Face Recognition with Android Device

MS Project report

Project advisor

Dr. Stefan Robila

Developed by

Pritesh Parekh

Spring 2015

Department of Computer Science

Montclair State University

Montclair, NJ 07043

Acknowledgements

I am thankful to all the people who directly or indirectly helped me in completion of this project.

To begin with, I am whole heartedly thankful to **Department of Computer Science** of **Montclair State University** for providing me a developing and creative environment that would always be treasured by the students as they leave the school as a confident person with better skills. Faculties always try to make student comfortable and always welcome ideas that would make **MSU** a better place to learn and grow.

I take immense pride in paying gratitude to **Dr. Stefan Robila**, Professor at MSU for allowing me to undertake project under his supervision. He is a person who has well maintained a bridge between a guide and a friend and thus making easy for me to communicate my ideas and doubts. He always appreciates innovative ideas and gives students a chance to implement them. The regular presentation of my project under his guidance evolved `us with better communication skills. His constant effort to improve me, his uncompromising demand to achieve perfection and giving me tactics as how to complete our project on deadlines can never overlooked.

I am also thankful to **Dr. Constantine Coutras**, Department Chair and Professor, MSU and **Dr. Angel Gutierrez**, Professor at MSU for agreeing to be the part of committee for this project.

I could not complete my project without the resources provided by MSU. They indirectly made a huge contribution in the completion of my project. I end with thanking the entire MSU family for providing all the required resource and their support.

Pritesh Parekh

parekhp1@mail.montclair.edu

Table of Contents

Acknowledgements 2

Table of Figures 6

ABSTRACT 7

Chapter 1: Introduction 8

1.1. Problem and Motivation 9

1.2. Development of the System 10

1.3 Comparison with other solutions 11

Chapter 2: Functional & Non Functional Requirements 13

2.1. Functional Requirements 13

2.2. Non Functional Requirements 13

2.3 System Requirements 13

2.3.1 Operating System 13

2.3.2 Android Requirements 14

2.3.3 Web Server 14

2.4.4 Technology 14

2.4.5 Database 14

2.4.6 Extra dependencies 14

2.4.7 System Flow 14

Chapter 3: Face Detection 16

3.1. OpenCV sample: Face detection 17

Chapter 4: Face Recognition Concepts 20

4.1. Image preprocessing 21

4.1.1. Histogram equalization 21

4.1.2. Local Binary Pattern 22

4.1.3. Tan Triggs preprocessing 23

4.2. Model 23

4.2.1. Eigenfaces 23

4.2.2. Fisherfaces 25

4.3. Classifier 26

4.4.1. Threshold 27

4.4.2. Validation 28

Chapter 5: Implementation Face Recognition 29
5.1. Step 1: Getting the data right 29
5.2. Step 2: Read the Data 30
5.3. Step 3: Making a model 30
5...3.1 Step 3.1: Reading Images and creating model 30
5.3.2. Step 3.2 how the model is created 31
5.4. Step 4: Recognition 31
5.5. Step 5: Classification and threshold 32
Chapter 6: Communication between Client and Server 33
6.1 Creating a model (Eigenfaces) via cell phone 33
6.2 Recognize 35
Chapter 7: Testing and results 36
7.1. Functional Testing 36
7.1.1. Test case 1: Capture button works properly 36
7.1.2. Test case 2: Recognize button works properly 37
7.1.3. Test case 3: Validation on Recognize Screen 37
7.1.4. Test case 4: Working of Train button 38
7.1.5. Test case 5: Validation on Train button 39
7.1.6. Test case 6: Validation on Train button 39
7.1.7. Test case 7: Suggestion for duplicate names on train page 40
7.1.8. Test case 8: Recognition result 40
7.1.9. Test case 9: Recognition result 41
7.1.10. Test case 10: Start server from Application 42
7.2. Recognition Testing 42
7.2.1. Test 1: With same one of the picture 43
7.2.2. Test 2: With different angle but same person 43
7.2.3. Test 3: Same person with different picture with smile 44
7.2.4. Test 4: With an image that is not a face 44
7.3. Conclusion: 44
Chapter 8: Installation and Configuration 45
8.1. Step1: Configuring of Android application Development 45
8.2. Step 2: Configure OpenCV SDK to Android 45

OpenCV SDK for Android can be downloaded from here. 45
8.3. Step 3: Import OpenCV projects to Eclipse 45
8.4. Step 4: Installing Python and OpenCV for Web server 46
8.5. Step 5: Installing dependencies 47
8.6. Step 6: Installing PHP and MySQL 48
8.7. Step 7: Making web server available to Client 48
Chapter 9: Future work 49
Appendix 52
Android application code 52
Layout files for the user interface of the application 52
Java code files for the Android application 54
Manifest.xml 63
PHP Web application code 65
Python programs that performs facer recognition 77

Table of Figures

Figure 1: Fast Access Android application 11
Figure 2: Face Recognition with OpenCV Android application 12
Figure 3 : Flow diagram 15
Figure 4: Cascade Classifier (Cascade Classification) 17
Figure 5: Real time face detection 18
Figure 6: Application interface showing capture button 19
Figure 7: Face detection 19
Figure 8: Saved cropped faces 19
Figure 9: Histogram Example OpenCV (OpenCV) 21
Figure 10: After histogram equalization (OpenCV) 21
Figure 11: Local Binary Pattern (Face Recognition with OpenCV) 22
Figure 12: Local Ternary Pattern 23
Figure 13: Eigenfaces example 25
Figure 14: Fisherfaces example 26
Figure 15: Classifier method 27
Figure 16: Validation result 28
Figure 17: Server.py 31
Figure 18: Showing model file 31
Figure 19: Client.py 31
Figure 20: Train Screen 33
Figure 21: Recognition screen 35
Figure 22: Recognition Result 35
Figure 23: Test case 1: Capturing a face 37
Figure 24: Recognition Screen 37
Figure 25: Validation on Recognize screen 38
Figure 26: Train Screen 38
Figure 27: Train screen validation 39
Figure 28: Validation on Train page 39
Figure 29: Suggestion for duplicate name 40
Figure 30: Recognition result 41
Figure 31: Recognition result of unknown image 41
Figure 32: Starting a server from application 42
Figure 33: Test images to create Eigenfaces 42
Figure 34: Recognition prediction with same picture 43
Figure 35: Recognition prediction with different angle 43
Figure 36: Recognition prediction with smile 44
Figure 37: Recognition prediction with not a face 44
Figure 38: Importing OpenCV to Eclipse 46
Figure 39: Build path 46
Figure 40: PIP screen 47

ABSTRACT

The goal of this project is to develop an Android application to perform face recognition that combines a mobile platform with a database server tasked with the management of the face images.

This project implements a client-server mechanism where a cell phone running the Android operating system works as a client and a PHP web application works as a web server. The client takes a picture of a person and sends the picture to web server to perform either face recognition or face training. The web server accepts the image and performs the required tasks by the client and sends the response back to the client. The aim of the project is to build a system which can be both storage and location independent. The system uses a cell phone to take pictures. Therefore, it is not location dependent. In addition, it uses a web server to store images. Thus, there will be no storage dependency. This project can be beneficial to educational as well as law enforcement agencies with a need to identify or authenticate people. This project is divided into two major parts, face detection and face recognition.

Face detection is defined as the identification of the human face from a given digital image. It is the first and vital part of the project. In this project, an Android application (client) performs the face detection. The application has an interface to take person's picture. Then the application will automatically detect the person's face and save that face to the cell phone's internal storage.

Face recognition is the process of matching a human query image to ones from a dataset of digital human images. Face recognition employs various pattern recognition algorithms such as Eigenfaces and Fisherfaces. The storage of the images and the face recognition step are done by a web server.

This project uses the Android application development for face detection, Python 2.7 for face recognition and PHP-MySQL to perform database and web development tasks. It also employs open source libraries such as OpenCV.

Chapter 1: Introduction

Face recognition is an easy task for humans. Since early childhood, we have been trained to use faces as a means of expressing our emotions, and as a way to identify people. However, for a computer to do the face recognition is not easy. Simple perturbations to the facial characteristics (such as observing the face from various angles, different illuminations, make-up, glasses) created challenges to computer based algorithms designed for face recognition only a decade ago. Yet, recently, there has been a lot of research and efforts made to perform face recognition with computers, and some results provide today an extremely high accuracy. For example, a recently AI research team of Facebook has published a paper introducing a technique called *DeepFace* to perform face recognition. It has a very high accuracy. "Our method reaches an accuracy of 97.35% on the Labeled Faces in the Wild (LFW) dataset, reducing the error of the current state of the art by more than 27%, closely approaching human-level performance" (Taigman, Yang, Ranzato, & Wolf, 2014)

Face recognition system can be divided into two parts. First, to locate a human face from a digital image called face detection. Second, the face characteristics are used to perform a match against ones stored in a database (this step constitutes the proper face recognition).

The characteristics or features used vary greatly. One of the simplest ways is to compute distances between iris position, nose, or ears. This method was first introduced by Takeo Kanade in 1973 in his paper. (Kande, 1973) However, it is very hard to measure correct distances and marker point. Therefore, there are other options to look for like Eigenfaces or Fisherfaces.

Eigenfaces are a set of eigenvectors to perform face recognition. It was introduced by Matthew Turk and Alex Pentland in 1991. The basic idea of this technique is to create a set of images and treat each image as a vector. Then we compute a mean from given images. Then we find the difference between the mean and the query image. So we can find which vector is close enough to the query image and we can recognize the query image.

Fisherfaces is also another way to perform face recognition. It can be thought of an extension to Eigenfaces. Eigenfaces has a limitation on various external sources like lighting, shadowing while Fisherfaces can eliminate these problems. It uses class specific linear position technique. It was first introduced by Peter Belhumeur, Joao~ P. Hespanha, and David J. Kriegman on July 1997. It uses linear discriminant analysis that performs a class-specific dimensionality invented by Sir R.A. Fisher. Therefore, it is called Fisherfaces.

This project uses Eigenfaces, Fisherfaces and a classifier to perform face recognition. It combines cell phone's location independence and the ability to store large data set of a web server.

This document is divided into seven chapters.

Chapter 1 introduces the overview, motivation and background information of the project

Chapter 2 states the functional requirements, system requirements, flow of the system.

Chapter 3 explains face detection in detail. It also provides code and methods to perform face detection.

Chapter 4 explains face recognition concepts in details. It gives basic understanding of face recognition algorithms and implementation.

Chapter 5 explains how the face recognition part of the project is set up.

Chapter 6 explains the communication between client and server via PHP

Chapter 7 states various test scenarios and results.

Chapter 8 states the installation and configuration procedure of the project.

Chapter 9 provides future work.

1.1. Problem and Motivation

In computer vision, face recognition has been a vast area for research and development. The face recognition process can be described as computer software that can identify a given person's image from previously stored dataset of digital images using various algorithms like pattern recognition or feature extraction.

Face recognition can be done by either using traditional facial recognition algorithms that identify the facial features of the person's image or a 3-dimensional recognition, which is fairly new and uses 3D sensors to capture the shape of the face. Facial recognition algorithms include Eigenfaces, Fisherfaces, Scale-invariant feature transform (SIFT), Speeded up Robust Features (SURF).

Facial recognition can have high accuracy under the best circumstances. These circumstances include appropriate light, having an appropriate face image like images similar to government id, database of similar images to compare and a perfect camera angle. However, wearing sunglasses or having a beard can result in undesired results.

Traditionally, the face recognition process needs a camera to capture images and a computer to process the image and recognize faces. The problem with this approach is that it works at a fixed location. To avoid this location dependency, with a decrease in price with smart phones and the increase in computational features in recent years, smart phones can be the best option to perform face recognition. In addition smart phones can be very useful to law enforcement department. For example, a police officer can identify a person by just taking a picture of him on the spot instead of taking him to the police station and identify him.

However, there are some problems to perform face recognition on smart phones as well. Firstly, accurate recognition requires a database of faces. Since smart phones are limited in memory, storing images on phone memory could lead to storage problem. Secondly, the computational speed of the smartphone is also limited. Therefore, recognition process could result in a long wait for the user. Finally, there are very few libraries available on smart phone platforms to implement facial recognition algorithms. Therefore, a developer may not have the liberty to use the most up to date libraries. Thus, combining powerful features of a computer and location independence of smart phone can be the solution to the given problem.

1.2. Development of the System

The following is a list of functionality that was proposed to be implemented through the application.

A. Face Detection: Face detection is the first and important step for the face recognition. The application will use cell phone's camera to detect face of the person. The application will use built in libraries from OpenCV that provides support for face detection. After detection of the face, application will save the face.

B. Face Recognition: Face recognition will run on the web server. This project will use the OpenCV library to implement famous face recognition algorithms. Android application has an interface to perform select image and send it to web server to recognize.

C. Communication with Web server: After saving the image, the application will make a web request to the web server to send saved image. A web server will perform face recognition and send response to the device.

1.3 Comparison with other solutions

There has been some work done in the area of face recognition with smart phones. In particular, with Android, there are some applications which perform face recognition. However, the scope of those applications is limited to unlock the device. In addition, they perform recognition on the phone. For example,

1. Face Recognition-Fast Access provides lock-unlock device facility. (Face Recogniton Fast Acess)

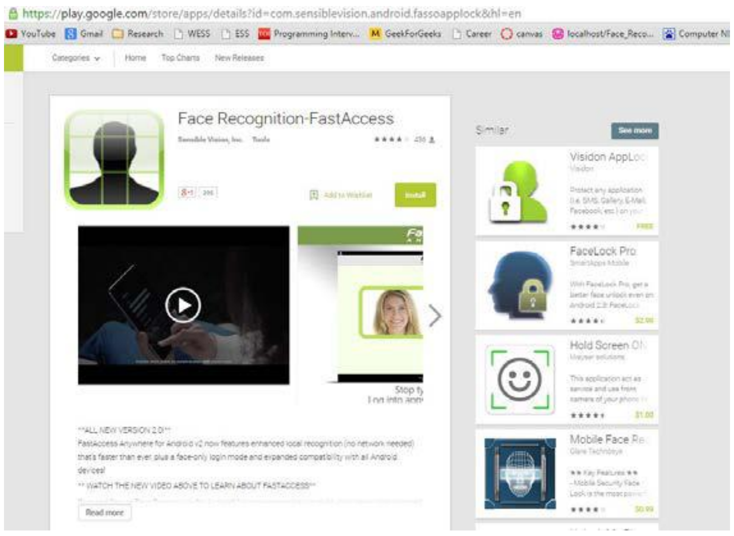
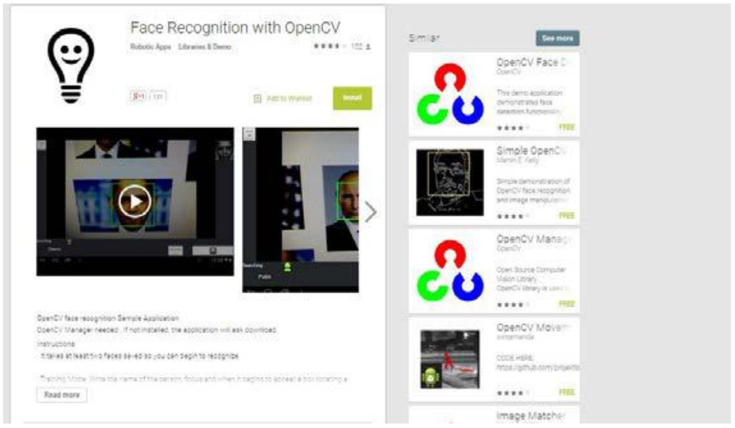

https://play.google.com/store/apps/details?id=com.sensiblevision.android.fassoapplock&hl=en
Face Recognition-FastAccess
Install
Similar
See more
Visidon AppLoc
FaceLock Pro
Hold Screen On
Mobile Face Re
Read more

Figure 1: Fast Access Android application

This application does the face recognition but it is limited to unlock the device. In addition, it also does not support multiple users.

2. Face Recognition with OpenCV – does the face recognition on local platform. (Face Recognition with OpenCV)



OpenCV face recognition Sample Application
OpenCV Manager needed . If not installed, the application will ask download.
Instructions
It takes at least two faces saved so you can begin to recognize

Figure 2: Face Recognition with OpenCV Android application

This application does the face recognition but the recognition is done on a phone. Therefore, there is a limitation of storage.

There are many other solutions that provide face recognition. For example, Face++ provides an API to perform face detection or face recognition. “Face++ services have basic and enterprise versions. Basic Face++ service is an API-based and 100% free. Enterprise version has a much better accuracy and system performance.” (Face++ - Tech Service)

Another concern with the apps tested is security and privacy. The documentation included with the apps does not provide information about the storage location of photos. Therefore, if an organization wants to use their service, privacy might be the concern. The other concern, it is not free for the enterprise use. It is only free for the basic service.

In comparison with these solutions, this application provides client server privilege with the ability to location and storage independence. In addition, since the data are stored on the user’s computer, privacy concerns can be addressed.

Chapter 2: Functional & Non Functional Requirements

2.1. Functional Requirements

This application is developed for an educational or law enforcement organization to identify or authenticate people. The system needs one administrator to store and train images in the web server. However, this project also has a facility to store and train images via cell phone. Therefore, anyone with an Android device and this application can be the administrator and perform training or recognition. The System flow is explained in the later part of the chapter.

For the user prospective, the system will need following three attributes from them.

- First Name
- Last Name
- Images

An additional attribute, a path where images are stored is generated automatically by the system.

- Path to Images

All processes are automated including training. However, for the maintenance administrator will have following responsibilities.

- Update the user table in database
- Validate the image path
- Check the server status

2.2. Non Functional Requirements

- This system can be accessible anywhere and anytime with proper server connection.
- The user should use latest Android operating system no older than 4.1 in order to run this application

2.3 System Requirements

This system is written assuming the following hardware and software environment requirements.

2.3.1 Operating System

- Microsoft Windows 7 or higher
- Memory: 1 GB (Minimum)
- Storage: Code requires minimum of 2GB,
    - as per requirement for image storage

2.3.2 Android Requirements

- Android Operating System 4.1 or higher
- OpenCV manager application
- Camera with good picture quality

2.3.3 Web Server

- Apache 2.4.9
- Flask server for python

2.4.4 Technology

This application has two parts. Each is created using different technologies.

1. Face detection
    a. Android app development
    b. Java
    c. OpenCV library 2.4.9
2. Face Recognition
    a. Python 2.7.9
    b. PHP 5.5.11
    c. OpenCV library

2.4.5 Database

- MySQL 5.6.17

2.4.6 Extra dependencies

- Microsoft Excel 2010
- Windows Task Scheduler
- Ngrok : To provide access of a computer via internet to the cell phone (Optional)

2.4.7 System Flow

The flow of the system is stated in below (Fig 3).

Step 1: The user takes a picture of a person.

Step 2: The application will ask user to what to do next. Does user want to perform face recognition or face training?

Step 2.1: If user wants to recognize, the application will take the user to recognition screen.

Step 2.1.1: The application will ask the user to select the desire image.

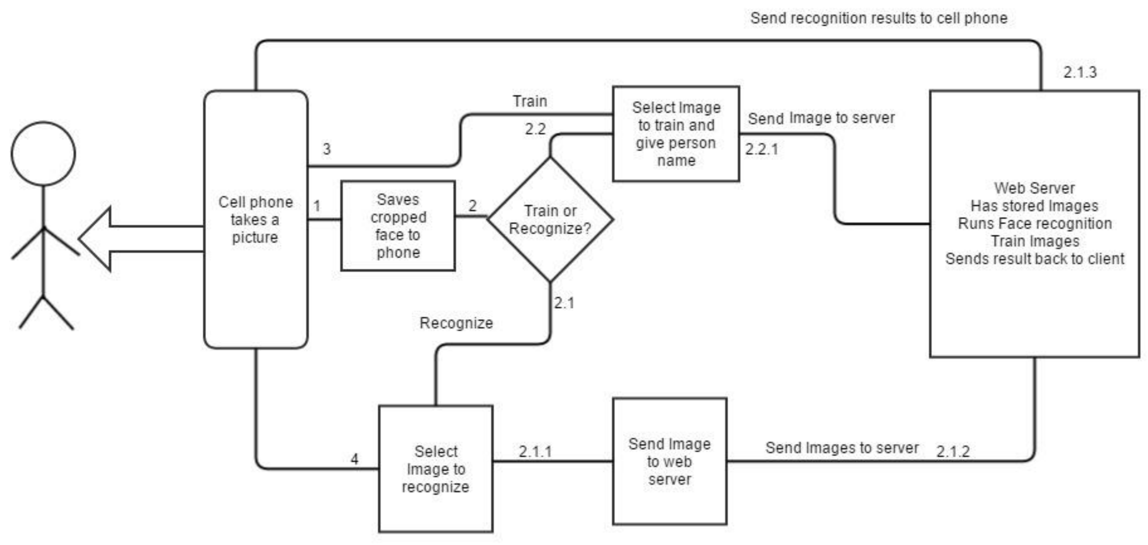

Send recognition results to cell phone
2.1.3
Train
2.2
3
Select Image to train and give person name
Send Image to server
2.2.1
Cell phone takes a picture
1
Saves cropped face to phone
2
Train or Recognize?
Web Server Has stored Images Runs Face recognition Train Images Sends result back to client
2.1
Recognize
4
Select Image to recognize
2.1.1
Send Image to web server
Send Images to server
2.1.2

Figure 3 : Flow diagram

Step 2.1.2: The application will send image to web server to perform recognition

Step 2.1.3: A web server will perform face recognition and send results back to cell phone.

Step 2.2: If user selects Train, app will open screen to train images and ask user to select image and give the image person name

Step 2.2.1: The application will send images and other related data to web server. Then web server will train that image for the particular person.

Step 3: It is an individual step to perform face training. User can train stored images.

Step 4: It is an individual step to perform face recognition. User can recognize stored images.

Chapter 3: Face Detection

Face detection is a very important aspect of the face recognition system. Face detection can be done in two ways. One is by detecting the face from an image. The other is to detect the face when you focus the camera to a person and save the cropped face to the memory. This application uses OpenCV libraries to perform face detection by the second method.

OpenCV is an open source library for pattern recognition and machine learning. It supports various algorithms like Eigenfaces, Fisherfaces, Principal Component Analysis (PCA), etc. It provides an implementation in Android, Python, C++, and JAVA. It is released under BSD license. Therefore, it is free for commercial or academic uses. It can be found at www.opencv.org.

OpenCV also provides support for mobile application development. It provides Standard Development Kit (SDK) for Android. The installation and configuration of the OpenCV Android SDK are explained in a later chapter of the document.

OpenCV uses Haar Feature-based Cascade Classifier for Object Detection. In that method a classifier is trained with samples of a particular object like a face or table. Those samples are called positive examples and scaled all samples to all sizes. The negative examples are made the same way. Once this classifier is trained, it can be applied to real world examples. A classifier gives 1 as a result, if it finds positive example and 0 for the negative example. “To search for the object in the whole image, one can move the search window across the image and check every location using the classifier. The classifier is designed so that it can be easily resized in order to be able to find the objects of interest in different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales.” (Cascade Classification)

Haar like features are the input for the basic classifier. The following diagram in Fig 4 explains the concept.

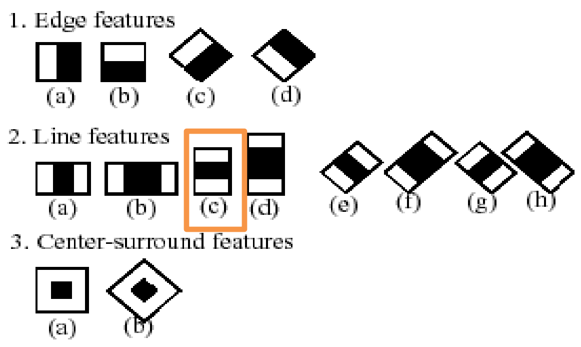


Figure 4: Cascade Classifier (Cascade Classification)

After a classifier is trained, it can be applied to a region of interest in an input image. In the case of the third line feature (2c) the response is calculated as the difference between the sum of image pixels under the rectangle covering the whole feature and the sum of the image pixels under the black stripe multiplied by 3 in order to compensate for the differences in the size of areas.

OpenCV implements this cascade classifier technique to Android Standard Development Kit (SDK) to support face detection. OpenCV SDK comes with CascadeClassifier.java file that has an implementation of the classifier. In addition, it provides sample face detection application. This project has used this face detection application and modified as per use of the application.

3.1. OpenCV sample: Face detection

This sample application comes with a feature that opens a camera and when the user focuses the camera to a person, it displays the green square shape around the face of a person. It works like a real time face detection system.

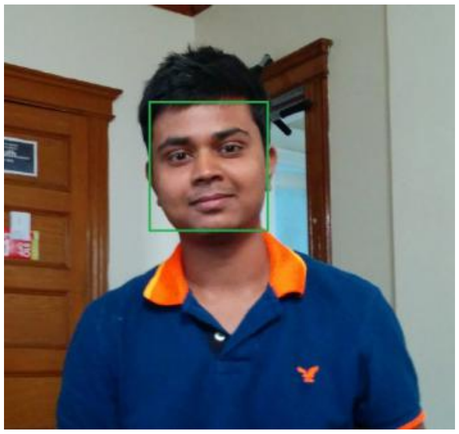


Figure 5: Real time face detection

The problem with the sample application is there was no mechanism to save the cropped face. Because this application worked as real time face detection, it did not have an interface or mechanism to save the cropped face.

To satisfy the need of the application, a method to crop and save the cropped face was implemented. The green square around the face is actually a matrix. One can write a program, to calculate the columns and rows of this matrix and save the region around face to the face.

The following code snippet explains the calculation of matrix.

```
Core.rectangle(mRgba, facesArray[i].tl(), facesArray[i].br(),
FACE_RECT_COLOR, 4);
Mat uncropped = mGray;
Rect roi = new Rect(facesArray[i].tl(), facesArray[i].br());
mFace = new Mat(uncropped, roi);
```

Here, **facesArray[i].t1()** is a value of columns and **facesArray[i].br()** is a value of rows of a square. Mat is an object of a Matrix class. As you can see we can create a new Matrix from values of columns and rows. Once we create a matrix, it is easy to covert that matrix into Bitmap and save to the internal storage. The following code snippet illustrates the method that saves the cropped face.

```
Bitmap bitmap = Bitmap.createBitmap(mFace.cols(), mFace.rows(),
Bitmap.Config.ARGB_8888);

Utils.matToBitmap(mFace, bitmap);
bitmap = Bitmap.createScaledBitmap(bitmap, 128, 128, false);
String root = Environment.getExternalStorageDirectory().toString();
            File myDir = new File(root + "/saved_images");
            myDir.mkdirs();
            Random generator = new Random();
            int n = 1000;
            n = generator.nextInt(n);
            String fname = "Image-"+ n +".jpg";
            File file = new File (myDir, fname);
            if (file.exists ()) file.delete ();
            try {
                  FileOutputStream out = new FileOutputStream(file);
                  bitmap.compress(Bitmap.CompressFormat.JPEG, 90, out);
                  out.flush();
                  out.close();

            } catch (Exception e) {
                  e.printStackTrace();
            }
```

Face Recognition with Android Device

This method saves the Image to gray scale and 128 by 128 resolutions. It saves images to **Saved_Images** named folder in internal storage.

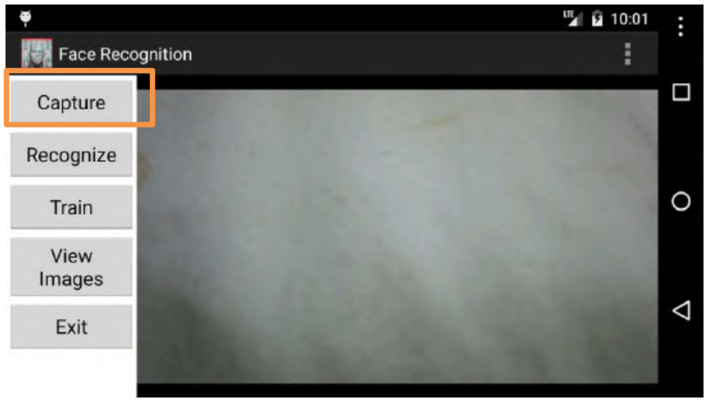


Figure 6: Application interface showing capture button

As you can see in Fig 6, this application has a button called "Capture" to save the face in the internal storage of a cell phone.

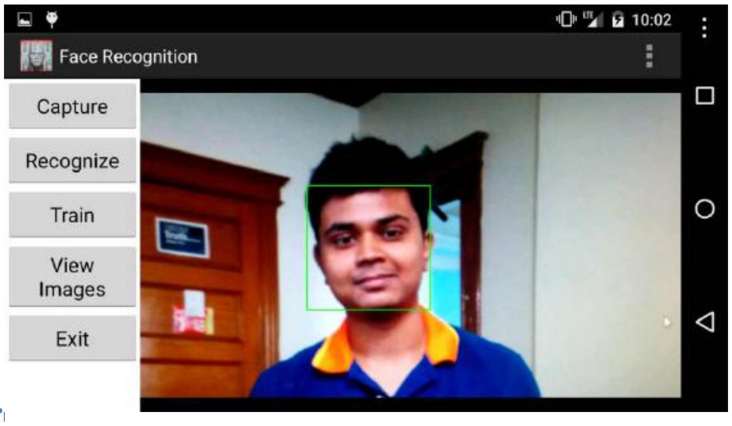


Figure 7: Face detection

After clicking Capture button it saves face with grey scale as shown in following figure.

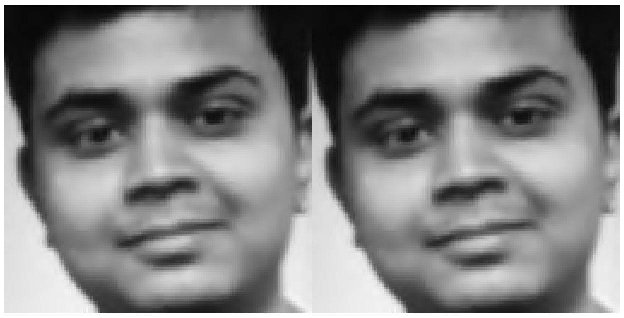


Figure 8: Saved cropped faces

Chapter 4: Face Recognition Concepts

Face recognition has been the subject of research for many years. To make a machine learn and able to recognize object or a face is not so easy. However, there have been much advancement in the area of computer vision and it allows us to perform face recognition with a certain level of accuracy. This document has already introduced algorithms like Eigenfaces or simple approach to recognize face like distance calculation.

OpenCV is an open source free library which provides implementations for both Eigenfaces and Fisherfaces. This project uses a combination of feature extraction and nearest available neighbor classifier to perform face recognition. The code is written in Python. A general overview of the face recognition part is as follows. Each part is explained after the overview.

Image Preprocessing:

To perform face recognition, it is essential to preprocess the images that will be used. The following approach was used for image preprocessing.

- Histogram Equalization
- Local Binary Patterns
- Tan Triggs Preprocessing

Model: This project uses the following techniques to create model from given dataset of images.

- Eigenfaces
- Fisherfaces
- Local Binary Pattern Histograms

Classifier: A classifier is used for predicting the nearest neighbor for the given image to recognize.

K-Nearest Neighbor

Cross Validation:

Validation of our model is important. We use the following technique to validate.

- k-fold Cross Validation

4.1. Image preprocessing

4.1.1. Histogram equalization

Image Histogram: An image histogram is a graphical representation of image's intensity. Therefore, histogram equalization improves the contrast of an image if we need to stretch the intensity. Consider following example,

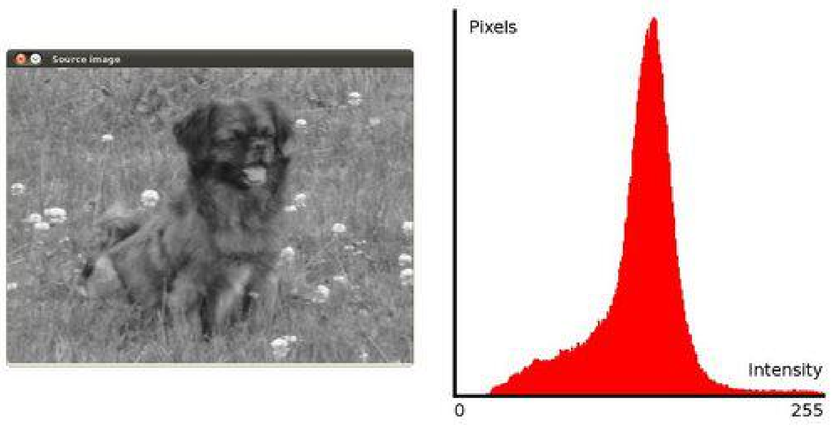


Figure 9: Histogram Example OpenCV (OpenCV)

One can see in the picture that pixels seem clustered around the middle of the available range of intensities. With the help of a Histogram equalization method one can stretch this intensity as per our need. So if this method applied to the above example, we get something like this

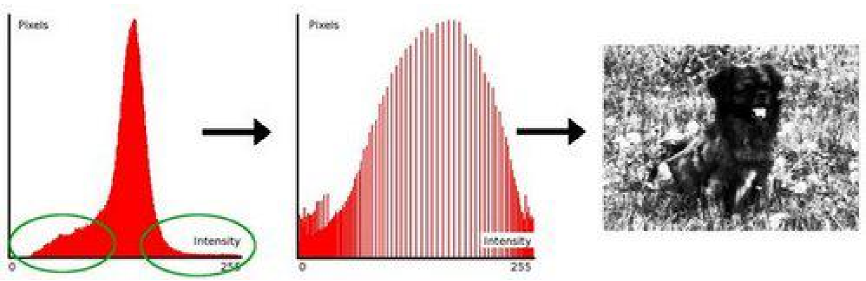


Figure 10: After histogram equalization (OpenCV)

After we apply the histogram equalization method, the intensity in this image is distributed equally. It is very important for this project to equalize the intensity of images to perform the face recognition.

OpenCV provides code and support to apply this method to our images. Therefore, we do not have to worry about algorithmic explanation. We can simply implement the method. Full implementation can be found here. [http://docs.opencv.org/doc/tutorials/imgproc/histograms/histogram_equalization/histogram_equalization.html]

4.1.2. Local Binary Pattern

As previously stated, this project uses a combination of Eigenfaces or Fisherfaces to train the model in order to perform face recognition. According to OpenCV documentation, the problem with Eigenfaces or Fisherfaces is that they treat data as a vector in high dimensional Image space. Therefore, variance in external sources can be the factor that affects prediction. The variance like lightening conditions or image angle or a shadow in the image can be the factor.

For example, we need very good lighting condition or 10 images of each person to get accuracy but we cannot promise that every time we train our model.

The local binary pattern works as a system where each pixel of an image is compared to its next neighbors. Therefore, we can think pixel as a center threshold it against its neighbor. Then calculate if the intensity of the center pixel is greater or equal than the neighbor pixel then mark the center pixel as 1 or 0. If you do like above, you will have a binary string like 1011101. A scenario is shown in the following example.

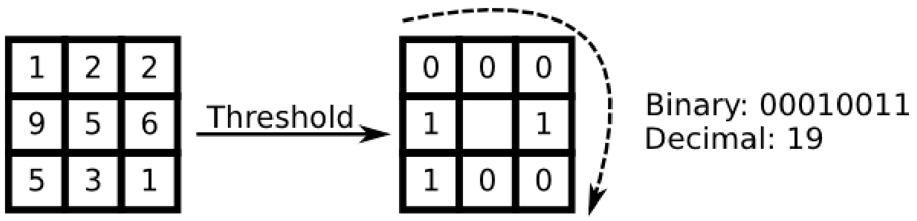


Figure 11: Local Binary Pattern (Face Recognition with OpenCV)

4.1.3. Tan Triggs preprocessing

As discussed earlier, it is important to preprocess the images for face recognition. To make the model strong and get maximum accuracy, there is a need to implement this technique too. The technique was introduced by Xiaoyang Tan and Bill Triggs in their paper. (Tan & Triggs, 2010)

It is basically a technique that enhances the performance of Local Binary Pattern and makes Local Ternary Patterns.

“LBP’s are resistant to lighting effects in the sense that they are invariant to monotonic gray-level transformations, and they have been shown to have high discriminative power for texture classification”. (Tan & Triggs, 2010)

So we extend LBP to three values code and make Local Ternary pattern (LTP). LTP does not threshold pixel to 0 or 1 like LBP. It uses a threshold constant to threshold pixel into three values. It can be computed as follows.

Consider K as a threshold constant; C is the value of the center pixel and a neighbor pixel P, so the threshold is:

1 if $P > C + K$

0 if $P < C – K$ and $P < C + K$

-1 if $P < C – K$

Figure 12: Local Ternary Pattern

So each threshold pixel has one of the three values.

4.2. Model

4.2.1. Eigenfaces

Eigenfaces are a technique to perform face recognition. It was first introduced by Matthew Turk and Alex Pentland in 1991. (Turk & Pentland, 1991) The basic idea of Eigenfaces is to treat the face recognition problem in the 2D recognition problem rather than 3D problem requiring geometry. We can define Eigenfaces as follow.

The basic idea of this technique is to create a set of images and treat each image as a vector. Then we compute a mean from given images. Then we find the difference between the mean and the query image. So we can find which vector is close enough to the query image and we can recognize the query image.

So basically, we take pictures of a person and take information mostly features and make a model. When we get a new image we compare those features and predict the results. The algorithm can be explained as follows.

1. We have N face images with size of (h * W), we put them into a set

   {x1, x2,…………., xn}

2. Then we compute the mean

   $$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \text{………………………………………………………………………………………………. (1)}$$

3. We can define Covariance Matrix S

   $$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^{T} \quad \text{……………………………………………………………… (2)}$$

4. Compute the eigenvalues λ_i and eigenvectors Vi of S

   $$Sv_i = \lambda_i v_i, i = 1, 2, \ldots, n \quad \text{…………………………………………………………………… (3)}$$

5. Order the eigenvectors descending by their eigenvalue. The k principal components are the eigenvectors corresponding to the k largest eigenvalues.

Principal Component Analysis (PCA) is a technique used to emphasize on a variation and bring out strong patterns in a dataset. It's often used to make data easy to explore and visualize.

To perform the face recognition using Eigenfaces, the following steps are taken.

- Project all trained images into a PCA subspace
- Project image that needs to recognize into a PCA subspace
- Find nearest neighbor between trained image and an image needs to recognize. The nearest neighbor will be the prediction.

Our project uses Eigenfaces approach to create model from trained images. Figure 13 provides the Eigenfaces generated from our project model.

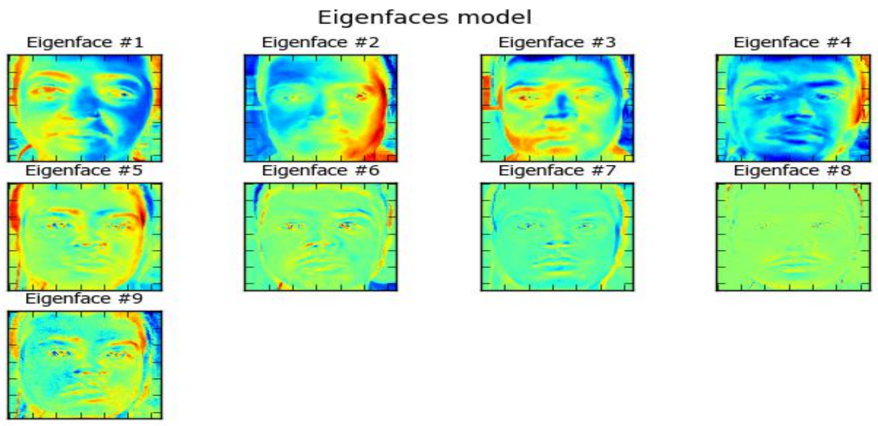


Figure 13: Eigenfaces example

OpenCV provides source code to generate Eigenfaces. The above picture is generated using that source code.

4.2.2. Fisherfaces

Fisherfaces is also another way to perform face recognition. It can be thought of an extension to Eigenfaces. Eigenfaces has a limitation on various external sources like lighting, shadowing while Fisherfaces can eliminate these problems. It uses class specific linear position technique. It was first introduced by Peter Belhumeur, Joao~ P. Hespanha, and David J. Kriegman on July 1997. It uses linear discriminant analysis that performs a class-specific dimensionality invented by Sir R.A. Fisher. Therefore, it is called Fisherfaces.

In Fisherfaces, “In order to find the combination of features that separates best between classes the Linear Discriminant Analysis maximizes the ratio of between-classes to withinclasses 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." (Face Recognition with OpenCV)

The Implementation is available for Fisherfaces on OpenCV. I have created my own Fisherfaces model and looks as follows.

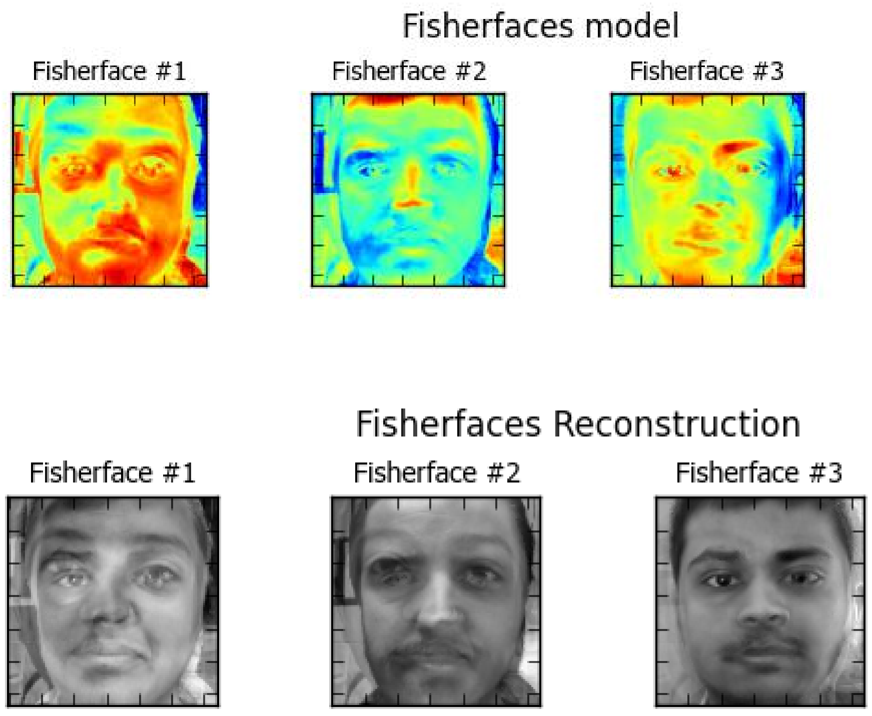


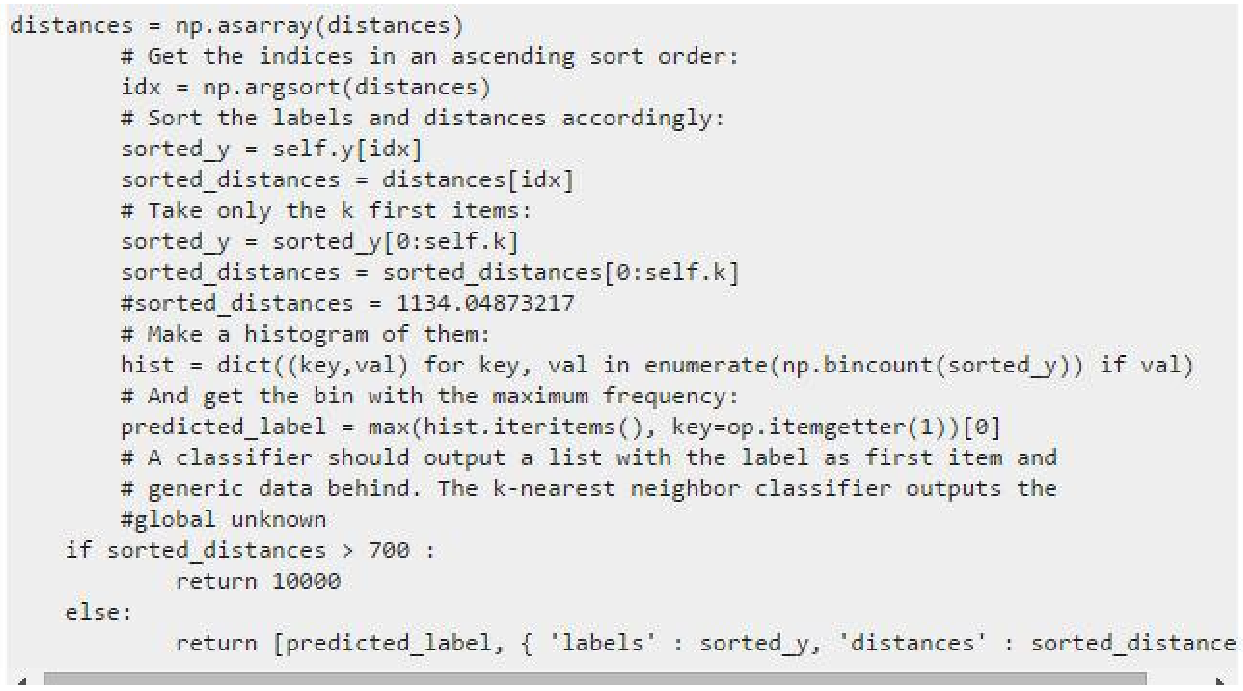
Figure 14: Fisherfaces example

4.3. Classifier

This project uses K nearest neighbor classification method to predict the recognition result. "The algorithm caches all training samples and predicts the response for a new sample by analyzing a certain number (*K*) of the nearest neighbors of the sample using voting, calculating a weighted sum, and so on. The method is sometimes referred to as "learning by example" because for prediction it looks for the feature vector with a known response that is closest to the given vector." (*K* nearest neighbor)

To measure distances we use Euclidian distance.

We calculate distances from all neighbors then we sort them into ascending order and we return the nearest neighbor as the prediction. The code snippet is given below.



```
distances = np.asarray(distances)
        # Get the indices in an ascending sort order:
        idx = np.argsort(distances)
        # Sort the labels and distances accordingly:
        sorted_y = self.y[idx]
        sorted_distances = distances[idx]
        # Take only the k first items:
        sorted_y = sorted_y[0:self.k]
        sorted_distances = sorted_distances[0:self.k]
        #sorted_distances = 1134.04873217
        # Make a histogram of them:
        hist = dict((key,val) for key, val in enumerate(np.bincount(sorted_y)) if val)
        # And get the bin with the maximum frequency:
        predicted_label = max(hist.iteritems(), key=op.itemgetter(1))[0]
        # A classifier should output a list with the label as first item and
        # generic data behind. The k-nearest neighbor classifier outputs the
        #global unknown
    if sorted_distances > 700 :
            return 10000
    else:
            return [predicted_label, { 'labels' : sorted_y, 'distances' : sorted_distance
```

Figure 15: Classifier method

Figure 15 shows the classifier method used. As we can see, the labels are sorted on the ascending order of the distance.

Sorted_distances = distances[idx]

At the end method returns the label (prediction) associated with the sorted distanced neighbors.

return [predicted_label, { 'labels' : sorted_y, 'distances' : sorted_distances }]

4.4.1. Threshold

It is very important to get accurate results in any system. Therefore, setting up a threshold is very important. As you can see from above code, I have set up the threshold to distance at 700. For a threshold above 700, the result is no longer acceptable. The following code does the threshold setting for us.

if sorted_distances > 700 :

```
        return 10000
else:
        return [predicted_label, { 'labels' : sorted_y, 'distances' : sorted_distances }]
```

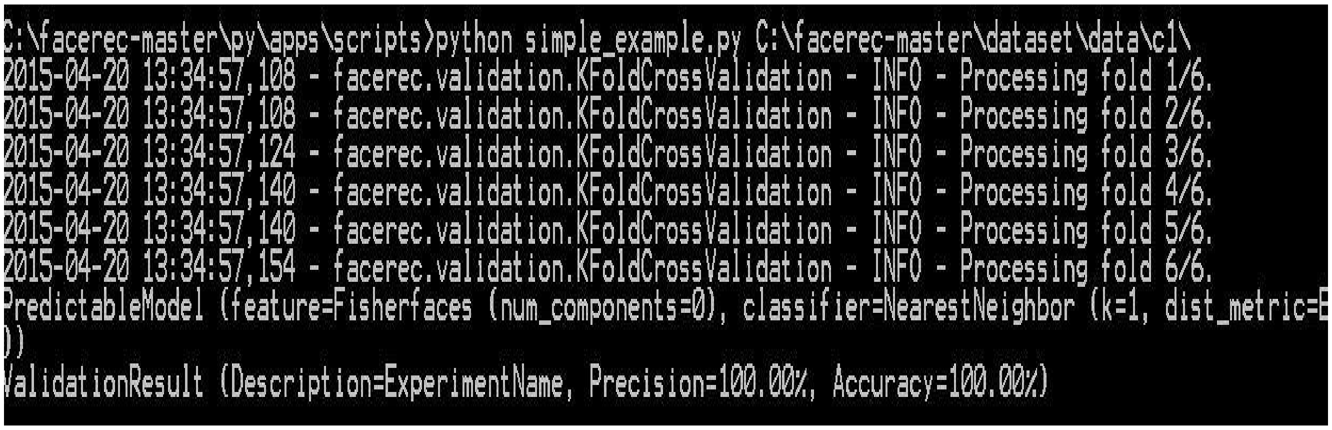
4.4.2. Validation

It is very important to validate our model. To get accurate results we must validate our model. Our project uses *k*-fold cross validation technique to validate its model.

k-fold cross validation: It is a technique to estimate performance of a classifier. It goes as follows.

1. Arrange all examples into random manner
2. Divide them into *k* folds
3. For all examples do following
   a. Train all examples using the same classifier
   b. Test all of them
   c. Compute examples that were wrongly classified
4. Return accuracy.
5. Run this method several times and you can validate your model and measure accuracy.

If we perform this *k*-fold cross validation method to AT & T test data base we can get roughly 95% accuracy.



```
C:\facerec-master\py\apps\scripts>python simple_example.py C:\facerec-master\dataset\data\c1\
2015-04-20 13:34:57,108 - facerec.validation.KFoldCrossValidation - INFO - Processing fold 1/6.
2015-04-20 13:34:57,108 - facerec.validation.KFoldCrossValidation - INFO - Processing fold 2/6.
2015-04-20 13:34:57,124 - facerec.validation.KFoldCrossValidation - INFO - Processing fold 3/6.
2015-04-20 13:34:57,140 - facerec.validation.KFoldCrossValidation - INFO - Processing fold 4/6.
2015-04-20 13:34:57,140 - facerec.validation.KFoldCrossValidation - INFO - Processing fold 5/6.
2015-04-20 13:34:57,154 - facerec.validation.KFoldCrossValidation - INFO - Processing fold 6/6.
PredictableModel (feature=Fisherfaces (num_components=0), classifier=NearestNeighbor (k=1, dist_metric=E
))
ValidationResult (Description=ExperimentName, Precision=100.00%, Accuracy=100.00%)
```

Figure 16: Validation result

I performed validation on my trained images and I got 100.00% accuracy in my model.

Chapter 5: Implementation Face Recognition

The above has explained the basic concept of face recognition as well as the libraries that are used in this project. This chapter will introduce the user how the face recognition part of the project is set up and how it's working.

First of all, our recognition part has some dependencies. We use third party libraries to preprocess images and perform mathematical calculations as we explained in the previous chapter.

Python Imaging Library (PIL): It is a library that adds image processing capabilities to python interpreter.

NumPy: NumPy is the fundamental package for scientific computing with Python. It has powerful N dimensional array object

SciPy: It is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:

Matplotlib: matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

All these libraries are available in windows; UNIX environment and the installation of them are explained in later chapters.

5.1. Step 1: Getting the data right

The structure of the data (image folders) is very important for this project. It uses the folder structure as shown in OpenCV documentation. In our case all the data would be images so we need to arrange images in following manner.

Person1

- Image1
- Image 2
- Image 3

Person2

- Image 1
- Image 2
- Image 3

Person3

- Image 1
- …..

We should make our folder structure as explained above. We can get reference from AT & T database as well which can be found on http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

Note: All images should be gray scaled and in 128 * 128 resolution. As our detection method does it for us we do not have to worry about it.

5.2. Step 2: Read the Data

We have a small python script to get the path of images from a CSV file. So we can make a CSV file containing the path information of all images.

```
jigar_patel;C:/facerec-master/dataset/data/c1/jigar_patel
pritesh_parekh;C:/facerec-
master/dataset/data/c1/pritesh_parekh
raju _vemula;C:/facerec-master/dataset/data/c1/raju _vemula
```

Now a python script will read this CSV file and provide images to the required program.

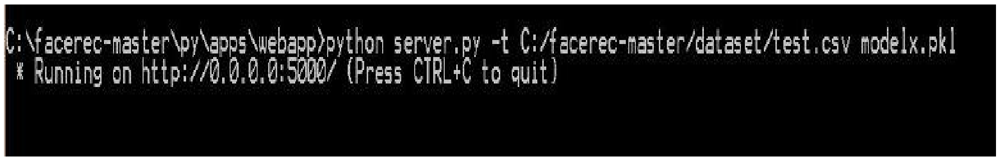
5.3. Step 3: Making a model

As explained in the previous chapter, this project uses a combination of feature extraction and nearest neighbor classification method to predict the result. Therefore, the following steps will explain how the model is actually generated.

5...3.1 Step 3.1: Reading Images and creating model

We have a python script called Server.py which initiates the flask server and does the following tasks.

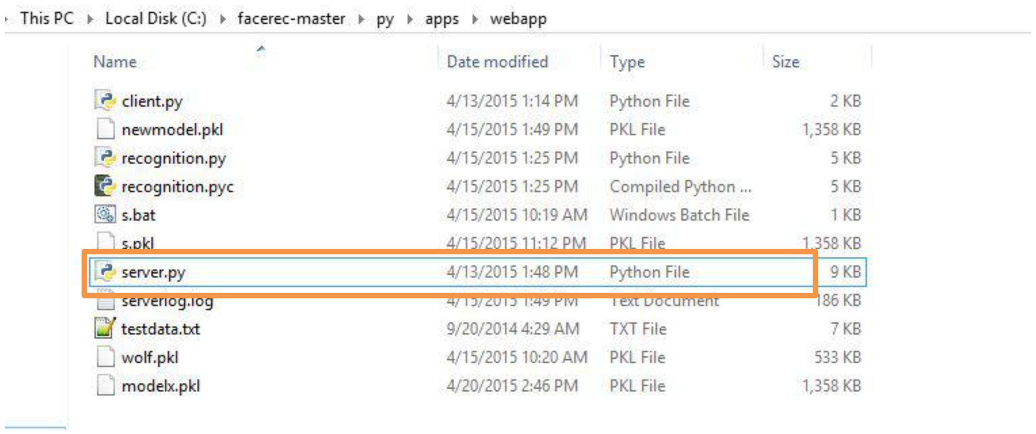
- It reads a CSV file for image paths
- It creates a model_name.pkl file which contains features from given dataset in CSV file whichever model name is given



```
C:\facerec-master\py\apps\webapp>python server.py -t C:/facerec-master/dataset/test.csv modelx.pkl
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```

Figure 17: Server.py

As you can see in the above picture, it takes a path for a CSV file and a model file name (modelx.pkl) as arguments and initiates server.



This PC ▸ Local Disk (C:) ▸ facerec-master ▸ py ▸ apps ▸ webapp

Name	Date modified	Type	Size
client.py	4/13/2015 1:14 PM	Python File	2 KB
newmodel.pkl	4/15/2015 1:49 PM	PKL File	1,358 KB
recognition.py	4/15/2015 1:25 PM	Python File	5 KB
recognition.pyc	4/15/2015 1:25 PM	Compiled Python ...	5 KB
s.bat	4/15/2015 10:19 AM	Windows Batch File	1 KB
s.pkl	4/15/2015 11:12 PM	PKL File	1,358 KB
server.py	4/13/2015 1:48 PM	Python File	9 KB
serverlog.log	4/15/2015 1:49 PM	Text Document	186 KB
testdata.txt	9/20/2014 4:29 AM	TXT File	7 KB
wolf.pkl	4/15/2015 10:20 AM	PKL File	533 KB
modelx.pkl	4/20/2015 2:46 PM	PKL File	1,358 KB

Figure 18: Showing model file

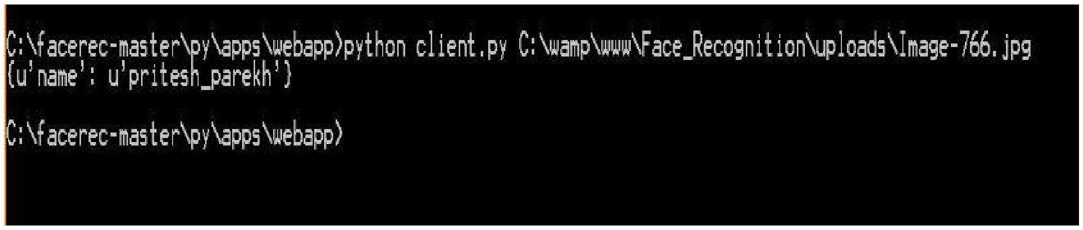
We can see the modelx.pkl file created by the server in our directory.

5.3.2. Step 3.2 how the model is created

When a dataset is given to Server.py, it calls another python file called Model.py to compute the mode. Model.py file has a method compute model from features. The method uses the features calculated from feature extraction algorithms explained above.

5.4. Step 4: Recognition

After the server is started, we know our program is ready to accept query images. So we have a client script to provide query images to a server script. It takes a query image as an argument and provides prediction as a result.



```
C:\facerec-master\py\apps\webapp>python client.py C:\wamp\www\Face_Recognition\uploads\Image-766.jpg
{u'name': u'pritesh_parekh'}

C:\facerec-master\py\apps\webapp>
```

Figure 19: Client.py

As shown in Fig 19, the program takes image path as an argument and return results.

5.5. Step 5: Classification and threshold

So how does prediction works. As explained in the previous chapter it finds the nearest neighbor and return the result. So a threshold can be set to verify results. For example, let's say we have measured that the distance greater than 700 gives us a false result. Therefore, one can set threshold at 700 and say over 700 is an unidentified image. Following code snippet shows the threshold setting for this project.

```
If sorted_distances > 700:
   return "Unknown Image"
else:
   return [predicted_label, {'labels' : sorted_y, 'distances' : sorted_distances }]
```

Here **sorted_distances** is a variable holding distance of the nearest neighbor. Thus, after a bit of testing one can measure that greater than 700 distance is not accurate. Therefore, greater than 700 is Unknown Image and otherwise is a predication.

Chapter 6: Communication between Client and Server

As mentioned earlier, this project implements a client - server mechanism. Therefore, communication between a client and a server is required. This project has two parts.

1. A cell phone that detects faces and sends those faces to server for either recognition or training.
2. A server that accepts images from cell phone and trains them or recognize them and sends back results.

Therefore, this project uses PHP and MySQL to handle client server request responses.

6.1 Creating a model (Eigenfaces) via cell phone

To create a new model (Eigenfaces) images need to be trained as explained in previous chapters. This process needs following three steps.

1. Create a new folder and rename it to person's name whose face we have captured.
2. Upload image to that person's folder
3. Write a path to that folder into a CSV file.
4. Start python server.

To perform the above steps automatically via the cell phone, A PHP program called "**Train.php**" was created. This program takes first name, last name, and the image as arguments and performs the process mentioned above. The third step (path) is automatically generated from the system.

So the Train.php file looks like following when accessed via phone.

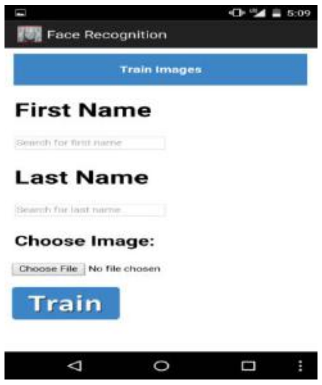


Figure 20: Train Screen

The attributes first name, last name and path are stored in the database. The fourth attribute, image is directly stored in a folder.

The attributes are stored in a table called “Model”. The structure of the table is described below.

Field	**DataType**	**Length**	**Primary**	**AutoIncrement**
ID	Int	100	Yes	Yes
first_name	Varchar	10	NO	NO
last_name	Varchar	10	NO	No
Path	Varchar	100	NO	NO

To avoid duplication of the record, the program will check into the database for the person’s first and the last name. If it finds the match for a person with same name, then it will only upload image to person folder which is already created.

If it does not find the match, then program will do following things.

1. Enter the details into database
2. Create a new folder with person first and last name
3. Upload image to that folder
4. Write a path into CSV file

After successful completion of all steps above, program will ask user to start the server.

It is difficult to start the server automatically via a PHP program because the server has to keep running and generate the output. Thus, if we run a server via a PHP program, the program will keep running. The following describes a workaround.

Step 1: Create a batch file that will start server.

```
c:\python27\python.exe C:/facerec-master/py/apps/webapp/server.py -t C:/facerec-master/dataset/test.csv modelname.pkl %*
```

Step 2: Create a schedule task that runs this batch file.

Step 3: Set trigger to scheduler task on a windows event.

Step 4: Create another batch file to register windows event.

```
eventcreate /ID 1 /L APPLICATION /T INFORMATION  /SO MYEVENTSOURCE /D "My first log"
```

Step 5: Create a PHP program that can run a batch file to register windows event.

Step 6: The batch file with windows even will trigger the task scheduler and the task scheduler will start the server.

6.2 Recognize

To perform face recognition with a PHP file is fairly easier than creating a model. We have a PHP webpage called “**Recognize.php”**. It basically looks like following

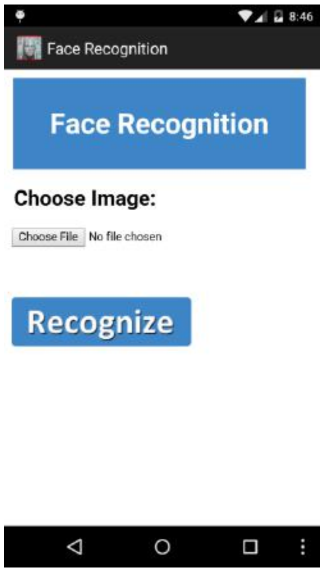


Figure 21: Recognition screen

It uploads the file to a web server. Then it executes python Client.py file and gives the image path. After a python file executes, it returns the results.

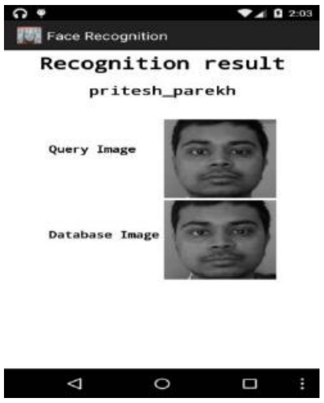


Figure 22: Recognition Result

Chapter 7: Testing and results

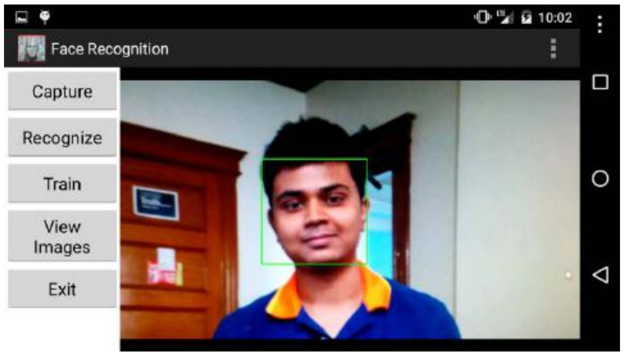
This project has implemented black box testing method to test the functional accuracy of the application. Black box testing can be defined as a method where testing examines the functionality of an application without looking to its code structure. Black box testing can be applied to every level and every function of the application.

Black box testing generally uses a template to create a testing scenario and the testing result. This project used a template consists of the test objective, task, the aim of the testing, result and special consideration. This chapter covers first functional testing and then recognition accuracy testing.

7.1. Functional Testing

7.1.1. Test case 1: Capture button works properly

Test Objective	Capture button works properly
Task	Open application Focus camera on a person
AIM	Clicking on Capture button should save face of a person and give option to user to perform Recognition or Training.
Result	It works as expected
Special Consideration	Lighting condition should be good



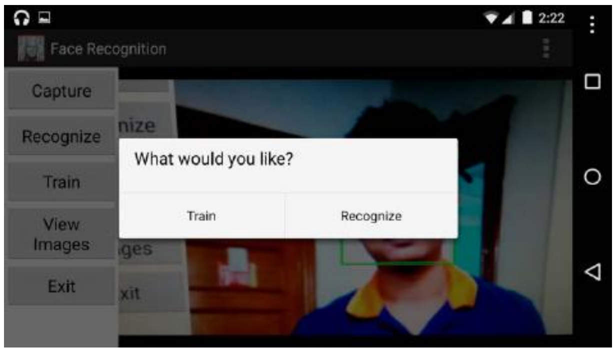


Figure 23: Test case 1: Capturing a face

7.1.2. Test case 2: Recognize button works properly

Test Objective	Recognize button works properly
Task	Click on Recognize Select an image to recognize
AIM	Clicking on Recognize button should redirect to screen where user can select an Image to recognize
Result	It works as expected
Special Consideration	None

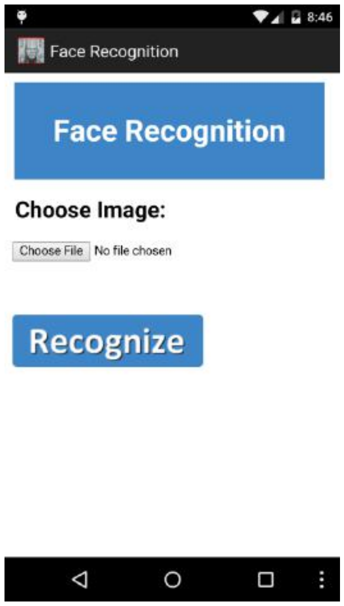
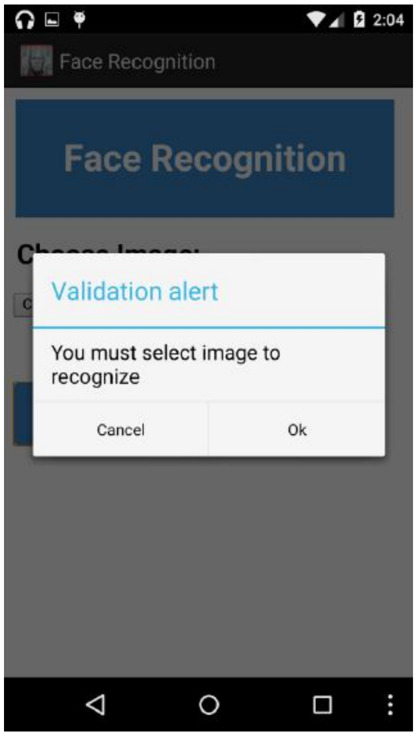


Figure 24: Recognition Screen

7.1.3. Test case 3: Validation on Recognize Screen

Test Objective	Validation works properly
Task	Click on Recognize button on Recognize screen without selecting and Image
AIM	It should populate an alert to select an Image
Result	It works as expected
Special Consideration	None



You must select image to recognize

Figure 25: Validation on Recognize screen

7.1.4. Test case 4: Working of Train button

Test Objective	Train button works properly
Task	Click on Train button
AIM	It should redirect to train screen
Result	It works as expected
Special Consideration	None

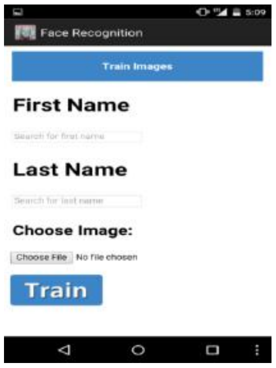
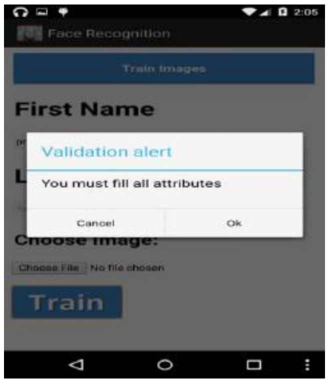


Figure 26: Train Screen

7.1.5. Test case 5: Validation on Train button

Test Objective	Validation on Train button works properly
Task	Click on Train button without entering anything or selecting an image.
AIM	It should give alert to fill all attributes
Result	It works as expected
Special Consideration	None



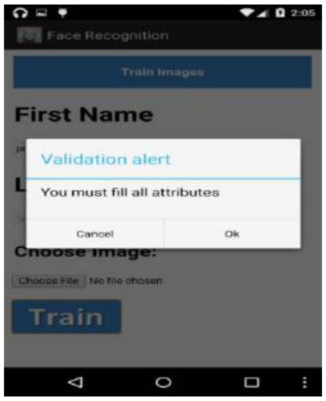
Validation alert

You must fill all attributes

Figure 27: Train screen validation

7.1.6. Test case 6: Validation on Train button

Test Objective	Validation on Train button works properly
Task	Click on Train button with entering one of the attributes
AIM	It should give alert to fill all attributes
Result	It works as expected
Special Consideration	None



Validation alert

You must fill all attributes

Figure 28: Validation on Train page

7.1.7. Test case 7: Suggestion for duplicate names on train page

Test Objective	Suggestion for duplicate names on train page
Task	Type one or two characters of first or last name that we already added on train screen
AIM	It should generate drop down menus for duplicate name
Result	It works as expected
Special Consideration	None

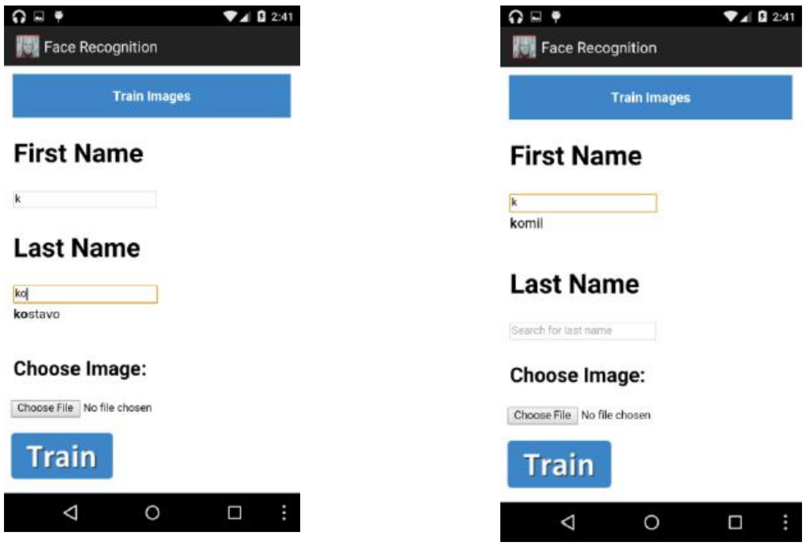


Figure 29: Suggestion for duplicate name

7.1.8. Test case 8: Recognition result

Test Objective	Recognition result for added person
Task	Recognize a new image of a person which is already in the database
AIM	It should give proper result
Result	It works as expected
Special Consideration	Lighting condition, angle of an image, smile, beard etc.

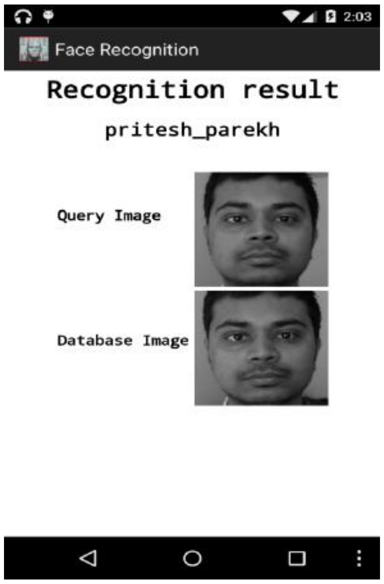


Figure 30: Recognition result

7.1.9. Test case 9: Recognition result

Test Objective	Recognition result for Unknown person or non-human image
Task	Recognize a new image of a Unknown person or non-human image
AIM	It should give Un known Image
Result	It works as expected
Special Consideration	Lighting condition, angle of an image, smile, beard etc.

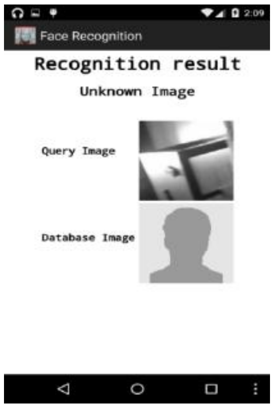
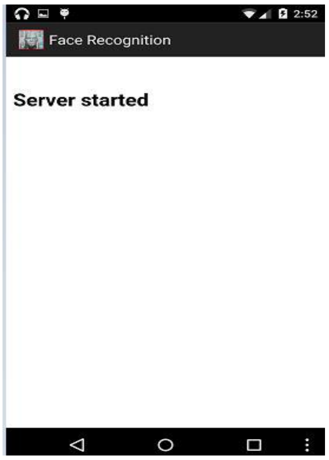


Figure 31: Recognition result of unknown image

7.1.10. Test case 10: Start server from Application

Test Objective	Start server from Application
Task	Start server from Application
AIM	It should start server
Result	It works as expected
Special Consideration	Connectivity between cell phone and server



Server started

Figure 32: Starting a server from application

7.2. Recognition Testing

To verify the accuracy of the recognition result, the system was tested in different lighting condition and different camera angles for one person. Therefore, minimum 7-8 images of each person should be used to create Eigenfaces for better accuracy.

Person Name: Pritesh Parekh

The following images were used to make Eigenfaces for the above person.

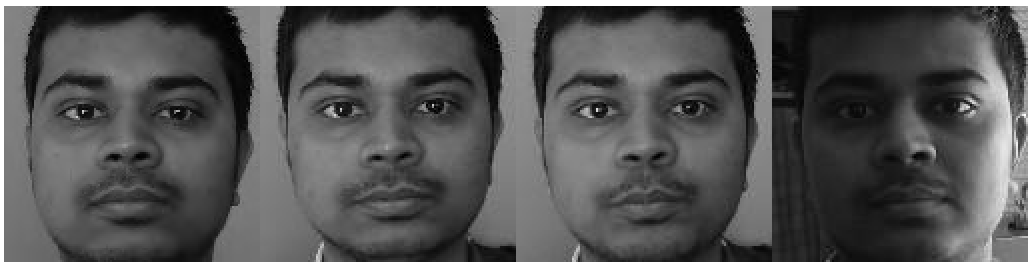
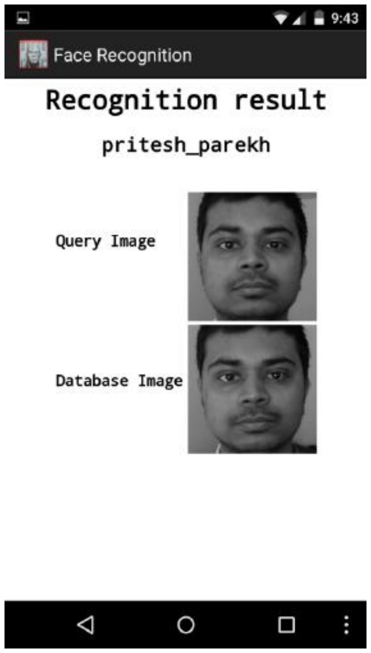


Figure 33: Test images to create Eigenfaces

7.2.1. Test 1: With same one of the picture

First, the system was tested with the same image that was used for making Eigenfaces. It gave the correct prediction.



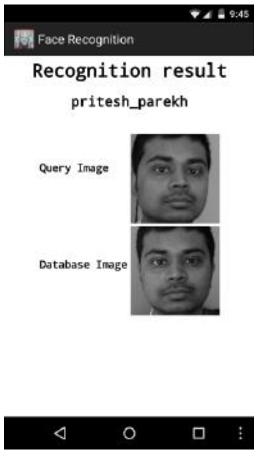
Recognition result

pritesh_parekh

Figure 34: Recognition prediction with same picture

7.2.2. Test 2: With different angle but same person

Second test consists of a query image of a same person, but with a different angle. It showed correct prediction.



Recognition result

pritesh_parekh

Figure 35: Recognition prediction with different angle

7.2.3. Test 3: Same person with different picture with smile

Third test consists of a query image of a same person, but with different picture and smile. It showed unknown image.

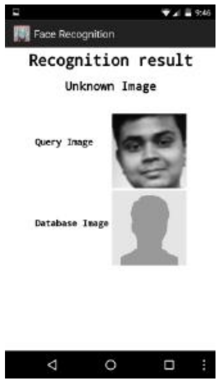


Figure 36: Recognition prediction with smile

7.2.4. Test 4: With an image that is not a face

This test consists of a query image that is not a face. It showed Unknown Image as expected.

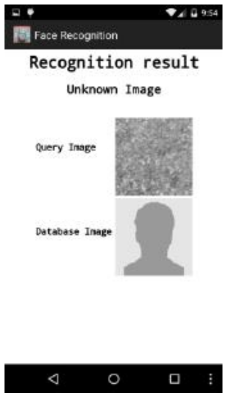


Figure 37: Recognition prediction with not a face

7.3. Conclusion:

The conclusion of the recognition testing is as follows.

- It is recommended that there should be at least 7-8 images with a different camera angle, light condition stored in the database.
- The lighting condition can be the major effect on the prediction.
- The smile on a face or a beard can lead to wrong prediction
- The camera on a cell phone is also a factor that can affect the prediction.
- No system can provide 100% accuracy for face recognition and this system is no exception.

Chapter 8: Installation and Configuration

This chapter covers the installation and configuration procedure for the application developer for this project. It has a description and screenshots for better understanding of a reader. It should be noted by a reader that this configuration only works for Windows environment.

8.1. Step1: Configuring of Android application Development

First of all we need to set up Android SDK and Eclipse. Android SDK and Eclipse can be obtained from Android's developer web site.

http://developer.android.com/sdk/installing/installing-adt.html

It has step by step guidelines to install and configure Android SDK in Eclipse environment.

8.2. Step 2: Configure OpenCV SDK to Android

OpenCV SDK for Android can be downloaded from here.
http://opencv.org/downloads.html

After download, extract the zip file. It will have following folders.

- **apk** : It consists of apk files of sample application provided by OpenCV.
- **doc** : It consists of documents like OpenCV tutorial, user guide etc.
- **sample**: It has eclipse projects for sample applications provided by OpenCV like Face detection.
- **Sdk:** It consists of system files for OpenCV

Now, follow step 3 to complete the installation.

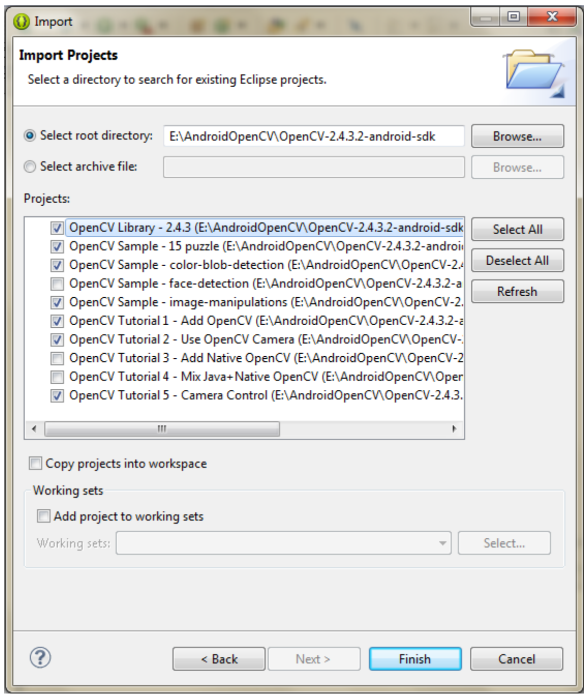
8.3. Step 3: Import OpenCV projects to Eclipse

First, open eclipse.

Second, click File – Import – General - Existing Projects into Workspace – Next then browse to the OpenCV SDK folder and locate sample folder.

Third, select all sample projects and import it to the eclipse as shown in Fig 38

Forth, locate face detection sample application from the list of eclipse projects.



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: E:\AndroidOpenCV\OpenCV-2.4.3.2-android-sdk

○ Select archive file:

Projects:

- [✓] OpenCV Library - 2.4.3 (E:\AndroidOpenCV\OpenCV-2.4.3.2-android-sdk
- [✓] OpenCV Sample - 15 puzzle (E:\AndroidOpenCV\OpenCV-2.4.3.2-androi
- [✓] OpenCV Sample - color-blob-detection (E:\AndroidOpenCV\OpenCV-2.4
- [ ] OpenCV Sample - face-detection (E:\AndroidOpenCV\OpenCV-2.4.3.2-a
- [✓] OpenCV Sample - image-manipulations (E:\AndroidOpenCV\OpenCV-2.
- [✓] OpenCV Tutorial 1 - Add OpenCV (E:\AndroidOpenCV\OpenCV-2.4.3.2-a
- [✓] OpenCV Tutorial 2 - Use OpenCV Camera (E:\AndroidOpenCV\OpenCV-
- [ ] OpenCV Tutorial 3 - Add Native OpenCV (E:\AndroidOpenCV\OpenCV-2
- [ ] OpenCV Tutorial 4 - Mix Java+Native OpenCV (E:\AndroidOpenCV\Oper
- [✓] OpenCV Tutorial 5 - Camera Control (E:\AndroidOpenCV\OpenCV-2.4.3.

[ ] Copy projects into workspace

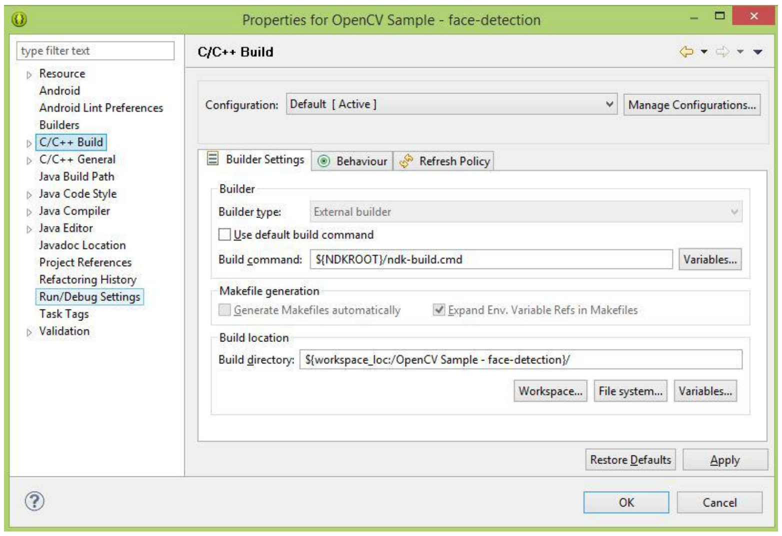
Working sets

[ ] Add project to working sets

Working sets:

Figure 38: Importing OpenCV to Eclipse

Fifth, right click on face detection and set up build path as shown in Fig 39.



Properties for OpenCV Sample - face-detection

C/C++ Build

Configuration: Default [Active]

Builder Settings | Behaviour | Refresh Policy

Builder

Builder type: External builder

[ ] Use default build command

Build command: ${NDKROOT}/ndk-build.cmd

Makefile generation

[ ] Generate Makefiles automatically   [✓] Expand Env. Variable Refs in Makefiles

Build location

Build directory: ${workspace_loc:/OpenCV Sample - face-detection}/

Figure 39: Build path

Fifth, Try to run sample application and if everything is correct app will run smoothly without any errors.

8.4. Step 4: Installing Python and OpenCV for Web server

Download Python 2.7.9 from its official website.

https://www.python.org/download/releases/2.7/

Then Download OpenCV for Windows from its official website from following link. Then perform following steps

http://opencv.org/downloads.html

1. Extract OpenCV (will be extracted to a folder OpenCV)
2. Copy..\opencv\build\python\x86\2.7\cv2.pyd
3. Paste it in C:\Python27\Lib\site-packages
4. Open Python IDLE or terminal, and type

```
>>> import cv2
```

If no errors shown, it is OK.

8.5. Step 5: Installing dependencies

As mentioned in previous chapters, this project has the following dependencies for image processing.

NumPy, SciPy, Matplotlib, PIL.

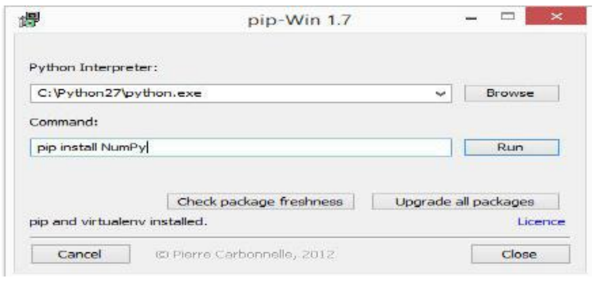
All these libraries can be obtained through PIP installer, which can be found here.

https://pip.pypa.io/en/latest/installing.html

For windows PIP comes with an exe file and it does not need to be installed. Exe file can be obtained from here

https://sites.google.com/site/pydatalog/python/pip-for-windows

After you have run PIP exe file you can install other libraries like below.



Python Interpreter: C:\Python27\python.exe

Command: pip install NumPy

pip and virtualenv installed.

Figure 40: PIP screen

You can install SciPy, Matplotlib, and PIL as shown in above Fig 40.

8.6. Step 6: Installing PHP and MySQL

Installation of PHP and MySQL is fairly straight forward. It is available on

http://www.wampserver.com/en/

8.7. Step 7: Making web server available to Client

If your web server can be accessed via internet then there is no need to perform this step.

However, if it is not then, there is a third party tool called Ngrok that can provide internet access to your web server.

“ngrok allows you to expose a web server running on your local machine to the internet. Just tell ngrok what port your web server is listening on.” (Ngrok)

Ngrok can be downloaded from https://ngrok.com/

Chapter 9: Future work

Every system has a scope for improvement and this project is not an exception. There is a scope of future enhancement in this project in the following areas.

Rate limiting: This application does not have rate limiting implemented. In a big organization where there is a situation where the rate of traffic sent or received is very high, this application may have undesired results. Therefore, for future work one can implement rate limiting.

Android-PHP Web Service: Currently, in this application a web page is accessed from a cell phone to train or recognize the images. For future enhancement, one can use a web service to handle client- server operation for better performance." Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks (Web Service Architecture)"

Face recognition: The face recognition algorithms are a vast area of research. This system uses Eigenfaces approach to train and recognize, but one can use latest technique or algorithms for better efficiency.

Face detection: Currently, this application only detects a face from a digital image. However, there are some advancement to find gender, race and age from a face. Therefore, one can add this functionality to the system." Face++ provides accurate technology of facial analysis, which can predict gender, age, race and expression from image and video." (Face++ - Tech Service)

Additional attributes: Currently, this application only stores first name and last name of a user in the database. For future enhancement, one can store additional attributes like gender, age, race of a person. Having all this information in the database will help the program to filter the database to find a match for a given query image.

For example, if a face detection finds that the given query image is a male and approximately age of 25 then it can give this information to face recognition program and face recognition can filter the database to 25 years old male person. Therefore, we have a faster recognition process.

References

Cascade Classification. (n.d.). Retrieved April 17, 2015, from OpenCV: http://docs.opencv.org/modules/objdetect/doc/cascade_classification.html?highlight=cascadeclassifier#haar-feature-based-cascade-classifier-for-object-detection

Dewey, J. (2013). Face Recognition. *Salem Press Encyclopedia*.

Face Recognition OpenCV. (n.d.). Retrieved April 20, 2015, from OpenCV: http://docs.opencv.org/modules/contrib/doc/facerec/facerec_tutorial.html

Face Recognition with OpenCV. (n.d.). Retrieved April 17, 2015, from Google Play: https://play.google.com/store/apps/details?id=org.opencv.javacv.facerecognition&hl=en

Face Recognition with OpenCV - OpenCV 2.4.11.0 documentation. (n.d.). Retrieved April 17, 2015, from OpenCV: http://docs.opencv.org/modules/contrib/doc/facerec/facerec_tutorial.html

Face Recogniton Fast Acess. (n.d.). Retrieved April 17, 2015, from Google Play: https://play.google.com/store/apps/details?id=com.sensiblevision.android.fassoapplock&hl=en

Face++ - Tech Service. (n.d.). Retrieved April 17, 2015, from Face Plus Plus: http://www.faceplusplus.com/tech-%E2%80%A2-service/

Gong, S., McKenn, S. J., & Psarrou, A. (2000). *Dynamic Vision: From Images to Face Recognition.* London: World Scientific Publishing Company.

K nearest neighbor. (n.d.). Retrieved April 20, 2015, from OpenCV: http://docs.opencv.org/modules/ml/doc/k_nearest_neighbors.html

Kanade, T. (1973, November). Picture Processing System by Computer Complex and Recognition of Human Faces.

Kande, T. (1973). Picture processing system by computer complex and recognition of human faces.

Ngrok. (n.d.). Retrieved from Ngrok: https://ngrok.com/docs#expose

OpenCV. (n.d.). Retrieved April 20, 2015, from http://docs.opencv.org/doc/tutorials/imgproc/histograms/histogram_equalization/histogram_equalization.html

Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2014). DeepFace: Closing the Gap to Human-Level Performance in Face Verification. *Conference on Computer Vision and Pattern Recognition (CVPR).*

Tan, X., & Triggs, B. (2010). Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions. *INRIA & Laboratoire Jean Kuntzmann, 655 avenue de l'Europe, Montbonnot 38330, France*, 15.

Turk, M., & Pentland, A. (1991). Eigenfaces for Recognition. *Cognitive Neuroscience*.

Wagner, P. (n.d.). *Imaplementing Face recognition*. Retrieved from bytefish: 2015

Web Service Architecture. (n.d.). Retrieved from W3.org: http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#introduction

Zhang, S., & Turk, M. (n.d.). *Eigenface Scholarpedia*. Retrieved April 2014, 2015, from Scholarpedi: http://www.scholarpedia.org/article/Eigenfaces

Appendix

The appendix contains code used for this project. It is divided into three parts.

1. An Android application code.
2. A PHP Web application code.
3. The Python code for recognition.

Android application code

Layout files for the user interface of the application

Face_detect_surface_view.xml

This file contains home screen for the Android application

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:background="@color/white"

  >
<Button
    android:id="@+id/button3"
    android:layout_width="120dp"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:onClick="onClickBtn"
    android:text="Capture" />
<org.opencv.android.JavaCameraView
    android:id="@+id/fd_activity_surface_view"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_alignParentRight="true"
    android:layout_alignParentTop="true"
    android:layout_toRightOf="@+id/button3"
    android:scaleType="fitXY"
   />
<Button
    android:id="@+id/button1"
    android:layout_width="120dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/button3"
    android:layout_toLeftOf="@+id/fd_activity_surface_view"
    android:onClick="onClickBtnUrl"
    android:text="Recognize" />

<Button
    android:id="@+id/Button01"
    android:layout_width="120dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/button1"
```

```
                android:layout_toLeftOf="@+id/fd_activity_surface_view"
                android:onClick="onClickBtnTrain"
                android:text="Train" />

            <Button
                android:id="@+id/button7"
                android:layout_width="120dp"
                android:layout_height="wrap_content"
                android:layout_alignParentLeft="true"
                android:layout_below="@+id/Button01"
                android:onClick="onClickViewPhotos"
                android:text="View Images" />

            <Button
                android:id="@+id/Button06"
                android:layout_width="120dp"
                android:layout_height="wrap_content"
                android:layout_below="@+id/button7"
                android:layout_toLeftOf="@+id/fd_activity_surface_view"
                android:onClick="onClickExtn"
                android:text="Exit" />

        </RelativeLayout>
```

Webview_xml

This file contains the user interface for recognition screen

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/white"

    >
<WebView
    android:id="@+id/webView1"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"></WebView>

 </RelativeLayout>
```

Webviewtrain.xml

This file contains the user interface for train screen.

```
<?xml version="1.0" encoding="utf-8"?>
<WebView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/webView2"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
```

```
            />
```

Java code files for the Android application

FdActivity.java

This file contains the implementation of capture, train, recognize and exit method.

```
            public void onClickBtn(View v)
                  {
                        final Context context = this;
                        Log.i(TAG, "OpenCV loaded successfully"+ "Column 
            Value=" +mFace.cols()+"Row Value"+mFace.rows());
                        Bitmap bitmap = Bitmap.createBitmap(mFace.cols(), 
            mFace.rows(), Bitmap.Config.ARGB_8888);
                        Log.i(TAG, "OpenCV loaded successfully"+ "Column 
            ValueX=" +(mFace.cols()-5)+"Row ValueX"+(mFace.rows()-5));
                        Utils.matToBitmap(mFace, bitmap);
                        bitmap = Bitmap.createScaledBitmap(bitmap, 128, 128, 
            false);
                        String root = 
            Environment.getExternalStorageDirectory().toString();
                        File myDir = new File(root + "/saved_images");
                        myDir.mkdirs();
                        Random generator = new Random();
                        int n = 1000;
                        n = generator.nextInt(n);
                        String fname = "Image-"+ n +".jpg";
                        File file = new File (myDir, fname);
                        if (file.exists ()) file.delete ();
                        try {
                              FileOutputStream out = new 
            FileOutputStream(file);
                              bitmap.compress(Bitmap.CompressFormat.JPEG, 90, 
            out);
                              out.flush();
                              out.close();

                              //Intent intent = new Intent(context, 
            WebViewActivity.class);
                              //startActivity(intent);

                        } catch (Exception e) {
                              e.printStackTrace();
                        }

                        DialogInterface.OnClickListener dialogClickListener = 
            new DialogInterface.OnClickListener() {
                              @Override
                              public void onClick(DialogInterface dialog, int 
            which) {
                                    switch (which){
                                    case DialogInterface.BUTTON_POSITIVE:
                                          Intent intent = new Intent(context, 
            WebViewActivity.class);
                                          startActivity(intent);
                                          //Toast.makeText(this, "Image saved
            successfully", Toast.LENGTH_LONG).show();
                                          break;

                                    case DialogInterface.BUTTON_NEGATIVE:
                                          intent = new Intent(context,
            WebViewActivity2.class);
                                          startActivity(intent);

                                          break;
                                    }
                              }
                        };

                        AlertDialog.Builder builder = new
            AlertDialog.Builder(context);
                        builder.setMessage("What would you
            like?").setPositiveButton("Recognize", dialogClickListener)
                        .setNegativeButton("Train",
            dialogClickListener).show();


                  }
                  public void onClickBtnTrain(View v)
                  {
                        final Context context = this;
                        Intent intent = new Intent(context,
            WebViewActivity2.class);
                        startActivity(intent);

                  };



                  public void onClickBtnUrl(View v)
                  {
                        final Context context = this;

                        /*Uri uri =
            Uri.parse("http://codd.cs.montclair.edu/~parekhp/cgi-
            bin/OnlineShopping/Index.php");
                        Intent intent = new Intent(Intent.ACTION_VIEW, uri);
                        startActivity(intent);*/
                        Intent intent = new Intent(context,
            WebViewActivity.class);
                        startActivity(intent);

                  }
                  public void onClickExtn (View v)
                  {
                        finish();
                  }
                  public void onClickViewPhotos (View v)
                  {
                        Intent intent = new
            Intent(Intent.ACTION_GET_CONTENT);
                        intent.addCategory(Intent.CATEGORY_OPENABLE);
                        Uri uri =
            Uri.parse(Environment.getExternalStorageDirectory().getPath()
                                    + "/saved_images");
                        intent.setDataAndType(uri, "*/*");
                        startActivityForResult(Intent.createChooser(intent,
            "File Browser"),1);
                  }
```

Note: This file contains code other than these methods in not stated here

WebViewActivity.Java

This file contains java code for face recognition screen.

```
public class WebViewActivity extends Activity {

      WebView web;
      // ProgressBar progressBar;

      private ValueCallback<Uri> mUploadMessage;
      private final static int FILECHOOSER_RESULTCODE=1;
      protected static final int PIC_REQUEST = 1;
       private JsResult mResult;
      @Override
      protected void onActivityResult(int requestCode, int
resultCode,
                  Intent intent) {
            if(requestCode==FILECHOOSER_RESULTCODE)
            {
                  if (null == mUploadMessage) return;
                  Uri result = intent == null || resultCode !=
RESULT_OK ? null
                              : intent.getData();
                  mUploadMessage.onReceiveValue(result);
                  mUploadMessage = null;
            }
      }
      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.webview);

            web = (WebView) findViewById(R.id.webView1);
            //progressBar = (ProgressBar)
findViewById(R.id.progressBar1);
            web.getSettings().setJavaScriptEnabled(true);

        // Other webview options
        web.getSettings().setLoadWithOverviewMode(true);

        //webView.getSettings().setUseWideViewPort(true);

        //Other webview settings
```

```
web.setScrollBarStyle(WebView.SCROLLBARS_OUTSIDE_OVERLAY);
        web.setScrollbarFadingEnabled(false);
        web.getSettings().setBuiltInZoomControls(true);
        web.getSettings().setPluginState(PluginState.ON);
        web.getSettings().setAllowFileAccess(true);
        web.getSettings().setSupportZoom(true);
            web = new WebView(this);
            web.getSettings().setJavaScriptEnabled(true);

      //web.loadUrl("http://codd.cs.montclair.edu/~parekhp/cgi-
bin/webserv/x.php");

      //web.loadUrl("http://55936bda.ngrok.com/webserv/x.php");

      //web.loadUrl("http://12241605.ngrok.com/Face_Recognition/R
ecognize.php");

      web.loadUrl("http://192.168.1.4/Face_Recognition/Recognize.
php");
            //web.setWebViewClient(new myWebClient());
            web.setWebChromeClient(new WebChromeClient());
            //web.setWebViewClient(new WebViewClient()
            web.setWebChromeClient(new WebChromeClient()
            {
                   public boolean onJsAlert(WebView view, String
url, String message, JsResult result) {
                        mResult = result;
                        AlertDialog dialog = new
AlertDialog.Builder(WebViewActivity.this)
                                .setTitle("Validation alert")
                                .setMessage("You must select
image to recognize")
                                .setOnCancelListener(new
CancelListener())
                                .setNegativeButton("Cancel", new
CancelListener())
                                .setPositiveButton("Ok", new
PositiveListener())
                                .create();

dialog.getWindow().setType(WindowManager.LayoutParams.TYPE_SYSTEM
_ALERT);
                        dialog.show();

                        return true;
                    }

                  //The undocumented magic method override
                  //Eclipse will swear at you if you try to put
@Override here
                  // For Android 3.0+

                  public void openFileChooser(ValueCallback<Uri>
uploadMsg) {

                        mUploadMessage = uploadMsg;
                        Intent i = new
Intent(Intent.ACTION_GET_CONTENT);
                        i.addCategory(Intent.CATEGORY_OPENABLE);
                        i.setType("image/*");

      WebViewActivity.this.startActivityForResult(Intent.createCh
ooser(i,"File Chooser"), FILECHOOSER_RESULTCODE);

                  }

                  // For Android 3.0+
                  public void openFileChooser( ValueCallback
uploadMsg, String acceptType ) {
                        mUploadMessage = uploadMsg;
                        Intent i = new
Intent(Intent.ACTION_VIEW);
                        i.addCategory(Intent.CATEGORY_OPENABLE);
                        i.setType("*/*");

      WebViewActivity.this.startActivityForResult(
                              Intent.createChooser(i, "File
Browser"),
                              FILECHOOSER_RESULTCODE);
                  }

                  //For Android 4.1
                  public void openFileChooser(ValueCallback<Uri>
uploadMsg, String acceptType, String capture){
                        mUploadMessage = uploadMsg;
                        //Intent intent = new
                        Intent intent = new
Intent(Intent.ACTION_GET_CONTENT);

      intent.addCategory(Intent.CATEGORY_OPENABLE);
                        Uri uri =
Uri.parse(Environment.getExternalStorageDirectory().getPath()
                            + "/saved_images");
                        intent.setDataAndType(uri, "*/*");

      startActivityForResult(Intent.createChooser(intent, "File
Browser"),1);
                  }

            });


            setContentView(web);


      }
      private class CancelListener implements
DialogInterface.OnCancelListener,
    DialogInterface.OnClickListener {

    @Override
    public void onCancel(DialogInterface dialogInterface) {
        mResult.cancel();
    }

    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        mResult.cancel();
    }
}

private class PositiveListener implements
DialogInterface.OnClickListener {

    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        mResult.confirm();
    }
}
      public class myWebClient extends WebViewClient
      {
            @Override
            public void onPageStarted(WebView view, String url,
Bitmap favicon) {
                  // TODO Auto-generated method stub
                  super.onPageStarted(view, url, favicon);
            }

            @Override
            public boolean shouldOverrideUrlLoading(WebView view,
String url) {
                  // TODO Auto-generated method stub

                  view.loadUrl(url);
                  return true;

            }

            @Override
            public void onPageFinished(WebView view, String url)
{
                  // TODO Auto-generated method stub
                  super.onPageFinished(view, url);

                  // progressBar.setVisibility(View.GONE);
            }
      }

      //flipscreen not loading again
      @Override
      public void onConfigurationChanged(Configuration
newConfig){
            super.onConfigurationChanged(newConfig);
      }

}
```

WebViewActivity2.java

This file contains java code for face training screen.

```
public class WebViewActivity2 extends Activity {

      WebView web;
       private JsResult mResult;
      // ProgressBar progressBar;

      private ValueCallback<Uri> mUploadMessage;
      private final static int FILECHOOSER_RESULTCODE=1;
      protected static final int PIC_REQUEST = 1;

      @Override
      protected void onActivityResult(int requestCode, int
resultCode,
                  Intent intent) {
            if(requestCode==FILECHOOSER_RESULTCODE)
            {
                  if (null == mUploadMessage) return;
                  Uri result = intent == null || resultCode !=
RESULT_OK ? null
                              : intent.getData();
                  mUploadMessage.onReceiveValue(result);
                  mUploadMessage = null;
            }
      }

      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.webviewtrain);

            web = (WebView) findViewById(R.id.webView2);
            //progressBar = (ProgressBar)
findViewById(R.id.progressBar1);
            web.getSettings().setJavaScriptEnabled(true);

        // Other webview options
        web.getSettings().setLoadWithOverviewMode(true);

        //webView.getSettings().setUseWideViewPort(true);

        //Other webview settings

web.setScrollBarStyle(WebView.SCROLLBARS_OUTSIDE_OVERLAY);
        web.setScrollbarFadingEnabled(false);
        web.getSettings().setBuiltInZoomControls(true);
        web.getSettings().setPluginState(PluginState.ON);
        web.getSettings().setAllowFileAccess(true);
        web.getSettings().setSupportZoom(true);
       // web.setClickable(false);
            web = new WebView(this);
            web.getSettings().setJavaScriptEnabled(true);

      //web.loadUrl("http://codd.cs.montclair.edu/~parekhp/cgi-
bin/webserv/x.php");
```

```
      //web.loadUrl("http://12241605.ngrok.com/Face_Recognition/T
rain.php");

      web.loadUrl("http://192.168.1.4/Face_Recognition/Train.php"
);
            //web.setWebViewClient(new myWebClient());
            web.setWebChromeClient(new WebChromeClient());
            //web.setWebViewClient(new WebViewClient()
            web.setWebChromeClient(new WebChromeClient()
            {
                  //The undocumented magic method override
                  //Eclipse will swear at you if you try to put
@Override here
                  // For Android 3.0+

                  public boolean onJsAlert(WebView view, String
url, String message, JsResult result) {
                  mResult = result;
                  AlertDialog dialog = new
AlertDialog.Builder(WebViewActivity2.this)
                          .setTitle("Validation alert")
                          .setMessage("You must fill all
attributes")
                          .setOnCancelListener(new
CancelListener())
                          .setNegativeButton("Cancel", new
CancelListener())
                          .setPositiveButton("Ok", new
PositiveListener())
                          .create();

dialog.getWindow().setType(WindowManager.LayoutParams.TYPE_SYSTEM
_ALERT);
                  dialog.show();

                  return true;
              }

                  public void openFileChooser(ValueCallback<Uri>
uploadMsg) {

                        mUploadMessage = uploadMsg;
                        Intent i = new
Intent(Intent.ACTION_GET_CONTENT);
                        i.addCategory(Intent.CATEGORY_OPENABLE);
                        i.setType("image/*");

      WebViewActivity2.this.startActivityForResult(Intent.createC
hooser(i,"File Chooser"), FILECHOOSER_RESULTCODE);

                  }

                  // For Android 3.0+
                  public void openFileChooser( ValueCallback
uploadMsg, String acceptType ) {
                        mUploadMessage = uploadMsg;
                        Intent i = new
Intent(Intent.ACTION_VIEW);
                        i.addCategory(Intent.CATEGORY_OPENABLE);
                        i.setType("*/*");

      WebViewActivity2.this.startActivityForResult(
                                    Intent.createChooser(i, "File
Browser"),
                                    FILECHOOSER_RESULTCODE);
                  }

                  //For Android 4.1
                  public void openFileChooser(ValueCallback<Uri>
uploadMsg, String acceptType, String capture){
                        mUploadMessage = uploadMsg;
                        //Intent intent = new
Intent(Intent.ACTION_GET_CONTENT);


                        //intent.setDataAndType(ui, "*/*");
                        //startActivity(intent);
                        Intent intent = new
Intent(Intent.ACTION_GET_CONTENT);

      intent.addCategory(Intent.CATEGORY_OPENABLE);
                        Uri uri =
Uri.parse(Environment.getExternalStorageDirectory().getPath()
                            + "/saved_images");
                        intent.setDataAndType(uri, "*/*");

      startActivityForResult(Intent.createChooser(intent, "File
Browser"),1);
                  }

            });


            setContentView(web);


      }
      private class CancelListener implements
DialogInterface.OnCancelListener,
    DialogInterface.OnClickListener {

    @Override
    public void onCancel(DialogInterface dialogInterface) {
        mResult.cancel();
    }

    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        mResult.cancel();
    }
}
```

```
private class PositiveListener implements
DialogInterface.OnClickListener {

    @Override
    public void onClick(DialogInterface dialogInterface, int i) {
        mResult.confirm();
    }
}

      public class myWebClient extends WebViewClient
      {
            @Override
            public void onPageStarted(WebView view, String url,
Bitmap favicon) {
                  // TODO Auto-generated method stub
                  super.onPageStarted(view, url, favicon);
            }

            @Override
            public boolean shouldOverrideUrlLoading(WebView view,
String url) {
                  // TODO Auto-generated method stub

                  view.loadUrl(url);
                  return true;

            }

            @Override
            public void onPageFinished(WebView view, String url)
{
                  // TODO Auto-generated method stub
                  super.onPageFinished(view, url);

                  // progressBar.setVisibility(View.GONE);
            }
      }

      //flipscreen not loading again
      @Override
      public void onConfigurationChanged(Configuration
newConfig){
            super.onConfigurationChanged(newConfig);
      }
```

Manifest.xml

This file contains the list of permissions needed from the user to run this application.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.opencv.samples.facedetect"
    android:versionCode="21"
    android:versionName="2.1" >
    <application android:label="@string/app_name"
        android:icon="@drawable/ic_launcher"

android:theme="@android:style/Theme.Holo.Light.DarkActionBar">
        <activity
            android:name="FdActivity"
            android:configChanges="keyboardHidden|orientation"
            android:label="@string/app_name"
            android:screenOrientation="landscape" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN"
/>

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".WebViewActivity"
            android:label="@string/app_name"
            android:screenOrientation="portrait"
            />
        <activity
            android:name=".WebViewActivity2"
            android:label="@string/app_name"
            android:screenOrientation="portrait"
            />
    </application>

    <supports-screens
        android:anyDensity="true"
        android:largeScreens="true"
        android:normalScreens="true"
        android:resizeable="true"
        android:smallScreens="true" />

    <uses-sdk android:minSdkVersion="8" />

    <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.INTERNET"
/>
 />
    <uses-permission
android:name="android.permission.SYSTEM_ALERT_WINDOW" />

    <uses-feature
        android:name="android.hardware.camera"
        android:required="false" />
    <uses-feature
        android:name="android.hardware.camera.autofocus"
        android:required="false" />
    <uses-feature
        android:name="android.hardware.camera.front"
```

```
            android:required="false" />
        <uses-feature
            android:name="android.hardware.camera.front.autofocus"
            android:required="false" />
        <uses-feature
    android:name="android.hardware.camera.autofocus" />
        <uses-feature android:name="android.hardware.camera.front" />
        <uses-feature
    android:name="android.hardware.camera.front.autofocus" />

    </manifest>
```

PHP Web application code

Recognize.php

```
<!DOCTYPE html>
<html>
<head>
      <script>
            function validateForm() {
                  var x =
document.forms["myForm"]["fileToUpload"].value;
                  if (x == null || x == "") {
                        alert("You must select the file");
                        return false;
                  }
            }
      </script>
      <style>
            h1 {
                  font-size: 150%;
            }

            h2 {
                  font-size: 100%;
            }

            p {
                  font-size: 200%;
                  font-color:#FFFFFF ;
            }
      </style>
</head>
<body>
            <form name="myForm" action="upload.php" method="post"
enctype="multipart/form-data" onsubmit="return validateForm()">
            <table style="width:100%" align="center" width="100%"
height="100%">
                  <tr>
```

```
                        <th bgcolor="#3d85c6"
align="center"><p><font face="Calibri" color="white"><b>Face
Recognition</b></font></p></th>
                  </tr>
                  <tr>
                        <td colspan="2"><h1>Choose
Image:</h1></td>
                  <td>
                  </tr>
            </table>
            <div>
                  <input type="file" name="fileToUpload"
id="fileToUpload"/>
            </div>
            <br/>
            <br/>
            <br/>
                  <input type="image" name="button"
src="images/button.png" />
            </form>
</body>
</html>
```

Train.php

```
<!DOCTYPE html>
<html>
<head>

<script type="text/javascript" src="jquery-
1.8.0.min.js"></script>
<script type="text/javascript">
$(function()
{                 $(".search").keyup(function()
                  {
                              var searchid = $(this).val();
                              var dataString = 'search='+
searchid;
                              if(searchid!='')
                              {
                                    $.ajax({
                                    type: "POST",
                                    url: "search.php",
                                    data: dataString,
                                    cache: false,
                                    success: function(html)
                                    {

      $("#result").html(html).show();
                                    }
                                    });
                              }return false;
                  });
                  jQuery("#result").live("click",function(e)
                  {
                                    var $clicked = $(e.target);
                                    var $name =
$clicked.find('.name').html();
                                    var decoded =
$("<div/>").html($name).text();
                                    $('#searchid').val(decoded);
                  });
                              jQuery(document).live("click",
function(e) {
                                    var $clicked = $(e.target);
                                    if (!
$clicked.hasClass("search")){
                                    jQuery("#result").fadeOut();
                                    }
                              });
                              $('#searchid').click(function(){
                                    jQuery("#result").fadeIn();
                              });


      ///////////////////////////////////////////////////

      /////////////////////////////////////////////////////
                  $(".searchX").keyup(function()
                  {
                              var searchidT = $(this).val();
                              var dataStringX = 'search='+
searchidT;
                              if(searchidT!='')
                              {
                                    $.ajax({
                                    type: "POST",
                                    url: "search1.php",
                                    data: dataStringX,
                                    cache: false,
                                    success: function(html)
                                    {

      $("#resultX").html(html).showX();
                                    }
                                    });
                              }return false;
                  });
                  jQuery("#resultX").live("click",function(e)
                  {
                        var $clicked = $(e.target);
                        var $nameX =
$clicked.find('.nameX').html();
                        var decoded =
$("<div/>").html($nameX).text();
                        $('#searchidX').val(decoded);
                  });
                  jQuery(document).live("click", function(e)
                  {
                        var $clicked = $(e.target);
                        if (! $clicked.hasClass("searchX")){
                        jQuery("#resultX").fadeOut();
                        }
                  });
                  $('#searchidX').click(function(){
                        jQuery("#resultX").fadeIn();
                  });

});
</script>
<script>
            function validateForm() {
                  var x =
document.forms["myForm"]["fileUpload"].value;
                  var y =
document.forms["myForm"]["firstname"].value;
                  var z =
document.forms["myForm"]["lastname"].value;
                  if (x == null || x == "") {
                        alert("You must select the file");
                        return false;
                  }
                  else if (y == null || y == "" && z == null || z
== "")
                  {
                        alert("You must select the file");
                        return false;
                  }
            }
      </script>
</head>
<body>
      <form name="myForm" action="z.php" method="post"
enctype="multipart/form-data" onsubmit="return validateForm()">
            <table style="width:100%" align="center" width="100%"
height="100%">
                  <tr>
```

```
                        <th bgcolor="#3d85c6"
align="center"><p><font face="Calibri" color="white"><b>Train
Images</b></font></p></th>
                  </tr>
                  <tr>
                        <td colspan="2"><h1>First Name</h1></td>
                  </tr>
                  <tr>
                        <td><input type="text" class="search"
id="searchid" placeholder="Search for first name"
name="firstname"/></td>
                  </tr>
                  <tr>
                        <td><div id="result"></td>
                  </tr>
                  <tr>
                        <td colspan="2"><h1>Last Name</h1></td>
                  </tr>
                  <tr>
                        <td><input type="text" class="searchX"
id="searchidX" placeholder="Search for last name"
name="lastname"/></td>
                  </tr>
                  <tr>
                        <td><div id="resultX"></td>
                  </tr>
                  <tr>
                        <td colspan="2"><h2>Choose
Image:</h2></td>
                  </tr>
            </table>
            <div>
                  <input type="file" name="fileUpload"
id="fileToUpload"/>
            </div>
            <br/>
            <input type="image" name="buttonT"
src="images/buttonT.png" />
      </form>
</body>
</html>
```

Search.php :

It is used to find duplicate first name and last name

```
<?php
$connection =
mysqli_connect('localhost','root','','facerecognition');
```

```
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }
if($_POST)
{
$q=$_POST['search'];


$qur ="select * from model where first_name like '%$q%' group by
first_name order by first_name";
//echo $qur;
$result=mysqli_query($connection,$qur);
if (!$result) {
    printf("Error: %s\n", mysqli_error($connection));
    exit();
}
while($row = mysqli_fetch_array($result))
{
$username=$row['first_name'];
//$email=$row['email'];
$b_username='<strong>'.$q.'</strong>';
//$b_email='<strong>'.$q.'</strong>';
$final_username = str_ireplace($q, $b_username, $username);
//$final_email = str_ireplace($q, $b_email, $email);

?>
<div class="show" align="left">
<span class="name"><?php echo $final_username;
?></span>&nbsp;<br/><br/>

</div>
<?php
}
}
mysqli_close($connection);
?>
```

Upload.php:

It is a program that performs recognition when recognize.php actions this page.

```
<html>
<body>
<?php
       session_start();
$target_dir = "uploads/";
$target_file = $target_dir .
basename($_FILES["fileToUpload"]["name"]);
$uploadOk = 1;
```

```
$imageFileType = pathinfo($target_file,PATHINFO_EXTENSION);
      if(isset($_POST["submit"])) {
                  $check =
getimagesize($_FILES["fileToUpload"]["tmp_name"]);
                  if($check !== false) {
                        $uploadOk = 1;
                  } else {
                        $uploadOk = 0;
                  }
            }
      if (file_exists($target_file)) {
       $uploadOk = 0;
      }
      if ($_FILES["fileToUpload"]["size"] > 500000) {
            echo "Sorry, your file is too large.";
            $uploadOk = 0;
      }
      if($imageFileType != "jpg" && $imageFileType != "png" &&
$imageFileType != "jpeg"
            && $imageFileType != "gif" ) {
            $uploadOk = 0;
      }
      if ($uploadOk == 0) {
      } else {
      if (move_uploaded_file($_FILES["fileToUpload"]["tmp_name"],
$target_file)) {
    } else {
    }
      }
$param1 = "first";
$param2 = "second";
$param3 = "third";
$fname = $_FILES["fileToUpload"]["name"];
$command = "python C:/wamp/www/Face_Recognition/facerec-
master/py/apps/webapp/client.py
C:/wamp/www/Face_Recognition/uploads/$fname";
$pid = popen( $command,"r");
echo "<body><pre>";
echo "<h1><center>Recognition result</center></h1>";
echo "<table align='center'>";
      //while( !feof( $pid ) )
      //{
            $a = fread($pid,256);
            $pos = strpos($a,"}");
            $x = -3;
            $rest = substr($a, 12, $x);
            ?>
            <h2><center><?php echo$rest?></center></h2>
            <?php
```

```
                  $pos1 = strpos($rest,"_");
                  $pos2 = 1 + $pos1;
                  $ns = substr($rest,$pos2);
                  //$pos3 = $pos1 - 6;
                  //$sn = substr($rest,$pos3);
                  //echo $sn;
            flush();
            ob_flush();
            echo "<script>window.scrollTo(0,99999);</script>";
            usleep(100000);
                  $connection =
mysqli_connect('localhost','root','','facerecognition');
                  if (mysqli_connect_errno())
                  {
                        echo "Failed to connect to MySQL: " .
mysqli_connect_error();
                  }
                  else
                  {
                        $qur ="select * from model where
last_name='$ns'";
                        //echo $qur;
                        $result=mysqli_query($connection,$qur);
                        if(mysqli_num_rows($result) > 0)
                        {
                                    $row =
mysqli_fetch_array($result);
                                    $path=$row['path'];
                              //    echo $path;

                        }
                        else
                        {
                              $path = "uploads/Unknown-
person.gif";
                              //echo $path;
                        }
                  //echo $path;

                  $_SESSION['pathx'] = $path;
                        //    echo $_SESSION['pathx'];
                  }
                  $iop = $_SESSION['pathx'];
                  //echo $iop;
      //    }
pclose($pid);
?>

      <tr>
```

```
            <td><h3>Query Image</h3></td>
            <td><br/><img src="uploads/<?php echo $fname;?>"
style="width:128px;height:128px"></td>
      </tr>
      <tr>

            <td><h3>Database Image</h3></td>
            <td><img src="<?php echo $iop;?>"
style="width:128px;height:128px"></td>
      </tr>
</table>

</body>
</html>
```

Z.php

It is a program that trains faces after Train.php actions this page

```
<?php
$fname = $_POST["firstname"];
$lname = $_POST["lastname"];
$connection =
mysqli_connect('localhost','root','','facerecognition');
if (mysqli_connect_errno())
{
      echo "Failed to connect to MySQL: " .
mysqli_connect_error();
}
else
{
      $qur ="select * from model where first_name = '$fname' and
last_name = '$lname'";
      //echo $qur;
      $result=mysqli_query($connection,$qur);
      if(mysqli_num_rows($result) > 0)
      {
            if (!$result)
            {
                  printf("Error: %s\n",
mysqli_error($connection));
                  exit();
            }
            $result=mysqli_query($connection,$qur);
            //echo "Gayu";
            while($row = mysqli_fetch_array($result))
            {
                  $firstname=$row['first_name'];
                  $lastname=$row['last_name'];
                  //echo $lastname;
                  if ($fname == $firstname && $lname==$lastname)
                  {
                        //echo "same";
                  }
                  else
                  {

                  }
            }
      }
      else
      {
                        //$filename = $_FILE['fileUpload'];
                        $filename =
($_FILES["fileUpload"]["name"]);
                        //echo $filename;
                        $path = "facerec-
master/dataset/data/c1/".$fname."_".$lname.'/'.$filename;
                        //echo $path;
                        $qur1 ="Insert into model
(first_name,last_name,path) values ('$fname','$lname','$path')";
                        //echo $qur1;
                        if
(!mysqli_query($connection,$qur1))
                        {
                              die('Error: ' .
mysqli_error($connection));
                        }
                        else
                        {
                              echo "Record Added";
                        }
      }
}
$foldername = $fname."_".$lname;

if (!file_exists("C:/wamp/www/Face_Recognition/facerec-
master/dataset/data/c1/$foldername"))
{
         mkdir("C:/wamp/www/Face_Recognition/facerec-
master/dataset/data/c1/$foldername");
         $target_dir = "C:/wamp/www/Face_Recognition/facerec-
master/dataset/data/c1/$foldername/";
       $target_file = $target_dir .
basename($_FILES["fileUpload"]["name"]);
         $uploadOk = 1;
        $imageFileType =
pathinfo($target_file,PATHINFO_EXTENSION);
```

```
            if(isset($_POST["submit"]))
            {
                  $check =
getimagesize($_FILES["fileUpload"]["tmp_name"]);
                  if($check !== false) {
                        $uploadOk = 1;
                  } else {
                        $uploadOk = 0;
                  }
            }
            if (file_exists($target_file)) {
                     $uploadOk = 0;
            }
            if ($_FILES["fileUpload"]["size"] > 500000) {
                  echo "Sorry, your file is too large.";
                  $uploadOk = 0;
            }if($imageFileType != "jpg" && $imageFileType !=
"png" && $imageFileType != "jpeg"
                  && $imageFileType != "gif" ) {
                  $uploadOk = 0;
            }
            if ($uploadOk == 0)
            {
            }
            else
            {
            if
(move_uploaded_file($_FILES["fileUpload"]["tmp_name"],
$target_file)) {
            } else {
    }
}
            $line = array
            (

      "$foldername;C:/wamp/www/Face_Recognition/facerec-
master/dataset/data/c1/$foldername",
            );
            $handle =
fopen("C:/wamp/www/Face_Recognition/facerec-
master/dataset/test.csv", "a");
            //$file = fopen("C:/facerec-
master/dataset/test.csv","a");
            fputcsv($handle, $line);
            fclose($handle);
}
else
{
```

```
            $target_dir = "C:/wamp/www/Face_Recognition/facerec-
master/dataset/data/c1/$foldername/";
            $target_file = $target_dir .
basename($_FILES["fileUpload"]["name"]);
            $uploadOk = 1;
        $imageFileType =
pathinfo($target_file,PATHINFO_EXTENSION);
            if(isset($_POST["submit"]))
            {
                  $check =
getimagesize($_FILES["fileUpload"]["tmp_name"]);
                  if($check !== false) {
                        $uploadOk = 1;
                  } else {
                        $uploadOk = 0;
                  }
            }
            if (file_exists($target_file)) {
                     $uploadOk = 0;
            }
            if ($_FILES["fileUpload"]["size"] > 500000) {
                  echo "Sorry, your file is too large.";
                  $uploadOk = 0;
            }if($imageFileType != "jpg" && $imageFileType !=
"png" && $imageFileType != "jpeg"
                  && $imageFileType != "gif" ) {
                  $uploadOk = 0;
            }
            if ($uploadOk == 0)
            {
            }
            else
            {
            if
(move_uploaded_file($_FILES["fileUpload"]["tmp_name"],
$target_file)) {
            } else {
            }
            }
      }

exec("event.bat");
echo "<h2>Server started</h2>";

//echo "file saved successfully";
?>
```

Python programs that performs facer recognition

This application uses facerec framework which is a popular framework that is used to implement OpenCV face recognition algorithms. It is free and under BSD license so any one can use it. (Wagner)

Client.py:

It reads the given query image and send it to server.py

```
import json
import base64
import urllib2

SERVER_ADDRESS = "http://localhost:5000"

class FaceRecClient(object):

    def __init__(self, url):
        self.url = url

    def getBase64(self, filename):
        with open(filename, "rb") as image_file:
            encoded_string = base64.b64encode(image_file.read())
        return encoded_string

    def request(self, api_func, request_data):
        url_func = "%s/api/%s" % (self.url, api_func)
        req = urllib2.Request(url=url_func, data =
json.dumps(request_data), headers = {'content-type':
'application/json'})
        res = urllib2.urlopen(req)
        return res.read()

    def recognize(self, filename):
        base64Image = self.getBase64(filename)
        #print base64Image
        json_data = { "image" : base64Image }
        #print json_data
        api_result = self.request("recognize", json_data)
        #print api_result
        print json.loads(api_result)

if __name__ == '__main__':
    from argparse import ArgumentParser

    parser = ArgumentParser()
    parser.add_argument("-s", "--server", action="store",
dest="host", default=SERVER_ADDRESS,
        help="Sets the endpoint for the server to call.",
required=False)
    parser.add_argument('image', nargs='+', help="Images to call
the server with.")


    args = parser.parse_args()

 faceRecClient = FaceRecClient(args.host)
    for image in args.image:
        faceRecClient.recognize(image)
```

Server.py

It reads CSV file to create Eigenfaces and also reads query image and sends it recognition.py to perform recognition

```
import cStringIO
import base64
try:
    from PIL import Image
except ImportError:
    import Image

# Flask imports:
from flask import Flask, request, request_finished, json, abort,
make_response, Response, jsonify
# facerec imports
# facerec imports:
import sys
sys.path.append("../../..")
from facerec.model import PredictableModel
from facerec.lbp import ExtendedLBP
from facerec.feature import SpatialHistogram
from facerec.distance import ChiSquareDistance
from facerec.classifier import NearestNeighbor

# logging
import logging
from logging.handlers import RotatingFileHandler

import recognition

app = Flask(__name__)

IMAGE_DECODE_ERROR = 10
IMAGE_RESIZE_ERROR = 11
PREDICTION_ERROR = 12
SERVICE_TEMPORARY_UNAVAILABLE = 20
UNKNOWN_ERROR = 21
```

```
INVALID_FORMAT = 30
INVALID_API_KEY = 31
INVALID_API_TOKEN = 32
MISSING_ARGUMENTS = 40

errors = {
    IMAGE_DECODE_ERROR : "IMAGE_DECODE_ERROR",
    IMAGE_RESIZE_ERROR  : "IMAGE_RESIZE_ERROR",
    SERVICE_TEMPORARY_UNAVAILABLE   :
"SERVICE_TEMPORARILY_UNAVAILABLE",
    PREDICTION_ERROR : "PREDICTION_ERROR",
    UNKNOWN_ERROR : "UNKNOWN_ERROR",
    INVALID_FORMAT : "INVALID_FORMAT",
    INVALID_API_KEY : "INVALID_API_KEY",
    INVALID_API_TOKEN : "INVALID_API_TOKEN",
    MISSING_ARGUMENTS : "MISSING_ARGUMENTS"
}

LOG_FILENAME = 'serverlog.log'
LOG_BACKUP_COUNT = 5
LOG_FILE_SIZE_BYTES = 50 * 1024 * 1024

def init_logger(app):
    handler = RotatingFileHandler(LOG_FILENAME,
maxBytes=LOG_FILE_SIZE_BYTES, backupCount=LOG_BACKUP_COUNT)
    handler.setLevel(logging.DEBUG)
    formatter = logging.Formatter('%(asctime)s - %(name)s -
%(levelname)s - %(message)s')
    handler.setFormatter(formatter)
    loggers = [app.logger, logging.getLogger('facerec')]
    for logger in loggers:
        logger.addHandler(handler)
def init_app(app):
    init_logger(app)

init_app(app)

@app.before_request
def log_request():
    app.logger.debug("Request: %s %s", request.method,
request.url)
class WebAppException(Exception):

    def __init__(self, error_code, exception, status_code=None):
        Exception.__init__(self)
        self.status_code = 400
        self.exception = exception
        self.error_code = error_code
        try:
            self.message = errors[self.error_code]
        except:
            self.error_code = UNKNOWN_ERROR
            self.message = errors[self.error_code]
        if status_code is not None:
            self.status_code = status_code

    def to_dict(self):
        rv = dict()
        rv['status'] = 'failed'
        rv['code'] = self.error_code
        rv['message'] = self.message
        return rv

   def __init__(self, error_code, status_code=None):
      self.error_code = error_code
      self.status_code = status_code

   def __call__(self, function):
      def returnfunction(*args, **kwargs):
         try:
            return function(*args, **kwargs)
         except Exception as e:
            raise WebAppException(self.error_code, e)
      return returnfunction

@app.errorhandler(WebAppException)
def handle_exception(error):
    app.logger.exception(error.exception)
    response = jsonify(error.to_dict())
    response.status_code = error.status_code
    return response
@ThrowsWebAppException(error_code = IMAGE_DECODE_ERROR)
def read_image(base64_image):
    enc_data = base64.b64decode(base64_image)
    file_like = cStringIO.StringIO(enc_data)
    im = Image.open(file_like)
    im = im.convert("L")
    return im

def preprocess_image(image_data):
    image = read_image(image_data)
    return image

# Get the prediction from the global model.
@ThrowsWebAppException(error_code = PREDICTION_ERROR)
def get_prediction(image_data):
        #global unknown
      image = preprocess_image(image_data)
      prediction = model.predict(image)

      return prediction
@app.route('/api/recognize', methods=['GET', 'POST'])
def identify():
    if request.headers['Content-Type'] == 'application/json':
            try:
                image_data = request.json['image']
            except:
                raise
WebAppException(error_code=MISSING_ARGUMENTS)
            prediction = get_prediction(image_data)
            response = jsonify(name = prediction)
            return response
            #print "test"
    else:
        raise WebAppException(error_code=INVALID_FORMAT)

if __name__ == '__main__':
    #print "=== Description ==="
    #print long_description
    # Parse the command line:
    from argparse import ArgumentParser

    parser = ArgumentParser()
    parser.add_argument("-t", "--train", action="store",
dest="dataset", default=None,
        help="Calculates a new model from a given CSV file. CSV
format: <person>;</path/to/image/folder>.", required=False)
    parser.add_argument("-a", "--address", action="store",
dest="host", default="0.0.0.0",
        help="Sets the endpoint for this server.",
required=False)
    parser.add_argument("-p", "--port", action="store",
dest="port", default=5000,
        help="Sets the port for this server.", required=False)
    parser.add_argument('model_filename', nargs='?',
help="Filename of the model to use or store")

args = parser.parse_args()

if args.dataset:
    model =
recognition.get_model_from_csv(filename=args.dataset,out_model_fi
lename=args.model_filename)
    else:
        model = recognition.load_model_file(args.model_filename)
    # Finally start the server:
    #print "=== Server Log (also in %s) ===" % (LOG_FILENAME)
    app.run(host=args.host, port=args.port, debug=True,
use_reloader=False, threaded=False)
```

Recognition.py:

It takes geiven query image from client and sends it to Server.py and after that provide results back to client.

```
import csv, os, sys
# Import PIL
try:
    from PIL import Image
except ImportError:
    import Image
# Import numpy:
import numpy as np
# Import facerec:
from facerec.feature import ChainOperator, Fisherfaces
from facerec.preprocessing import Resize
from facerec.dataset import NumericDataSet
from facerec.distance import EuclideanDistance
from facerec.classifier import NearestNeighbor
from facerec.model import PredictableModel
from facerec.validation import KFoldCrossValidation
from facerec.serialization import save_model, load_model

class PredictableModelWrapper(object):

    def __init__(self, model):
        self.model = model
        self.numeric_dataset = NumericDataSet()

    def compute(self):
        X,y = self.numeric_dataset.get()
        self.model.compute(X,y)

    def set_data(self, numeric_dataset):
        self.numeric_dataset = numeric_dataset

    def predict(self, image):
        prediction_result = self.model.predict(image)
        # Only take label right now:
        print prediction_result
        if prediction_result == 10000:
            return "Unknown Image"
        else:
            num_label = prediction_result[0]
            str_label =
self.numeric_dataset.resolve_by_num(num_label)
```

```
            return str_label

    def update(self, name, image):
        self.numeric_dataset.add(name, image)
        class_label = self.numeric_dataset.resolve_by_str(name)
        extracted_feature = self.feature.extract(image)
        self.classifier.update(extracted_feature, class_label)

    def __repr__(self):
        return "PredictableModelWrapper (Inner Model=%s)" %
(str(self.model))

def get_model(numeric_dataset, model_filename=None):
    feature = ChainOperator(Resize((128,128)), Fisherfaces())
    classifier = NearestNeighbor(dist_metric=EuclideanDistance(),
k=1)
    inner_model = PredictableModel(feature=feature,
classifier=classifier)
    model = PredictableModelWrapper(inner_model)
    model.set_data(numeric_dataset)
    model.compute()
    if not model_filename is None:
        save_model(model_filename, model)
    return model

def read_images(path, identifier, numeric_dataset):
    for filename in os.listdir(path):
        try:
            img = Image.open(os.path.join(path, filename))
            img = img.convert("L")
            img = np.asarray(img, dtype=np.uint8)
            numeric_dataset.add(identifier, img)
        except IOError, (errno, strerror):
            print "I/O error({0}): {1}".format(errno, strerror)
        except:
            print "Unexpected error:", sys.exc_info()[0]
            raise

def read_from_csv(filename):
    numeric_dataset = NumericDataSet()
    with open(filename, 'rb') as csvfile:
        reader = csv.reader(csvfile, delimiter=';',
quotechar='#')
        for row in reader:
            identifier = row[0]
            path = row[1]
            read_images(path, identifier, numeric_dataset)
    return numeric_dataset
```

```
def get_model_from_csv(filename, out_model_filename):
    numeric_dataset = read_from_csv(filename)
    model = get_model(numeric_dataset, out_model_filename)
    return model

def load_model_file(model_filename):
    load_model(model_filename)
```

Feature.py :

This files has a class that contains the implementation of Eigenfaces.

```
class PCA(AbstractFeature):
    def __init__(self, num_components=0):
        AbstractFeature.__init__(self)
        self._num_components = num_components

    def compute(self,X,y):
        # build the column matrix
        XC = asColumnMatrix(X)
        y = np.asarray(y)
        # set a valid number of components
        if self._num_components <= 0 or (self._num_components > 
XC.shape[1]-1):
            self._num_components = XC.shape[1]-1
        # center dataset
        self._mean = XC.mean(axis=1).reshape(-1,1)
        XC = XC - self._mean
        # perform an economy size decomposition (may still 
allocate too much memory for computation)
        self._eigenvectors, self._eigenvalues, variances = 
np.linalg.svd(XC, full_matrices=False)
        # sort eigenvectors by eigenvalues in descending order
        idx = np.argsort(-self._eigenvalues)
        self._eigenvalues, self._eigenvectors = 
self._eigenvalues[idx], self._eigenvectors[:,idx]
        # use only num_components
        self._eigenvectors = 
self._eigenvectors[0:,0:self._num_components].copy()
        self._eigenvalues = 
self._eigenvalues[0:self._num_components].copy()
        # finally turn singular values into eigenvalues
        self._eigenvalues = np.power(self._eigenvalues,2) / 
XC.shape[1]
        # get the features from the given data
        features = []
        for x in X:
            xp = self.project(x.reshape(-1,1))
            features.append(xp)
        return features

    def extract(self,X):
        X = np.asarray(X).reshape(-1,1)
        return self.project(X)

    def project(self, X):
        X = X - self._mean
        return np.dot(self._eigenvectors.T, X)

    def reconstruct(self, X):
        X = np.dot(self._eigenvectors, X)
        return X + self._mean

    @property
    def num_components(self):
        return self._num_components

    @property
    def eigenvalues(self):
        return self._eigenvalues

    @property
    def eigenvectors(self):
        return self._eigenvectors

    @property
    def mean(self):
        return self._mean

    def __repr__(self):
        return "PCA (num_components=%d)" % (self._num_components)
```

Classifier.py

Class contains the nearest neighbour calculation.

```
class NearestNeighbor(AbstractClassifier):
    """
    Implements a k-Nearest Neighbor Model with a generic distance
metric.
    """
    def __init__(self, dist_metric=EuclideanDistance(), k=1):
        AbstractClassifier.__init__(self)
        self.k = k
        self.dist_metric = dist_metric
        self.X = []
        self.y = np.array([], dtype=np.int32)

    def update(self, X, y):
        """
        Updates the classifier.
        """
        self.X.append(X)
        self.y = np.append(self.y, y)

    def compute(self, X, y):
        self.X = X
        self.y = np.asarray(y)

    def predict(self, q):
        distances = []
        for xi in self.X:
            xi = xi.reshape(-1,1)
            d = self.dist_metric(xi, q)
            distances.append(d)
        if len(distances) > len(self.y):
            raise Exception("More distances than classes. Is your
distance metric correct?")
        distances = np.asarray(distances)
        # Get the indices in an ascending sort order:
        idx = np.argsort(distances)
        # Sort the labels and distances accordingly:
        sorted_y = self.y[idx]
        sorted_distances = distances[idx]
        # Take only the k first items:
        sorted_y = sorted_y[0:self.k]
        sorted_distances = sorted_distances[0:self.k]
        #sorted_distances = 1134.04873217
        # Make a histogram of them:
        hist = dict((key,val) for key, val in
enumerate(np.bincount(sorted_y)) if val)
        # And get the bin with the maximum frequency:
        predicted_label = max(hist.iteritems(),
key=op.itemgetter(1))[0]
        # A classifier should output a list with the label as
first item and
        # generic data behind. The k-nearest neighbor classifier
outputs the
        #global unknown
      if sorted_distances > 700 :
            return 10000
      else:
            return [predicted_label, { 'labels' : sorted_y,
'distances' : sorted_distances }]

        #distance = distances[0]
        #distance = [6, { 'labels' : sorted_y, 'distances' :
sorted_distances }][0]
        #print '\n'.join(str(p) for p in sorted_y)
        #return [predicted_label]
        #test = [none]
        #return [predicted_label, { 'labels' : sorted_y,
'distances' : sorted_distances }]

    def __repr__(self):
        return "NearestNeighbor (k=%s, dist_metric=%s)" %
(self.k, repr(self.dist_metric))
```