Person Identification and Verification with Signature Images and EEG Signals

Sweety Shukla

Department of Computer Science and Engineering
National Institute of Technology Rourkela

Person Identification and Verification with Signature Images and EEG Signals

Thesis submitted in partial fulfillment

of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Sweety Shukla

(Roll Number: 116CS0236)

based on research carried out

under the supervision of

Prof. Korra Sathya Babu

May, 2020

Department of Computer Science and Engineering
National Institute of Technology Rourkela

Department of Computer Science and Engineering
National Institute of Technology Rourkela

Prof. Korra Sathya Babu
Assistant Professor

May 25, 2020

Supervisor's Certificate

This is to certify that the work presented in the thesis entitled *Person Identification and Verification with Signature Images and EEG Signals* submitted by *Sweety Shukla*, Roll Number 116CS0236, is a record of original research carried out by her under my supervision and guidance in partial fulfillment of the requirements of the degree of *Bachelor of Technology* in *Computer Science and Engineering*. Neither this thesis nor any part of it has been submitted earlier for any degree or diploma to any institute or university in India or abroad.

_____
Korra Sathya Babu

Declaration of Originality

I, *Sweety Shukla*, Roll Number *116CS0236* hereby declare that this thesis entitled *Person Identification and Verification with Signature Images and EEG Signals* present my original work carried out as a undergraduate student of NIT Rourkela and, to the best of my knowledge, contains no material previously published or written by another person, nor any material presented by me for the award of any degree or diploma of NIT Rourkela or any other institution. Any contribution made to this research by others, with whom I have worked at NIT Rourkela or elsewhere, is explicitly acknowledged in the dissertation. Works of other authors cited in this dissertation have been duly acknowledged under the section "Reference'. I have also submitted my original research records to the scrutiny committee for evaluation of my dissertation.

I am fully aware that in case of any non-compliance detected in future, the Senate of NIT Rourkela may withdraw the degree awarded to me on the basis of the present dissertation.

May 25, 2020
NIT Rourkela

Sweety Shukla

Acknowledgment

I am sincerely grateful to Professor Korra Sathya Babu for giving me the chance to work in the field that I am interested in. The opportunity has not only helped me refine my practical skills, but has also enabled me explore beyond the limitations of a well-defined curriculum. He has also helped me take an interest in academic writing and guided me for it, which will prove to be invaluable further along my career. I shall remain indebted for his guidance and support.

May 25, 2020
NIT Rourkela

Sweety Shukla
Roll Number: 116CS0236

Abstract

Signature is a unique biometric attribute. The handwritten sign of an individual is utilized as a recognizable proof of a person on the grounds that every individual has an unmistakable signature and each mark has its own behavioral or physiological qualities. Be that as it may, signature as a recognizable proof model can be effortlessly duplicated, emulated or even stolen and therefore it has certain drawbacks; for example, lack of reliability and security. In this work, a multimodal person recognition system has been proposed that works by combining features of two inter-connected biometric qualities, - handwritten signatures and signals of the brain (Electroencephalogram). The advantage of using EEG along with signatures is that it is highly confidential and so unique to a person that it cannot be stolen or even copied. In this work, we have proposed to combine both the traits and to use machine learning techniques and neural networks to build an EEG-signature based person identification and verification system.

Keywords*: *Biometrics*; *Signature Recognition*; *EEG*; *Neural Networks.

Contents

Supervisor's Certificate ii

Declaration of Originality iii

Acknowledgment iv

Abstract v

List of Figures viii

List of Tables ix

1 Introduction 1

2 Related Work 2

3 Methodology 5

3.1 Brain Computer Interface 5

3.2 Electroencephalogram (EEG) Signals 5

3.3 Convolutional Neural Networks 6

3.4 VGG 16 Model 8

3.5 Principal Components Analysis 8

3.6 Artificial Neural Networks 8

3.7 Random Forest 10

3.8 Bagging Classifier 10

3.9 ADA boosting 10

4 Experiment and Results 12

4.1 Problem Statement 12

4.2 Software Installation 12

4.3 Overall Workflow 13

4.4 Dataset Description 13

4.5 Dataset Preprocessing 14

4.6 Model 1- CNN for verifying users based on signature images only 14

4.7 Model 2- ANN model for verifying users by using EEG signal data 17
4.8 Model 3- ANN model for verifying users by using EEG and signature images 19
4.9 Model 4- CNN model for verifying users using the EEG and signatures . . 22
4.10 Model 5- CNN (with PCA) model for verifying users using the EEG and signatures 25
4.11 Model 6- Ensemble model (PCA+RF+Bagging) for verifying users with EEG and signature images 28
4.12 Model 7- Ensemble model (PCA+RF+AdaBoosting) for verifying users with EEG and signature images 29

5 Analysis and Discussion **31**
5.1 Neural Networks Configurations 31
5.2 Accuracy and Loss metrics of Models 31
5.3 Analysis - Pros, cons and Novelty of Models 32

6 Conclusion **33**

References **34**

List of Figures

3.1 CNN Architecture 7
3.2 ANN Architecture 9

4.1 Flowchart of the Work 13
4.2 Structure of VGG model 15
4.3 Summary of CNN (Model 1) 15
4.4 Train and Val Set Accuracy Plots of CNN (Model 1) 16
4.5 Train and Val Set Loss Plots of CNN (Model 1) 16
4.6 Running of the CNN (Model 1) 17
4.7 Summary of ANN (Model 2) 18
4.8 Train and Val Set Accuracy Plots of ANN (Model 2) 18
4.9 Train and Val Set Loss Plots of ANN (Model 2) 19
4.10 Running of the ANN (Model 2) 19
4.11 Summary of ANN (Model 3) 20
4.12 Train and Val Set Accuracy Plots of ANN (Model 3) 21
4.13 Train and Val Set Loss Plots of ANN (Model 3) 21
4.14 Running of the ANN (Model 3) 22
4.15 Summary of CNN (Model 4) 23
4.16 Train and Val Set Accuracy Plots of CNN (Model 4) 24
4.17 Train and Val Set Loss Plots of CNN (Model 4) 24
4.18 Running of the CNN (Model 4) 25
4.19 Summary of CNN (Model 5) 26
4.20 Train and Val Set Accuracy Plots of CNN (Model 5) 27
4.21 Train and Val Set Loss Plots of CNN (Model 5) 27
4.22 Running of the CNN (Model 5) 28
4.23 Confusion Matrix for Ensemble Technique (Model 6) 29
4.24 Confusion Matrix for Ensemble Technique (Model 7) 30

List of Tables

2.1 Optimal Neural Network Configurations in Each Method 2
2.2 Total Recognition Rate (RR) in Each Method 2
2.3 Mean Correct Recognition Rates (CRR) 3

5.1 Optimal Neural Network Configurations 31
5.2 Results Obtained for Each Method 31

Chapter 1

Introduction

Person identification has huge demands in many areas, especially in security related domain. It has applications in places such as mobile/desktop login, building entryway control, computerized media access, exchange confirmation, secure remote access, and official update of database. Biometric frameworks utilize various types of unique attributes such as fingerprint sensing, face recognition, voice recognition and handwritten signatures. However, these methods may suffer from a number of drawbacks which include imitation, less reliability and lesser security. [1] Specifically, signatures as a measure of biometric do not provide 100 % valid and authentic verification and identification. They are highly prone to forgery, stealing and misutilization. Most of the times, it is difficult to distinguish the forged signs from the genuine ones.

Electroencephalogram (EEG) signals are the unique signatures of the brain's neural activity in real time. Since each biometric attribute has its own upsides and downsides, scientists are investigating new methods of person identification depending on various circumstances. As of now, most EEG signals have been applied in R & D of BCI where principle objective is improving the correspondence and commanding of voluntary organs capacities of the incapacitated/specially abled individuals. [2]

In this work, I have proposed a novel multimodal individual recognition by joining two connected biometric characteristics - signature and EEG signals (Electroencephalograms). When a person signs, those handwritten strokes generate EEG signals in the cerebrum. At the point when an individual is signing on any piece of paper, EEG catches unique signs for the different people from the multiple anodes placed on the skull/cortex, or in certain areas on the cranium. The advantage of utilizing EEG alongside signature is that it is exceptionally private (since it relates to a motor-brain coordination), it is difficult to mimic (as cerebral is related to stress and mind-set of an individual, an assailant can't drive the individual to replicate his/her psychological condition exactly same as in the normal condition) and reasonably resistant to theft.

Chapter 2

Related Work

In the paper by Howida AbdelFattah Shedeed [3], individual recognition was implemented by utilizing the EEG signals. Only four channels were used to update the framework strength and they accomplished a detection rate for three subjects that came to 100%. Four diverse component extraction procedures (DFT and Wavelet Packet Decomposition) were tried, Multi-layer Perceptron (MLPs) Neural Network prepared by a standard back-propagation calculation was utilized for characterization in this model. The best three attributes extraction methods that created top order results were utilized at the end to improve the recognition precision to 100%. The results they they obtained as summarized in the following Table 2.1 and Table 2.2.

Table 2.1: Optimal Neural Network Configurations in Each Method

Model No	No. of Input Nodes	No. of epochs	No. of hidden layer	Nodes in hidden layer
1st	92	100	1	1000
2nd	40	100	1	100
3rd	72	1000	1	100
4th	96	1000	1	100

Table 2.2: Total Recognition Rate (RR) in Each Method

Method	RR1	RR2	RR3	Total Correct Ratio	Total Rate (in %)
1	0.8	1	1	0.933	93
2	0.6	0.6	0.8	0.667	66
3	1	0.8	1	0.933	93
4	0.8	0.8	1	0.877	87

Hend A Hadi had done EEG based Individual Verification and Identification on CSU Dataset by utilizing the energy of Sliced DFT Spectra [4]. In this paper, the use of lowest number of tasks and channels (i.e., one channel per task) was tested to achieve

high recognition rates without the need of features fusion step. This kept the required computational complexity as low as possible. The discrete Fourier transform was applied, and a set of mean energy values each taken for certain partition of the frequency spectra was proposed to be the feature vector, then the intra/inter scatter analysis was performed and the normalized Euclidean distance measure was used. An accuracy of 99.34% was reached on user verification and 100% on identification.

T.Wilaiprasitporn [5] had done EEG based individual recognition using the deep learning approach. They used EEG data from DEAP which had 32 subjects in total. The model used was CNN (for capturing spatial patterns in EEG data) and GRU/LSTM (for capturing temporal patterns). The deep neural network model began with two dimensional Convolution NN (applied to the structure of the mesh) for three hidden layers. Considering it as a periodic data, the two dimensional Convolution NN was applied to each fixed time sliding window, with common variables/parameters among them. The structure hence formed is known as a Time Distributed 2D-Conv NN layer. After that, a Time Distributed (FC) layer was implemented for feature extraction and subsampling. Capturing of time distribution trend in the structure was done by 2 repetitive layers (RNN such as LSTM or GRU layers) which was implemented as per the dimension of the fixed time duration (sliding) windows/slots. At last, Fully Connected layer was added to the repetitive output at the last time step with a softmax work function. They got the outcomes as appearing in table 2.3.

Table 2.3: Mean Correct Recognition Rates (CRR)

	Mean Correct RR %		
States	Conv-GRU	Conv-LSTM	SVM
LL	99.90 ± 0.10	99.79 ± 0.12	33.02 ± 1.58
LH	99.71 ± 0.19	100	36.38 ± 1.71
HL	99.86 ± 0.14	99.86 ± 0.14	36.25 ± 2.45
HH	99.87 ± 0.12	99.74 ± 0.26	33.59 ± 1.65
All States	100.0	99.79 ± 0.14	33.02 ± 1.58

In the paper by Xu huang [6], they did person identification on 2 datasets (one having 45 subjects and the other having 122 subjects). They used MATLAB builtin nn-traintool to make a classifier with feed forward error back propagation network. The activation function used by the NN engine was continuous tan-sogmoid. The first classifier was trained to correctly recognize all the forty five subjects involved in the testing process with a MSE of $1.98 * 10^{-7}$. The second classifier was able to effectively recognize 116 out of 122 people with an error of 0.00186.

Rajkumar Saini, in their paper [7], tried identifying users with signature and EEG with the help of HMM model. By using PHOG feature extraction on the signature images dataset, they got an accuracy of 84.9 % (6 HMM states and 5 GMMs). By using only EEG signals, they got an accuracy of 95.65 % (6 HMM states and 8 GMMs).With early fusion technique on signature and EEG data, they got an accuracy of 93% (6 HMM states and 8 GMMs) and with late fusion technique, they achieved an accuracy score of 98% .

Chapter 3

Methodology

3.1 Brain Computer Interface

Brain Computer Interface (BCI) as a measure of biometric traits are non-invasive in nature. A few techniques utilizing cerebrum activity signals are Electroencephalogram (EEG), Magnetoencephalography (MEG), Functional Magnetic Resonance Imaging (fMRI), and Near Infrared Spectroscopy (NIS). Be that as it may, MEG, functional MRI and NIS are costly or cumbersome. Functional MRIs and NISs don't gauge neural movement and subsequently, cannot be reliably used as mobile BCI frameworks. EEG signals give the signals of neural movements and thus, are being looked into in order to utilize them as biometric qualities.

3.2 Electroencephalogram (EEG) Signals

The electroencephalogram (EEG) is a recording of the activity/movement of the neurons in our brain. The pre-synaptic neuron is converted into a chemical signal in the synaptic cleft, and finally, back into an electrical signal in the post-synaptic neuron. When neurotransmitters cross the synapse cleft and bind onto post-synaptic receptors, this starts a process that temporarily alters the electrical charge of the post-synaptic neuron. Millions of such neuron activities taking place together enable the EEG equipment to detect the electrical changes underneath an electrode.

The main frequencies of human EEG signals are as follows:

1. Delta waves have the lowest frequency - between zero and four hertz.[8] The delta waves are generated from the deepest portion of the human brain. These waves have a very low frequency and a very high amplitude. They are also associated with the highly conscious state of our brain. These waves lead to the release of hormones such as prolactin. They are generally associated with dreamless sleep action of the brain. These waves signify that the cortex of the brain has been inhibited.

2. Theta waves frequency range lies in between four and eight hertz. They occur most often during normal sleep or even during deep meditation. Theta waves signify the creativity and deep trance of the brain. It is generally native to the hippocampus of the brain. The high frequency of the waves leads to a decrease in their amplitude.

3. Alpha waves have a frequency in the range of 8-13 hertz, these waves occur when our brain is in a relaxed state. They are slowly changing in nature and hence, last for a longer period of time compared to the beta waves. They are responsible for stimulating creative thinking in our brain. It resembles that the brain is in a relaxed mode and leads to serotonin production.

4. Beta waves range between 13 and 30 hertz. They are associated with being normally awake and attentive towards cognitive tasks or the outside world. It is usually seen on the both sides of the brain in balanced conveyance and is significantly apparent frontally. It is enhanced by sleep inducing drugs such as the benzodiazepines and the barbiturates. It can get reduced or become completely missing in regions of cortical harm.

5. Gamma waves lie in the frequency range of 30-40 Hertz. Their amplitude is low. Gamma waves can be used to detect and confirm brain disorders and are a significant indicator of brain synchronization.

3.3 Convolutional Neural Networks

The CNN model consists of mainly three layers namely the Convolutional layer, the pooling layer (max or mean) and the third part is a dense layer. A CNN model is generally able to get and detect the patterns in any dataset. They perform this feature extraction by using the filters. Filters are randomly initialized matrices which are of a specified size and slide over the entire input performing the convolution operation on them and extracting the important features of them. The number of filters is an important parameter to detect the features in the input. Convolution can be defined as follows:

$$c(n) = \Sigma_{k=0}^{\infty} a(k)b(n_0 - k) + \Sigma_{k=-1}^{-\infty} a(k)b(n_0 - k) \tag{3.1}$$

$where$
$c = convolution\, output$
$b = input\, signal$
$a = excitation\, signal$

The second part of the CNN performs the pooling operation, (for example: max pooling technique takes the maximum of each of the filters that are applied to it. This max pooling operation helps to shorten the input for the next layer and makes the learning of the model easier compared to the initial layer. Max pooling is described by the following equation:

$$n_{out} = \frac{n_{in} + 2p - k}{s} + 1 \tag{3.2}$$

$where$
$n_{in} = input\ features$
$n_{out} = output\ features$
$k = kernel\ size$
$p = pool\ size$
$s = stride\ size$

Then the matrix is flattened to form a column vector. Finally, the dense component is used for generating the network output with the classified class using the softmax activation function which can be mathematically given by:

$$S(x_i) = \frac{e^{x_i}}{\Sigma_j\, e^{x_j}} \tag{3.3}$$

$S(x_i) = softmax\ output\ for\ x_i$
$x_i = input\ value\ to\ the\ softmax$

The maximum output of the softmax layer is then taken and finally the classified output is obtained.

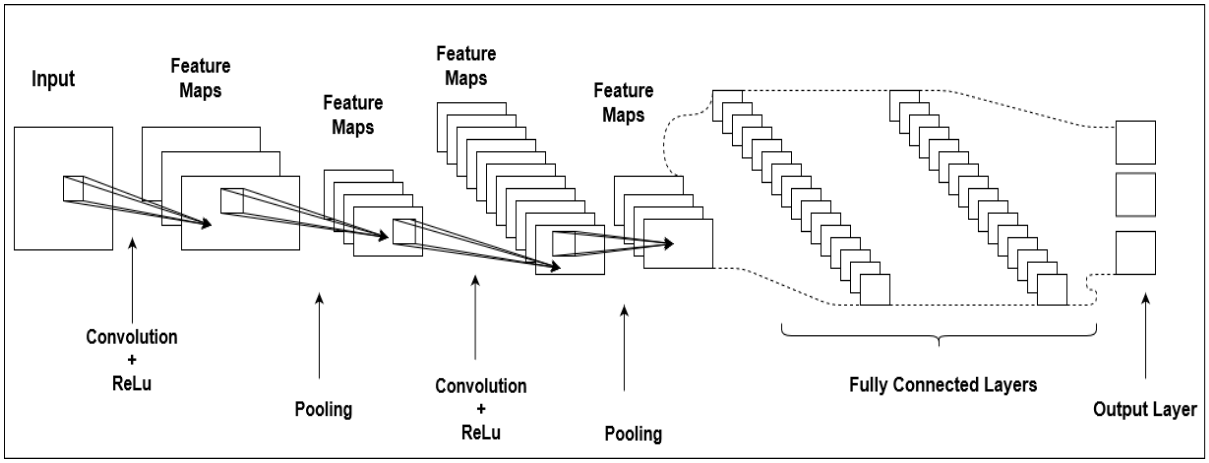

Figure 3.1: CNN Architecture

The model follows a path of backpropagation as per the minimum loss path, here the loss function used is categorical crossentropy loss function which can be given by:

$$CE = -\Sigma_i^C \, t_i log(s_i) \tag{3.4}$$

$where$
$CE = Loss\, for\, the\, iteration\; C = Classes\, in\, the\, dataset\; t_i = groundtruth\, class\; s_i = Predicted\, class$

3.4 VGG 16 Model

The VGG 16 is a type of very deep Convolutional Neural Network model which is used for image classification datasets. It has 16 layers and was developed by the VGG team to work on the Image Net dataset. The model has softmax as the final layer activation function and its weights are best adapted for recognition tasks.

3.5 Principal Components Analysis

PCA finds an additional positioning of dimensions (or a premise of perspectives) with an objective that all the measurements are linearly independent and symmetrical and positioned according to the difference of information that they carry along them. It means that more important principal axis (progressively spread out data) occurs first. Hence, PCA is widely used for feature extraction and dimensionality reduction process before feeding the data into the models.
For the computation of PCA, the mean of the data is first removed from the data so that it becomes centred at zero, and then the covariance of the data is calculated by the following formula:

$$C = \frac{1}{N-1} X^T X \tag{3.5}$$

$where$
$C = co - variance\, matrix$
$N = Number\, of\, data$
$X = Matrix\, after\, zero\, centering$

The principal components are then extracted by computing the eigen vectors of the covariance matrix.

3.6 Artificial Neural Networks

Artificial NNs consists of interconnections, which try to mimic natural human intellect by updating the connection weights. The neurons are associated by connections and they

communicate with one another. The interconnections can take input information and perform standard operations on them. The consequence of these procedures are passed to different neurons. The yield at every interconnection is called its activation. Each interconnection is related via weights. Artificial NNs are designed for mimicking natural neurons, which happens by constantly modifying weights. If the model gives a desired/good yield, there is no need to modify the weights any further.

The model uses RelU (Rectified Linear Unit) activation functions which are given by the following:

$$f(x) = \begin{cases} 0 & \text{if x} \leq 0 \\ x & \text{if x} > 0 \end{cases}$$

$where$

$f(x) = function\, output$

$x = input$

The model also uses the Leaky ReLU function which can be given by the following formula:

$$f(x) = \begin{cases} ax & \text{if x} \leq 0 \\ x & \text{if x} > 0 \end{cases}$$

$where$

$f(x) = function\, output$

$x = input$

In any case, in the event that the model creates a undesired/poor yield or errors, at that point, the system modifies the loads so as to improve the outcomes.

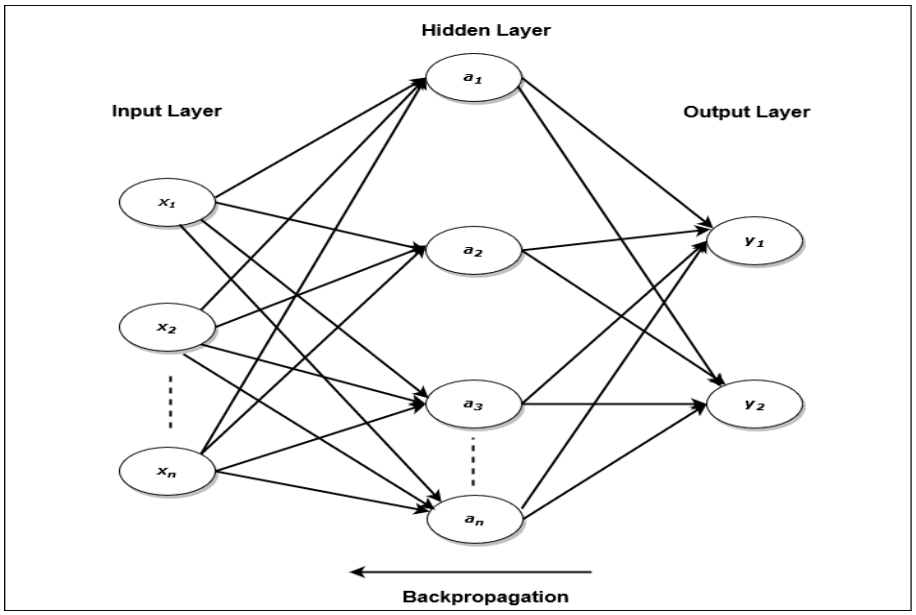

Figure 3.2: ANN Architecture

3.7 Random Forest

Decision trees are a popular form of Machine Learning algorithms which are used for regression and classification datasets. It uses a ”how to” basis for classification, asking the questions which match the output format in the best way. Hence, the classifier runs on the basis of extracting the most important features from the data. The problem faced with the decision trees is that they are prone to have a data overfitting over the dataset.
The random forest model uses a number of decision trees by adding more generalization and also at the same time decreasing the probability of the model overfitting at any point on the training set. They are a form of ensemble learning algorithms which use the multiple decision trees for predicting or classifying. The random forests achieve a higher amount of accuracy by introducing random variations in the trees. The only thing that has to be ensured to make a random forest reproducible is to make the random state parameter always stay fixed. Hence, by adding a random variation in building the trees and by choosing random features the model becomes an improvement on the existing decision trees.

3.8 Bagging Classifier

Bagging is the short form for bootstrap aggregating. When data is fit by using complex models, averaging out of models leads to a smoother fitting and also to a balance between the bias and the variance in the model. This also helps in decreasing the probability of occurrence of an overfitting. The bagging classifier upon receiving the input data, resamples it as the first step. Once resampling is done it takes the classifier predictions on this new data. Finally, the majority vote is taken in the end to obtain the classification result. Bagging classifiers are generally used in concatenation with ensemble learning classifiers such as decision trees or random forests to improve their accuracy and at the same time reduce the chances of overfitting of the data.

3.9 ADA boosting

The ADA standing for Adaptive Alpha boosting algorithm belongs to the family of boosting algorithms which help to convert the weak learnt algorithms to strong learnt algorithms. The ADA algorithms sees each of the data point in the same way and does not try to classify the rare data or the commonly occurring data. The model at first gives an equal weight to all the data points as $\alpha_i = \frac{1}{N}$ where the α_i is the weight of any particular data point. The algorithm begins with an average weight since that provides for the easier and faster

learning for the boosting process.

The model first moves through the data points by learning with the data f_t with the weights as α_i. Once iterated through the dataset, it then tries to assign the new weights to the data points. The points where the classifier f_t runs good the weight of those needs to be increased and the points where the f_t fails needs to be given less weights and this is done by the model using a weighted error processing. The weighted error formula can be given as follows:

$$WeightedError = \frac{\text{Total weight of mistakes}}{\text{Total weight of all data points}} \tag{3.6}$$

Now the model runs on these iteratively till it has found the f_1, f_2, f_3, f_4. These are the four best classifiers obtained as per the training data of the model. The prediction of the output class $\hat{y}$ is done by the model by doing the weighted combination of all these four classifiers. Hence, the model helps in the boosting of a weak learner by combining a bunch of weak learners with the some weights as per the classifier performance on the data.
For the evaluation of a model the various performance metrics that can be used are as listed below:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{3.7}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.8}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.9}$$

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.10}$$

$where$
$TP =$ True Positive
$FP =$ False Positive
$TN =$ True Negative
$FN =$ False Negative

Chapter 4

Experiment and Results

4.1 Problem Statement

Designing a robust multimodal approach for person identification and verification with the use of both EEG and signature data. Developing several models and comparing them on the basis of various performance metrics such as accuracy(training, validation and testing), precision, recall, f1 score, time taken for model training and overall model complexity.

4.2 Software Installation

All the codes were executed in Python 3 on Spyder IDE (in Anaconda Navigator). The system had a RAM of 8GB, intel i5 CPU with a GPU (Nvidia GeForce 920 M- 2GB graphics card memory). The following software and libraries were installed:

Anaconda Navigator: It is a deep learning software that allows launching applications and managing machine learning and deep learning packages, environments, and channels. It is available in Windows 7 and above, Macintosh operating systems, and Linux (Ubuntu, Kali Linux, etc.).

Numpy: It is a package available in the Python 3 language, including helpful operational features for gigantic, high-dimensional matrices and arrays, alongside an enormous collection of high level numerical functions to operate on them.

Pandas: It is a python package composed for data investigation, manipulation and doing several operations on them. Specifically, it provides with data structures and methods for managing numerical tables and time distributed data formats.

Matplotlib: It is a plotting package (to create graphs, charts and figures) for displaying and creating data visual representations in Python 3 language.

Keras: It is a neural networks API, coded in Python and competent in running on top of backends like TensorFlow, Microsoft Cognitive Toolkit, or Theano. It was developed with an objective of allowing fast experimentation and implementation of neural networks.

Scikit-learn: It is an ML package for the Python 3 developed by Google. Its functions include various grouping, regression, clustering and dimensionality reduction algorithms.

4.3 Overall Workflow

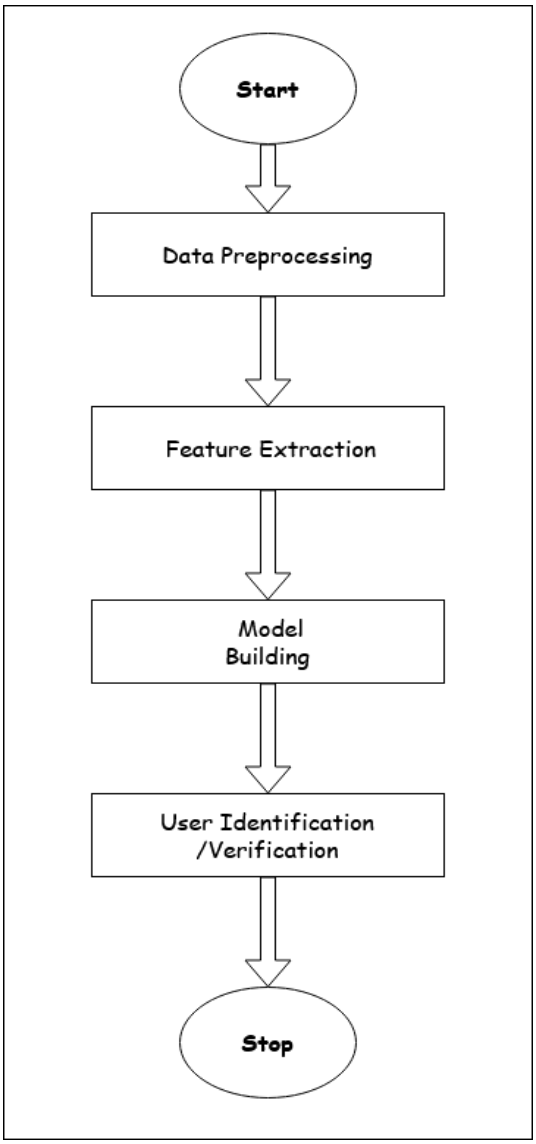

Figure 4.1: Flowchart of the Work

4.4 Dataset Description

The required dataset was taken from the Department of Computer Science, Indian Institute of Techonology (IIT) Roorkee. It had thirty subjects- in the age-group of 15-55 years from IIT Roorkee. Signatures and their corresponding EEG were collected 10 times from each subject under different conditions for the environment. 10 impersonating signs and EEG were collected from each subject. Total number of signs collected in total for the dataset was 300 from 30 different subjects under varying conditions .

The EEG dataset was obtained at the same time as the signatures dataset. It is a 14 channel EEG data which has been measured by using EEG headsets. These specialized headsets have the measuring rods placed at varying distances so that a group of neurons give out the data and not a single neuron. Also, it was made sure that the subject performing the signature was in an active state of mind and not lying or sleeping. The EEG data is then read for the whole time the signature has been performed by the person. Finally, the obtained data has been converted into .txt format with '/t' escape character as the splitter for the different channels. Further, it has been split in 80: 20 ratio for training and testing purposes.

4.5 Dataset Preprocessing

The image data was first read in the model and since the images taken are in the RGB format, they are first converted to the gray scale format. Following it, the EEG data in the .txt files are read by using the Pandas library with '/t' as the splitter.

After reading both the datasets, the image dataset needs to be cropped as per the data in the EEG before combining them to form the matrix input for any of the models. Since the EEG dataset are read as dataframes, they are first converted to arrays. We now have obtained the dataset in a formatted way with the same sizes for both the images as well as the EEG files. The further steps for the preprocessing are performed depending upon the model that is to be used.

4.6 Model 1- CNN for verifying users based on signature images only

The signature images were resized to 224 x 224 pixels (to bring uniformity). Initial weights were set to the weights derived in the Vgg16 model (CNN Model) on ImageNet data. Vgg16 model was trained on train, test, and validation set to generate .npy files (numpy arrays). The vgg model itself consists of several convolution layers and as well as max pooling layers. The structure of the VGG model looks as as shown in the figure 4.2. The weights obtained after applying the dataset to the VGG model are then passed into our model which uses the concept of transfer learning with 1 input layer, 2 hidden layers (50 and 30 nodes), dropout regularization (0.15 in both the layers) and 1 output layer (30 nodes). Leaky ReLU function was used in the inner layers and softmax was used in the output layer. Optimiser used was rmsprop, number of iterations was set to 50, loss function was categorical cross-entropy to classify the categorical data into different classes and batch size of 4 was taken for training.

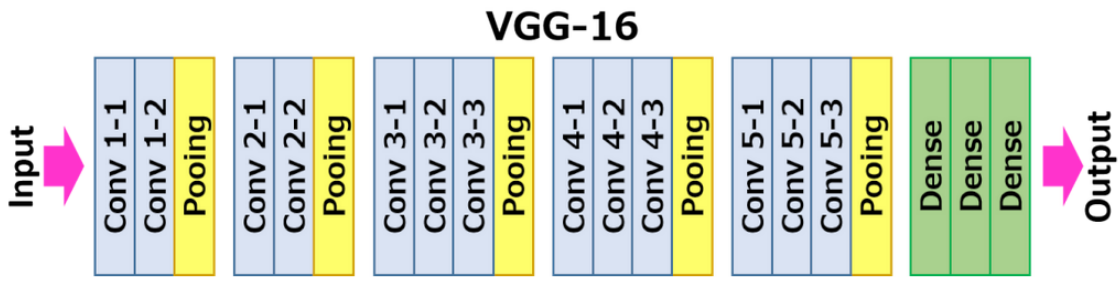

Figure 4.2: Structure of VGG model

```
In [18]: model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten_3 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_24 (Dense)             (None, 50)                1254450
_________________________________________________________________
dropout_13 (Dropout)         (None, 50)                0
_________________________________________________________________
dense_25 (Dense)             (None, 30)                1530
_________________________________________________________________
dropout_14 (Dropout)         (None, 30)                0
_________________________________________________________________
dense_26 (Dense)             (None, 30)                930
=================================================================
Total params: 1,256,910
Trainable params: 1,256,910
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.3: Summary of CNN (Model 1)

Results

Train Accuracy= 96.67%

Train Loss=0.08

Val Accuracy= 96 %

Val Loss= 0.18

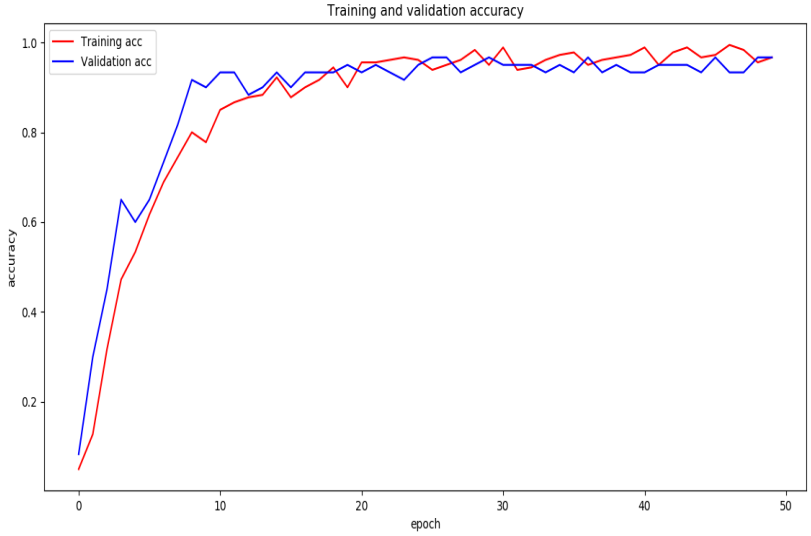

Figure 4.4: Train and Val Set Accuracy Plots of CNN (Model 1)

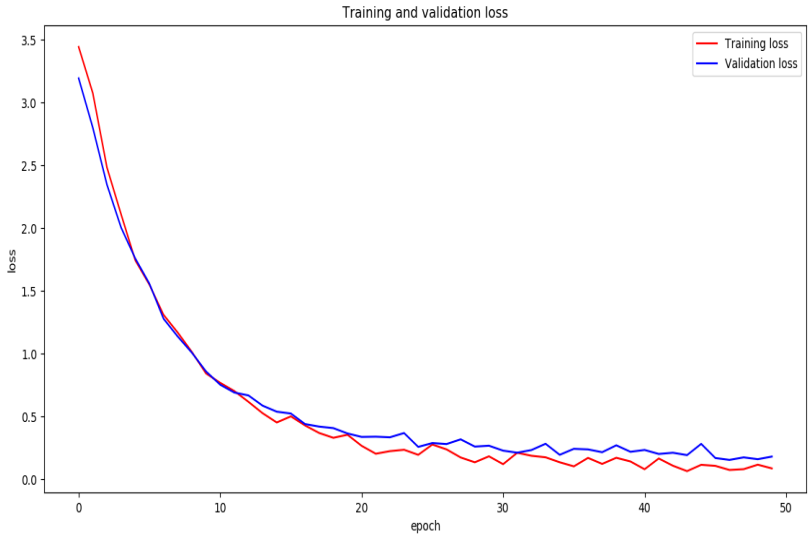

Figure 4.5: Train and Val Set Loss Plots of CNN (Model 1)

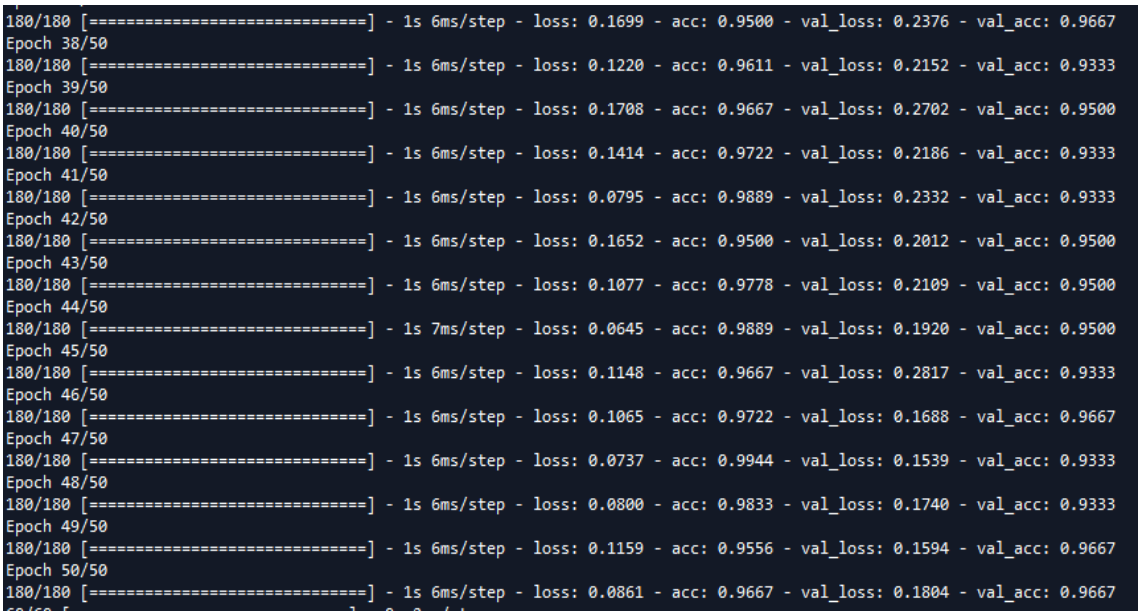

Figure 4.6: Running of the CNN (Model 1)

4.7 Model 2- ANN model for verifying users by using EEG signal data

The data received from the preprocessing step is processed by a PCA with $n\,components$ = 10. And the obtained output is combined for all the EEG datasets/dataframes. The dataset was divided into training and testing sets with 8 EEG data as the training and 2 EEG data for testing per person.

The dataframe obtained was then passed through the neural network with input dense layer of 256 neurons, and an input shape of (,10) signifying that there will be 10 columns in the input that is received. 4 hidden layers consisting of 128, 64, 50 and 50 nodes respectively are then added to the model. All the dense layers are fully connected and using the data they have received and the activation functions the dense layer gives an output as per the shapes specified in the figure 4.7.

All input layers have used relu activation function since it was found as the best fit for the classification data. And the output layer used softmax activation function to give the probabilities of the EEG belonging to a particular subject in the dataset. Dropout regularization of 5 % was implemented in all hidden layers to avoid the scenario of model overfitting to the training data. The model was run for 50 epochs with a batch size of 16. Categorical cross-entropy loss function was used as the output are classes with an adam optimiser.

```
In [10]: model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_12 (Dense)             (None, 256)               2816
_________________________________________________________________
dropout_7 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_13 (Dense)             (None, 128)               32896
_________________________________________________________________
dropout_8 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_14 (Dense)             (None, 64)                8256
_________________________________________________________________
dropout_9 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_15 (Dense)             (None, 50)                3250
_________________________________________________________________
dense_16 (Dense)             (None, 50)                2550
_________________________________________________________________
dense_17 (Dense)             (None, 29)                1479
=================================================================
Total params: 51,247
Trainable params: 51,247
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.7: Summary of ANN (Model 2)

Results

Train Time= 18 mins

Train Accuracy= 96.67%

Train Loss=0.08

Val Accuracy= 96 %

Val Loss= 0.18

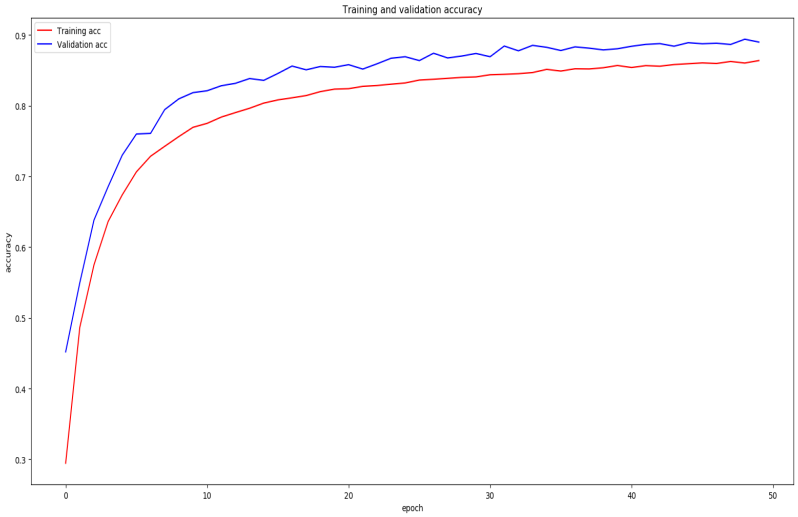

Figure 4.8: Train and Val Set Accuracy Plots of ANN (Model 2)

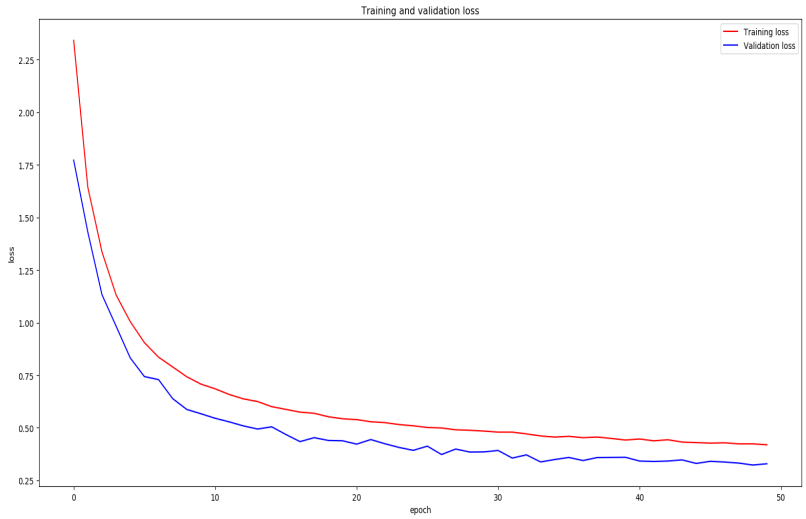

Figure 4.9: Train and Val Set Loss Plots of ANN (Model 2)

```
Epoch 43/50
83984/83984 [==============================] - 22s 258us/step - loss: 0.4430 - acc: 0.8559 -
val_loss: 0.3418 - val_acc: 0.8878
Epoch 44/50
83984/83984 [==============================] - 21s 255us/step - loss: 0.4320 - acc: 0.8582 -
val_loss: 0.3473 - val_acc: 0.8842
Epoch 45/50
83984/83984 [==============================] - 21s 256us/step - loss: 0.4295 - acc: 0.8594 -
val_loss: 0.3305 - val_acc: 0.8891
Epoch 46/50
83984/83984 [==============================] - 21s 252us/step - loss: 0.4270 - acc: 0.8605 -
val_loss: 0.3405 - val_acc: 0.8876
Epoch 47/50
83984/83984 [==============================] - 21s 253us/step - loss: 0.4285 - acc: 0.8597 -
val_loss: 0.3371 - val_acc: 0.8883
Epoch 48/50
83984/83984 [==============================] - 22s 261us/step - loss: 0.4235 - acc: 0.8625 -
val_loss: 0.3321 - val_acc: 0.8866
Epoch 49/50
83984/83984 [==============================] - 22s 258us/step - loss: 0.4235 - acc: 0.8604 -
val_loss: 0.3228 - val_acc: 0.8940
Epoch 50/50
83984/83984 [==============================] - 22s 258us/step - loss: 0.4190 - acc: 0.8638 -
val_loss: 0.3289 - val_acc: 0.8899
```

Figure 4.10: Running of the ANN (Model 2)

4.8 Model 3- ANN model for verifying users by using EEG and signature images

All the images files were read, converted from 3D to gray scale and then appended (for every individual). Then, all the EEG files were read and appended. After this, signature image dataframe was concatenated with its corresponding EEG dataframe. Extra rows were removed so as to bring uniformity in the input shape.

PCA was applied on this transformed dataframe to extract the top 80 features. Previously tested for 120, 100, 70 and 60 components, but the model had poor accuracy. This was then fed into an ANN with 4 hidden layers having 128, 64, 50, 50 fully connected nodes in each layer respectively. All the inner layers used relu activation function since the dataset was a categorical one.

Since an ANN model is being used, the classes are first converted into the categories, which gives an output similar to the one hot encoding format. Softmax function was used in the output layer. To avoid overfitting and in order to obtain the best results, the model uses the Early stopping method from the Tensorflow library with delta value of 10^{-2} and patience of 30 epochs with the monitoring parameter as accuracy of the model. Which means that if the model accuracy does not increase in a range greater than 10^{-2} for more than 30 epochs the model stops the training process. The early stopping criteria restores the model to the best weights. The model is run for 250 epochs, with loss function being categorical cross-entropy for the output being categorical data and optimiser being adam which has an adaptive learning rate. The training stops at epoch number 189 with an accuracy of 86.2%.

```
In [12]: model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_18 (Dense)             (None, 256)               20480
_________________________________________________________________
dropout_10 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_19 (Dense)             (None, 128)               32896
_________________________________________________________________
dropout_11 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_20 (Dense)             (None, 64)                8256
_________________________________________________________________
dropout_12 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_21 (Dense)             (None, 50)                3250
_________________________________________________________________
dense_22 (Dense)             (None, 50)                2550
_________________________________________________________________
dense_23 (Dense)             (None, 30)                1530
=================================================================
Total params: 68,962
Trainable params: 68,962
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.11: Summary of ANN (Model 3)

Results

Train Time= 22 mins

Train Accuracy = 86.2 %

Train Loss= 0.48

Val Accuracy= 84.1

Val Loss= 0.57

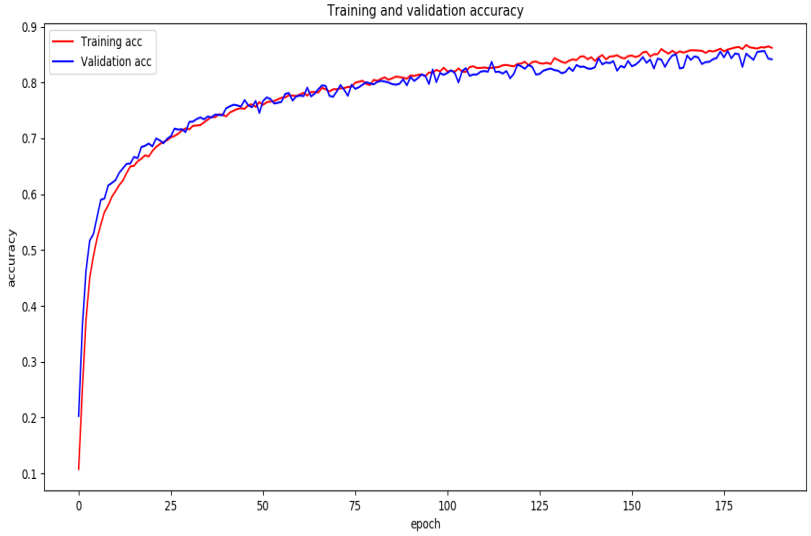

Figure 4.12: Train and Val Set Accuracy Plots of ANN (Model 3)

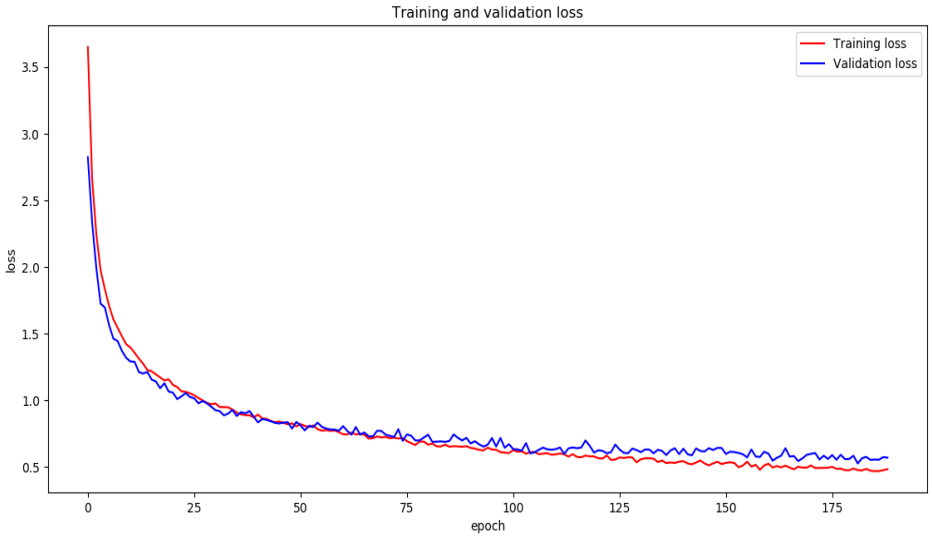

Figure 4.13: Train and Val Set Loss Plots of ANN (Model 3)

```
15796/15796 [==============================] - 7s 442us/step - loss: 0.5025 - acc: 0.8557 - val_loss: 0.5909 - val_acc: 0.8451
Epoch 177/250
15796/15796 [==============================] - 7s 450us/step - loss: 0.4874 - acc: 0.8591 - val_loss: 0.5555 - val_acc: 0.8565
Epoch 178/250
15796/15796 [==============================] - 7s 428us/step - loss: 0.4887 - acc: 0.8609 - val_loss: 0.5930 - val_acc: 0.8434
Epoch 179/250
15796/15796 [==============================] - 7s 424us/step - loss: 0.4778 - acc: 0.8628 - val_loss: 0.5602 - val_acc: 0.8519
Epoch 180/250
15796/15796 [==============================] - 7s 421us/step - loss: 0.4772 - acc: 0.8637 - val_loss: 0.5615 - val_acc: 0.8508
Epoch 181/250
15796/15796 [==============================] - 7s 425us/step - loss: 0.4892 - acc: 0.8599 - val_loss: 0.5858 - val_acc: 0.8280
Epoch 182/250
15796/15796 [==============================] - 7s 426us/step - loss: 0.4782 - acc: 0.8674 - val_loss: 0.5276 - val_acc: 0.8519
Epoch 183/250
15796/15796 [==============================] - 7s 422us/step - loss: 0.4759 - acc: 0.8631 - val_loss: 0.5670 - val_acc: 0.8462
Epoch 184/250
15796/15796 [==============================] - 7s 452us/step - loss: 0.4872 - acc: 0.8619 - val_loss: 0.5770 - val_acc: 0.8405
Epoch 185/250
15796/15796 [==============================] - 7s 433us/step - loss: 0.4727 - acc: 0.8609 - val_loss: 0.5548 - val_acc: 0.8548
Epoch 186/250
15796/15796 [==============================] - 7s 427us/step - loss: 0.4707 - acc: 0.8633 - val_loss: 0.5570 - val_acc: 0.8559
Epoch 187/250
15796/15796 [==============================] - 7s 429us/step - loss: 0.4703 - acc: 0.8628 - val_loss: 0.5556 - val_acc: 0.8565
Epoch 188/250
15796/15796 [==============================] - 7s 427us/step - loss: 0.4783 - acc: 0.8648 - val_loss: 0.5756 - val_acc: 0.8428
Epoch 189/250
15796/15796 [==============================] - 7s 421us/step - loss: 0.4845 - acc: 0.8622 - val_loss: 0.5710 - val_acc: 0.8417
Epoch 00189: early stopping
```

Figure 4.14: Running of the ANN (Model 3)

4.9 Model 4- CNN model for verifying users using the EEG and signatures

All the images were read and converted to a gray scaled format and concatenated with the EEG dataframes with properly sized columns and cropping of the images to maintain the size uniformity.

The model uses a Convolution 2D function, hence, needs the data to be a 4 dimensional matrix. This is achieved by introducing the dummy dimensions using the 'expand dims' function which adds the dimensions to the end part of the dataframes. It is then rotated around to set the matrix as the number of features being in the last dimension and the number of rows being the first dimension.

The first convolution layer of the model consists of 100 input neurons with shape as [1,1,270], followed by max pooling with 100 neurons. The model is then passed through another layer of convolution with 100 neurons, followed by max pooling with 50 neurons. The model uses 4 hidden layers, with 128, 64, 50 and 64 fully connected nodes in the order as shown in the figure 4.14.

All the layers implement relu activation functions found to be the best fit for the categorical data. Dropout of 10% was added in one layer and 5% in the subsequent 2 layers. The output layer consists of 30 nodes corresponding to the 30 output classes with softmax

activation function for giving the probabilities of the output belonging to a particular class. The loss function being used is categorical cross-entropy and optimiser is adam. The model was run for 500 epochs with delta value of 10^{-3} and patience of 50 neurons. The training stops at 239 epochs with an accuracy of 94%.

```
In [4]: model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 100, 1, 266)       600
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 100, 1, 133)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 50, 1, 131)        15050
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 50, 1, 65)         0
_________________________________________________________________
dropout_1 (Dropout)          (None, 50, 1, 65)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 3250)              0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               832256
_________________________________________________________________
dropout_2 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_3 (Dense)              (None, 64)                8256
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 50)                3250
_________________________________________________________________
dense_5 (Dense)              (None, 64)                3264
_________________________________________________________________
dense_6 (Dense)              (None, 30)                1950
=================================================================
Total params: 897,522
Trainable params: 897,522
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.15: Summary of CNN (Model 4)

Results

Train Time= 90 mins

Train Accuracy = 94 %

Train Loss= 0.23

Val Accuracy= 92 %

Val Loss= 0.39

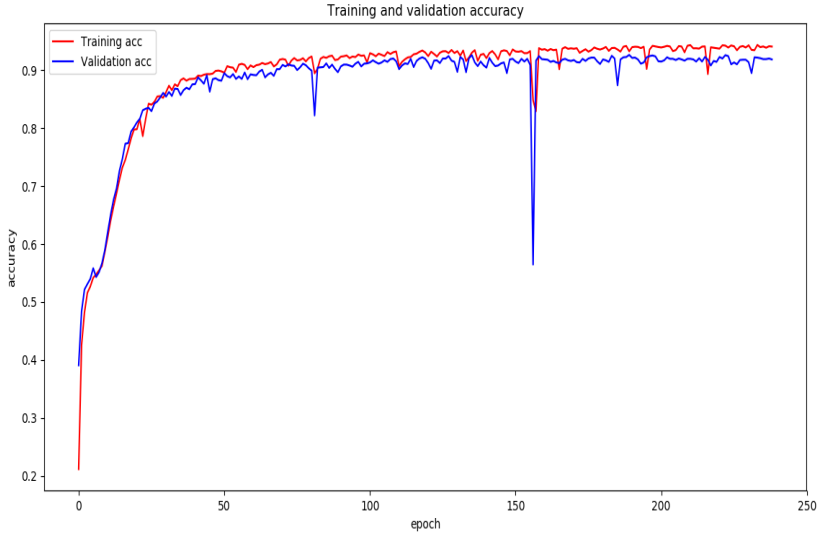

Figure 4.16: Train and Val Set Accuracy Plots of CNN (Model 4)

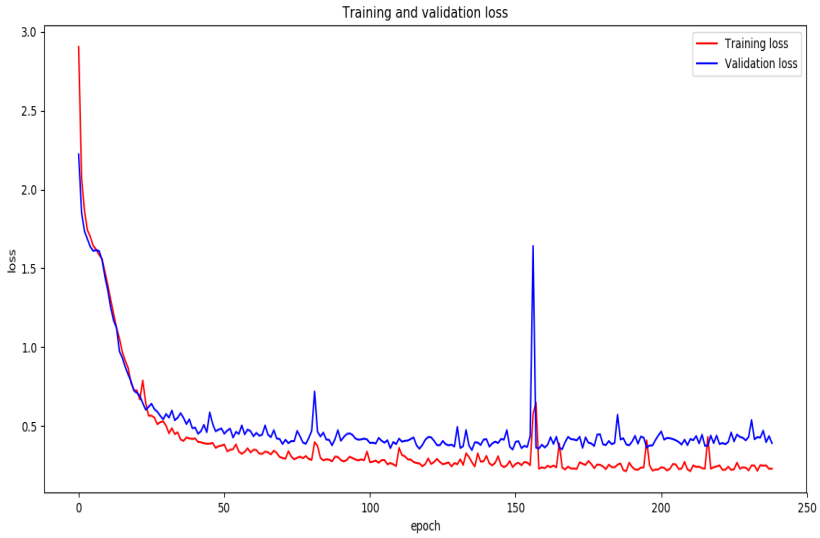

Figure 4.17: Train and Val Set Loss Plots of CNN (Model 4)

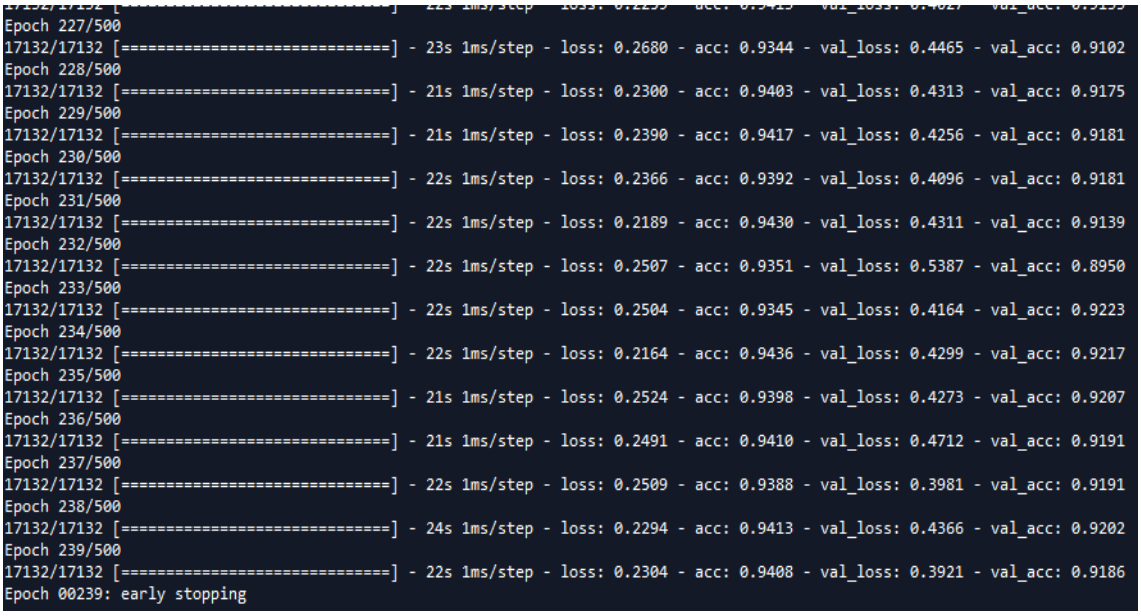

Figure 4.18: Running of the CNN (Model 4)

4.10 Model 5- CNN (with PCA) model for verifying users using the EEG and signatures

All the images were read and converted to a gray scale format and concatenated with the EEG dataframes after performing a PCA on them with $n\,components = 80$. It hs been tested for 120, 100, 70 and 60 components also, but the model yielded poor accuracy and very high loss. This obtained dataframe was passed into the Conv NN with input layers having 128 nodes and 5 filters. The output is passed into a max pooling layer to find the feature which gives the maximum among all the filters. This process was again repeated for a second convolution layer and then a max pooling operation was performed. The output of the max pooling layer is passed into the flatten layer to convert it to a single dimensional matrix which can be computed by the 3 hidden layers having 50, 30 and 25 nodes fully connected neurons respectively.

Relu activation function has been used in input and the hidden layers and softmax function in the output layer. All the hidden layers have 20 % dropout regularisation to avoid any form of model overfitting. The model was run with categorical cross-entropy loss function, adam optimiser and for 200 epochs with early stopping.

```
In [7]: model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 100, 1, 75)        600
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 100, 1, 37)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 50, 1, 35)         15050
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 50, 1, 17)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 50, 1, 17)         0
_________________________________________________________________
flatten_2 (Flatten)          (None, 850)               0
_________________________________________________________________
dense_7 (Dense)              (None, 128)               108928
_________________________________________________________________
dropout_5 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_8 (Dense)              (None, 50)                6450
_________________________________________________________________
dense_9 (Dense)              (None, 30)                1530
_________________________________________________________________
dense_10 (Dense)             (None, 25)                775
_________________________________________________________________
dropout_6 (Dropout)          (None, 25)                0
_________________________________________________________________
dense_11 (Dense)             (None, 30)                780
=================================================================
Total params: 134,113
Trainable params: 134,113
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.19: Summary of CNN (Model 5)

Results

Train Time= 40 mins

Train Accuracy = 80.9 %

Train Loss= 0.61

Val Accuracy= 82.9 %

Val Loss= 0.54

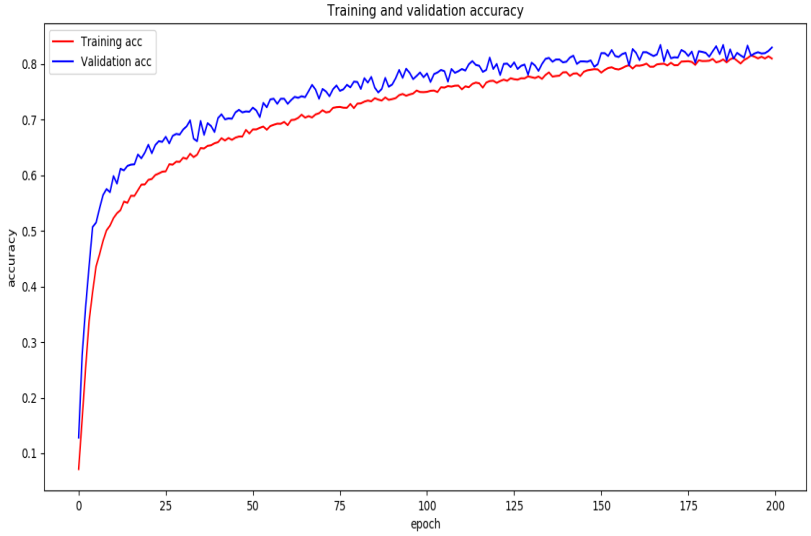

Figure 4.20: Train and Val Set Accuracy Plots of CNN (Model 5)

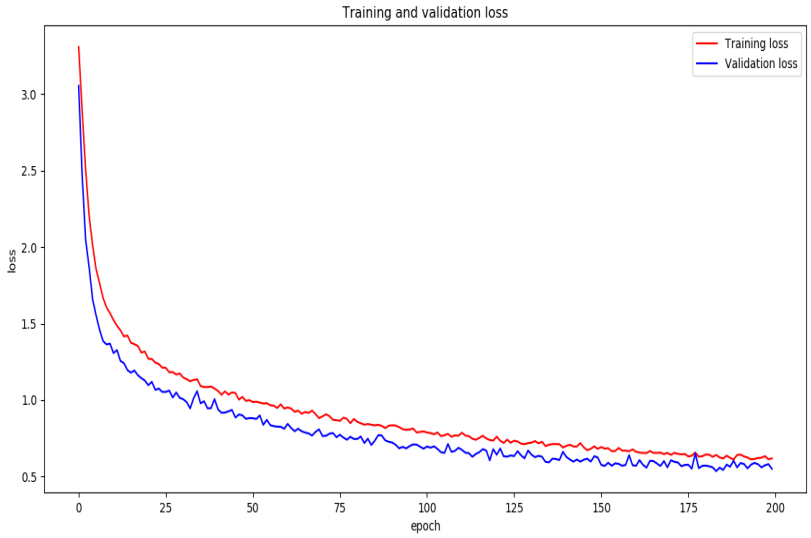

Figure 4.21: Train and Val Set Loss Plots of CNN (Model 5)

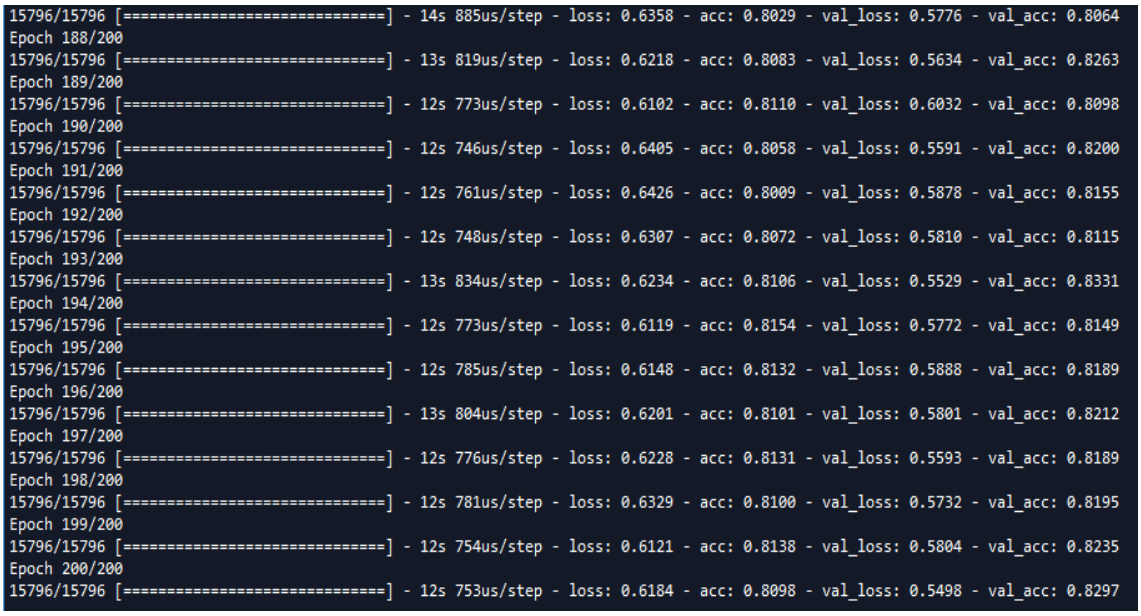

Figure 4.22: Running of the CNN (Model 5)

4.11 Model 6- Ensemble model (PCA+RF+Bagging) for verifying users with EEG and signature images

All the classes were numbered from 0 to 29, the dataset was normalized to bring down the variables in the same range. Following this, PCA ($n\,components$ = 10) was performed on the concatenated dataframes of the images and EEGs. Due to hardware limitations, $n\,components$ = 20 or more could not be executed by the Bagging model. With this new dataframe, model was trained using the Random Forest classifier with the criteria of splitting being gini index and the number of estimators as 30 with each esimator having a maximum depth of . For boosting the output of the dataset in order to achieve higher accuracy, a Bagging classifier was fit to the Random Forest model. 300 estimators were used in the bootstrap aggregator (Bagging), to obtain the following results on the dataset:

Results

Train Time= 20 mins

Accuracy = 96.6 %

Precision= 96.72 %

Recall= 95.7 %

F1-score= 96.1 %

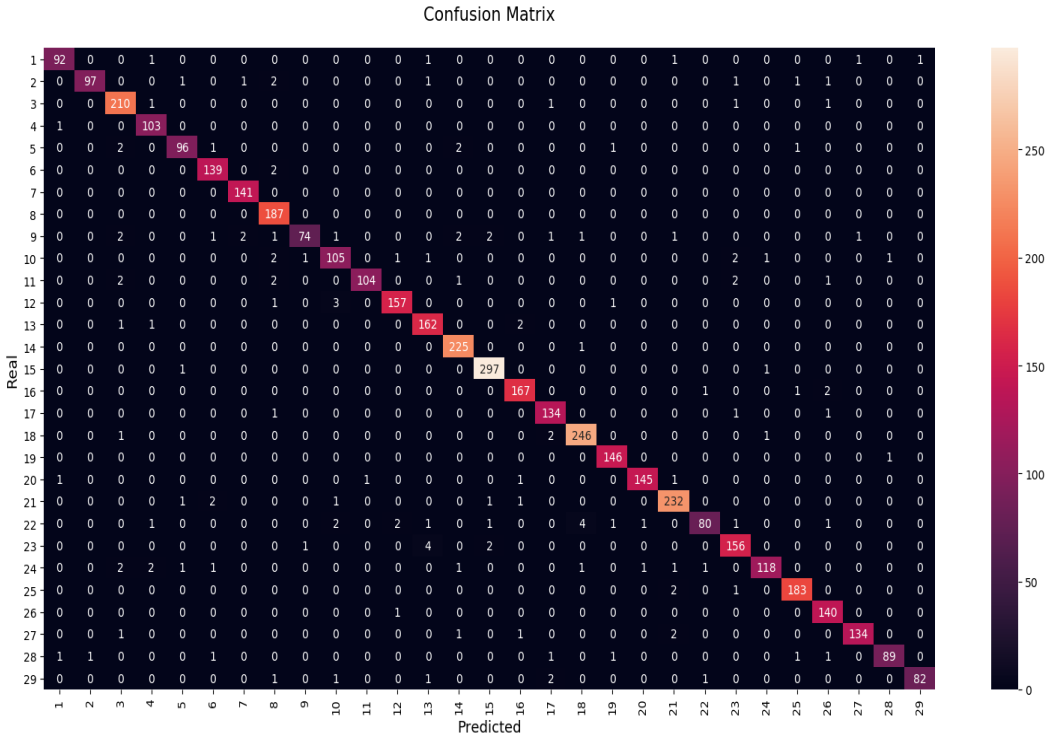

Figure 4.23: Confusion Matrix for Ensemble Technique (Model 6)

4.12 Model 7- Ensemble model (PCA+RF+AdaBoosting) for verifying users with EEG and signature images

All the classes were numbered from 0 to 29, the dataset was normalized to get all the variables in the same range and reduce the training time. Following this, PCA ($n\,components$ = 20) was performed on the concatenated dataframes of the images and EEGs to get the top 20 best features and also to reduce the overall dimension for the faster processing. For $n\,components$ < 20, the model yielded lesser accuracy and with $n\,components$ > 20, hardware limitations were encountered. With this obtained dataframe, model was trained using the Random Forest classifier with the criteria of splitting being entropy, the max depth in the trees have been set to None, hence the model will expand till the leaves are perfectly clear on the prediction of the sample and the number of estimators as 500.

Apart from these compulsory parameters, the optional parameters of bootstrapping was set at False so that the tree is not built from the data duplicates and uses the whole training set for the model building process. The random state parameter has been set at 42 so that the same results get reproduced every time the model is run. The model automatically sets the n outputs parameter as 30 since the 30 classes needs to be predicted. Base estimator

used for the Random Forest model is a Decision Tree Classifier. The Random Forest Classifier alone gives an accuracy of the 48%.

For boosting the output of the datasets in order to achieve higher performance, an Ada Boost classifier was fit to the Random Forest models (base estimators). The algorithm used for the Ada Boost was 'SAMME.R'. The 'SAMME.R' is an algorithm which gives us a probability of the output for belonging to a particular subject. Combined with the Classifier, this helps in increasing model performance. The learning rate was set to the default value of 1 and the total number of estimators were set to 10. The random state was kept at 42 which gives the same reproducible results at all times. While fitting the model the intial sample weights has been set to None, hence the model starts training from the average value of the weights. The ensemble technique gave the following results:

Results

Train Time= 15 mins

Accuracy = 98.76 %

Precision= 98.88 %

Recall= 98.32 %

F1-score= 98.55 %

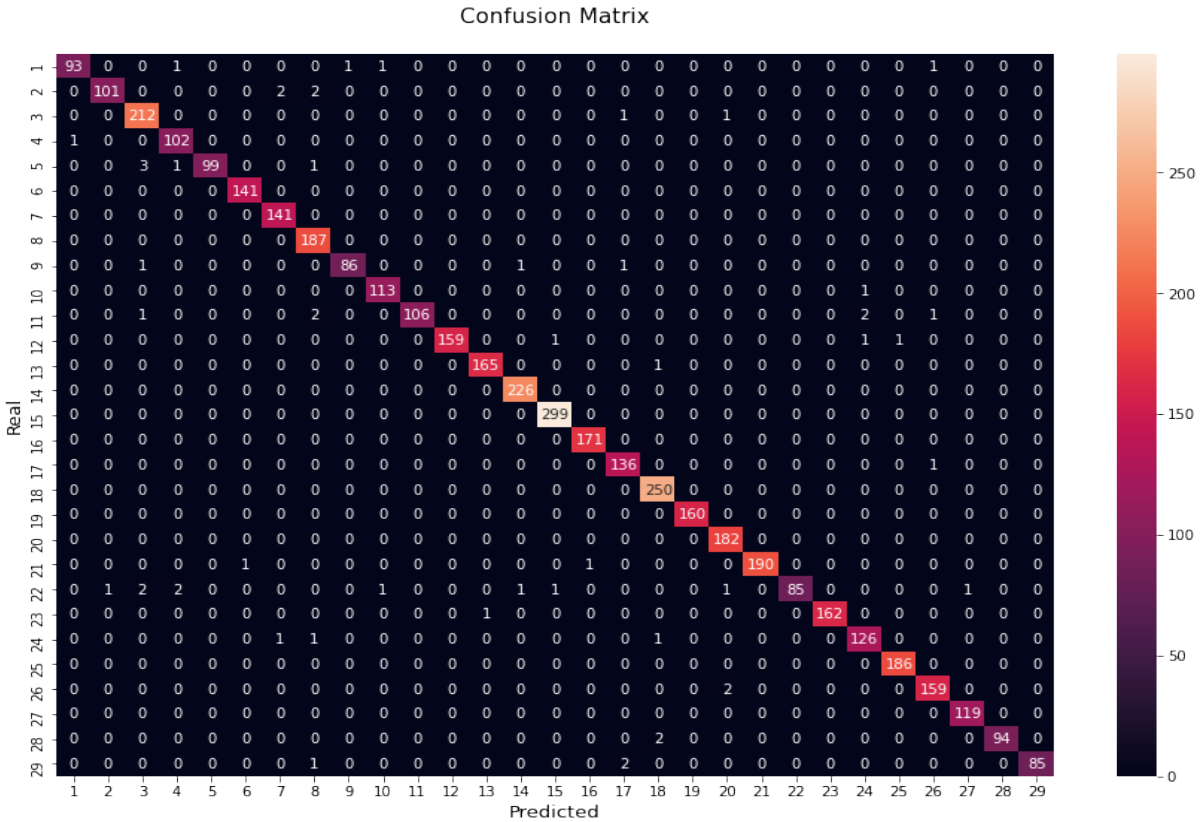

Figure 4.24: Confusion Matrix for Ensemble Technique (Model 7)

Chapter 5

Analysis and Discussion

5.1 Neural Networks Configurations

After the implementation of all the models, the optimal deep neural network layouts are:

Table 5.1: Optimal Neural Network Configurations

Model No.	Type	No. of input nodes	No. of hidden layers	Nodes in hidden layers
1	CNN	30	2	50, 30
2	ANN + PCA	256	4	128, 64, 50, 50
3	ANN	256	4	128, 64, 50 , 50
4	CNN	256	4	128, 64, 50, 64
5	CNN + PCA	128	3	50, 30, 25

5.2 Accuracy and Loss metrics of Models

The final results obtained are summarized as below:

Table 5.2: Results Obtained for Each Method

Model No.	Data Used	Model	Accuracy (%)	Loss	Train Time (mins)
1	Signature	CNN	96.6	0.08	5
2	EEG	ANN (with PCA)	86.3	0.41	18
3	Signature and EEG	ANN	86.2	0.48	22
4	Signature and EEG	CNN	94.2	0.39	90
5	Signature and EEG	CNN (with PCA)	80.9	0.61	40
6	Signature and EEG	Ensemble (Bagging)	96.6	N.A	20
7	Signature and EEG	Ensemble (AdaBoost)	98.8	N.A	15

5.3 Analysis - Pros, cons and Novelty of Models

After analysing the results obtained in various models, we can conclude that Convolutional Neural Network (Model 4) and Ensemble Technique (Model 7) are working the best in person recognition by extracting and making use of the features of both signature images and the corresponding EEG signals. In all these models, the difference between the training accuracy and the validation accuracy is very less 2%, which implies that the model is learning properly, and not simply over-fitting the data. The loss is also significantly low in case of both training as well as validating. The accuracy obtained is high which is a sign of good performance.

In case of deep learning models, CNN is performing much better than ANN in terms of accuracy. However, the model size and training time complexity is too large. From the model outputs, we can also see that model performance in most cases, increases by using feature extraction by PCA. However, it has been observed that when the PCA and the Convolutional networks are combined, they give comparatively poorer results, thus signifying that the CNN model not only takes the best features but as well as the best distinguishable features from the data fed into it. Thus, CNN model has a more sophisticated feature extracting technique.

While using Machine learning based models, the performance is significantly low if we use simple, standalone base estimators such as Decision Trees, Random Forest or SVM. However, the performance improves drastically by implementing feature extraction techniques like PCA and by using ensemble algorithms such as AdaBoost, XGBoost or Bagging. The main advantage of these ensemble techniques based models is that their complexity, size and training time is much lesser than similar performing deep neural network models. So, with the given dataset, it is more convenient to use Machine learning based ensemble models. Deep neural networks can be highly beneficial while using larger datasets.

Chapter 6

Conclusion

Machine Learning and Deep Neural Network models for person identification and verification were built. The CNN model for identifying persons by using only signature images had an accuracy of 96%. ANN model using only EEG data had an accuracy of 86%. The multimodal approach by combining the features of both signature and EEG data had an accuracy of 94.2% (with CNN) and 98.8% (with Ensemble method). All of these models are giving good performance indicating the huge potential of EEG and signature as user identification and verification methods. Since the CNN Model-4 and Ensemble Model-6 are giving similar results, the CNN Model (Deep Neural Networks based model which is complex in nature and takes more amount of time) can be used for larger datasets. The Ensemble Method (PCA, Random Forest and Boosting - it has comparatively simpler architecture and hence, has a low training time) can be efficiently used on smaller datasets.

Scope for Further Research

The above implemented models have to be hypertuned to increase the efficiency, robustness and performance. These models have to be trained on larger datasets and tested for more number of people. If successful, these can be implemented in real-life applications, thus giving birth to more reliable, secure and efficient security systems.

References

[1] M. R. Boubakeur, G. Wang, C. Zhang, and K. Liu, "Eeg-based person recognition analysis and criticism," in *2017 IEEE International Conference on Big Knowledge (ICBK)*. IEEE, 2017, pp. 155–160.

[2] H. Xu and K. N. Plataniotis, "Affect recognition using eeg signal," in *2012 IEEE 14th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 2012, pp. 299–304.

[3] H. A. Shedeed, "A new method for person identification in a biometric security system based on brain eeg signal processing," in *2011 World Congress on Information and Communication Technologies*. IEEE, 2011, pp. 1205–1210.

[4] H. A. Hadi and L. E. George, "Eeg based user identification and verification using the energy of sliced dft spectra," *International Journal of Science and Research (IJSR)*, vol. 6, no. 9, pp. 46–51, 2017.

[5] T. Wilaiprasitporn, A. Ditthapron, K. Matchaparn, T. Tongbuasirilai, N. Banluesombatkul, and E. Chuangsuwanich, "Affective eeg-based person identification using the deep learning approach," *IEEE Transactions on Cognitive and Developmental Systems*, 2019.

[6] X. Huang, S. Altahat, D. Tran, and D. Sharma, "Human identification with electroencephalogram (eeg) signal processing," in *2012 International symposium on communications and information technologies (ISCIT)*. IEEE, 2012, pp. 1021–1026.

[7] R. Saini, B. Kaur, P. Singh, P. Kumar, P. P. Roy, B. Raman, and D. Singh, "Don't just sign use brain too: A novel multimodal approach for user identification and verification," *Information Sciences*, vol. 430, pp. 163–178, 2018.

[8] P. Campisi, D. La Rocca, and G. Scarano, "Eeg for automatic person recognition," *Computer*, vol. 45, no. 7, pp. 87–89, 2012.