ATM Security system based on Face recognition, PIN and OTP

PROJECT REPORT

Submitted in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

Mohammed Aslam P S (MDL15EC077)

Sandra Maria Roy (MDL15EC103)

Sasna Sunder (MDL15EC104)

Suhaina J (MDL15EC116)

DEPARTMENT OF ELECTRONICS

GOVT. MODEL ENGINEERING COLLEGE

THRIKKAKARA, COCHIN - 682 021

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

MAY 2019

Bonafide certificate

MODEL ENGINEERING COLLEGE

THRIKKAKARA, KOCHI-21

DEPARTMENT OF ELECTRONICS

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

This is to certify that the Project Report entitled

Submitted by

Mohammed Aslam P S (MDL15EC077)

Sandra Maria Roy (MDL15EC103)

Sasna Sunder (MDL15EC104)

Suhaina J (MDL15EC116)

is a bonafide account of their work done under our supervision

Project Co-ordinator
Dr. Binesh T
Associate Professor
Dept. of Electronics

Project Guide
Mr. Ajay Nath S A
Assistant Professor
Dept. of Electronics

Head of the Department
Dr. Laila D
Professor
Dept. of Electronics

Acknowledgment

First of all , we are thankful to the Lord, Almighty whose enormous grace and blessings enabled us to complete the Project. We express our sincere gratitude to **(Prof.) Dr. Vinu Thomas, Principal, Govt. Model Engineering College** for opening up a plethora of possibilities and resources for our use.

Next we express our heartfelt thanks to **(Prof.) Dr. Laila D, Head of Department of Electronics Engineering**. She has offered infallible support and timely advice and was pivotal in the completion of this Project. We are greatly obliged to the Project coordinator **Dr. Binesh T , Associate Professor, Department of Electronics** and **Smt.Thushara H P, Assistant Professor, Department of Electronics** for allowing us to use the facilities available and for providing us with right guidance and advice at the crucial junctures and for showing us the right way. We are thankful to our Seminar guide **Mr. Ajay Nath S A, Assistant Professor of Department of Electronics Engineering** for his support and advice he has given us throughout the project.

We also extend our sincere gratitude to our teachers, non-teaching staff and classmates who were ready with a positive comment all the time.

Abstract

In order to provide reliable security solution to the people, the Project is focused on Design and Implementation of Face Recognition based ATM Security System using OpenCV,machine learning and deep learning. The system is implemented on the credit card size Raspberry Pi board with extended capability of open source Computer Vision (OpenCV) software which is used for Image processing operation. High level security mechanism is provided by the consecutive actions such as initially system captures the human face and check whether the face/user is an already registered user with details in the database. If the face is not recognized then it notifies the user that he has no bank account. If face is recognized, then the user is asked to enter the unique Personal Identification Number(PIN). If PIN entered by user is wrong , then the account holder can't proceed further. If PIN is correct, user has to enter One Time Password (OTP) that is sent to the registered mobile number.If OTP entered by user is wrong even after 3 attempts, then the request cannot be processed. Once the OTP entered is right, the user can proceed to further transactions.

Keywords- Face recognition, Raspberry Pi, OpenCV , Machine learning, Deep learning, PIN, OTP.

Table of Contents

List of Figures

List of Abbreviations

API - Application Program Interface

ARM - Advanced RISC Machine

ATM - Automated Teller Machine

CPU - Central Processing Unit

CSI - Camera Serial Interface

DC - Direct Current

FDA - Fisher Discriminant Analysis

GB - Giga Bits

GPIO - General Purpose Input Output

GSM - Global System for Mobile

GUI - Graphical User Interface

HDMI - High-Definition Multimedia Interface

IDE - Integrated Development Environment

JPEG - Joint Photographic Experts Group

LAN - Local Area Network

LCD - Liquid Crystal Display

LFDA - Local Fisher Discriminant Analysis

LXDE - Lightweight X11 Desktop Environment

ML - Machine Learning

MLL - Machine Learning Library

OpenCV - Open source Computer Vision

OS - Operating System

OTP - One-Time Password

PCA - Principal Component Analysis

PIN - Personal Identification Number

PIXEL - Pi Improved Xwindows Environment, Lightweight

RAM - Random Access Memory

SD - Secure Digital

SIFT - Scale-Invariant Feature Transform

SMS - Short Message Service

SQL - Standardized Query Language

USB - Universal Serial Bus

Chapter 1

Introduction

An Automatic Teller Machine (ATM) is a computerized machine that is used to withdraw cash from a customer's respective bank account. As financial users prefer ATM for cash withdrawals, cash deposits and many other transaction, the banks are focusing a lot over the security of ATMs.Hence ATM should be protected properly from the criminal activities or from any unwanted things.

Due to rapid development in science and technology, upcoming innovations are being built-up with strong security. But on the other hand, threats are also being posed to destroy this security level. Though enhancement in automation has made a positive impact overall, various financial institutions like banks and applications like ATM are still subjected to thefts and frauds. The existing ATM model uses a card and a PIN which gives rise to increase in attacks in the form of stolen cards, or due to statically assigned PINs, duplicity of cards and various other threats.Then another major problem is hacking of PIN. There are other

fraudulent attacks like eavesdropping, spoofing , brute force attacks, blackmailing the user. In the worst case there can also be ATM machine Robbery.

To overcome these problems,the project 'ATM Security system based on Face recognition, PIN and OTP' consists of conventional features ie is Personal Identification Number (PIN) along with additional features like face recognition and one-time password (OTP) is used. Database holds information about a user's account details, images of his/her face and a mobile number which will improve security to a large extent.

First, the user will come to the ATM machine and a live image is captured through the Pi Camera interfaced with Raspberry Pi defining as the ATM system, which is compared with the images stored in the database. If the face is recognised, then the user is notified to type the PIN. If the PIN matches, an OTP will be sent to the corresponding registered mobile number. If the user correctly enters the OTP, the transaction can proceed. Therefore, the combination of face recognition algorithm,PIN and an OTP drastically reduces the chances of fraud. Inorder to obtain better accuracy deep learning based linear discrimint classification method is utilized. And excecuted the same in windows OS.

Chapter 2

Literature Survey

Mohsin Karovaliya in paper [2] proposes Eigenface based method for the face recognition. The model shows the qualitative analysis of algorithms used based on the metrics of existing algorithms. According to the statistics, PCA based face recognition is very accurate, requires less computation time and less storage space as trainee images are stored in the form of their projections on a reduced basis.The drawback of using Eigenface based method is that it can sometimes be spoofed by the means of fake masks or photos of an account holder. To overcome this problem 3D face recognition methods can be used. However, its computation cost is high.

The paper [3] suggests a vibration sensor which senses vibrations produced from ATM machine whenever robbery occurs. This system uses ARM controller based embedded system to process real time data collected using the vibration sensor. Once the vibration is sensed the beep sound will occur from the buzzer.

DC Motor is used for closing the door of ATM. Some other additional security measures are used. This will prevent the robbery and the person involving in robbery can be easily caught. Software implementation is deployed using two software packages, first one is the Keil Vision 3.0. Second one is the Flash magic simulator. Keil Vision Debugger accurately simulates on-chip peripherals. This system helps in rapid reaction and minimization of loss by detecting the ATM machine at real-time when it has been stolen can be found through GSM technology.

In paper [4], the finger print recognition is done with curvelet transform by finding the Euclidean distance between the two corresponding finger codes. The test finger code is compared with the entire finger codes in the database. If it matches, an OTP will be send to the corresponding registered mobile number.Preprocessing is done by using built in mathlab function imread. Histogram equalization method helps in improving the global contrast of an image by moving slightly the intensity distribution on a histogram. This permits areas of lower local contrast to gain a higher contrast without affecting the global contrast. Histogram equalization fulfill this by effectively spreading out the most frequent intensity values. The feature extraction can be done using curvelet transform and by using FFT.

The ATM security is enhanced by adding GSM module to generate OTP in paper [5]. When there is a network problem in GSM technology, instead of GSM here implements Bluetooth connection with ATM, Which generates OTP reference through user mobile. GSM modems can be a quick and efficient way to get

started with SMS, because a special subscription to an SMS service provider is not required. A GSM modem could also be a standard GSM mobile phone with the appropriate cable and software driver to connect to a serial port or USB port

The paper [6] suggests fisher faces algorithm for face recognition. Compared to globality based supervised dimensionality reduction methods such as Fisher Discriminant Analysis (FDA), locality based ones including Local Fisher Discriminant Analysis (LFDA) have attracted increasing interests since they aim to preserve the intrinsic data structures and are able to handle multimodally distributed data. However, both FDA and LFDA are usually solved via a ratio trace form to approximate the trace ratio, which is the Fisher's original objective criterion.

In paper, [7] Deep Learning method is introduced with as a part of learning based strategy to provide a complete analysis about the face samples present in the system. The visible general problems in face recognition are fraudulent faces and the factors affecting recognition accuracy such as noise, diversions in the angle, poses and expression. These problems are the main cause for system to lose its perfection, here it improves the accuracy of recognition by keeping the track of history information about the faces arriving as an input. The experimental results acquired on YALE and ORL database shows that this is an efficient method.

One Touch Multi-banking Transaction ATM System using Biometric and GSM Authentication is implemented in paper [8] The account details of user are stored on cloud in a centralized manner. The account details of all the bank accounts of

account holder are displayed. User needs to select one of the bank accounts for the transaction. If user wants to transfer the money or debit the money from account he will get OTP on registered mobile number. The GSM module generates OTP for enforce authenticate transaction from the bank side.

Chapter 3

Design Phase

3.1 Block Diagram

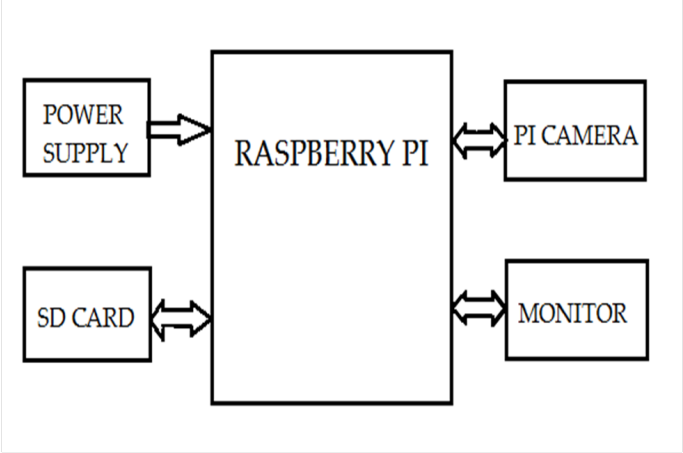
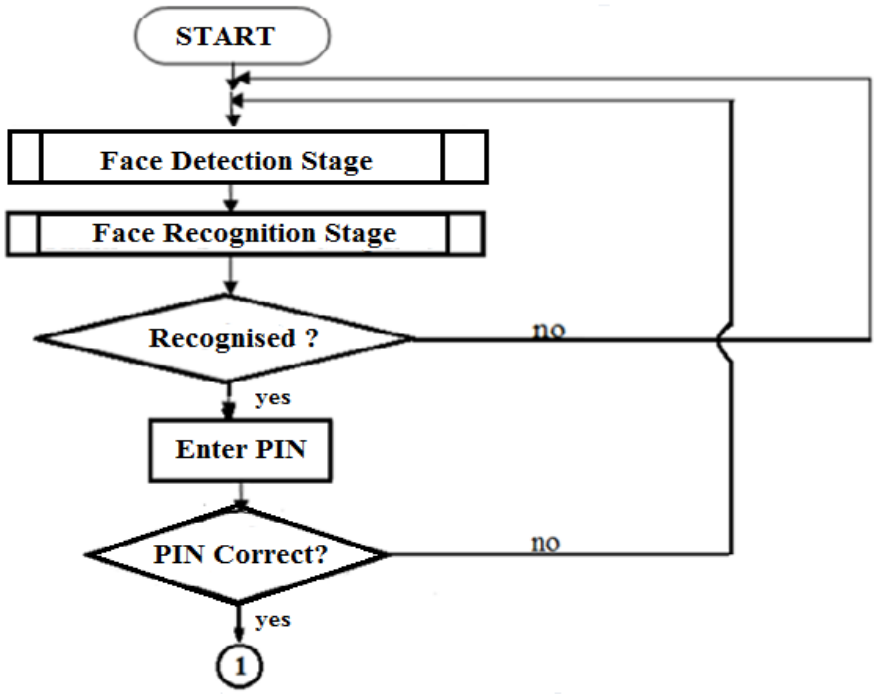


Figure 3.1: Block Diagram[1]

The system contains Raspberry Pi board as the main processor. The SD card is used to contain the dedicated operating system which is compatible with the raspberry pi board. The Human face is captured by the Pi Camera which can be

directly interfaced with the Raspberry Pi board. The monitor displays the messages for user interface.

3.2 System Flow



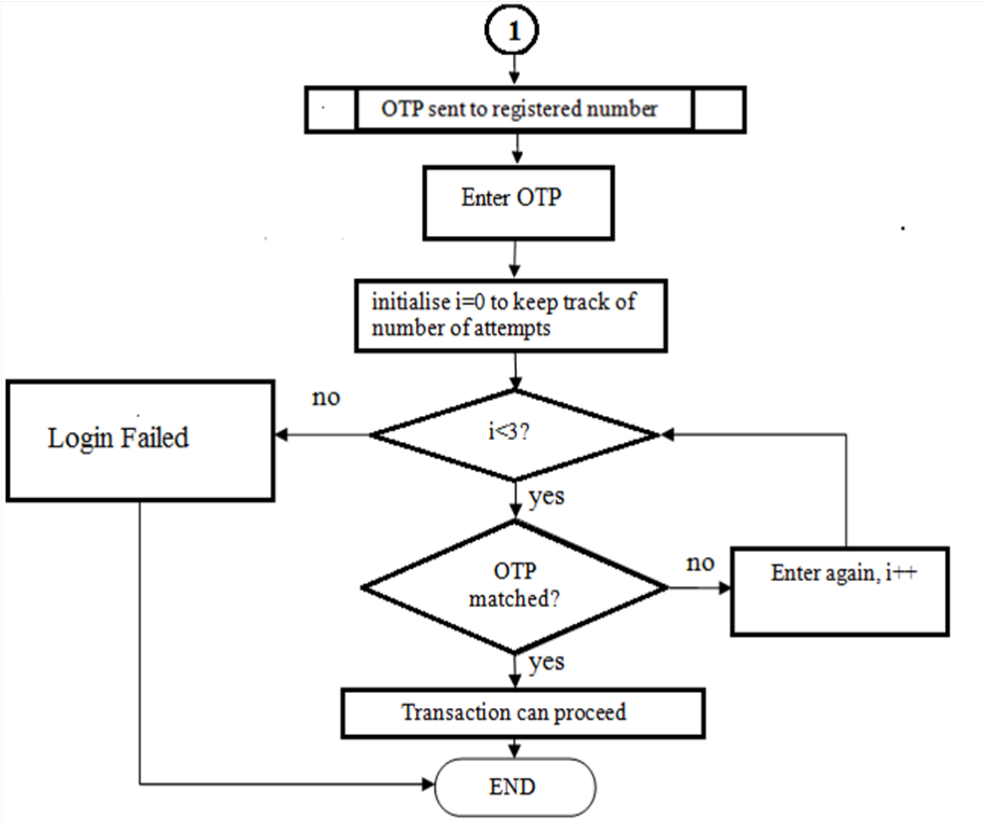


Figure 3.2: Flowchart[1]

3.3 System Overview

3.3.1 Face Detection and Face Recognition

Face Detection

The algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then it is required to extract features from it. Features are nothing but numerical information extracted from the images that can be used to distinguish one image from another; for example, a

histogram (distribution of intensity values) is one of the features that can be used to define several characteristics of an image even without looking at the image, such as dark or bright image, the intensity range of the image, contrast, and so on. Using Haar features is a efficient method for face detection. These features are just like the convolution kernel. The convolution can be summarized by locating a pixel from the image, then crop out a sub-image with the selected pixel as the center from the source image with the same size as the convolution kernel. Calculate an element-wise product between the values of the kernel and sub- image. Add the result of the product. Put the resultant value into the new image at the same place where you picked up the pixel location.

Each feature is a single value obtained by subtracting the sum of the pixels under the white rectangle from the sum of the pixels under the black rectangle. Now, all possible sizes and locations of each kernel are used to calculate plenty of features. Each feature calculation, requires to find the sum of the pixels under the white and black rectangles. The concept of integral image is very useful to solve this. Integral images are those images in which the pixel value at any (x,y) location is the sum of the all pixel values present before the current pixel.

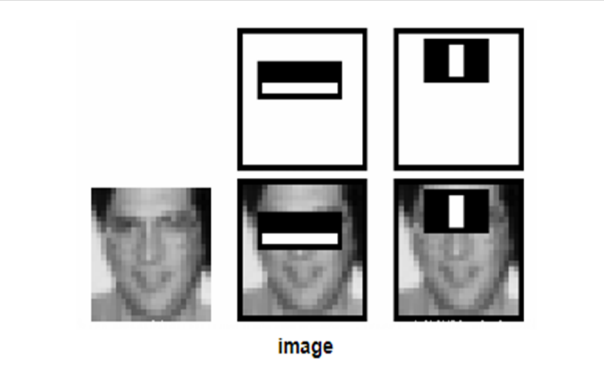


Figure 3.3: Haar cascade[1]

The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose.

Face Recognition

Face recognition is an easy task for humans. Face recognition based on the geometric features of face is probably the most intuitive approach to face recognition. One of the first automated face recognition systems was marker points (position of eyes, ears, nose etc.) were used to build a feature vector (distance between the points, angle between them etc). The recognition was performed by calculating the euclidean distance between feature vectors of a probe and reference image. Some of the latest work on geometric face recognition was, a 22-dimensional feature vector and experiments on large datasets have shown that geometrical features alone may not carry enough information for face recognition.

I. Method one based on machine learning

Machine learning (ML) is the scientific study of algorithms and statistical mod-

els that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering, and computer vision, where it is infeasible to develop an algorithm of specific instructions for performing the task
Machine learning uses types of automated algorithms which learn to predict future decisions and model and model functions using data fed to it.

The Eigenfaces and Fisherfaces methods took a holistic approach to face recognition. Recently various methods for a local feature extraction emerged. To avoid the high-dimensionality of the input data only local regions of an image are described, the extracted features are hopefully more robust against partial occlusion, illumination and small sample size. Algorithms used for a local feature extraction are Gabor Wavelets , Discrete Cosinus Transform and Local Binary Patterns.
Mainly there are three easy steps to computer coding facial recognition, which are similar to the steps that human brain use for recognizing faces. These steps are:

(1)Data Gathering: Gather face data (face images in this case) of the people you want to identify.
(2)Train the Recognizer: Feed that face data and respective names of each face to the recognizer so that it can learn.

(3)Recognition: Feed new faces of that people and see if the face recognizer, just trained before, recognizes them.

OpenCV has two built-in face recognizers. The names of those face recognizers are: EigenFaces and FisherFaces.

(i) EigenFaces Face Recognizer Algorithm

In this algorithm, a facial image is a point from a high-dimensional image space and a lower-dimensional representation is found, where classification becomes easy. The lower-dimensional subspace is found with Principal Component Analysis(PCA), which identifies the axes with maximum variance. While this kind of transformation is optimal from a reconstruction standpoint, it doesn't take any class labels into account. Imagine a situation where the variance is generated from external sources, let it be light. The axes with maximum variance do not necessarily contain any discriminative information at all, hence a classification becomes impossible. So a class-specific projection with a Linear Discriminant Analysis was applied to face recognition. The basic idea is to minimize the variance within a class, while maximizing the variance between the classes at the same time.

This algorithm considers the fact that not all parts of a face are equally important or useful for face recognition. Indeed, when you look at someone, you recognize that person by his distinct features, like the eyes, nose, cheeks or forehead; and how they vary respect to each other. Focus is on the areas of maximum change. For example, from the eyes to the nose there is a significant change, and

same applies from the nose to the mouth. When multiple faces are given, comparison is done by looking at these areas, because by catching the maximum variation among faces, they help to differentiate one face from the other.

This is how EigenFaces recognizer works. It looks at all the training images of all the people as a whole and tries to extract the components which are relevant and useful and discards the rest. These important features are called principal components.

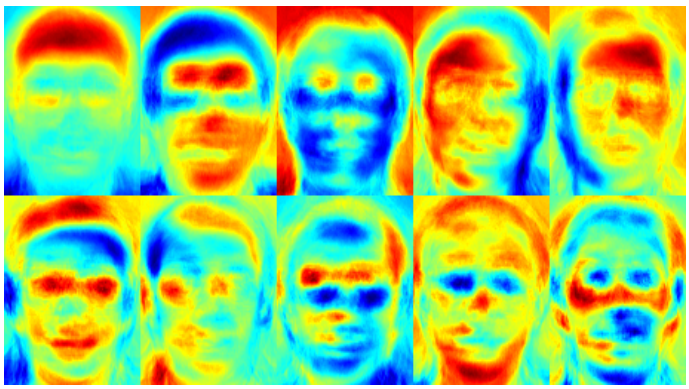


Figure 3.4: EigenFaces Face Recognizer Principal Components.[9]

So, EigenFaces recognizer trains itself by extracting principal components, but it also keeps a record of which ones belong to which person. Thus, whenever a new image is introduced to the algorithm, it repeats the same process as follows: Extract the principal components from the new picture. Compare those features with the list of elements stored during training. Find the ones with the best match. Return the 'person' label associated with that best match component. In simple words, it's a game of matching.However, one thing to note in above image is that EigenFaces algorithm also considers illumination as an important feature. In con-

sequence, lights and shadows are picked up by EigenFaces, which classifies them as representing a 'face'.Face recognition picks up on human things, dominated by shapes and shadows: two eyes, a nose, a mouth.

(ii) FisherFaces Face Recognizer Algorithm

This algorithm is an improved version of the Eigenfaces. Eigenfaces looks at all the training faces of all the people at once and finds principal components from all of them combined. By doing that, it doesn't focus on the features that discriminate one individual from another. Instead, it concentrates on the ones that represent all the faces of all the people in the training data, as a whole. Since EigenFaces also finds illumination as a useful component, it will find this variation very relevant for face recognition and may discard the features of the other people's faces, considering them less useful. In the end, the variance that EigenFaces has extracted represents just one individual's facial features. This can be done by tunning EigenFaces so that it extracts useful features from the faces of each person separately instead of extracting them from all the faces combined. In this way, even if one person has high illumination changes, it will not affect the other people's features extraction process.

The Principal Component Analysis (PCA), which is the core of the Eigenfaces method, finds a linear combination of features that maximizes the total variance in data. While this is clearly a powerful way to represent data, it doesn't consider any classes and so a lot of discriminative information may be lost when throwing components away.

The Linear Discriminant Analysis performs a class-specific dimensionality reduction. In order to find the combination of features that separates best between classes the Linear Discriminant Analysis maximizes the ratio of between-classes to within-classes scatter, instead of maximizing the overall scatter. The idea is simple: same classes should cluster tightly together, while different classes are as far away as possible from each other in the lower-dimensional representation.

Precisely, FisherFaces face recognizer algorithm extracts principal components that differentiate one person from the others. In that sense, an individual's components do not dominate (become more useful) over the others.

Below is an image of principal components using FisherFaces algorithm.

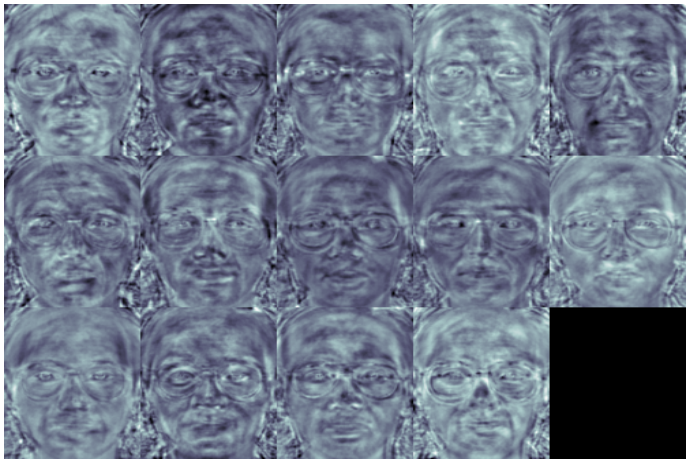


Figure 3.5: FisherFaces Face Recognizer Principal Components.[9]

One thing to note here is that FisherFaces only prevents features of one person from becoming dominant, but it still considers illumination changes as a useful feature. But light variation is not a useful feature to extract as it is not part of the

actual face, another face recognizer Algorithm must be used.

II. Method two based on deep learning

Machine Learning yielded about 75% accuracy for face recognition. Moreover the model was very sensitive to lighting conditions.

Hence a new method was designed for face recognition and that is deep learning. This model gives an accuracy upto 95% .

Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on artificial neural networks.

In the case of machine learning, the algorithm needs to be told how to make an accurate prediction by providing it with more information, whereas, in the case of deep learning, the algorithm is able to learn that through its own data processing. It is similar to how a human being would identify something, think about it, and then draw any kind of conclusion.Deep learning interprets data features and its relationships using neural networks which pass the relevant information through several stages of data processing.

The key here is to get a deep Convolutional Neural Network(CNN) to produce a bunch of numbers that describe a face (known as face encodings). When two different images of the same person are passed to the network, the network should return similar outputs (i.e. closer numbers) for both images, whereas when images of two different people are passed, the network should return very different outputs for the two images. This means that the neural network needs to be

trained to automatically identify different features of faces and calculate numbers based on that.

3.3.2 PIN Matching

A Personal Identification Number (PIN), is a numeric or alpha-numeric password used in the process of authenticating a user accessing a system.
The PIN used is a 4-digit number. The details of user along with their PINs are already stored in the database. Once the user enters the PIN, it is checked with PIN in the database. If the PIN matches with the user face and details, then system proceeds to next step. The user has three chances of entering PIN. If PIN is not matched even the third time, then the account will be temporarily blocked to ensure safety to the real user indicating that the person who arrived at the ATM is probably a robber.

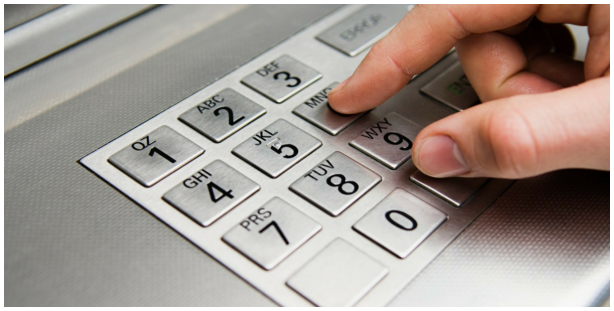


Figure 3.6: PIN

3.3.3 OTP Generation and Verification

A one-time password (OTP), also known as one-time pin, is a password that is valid for only one login session or transaction, on a computer system or other

digital device. OTPs avoid a number of shortcomings that are associated with traditional (static) password-based authentication.

For implementing OTP, Textlocal website is used to send SMS (an OTP) to user's mobile number. The idea to use mobile phones is preferred over e-mail because the people in rural areas have simple phones which can receive text messages but have no internet connections and e-mail facilities. Since mobile phones are ubiquitous, mobile phones are used so that everyone can take the benefit of the proposed system. The user will receive OTP immediately after passing the face recognition test and entering the PIN. Once OTP is received user has to enter the code which is of 4-digit. User gets three chances to enter the OTP. If the code is entered incorrectly in three consecutive attempts account gets temporarily blocked and notification is sent to registered mobile number. This feature is added in order to restrict the fraudulent means of attacking the account of a user by wearing masks or in rare cases, if unauthorized user's face mistakenly matches authorized user's face.

Chapter 4

Implementation Phase

4.1 Hardware Overview

The main hardware components used are:

1.Raspberry Pi

2.Pi Camera

4.1.1 Raspberry Pi

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries. It is a low cost general-purpose computer, usually with a Linux operating system, and the ability to run multiple programs.

The specifications include:

• Model used– Raspberry Pi 3 Model B • CPU - Quad Core 1.2GHz Broadcom

BCM2837 64bit • RAM - 1GB RAM • BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board • 100 Base Ethernet • 40-pin extended GPIO • 4 USB 2 ports • 4 Pole stereo output and composite video port • Full size HDMI • CSI camera port for connecting a Raspberry Pi camera • DSI display port for connecting a Raspberry Pi touchscreen display • Micro SD port for loading your operating system and storing data • Upgraded switched Micro USB power source up to 2.5A.

Figure 4.1: Raspberry Pi

4.1.2 Pi Camera

The Raspberry Pi Camera Module is a custom designed add-on for Raspberry Pi. It attaches to Raspberry Pi by way of one of the two small sockets on the board upper surface. This interface uses the dedicated CSI interface, which was designed especially for interfacing to cameras. The CSI bus is capable of extremely high data rates, and it exclusively carries pixel data.

The resolution of the Pi Camera used: 5 Mega Pixels.

Figure 4.2: Picamera

4.2 Software overview

I. Method one based on machine learning.

4.2.1 Raspbian

Raspbian is a Debian-based computer operating system for Raspberry Pi. There are several versions of Raspbian including Raspbian Stretch and Raspbian Jessie. Since 2015, it has been officially provided by the Raspberry Pi Foundation as the primary operating system for the family of Raspberry Pi single-board computers.

The operating system is still under active development. Raspbian is highly optimized for the Raspberry Pi line's low-performance ARM CPUs.Raspbian uses PIXEL(Pi Improved X-Window Environment, Lightweight), as its main desktop environment as of the latest update. It is composed of a modified LXDE desktop environment and the Openbox stacking window manager with a new theme and few other changes. The distribution is shipped with a copy of computer algebra program Mathematica and a version of Minecraft called Minecraft Pi as well as a lightweight version of Chromium as of the latest version.

Version of Raspbian used: Raspbian Stretch.
A 32 GB Memory Card was used for storing the Rasbian OS.

4.2.2 OpenCV

Open Source Computer Vision Library is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

One of OpenCV's goal is to provide a simple-to-use computer vision infrastructure that helps people build fairly sophisticated vision applications quickly. The OpenCV library contains over 500 functions that span many areas in vision, including factory product inspection, medical imaging, security, user interface, camera calibration, stereo vision, and robotics. Because computer vision and machin learning often go hand-in-hand, OpenCV also contains a full, general purpose Machine learning Library (MLL). This sublibrary is focused on statistical pattern recognition and clustering. The MLL is highly useful for the vision tasks that are at the core of OpenCV's mission, but it is general enough to be used for any machine learning problem.
Version used: OpenCV 4.0.0

4.2.3 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Used Python for Face Detection codes in OpenCV.

Version used: Python 3.7.2

4.2.4 Netbeans Java

Netbeans is an open source integrated development environment (IDE) for Java. Netbeans allows applications to be developed from a set of modular software components called modules. NetBeans runs on Windows, macOS, Linux and Solaris. In addition to Java development, it has extensions for other languages like PHP, C, C++, HTML5, and JavaScript. Applications based on Netbeans, including the Netbeans IDE, can be extended by third party developers.

The Netbeans platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Some features of the platform are:User interface management (e.g. menus and toolbars), User settings management, Storage management (carries out efficient storage), Window management, Wizard framework (supports step-by-step dialogs), Netbeans Visual Library Integrated development tools. Netbeans IDE modules include netbeans profiler, a GUI design tool and Netbeans Javascript editor.

Netbeans IDE for java was used for creating different Graphical User Interfaces(GUI) similar to the conventional ATM system GUI, which included several GUIs for starting the camera for face detection, displaying the username of the customer, entering the customer's high security PIN and entering the OTP sent to the customer's registered mobile number.
Version used: 8.2

4.2.5 Xampp

Xampp is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. It makes transitioning from a local test server to a live server possible.Using Xampp, we created account database of customer consisting of Customer name, username, PIN and mobile number using Xampp.

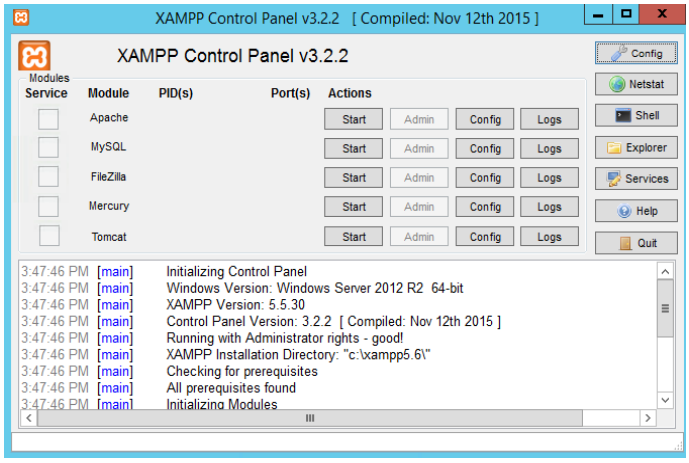


Figure 4.3: Xampp Control Panel

Xampp also comes with a number of other modules including OpenSSL, phpMyAdmin, MediaWiki, Joomla, WordPress and more. Self-contained, multiple instances of Xampp can exist on a single computer, and any given instance can be copied from one computer to another. Xampp is offered in both a full and a standard version (Smaller version). Xampp also provides support for creating and manipulating databases in MariaDB and SQLite among others

Xampp's designers intended it for use only as a development tool, to allow website designers and programmers to test their work on their own computers without any access to the Internet. To make this as easy as possible, many important security features are disabled by default. Xampp has the ability to serve web pages on the World Wide Web. A special tool is provided to password-protect the most important parts of the package.

Used Xampp Control Panel to create the User account database consisting of Customer Name, Username, PIN and Mobile Number.

4.2.6 Textlocal Messenger

Textlocal is a mobile communications company founded in 2005,having various products which includes Sending and receiving SMS online, Sending and receiving MMS online, SMS API gateway, SMS attachments, Email to SMS, Mobile web pages, Mobile forms and surveys, Mobile vouchers, tickets and loyalty cards,etc. It generates API (Application Programming Interface) keys and we can use those keys for sending and recieving SMS online.

Used for sending OTP to the registered Customer's Mobile Number.
Version used: Textlocal Messenger 3.0

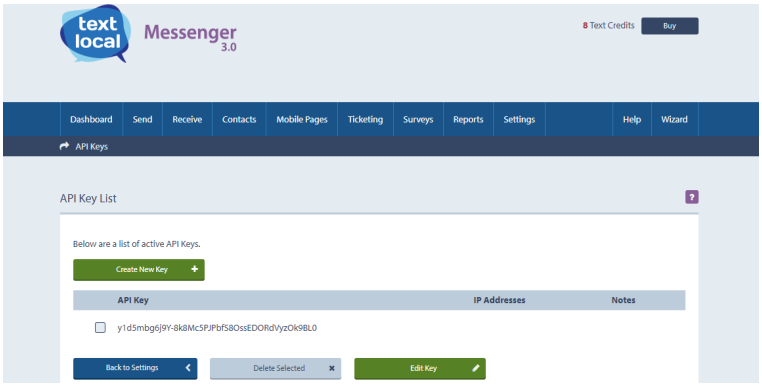


Figure 4.4: Textlocal Messenger

These were the software tools used for the project when face recognition was done using machine learning.

II.Method two based on deep learning.

Deep learning was implemented using additional softwares like Anaconda with libraries TensorFlow,Keras,OpenCV,Matplotlib and scikitlearn installed in it.

4.2.7 Anaconda

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda.
Anaconda navigator was used to install packages like tensorflow,keras,scikit in windows.
Anaconda prompt is used to run the python file (classifier.py) for face recognition.
Version used: Anaconda with Python 3.7.

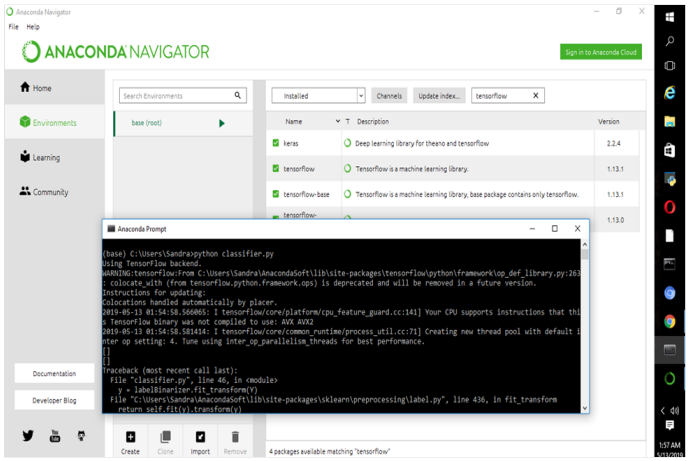


Figure 4.5: Anaconda navigator and anaconda prompt

4.2.8 TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

Tensorflow base and cpu were installed.

4.2.9 Keras

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.Keras helps in removing complexities of tensorflow. Keras is run with tensorflow in the backend.

4.2.10 Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+.

4.2.11 Scikit-learn

Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language.It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.
Scikit-image and scikit-learn were used.

4.3 Implementation Results

4.3.1 Phase 1: Face Detection

Face Detection phase was done using OpenCV with the help of python programming language. Face Detection was implemented in Windows Operating System as an initial stage using the Webcamera of the Laptop. So, during image capture using Webcamera, the computer detected the available faces from the image, using Haar Algorithm.

Further, it was implemented onto Raspberry Pi hardware using the Pi Camera ,which is interfaced with the Raspberry Pi

4.3.2 Phase 2: Creation of Image Database

Using the Face detection concept, we created a face image database of 60 people with 100 images each, by entering the name of each person during image capture. So, the 100 images of each person will be saved using the names entered. Images were captured in different lighting conditions. The images captured was performed with an Image Enhancement technique called Sharpening. The sharpened images were saved in JPEG format with dimensions 120X120.

The Face images were trained using Java codes.

4.3.3 Phase 3: Face Recogntion

We initially implemented Face Recognition by displaying the corresponding names of the detected faces (from the image dataset) and the Confidence value in percentage, as shown in Figure 4.6. Face recognition has been implemented using Fisher Faces algorithm and using the concept of confusion matrix, the accuracy was measured as 80 percent.

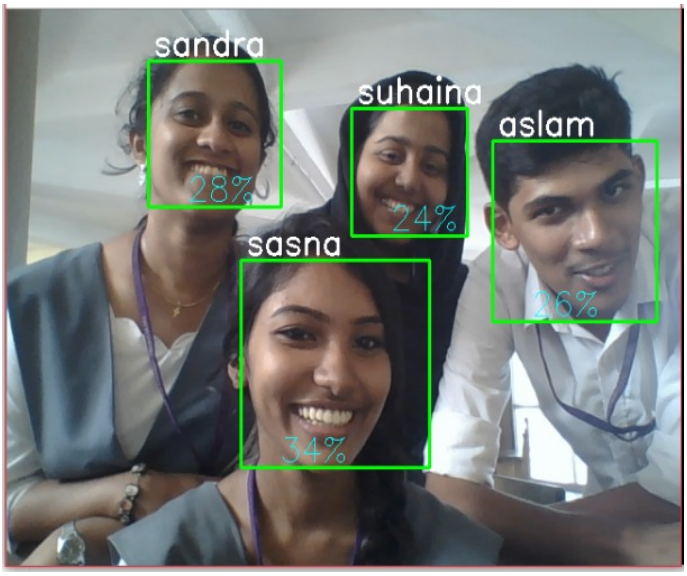


Figure 4.6: Face recognition

To improve face recognition accuracy and provide an efficient security layer for the ATM system, we came into the idea of using Deep Learning.

4.3.4 Phase 4: Creation of User account Database

The database was created using Xampp software in the server(windows system). The database consists of fields like user id(uid), username(uname), full name(Name), bankname(bank), password(pass) and mobile number(phone).

☐	Edit	Copy	Delete	1	ajay	Ajaynath S A	HDFC	1912	9447990618
☐	Edit	Copy	Delete	2	ajith	Ajith Kurian Sabu	SBI	1999	9495587572
☐	Edit	Copy	Delete	3	aleenatg	Aleena Terese George	SBI	1976	9497046818
☐	Edit	Copy	Delete	4	amal	Amal Raj	HDFC	1995	8606670730
☐	Edit	Copy	Delete	5	anitta	Anitta Thankachan	IOB	1972	7902380286
☐	Edit	Copy	Delete	6	anjitham	Anjitha M	IOB	1974	8281817496
☐	Edit	Copy	Delete	7	ashad	Ashad Anil	HDFC	1979	7558870230
☐	Edit	Copy	Delete	8	athul	Athul Unnikrishnan	SBI	1983	9961416939
☐	Edit	Copy	Delete	9	dibin	Dibin C T	SBI	1980	8301939186
☐	Edit	Copy	Delete	10	gayathrivi	Gayathri Vinod	IOB	1985	8281817308
☐	Edit	Copy	Delete	11	hannah	Hannah Basil	HDFC	1970	8606675375
☐	Edit	Copy	Delete	12	induja	Induja Sasidharan	IOB	1908	8113009403
☐	Edit	Copy	Delete	13	joex	Joe Xavier	SBI	1982	8138053803
☐	Edit	Copy	Delete	14	meera	Meera P S	ICICI	1977	8281805315
☐	Edit	Copy	Delete	15	aslam	Muhammed Aslam P.S	ICICI	1909	9497798781
☐	Edit	Copy	Delete	16	nikhil	Nikhil M Jeby	IOB	1971	8156915169
☐	Edit	Copy	Delete	17	sandra	Sandra Maria Roy	SBI	1907	9605169753
☐	Edit	Copy	Delete	18	sasna	Sasna Sunder	ICICI	1904	9895114934
☐	Edit	Copy	Delete	19	sherin	Sherin Raju	IOB	1978	9497133201
☐	Edit	Copy	Delete	20	suhaina	Suhaina J	HDFC	1903	8301981962
☐	Edit	Copy	Delete	21	teresa	Teresa Abraham	ICICI	1986	8089344798
☐	Edit	Copy	Delete	22	viswam	Viswam Gopinath	IOB	1902	8301981961
☐	Edit	Copy	Delete	23	aslam	Muhammed Aslam P.S	SBI	1987	9497798781

Figure 4.7: Account Database

4.3.5 Phase 5: Implementing the the Face Recognition phase onto the ATM GUI

We used Netbeans Java software to provide a Graphical User Interface (GUI) with a flow of security layer. Different GUI frames were created similar to the conventional ATM system. In addition to the usual ATM GUIs, we have Face Recognition Stage in the beginning and an OTP verification stage at the end.

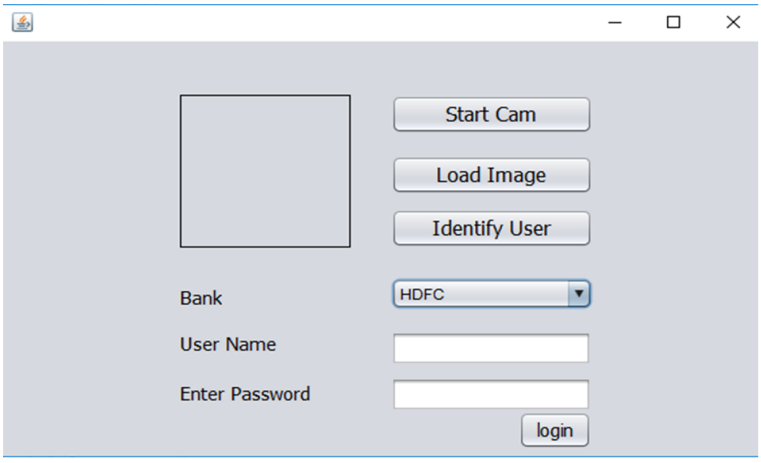


Figure 4.8: First GUI frame

The First GUI Frame of our ATM system is as shown in Figure .4.7. The Frame has the following Options:

1. Start Cam: This button allows to start camera device, which hereby is the Pi Camera. Then it proceeds with the Face Detection Phase. During Face Detection, it allows to detect only one face at a time.

2.Load Image: This button allows to load the face image, which has been detected during Face Detection, in the square vacancy provided on the top left portion of the frame.

3.Identify User: This button allows to proceed with the Face Recognition Phase. The loaded image on the box, is verified with the image Database created and displays the username of the corresponding user in the 'User Name' label.

4.Choose Bank Option: After completion of the Face recogntion stage, the user is allowed to choose the Bank where she/he has the Account.

5.Enter Password: The user is allowed to enter the high security PIN in the field provided.

4.3.6 Phase 6: PIN Matching

Once the Face Detection, Face Recognition stages are completed, the frame will be obtained as shown in Figure 4.8, provided with an example Face image.

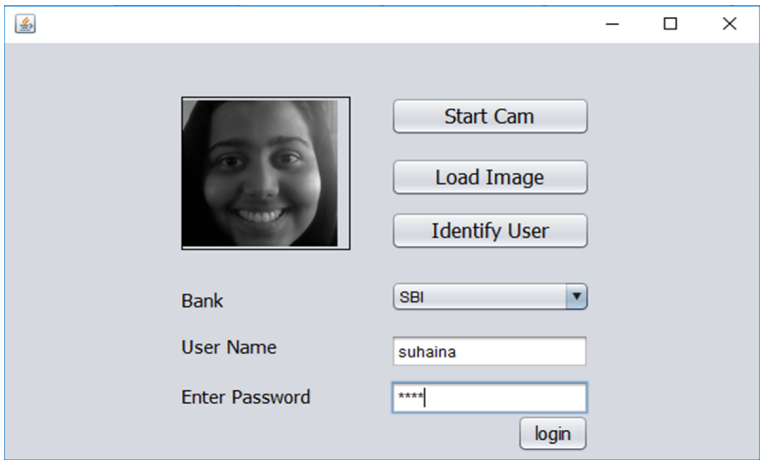


Figure 4.9: GUI frame after the user has entered the details

The user is allowed to enter the PIN in the field provided and is then allowed to proceed further using 'Login' button.

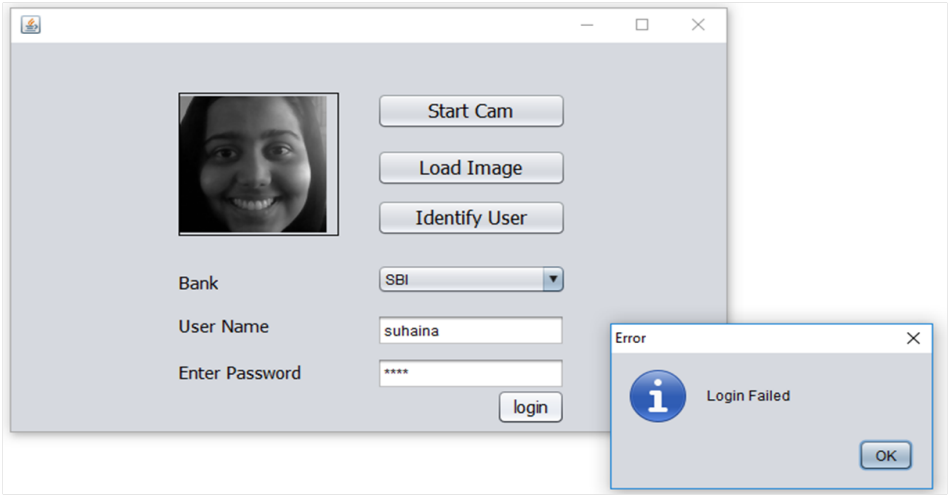


Figure 4.10: Identifies user and displays as shown if the entered Password (PIN) is incorrect

As shown in figure above, if the entered PIN does not match with the one present in the User Account Database, then a pop-up message showing 'Login Failed' will be displayed.

4.3.7 Phase 7: OTP generation and Verification

Once the entered PIN is verified with the PIN in the database, the user will be able to Login further. Then, a 4-digit One-Time Password will be generated and sent to the user's registered 10-digit mobile number within few seconds.Then, the user is allowed to enter the generated OTP in the field provided and proceed for 'Submit'.

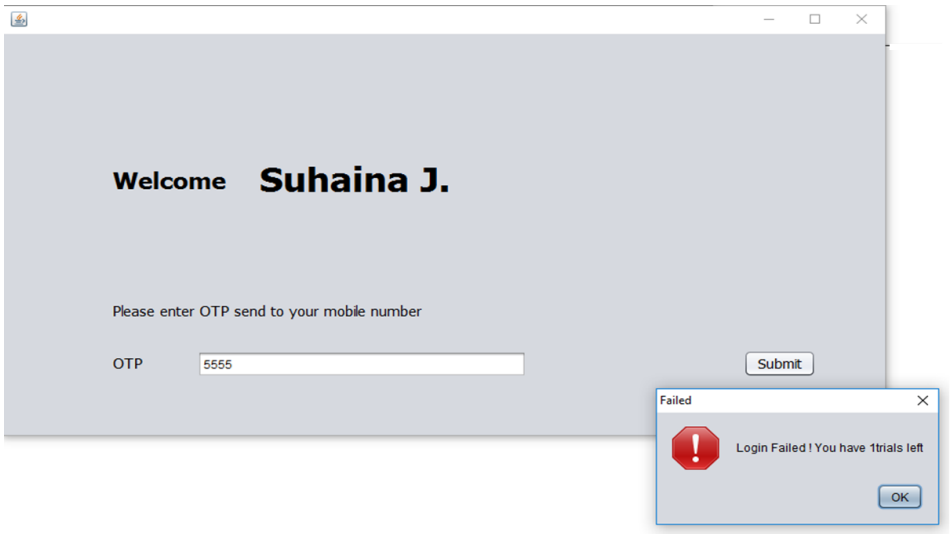


Figure 4.11: Entering the One-time password

As shown in figure above, if the OTP entered is not verified, then it displays a pop-up message showing 'Login Failed'. The user is allowed to enter the correct OTP for upto a maximum of 3 trials.

Figure 4.12: Verifying the OTP

If the OTP matches, then it displays the Name of the Customer as 'Welcome <Name of Customer>', as shown above.Once the OTP entered is verified, the user can proceed to further transaction ,which is provided by the conventional ATM system. A pop-up message showing 'Login Successful!' will be displayed as shown in above figure.

In the hardware part, the Database which includes the Face image database and the User account database is stored within a Windows 10.0 Laptop , defining it as the Server. The Real-time ATM system is defined in a Raspberry Pi hardware with the help of a monitor screen, keyboard and mouse.

Chapter 5

Conclusions and Future Scope

5.1 Conclusion

Facial recognition has proven to be one of the most secure methods of all biometric systems to a point for high level security and to avoid ATM robberies and provide security for ATM.

In the proposed project, it replaces the traditional ATM system. It has advantages such as saves manufacturing cost of cards and overcomes drawbacks of the traditional system like carrying the ATM card, losing of card, fraud calls related to ATM card, etc. and provides high security by using authentication like fingerprint and OTP systems

With new improved techniques in the field of artificial Intelligence that help eliminate more disturbances and distortions, the rate of effectiveness of the system

can be improved.

5.2 Future Scope

This project can be used for real time security applications like in ATM security systems, military applications, high security companies. This can also be used in bank locker access. Lighting provided to the system is a key factor to be taken care of. Usage of high speed computers can improve the efficiency.

Bibliography

[1] J.J.Patoliya, M.M. Desai, "Face Detection based ATM Security System using Embedded Linux Platform ", *2nd International Conference for Convergence in Technology (I2CT)*, 2017.

[2] M.Karovaliyaa, S.Karediab, S.Ozac, Dr.D.R.Kalbande, "Enhanced security for ATM machine with OTP and Facial recognition features",*International Conference on Advanced Computing Technologies and Applications (ICACTA)*, 2015.

[3] Sivakumar T. 1 , G. Askok 2 , k. S. Venuprathap, "Design and Implementation of Security Based ATM theft Monitoring system", *International Journal of Engineering Inventions*, Volume 3, Issue 1, 2013.

[4] C. Bhosale, P. Dere, C. Jadhav, "ATM security using face and fingerprint recognition", *International Journal of Research in Engineering, Technology and Science*, Volume VII, Special Issue, Feb 2017.

[5] Manoj V , M. Sankar R , Sasipriya S , U. Devi E, Devika T , "Multi Authentication ATM Theft Prevention Using iBeacon", *International Research Journal of Engineering and Technology (IRJET)*.

[6] L. Wang,H. Ji, Y. Shi, " Face recognition using maximum local fisher discriminant analysis", *18th IEEE International Conference on Image Processing*, 2011.

[7] K.Shailaja and Dr.B.Anuradha, "Effective Face Recognition using Deep Learning based Linear Discriminant Classification ", *IEEE International Conference on Computational Intelligence and Computing Research*, 2016.

[8] H. R. Babaei, O. Molalapata and A.H.Y Akbar Pandor, "Face Recognition Application for Automatic Teller Machines (ATM)", *International Conference on Information and Knowledge Management (ICIKM)*, 2012.

[9] https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec-tutorial.htmlface-recognition

[10] https://www.superdatascience.com/opencv-face-recognition/

[11] https://www.rankred.com/face-recognition-algorithms-techniques/

[12] https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/

[13] https://towardsdatascience.com/facial-recognition-using-deep-learning-a74e9059a150

Appendix

1. Python code for Face Detection

```
import cv2

import sys

import logging as log

import datetime as dt

from time import sleep

from PIL import Image


cascPath = "haar.xml"

faceCascade = cv2.CascadeClassifier(cascPath)

video_capture = cv2.VideoCapture(0)

anterior = 0

while True:

if not video_capture.isOpened():

sleep(5)

pass

ret, frame = video_capture.read()

gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
faces = faceCascade.detectMultiScale(
gray,
scaleFactor=1.1,
minNeighbors=5,
minSize=(30, 30)
)
for (x, y, w, h) in faces:
cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
cv2.imshow('Video', frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
crop_img = frame[y:y+h, x:x+w]
crop_img=resized = cv2.resize(crop_img,(120,120), interpolation = cv2.INTER_AREA)
cv2.imwrite('input.png',crop_img)
break
video_capture.release()
cv2.destroyAllWindows()

import cv2  #importing modules of opencv
import numpy as np
import os

cam = cv2.VideoCapture(0)   #sets video source to videocam
cam.set(3, 640)   # set video width
cam.set(4, 480)   # set video height
face_detector = cv2.CascadeClassifier(r'C:\Users\Sandra\Desktop\PROJECT\pyth\haarcascade_frontalface
# For each person, enter one numeric face id
```

```
name = input('\n enter name end press <return> ==>  ')
print("\n [INFO] Initializing face capture. Look the camera and wait ...")
# Initialize individual sampling face count
count = 0
while(True):
    ret, img = cam.read()
    # Create our shapening kernel, it must equal to one eventually
    kernel_sharpening = np.array([[-1,-1,-1],
                                  [-1, 9,-1],
                                  [-1,-1,-1]])
    # applying the sharpening kernel to the input image & displaying it.
    sharpened = cv2.filter2D(img, -1, kernel_sharpening)
    #cv2.imshow('Image Sharpening', sharpened)
    #converting sharpened image to gray.
    gray = cv2.cvtColor(sharpened, cv2.COLOR_BGR2GRAY)
    faces = face_detector.detectMultiScale(gray, 1.3, 10)
    for (x,y,w,h) in faces:
        cv2.rectangle(img, (x,y), (x+w,y+h), (0,0,255), 2)
        count += 1
        # Save the captured image into the datasets folder
        #resizing image size to 120X120 pixel size
        crop_img = img[y:y+h, x:x+w]
        crop_img= cv2.resize(crop_img, (120,120), interpolation = cv2.INTER_AREA)
        #saving to folder
        cv2.imwrite("datastet2/" + str(name) + '.' + str(count) + ".jpg", gray[y:y+h,x:x+w])
        cv2.imshow('image', crop_img)
```

```
    k = cv2.waitKey(2) & 0xff # Press 'ESC' for exiting video
    if k == 27:
        break
    elif count >= 100: # Take 30 face sample and stop video
         break
# Do a bit of cleanup
print("\n [INFO] Exiting Program and cleanup stuff")
cam.release()
cv2.destroyAllWindows()
```

2. Java codes for Face Recognition based on machine learning

```
package facerecognition;

import GUI.login;
import java.io.FileNotFoundException;
import java.io.IOException;
public class FaceRecognition {

    public static void main(String[] args) throws IOException, FileNotFoundException
    , ClassNotFoundException {
       new login().setVisible(true);
 }
}
package facerecognition;

import java.awt.Graphics;
```

```
import java.awt.image.BufferStrategy;
import java.awt.image.BufferedImage;
import java.awt.image.DataBufferByte;
import java.io.File;
import java.io.IOException;
import javax.imageio.ImageIO;
import javax.swing.JFrame;
import javax.swing.JPanel;
import org.opencv.core.Core;
import org.opencv.core.Mat;
import org.opencv.core.MatOfRect;
import org.opencv.core.Point;
import org.opencv.core.Rect;
import org.opencv.core.Scalar;
import org.opencv.imgproc.Imgproc;
import org.opencv.objdetect.CascadeClassifier;
import org.opencv.videoio.VideoCapture;
public class HumanDetection {

    public static JFrame frame;
    BufferedImage img;
    public static int WIDTH = 600;
    public static int HEIGHT = 400;
    public HumanDetection() throws InterruptedException, IOException {
        System.load( "C:\\opencv\\build\\java\\x64\\opencv_java400.dll" );
      VideoCapture capture = new VideoCapture(0);
```

```
Mat matrix = new Mat();
capture.read(matrix);
frame=new JFrame();
frame.setSize(HEIGHT,WIDTH);
frame.setVisible(true);
JPanel panel=new JPanel();
frame.add(panel);
int i=0;
String xmlFile = "face.xml";
CascadeClassifier classifier = new CascadeClassifier(xmlFile);

while(true){
 if( capture.isOpened()) {
   if (capture.read(matrix)) {
      MatOfRect faceDetections = new MatOfRect();
      classifier.detectMultiScale(matrix, faceDetections);
      System.out.println(String.format("Detected %s faces",
         faceDetections.toArray().length));

      for (Rect rect : faceDetections.toArray()) {
         Imgproc.rectangle(matrix, new Point(rect.x, rect.y),
            new Point(rect.x + rect.width, rect.y + rect.height),
            new Scalar(0, 0, 255), 3);
      }
      BufferedImage image = new BufferedImage(matrix.width(),
       matrix.height(), BufferedImage.TYPE_3BYTE_BGR);
```

```
            byte[] data = ((DataBufferByte) image.getRaster().getDataBuffer()).getData();
            matrix.get(0, 0, data);
            File outputfile = new File("saved"+ i++ +".png");
            Graphics g= panel.getGraphics();
            g.drawImage(image, 0, 0, WIDTH, HEIGHT, null);
            g.dispose();
            System.out.println("writing...");
            Thread.sleep(1000/15);
    }
      }
      }
    }
    public static void main(String[] args) throws InterruptedException, IOException {
        new HumanDetection();
    }
}
package facerecognition;

import java.io.Serializable;
import org.bytedeco.javacpp.opencv_face;
public class ModelWrapper implements Serializable{
    opencv_face.FaceRecognizer faceRecognizer;

}
package facerecognition;
```

```
public class Newuserdatabase extends javax.swing.JFrame {

    public Newuserdatabase() {

        initComponents();

    }

    private void initComponents() {


        jLabel1 = new javax.swing.JLabel();

        jLabel2 = new javax.swing.JLabel();

        jLabel3 = new javax.swing.JLabel();

        jLabel4 = new javax.swing.JLabel();

        jTextField1 = new javax.swing.JTextField();

        jTextField2 = new javax.swing.JTextField();

        jTextField3 = new javax.swing.JTextField();

        jTextField4 = new javax.swing.JTextField();

        jButton1 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jLabel1.setText("uname");

        jLabel2.setText("password");

        jLabel3.setText("name");

        jLabel4.setText("jLabel4");

        jTextField1.setText("jTextField1");

        jTextField1.addActionListener(new java.awt.event.ActionListener() {

            public void actionPerformed(java.awt.event.ActionEvent evt) {

                jTextField1ActionPerformed(evt);

            }

        });
```

```
jTextField2.setText("jTextField2");
jTextField3.setText("jTextField3");
jTextField4.setText("jTextField4");
jButton1.setText("ADD");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addGap(38, 38, 38)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
            .addComponent(jButton1)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
            false)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(jLabel4)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
                    javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
```

```
                        PREFERRED_SIZE))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
                    layout.createSequentialGroup()
                        .addComponent(jLabel3)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
                        javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED_SIZE,
                        javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                        PREFERRED_SIZE))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
                    layout.createSequentialGroup()
                        .addComponent(jLabel2)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
                        javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE,
                        javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                        PREFERRED_SIZE))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
                    layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addGap(93, 93, 93)
                        .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE,
                        javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                        PREFERRED_SIZE))))
            .addContainerGap(178, Short.MAX_VALUE))
    );
```

```
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addGap(67, 67, 67)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
            .addComponent(jLabel1)
            .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE,
            javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(24, 24, 24)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel2)
            .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE,
            javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(22, 22, 22)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel3)
            .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED_SIZE,
            javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(20, 20, 20)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel4)
            .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED_SIZE,
            javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 33,
        Short.MAX_VALUE)
        .addComponent(jButton1)
```

```
            .addGap(31, 31, 31))
    );
    pack();
}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    Newuserdatabase nudb=new Newuserdatabase();
 }
private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {
}
public static void main(String args[]) {
 try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(Newuserdatabase.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(Newuserdatabase.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
```

```
            java.util.logging.Logger.getLogger(Newuserdatabase.class.getName()).
            log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(Newuserdatabase.class.getName()).
            log(java.util.logging.Level.SEVERE, null, ex);
        }
       java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Newuserdatabase().setVisible(true);
            }
        });
    }
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
}


package facerecognition;
import java.io.File;
import java.io.FilenameFilter;
```

```
import java.nio.IntBuffer;
import java.util.HashMap;
import java.util.Map;
import static org.bytedeco.javacpp.opencv_core.CV_32SC1;
import static org.bytedeco.javacpp.opencv_core.CV_8UC1;
import static org.bytedeco.javacpp.opencv_imgproc.*;
import static org.bytedeco.javacpp.opencv_imgcodecs.imread;
import static org.bytedeco.javacpp.opencv_imgcodecs.IMREAD_GRAYSCALE;
import org.bytedeco.javacpp.BytePointer;
import org.bytedeco.javacpp.IntPointer;
import org.bytedeco.javacpp.DoublePointer;
import org.bytedeco.javacpp.opencv_core.Mat;
import org.bytedeco.javacpp.opencv_face.FaceRecognizer;
import org.bytedeco.javacpp.opencv_face.FisherFaceRecognizer;
import org.bytedeco.javacpp.opencv_face.EigenFaceRecognizer;
import org.bytedeco.javacpp.opencv_face.LBPHFaceRecognizer;
import org.bytedeco.javacpp.opencv_core.MatVector;
import org.bytedeco.javacpp.opencv_core.Size;
import org.opencv.imgproc.Imgproc;

public class OpenCVFaceRecognizer {
    public static void main(String[] args) {
        String trainingDir = "D:\\FaceRecognition\\datastet2";
        Mat testImage = imread("D:\\FaceRecognition\\test\\qry.jpg", IMREAD_GRAYSCALE);
        File root = new File(trainingDir);
        FilenameFilter imgFilter = new FilenameFilter() {
```

```
    public boolean accept(File dir, String name) {
        name = name.toLowerCase();
        return name.endsWith(".jpg") || name.endsWith(".pgm") || name.endsWith(".png");
    }
};
File[] imageFiles = root.listFiles(imgFilter);
MatVector images = new MatVector(imageFiles.length);
Mat labels = new Mat(imageFiles.length, 1, CV_32SC1);
IntBuffer labelsBuf = labels.createBuffer();
int counter = 0;
HashMap<String,Integer> users=new HashMap<String, Integer>();
for (File image : imageFiles) {
    System.out.println(""+image.getName());
    Mat img = imread(image.getAbsolutePath(), IMREAD_GRAYSCALE);
    int label=-1;
    String slabel = image.getName().split(".")[0];
    if(!users.containsKey(slabel))
    {
        users.put(slabel, users.size()+1);
    }
    label=users.get(slabel);
    images.put(counter, img);
    labelsBuf.put(counter, label);
    counter++;
}
FaceRecognizer faceRecognizer = FisherFaceRecognizer.create();
```

```
        faceRecognizer.train(images, labels);
        IntPointer label = new IntPointer(1);
        DoublePointer confidence = new DoublePointer(1);
        faceRecognizer.predict(testImage, label, confidence);
        int predictedLabel = label.get(0);

        String user="";
        for (Map.Entry<String,Integer> entry : users.entrySet())
        {
           if(entry.getValue()==predictedLabel)
           {
               user=entry.getKey();
               break;
           }
        }
        System.out.println("Predicted label: " + user);
    }
}

package facerecognition;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.util.HashMap;
```

```
import java.util.Map;
import org.bytedeco.javacpp.DoublePointer;
import org.bytedeco.javacpp.IntPointer;
import org.bytedeco.javacpp.opencv_core;
import org.bytedeco.javacpp.opencv_face;
import static org.bytedeco.javacpp.opencv_imgcodecs.IMREAD_GRAYSCALE;
import static org.bytedeco.javacpp.opencv_imgcodecs.imread;

public class RecognizeFace {

    ModelWrapper model=new ModelWrapper();
     HashMap<String,Integer> users;
     double confidence=0;
    public RecognizeFace() throws FileNotFoundException, IOException, ClassNotFoundException {

        model.faceRecognizer=opencv_face.FisherFaceRecognizer.create();
        model.faceRecognizer.read("model.ser");//trained model
       FileInputStream fis = new FileInputStream("users.ser");
        ObjectInputStream ois = new ObjectInputStream(fis);
        users = (HashMap<String, Integer>) ois.readObject();
        ois.close();
    }
    public String Predict(String path)
    {

        opencv_core.Mat testImage = imread(path, IMREAD_GRAYSCALE);
```

```
        IntPointer label = new IntPointer(1);
        DoublePointer confidence = new DoublePointer(1);
        model.faceRecognizer.predict(testImage, label, confidence);
        this.confidence=confidence.get();
        System.out.println("Confidence  - "+confidence.get());
        int predictedLabel = label.get(0);
        String user="";
        for (Map.Entry<String,Integer> entry : users.entrySet())
        {
           if(entry.getValue()==predictedLabel)
           {
               user=entry.getKey();
               break;
           }
        }
        System.out.println("Predicted label: " + user);
               return user;
    }
    public double getConfidence()
    {
        return confidence;
    }
}
package facerecognition;

import java.io.File;
```

```
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FilenameFilter;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.nio.IntBuffer;
import java.util.HashMap;
import org.bytedeco.javacpp.opencv_core;
import static org.bytedeco.javacpp.opencv_core.CV_32SC1;
import org.bytedeco.javacpp.opencv_face;
import static org.bytedeco.javacpp.opencv_imgcodecs.IMREAD_GRAYSCALE;
import static org.bytedeco.javacpp.opencv_imgcodecs.imread;
public class TrainModel {

    String path="";
    ModelWrapper model;
    public TrainModel() {
        model=new ModelWrapper();
    }
    void setDataSetPath(String path)
    {
        this.path=path;
    }
    void trainModel() throws FileNotFoundException, IOException
    {
        String trainingDir = path;
```

```
File root = new File(trainingDir);
FilenameFilter imgFilter = new FilenameFilter() {
    public boolean accept(File dir, String name) {
        name = name.toLowerCase();
        return name.endsWith(".jpg") || name.endsWith(".pgm") || name.endsWith(".png");
    }
};
File[] imageFiles = root.listFiles(imgFilter);
opencv_core.MatVector images = new opencv_core.MatVector(imageFiles.length);
opencv_core.Mat labels = new opencv_core.Mat(imageFiles.length, 1, CV_32SC1);
IntBuffer labelsBuf = labels.createBuffer();
int counter = 0;
HashMap<String,Integer> users=new HashMap<String, Integer>();
for (File image : imageFiles) {
    System.out.println(""+image.getName());
    opencv_core.Mat img = imread(image.getAbsolutePath(), IMREAD_GRAYSCALE);
    int label=-1;
    String fname=image.getName();
    String parts[]=fname.split("\\.");
    String slabel = parts[0];
    if(!users.containsKey(slabel))
    {
        users.put(slabel, users.size()+1);
    }
    label=users.get(slabel);
    images.put(counter, img);
```

```
            labelsBuf.put(counter, label);
            counter++;
        }

        model.faceRecognizer= opencv_face.FisherFaceRecognizer.create();
        model.faceRecognizer.train(images, labels);
        model.faceRecognizer.write("model.ser");
        FileOutputStream fos;
        ObjectOutputStream oos;
        fos = new FileOutputStream("users.ser");
        oos = new ObjectOutputStream(fos);
        oos.writeObject(users);
        oos.close();
    }
    public static void main(String[] args) throws IOException {
        TrainModel model=new TrainModel();
        model.setDataSetPath("D:\\FaceRecognition\\datastet2");
        model.trainModel();
    }
}
```

3. Python Code for face recognition based on deep Learning

```
import OS
from PIL import Image
import numpy as np
import os
```

```
import cv2
from matplotlib import pyplot as plt
from keras.applications.vgg16 import VGG16
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input
import numpy as np
from keras.applications.vgg16 import decode_predictions
model = VGG16(weights='imagenet', include_top=False)
X = []
Y = []
base_path='E:/PROJECTS/FACEREC/data/colour_photos_laptop'
source_path=base_path
for child in os.listdir(source_path):
    sub_path = os.path.join(source_path, child)
    bsub_path = os.path.join(base_path, child)
    if os.path.isdir(sub_path):
        for data_file in os.listdir(sub_path):
            Qry = Image.open(os.path.join(sub_path, data_file))
            Qry = np.array(Qry.resize((224,224)))
            Qry = Qry.reshape([-1,224,224,3])
            features_train=model.predict([Qry])
            X.append(features_train.flatten())
            Y.append(child)
print(X)
print(Y)
```

```
# In[2]:


from sklearn.preprocessing import LabelBinarizer

labelBinarizer = LabelBinarizer()

y = labelBinarizer.fit_transform(Y)

print(y)


# In[3]:


from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(np.array(X), np.array(y), test_size=0.2, random_


import tensorflow as tf

from tensorflow.python.keras import layers

from tensorflow.python.keras import models

import random

random.seed(42)

np.random.seed(42)

tf.set_random_seed(42)

dnnModel=models.Sequential()

dnnModel.add(layers.Dense(50,activation="relu",input_shape=(25088,)))

dnnModel.add(layers.Dense(30,activation="relu"))

dnnModel.add(layers.Dense(30,activation="relu"))

dnnModel.add(layers.Dense(30,activation="relu"))

dnnModel.add(layers.Dense(4,activation="softmax"))

dnnModel.summary()
```

```
dnnModel.compile(optimizer="adam",loss="categorical_crossentropy",metrics=["accuracy"])
tbCallBack=tf.keras.callbacks.TensorBoard(log_dir='./Graph',histogram_freq=0,write_graph=True,write_
dnnModel.fit(X_train,y_train,epochs=50,batch_size=64,callbacks=[tbCallBack])
testloss, testAccuracy=dnnModel.evaluate(X_test,y_test)
print(testAccuracy)
```

4. Java codes for creation of different GUI

```
package GUI;

import java.awt.GraphicsConfiguration;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;

public class OTPVerification extends javax.swing.JFrame {

    String uname,name,phone;
    int trials=0;
    public OTPVerification() {
        initComponents();
    }

    public OTPVerification(String uname,String name, String phone) {

        initComponents();
```

```
        this.uname = uname;
        this.name = name;
        this.phone = phone;
        usertext.setText(name);
        sms.sendSms(phone);
    }
    private void initComponents() {

        hailL = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        otpTF = new javax.swing.JTextField();
        SubmitB = new javax.swing.JButton();
        usertext = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        hailL.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
        hailL.setText("Welcome");
        jLabel2.setFont(new java.awt.Font("Tahoma", 0, 15)); // NOI18N
        jLabel2.setText("Please enter OTP send to your mobile number");
        jLabel3.setFont(new java.awt.Font("Tahoma", 0, 15)); // NOI18N
        jLabel3.setText("OTP");
        otpTF.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                otpTFActionPerformed(evt);
            }
```

```
});

SubmitB.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
SubmitB.setText("Submit");
SubmitB.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        SubmitBActionPerformed(evt);
    }
});

usertext.setFont(new java.awt.Font("Tahoma", 1, 36)); // NOI18N
javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout
    .Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addGap(108, 108, 108)
        .addGroup(layout.createParallelGroup(javax.swing.
        GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
               .addGroup(layout.createParallelGroup(javax.
               swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel2, javax.swing.GroupLayout.
                    PREFERRED_SIZE, 449, javax.swing.GroupLayout.PREFERRED_SIZE)
                   .addGroup(layout.createSequentialGroup()
```

```
                        .addComponent(haiL)
                        .addGap(33, 33, 33)
                        .addComponent(usertext, javax.swing.GroupLayout.
                        PREFERRED_SIZE, 441, javax.swing.GroupLayout.
                        PREFERRED_SIZE)))
                    .addGap(0, 173, Short.MAX_VALUE))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
                    65,javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(18, 18, 18)
                    .addComponent(otpTF, javax.swing.GroupLayout.PREFERRED_SIZE,323,
                    javax.swing.GroupLayout.
                    PREFERRED_SIZE)
                    .addPreferredGap(javax.swing.LayoutStyle.
                    ComponentPlacement.RELATED, 209,Short.MAX_VALUE)
                    .addComponent(SubmitB)
                    .addGap(68, 68, 68))))
);
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(142, 142, 142)
                .addComponent(haiL)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
```

```
                    97, Short.MAX_VALUE))
                .addGroup(javax.swing.GroupLayout.Alignment.
                TRAILING, layout.createSequentialGroup()
                    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(usertext, javax.swing.GroupLayout.PREFERRED_SIZE,
                    52, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(87, 87, 87)))
            .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
            56, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(otpTF,javax.swing.
                GroupLayout.PREFERRED_SIZE,javax.swing.GroupLayout.DEFAULT_SIZE,javax.swing.
                GroupLayout.PREFERRED_SIZE)
                .addComponent(jLabel3,javax.swing.GroupLayout
                .PREFERRED_SIZE, 44, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(SubmitB))
            .addGap(55, 55, 55))
    );

    pack();
}

private void SubmitBActionPerformed(java.awt.event.ActionEvent evt) {
    if(trials<3)
    {
```

```
            String otp=otpTF.getText();
            if(otp.equals(Globals.OTP))
            {
                JOptionPane.showMessageDialog(null, "Login Successfull","Success",1);
                JOptionPane.showMessageDialog(null, "Enter Amount","Success",1);
            }
            else
            {
                trials++;
               JOptionPane.showMessageDialog(null, "Login Failed ! You have
               "+(3-trials)+" trials left","Failed",0);
            }
        }
        else
        {
            JOptionPane.showMessageDialog(null, "Access Denied","Failed",0);
        }
    }
    private void otpTFActionPerformed(java.awt.event.ActionEvent evt) {
    }
    public static void main(String args[]) {
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
            getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
```

```
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(OTPVerification.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(OTPVerification.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(OTPVerification.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(OTPVerification.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    }
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new OTPVerification().setVisible(true);
        }
    });
}
private javax.swing.JButton SubmitB;
private javax.swing.JLabel haiL;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
```

```
    private javax.swing.JTextField otpTF;

    private javax.swing.JLabel usertext;

}
```