people’s EEG signals and I got 82% correctness. I thought it is a good starting correctness at the time.

Then, I got into stage two of trials. We found that we should start from 20 mintues of the original EEG because the EEG data has a lot of noise data at the beginning and we decided to not have overlap this time because it was kind of “cheat” to increase correctness. Besides, we decided to decrease the 1000 ML data for each person to 720 ML data since we do not think it will change a lot. This time, after we finished training all the data, we found that the score decrease to 77% correctness. At first, I think it is because we decrease the number of ML data because although with no overlap would decrease the correctness, decreasing the noise of original data would increase the correctness. However, after I calculating all the data, I found that ignoring the starting section of EEG signals would not increase the correctness a lot. Instead, the overlap would help a lot on the correctness.

Thus, we started our stage three of trials. Manoj also suggested that we should start from 5 minutes of EEG signal because we would need some of the EEG signals for awake people. Besides, we need to correct the input of the channels in features.py because for each person’s EDF signal file, the channel has different place and we need to manual extract them. Finally, we decided to set up our final calculating information: No overlap, start from 5 minutes(60000), 720 data for each person, 1000 EEG signals for each training data and correct

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Trial count | Number of EEG data for each training data | Number of overlap for calculating each training data | Number of EEG data for starting calculating data | With correct channels? | Features | Number of people involved in training | Correctness |
| 1 | 1000 | 500 | 0 | No | rms, ApprEnt, LZComp, mpf, sef (6 features) | 5 | 92%(one channel) |
| 2 | - | - | - | - | - | 20 | 82%(one channel) |
| 3 | 720 | 0 | 240000 | No | No new features (6 features) | 20 | 77% |
| 4 | 720 | 0 | 60000 | Yes | No new features(6 features) | 20 | 79% |
| 5 | 720 | 0 | 60000 | Yes | Average Frequency, Hjorth Parameters, Total Energy/Power, Band Powers, Delta Ratio, BA Ratio and Beta Ratio | 20 | Pending.. |

Table 1: stages of trials with different parameters

channels. This time, to our surprise, the correctness increased to 79%, which is not a great improving.

In our final stage, we decided to make some great improvement to our correctness by adding more features into it. I first worked on extracting the features in features.py file by adding Average Frequency, Hjorth Parameters, Total Energy/Power, Band Powers, Delta Ratio, BA Ratio and Beta Ratio into our features. I am currently in our final stage and I just finished calculating all the training data. The score is still pending.

1. **EEG ML Code Understanding**

I implemented the ML tools in train.ipynb file. I stored all the training datas in a file for each calculation and extracted them into a number of experimentors\*number of training data 2d array in this file. In the first trial, I tried to separate the channels for each experimentor. However, I found that we can separate each channels with different features as a new features so that we can get more features for each person. For example, at first, we have 6 features for each channels and 6 channels for each person. After combination, we got 36 features for each person.

Then, I started implementing ML tools to our data. In the beginning trial with small amount of experimentors as input, I found that SVM, Decision Tree and Adaboost got better score and using them as input for VotingClassifier, I got a great score. However, after increasing the number of experimentors as input, the RandomForest tool always give me a better score for each trials. Besides, in the paramters of RandomForest, the max\_depth should be around 20 and n\_estimators should be around 280 in order to have a best score. At the beginning, the parameter for DecisionTree has criterion as ‘entropy’, splitter as ‘best’, max\_depth as 10 and max\_leaf\_nodes as 340 for the best score and the best for AdaBoost has n\_estimators as 100 and algorithm as ‘SAMME’ for the best score. I used the VoterClassifier only at the beginning stage because for 20 experimentor trainning, only the RandomForest got a reasonable great score.

1. **EEG new features extraction**

For implementing new features, I want to simply introduce the features that I have implemented. For the Hjorth Paramters, I implemented Activity(Variance), Mobility(Standard Deviation) and complexity based on the introduction of the paper of the project. The Average Frequency is basically the time that EEG signals have passed 0. And for the frequency features, I extracted features based on a paper of Walker lab[[2]](#footnote-2) in UC Berkley. Using the python library scipy, I implemented the Total Power, Band Power, Delta Ratio, BA Ratio and Beta Ratio.

1. **Summary and Future Plan**

Our next step is to finish training the features based our final stage calculation. We want our correctness to be higher than 70 percent and also as better as possible. Then, we would start to implement my algorithm in the lab and record videos for our experiments to evaluate my algorithms. For our future long terms of plan, I would summarize all the process and work for the preparation of the paper. I might need to redo the calculation of each steps and gather the information for each ML tools, that can be filled into the paper.

1. Scikit-learn: It is a popular machine learning tool and public for users to use. <https://scikit-learn.org/stable/> [↑](#footnote-ref-1)
2. Raphael Vallat, Postdoctoral fellow Walker Lab: <https://raphaelvallat.com/bandpower.html> [↑](#footnote-ref-2)