Mobile Computing: BrainNet Individual Project Report

Shiva Chandrashekar ASU ID: 1215106256
snchand1@asu.edu

***Abstract*— The rapid development of EEG sensors has opened the doors for potentially using it as a means for authentication for mobile devices, making it a replacement for the current biometric techniques. This process makes it quicker as well as more secure. The project investigates more on how the data from the EEG sensors (provided before-hand) can be processed in both the local fog server and the remote cloud server. We represent the results by comparing the time taken by the selected server to complete the processing. The processing involves running one of the 4 Machine Learning algorithms (KNN, SVM, Naive-Bayes, Decision trees) for the corresponding edf file(EEG-data) on either of the servers. The validation accuracy for all the techniques is also represented as a graph in the results.**

I. INTRODUCTION

The motivation came from the fact that in current times, most of the secure authentication methods involved multi-factor authentication, where for instance the user had to provide access through an already installed mobile app (DUO Mobile) as well as provide a password. This meant that the user had to spend extra time logging in for the sake of security. With the development of EEG sensors, we are able to provide high security as well as reduce the time taken for users to be able to log in to the system (since it is single-factor authentication). The reason for EEG data being highly secure lies in the fact that it is extremely unique.

II. PROJECT OVERVIEW

there were 3 basic elements to construct the architecture, the features of the components are displayed below.

Android Device

- Model: Mi-A1
- OS: Android Pie - Api 28
- Processor:Snapdragon 625
- RAM:4GB

Fog Server

- Network: Tomcat
- OS: Windows 10
- Processor:Intel core i7-8750H
- RAM:8GB

Cloud Server

- Network: vpc-0c9fa16b
- OS: Linux kernel 4.14
- Processor: 64-bit (x86)
- RAM: 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only

A. Application walk-through

The user is expected to feed in the username, pick the classifier of choice (KNN, SVM, Naive-Bayes, Decision trees) and finally pick the edf file from the drop-down menu as depicted in Fig. 1. Once the user clicks the login button, the app automatically configures which server would be the most efficient in terms of execution time and returns the result as depicted in Fig. 2. The notification box tells us if the user is authenticated, which is the case in the given example meaning that the username and selected edf file do in fact match. So it returns the execution time for the machine learning algorithm and the latency. The selected example runs faster in the fog server.

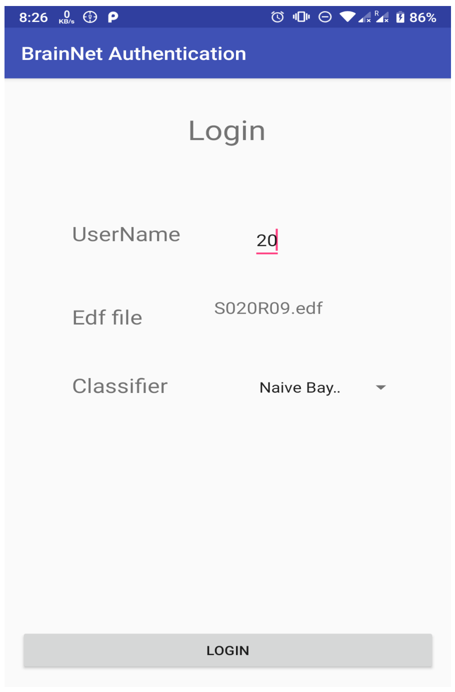
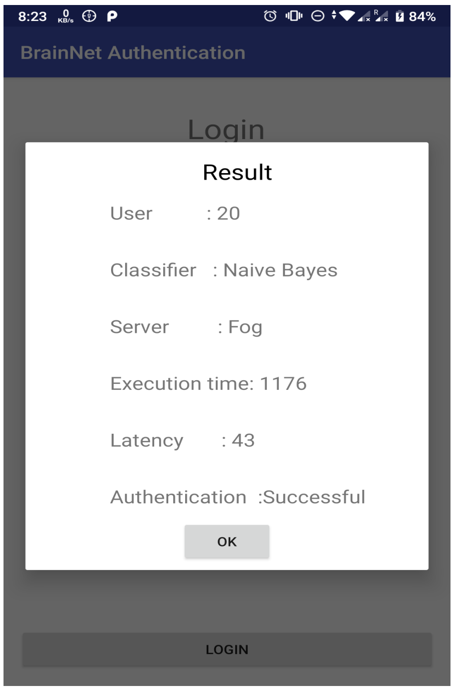

8:26 86%
BrainNet Authentication
Login
UserName 20
Edf file S020R09.edf
Classifier Naive Bay..

Fig. 1: Initial Screen

B. The Algorithm to select the best server

To decide on which server to use for our processing, we compare the latency between the local fog server and the remote cloud. Each time an ML algorithm has to be run, an asynchronous call is sent to each one of the servers and once the response is received, it is parsed and the latency comparison is done to determine the faster server. It was also found that the execution time is almost equal for both


BrainNet Authentication

Login

Result

User : 20

Classifier : Naive Bayes

Server : Fog

Execution time: 1176

Latency : 43

Authentication :Successful

Fig. 2: result screen

the cloud and fog server, hence the latency is used as the most important factor in determining the faster server.

C. *Server Configuration*

Once the edf file is selected in the app, it is directly uploaded into the Amazon S3 bucket. For the cloud server, we have used AWS. configurations were made to the Amazon EC2 instances to run a back end server which is responsible for running the machine learning models and provide the output for the edf file. This is initiated using a rest call.

The Springboot framework will handle The REST calls which in-turn will spawn a thread for each HTTP request to the server. The EC2 instance is constantly listening to receive the HTTP GET requests. Once it is received, the file name is extracted from the URL query, which is used to retrieve the corresponding file from the S3 bucket and run the selected ML algorithm on it. Once processed the output is sent back to the front-end. A public IP is issued by amazon for the EC2 instance can be accessed via HTTP calls to AWS using port 8080. To set up, start the EC2 instance, use ssh to log in to the instance and run the command **java -jar Brainet-1.0.0.jar**.

The fog-servers runs in the desktop using a Tomcat instance running on them (imported with the Springboot framework). The client requests are all placed at port 8080, which is the default setting. The app is designed in a way in which it compares the latency of both the servers and picks the one with the lower latency. The server request is sent with URL which contains the info to which server the algorithm has picked and the file to be uploaded for authentication. Only after the user passes the authentication process, can he/she move into the other tasks in the application

TABLE I: Latency comparison

Algorithm	Fog Server(ms)	Cloud Server(ms)
KNN	30	110
SVM	29	102
Naive-bayes	27	103
Decision trees	34	110

D. *Data-set Details*

We have used the EEG Motor Movement/Imagery Dataset from PhysioNet. There are 14 experiments performed on 64-channel EEG. The data provided was in .edf format which was converted into .csv format for easy implementation of the Machine Learning Algorithms. We were given a total of 109 labels and in each label, there were 14 files(109 * 14 = 1526 samples). The data set was split into 3 chunks. One big chunk which constituted 75% of the whole data-set which was used for training, Another chunk constituting 10% of the total data for cross-validation and the remaining for testing.

III. RESULTS

The figures below (Fig. 3 and Fig. 4) depict the validation and test accuracy for the discussed ML algorithms. It is found that accuracy values do not vary based on which server it runs on. SVM was found to have the best validation and test accuracy and decision trees being the least accurate. The lower accuracy of decision trees is attributed to overfitting and using the smote framework will yield better results. KNN and naive Bayes show very similar results. In terms of latency, as clearly indicated in Table 1, the fog server is twice as fast as the cloud. It was also found that the power consumption for processing does not vary with the servers being used and the usage of battery is negligible.

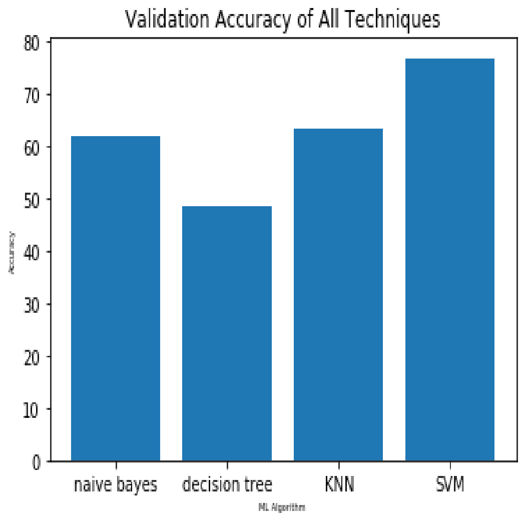


Fig. 3

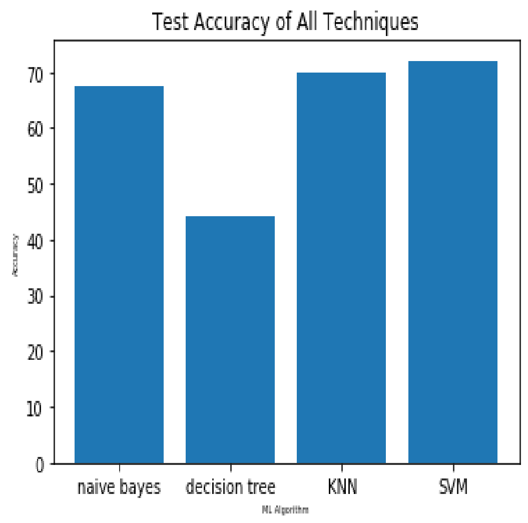


Fig. 4

TABLE II: Division of work

Serial No	Task	Person
1	Data Preprocessing & Analysis	Arvinth, Shiva
2	Cloud Server Setup	Vedavi
3	ML Algorithm Implementation	Shiva, Vedavi
4	Android UI implementation	Arvinth
5	Server Scripts	Shiva
6	Fog Server Setup	Vedavi
7	Network Connections in Android	Arvinth
8	Offloading ALgorithm	Shiva, Vedavi
9	Report Creation	Arvinth, Shiva, Vedavi

IV. CONCLUSION

The implementation takes EEG signals (represented in edf files) as the input in a mobile application and uses it as an authentication mechanism making the process faster and more secure. The various ML algorithms used in the process were processed in 2 different servers, and the most efficient server was automatically configured based on the latency.

Once all the processing is done, the app returns if the user is authenticated or not along with the latency and execution time. It was found that even though it took a considerably longer for collecting EEG data when compared to data for biometric authentication, the uniqueness and single factor capability puts it ahead of its competitors.

V. Contribution to the Project

I began by analyzing and preprocessing the edf file that was provided to understand the nuances of the data, and also learned about how the various ML algorithms would work on the data. Once the cloud server was set up, I wrote the server scripts to connect the front-end to the back-end using the Spring Framework (HTTP request and response). Once I was done with writing the ML implementation, I had to offload it to the remote cloud. To implement the ML techniques, I used the sci-kit learn package on python that provided me with the basic implementation of the actual model.

I was responsible for fitting the data in the format that the package expected. Once I received the output, I fine-tuned the parameters to achieve better accuracy. I also tried to oversample and under-sample the data to achieve better results.

VI. Skills Gained

Writing the server scripts gave me a good understanding of how industries would split their data processing load among their various remote servers. I was able to improve on my existing knowledge of python (pandas) by pre-processing the edf data. practically using the specified ML techniques gave me an understanding of the nuances for Fine-tuning parameters for better accuracy. Apart from the programming, typing out the document, helped me at phrasing sentences and conveying more meaning with fewer words.

Apart from the technical aspect, I was able to improve my verbal communication to a large extent. Since the project required each of the team members to work independently on one particular aspect of the project, I made sure to delegate the responsibilities precisely to avoid any kind of overlap or confusion. Putting together all of our work was also another challenge, which was made easier by efficient communication in our initial phase.

VII. ACKNOWLEDGMENTS

I would like to thank Dr. Ayan Banarjee for allowing me to investigate such a problem and helping us with every step of the way. I would also like to thank the Impact lab and the TAs Mohamad Soudki and Junghyo Lee. Special thanks to my team members for their continued support

- Team Members
  1) Arvinthan Sundaram Govindaraju
  2) Vedavi Balaji

References

[1] Javed Sohankar, Koosha Sadeghi, Ayan Banerjee and Sandeep K.S. Gupta.2015.E- BIAS: A Pervasive EEG-Based Identification and Authentication System. In Proceedings of the 11th ACM Symposium on QoS and Security for Wireless and Mobile Networks(Q2SWinet'15). ACM, New York, NY, USA.

[2] https://physionet.org/pn4/eegmmidb/https://physionet.org/pn4/eegmmidb/.