Mobile Computing Project Portfolio - Biometric Liveness Detection

Sai Prathik Mandyala
smandyal@asu.edu
Arizona State University

Abstract— **Biometric Liveness is about the classification of a biometric signal from a compound signal consisting of an electroencephalogram(EEG) signal, attack vectors and noise. We have built an android application to detect the liveness of a biometric signal. The input brain signal is passed through Internet and the application detects whether the given brain signal is generated from a physically present human being or an inanimate spoof artifact or injected video/data. The application has a Java server actively listening for input brain signal and sends it to liveness detection to a server with trained Machine Learning models to decide the liveness of the brain signal. We have used various combinations of feature extraction methods and machine learning models to provide better accuracy.**

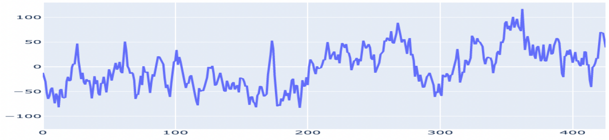


Fig. 1. Raw Input Brain Signal

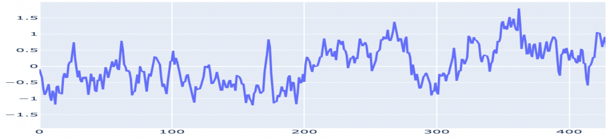


Fig. 2. Normalized input brain signal

I. INTRODUCTION

Security applications are vulnerable to attacks, despite numerous developments in biometric systems. We created an android application that accepts a brain input signal and identifies the biometric signal's liveness in this project. The time series data in the form of a.mat file is utilised as the input brain signal. This project's major purpose may be broken into two parts:

1) The input signal should belong to a live human being and not artificially generated.
2) The input brain signal being captured at the current point of time and not prerecorded.

In this android application we have used client server architecture to send input brain signal to server and display the response on android application. The application shows the server response on the application screen by displaying appropriate symbols. The time-series EEG input signal is pre-processed before extracting features using standard deviation normalization. Let's say that the input brain signal is a set of observations ordered in time, observed at discrete set of evenly spaced time intervals: x_t at times t=1,2,3,....,N where N is the length of the time series. The input brain signal is modified using the following formula :

$$\frac{y - mean}{SD}$$

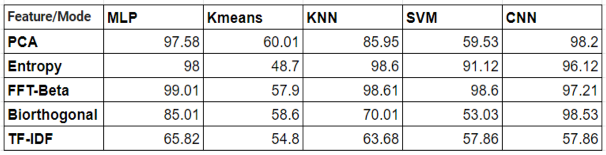
where mean is arithmetic mean and SD is standard deviation.

The pre-processed time series data is sent to various feature extraction techniques like Principle Com- into the desired dimension, we have used Machine Learning models like Support vector Machine(Linear Kernel), Multi-layer Perceptron(MLP), K-means, K-Nearest Neighbours(KNN), Convolution Neural Network(CNN), and the ensemble model to classify whether the brain signal is fake or real.

We have tried various combinations of features and machine learning models to provide better accuracy and the same is presented in the report. In order to make our model robust enough, we have generated attack signal data using Noise Addition, Random Signal Generation and Generative Adversarial Network(GAN) signal generation techniques.

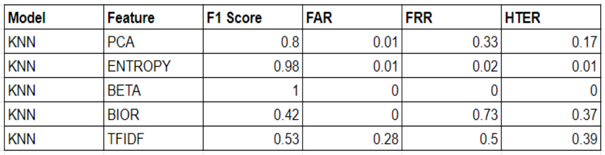
II. RESULTS

As discussed during the 80% completion meeting, for extra credit we have provided all 36 combinations. When using the PCA as a feature, we are first taking all the features and then applying PCA on top of it, we are taking top 20 PCA components when applying the PCA. This leads to 66.23% coverage. For comparison we are taking accuracy, FAR, FRR, HTER and F1 score metrics for each of the combination.



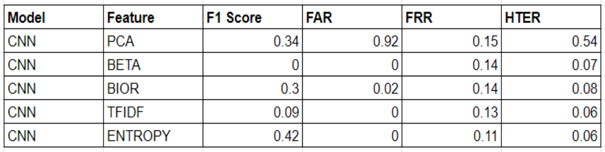
Feature/Mode	MLP	Kmeans	KNN	SVM	CNN
PCA	97.58	60.01	85.95	59.53	98.2
Entropy	98	48.7	98.6	91.12	96.12
FFT-Beta	99.01	57.9	98.61	98.6	97.21
Biorthogonal	85.01	58.6	70.01	53.03	98.53
TF-IDF	65.82	54.8	63.68	57.86	57.86

Fig. 3. Accuracies for all features against all Machine Learning models



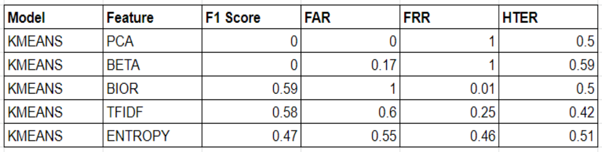
Model	Feature	F1 Score	FAR	FRR	HTER
KNN	PCA	0.8	0.01	0.33	0.17
KNN	ENTROPY	0.98	0.01	0.02	0.01
KNN	BETA	1	0	0	0
KNN	BIOR	0.42	0	0.73	0.37
KNN	TFIDF	0.53	0.28	0.5	0.39

Fig. 4. Output metric for all features against KNN model



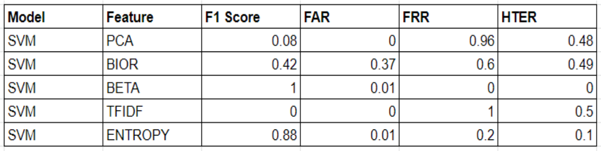
Model	Feature	F1 Score	FAR	FRR	HTER
CNN	PCA	0.34	0.92	0.15	0.54
CNN	BETA	0	0	0.14	0.07
CNN	BIOR	0.3	0.02	0.14	0.08
CNN	TFIDF	0.09	0	0.13	0.06
CNN	ENTROPY	0.42	0	0.11	0.06

Fig. 5. Output metric for all features against CNN model



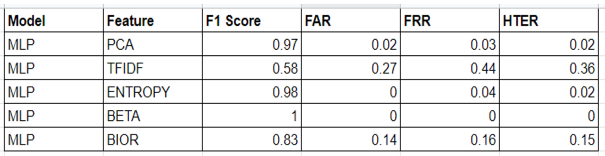
Model	Feature	F1 Score	FAR	FRR	HTER
KMEANS	PCA	0	0	1	0.5
KMEANS	BETA	0	0.17	1	0.59
KMEANS	BIOR	0.59	1	0.01	0.5
KMEANS	TFIDF	0.58	0.6	0.25	0.42
KMEANS	ENTROPY	0.47	0.55	0.46	0.51

Fig. 6. Output metric for all features against K-means model



Model	Feature	F1 Score	FAR	FRR	HTER
SVM	PCA	0.08	0	0.96	0.48
SVM	BIOR	0.42	0.37	0.6	0.49
SVM	BETA	1	0.01	0	0
SVM	TFIDF	0	0	1	0.5
SVM	ENTROPY	0.88	0.01	0.2	0.1

Fig. 7. Output metric for all features against SVM model



Model	Feature	F1 Score	FAR	FRR	HTER
MLP	PCA	0.97	0.02	0.03	0.02
MLP	TFIDF	0.58	0.27	0.44	0.36
MLP	ENTROPY	0.98	0	0.04	0.02
MLP	BETA	1	0	0	0
MLP	BIOR	0.83	0.14	0.16	0.15

Fig. 8. Output metric for all features against MLP model

III. CONTRIBUTIONS

Feature Extraction, Machine Learning Model Implementation, Attack Signal Generation, and Android Application Backend Development were the four primary components of our project. My role covered a diverse range of tasks within Feature Extraction, Attack Vector Generation and Machine Learning Model Implementation. I have listed out the tasks under my responsibility as part of the project below :

1) **Ensemble Learning:** We used five Machine Learning models CNN, KNN, K-means, SVM and MLP. Then we used an ensemble model based on each model's votes on the signal's liveness. For the input brain signal, a majority decision will be used.
2) **Unsupervised Learning:** For unsupervised Learning, I utilised the K-means method. K-means finds the closest cluster for each data point after identifying k centroids. It adjusts it throughout several epochs until the centroids' co-ordinates do not change. In this situation, I used three for k. The input brain activity is matched to the closest cluster in this model, which divides the training data into three clusters. The label of the assigned cluster's centroid is applied to the input. The K-means model was created with the sklearn package.
3) **Signal Generation Using GAN :** For the signal production component, we used the Generative Adversarial Network (GAN) to generate attack time-series. GAN generates data that reflects the complex dynamics of timeseries data over long periods of time. We used Jinsung Yoon et al TimeGAN's method, which was previously recommended. TimeGAN is a GAN network that uses supervised loss to capture the conditional distribution of time in EEG data, making it unique among GAN networks. To make the data compatible with the TimeGAN model, we first converted it to three dimensions. The test data was then used to train the TimeGAN synthesiser, which was then saved for future use. We next

made synthetic EEG data using the synthesiser we saved in the previous phase, trained a Recurrent Neural Network (RNN) on both the test and synthetic EEG data, and compared the results.

4) **Random Inputs :** We make use of the inbuilt python random input generator for this task. We generated random inputs equivalent to the number of features of input signals and then generated enough signals each containing the random inputs as generated by the randomizer.

5) **Term Frequency-Inverse Document Frequency :** I contributed in this part of the feature extraction process. Term frequency and inverse word frequency show us a two way image of a particular words importance across documents with frequency and inverse frequency. The term frequency measures the count of a word in a particular document and inverse document frequency measures frequency of the word across the remaining documents. TFIDF value is the product of TF X IDF value. Since this is not a NLP related work, we assumed the discrete signals as words. To convert the signal into words we rounded the EEG values to two decimal places and converted it into a string. Then the TFIDF logic was applied to each input signal. We have used scikit learn library to implement TF-IDF feature extraction

IV. Conclusions

The results and conclusion discussed above are Accuracy, F1 score, FAR(False Acceptance Rate), FRR(False Rejection Rate) and HTER(Half Total Error Rate) for 30 combinations of 5 features(TF-IDF, PCA, Entropy, FFT(Beta band), and Biorthogonal(Bior)) and 6 Machine Learning models(Knn, K-means, SVM(Linear Kernel), CNN, MLP, and Ensemble) are calculated for training as well as test data. The analysis and visualization with the data is shown in the Jupyter notebook attached.

We observe that, when it comes to accurate predictions, non linear machine learning models like Multi Layered Perception is the best when it comes to identifying fake or attack signals. The primary reasoning being, it utilizes multiple hidden layers which gives it the ability to find inherent differences which normal linear models like SVM would not be able to identify, as their hypothesis imitates a line equation.

One more striking thing that we observed was the accuracy of the CNN model. The [1] paper discusses that in detail. We too have observed that CNN was picking up good accuracy even for a few epochs. CNN is majorly only used in domain of images, but seen in a different perspective we can apply them to signal processing. So CNN's capability of finding local features and patterns using it kernels and max pooling layers would have seen what normal ML models would not see.

Out of the features taken into consideration for the project we have seen that Fourier Transform and PCA do a good job when it comes to dimensional reduction and feature extraction. PCA has shown us promise as it reduces the dimensions and converts the signal into representative array which have the best information to describe them. This quality might have made our predictions accurate. FFT on the other hand converts the signal into energy domain with frequency and phase. We feel that this decomposition into frequency domain unearths features which can be exploited by ML models when it comes to predictions or classification, like how we have observed in this project.

The Liveliness Detection model, even though it has a good accuracy we should always keep in mind that we are predicting the EEG value depending on the highest accuracy of our models. Their is a probabilistic reasoning behind all the assumptions and it relies on Bayesian's logic used to rightly detect them. Thus we are stating the accuracy, the assumption, and seeing that it is related to medical field detecting this could be taken as a disclaimer for the feasibility of our solution. On the bright side all Machine Learning models do have to go through it and we have done a good job putting forth the pro and cons.

V. Lessons Learned

I discovered that, like time series data in general, brain signals aren't particularly helpful for direct usage in machine learning models in their raw form, and that effective feature extraction is necessary to extract relevant information from them. To speed up development and testing, use a sample of the available datasets and store intermediate datasets in files.

References

[1] S. Kiranyaz, T. Ince, O. Abdeljaber, O. Avci and M. Gabbouj, "1-D Convolutional Neural Networks for Signal Processing Applications," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 8360-8364, doi: 10.1109/ICASSP.2019.8682194.

[2] Zahid Akhtar, Christian Micheloni, G.L.Foresti, ”Biometric Liveness Detection: Challenges and Research Oppurtunities”, September 2015IEEE Security and Privacy Magazine 13(5):63-72 DOI:10.1109/MSP.2015.116

[3] Stéphane G. Mallat (1999). ”A Wavelet Tour of Signal Processing. Academic Press.” ISBN 978-0-12-466606-1.

[4] Van Loan, Charles (1992). ”Computational Frameworks for the Fast Fourier Transform. SIAM”.

[5] Kent, Ray D.; Read, Charles (2002). ”Acoustic Analysis of Speech.” ISBN 0-7693-0112-6.

[6] Barnett, T. P. & R. Preisendorfer. (1987). ”Origins and levels of monthly and seasonal forecast skill for United States surface air temperatures determined by canonical correlation analysis”. Monthly Weather Review. 115 (9): 1825

[7] Danielson, Gordon C.; Lanczos, Cornelius (1942). ”Some improvements in practical Fourier analysis and their application to x-ray scattering from liquids”. Journal of the Franklin Institute. 233 (4): 365–380.

[8] Geiger, Bernhard; Kubin, Gernot (January 2013). ”Signal Enhancement as Minimization of Relevant Information Loss”. Proc. ITG Conf. On Systems, Communication and Coding.

[9] YShentov, Ognjan V.; Mitra, Sanjit K.; Heute, Ulrich; Hossen, Abdul N. (1995). ”Subband DFT. I. Definition, interpretations and extensions”.

[10] Danielson, Gordon C.; Lanczos, Cornelius (1942). ”Some improvements in practical Fourier analysis and their application to x-ray scattering from liquids”. Journal of the Franklin Institute. 233 (4): 365–380. doi:10.1016/S0016-0032(42)90767-1

[11] Tawhid MNA, Siuly S, Wang H, Whittaker F, Wang K, Zhang Y (2021) A spectrogram image based intelligent technique for automatic detection of autism spectrum disorder from EEG. PLoS ONE 16(6): e0253094. https://doi.org/10.1371/journal.pone.0253094