Automated Attendance System Using Face Recognition

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

In

Computer Science and Engineering/Information Technology

By

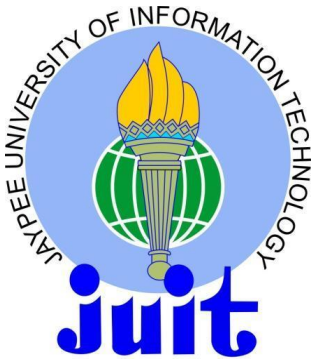
Mayank Sharma (141336)

Piyush Thakur (141339)

Under the supervision of

Mr. Nitin Kumar

To



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

We hereby declare that the work presented in this report entitled **"Automated attendance system using face recognition "** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2018 to May 2018 under the supervision of **Mr.Nitin Kumar,** Assistant Professor in the Department of Computer Science.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Mayank Sharma 141336

Piyush Thakur 141339

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Mr. Nitin Kumar
Assistant Professor
Department of Computer Science
Dated:

ABSTRACT

The goal of this project is to create the automated attendance system using face recognition. When a user takes a picture of a human, our application searches related information in a database using image recognition. Since a user of the application can take a picture under different circumstances, the used image recognition algorithm had to be invariant to changes in illumination and view point. The execution of the algorithm had to run on the mobile phone, so there was need for a lightweight image recognition algorithm. A couple of these invariants can be grouped as a feature vector, identifying an image uniquely. By computing the distances between the vectors of an unknown image and known database images, a best match can be selected.

Android is flexible and provides many tools for developing applications. This allowed to develop our museum guide application in a limited amount of time. We explored, evaluated and used many of Android's possibilities.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Attendance systems of old practices are not quite efficient now a days for keeping track on student's attendance. Student enrollment in schools and colleges increasing every year and taking each student attendance plays a very vital role. So, it is necessary to discuss the effective system which records the attendance of a student automatically. [17,18]

Maintaining the attendance is very important in all the schools/colleges for checking the performance of students. Every school/college has its own method in this regard. Some are taking attendance of students manually using attendance registers or marking attendance sheets or file based approach and some have adopted the methods of automatic attendance using some biometric techniques. But in these methods, students have to wait for a long time in making a queue at the time they enter inside the classroom.[17,18]

Many biometric systems are available in the market but the key authentications are same in all of the techniques. Every biometric system consists of enrollment process in which the unique features of a person is stored in the database and after that, there are some processes of identification and verification of the person. These two processes compare the biometric feature of a person with previously stored template captured at the time of enrolment of a student. Biometric templates can be of many types like Fingerprints , Eye Iris, voice etc. Our system uses the face recognition approach for the automatic attendance of the students in the classroom environment without student intervention. The purpose of developing the new attendance management system is to computerize the traditional methods of taking the attendance. Therefore, in order to drag the attention of students and make them interactive in observing technologies, we

try to move on to the latest upcoming trends on developing attendance systems. This is the reason for college/school attendance management system to come up with an approach that ensures a strong contribution of students in classrooms.[17,20,21]

To track the attendance of the students, we have introduced the attendance management system. With the introduction of this attendance system, skipping classes for students without the staff's knowledge have become difficult. Attendance management system is to count the number of students and urge students to attend the classes on time, so as to improve the quality of teaching.[18,20]

Usually, a roll-call is taken to determine whether the student is present in the class or not, which usually wastes a lot of time. In recent years, with the emerging technology and with the development of deep learning, face recognition has made great achievements, which leads us to a new way of thinking to solve the problem of student's enrollment. So, in order to save time, the idea to count the number of students in a class automatically based on face recognition is incorporated. This system is developed by using face recognition technique which is used to detect the face of an individual. There are many different face recognition algorithms introduced to increase the efficiency of the system. The system provides an increased accuracy due to the use of a large number of features like Shape, color, LBP[11], wavelet, Auto-Correlation etc. of the face. However, the face recognition still remains a challenging problem for us because of its fundamental difficulties regarding various factor like illumination changes, face rotation, facial expression etc.[21]

1.2 Problem Statement

When there are so many students in a school/college, it becomes more and more difficult to mark attendance for each student and it is time consuming too. The Existing system of any institute is a manual entry for the students. This system faces the issue of wastage of time and also becomes complicated when the strength is more than the usual. Here, the attendance is being carried out in the hand written registers. It is very tedious job for us to maintain the record of the user.

Whenever we have to measure the performance of students, finding and calculating the average of the attendance of each enrolled student is also a very complicated task for us. The human effort is more here. The retrieval of the information is not a piece of cake as the records are maintained in the hand written registers. This existing system requires correct feed on input into the respective field. Therefore we are in a need of an automated system for marking and maintaining attendance of the students. Let us suppose that the wrong inputs are entered, the application resist to work. So, the user finds it difficult to use the existing system.[17]

1.3 Objectives

Our objective is to detect unique faces with the help of mobile camera amidst the other natural components like walls, backgrounds etc. and to extract the unique features faces amongst other face characteristics such as beard, spectacles etc. of a face useful for face detection and recognition.[18]

1.4 Methodology

In our proposed system, the system is instantiated by the mobile .After it triggers then the system starts processing the image of the students for which we want to mark the attendance. [18]

Image Capturing phase is one in which we capture the image of the students. This is the very basic phase from which we start initializing our system. We capture an image from our camera which predominantly checks for certain constraints like lightning, spacing, density, facial expressions etc. The captured image is resolute according to our requirements. Once it is resolute, we make sure it is either in .png or .jpeg format.

We take different frontal postures of an individual so that the accuracy can be attained to the maximum extent. This is the training database in which we classify every individual based on labels. For the captured image, from an every object we detect only frontal faces. This detects only face and removes every other parts since we are exploring the features of

faces only. These detected faces are stored somewhere in the database for further enquiry. Features are extracted in the extraction phase.

The detected bounding boxes are further queried to look for features extraction and the extracted features are stored in a matrix. For every detected phase, this feature extraction is done. Features that we look here are shape, edge, color, wavelet, auto-correlation and LBP [9, 11]. Face is recognized once we completed the extracting features. The features which is already trained with every individual is compared with the detected faces features and if both features match, then it is recognized. Once it recognizes, it is going to update in the student attendance database. Once the process is completed, the testing images remains. [18]

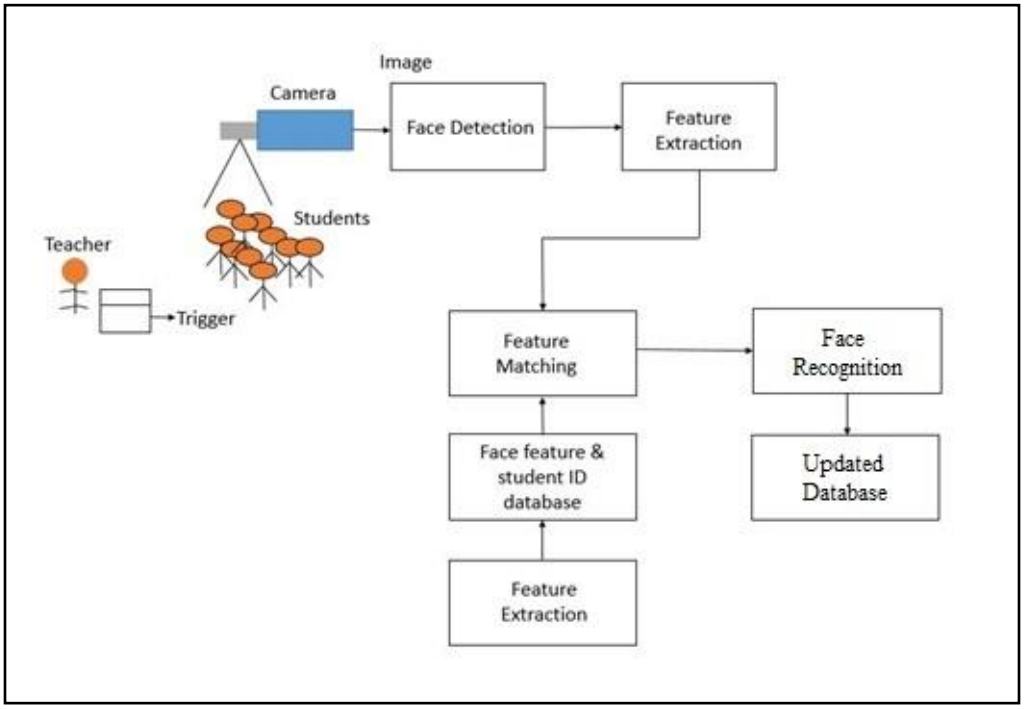


Figure 1.4.1**:** System Model of face detection & recognition

Chapter 2

Literature Survey

One of the most successful applications of image analysis and understanding, face recognition has recently received a significant attention, especially during the past few years. In addition to this, the problem of machine recognition of human faces continues to attract researchers from disciplines such as image processing, pattern recognition, neural networks, computer vision, computer graphics and psychology. The strong need for user-friendly systems that can secure our assets and protect our privacywwithout losing our identity in a sea of numbers is obvious.

We as humans use faces to recognize and identify our friends and family. Computers can now also identify people automatically using stored information such as figure, iris or face to identify a particular person. Earlier many face recognition algorithms were used to achieve fully automated face identification process. The first face recognition system was created in the 1960s. It was not fully automated and it required anual inputs of the location of the eyes, ears, nose and mouth on the images then it calculates a distance to some common point then it compares it to the stored data. The still image problem has several inherent advantages and disadvantages. For applications such as driver's license, due to the controlled nature of the image acquisition process, the segmentation problem is rather easy. However, if only a static picture of an airport scene is available, automatic location and segmentation of a face could pose serious challenges to any segmentation algorithm.

On the other hand, if a video sequence is available, segmentation of a moving person can be more easily accomplished using motion as a cue. But the small size and low image quality of faces captured from video can significantly increase the difficulty in recognition. Face recognition and sometime is called face identifying is simply putting a label to known faces just like human as mentioned above, we learn the faces of our family

and celebrities just by looking at their faces. Since the 1970s there was many techniques and algorithms were developed for a machine to learn to recognize known faces. Most of the recent techniques involve at least three steps:

• Face detection:

• Face preprocessing:

• Face recognition

2.1 Face Detection

Face detection is a process of locating a face inside an image frame, regardless of the identity of that face. Before recognizing a face, it is first essential to detect and extract the faces from the original pictures. Face Detection target on finding the faces (area and size) in an image and probably extract them to be used by the face recognition algorithm. In recent years, many methods are proposed for detecting the face [20].

In face detection methods, those who are depending on training sets to capture the huge unevenness in facial features have enticed much attention and given the best results. Generally these methods scan the input picture at all potential area and scales then as the sub windows either as non-face or face. Viola and Jones [1] presented an effective detection technique using *Haar-like features* and AdaBoost as a quick training algorithm.For recognizing a face, the algorithms compare only faces. Any other element in the picture that is not part of a face deteriorates the recognition.

There are several existing algorithms for detecting faces.Prior to year 2000 there were many techniques for face detection, however they were mostly unreliable, slow and require manual inputs. In 2001 Viola and Jones[1] invented the Haar-based-cascade[15] Classifier that revolutionize the face detection method. It can detect objects in real time with an accuracy of 95%. It works not only for frontal face view but it can detect faces from side view as well.

2.1.1 Haar-cascade classifier

Haar-cascade is a method, invented by Viola and Jones [1], which trains a machine learning for detecting objects in a picture. In this context, it can be used to detect faces. The basic idea of the Haar-based[15] face detector is that if you look at most frontal faces, the region with the eyes should be darker than the forehead and cheeks, and the region with the mouth should be darker than cheeks, and so on.

It typically performs about 20 stages of comparisons like this to decide if it is a face or not, but it must do this at each possible position in the image and for each possible size of the face, so in fact it often does thousands of checks per image.. The name of this method is composed of two important words, Haar and Cascade. Haar belongs to Haar-like features which is a weak classifier and will be used for the face recognition.

A Haar-like feature is a rectangle which is split into two, three or four rectangles. Each rectangle is black or white. Figure 5 shows the different possible features. A Haar-cascade needs to be trained with various positive and negative pictures. The objective is to extract the combination of these features that represents a face. While a positive picture contains the object which has to be recognized, a negative picture represents a picture without the object.

In the context of face detection, a positive picture possesses a face, and a negative picture does not. This machine learning requires grayscale pictures. The intensity of gray will be used to detect which feature is represented. These features can be found by calculating the sum of the dark pixels in an area subtracted by the sum of the bright pixels.

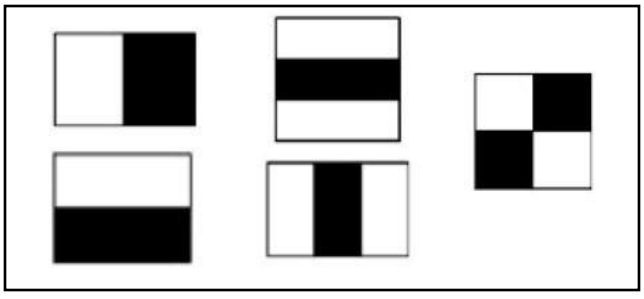


Figure 2.1.1: The 5 Haar-like features used for detecting faces [15]

The basic principles that Viola and Jones [1] method is based on are as follows:

• Images used in the integral representation that allows a machine to calculate the necessary object features (in this context, face features)

• Using Haar-like features, the desired feature of the face can be found.

• Adaptive Boosting used to select the most suitable characteristics for the desired object to this part of the image.

• All the features are input to the classifier, which gives the result true or false.

The extracted combination of features from the training part will be used for detecting faces in a picture. To detect a face in an unknown picture the combination of the features will be researched. The features are tried to be matched only in a block of pixels defined by a scale. Each feature of the combination will be tried to be matched one by one in the block. If one of the features does not appear in the block, the research in it will be stopped. The remaining features will not be tested because the machine concludes that there is no face in this block. Then, a new block is taken, and the process will be repeated.

The 5 Haar-like features used for detecting faces pixels with the researched combination in cascade which explains the second word in the name of the method. This method is efficient to detect an image without faces because only a few tests need to be run to infer that the image does not contain a face. A face is consequently detected when each feature of the combination has been recognized correctly in a block. We can see that the eyes are darker than the cheeks and the middle of the nose is brighter. All these features which were extracted from the training are used to find a pattern to represent a face. The process will proceed block by block until the last one. After checking the last block, the scale is increased, and the detection process starts again. The process is repeated several times with different scales to detect faces of different size. Only few pixels are different between two neighbor blocks. Therefore, each time a face is detected in a picture, the same face is detected in different blocks.

All the detected faces that concern the same person are merged and are considered as one at the end of the entire process. The accumulation of these weak classifiers builds a face detector able to detect faces very fast with a suitable accuracy. A Haar-cascade classifier has to be trained only once. Thus, it is possible to create one's own Haar-cascade or use one which has already been trained.

The downside of Haar-based-cascade is the training of the classifier is very slow, and it can take up to a week using a modern PC.

2.1.2 Local Binary Patterns (LBP)

The Local Binary Pattern operator is also known as LBP and was first presented by [7] for byte adaptation [8]. The Local Binary Pattern (LBP) is a basic, effective and dominant texture operator, which labels the pixels of a picture by thresholding the neighbouring of each pixel, resulting in a binary number as shown in Figure 3.5.2. It was initially introduced by Ojala et al. [7]. For each pixel, the algorithm considers the eight (8) neighbouring pixels. Then based on the gray-scale value of the selected pixel, it allocates the neighboring pixels the value of 0 or 1. Therefore, every pixel will have a string of binary values. Figures 3.5.3 shows an example of the calculation.

$$\mathrm{LBP}(x_p,y_p)= \sum_{n=0}^{7} s(in - ip)2^n$$

Where (x_p, y_p) is the pixel of an image, n signifies the neighboring pixel, (in) and (ip) the individual gray level of neighboring and central pixel, and s(x) can be described by:

$$S(x) = 1,\ x \geq 0$$
$$S(x) = 0,\ x < 0$$

The bit is attained for every neighbouring pixel and is used in a pre-defined order to create a certain result. The final result will be in between 0 and 255, using 8 neighbouring pixels. Figure 3.5.3 shows the whole Local Binary Pattern operator extraction process.

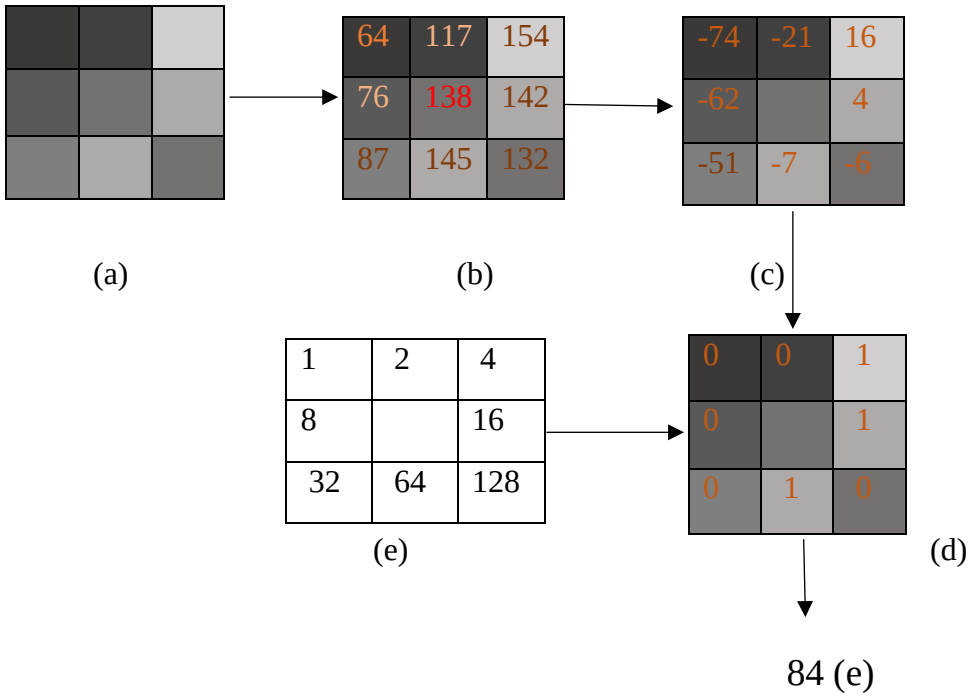


Figure 2.1.2: LBP calculation example [10]

The Local Binary Pattern extraction process:

(a) The picture which is to be fragment so that it can be processed.

(b) Gray levels are shown in the image.

(c) Shows subtraction results from edged pixels and the center one.

(d) Zero is assign to the pixel where subtraction results is less than 0 and 1 is assign if the subtraction results larger or equal zero (0).

(e) Binary matrix values are shown.

In the last step, the binary matrix values in (e) are added together for every cell where the value is one (1) in (d).

• Comparison between LBP vs Haar-cascade classifier

The basic idea of the LBP-based face detector is similar to the Haar-based one, but it uses histograms of pixel intensity comparisons, such as edges, corners, and flat regions. Haar-like based classifier was very slow to detect a face in the frame Table.[3]

Haar Fps	LBP Fps	Status
0.98	4.53	Face in frame
0.99	4.52	Face in frame
0.98	4.57	Face in frame
0.98	4.57	Face in frame
0.99	4.53	Face in frame
0.98	4.57	Face in frame
1.43	5.74	No Face in frame
1.20	5.75	No Face in frame
1.20	5.13	No Face in frame
1.40	5.13	No Face in frame
1.32	5.74	No Face in frame
1.20	5.13	No Face in frame
1.1375	**4.9925**	**Average**

Table 2.1: Shows the comparison between LBP vs Haar-cascade classifier.[3]

2.2 Face Pre-processing

Any of the previous methods can be used for extracting faces from input pictures. The next step is to pre-process these faces in order to make the training phase easier and improve the probability to recognize a person correctly. The training data will be standardized. Not all the pictures have the same zoom on the face and have maybe not all the same size. Most of the algorithms for facial recognition require the same size for the entire training set. Pre-processing includes different modifications. First of all, the faces need to be centered in the picture in the same way. The location of the two eyes and the nose is often used as a landmark for centering faces. The aim is to have the eyes at the same level and the nose at the same position for all images. To apply these modifications, the coordinates of the landmarks are needed. For that, it is possible to use a Haar-cascade classifier for detecting nose and eyes.[3]

After detecting a face in the frame, we can now process the face inside the green rectangle. Face recognition is susceptible to changes in lighting conditions, face orientation, face expression, so it is paramount to diminish these differences as much as possible. There are numerous techniques to eliminate those issues. Some of these techniques are:-

• Geometrical transformation and cropping: This procedure includes resizing of the image and rotating the image as well as removing background.

Figure 2.2.1: Face image converted to gray scale and cropped [3]

•Histogram equalization: This process standardizes the brightness and contrast of the image.

Figure 2.2.2: Histogram of the image equalized [3]

• Smoothing:In this process, the image noise is eliminated by applying some filters.

Figure 2.2.3: Filter applied on the face [3]

• Elliptical mask: The elliptical mask removes some remaining background.

Figure 2.2.4: Elliptical mask applied on the image [3]

Processing images can be computationally expensive in a PC or laptop, similarly in mobile devices. It requires higher processing power. Therefore, minimalizing the image processing in the mobile device is a must to achieve a real-time face recognition system.

2.3 Face Recognition

We will discuss current developments in face recognition in upcoming various sections. In Section 1, we briefly review issues that are relevant from a psychophysical point of view. Section 2 will provide a detailed review of recent developments in face recognition techniques using still images. In Section 3, face recognition techniques based on video are reviewed. Data collection and performance evaluation of face recognition algorithms are addressed in Section 4 with descriptions of representative protocols. In Section 5, we discuss two important problems in face recognition that can be mathematically studied, lack of robustness to illumination and pose variations, and we review proposed methods of overcoming these limitations.

2.3.1 Psychophysics/Neuroscience issues relevant to face recognition

Human recognition processes utilize a broad spectrum of stimuli, obtained from many, if not all, of the senses (visual, auditory, olfactory, tactile, etc.). In many situations, contextual knowledge is also applied, for example, surroundings play an important role in recognizing faces in relation to where they are supposed to be located. It is futile to even attempt to develop a system using existing technology, which will mimic the remarkable face recognition ability of humans.[16]

Many studies in psychology and neuroscience have direct relevance to engineers interested in designing algorithms or systems for machine recognition of faces. For example, findings in psychology about the relative importance of different facial features have been noted in the engineering literature. On the other hand, machine systems provide tools for conducting studies in psychology and neuroscience.

It is traditionally believed that face recognition is a dedicated process different from other object recognition tasks.

Both holistic and feature information are crucial for the perception and recognition of faces. Hair, face outline, eyes and mouth (not necessarily in this order) have been determined to be important for perceiving and remembering faces. It has long been informally observed that photographic negatives of faces are difficult to recognize. However, relatively little work has explored why it is so difficult to recognize negative images of faces. Experiments were conducted to explore whether difficulties with negative images and inverted images of faces arise because each of these manipulations reverses the apparent direction of lighting, rendering a top-lit image of a face apparently lit from below. It was demonstrated in that bottom lighting does indeed make it harder to identity familiar faces. Based on neurophysiological studies, it seems that analysis of facial expressions is accomplished in parallel to face recognition. Some prosopagnosic patients, who have difficulties in identifying familiar faces, nevertheless seem to recognize expressions due to emotions.[16]

2.3.2 Face recognition from still images

Face recognition involves three key steps:
(1)Detection and rough normalization of faces.
(2)Feature extraction and accurate normalization of faces.
(3)Identification and/or verification.

Sometimes, different subtasks are not totally separated. For example, the facial features (eyes, nose, mouth) used for face recognition are often used in face detection.

Though fully automatic face recognition systems must perform all three subtasks, research on each subtask is critical. This is not only because the techniques used for the individual subtasks need to be improved, but also because they are critical in many different applications. For example, face detection is needed to initialize

face tracking, and extraction of facial features is needed for recognizing human emotion, which is in turn essential in Human-Computer Interaction (HCI) systems. Isolating the subtasks makes it easier to assess and advance the state of the art of the component techniques. Earlier face detection techniques could only handle single or a few well-separated frontal faces in images with simple backgrounds, while state-of-the-art algorithms can detect faces and their poses in cluttered backgrounds.

Key Steps Prior to Recognition: Face Detection and Feature Extraction

The first step in any automatic face recognition systems is the detection of faces in images. Depending on the type of classification system, features can be local features such as lines or fiducial points, or facial features such as eyes, nose, and mouth. Face detection may also employ features, in which case features are extracted simultaneously with face detection. Feature extraction is also a key to animation and recognition of facial expressions. Without considering feature locations, face detection is declared successful if the presence and rough location of a face has been correctly identified.

Three types of feature extraction methods can be distinguished:

(1) Generic methods based on edges, lines, and curves.

(2) Feature template based methods that are used to detect facial features such as eyes.

(3) Structural matching methods that take into consideration geometrical constraints on the features.[16]

These methods have difficulty when the appearances of the features change significantly, for example, closed eyes, eyes with glasses, open mouth. A template-based approach in detecting the eyes and mouth in real images was introduced. This method is based on matching a predefined parameterized template to an image that contains a face region. Two templates are used for matching the eyes and mouth respectively.

The shape model (mean shape, orthogonal mapping matrix **P***s* and projection vector **b***s*) is generated by representing each set of landmarks as a vector and by applying

Principal Component Analysis (PCA) to the data. Then, after each sample image is warped so that its landmarks match the mean shape, texture information can be sampled from this shape-free face patch. PCA to this data leads to a shapefree texture model (mean texture, **P***g* and **b***g*). To explore the correlationn between the shape and texture variations, a third PCA is applied to the concatenated vectors (**b***s* and **b***g*) to obtain the combined model in which one vector **c** of appearance parameters controls both the shape and texture of the model. To match a given image and the model, an optimal vector of parameters (displacement parameters between the face region and the model, parameters for linear intensity adjustment, and the appearance parameters **c**) are searched by minimizing the difference between the synthetic image and the given one. After matching, a best-fitting model is constructed that gives the locations of all the facial features and can be used to reconstruct the original images.

2.3.3 Recognition from Intensity Images

Many methods of face recognition have been proposed during the past 30 years. Face recognition is such a challenging yet interesting problem that it has attracted researchers who have different backgrounds: psychology, pattern recognition, neural networks, computer vision & computer graphics. It is due to this fact that the literature on face recognition is vast and diverse. The usage of a mixture of techniques makes it difficult to classify these systems based purely on what types of techniques they use for feature representation or classification. To have a clear and high-level categorization, we instead follow a guideline suggested by the psychological study of how humans use holistic and local features. Specifically, we have the following categorization:-

(1) **Holistic matching methods**. These methods use the whole face region as the raw input to a recognition system. One of the most widely used representations of the face region is eigenpictures , which are based on principal component analysis.

(2) **Feature-based (structural) matching methods.** Typically, in these methods, local features such as the eyes, nose, and mouth are first extracted and their locations and local statistics (geometric and/or appearance) are fed into a structural classifier.

(3) **Hybrid methods.** Just as the human perception system uses both local features and the whole face region to recognize a face, a machine recognition system should use both. One can argue that these methods could potentially offer the best of the two types of methods.

2.3.4 Face recognition from image sequences

A typical video-based face recognition system automatically detects face regions, extracts features from the video, and recognizes facial identity if a face is present. In surveillance, information security, and access control applications, face recognition and identification from a video sequence is an importan problem. Face recognition based on video is preferable over using still images, motion helps in recognition of familiar faces when the images are negated, inverted or threshold. It was also demonstrated that humans can recognize animated faces better than randomly rearranged images from the same set. Though recognition of faces from video sequence is a direct extension of still image-based recognition, in our opinion, true video based face recognition techniques that coherently use both spatial and temporal information started only a few years ago and still need further investigation. Significant challenges for video-based recognition still exist; we list several of them here.[16]

(1) **The quality of video is low**. Usually, video acquisition occurs outdoors (or indoors but with bad conditions for video capture) and the subjects are not cooperative; hence there may be large illumination and pose variations in the face images. In addition, partial occlusion and disguise are possible.

(2)**Face images are small.** Again, due to the acquisition conditions, the face image sizes are smaller (sometimes much smaller) than the assumed sizes in most

still-image-based face recognition systems. Small-size images not only make the recognition task more difficult, but also affect the accuracy of face segmentation, as well as the accurate detection of the fiducial points/landmarks that are often needed in recognition methods.

(3) **The characteristics of faces/human body parts.** During the past eight years, research on human action/behavior recognition from video has been very active and fruitful. Generic description of human behavior not particular to an individual is an interesting and useful concept. One of the main reasons for the feasibility of generic descriptions of human behavior is that the intraclass variations of human bodies, and in particular faces, is much smaller than the difference between the objects inside and outside the class. For the same reason, recognition of individuals within the class is difficult. For example, detecting and localizing faces is typically much easier than recognizing a specific face. Before we examine existing video-based face recognition algorithms, we briefly review three closely related techniques: face segmentation and pose estimation, face tracking, and face modeling. These techniques are critical for the realization of the full potential of video-based face recognition.

(4) **Face and Feature Tracking.** After faces are located, the faces and their features can be tracked. Face tracking and feature tracking are critical for reconstructing a face model (depth) and feature tracking is essential for facial expression recognition and gaze recognition. Tracking also plays a key role in spatiotemporal based recognition methods which directly use the tracking information. In its most general form, tracking is essentially motion estimation. For images like faces, some regions are too smooth to estimate flow accurately, and sometimes the change in local appearances is too large to give reliable flow. Fortunately, these problems are alleviated thanks to face modeling, which exploits domain knowledge. In general, tracking and modeling are dual processes: tracking is constrained by a generic 3D model or a learned statistical model under deformation, and individual models are refined through tracking. Face tracking can be roughly divided into three categories:

(1) **Head tracking**, which involves tracking the motion of a rigid object that is performing rotations and translations.

(2) **Facial feature tracking**, which involves tracking non-rigid deformations that are limited by the anatomy of the head, that is, articulated motion due to speech or facial expressions and deformable motion due to muscle contractions and relaxations.

(3)**Complete tracking**, which involves tracking both the head and the facial features

2.3.5 Evaluation of face recognition systems.

Given the numerous theories and techniques that are applicable to face recognition, it is clear that evaluation and benchmarking of these algorithms is crucial. One of the most important facts learned in these evaluations is that large sets of test images are essential for adequate evaluation. It is also extremely important that the samples be statistically as similar as possible to the images that arise in the application being considered. Scoring should be done in a way that reflects the costs of errors in recognition. In planning an evaluation, it is important to keep in mind that the operation of a pattern recognition system is statistical, with measurable distributions of success and failure. These distributions are very application-dependent , and no theory seems to exist that can predict them for new applications. This strongly suggests that an evaluation should be based as closely as possible on a specific application.

2.3.6 Two issues in face recognition: Illumination and Pose variation

In this section, we discuss two important issues that are related to face recognition. However, face recognition in an uncontrolled environment is still very challenging. Though many existing systems build in some sort of performance invariance by applying preprocessing methods such as histogram equalization or pose learning, significant illumination or pose change can cause serious performance degradation. In addition, face images can be partially occluded, or the system may need to recognize a person from an

image in the database that was acquired some time ago. These problems are unavoidable when face images are acquired in an uncontrolled, uncooperative environment, as in surveillance video clips.

The Illumination Problem in Face Recognition

The illumination problem is illustrated in Figure 4, where the same face appears different due to a change in lighting.The changes induced by illumination are often larger than the differences between individuals, causing systems based on comparing images to misclassify input images.

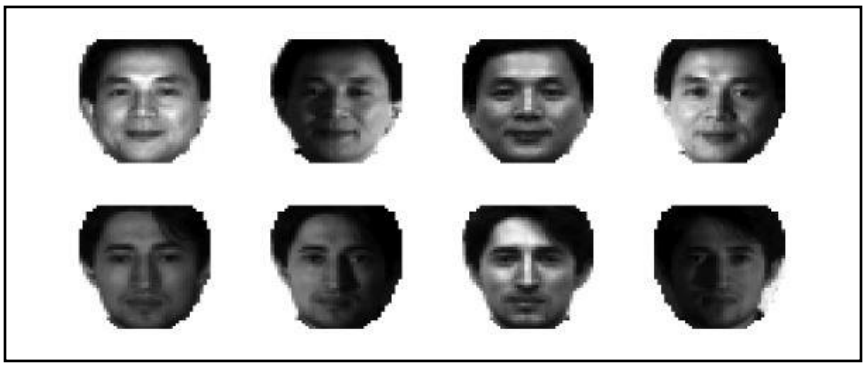


Figure 2.3.5: In each row, the same face appears differently under different illuminations [16]

The Pose Problem in Face Recognition

It is not surprising that the performance of face recognition systems drops significantly when large pose variations are present, in the input images. When illumination variation is also present, the task of face recognition becomes even more difficult. Here we focus on the out of plane rotation problem, since in-plane rotation is a pure 2D problem and can be solved much more easily.

2.4 Facial recognition's algorithms

After collecting enough images for the person (Training Set), we can now use one of many algorithms for training the system to learn the face. Like most algorithm in machine learning, training of system must be completed first. There are several approaches for recognizing a face. The algorithm can use statistics, try to find a pattern which represents a specific person or use a convolutional neural network. These different approaches can be observed through the explanations of different algorithms. Some of the face recognition algorithms are the flowing:

- Eigenfaces[15].
- Fisherfaces (also referred to as Linear Discriminant Analysis (LDA). [14]
- Local Binary Pattern (LBP) histograms. [7]

2.4.1 Eigenfaces

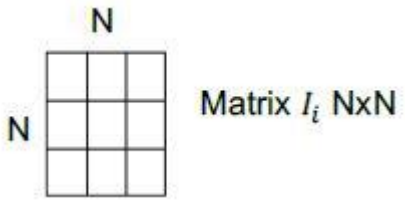
Eigenfaces [15] is a method for performing facial recognition based on a statistical approach. The aim of this method is to extract the principal components which affect the most the variation of the images. This is a holistic approach, the treatment for predicting a face is based on the entire training set. There is no specific treatment between images from two different classes.

A class represents a person. Pre-processed pictures with grayscale are required for training the machine learning. Each pixel of a picture represents one dimension, it means a 96x96 pixels image is represented into 96 x 96 = 9216 dimensions. Eigenfaces is based on Principal Component Analysis (PCA) [10] for reducing the number of dimensions while preserving the most important information. The training part of Eigenfaces is to calculate the eigenvectors and the related eigenvalues of the covariance matrix of the training set.

Various Steps of implementing Eigenfaces algorithm:-

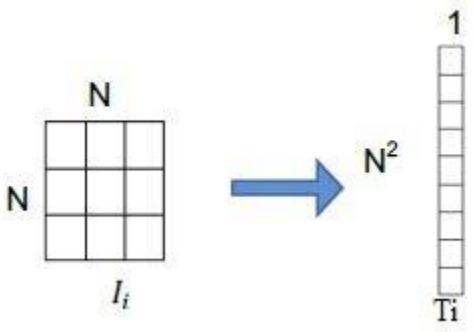
Step 1- Transform images into matrix.

Each pixel of an image represents a number. Thus, we can easily represent them into a matrix NxN where each item of the matrix is a pixel. Each training image becomes a matrix I (I_1, I_2,…., I_m where m equals number of images)



Step 2- Adapt the matrix I_i into vector T_i.

A matrix is a high-dimensional space compared to a vector which is a lower-dimensional space. Therefore, each row of the matrix I_i (will be concatenated and then transposed for representing the vector Ti.



Step 3 - Calculate the average of the vectors T_i.

$$\Psi = \frac{1}{M}\sum_{i=1}^{M} T_i$$

The sum of each vector T_i is calculated, and then the sum is divided by the number of images m which gives the vector Ψ representing the average.

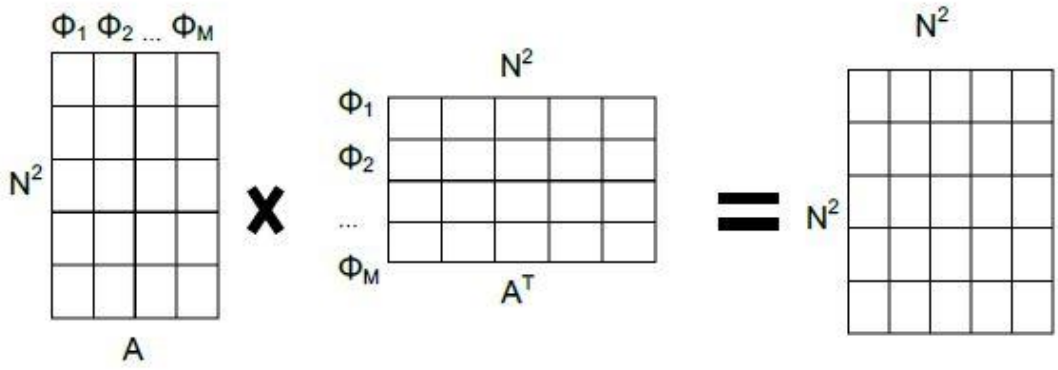
Step 4 - Subtract the average from the vector T_i.

$\Phi_i = T_i - \Psi$

Each image represented by the vector T_i will subtract the average of all the pictures. The result of the subtractions is represented by the vector Φ_i.

Step 5 - Compute the covariance matrix C.

$$C = \frac{1}{M}\sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T$$



Then, the covariance is calculated based on the Φ_n. The vectors Φ_n are grouped to represent the matrix A. The covariance matrix C is computed by multiplying the matrix A with the transposition of the matrix A called A^T. The relation between a matrix A, its eigenvectors and its eigenvalues is represented with the formula below:

$Av = \lambda v \; or \; (A - \lambda \, I_n) \, v = 0$

$v = eigenvector$

$\lambda = eigenvalue$

$I_n = identity \; matrix \; of \; A$

First, the eigenvalues need to be calculated. Then, the eigenvalues will be used for computing the eigenvectors. The following formula is used for obtaining the eigenvalues.

$\det(A - \lambda I_2) = 0$

Step 6 - Calculate the eigenvectors with their related eigenvalues.

There are two options for computing the eigenvectors. Either we calculate the eigenvectors u_i of AA^T or the eigenvectors V_i of A^TA. These two ways have the same eigenvalues, and the relation between the eigenvectors is $u_i = Av_i$. AA^T gives a matrix N^2xN^2 as a result in contrast to A^TA which gives a matrix MxM. Most of the time we would prefer the second one because usually, the number of pictures in the training set is smaller than the number of pixels. Therefore, it is faster to accomplish the calculation. A^TA can have a maximum of M eigenvalues and eigenvectors contrary to AA^T which can reach a number of N^2 eigenvalues and eigenvectors. The M eigenvalues represent the biggest eigenvalues included in N^2. The eigenvectors u_i need to be normalized, and the normalization has to equal 1, $\| u_i \| = 1$.

Step 7 - K eigenvectors.

The M eigenvectors are sorted in descending order based on the eigenvalues. Only K eigenvectors are kept. K is smaller than M and is decided by the user of the algorithm. All the training pictures can be represented by a combination of the K eigenvectors.

$$\Phi_i = \sum_{j=1}^{K} w_j u_j, (w_j = u_j^T \Phi_i)$$

K = number of eigenvectors

u_j = eigenvectors at the index j

Φ_i = Image i - the average

Each eigenvector which is also called eigenface represents a part of each image in the training data. An image can be decomposed through each eigenface.

2.4.2 Fisherfaces

Since Eigenfaces' maximization of both in-class and between-class scatter is undesirable, we wish to implement a method that maximizes the scatter between classes, while minimizing it within a class. This algorithm is a modification of Eigenfaces, thus also uses Principal Components Analysis(PCA)[10]. The main modification is that Fisherfaces takes into consideration classes. As it has been said previously, Eigenfaces

does not make the difference between two pictures from different classes during the training part. Each picture was affected by the total average.

The Fisherfaces[14] approach is based on linear sub-spaces. It also uses a holistic approach. If we can train a specific face using different images of the same face under arbitrary lighting conditions, we can extract the underlying static face and represent it as a 3D linear subspace, indifferent to lighting directions, in a Lambertian surface. This highlights one of the most significant differences between the Eigenfaces method and the Fisherfaces method, specifically that the latter uses a labeled training set in order to gain a more accurate classification.

Since the training set is labeled, we can use class specific linear methods to reduce the dimensionality of the feature space. In Fisherspaces, a specific method of LDA (linear discriminant analysis) -Fisher's Linear Discriminant (FLD) is one of said class specific method, in that it reforms the scatter in order to make it more viable for classification.

2.4.3 Local Binary Patterns Histograms (LBPH)

LBPH Parameters

The LBPH uses 4 parameters:

Radius: The radius is used to shape the circular local binary pattern and denotes the radius around the central pixel. It is usually set to one (1).

Neighbors: The number of sample points to shape the circular local binary pattern. Keep in mind: the more sample points you include, the greater the computational cost. It is usually set to Eight (8).

Grid X: The number of cells in the horizontal direction. More the cells, the finer the grid, the higher the dimensionality of the ensuing feature vector. It is usually set to 8.

Grid Y: The number of cells in the vertical direction. More the cells, the finer the grid,

the higher the dimensionality of the ensuing feature vector. It is usually set to 8.

Local Binary Pattern (LPB)

In 2002, Ojala et al. [7] extended their original Local Binary Pattern operator to a spherical neighbourhood of diverse radius size. It can also be used to implement the simple rotation-invariant descriptor. The LBP operator has been protracted to consider diverse neighbourhood sizes [8]. For example, the operator $LBP_{4,1}$ uses only 4 neighbours whereas $LBP_{16,2}$ uses 16 neighbours on a circle of radius 2. The operator $LBP_{P,R}$ refers to a neighbourhood size of P equally spaced pixels on a circle of radius R that shapes a circularly symmetric neighbour set.

A LBP is named uniform if in the binary pattern there are at most 2 bitwise transitions from zero to one, or contrariwise once the bit pattern is traversed circularly. [2] For instance, the patterns 00000000 (zero transitions), 01110000 (two transitions), and 11001111 (two transitions) square measure uniform whereas the patterns 11001001 (four transitions) and 01010010 (six transitions) are non-uniform. Figure 3.5.4 shows the example of the texture primitives.

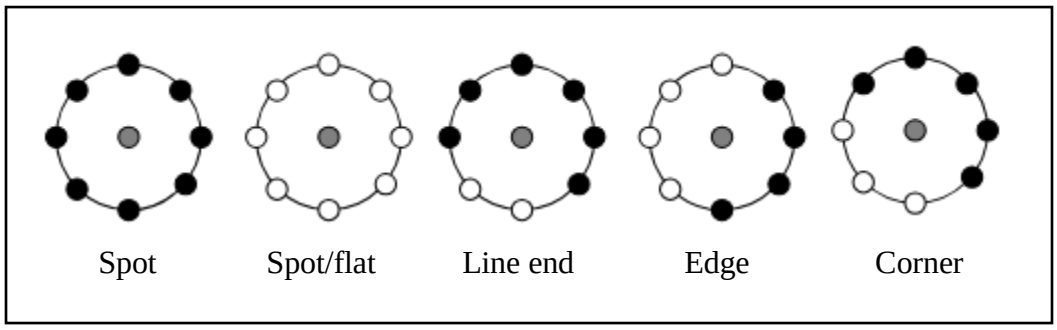


Figure 2.4.3: Above Example shows that Local Binary Pattern can detected Texture primitives (white circles signify 1 and black circle's 0) [9]

Cascade Classifiers

Cascade classifier is a sequence of weak classifiers for effective classification of image

areas. The weak classifier is intended to choice the single Local Binary Pattern histogram bin which best splits the positive and negative samples. Similar to [1], a weak classifier $h_j(x)$ contains feature (f_j) which relates to each Local Binary Pattern histogram bin, a threshold θ_j and a parity p_j representing the direction of the inequality sign [9]:

$$H_i(x) = 1 \text{ if } pjf_j(x) < p_j\ \theta_j$$
$$H_i(x) = 0 \text{ otherwise}$$

Found weak classifiers are used to compose a strong classifier. Figure 3.5 shows how the weak classifier works.

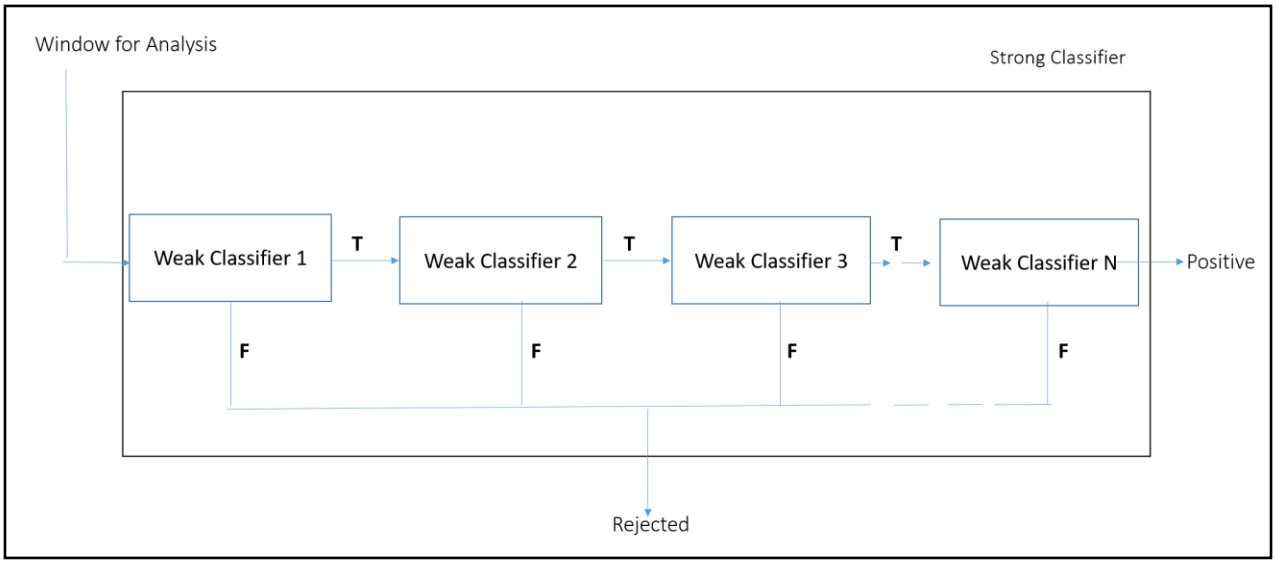


Figure 2.4.4: Cascade Classifier Concept [10]

Chapter 3

System Development

3.1 Mobile Operating System

Android

Android is a powerful open source **Operating System** based on Linux. It is used in cell phones such as tablet computers and smartphones. Android was created by the Open Handset Alliance, and afterward, drove by Google, and different organizations.

Why Android?

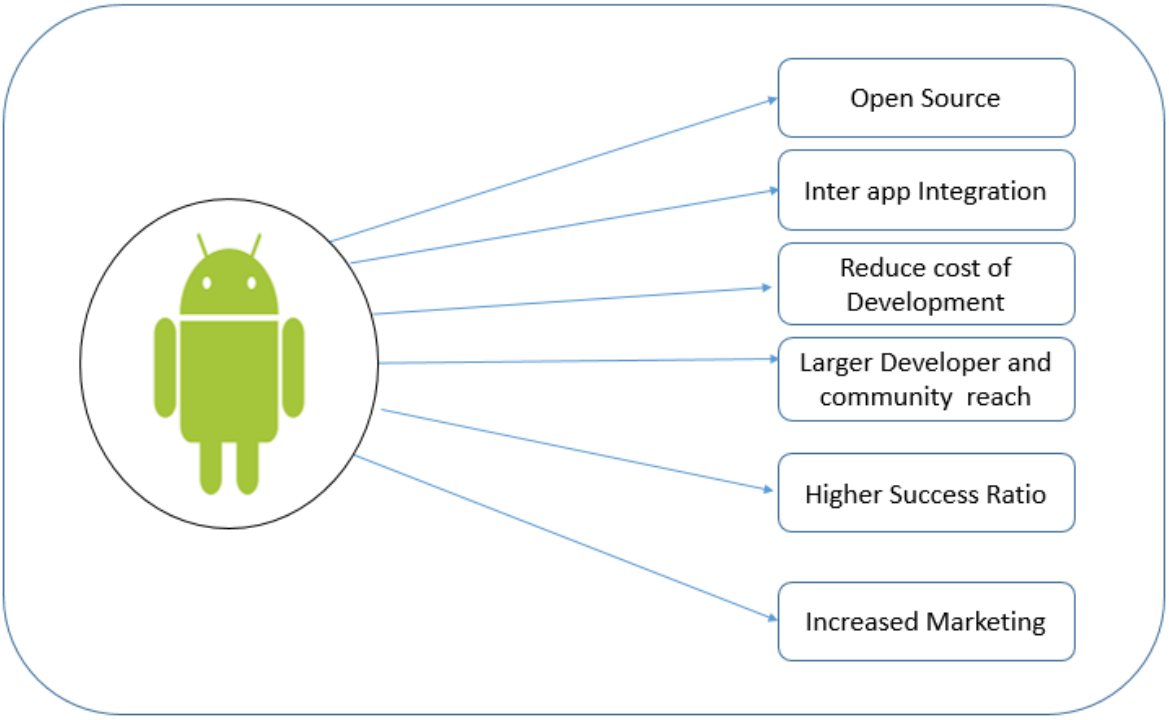


Figure 3.1: Advantages of using Android. [13]

3.2 Programming Environment

3.2.1 Java

Java is an exceptionally famous programming language created by Sun Microsystems (now owned by Oracle). C and C++ programming language are developed a long before java, Java consolidates a significant number of the intense features of those dominant languages whereas addressing some of their flaws

Some important features of Java are:

- Learning and understanding Java is easy.
- Java is designed to be platform independent and protected, utilizing virtual machines.
- Java is object-oriented.

Android depends extremely on Java basics. The Android SDK contains numerous standard Java libraries like graphics libraries, data structure libraries networking libraries, math libraries. Android studio also includes special Android libraries that helps developer to create amazing Android applications.

3.2.2 XML Language

XML full form is Extensible Markup Language. XML is a markup language just like HTML which is used to define data. XML is understandable both by human and machine and it is scalable and easy to develop. XML tags are not predefined in XML, therefore we have to define our own Tags .In Android XML, is used for making our layouts because it is a lightweight language which does not make our layout heavy.

3.3.3 Android Studio

Android Studio offers the quickest tools for the construction of application on every type of Android gadget. Android Studio is the integrated development environment (IDE) for Google's Android operating system constructed on Jet Brains' IntelliJ IDEA software and deliberately design for Android development. Android studio is easily obtainable for download. You can download android studio on Linux, Mac OS, and Windows-based OS. It is a substitute for the Eclipse Android Development Tools (ADT) as the main IDE for local Android application development.

3.3.4 Open CV Library

Open CV (Open Source Computer Vision) is a free graphics library generally targeted the real-time computer vision. Open CV was developed by Intel to process images. Open CV was destined for image processing as it is pre-loaded with many functions and many algorithms. As Open CV is pre-loaded with these functions and algorithms it helps researchers to solve the vision-related problems. Open CV provides applications that will assist in the training of cascade classifier. We have use the open CV 2.4.10 for developing our application.

3.3.5 Open CV Android

Open CV begins helping android in Early 2010. Since at that point the OpenCV4 Android is being created to boost the advancement of Open CV for Android. "OpenCV4Android is the formal title of the Android port of the Open CV library" (Open CV Dev Group, 2013). Most Open CV usefulness is ported to Java API, which implies the API isn't, however, develop driving to some need of capacities that are accessible within the full adaptation. OpenCv4 Android is accessible as an SDK with a set of tests and Javadoc documentation for Open CV Java API. It too contains pre-built apk-files,

which you'll run on your gadget right away. Open CV offers a set of tests for Android engineers. These tests appear how Open CV can be utilized from Java and local level of Android.

3.3 Proposed Model

Face recognition model which propose for the detection and recognition of the student faces for marking their attendance. The main modules used are:

1) **Dataset Generation form training:**

   This is first stage in which face dataset of the user is created, in which 10-20 images of each user are taken and the features used are user ID and username.

2) **Face Detection**

   For the face detection LBP-based face detector is used [2].

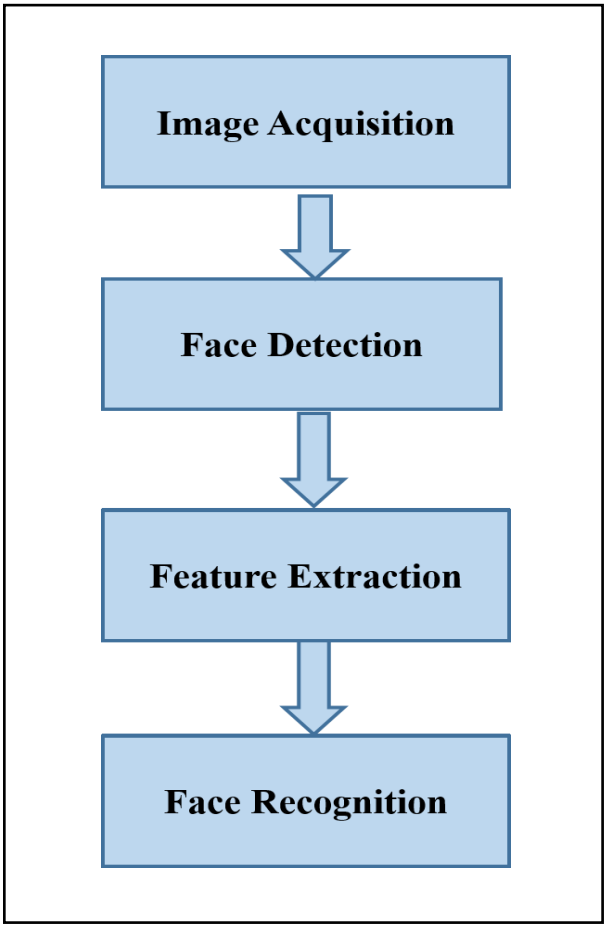
3) Pre-processing:

   Steps involve in pre-processing are:
   a) **Resizing**: Face is resized to a fixed pixel resolution after the face is detected
   b) **Cropping**: Background is removed from the image.
   c) **Grayscale Conversion:** The conversion of a colour image into a grayscale image

4) **Feature Extraction and Recognition**

   Histogram principle based algorithm is used for feature extraction and recognition.[2][3] The simple LBPH algorithm is selected for correct real-time processing of data as it is computational complexity is less and is more effective

compared to the other face recognition algorithms.



3.4 Image Acquisition

Face recognition technology can be obtain from almost any camera or video system that creates the image of ample quality and determination. We are using our mobile phones (Motorola G4 plus & Samsung Galaxy J7) powered by android for the image acquisition

3.5 Face Detection

Face detection has been enhanced as far as speed with the application of Haar-features with the involvement of the Viola-Jones [1] object detection method. One of the key confinements in early boosting-based methodologies is the strength to brightening and incomplete impediment of the face. To adapt to these restrictions, we propose to utilize

Local Binary Pattern (LBP) features to detect the face on our android application.

The simple plan of the LBP-based face detection is comparable to the Haar-based one, however, it uses histograms of picture for intensity comparisons, like boundaries, angles, and plane regions [4]. The Haar-based classifier was terribly slow to observe the face within the frame. First the face image is distributed into smaller regions from that the Local Binary Pattern (LBP) [6, 7] area unit taken out and concatenated into one feature histogram expeditiously expressing the face in the image. The textures in the facial areas are domestically converted by the LBP patterns whereas the full form of the face is recuperated by the development of the face feature histogram.

Steps for face Detection:

Step 1: Load the LBP cascade

Step 2: Instantiate the Open CV Cascade Classifier

Step 3: For each video frame received, call the cascade classifier

Explanation of Code:

1: It loads the native Open CV library to use Java API. An instance of Cascade Classifier is created, transient it the forename of the file from where the classifier is loaded.

2: Detect Multi-Scale technique is employed on the classifier fleeting it the given picture and MatOfRect object.

3: MatOfRect is accountable to do face detections after processing.

4: The method is rehashed for doing all the face location and marks the picture with rectangles and at the end, the picture is saved as an output.png record

The D**etect Multi-Scale** function spots objects of diverse sizes in the input image. The detected objects are given back as a list of rectangles. The parameters are:

- **Image:** Matrix of the sort CV_8U holding a picture where objects are discovered.
- **Objects**: Vector of rectangles where every rectangle comprises of the detected object.
- **Scale Factor**: This Parameter insists to which extent the image size is reduced at each image scale.
- **Min Neighbours**: This parameter insists on what percentage neighbours every candidate rectangle ought to be needed to keep it.
- **Flags**: The Constraint with the identical sense for an old cascade as in the sense CV Haar Detect Objects.
- **Min Size**: Minimum probable object size and objects lesser than that are ignored.
- **Max Size**: Maximum probable object size and objects greater than that are ignored

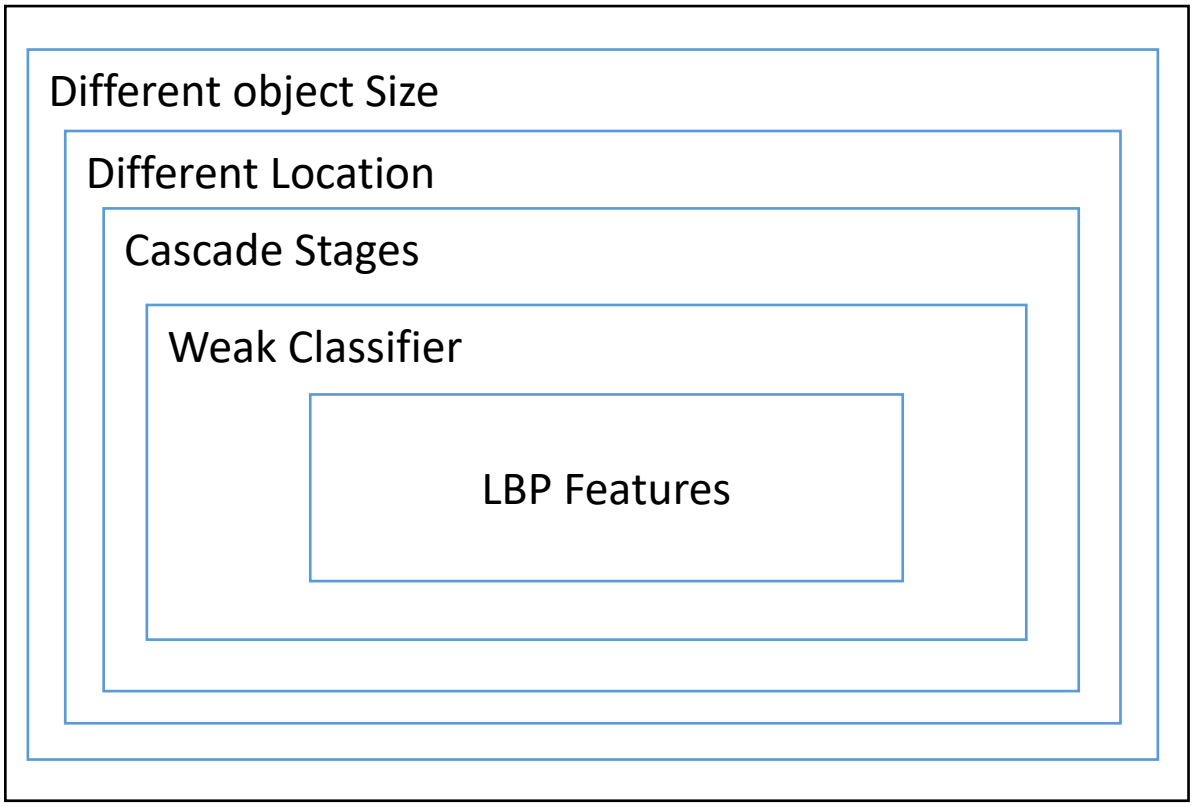


Figure 3.5.1: Working of face detector. [9]

3.5.1 LBP Detection Algorithm

The algorithm used could be a variant of the cascade rule introduced by Viola and Jones [1] and uses LBP options rather than Haar-like options so as to own quicker process and boosted classifiers. The LBP rule slides its process window over the item image for evaluating the serial stages of the cascade rule by marking their options. Every feature is represented by three × three neighbours' rectangular areas.

The value of every feature is calculated by examination the central space with the neighbour space around it (8 neighbours). The result's within the sort of associate degree 8-bit binary price referred to as LBP. Variety of options represent the stage of cascade rule. Every feature has positive and negative weights related to it. For the case wherever the feature is inconsistence with the item to be detected, the positive weight is extra to the total. For the case wherever the feature is inconsistent with the item, the negative value is added to the sum. The sum is then matched to the threshold of the stage.

If the sum is less than threshold, the stage fails and therefore the cascade shut down early, and, thus, the process window moves to the successive window. If the total is higher than the threshold, successive stage of the cascade is tried. In general, if no stage rejects a candidate window, it is assumed that the item has been detected.

In order to avoid the redundancy of computing the integral of rectangles, the integral pictures are calculated to speed up the calculation of the feature Figure 3.5.2 shows the Face Detection of a Student in a classroom when we took the photo of the students. A green rectangular frame appear in the face of the students.

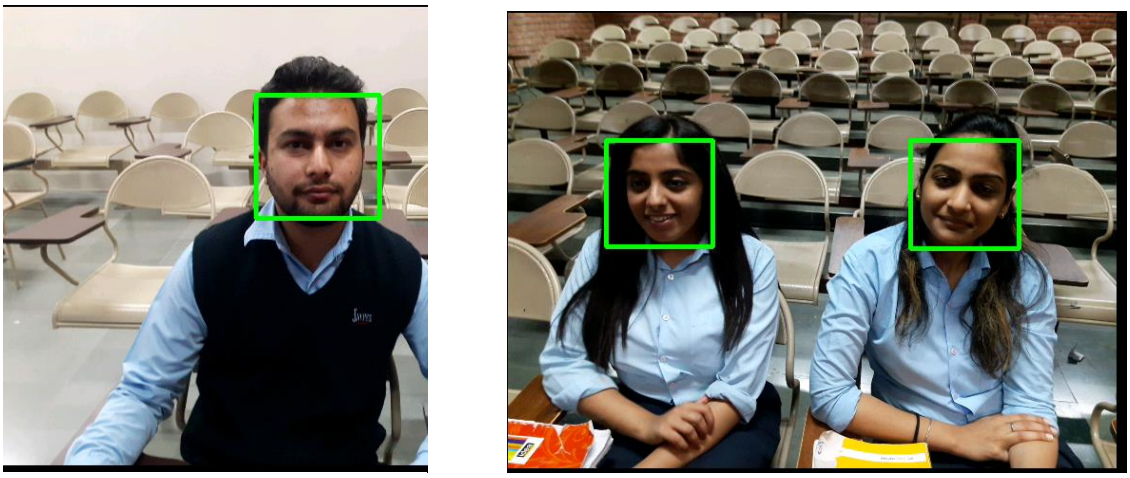


Figure 3.5.2: Green rectangular frame on the detected faces.

3.6 Face Pre-Processing

Face pre-processing is used after detecting a face in the frame, as face is detected in figure3.5.2 now the face which is inside the green rectangle is processed. Face recognition is prone to changes in lighting situations, face alignment, and face appearance, so it is supreme to reduce these changes as much as possible. There are several methods to eradicate those issues. (Baggio, et al., 2012)[4]:

Geometrical cropping: These functions modify the geometry of a picture by resizing or cropping the image. Figure 3.6.1 shows the cropped image which is detected in figure 3.5.2.

Figure 3.6.1: Cropped image of face detected.

Grayscale: An image may be delineated in grayscale. Every pixel of an image in a very grayscale is delineated by variety from zero to 255 that corresponds to an intensity

of grey of the pixel. This range is holding on in one computer memory unit. Zero represents the colour black and 255 is white. As it is going to be shown, grayscale is usually employed in computer vision and makes treatment of pictures easier once color isn't essential. Figure 3.6.2 shows the gray scale of the figure 3.6.1.

Figure 3.6.2: Gray scaled image of cropped image

3.7 Face Recognition using Local Binary Pattern Histogram

The face recognition systems operate basically in two modes:

Verification or authentication of a facial image: Compares the given facial picture with the facial picture identified with the client which is requiring the confirmation. It is a 1x1 comparison.

Identification or facial recognition: Compares the given facial picture with all facial picture from a dataset with the objective to discover the client that matches that face. It is a 1xN comparison.

3.7.1 Working of Local Binary Pattern Histogram:

LBP basically recapitulate the local structure in an image by matching each pixel with its neighbourhood. Take a pixel as center and threshold its neighbours against. In Feature-based Approach, local features on the face such as eyes and nose are detected and based upon which recognition is performed. The main idea is to divide the LBP image into local regions and extract a histogram from each. These histograms are called Local Binary

Patterns Histograms [11].

1. Present new image to the recognizer.
2. The recognizer creates a histogram for that new image.
3. The new histogram is compared with the histogram it already has.
4. Finally, it detects the best match and returns the student name associated with that best match. [12]

3.7.2 Employing the LBP operation:

The initial calculative step of the LBPH is to generate the intermediary image that defines the original image in an enhanced way, by underlining the facial features.

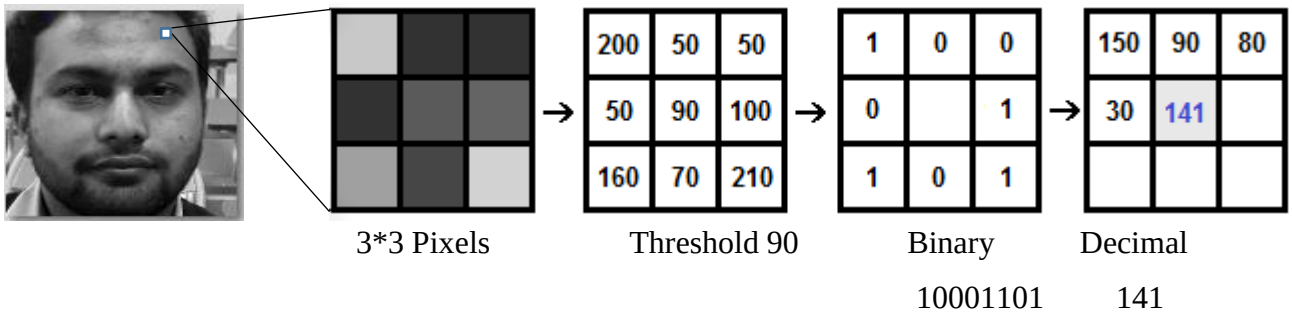


Figure 3.7.2: Operation of local binary pattern histogram.

Based on the above figure 3.7.2, let's break it into several steps so that we can fathom it easily:

1) Assume that we have a facial image in grayscale.
2) We can get the part of this facial picture as a window of 3x3 pixels.
3) It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
4) Then, we need to take the central value of the matrix to be used as the threshold.
5) This value will be used to define the new values from the 8 neighbors.

6) For each neighbor of the central value (threshold), we set a new binary value. We set 1 for values equal or higher than the threshold and 0 for values lower than the threshold.

7) Now, the matrix will contain only binary values (ignoring the central value). We need to concatenate each binary value from each position from the matrix line by line into a new binary value (e.g. 10001101).

8) Then, we convert this binary value to a decimal value and then set it to the central value of the matrix, which is actually a pixel from the original picture.

9) At the end of LBP procedure, we have a new image which represents the better characteristics of the original image.

3.7.3 Extracting the Histograms:

Now, using the facial picture generated in the last step, we can use the Grid X and Grid Y parameters to divide the image into multiple grids.

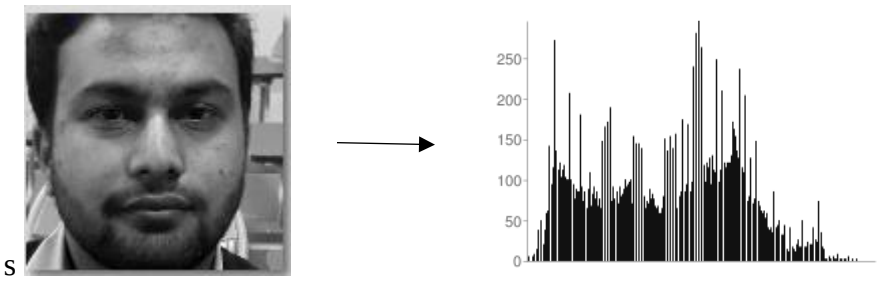


Figure 3.7.3: Histogram of the gray scaled image.

We can extract the histogram of each region of the figure: 3.7.1 as follows:

1) As we have a facial image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.

2) Then, we need to focus each histogram so that we can create a new histogram. Supposing we've 8x8 grids, we will have 8x8x256=16.384 positions in the final histogram. The final histogram represents the better characteristics of the initial image.

3.7.4 Performing the face recognition:

In this step, the algorithm is already trained. Each histogram created is used to represent each image from the training dataset. So, given an input picture, we perform the steps again for this new image and creates a histogram of this image.

1) So to find the image that matches the given image. We just have to compare two histograms and return the face with the closest histogram.

2) We can use several methods to compare the two histograms (calculate the distance between two histograms), for example: Euclidean distance, chi-square, absolute value, etc.

3) So the algorithm output is the ID from the image with the closest histogram. The algorithm should also return the calculated distance, which can be used as a 'confidence' measurement, Lower confidences are superior because it means the distance between the two histograms is closer.

4) We can then use a threshold and the 'confidence' to automatically estimate if the algorithm has correctly recognized the image. We can assume that the algorithm has successfully recognized the face if the confidence is lower than the threshold defined.

3.8 System Design for E- Notifier:

The system design can be remarkably expounded from the below Use Case diagrams:

To show the user's involvement with the system the best way is to represent it in a use case diagram

In this application there are two forms of users: User and admin. Figure 3.8.1and Figure 3.8.2 depicts their use case diagram

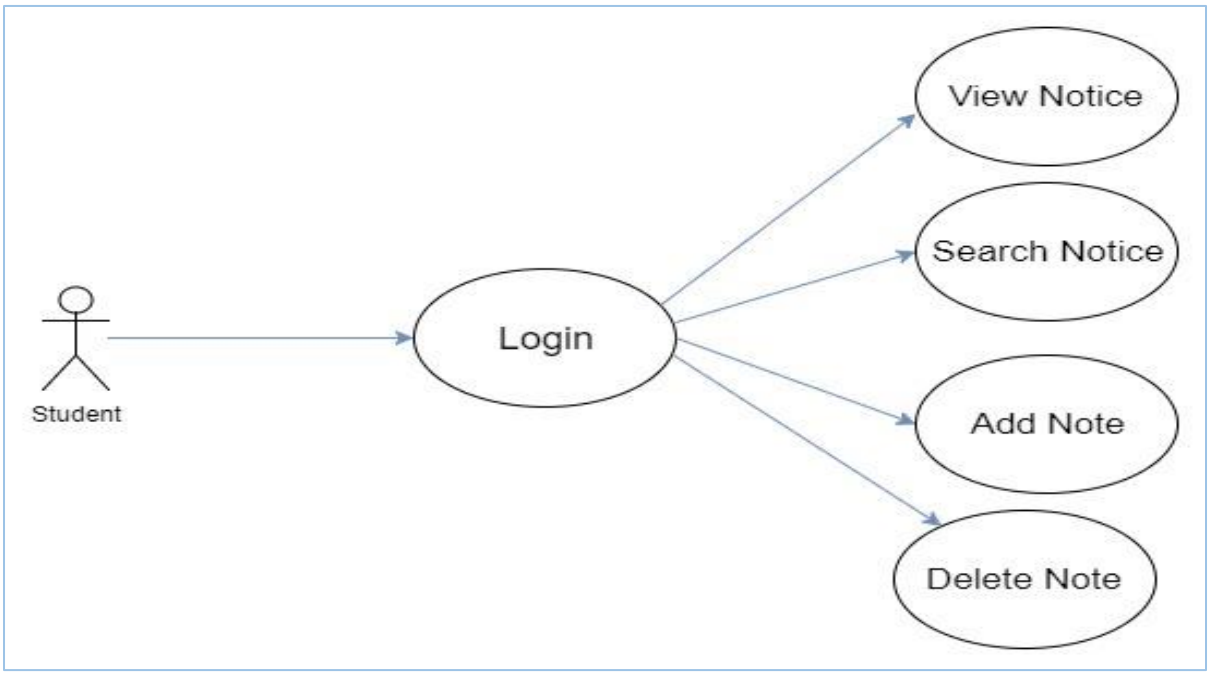


Figure3.8.1 Use Case Diagram of the User.

The Figure 3.8.1 shows what student can do with this application. After login in the application user can outlook the notices, and note and delete note and can search for specific notices.

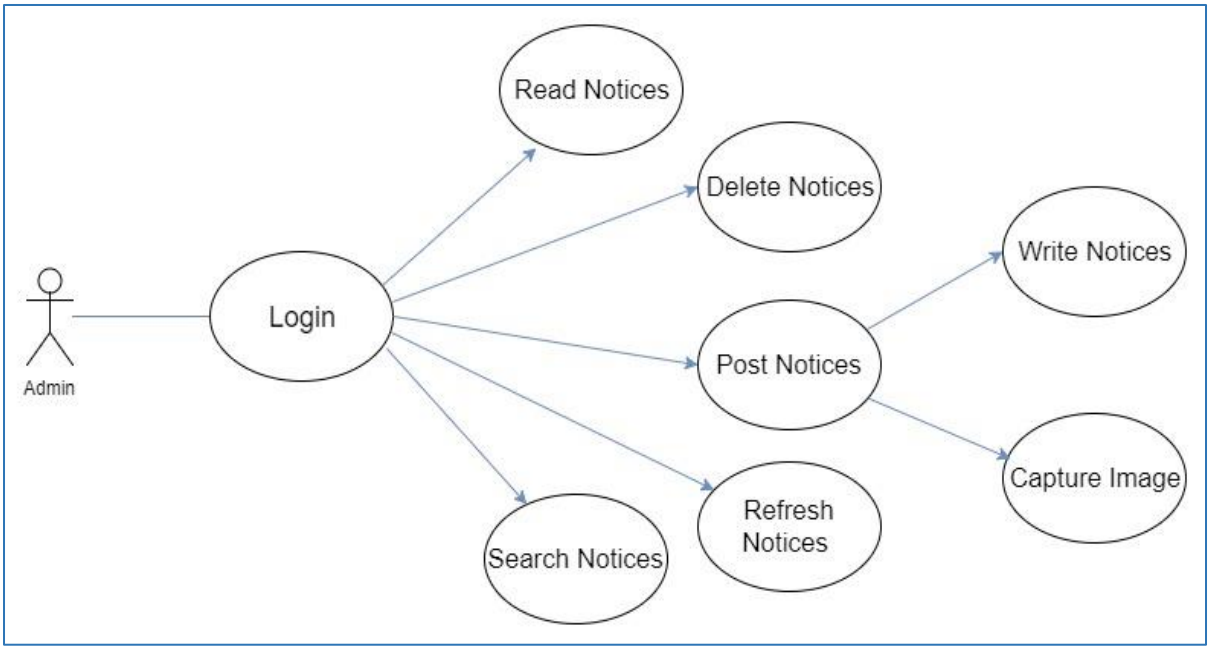


Figure 3.8.2 Use Case Diagram of the Admin.

Figure 3.8.2 depicts the privileges of admin. After Login in the application admin can post the notice in addition to viewing it. Admin can also post images as the notices. Images are chosen from the gallery. Admin can also send the pdf attachments.

3.9 User Interface Design

Design of application with which the user interacts is called the user interface design. User Interface Design should be design in such a way that it becomes easy and simple for the user to use. Besides that it should have simple feel and look.

Figure 3.7.2 the first page which is presented to the user. In order to access the notices user has to enter his username and password in the login page. First page contain three buttons, login, register and forget password. Figure 3.7.3 the registration page where the user get registered with the servers of this application.

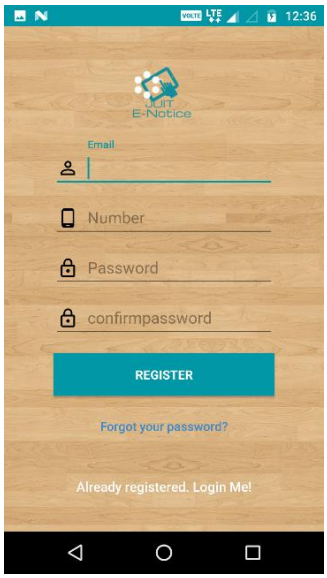


Figure 3.9.1: Registration Page of E-Notifier

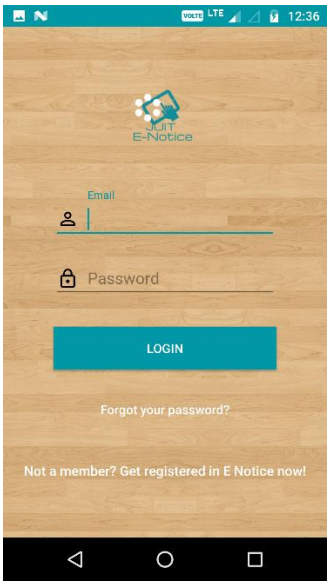


Figure 3.9.2: Login Page of E-Notifier.

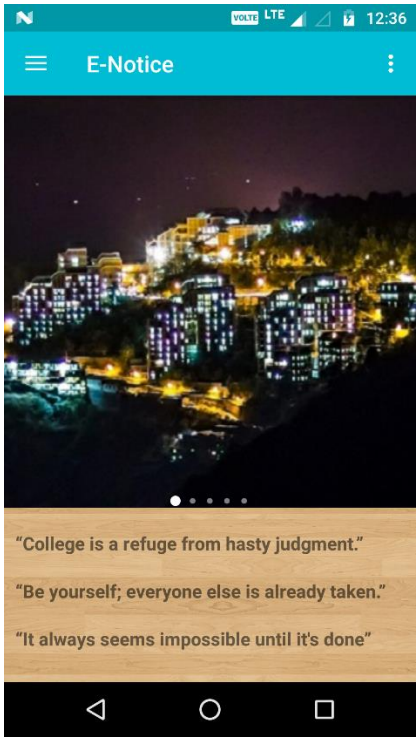


Figure 3.9.3: First Page of E-Notifier.

Chapter 4

Performance Analysis

4.1 Training Activity

Local Binary Pattern algorithm is used train the system after taking the image of the faces. XML file is created which contains the data about the face when you train the system. This process is known as the trained classifier. Detection of the face, training the system and the processing of the image are done in the training activity of the application.

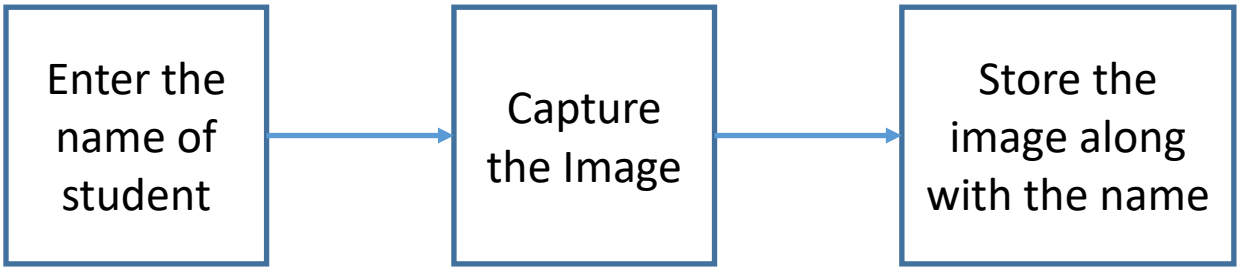


Figure 4.1.1: shows how to train the application.

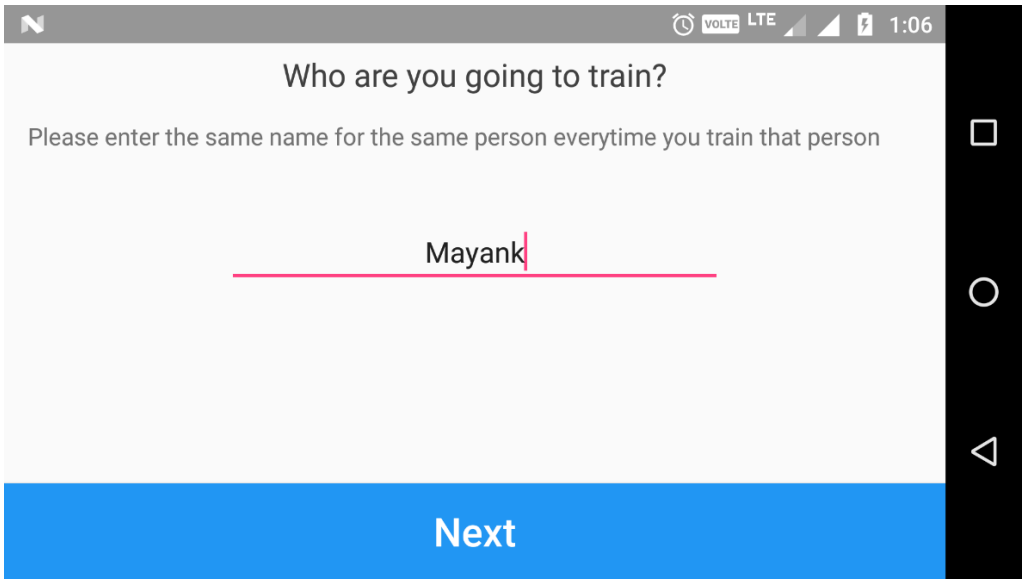
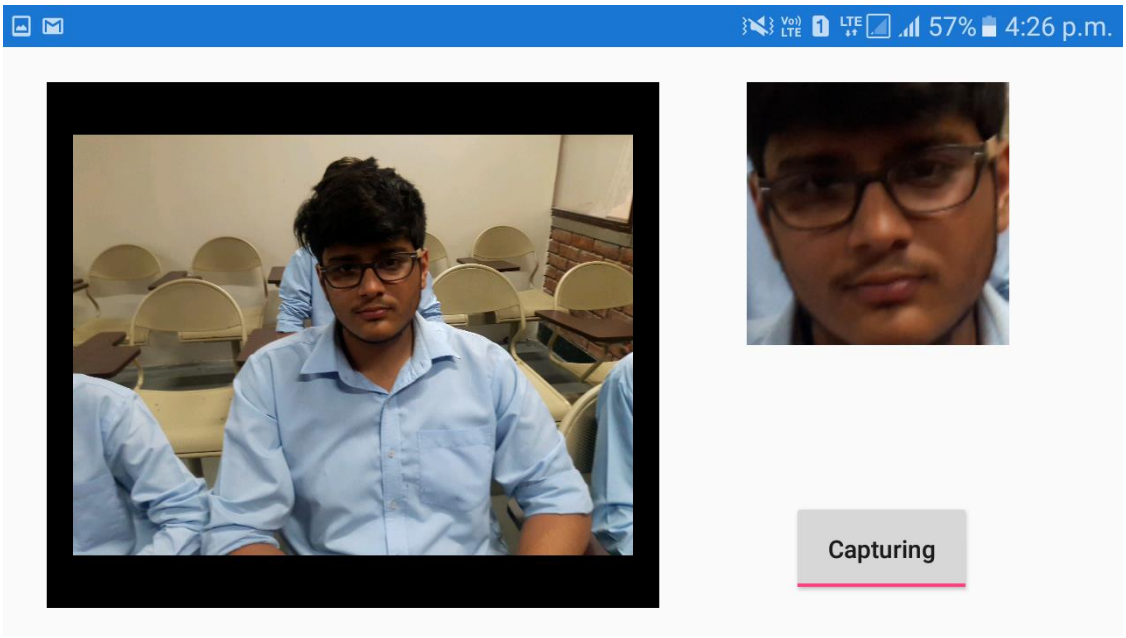


Figure 4.1.2: Entering the name of student.

Figure 4.1.3 Shows the process of capturing of the face after clicking on the capture button. Capturing the face which is inside green Rectangle.



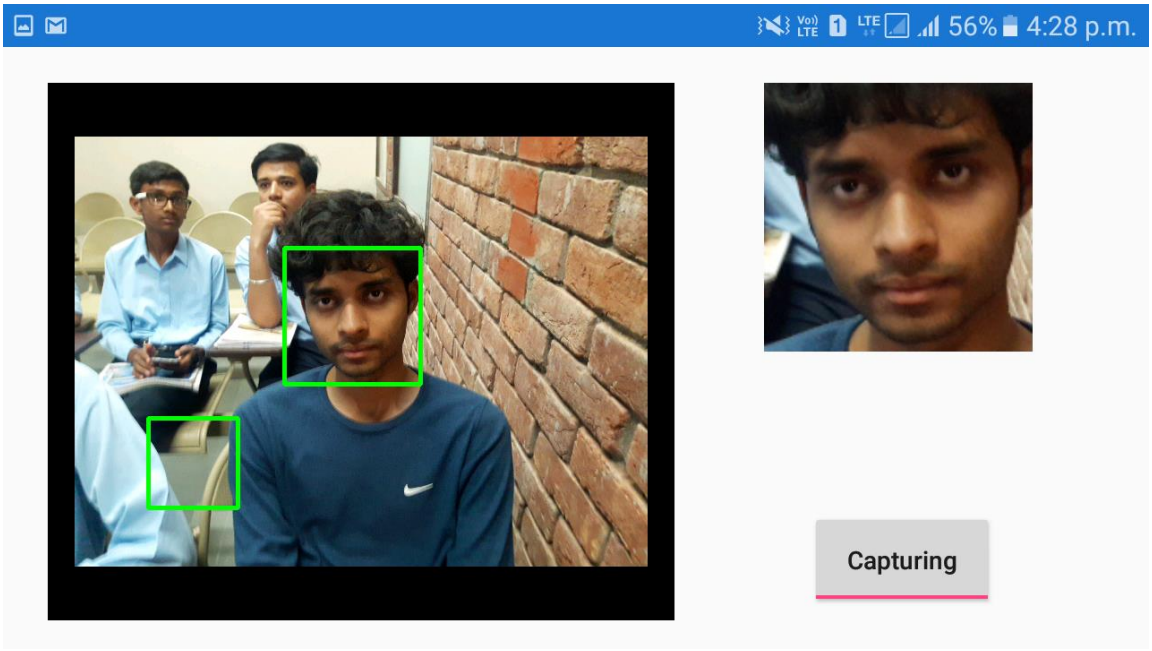


Figure 4.1.3: Capturing the face which is inside green rectangle.

Figure 4.1.4 Shows that the face is captured which is inside the green rectangle and the image is stored.

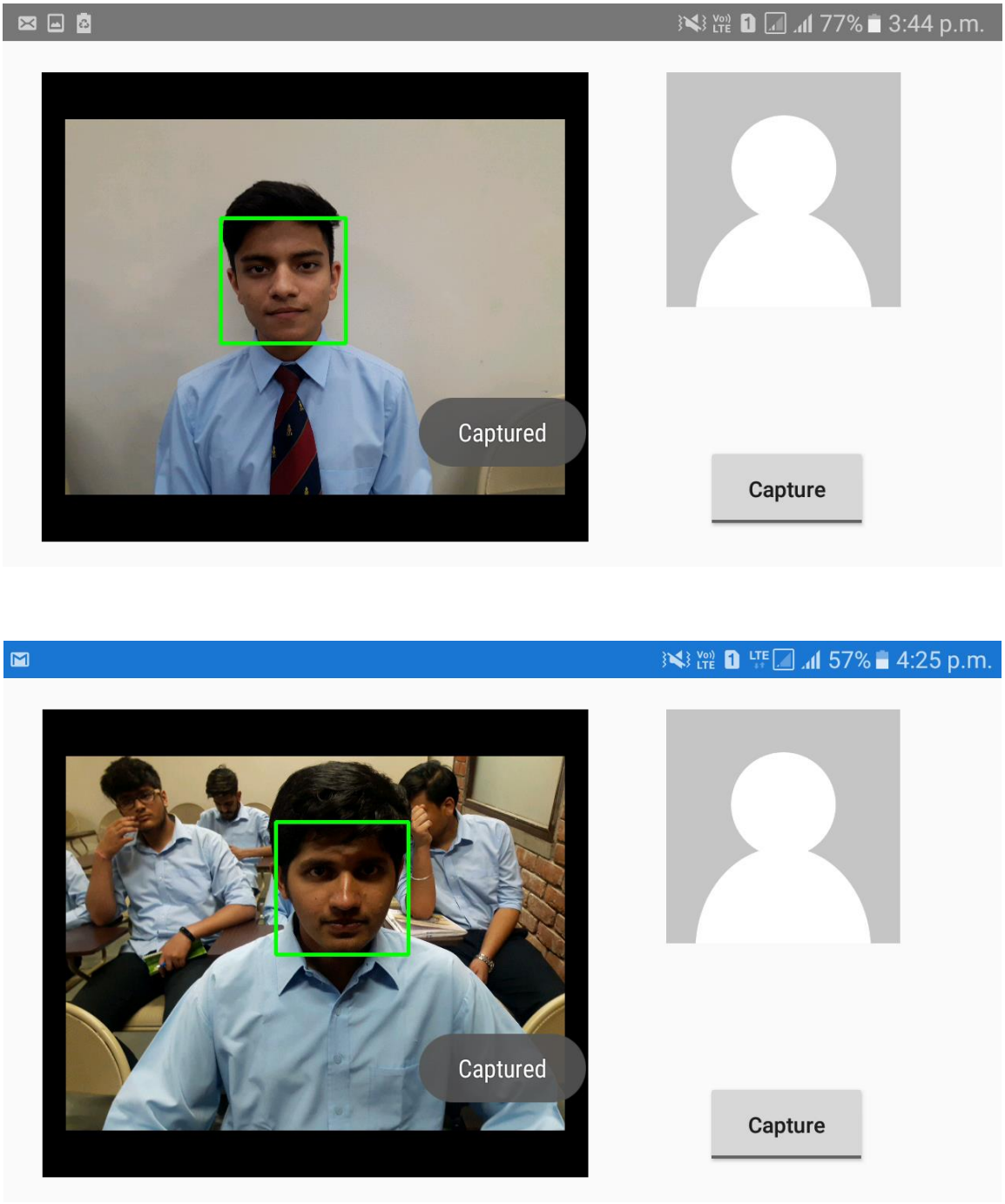


Figure 4.1.4: Image inside green rectangle is captured and stored.

Generation of Test Image’s for the training the system.

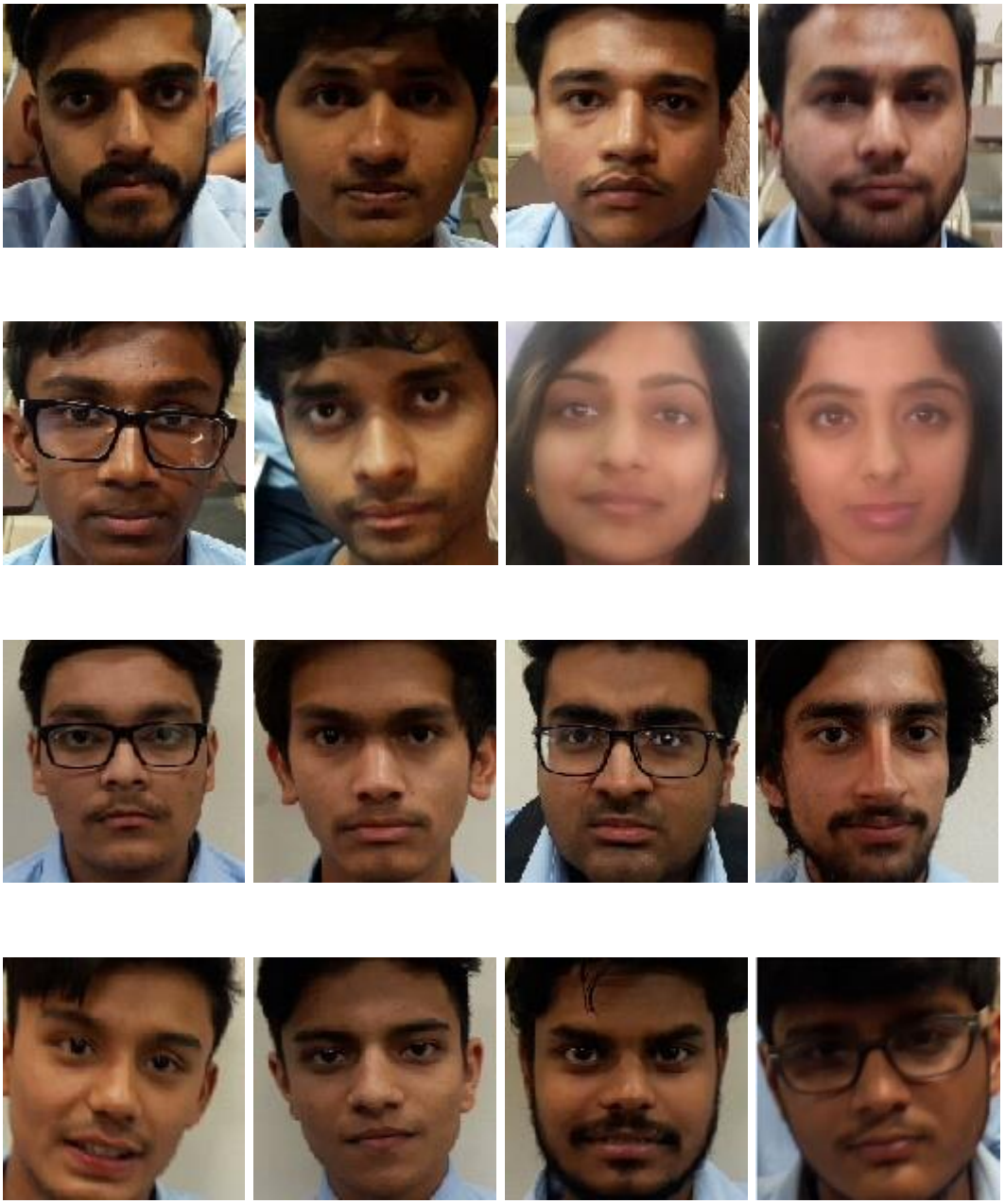


Figure 4.1.5: Training the System with the test image.

4.2 Recognizing Faces

Once the training of the system is done, the testing of the system can be done through the recognize activity. Whenever the face is detected it is firstly converted into a vector symbol from 2D to ID. The detected face is taken as an input for the Boolean function. Being a Boolean function it returns a true whenever a face gets a close match else false is returned. Further when a close match is found the identity of the person is extracted via the recognize function.

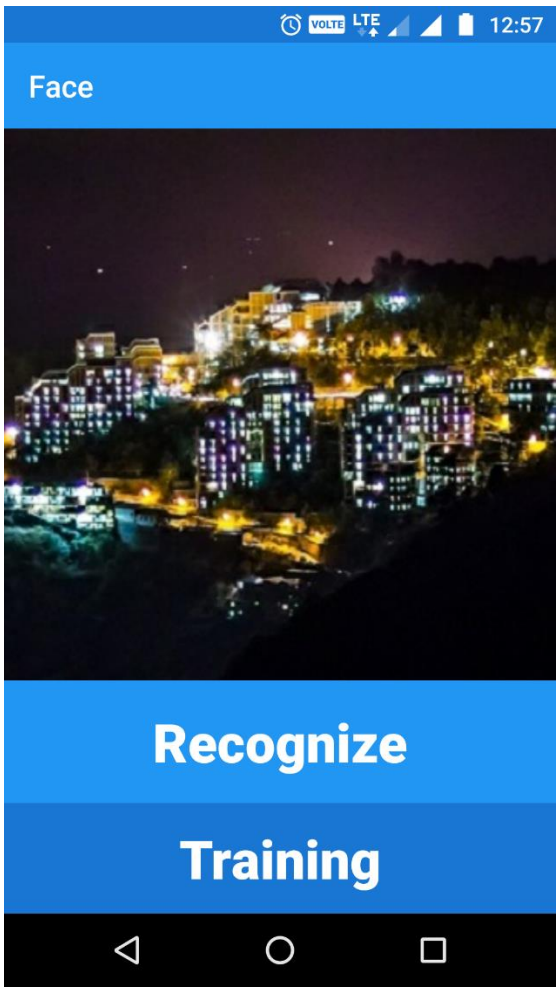


Figure 4.2.1: Landing Page of face recognition application

Figure 4.2.2 shows the face recognition of two students when they are looking in the camera and when they are not looking into the camera.

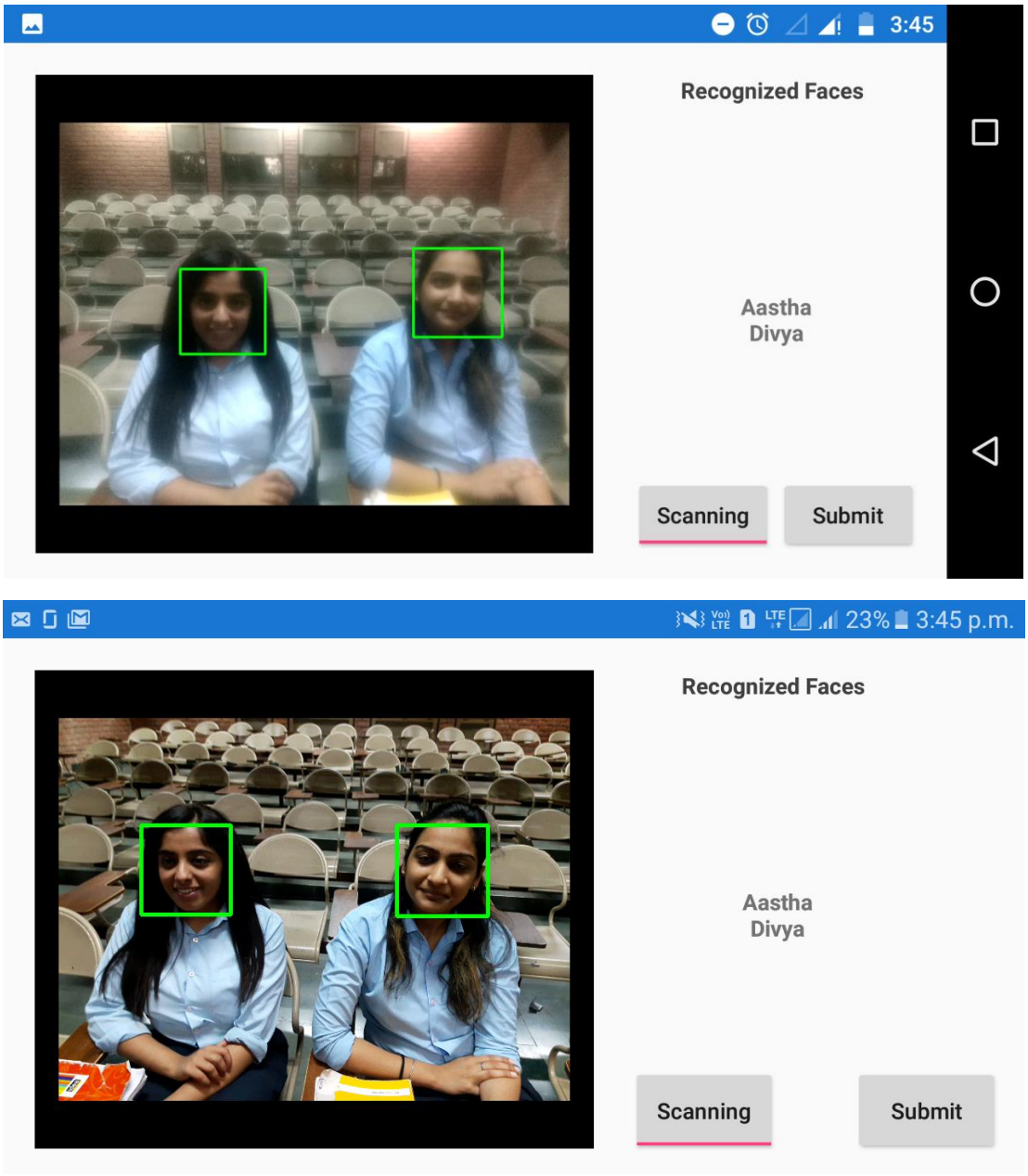


Figure 4.2.2: Face Recognition of two Students.

Successfully detected and recognize the faces of two students.

Figure 4.2.3 shows the face recognition of three students when all of them are in a single row (Front row) and when they are in different row (two at front and one at second row).

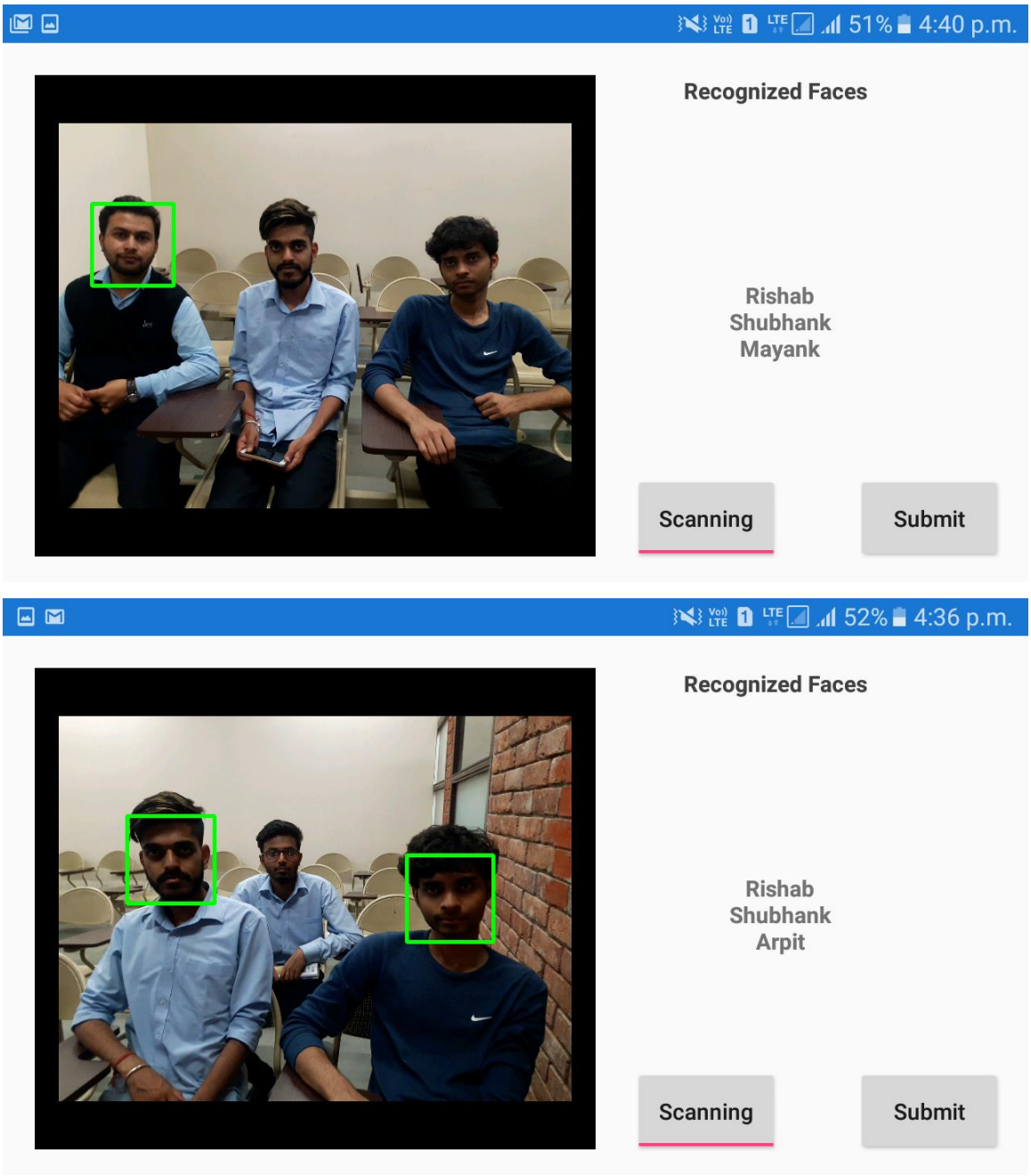


Figure 4.2.3: Face Recognition of Three Students.

Successfully detected and recognize the faces of three students.

Figure 4.2.4 shows the face recognition of four students when all of them are in a single row (Front row) and when they are in different row (two at front and two at second row) but under different lighting conditions.

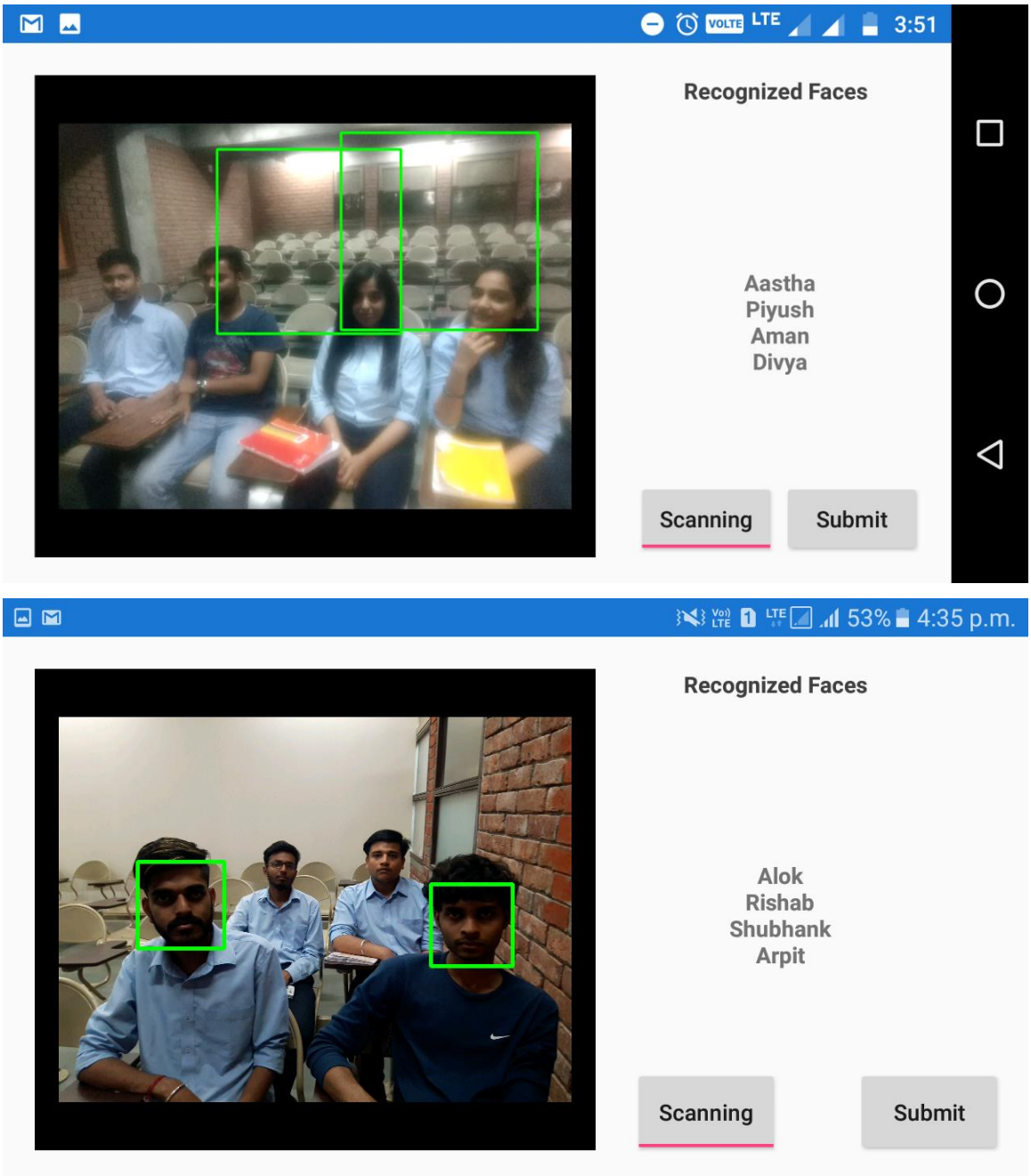


Figure 4.2.4: Face Recognition of four Students.

Successfully detected and recognize the faces of four students.

Figure 4.2.5 and Figure 4.2.6 shows the face recognition of five students and seven students when few seats are vacant (in front and second row) and when all seats are occupied vacant (in front and second row).

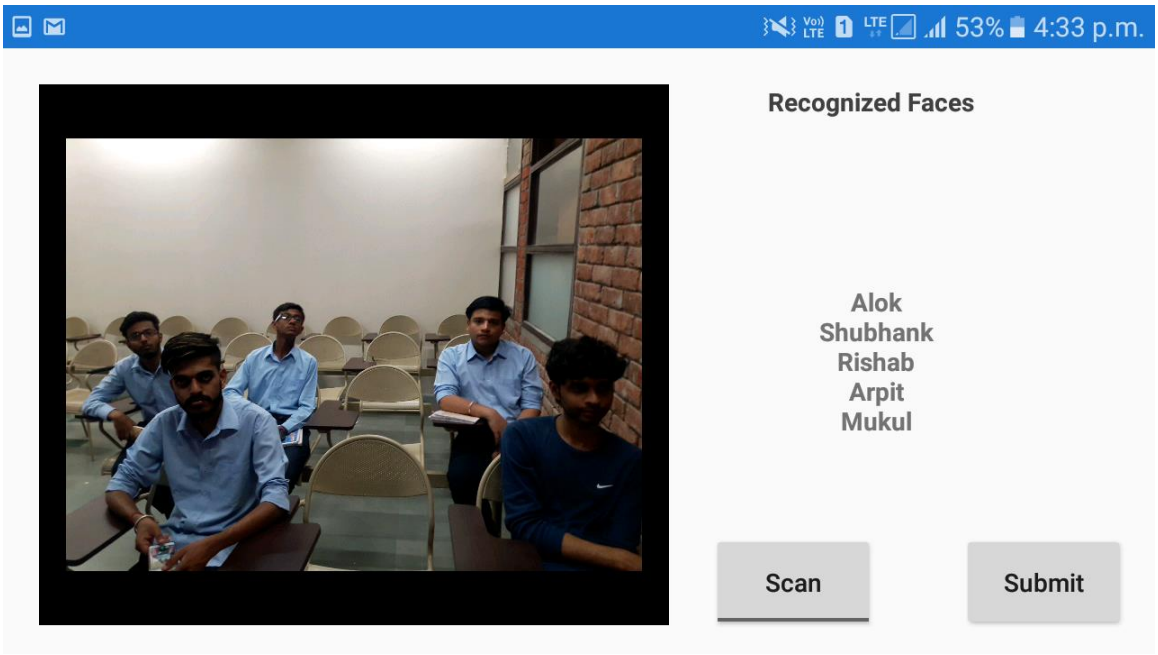


Figure 4.2.5: Face Recognition of five Students.

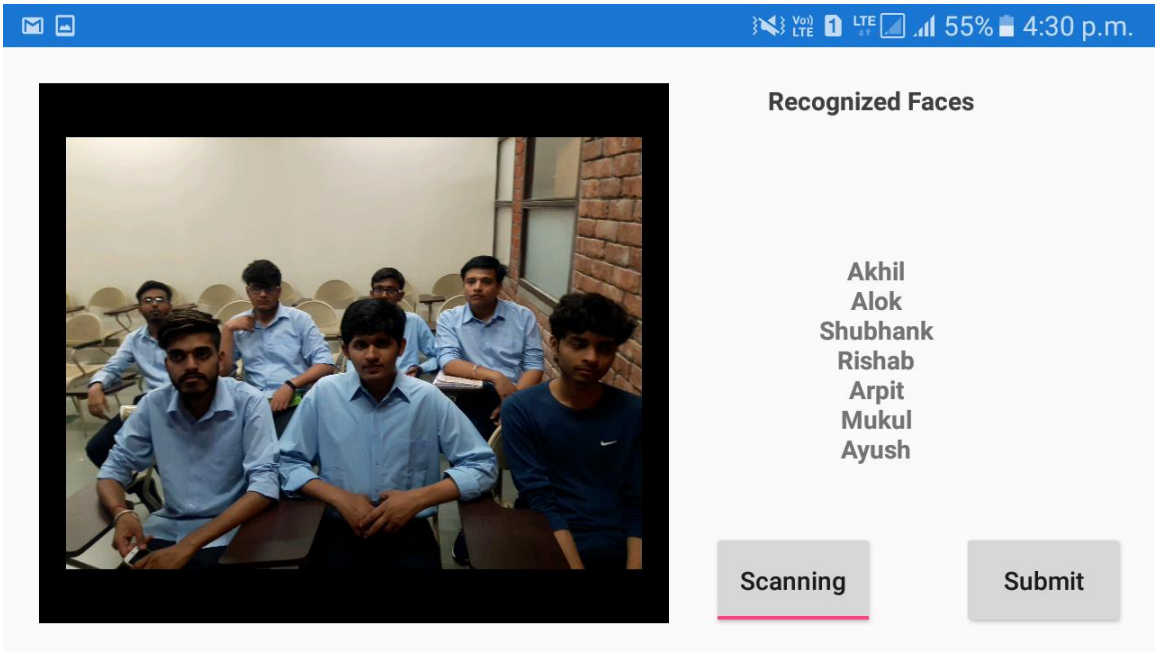


Figure 4.2.6: Face Recognition of Seven Students.

Successfully detected and recognize the faces of five and seven students.

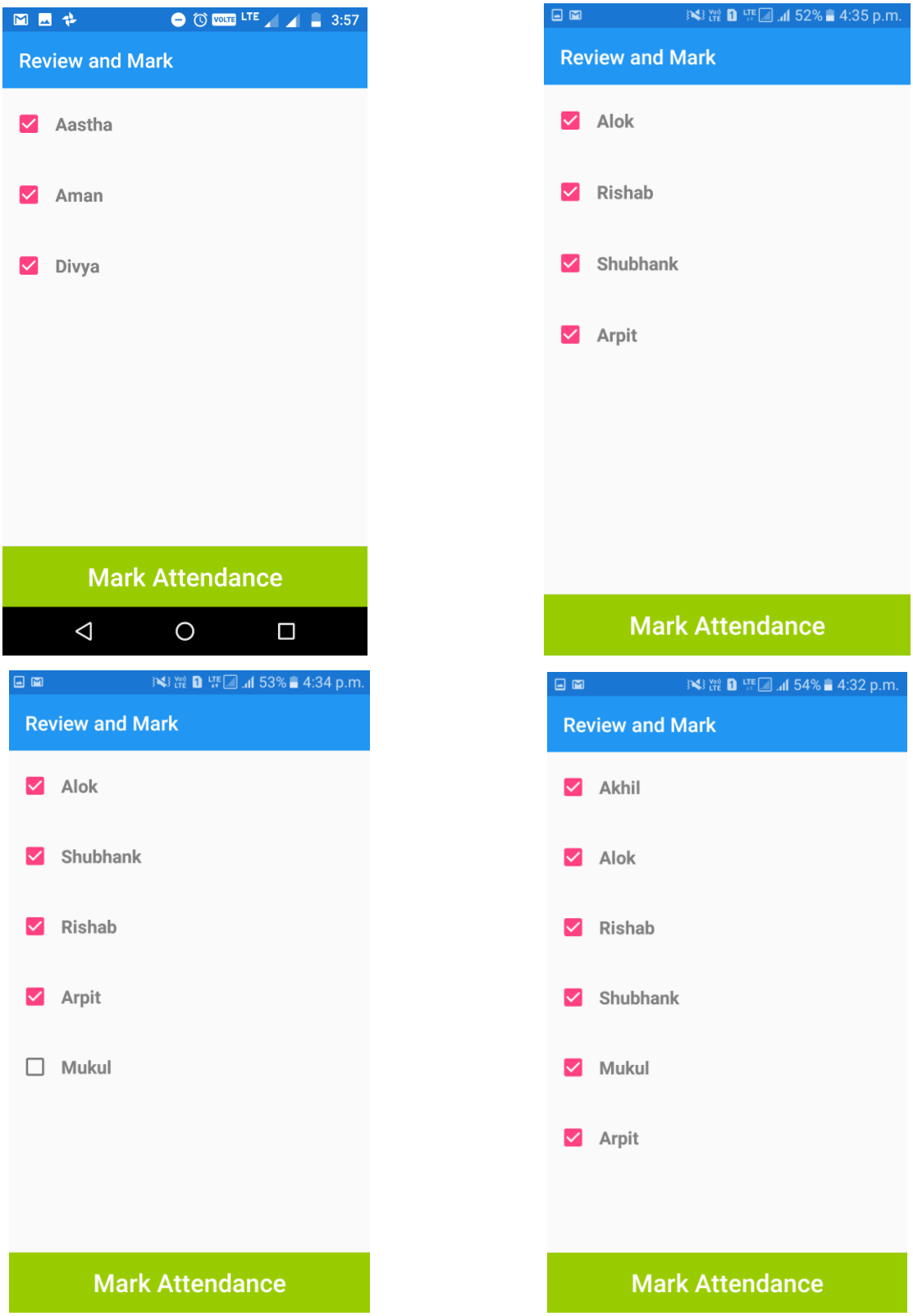


Figure 4.2.7: shows the Marking of attendance and reviewing it.

Table 4.2 shows the number of face detected and recognized ant their detection and recognition rate.

No. of students	No of face detected	Detection Rate	No. of face Recognize	Recognition rate
1	1	100%	1	100%
2	2	100%	2	100%
3	3	100%	3	100%
4	4	100%	4	100%
5	5	100%	5	100%
6	6	100%	6	100%
7	7	100%	7	100%

Table 4.2 No of face detected and recognized.

4.3 Drawbacks:

Figure 4.3.1 shows how application is unable to detect and recognise if the student face is not in the frame, as he face is facing downward hence unable to mak his attendance

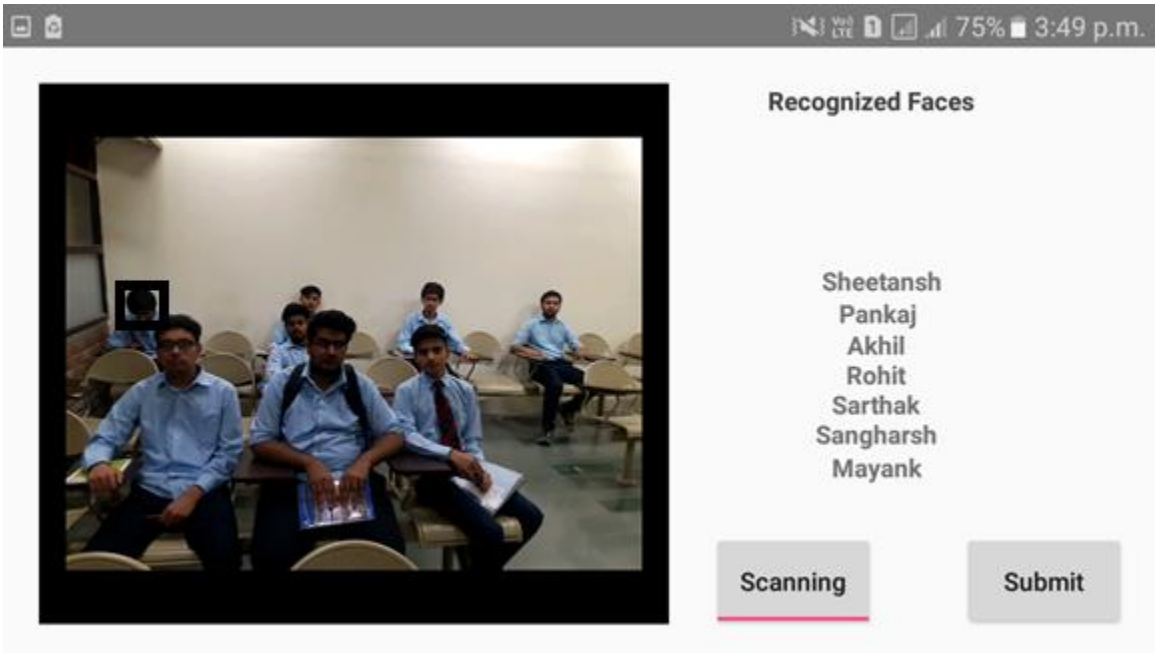


Figure 4.3.1: Unable to detect and recognize

Figure 4.3.2 shows how application is unable to detect and recognise if the student face is not completely in the frame,hence unable to mak his attendance

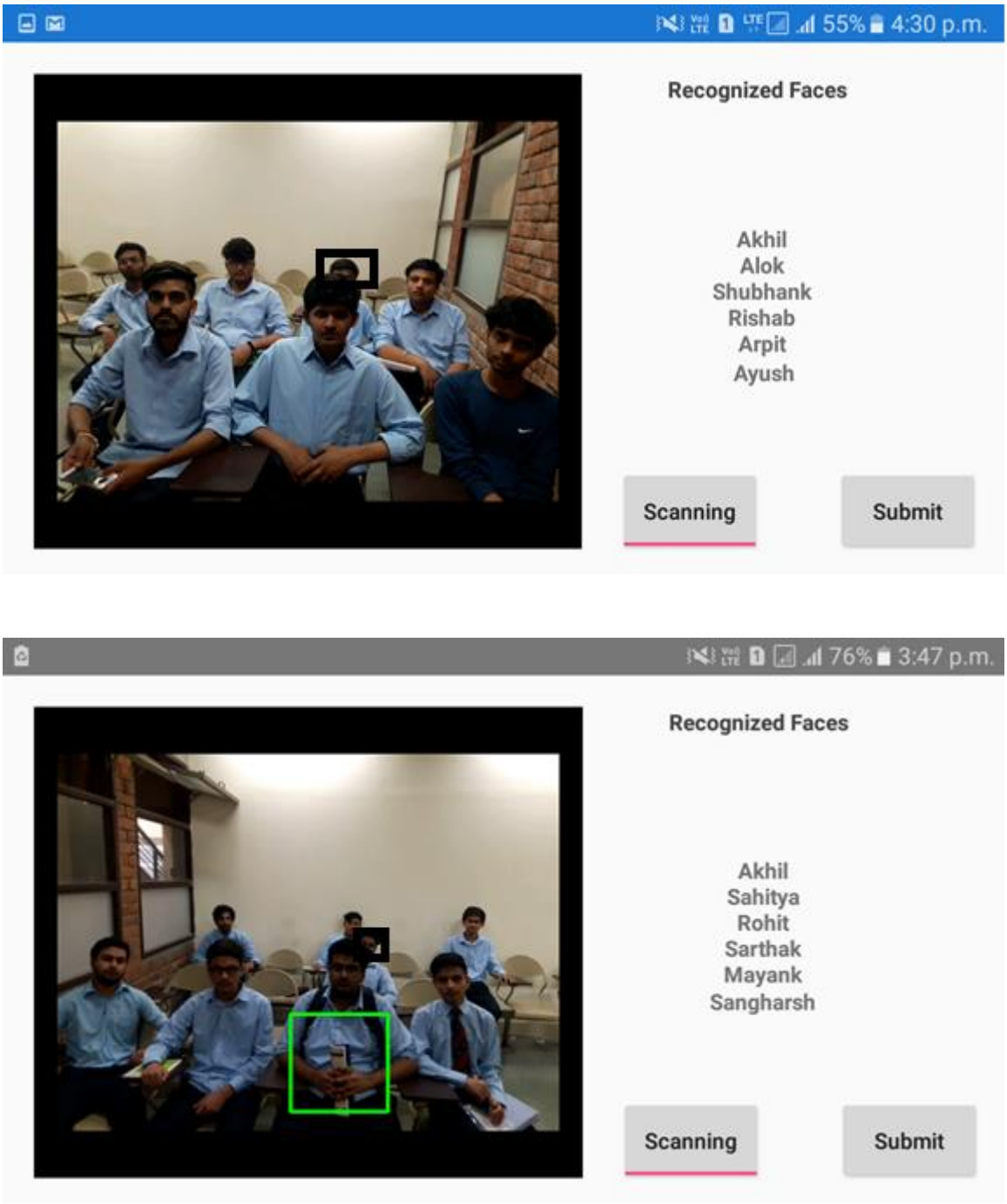


Figure 4.3.2: Unable to detect and recognize

Figure 4.3.3 shows how application is unable to detect and recognise if the student face is completely in the frame but is far fom the camera limits,hence unable to mak his attendance

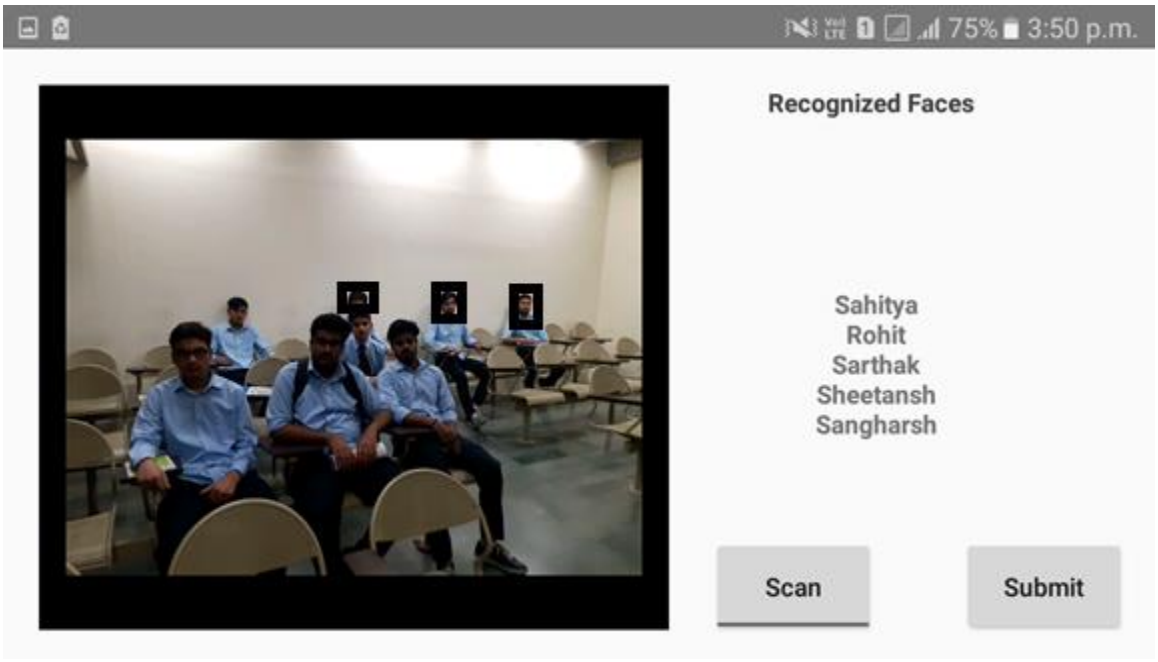


Figure 4.3.3: Unable to detect and recognize

Chapter 5

Conclusion

5.1 Conclusions

In this project, we studied various algorithms for the implemention of face recognition system in mobile phones. Eigenfaces machine-learning algorithm was the engine of training the system after applying some filters on the image. Furthermore, the Eigenfaces algorithm allows the application to recognize the face realtime. Eigenfaces was not very sensitive to a change in the number of subjects during the first phase, however, an increase in size of the training set helped the algorithm to correct its wrong prediction. An increase in the data set did not help to recognize more subjects, but it turned correct predictions into wrong ones. Eigenface was not accurate in the second phase.

Fisherfaces had better results in the first phase with a larger amount of data. However, its behavior was different in the second phase. Sometimes with 20 pictures, the results were better, but with 40 pictures the results were the same or worse. This algorithm was most of the time the worst in the two phases. It also had a very low accuracy in the second phase.

A face is detected using the Local binary pattern cascade classifier. After testing the Haar-like cascade classifier, the speed of the detection was very low as compared to LBP that always had at most 96 % in the first phase and is comparatively better algorithm. In the second phase, this algorithm had a consequent drop in its accuracy compared to the first phase. An increase in the number of subjects dramatically changed its prediction. In each phase, an increase in the training data had a positive effect or no effect.

We have identified various issues in our face recognition systems:-

- Illumination Problem: - The illumination problem is that where the same face appears different due to a change in lighting. The changes induced by illuminatio are often larger than the differences between individuals, causing systems based on comparing images to misclassify input images.

- The Pose Problem in Face Recognition: - It is not surprising that the performance of face recognition systems drops significantly when large pose variations are present in the input images. When illumination variation is also present, the task of face recognition becomes even more difficult. Here we focus on the out-of-plane rotation problem, since in-plane rotation is a pure 2D problem and can be solved much more easily.

- Distance Problem: - Our face recognition system fails if the distance between the camera and the face exceeds more than 3 meters.

- Camera Problem: - Resolution of a camera plays a very important role in recognizing the image. Better the camera, more accurately it will recognize the face. For example, we have tested our face recognition system with two phones and we have found that the phone with high resolution camera does the better than the other.

- Number of people: - Our face recognition system is limited to only 7-8 people at a time. If this number exceeds, our system fails to recognize.

5.2 Future Scope

As part of the future work, we would like to develop an application that would allow the user to add or delete face classes in the training set. This would give users the freedom to define their own user groups rather than a pre-defined set on the server. We would also like to explore better algorithms for face detection and face recognition to increase the number of student to be detected and recognize.

Bibliography

[1] Viola, Paul, and Michael J. Jones. "Robust real-time face detection." International journal of computer vision 57.2 (2004): 137-154.

[2] Ahonen, T., Hadid, A. and Pietikainen, M., 2006. Face description with local binary patterns: Application to face recognition. IEEE transactions on pattern analysis and machine intelligence, 28(12), pp.2037-2041.

[3] Alrashed, H.H., 2016. Detecting live person for the face recognition problem: submitted in partial fulfilment of the requirements for the degree of Master of Information Sciences, Massey University (Doctoral dissertation, Massey University).

[4] Baggio, D. L., Emami, S., Escriva, D. M., Ievgen, K., Mahmood, N.,Saragih, J., et al. (2012). Mastering openCV with practical computer vision projects: Step-by-step tutorials to solve common real-world computer vision problems for desktop or mobile, from augmented reality and number plate recognition to face recognition and 3D

[5] https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework

[6] T. Ojala, M. Pietika¨inen, and D. Harwood, "A Comparative Study of Texture Measures with Classification Based on Feature Distributions," Pattern Recognition, vol. 29, no. 1, pp. 51-59, 1996.

[7] T. Ojala, M. Pietika¨inen, and T. Ma¨enpa¨a¨, "Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns," IEEE Trans. Patt

[8] L. Wang and D. He, Texture classification using texture spectrum, Pattern Recognition, 23(8)905-910, 1990.

[9] Chang-Yeon, J., 2008. Face Detection using LBP features. Final Project Report, 77.

[10] Abrah&atildeo, W., Oliveira, G., Salgueiro, L., Diaz, D.H., Gomez, M.A. and Barbosa, J., A comparison of Haar-like, LBP and HOG approaches to concrete and asphalt runway detection in high resolution imagery.

[11] docs.opencv.org/2.4/modules/contrib/doc/fa cerec/facerec_tutorial.html

[12] Sarabjit Singh, AmritpalKaur, Taqdir, A Face Recognition Technique using Local Binary Pattern Method, International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 3, March 2015.

[13] https://www.tutorialspoint.com/android/android_overview.htm

[14] Ortega-Garcia, J., Bigun, J., Reynolds, D. and Gonzalez-Rodriguez, J., 2004. Authentication gets personal with biometrics. *IEEE signal processing magazine*, *21*(2), pp.50-62.

[15] Delbiaggio, N., 2017. A comparison of facial recognition's algorithms.

[16] Zhao, W., Chellappa, R., Phillips, P.J. and Rosenfeld, A., 2003. Face recognition: A literature survey. *ACM computing surveys (CSUR)*, *35*(4), pp.399-458.

[17] Pandey, S., Singh, K., Wadkar, M. and Vadalkar, H., SMART APPLICATION FOR ATTENDANCE MARKING SYSTEM USING FACIAL RECOGNITION.

[18] Kar, N., Debbarma, M.K., Saha, A. and Pal, D.R., 2012. Study of implementing automated attendance system using face recognition technique. *International Journal of computer and communication engineering*, *1*(2), p.100.

[19] Joseph, J. and Zacharia, K.P., 2013. Automatic attendance management system using face recognition. *International Journal of Science and research*, *2*(11), pp.328-330.

[20] Kutty, N.M. and Mathai, S., Face Recognition-A Tool for Automated Attendance System.

[21] Balcoh, N.K., Yousaf, M.H., Ahmad, W. and Baig, M.I., 2012. Algorithm for efficient attendance management: Face recognition based approach. *IJCSI International Journal of Computer Science Issues*, *9*(4), pp.146-150.