A PROJECT REPORT

On

FACE DETECTION ATTENDANCE SYSTEM

For the partial fulfillment of the award of the degree of

Bachelor of Technology

(Computer Science & Engineering)

Session 2017-2021

Submitted By

Roopali Khandelwal

S17CSE039

Department of CSE, NGFCET, Palwal

Supervised By

Dr. Kuldeep Tomar

Ms. Shivani Sharma

Department of CSE, NGFCET, Palwal

Submitted To

J.C. Bose University of Science & Technology, YMCA Faridabad

ACKNOWLEDGEMENT

I take immense pleasure in thanking our honorable **Vice-Chairman, Mr. Ashwini Prabhakar** for having permitted us to carry out this work. We are also very thankful for his constant motivation and guidance.

I am highly indebted to **Dr. Sharat Kaushik, Director-Principal** for his guidance, motivation and we also thank him for all the resources provided to us, as well as for providing necessary information regarding the work & his valuable support in completing the work.

I express my sincere gratitude to **Dr. Kuldeep Tomar, Professor & HOD (CSE) and Ms. Shivani Sharma, Assistant Professor (CSE)** for their guidance and especially for their emphasis on systematic approach, details and rigor in the process of work. I also wish to thank them for providing his valuable time, guidance and all the support throughout my work.

I would also like to thank to all faculty members of my department with whom I had fruitful interactions.

My thanks and appreciations also go to my classmates in exploring this and people who have willingly helped us out with their abilities.

Roopali Khandelwal

S17CSE039

B.Tech. 6TH semester

TABLE OF CONTENT

LIST OF FIGURES

ABBREVIATION

FDAS	-Face Detection Attendance System
LBP	-Local Binary Pattern
LBPH	-Local Binary Pattern Histograms
HOG	-Histograms of Oriented Gradients
OpenCV	-Open Computer Vision
PIL	-Python Imaging Library
CSV	-Comma Separated Values

ABSTRACT

The attendance management system is a difficult process if it is done manually. The smart and automated attendance system using the various ways of biometrics. Face recognition is one of them. By using this system, the issue of fake attendance and proxies can be solved. The major steps in this system are detecting the comparison of detected faces that can be done by crosschecking with the database of student's faces. This smart system will be an effective way to maintain the attendance and records of students. Attendance automation has become one of the most important needs in educational institutions and workplaces across the world since it saves time and accuracy too. The face recognition system needs the least human cooperation and is viable too. Here input to the system is a video and output is an excel sheet with attendance of the students in the video.

Keywords- Attendance, Face recognition, smart system, biometrics

CHAPTER 1
INTRODUCTION

Face Recognition technique is one of the most efficient biometric techniques for the identification of people [1]. We can utilize it in the field of education for managing the attendance of students. There are lots of colleges and schools in which thousands of students are taking education. In every classroom there are about ninety to a hundred students are studying. Also in every few days, a new school or college is opened. To maintain the attendance and records of these so many numbers of students is a very difficult task.

The process of this face detection system is divided into various steps, but the important steps are the detection of face and recognition of a face. Firstly, to mark the attendance of students, the image of the students' faces will be required. This image can be snapped from the camera device, which will be placed in the classroom can be covered. This image will act as input to the system. For the effective face detection, the image processing techniques like grayscale conversion of image and histogram equalization [2]. To identify the students sitting on the last rows neatly, the histogram equalization of the image needs to be done.

Face Detection Attendance System (FDAS) is the complete system to record attendance without much interference in very little time. It saves time and effort from the teacher in the university where we only get one hour for one class. FDAS record attendance with the help of face recognition. It marks the attendance of all students in the class by obtaining data from the database and match with present face data and saves the result in the database (Excel Sheet). This system makes the attendance process easy and minimizes the interference of the teachers. Which provides them more time to teach and take full advantage of their period time.

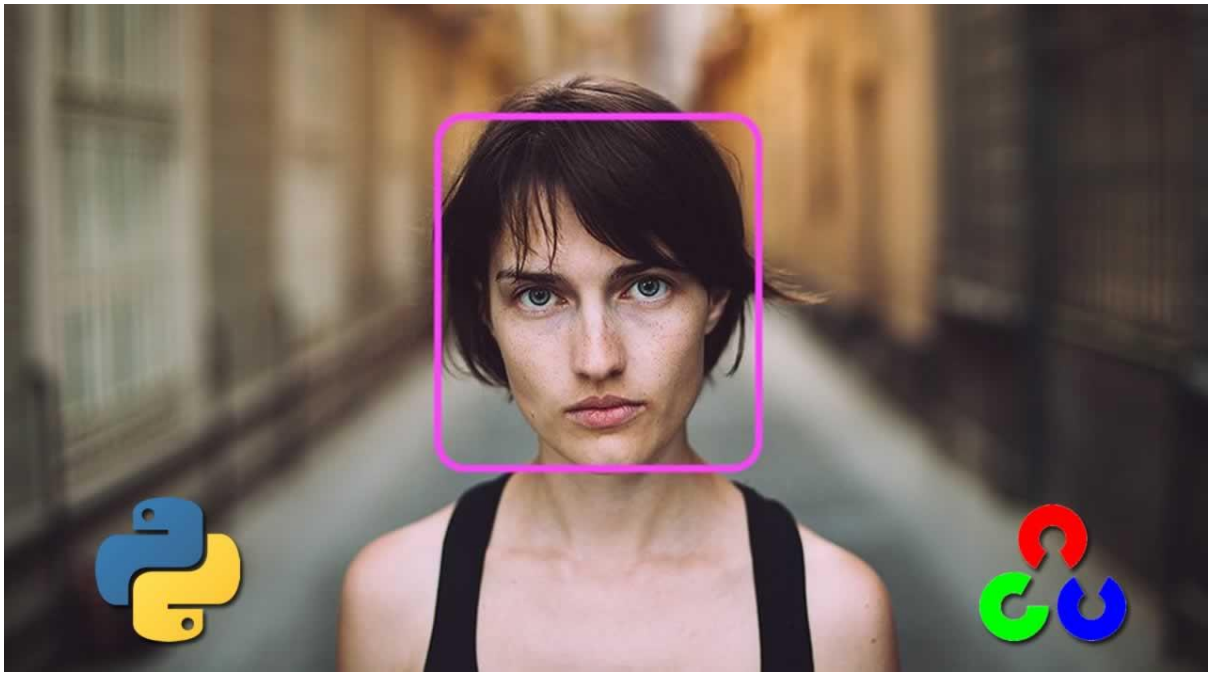


Fig 1.1: Face detection using OpenCV

The main objective of this project is to develop a face recognition based automated student attendance system. To achieve better performance, the test images and training images of this proposed approach are limited to frontal and upright facial images that consist of a single face only. The test images and training images must be captured by using the same device to ensure no quality difference. Also, the students must register in the database to be recognized. The enrolment can be done on the spot through the user-friendly interface.

In a classroom with large numbers of students, it is a very tedious and time-consuming task to take the attendance manually. Therefore, we can implement an effective system that will mark the attendance of students automatically by recognizing their faces.

1.1 PROBLEM STATEMENT

It will be a waste of time to mark attendance in every period. Taking attendance is one of the important and daily tasks which our university teachers must do. This takes an average of 10% time of our period, sometimes more than that. While taking attendance teachers must perform many tasks if we differentiate them there will be 4 - 5 steps. First, when they come to the class, they need to open their register they call each roll numbers, and sometimes a student cannot able to listen or missed their chance. After that teachers need to call all the absentees for in case there may be any student present and mistakenly marked absent. In the end, the teacher needs to headcount all the present students, headcount should match the number of present students if it's not there may be any case of proxy. To find that student-teacher has to go through all the

steps again.

The traditional student attendance marking technique is often facing a lot of trouble. The face recognition student attendance system emphasizes its simplicity by eliminating classical student attendance marking techniques such as calling student names or checking respective identification cards. There are not only disturbing the teaching process but also cause a distraction for students during exam sessions. Apart from calling names, the attendance sheet is passed around the classroom during the lecture sessions. The lecture class especially the class with many students might find it difficult to have the attendance sheet being passed around the class.

Thus, face recognition student attendance system is proposed to replace the manual signing of the presence of students who are burdensome and cause students to get distracted to sign for their attendance. Furthermore, the face recognition based automated student attendance system able to overcome the problem of fraudulent approach and lecturers does not have to count the number of students several times to ensure the presence of the students.

1.2 Scope

Our project targets the students of different academic levels and faculty members. The main constraint we faced is distinguishing between identical twins. This situation is still a challenge to biometric systems, especially facial recognition technology. According to Phillips and his co-researcher paper [2] to get the best results of the algorithms your system employed, they should run under certain conditions for taken pictures (i.e… age, gender, expressions, studio environmental, etc.) otherwise, the problem is still ongoing.

1.3 Objective

Our primary goal is to help the lecturers, improve, and organize the process of track and manage student attendance and absenteeism. Additionally, we seek to:

- It provides a valuable attendance service for both teachers and students.
- Reduce manual process errors by providing an automated and reliable attendance system uses face recognition technology.
- Increase privacy and security in which students cannot presenting themselves or his friend while they are not.
- Produce monthly reports for lecturers.

- Flexibility, Lectures capability of editing attendance records.

1.4 Review of Literature

In the LBP approach for texture classification, the occurrences of the LBP codes in an image are collected into a histogram. The classification is then performed by computing simple histogram similarities. However, considering a similar approach for facial image representation results in a loss of spatial information and therefore one should codify the texture information while retaining also their locations. One way to achieve this goal is to use the LBP texture descriptors to build several local descriptions of the face and combine them into a global description. Such local descriptions have been gaining interest lately which is understandable given the limitations of the holistic representations. These local feature-based methods are more robust against variations in pose or illumination than holistic methods [3].

The basic methodology for LBP based face description proposed by Ahonen et al. (2006) is as follows: The facial image is divided into local regions and LBP texture descriptors are extracted from each region independently. The descriptors are then concatenated to form a global description of the face, as shown in Fig. 1.

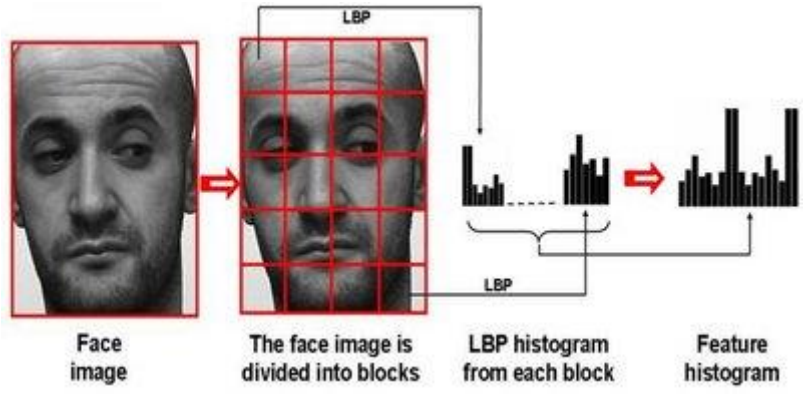


Fig 1.2: Face description with local binary patterns

This histogram effectively has a description of the face on three different levels of locality: the LBP labels for the histogram contain information about the patterns on a pixel-level, the labels are summed over a small region to produce information on a regional level and the regional histograms are concatenated to build a global description of the face [4].

It should be noted that when using the histogram-based methods the regions do not need to be rectangular. Neither do they need to be of the same size or shape, and they do not necessarily have to cover the whole image. It is also possible to have partially overlapping regions [5].

The two-dimensional face description method has been extended into the spatiotemporal domain (Zhao and Pietikäinen 2007). Fig. 1 depicts the facial expression description using LBP-TOP. Excellent facial expression recognition performance has been obtained with this approach [6].

Since the publication of the LBP based face description, the methodology has already attained an established position in face analysis research and applications. A notable example is the illumination-invariant face recognition system proposed by Li et al. (2007), combining NIR imaging with LBP features and Adaboost learning. Zhang et al. (2005) proposed the extraction of LBP features from images obtained by filtering a facial image with 40 Gabor filters of different scales and orientations, obtaining outstanding results. Hadid and Pietikäinen (2009) used spatiotemporal LBPs for face and gender recognition from video sequences, while Zhao et al. (2009) adopted the LBP-TOP approach to visual speech recognition achieving leading-edge performance without error-prone segmentation of moving lips [7].

CHAPTER 2

PROBLEM ANALYSIS

This project involves taking attendance of the student using a biometric face recognition software. The main objectives of this project are:

- Capturing the dataset: Herein, we need to capture the facial images of a student and store it into the database.
- Training the dataset: The dataset needs to be trained by feeding it with algorithms so that it correctly identifies the face.
- Face recognition: Based on the data captured, the model is then tested. If the face is present in the database, it should be correctly identified. If not,
- Marking attendance: Marking the attendance of the right person into the excel sheet. The model must be trained well to increase its accuracy.

2.1 Product Definition

The system is being developed for deploying an easy and secure way of taking down attendance. The software would first capture an image of all the authorized persons and stores the information into the database. The system would then store the image by mapping it into a face coordinate structure. Next time whenever the registered person enters the premises the system recognizes the person and marks his attendance.

2.2 Feasibility Study

When it comes to feasibility, it is feasible for a mid-size organization to a large organization as we are dealing with time-saving and manpower, we can say that this is an onetime investment kind of service where we install the system and camera and then all we need to do is use the system and keep maintaining and improve the features [8].

Currently, either manual or biometric attendance systems are being used in which manual is hectic and time-consuming. The biometric serve one at a time, so there is a need for such a system that could automatically mark the attendance of many persons at the same time.

This system is cost-efficient, no extra hardware required just a daily laptop, mobile or tablet, etc. Hence it is easily deployable [9].

There could be some price of cloud services when the project is deployed on the cloud.
The work of the administration department to enter the attendance is reduced and stationary cost so every institute or organization will opt for such time and money-saving system.
Not only in institutes or organizations, but it can also be used at any public place or entry-exit gates for advanced surveillance [10].

2.3 Existing System

For now, there is not a very used existing system that is present for marking attendance online via facial recognition however facial recognition software is widely used for security purposes and person counting in a casino and other public places. No doubt later this system can be implemented when there will be growth in the reliability of facial detection software [11].

Coming to our project which has been made using open cv library, the software identifies 80 nodal points on a human face. In this context, nodal points are endpoints used to measure variables of a person's face, such as the length or width of the nose, the depth of the eye sockets, and the shape of the cheekbones. The system works by capturing data for nodal points on a digital image of an individual's face and storing the resulting data as a faceprint. The faceprint is then used as a basis for comparison with data captured from faces in an image or video.

Face recognition consists of two steps, in first step faces are detected in the image and then these detected faces are compared with the database for verification. Several methods have been proposed for face detection.

The efficiency of the face recognition algorithm can be increased with the fast face detection algorithm. Our system utilized the detection of faces in the classroom image. Face recognition techniques can be divided into two types Appearance-based which uses texture features that are applied to the whole face or some specific Regions, other is Feature is using classifiers which uses geometric features like mouth, nose, eyes, eyebrows, cheeks, and relation between them.

CHAPTER 3

SOFTWARE REQUIREMENT ANALYSIS

3.1 Introduction

The main purpose of preparing this software is to give a general insight into the analysis and requirement of the existing system or situation and for determining the operating characteristics of the system.

3.2 General Description

This project requires a computer system with the following software's:

- Operating System – minimum Window 7 or later (latest is best)
- Minimum Python 3.7 or later (including all necessary libraries)
- Microsoft Excel 2007 or later.

3.3 Specific Requirements

This face recognition attendance system project requires the OpenCV-a python library with a built-in LBPH face recognizer for training the dataset captured via the camera.

Coming to the technical feasibility following are the requirements: Hardware and Software Requirements.

Processor:	Intel Processor i3 (latest is recommended)
Ram:	4 GB (Higher would be good)
Hard disk:	40 GB
Monitor:	RBG led
Keyboard:	Basic 108 key keyboard
Mouse:	Optical Mouse (or touchpad would work)
Camera:	4 Megapixel Webcam (Or more)

CHAPTER 4

DESIGN

4.1 System Design

In the first step, the image is captured from the camera. There are illumination effects in the captured image because of different lighting conditions and some noise which is to be removed before going to the next steps. Histogram normalization is used for contrast enhancement in the spatial domain. A median filter is used for the removal of noise in the image. There are other techniques like FFT and low pass filter for noise removal and smoothing of the images, but the median filter gives good results.

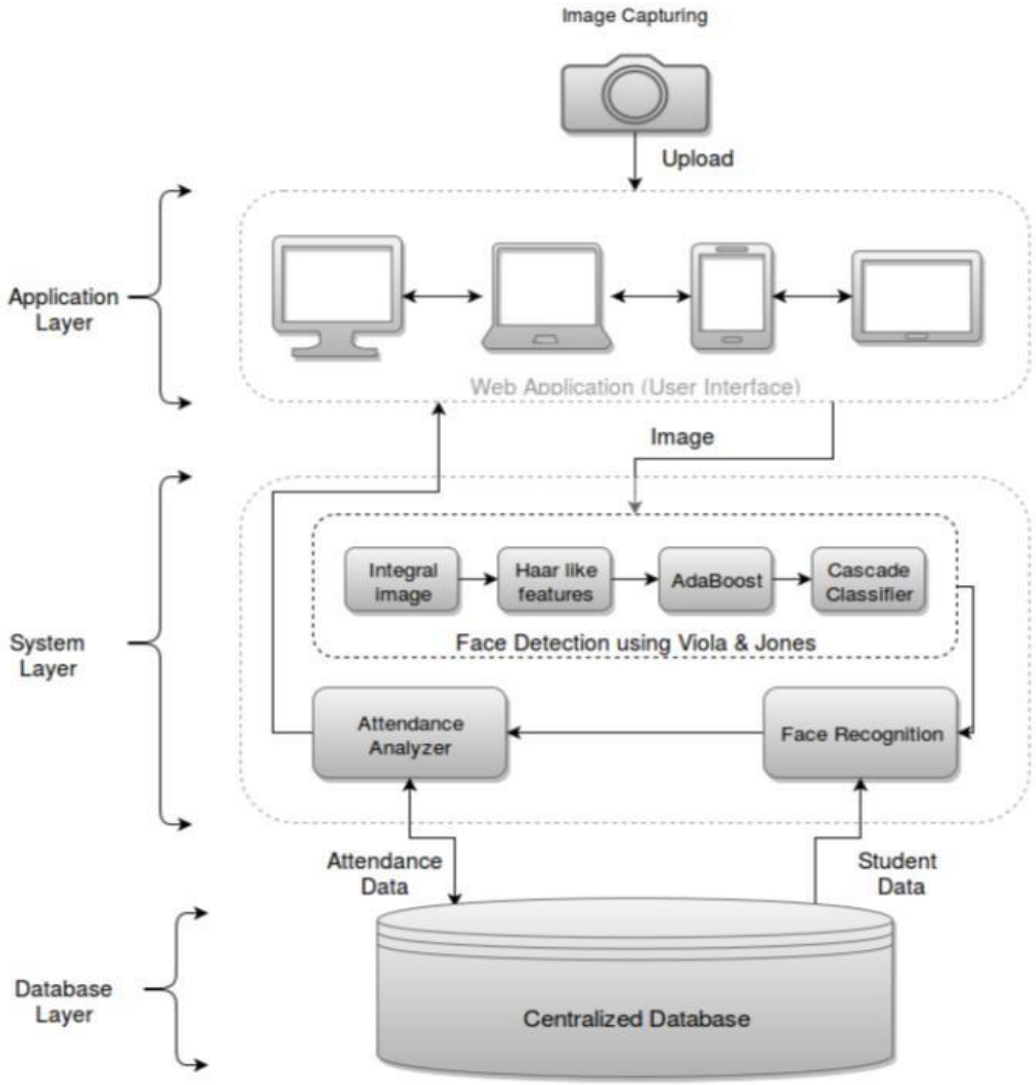


Fig 4.1: Database Design

4.2 Design Notation

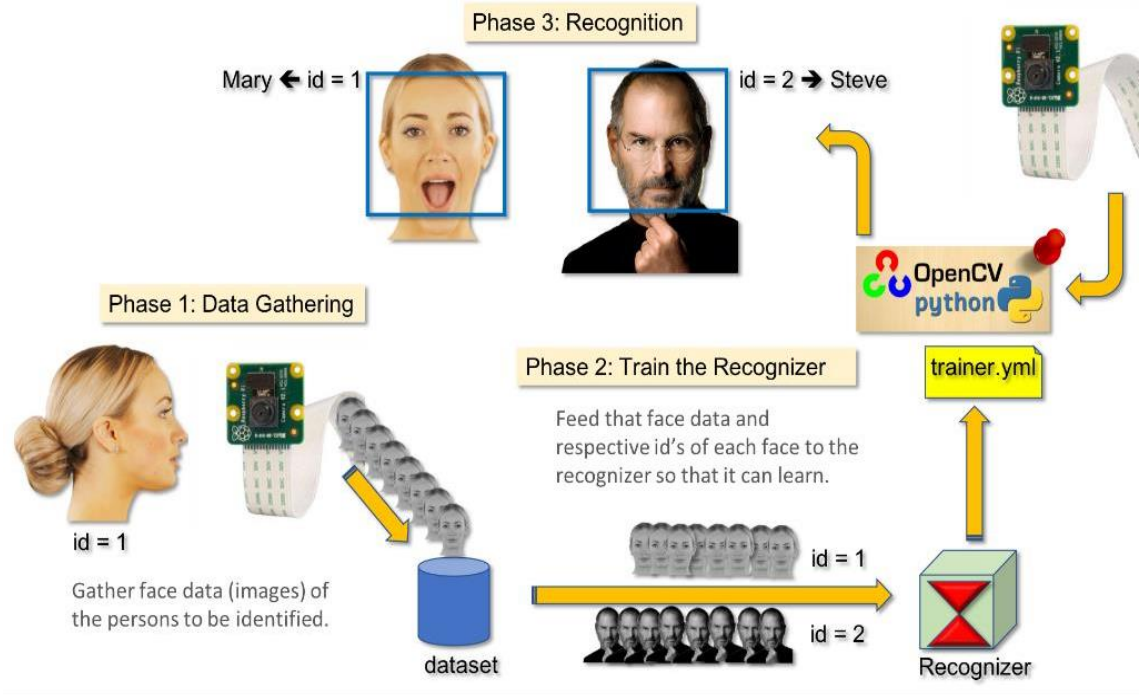


Fig 4.2: Design Notation

CHAPTER 5

TESTING

5.1 Functional Testing

Functional testing is done at every function level, for example, there is a function named os.path.abspath which is responsible to find the directory for the dataset, that function is tested if the directory is created or not. Similarly, all the functions are tested separately before integrating.

5.2 Structural Testing

Structural testing is performed after we integrated each module. After integrating the path creation function and capture image function then we tested that after capturing the image the photos were saved in the correct directory that was created by os.path.abspath.

5.3 Levels of Testing

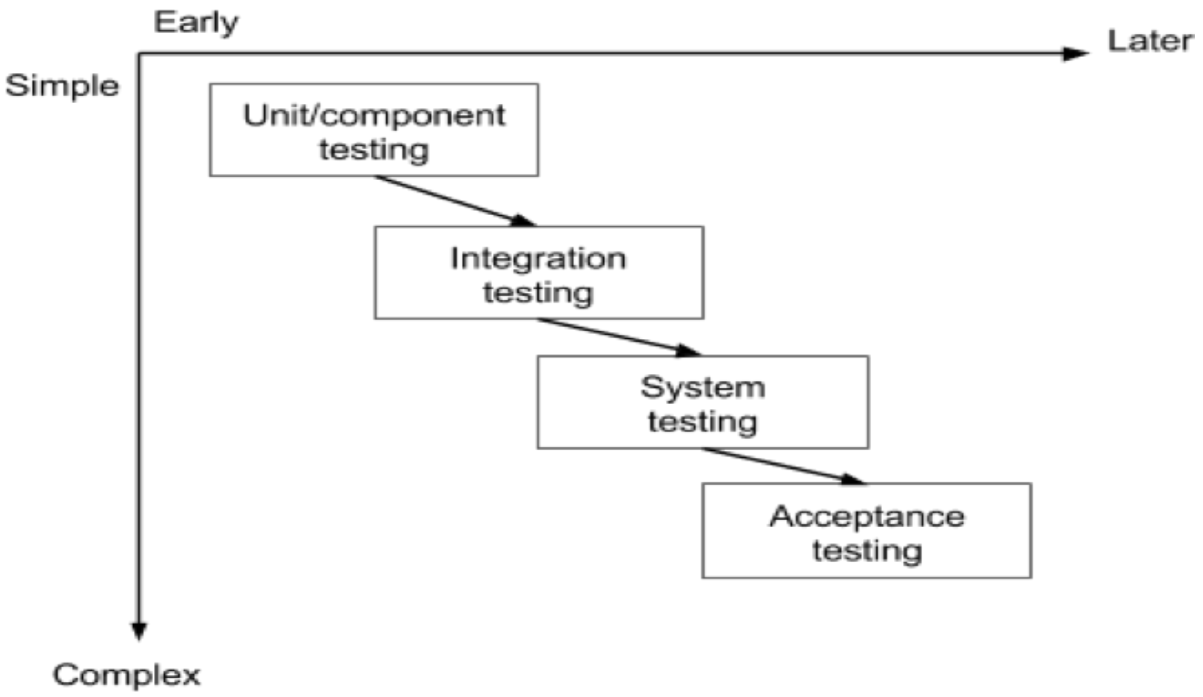


Fig 5.1: Levels of Testing

Unit testing has been performed on the project by verifying each module independently, isolating it from the others. Each module is fulfilling the requirements as well as the desired functionality. Integration testing would be performed to check if all the functionalities are working after integration.

5.4 Testing the Project

Testing early and testing frequently is well worth the effort. By adopting an attitude of constant alertness and scrutiny in all your projects, as well as a systematic approach to testing, the tester can pinpoint any faults in the system sooner, which translates in less time and money wasted later.

CHAPTER 6

IMPLEMENTATION

6.1 Implementation of the project

The complete project is implemented using python 3.7 (or later), the main library used was OpenCV, NumPy, pandas, shutil and ms excel is used for database purpose to store the attendance.

Capturing the dataset

The first and foremost module of the project is capturing the dataset. When building an on-site face recognition system and when you must have physical access to a specific individual to assemble model pictures of their face, you must make a custom dataset. Such a framework would be necessary for organizations where individuals need to physically appear and attend regularly.

To retrieve facial images and make a dataset, we may accompany them to an exceptional room where a camcorder is arranged to (1) distinguish the (x, y)- directions of their face in a video stream and (2) store the frames containing their face to the database. We may even play out this process over days or weeks to accumulate instances of their face in:

- Distinctive lighting conditions
- Times of day
- Mind-sets and passionate states to make an increasingly differing set of pictures illustrative of that specific individual's face.

This Python script will:

1. Access the camera of the system
2. Detect faces
3. Write the frame containing the face to a dataset

Face Detection using OpenCV

In this project we have used OpenCV, to be precise, the Haar-cascade classifier for face detection. Haar Cascade is an AI object detection algorithm proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is an AI-based methodology where a cascade function is prepared from a great deal of positive and negative pictures (where positive pictures are those where the item to be distinguished is available, negative is those where it isn't). It is then used to tell objects in different pictures. Fortunately, OpenCV offers pre-trained Haar cascade algorithms, sorted out into classifications (faces, eyes, etc.), based on the pictures they have been prepared on[12].

Presently let us perceive how this algorithm solidly functions. The possibility of Haar cascade is separating features from pictures utilizing a sort of 'filter', like the idea of the convolutional kernel. These filters are called Haar features and resemble that:

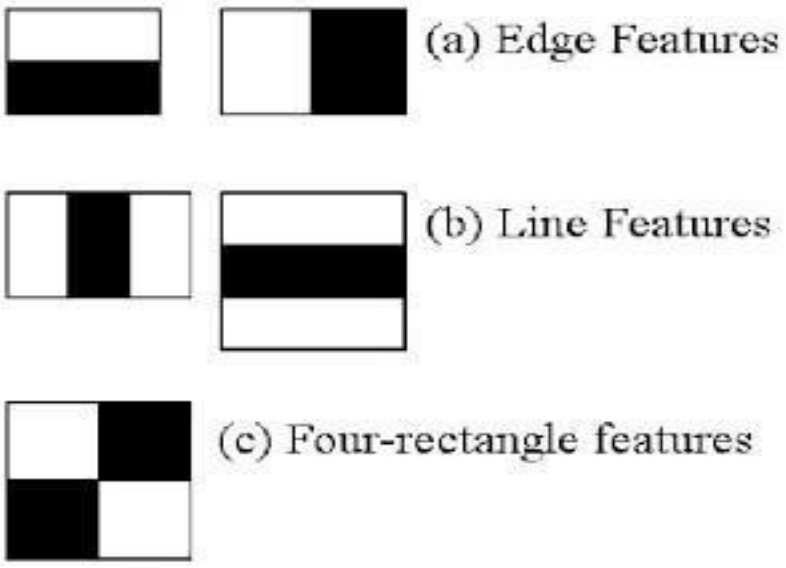


Fig 6.1: Haar features

The idea is passing these filters on the picture, investigating one part at the time. At that point, for every window, all the pixel powers of, separately, white and dark parts are added. Ultimately, the value acquired by subtracting those two summations is the value of the feature extracted. In a perfect world, an extraordinary value of a feature implies it is pertinent. On the

off chance that we consider the Edge (a) and apply it to the accompanying B&W pic:

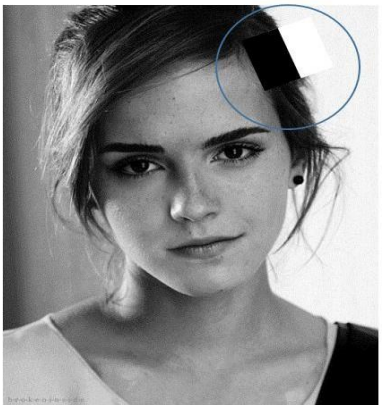


Fig 6.2: B&W pic

We will get a noteworthy value, subsequently, the algorithm will render an edge highlight with high likelihood [13]. The genuine intensities of pixels are never equivalent to white or dark, and we will frequently confront a comparable circumstance:

Fig 6.3: Matrix of pixel

By and by, the thought continues as before: the higher the outcome (that is, the distinction among highly contrasting black and white summations), the higher the likelihood of that window of being a pertinent element.

To give you a thought, even a 24x24 window results in more than 160000 highlights, and windows inside a picture are a ton. How to make this procedure progressively effective? The arrangement turned out with the idea of a Summed-area table, otherwise called Integral Image. It is an information structure and algorithm for producing the sum of values in a rectangular

subset of a matrix. The objective is to diminish the number of calculations expected to get the summations of pixel intensities in a window.

The following stage additionally includes proficiency and enhancement. Other than being various, features may likewise be unessential. Among the features we get (that are more than 160000), how might we choose which ones are great? The response to this inquiry depends on the idea of Ensemble strategy: by consolidating numerous algorithms, weak by definition, we can make a solid algorithm. This is cultivated utilizing Ad boost which both chooses the best features and prepares the classifiers that utilize them. This algorithm builds a strong classifier as a simple mix of weighted basic weak classifiers [14].

We are nearly done. The last idea which should be presented is the last component of advancement. Even though we decreased our 160000+ highlights to an increasingly reasonable number, the latter is still high: applying every one of the features on every one of the windows will consume a great deal of time. That is the reason we utilize the idea of Cascade of classifiers: rather than applying every one of the features on a window, it bunches the features into various phases of classifiers and applies individually. On the off chance that a window comes up short (deciphered: the distinction among white and dark summations is low) the primary stage (which typically incorporates few features), the algorithm disposes it: it won't think about outstanding features on it. If it passes, the algorithm applies the second phase of features and continues with the procedure.

Storing of data

To store the captured dataset, we use the CSV file. A CSV file (Comma Separated Values file) is a type of plain text file that uses specific structuring to arrange tabular data. Because it’s a plain text file, it can contain only actual text data—in other words, printable ASCII or Unicode characters.

The structure of a CSV file is given away by its name. Normally, CSV files use a comma to separate each specific data value.

In general, the separator character is called a delimiter, and the comma is not the only one used. Other popular delimiters include the tab (\t), colon (:), and semi-colon (;) characters. Properly

parsing a CSV file requires us to know which delimiter is being used.

CSV files are normally created by programs that handle large amounts of data. They are a convenient way to export data from spreadsheets and databases as well as import or use them in other programs. For example, you might export the results of a data-mining program to a CSV file and then import that into a spreadsheet to analyze the data, generate graphs for a presentation, or prepare a report for publication.

CSV files are very easy to work with programmatically. Any language that supports text file input and string manipulation (like Python) can work with CSV files directly.

The CSV library provides functionality to both reads from and writes to CSV files. Designed to work out of the box with Excel-generated CSV files, it is easily adapted to work with a variety of CSV formats. The CSV library contains objects and other code to read, write, and process data from and to CSV files.

The face recognition systems can operate basically in two modes:

- Verification or authentication of a facial image: it compares the input facial image with the facial image related to the user which is requiring the authentication. It is a 1x1 comparison.
- Identification of facial recognition: it compares the input facial image with all facial images from a dataset to find the user that matches that face. It is a 1xN comparison.

There are different types of face recognition algorithms, for example:

- Eignefaces (1991)
- Local Binary Pattern Histograms (LBPH) (1996)
- Fisherfaces (1997)
- Scale Invariant Feature Transform (SIFT) (1999)
- Speed Up Robust Features (SURF) (2006)

This project uses the LBPH algorithm.

Training Dataset

As it is one of the easier face recognition algorithms, I think everyone can understand it without major difficulties.

Local Binary Pattern (LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

It was first described in 1994 (LBP) and has since been found to be a powerful feature for texture classification. It has further been determined that when LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

Using the LBP combined with histograms we can represent the face images with a simple data vector.

DETAILED EXPLANATION OF WORKING OF LBPH

1. **Parameters**: the LBPH uses 4 parameters:

- **Radius**: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.
- **Neighbors**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.
- **Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.
- **Grid Y**: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

2. **Training the Algorithm**: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let's see the LBPH computational steps.

3. **Applying the LBP operation**: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameter's **radius** and **neighbors**.

The image below shows this procedure:

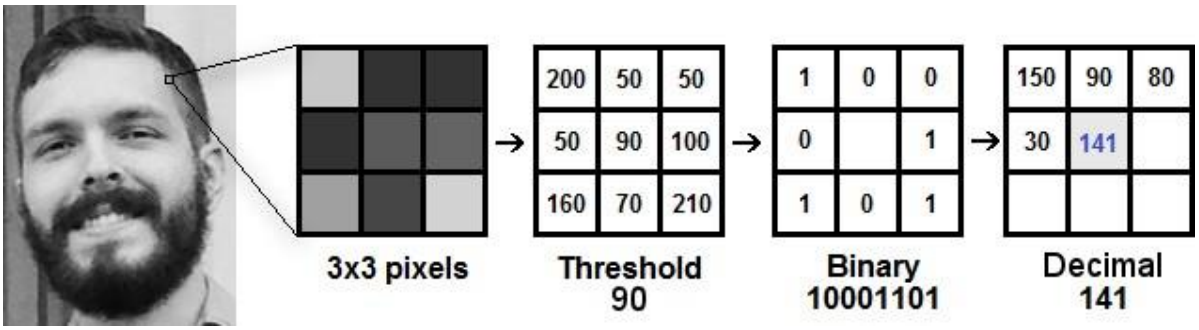


Fig 6.4: Working of LBPH

Based on the image above, let's break it into several small steps so we can understand it easily:

- Suppose we have a facial image in grayscale.
- We can get part of this image as a window of 3x3 pixels.
- It can also be represented as a 3x3 matrix containing the intensity of each pixel (0~255).
- Then, we need to take the central value of the matrix to be used as the threshold.
- This value will be used to define the new values from the 8 neighbors.
- 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.
- 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). Note: some authors use other approaches to concatenate the binary values (e.g. clockwise direction), but the result will be the same.
- Then, we convert this binary value to a decimal value and set it to the central value of the matrix, which is a pixel from the original image.
- At the end of this procedure (LBP procedure), we have a new image that represents better the characteristics of the original image.
- **Note**: The LBP procedure was expanded to use a different number of radius and neighbors, it is called Circular LBP.

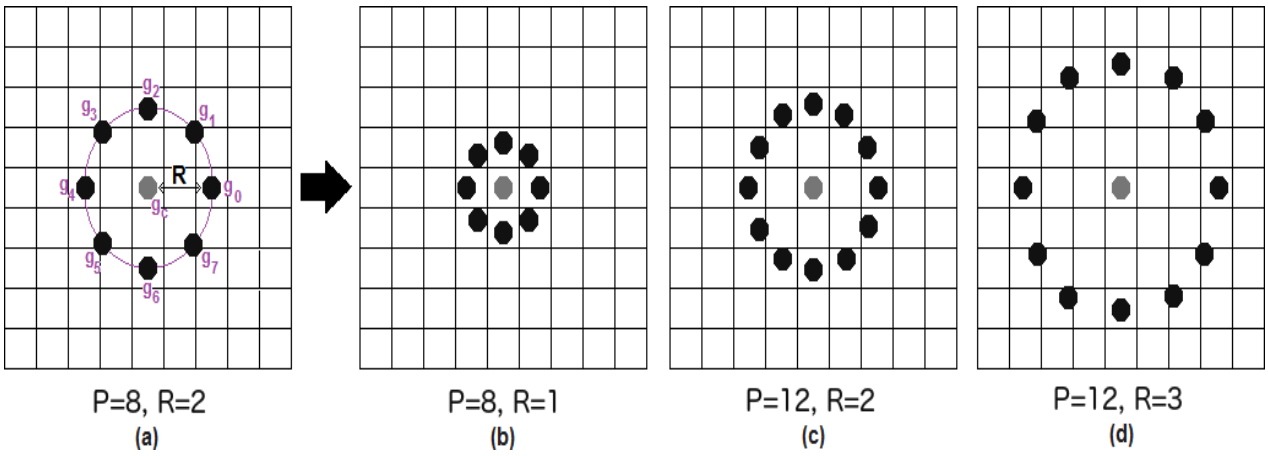


Fig 6.5: Bilinear interpolation

It can be done by using **bilinear interpolation**. If some data point is between the pixels, it uses the values from the 4 nearest pixels (2x2) to estimate the value of the new data point.

Extracting the Histograms: Now, using the image generated in the last step, we can use

Grid X and **Grid Y** parameters to divide the image into multiple grids, as can be seen in the following image:

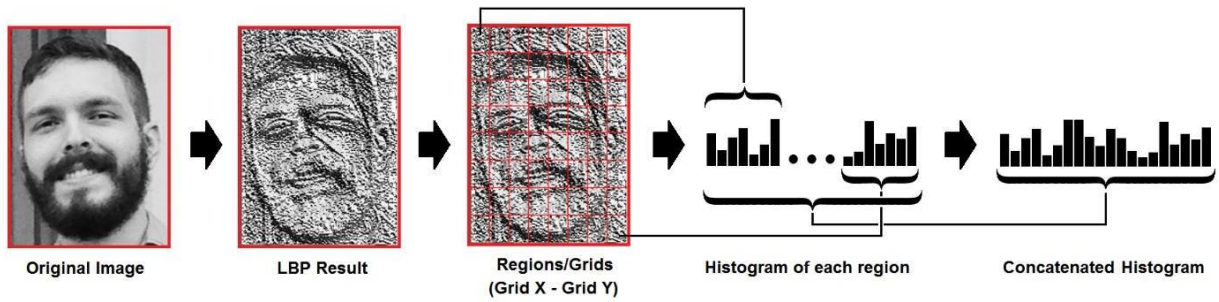


Fig 6.6: Extracting Histograms

Based on the image above, we can extract the histogram of each region as follows:

- As we have an image in grayscale, each histogram (from each grid) will contain only 256 positions (0~255) representing the occurrences of each pixel intensity.
- Then, we need to concatenate each histogram to create a new and bigger histogram. Supposing we have 8x8 grids, we will have 8x8x256=16.384 positions in the final

histogram. The final histogram represents the characteristics of the image's original image

After detecting the face, the image needs to crop as it has only faced nothing else focused. To do so Python Imaging Library (PIL) or also known as Pillow is used. PIL is a free library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.

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 image, we perform the steps again for this new image and creates a histogram that represents the image.

- So, to find the image that matches the input image we just need to compare two histograms and return the image with the closest histogram.
- We can use various approaches to compare the histograms (calculate the distance between two histograms), for example, **Euclidean distance**, **chi-square**, **absolute value**, etc. In this example, we can use the Euclidean distance (which is quite known) based on the following formula:

$$D = \sqrt{\sum_{i=1}^{n}(hist1_i - hist2_i)^2}$$

Fig 6.7: Euclidean distance formula

- 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.
- 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 if the confidence is lower than the threshold defined.

Conclusions

- LBPH is one of the easiest face recognition algorithms.
- It can represent local features in the images.
- It is possible to get great results (mainly in a controlled environment).
- It is robust against monotonic greyscale transformations.
- It is provided by the OpenCV library (Open Source Computer Vision Library).

Marking Attendance

The face(s) that have been recognized are marked as present in the database. Then the entire attendance data is written into an excel sheet that is dynamically created using the CSV library of python. First, an instance is created for excel application and then a new excel workbook is created and an active worksheet of that file is fetched. Then data is fetched and written in the excel sheet.

6.2 Post Implementation and Software Maintained

After the implementation of the software post-implementation can be done by adding more features in the software like SMS tracking of the attendance of a particular student for example if he is absent then an automatic scheduled SMS can be sent to their parent.

About the software maintenance part, only the features can be maintained and improved over time and dataset could be large over time so a better data structure can be used for faster fetching of data could be done, for that purpose cloud storage can be used to minimize the latency of fetching data of a student.

CHAPTER 7
PROJECT LEGACY

7.1 Current Status of Project

The current status of the project contains all the separate modules of capture, training, recognizer, and result. The first module that is capture data set is created to fetch the details of students, write them in the database, capture the photo of the student and name the files accordingly to train the photos to recognize for the attendance later. After capturing the dataset, the next module comes that is train the dataset that trains the photos captured using the LBPH algorithm. The next module marks the attendance and writes attendance in the database as well as an excel file. The last module opens the excel file to see the attendance.

7.2 Remaining Area of Concern

Most of the remaining area of concern is the technical hurdles that come when taking images of a group of students, for that we can do is use better machines that can handle recognizing multiple people at a time, also the camera and its orientation and most important lightening conditions are important for that HDR cameras can be used which can handle backlight and produce better results.

The ambient light or artificial illumination affects the accuracy of face recognition systems. This is because the basic idea of capturing an image depends on the reflection of the light of an object and the higher the amount of illumination, the higher the recognition accuracy.

Another difficulty which prevents face recognition (FR) systems from achieving good recognition accuracy is the camera angle (pose). The closer the image pose is to the front view, the more accurate the recognition.

For face recognition, changes in facial expression are also problematic because they alter details such as the distance between the eyebrows and the iris in case of anger or surprise or change the size of the mouth in the case of happiness.

Simple non-permanent cosmetic alterations are also a common problem in obtaining good recognition accuracy. One example of such cosmetic alterations is make-up which tends to slightly modify the features. These alterations can interfere with contouring techniques used to

perceive facial shape and alter the perceived nose size and shape or mouth size. Other color enhancements and eye alterations can convey misinformation such as the location of eyebrows, eye contrast, and canceling the dark area under the eyes leading to a potential change in the appearance of a person.

Although wearing eyeglasses is necessary for many human vision problems and certain healthy people also wear them for cosmetic reasons but wearing glasses hides the eyes which contain the greatest amount of distinctive information. It also changes the holistic facial appearance of a person.

7.3 Technical and Management lesson learned

The technical lesson that we learned is code should be in the module so that our team can work better on modules and it can be integrated easily and if there is any error we don't have to go through all the code and another lesson that learned is that report should be written while we code the modules that give precise report over the code.

While testing the code sometimes we make changes to the original code and later it becomes a mess when we want to add or remove that tested part. For that, we created a separate test.py file for testing all kinds of changes that we did in the source code.

Coming to Management lessons most of the problems we faced is while creating the report file and integration of code, while integration many errors came that we are not aware of so what we can do is integrate the completed modules at the time when they are completed so that they create fewer errors while integrating with other modules.

CHAPTER 8

USER MANUAL

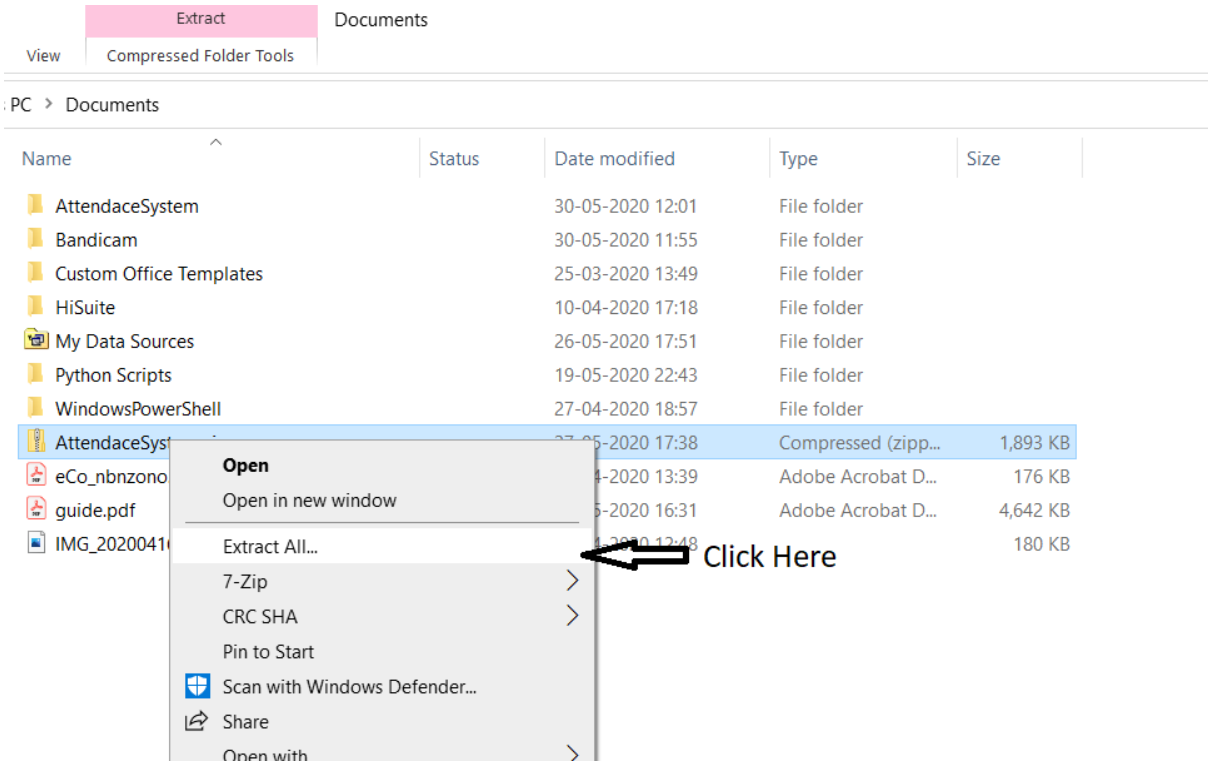
To run the software first we need to capture the image which we do use open cv video capture function that open camera for few seconds and capture only facial part of the image using frontal face haarcascades classifier, A haarcascades is a classifier which is used to detect particular objects form a source that is image or video, in our case, it was frontal face detection so the cascade file contains the information about the facial symmetry of the human face.

After capturing the face, the images need to train, for that there is a separate code that used for training the images what the trainer does is it set the image to the person who's image is.

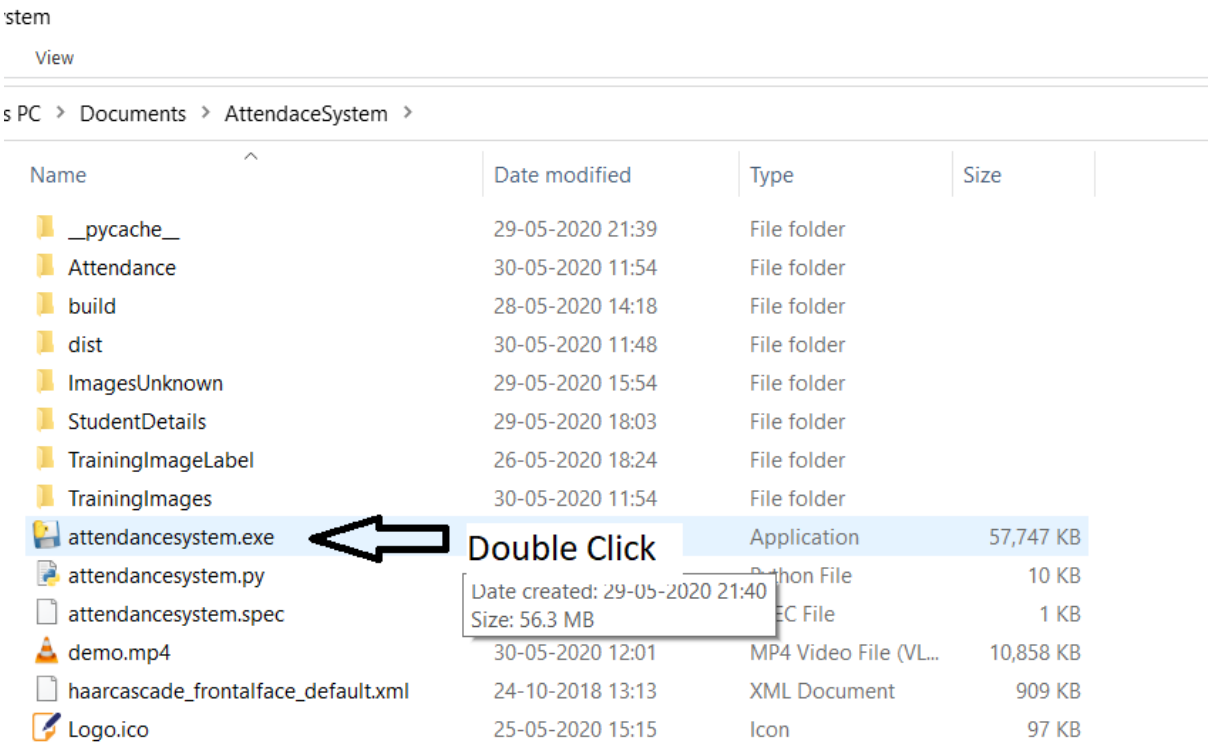
And the last step is recognizing or mark attendance for that open the recognizer python file and it will open a camera window that will capture the person in the frame and the good thing about this algorithm is it can recognize more than 1 person in the frame depending on the lighting and image condition. After that, the recognized person is marked present in the excel sheet.

How to use:

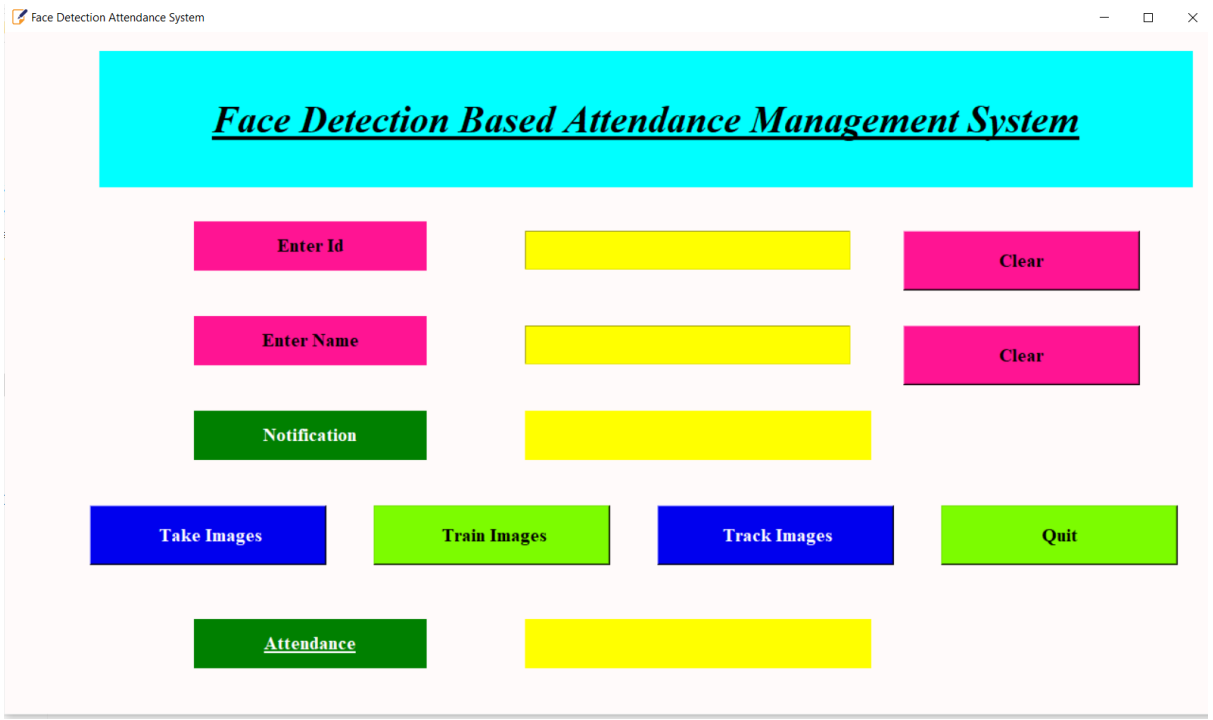
Step 1: Unzip the compressed folder of the software.



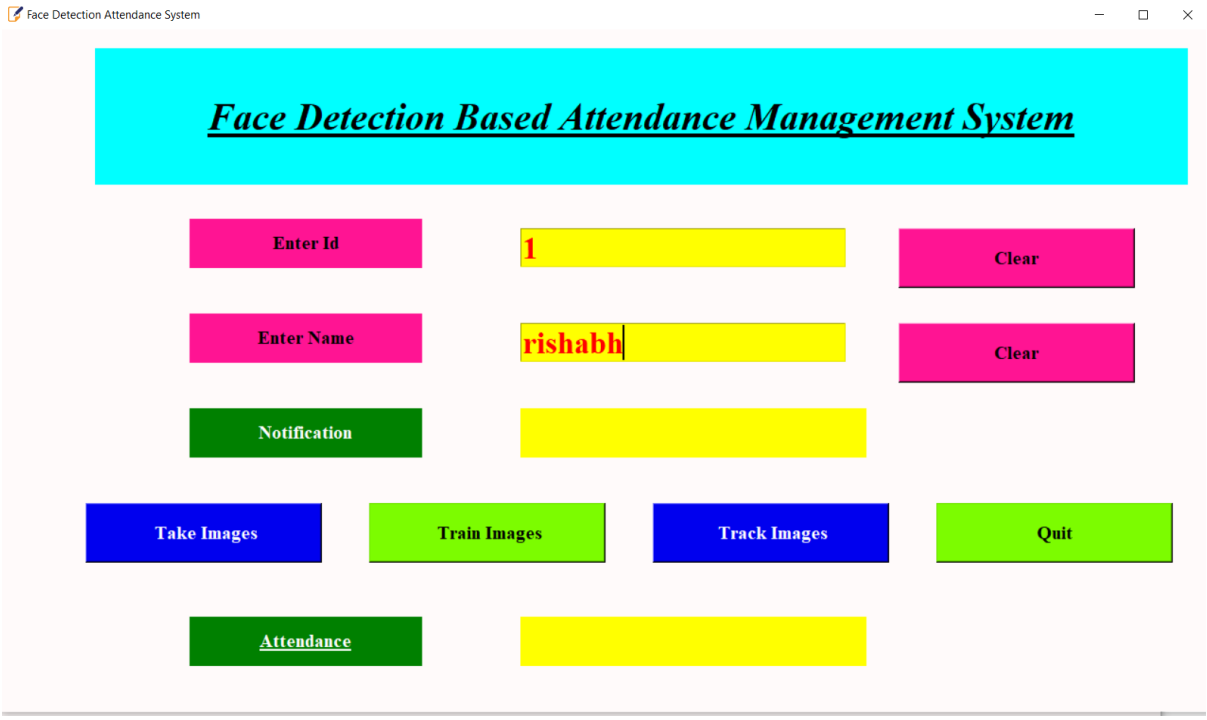
Step 2: Double-click on the “attendancesystem.exe”.



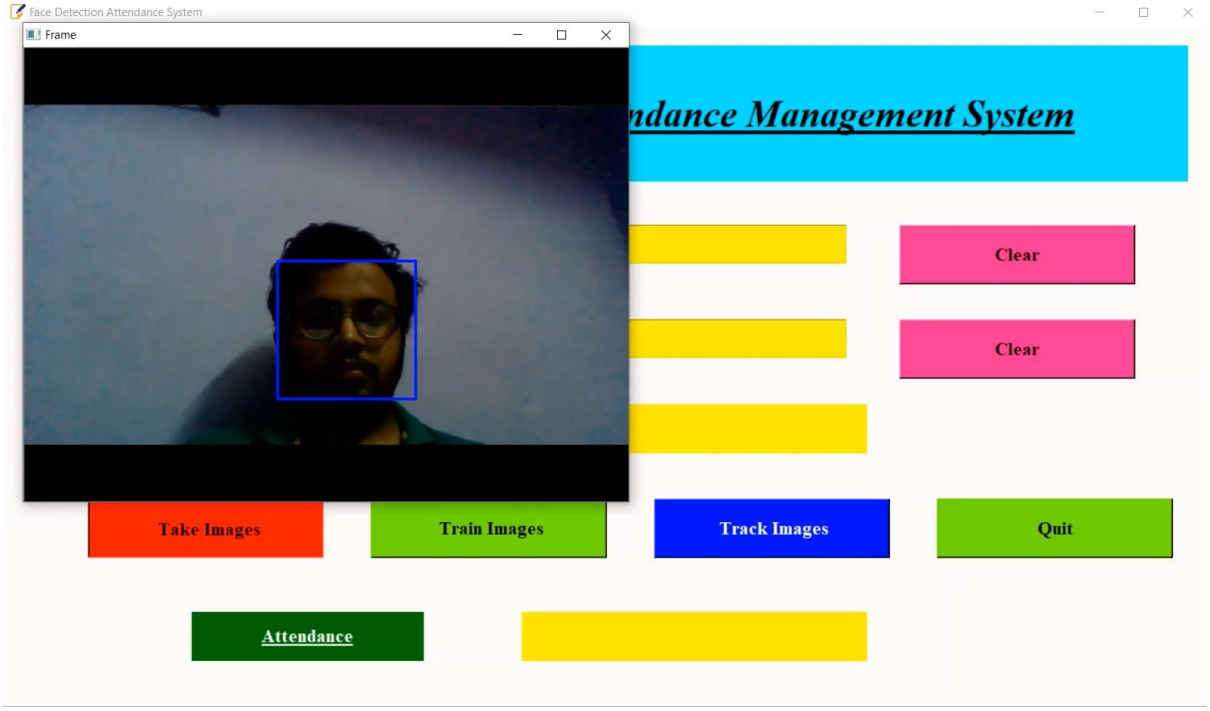
Step 3: Now Enter the Id and name in the corresponding box.



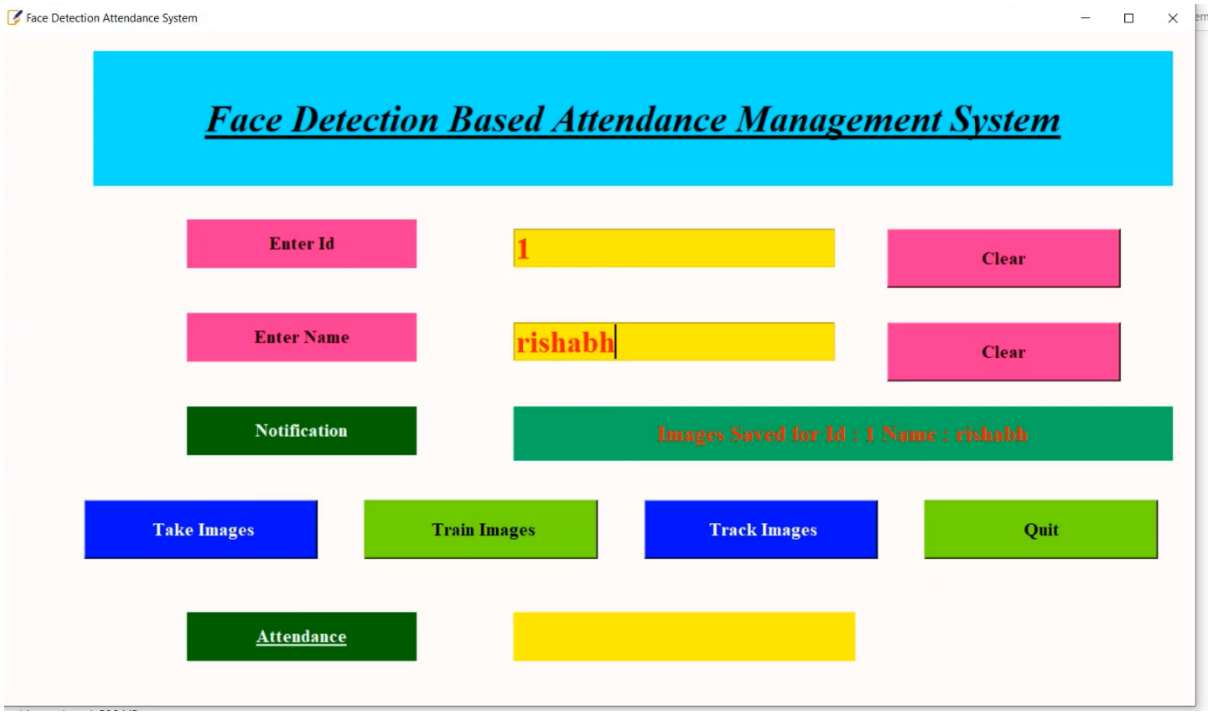
Step 4: Click on the “Take Images” button to take images for students individually. (only for the first time)



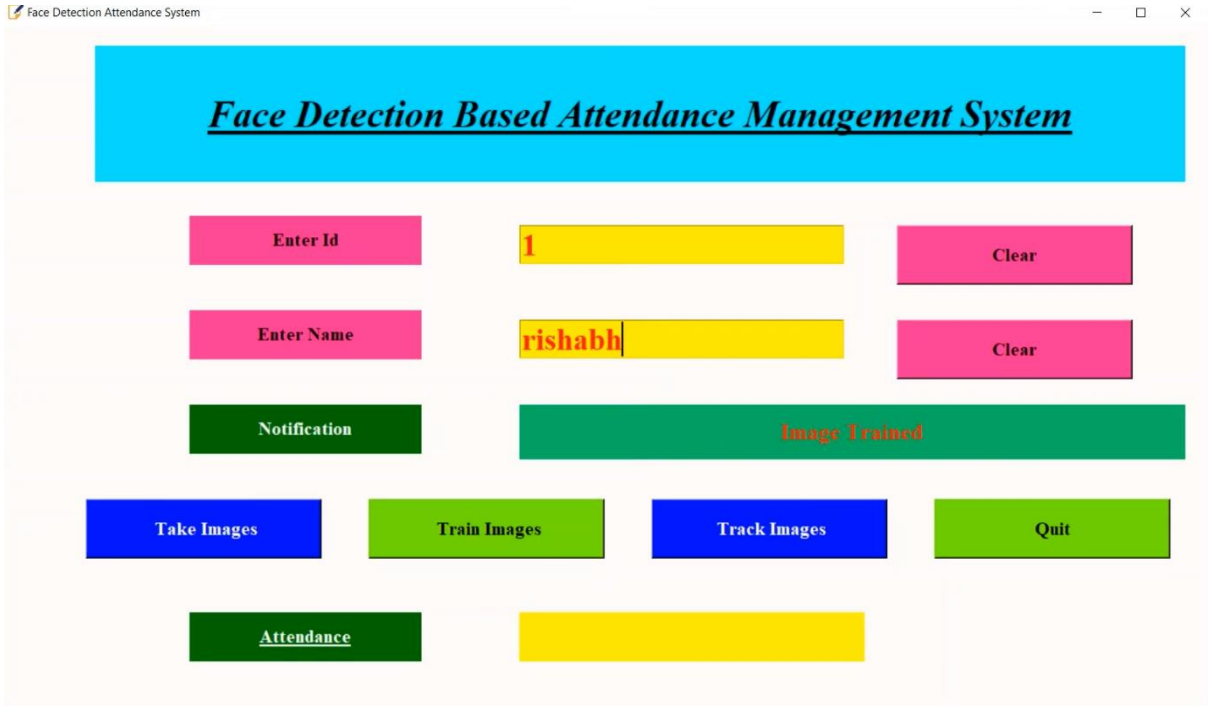
Step 5: Move your face in up, down, left, and right slowly. (only first time)



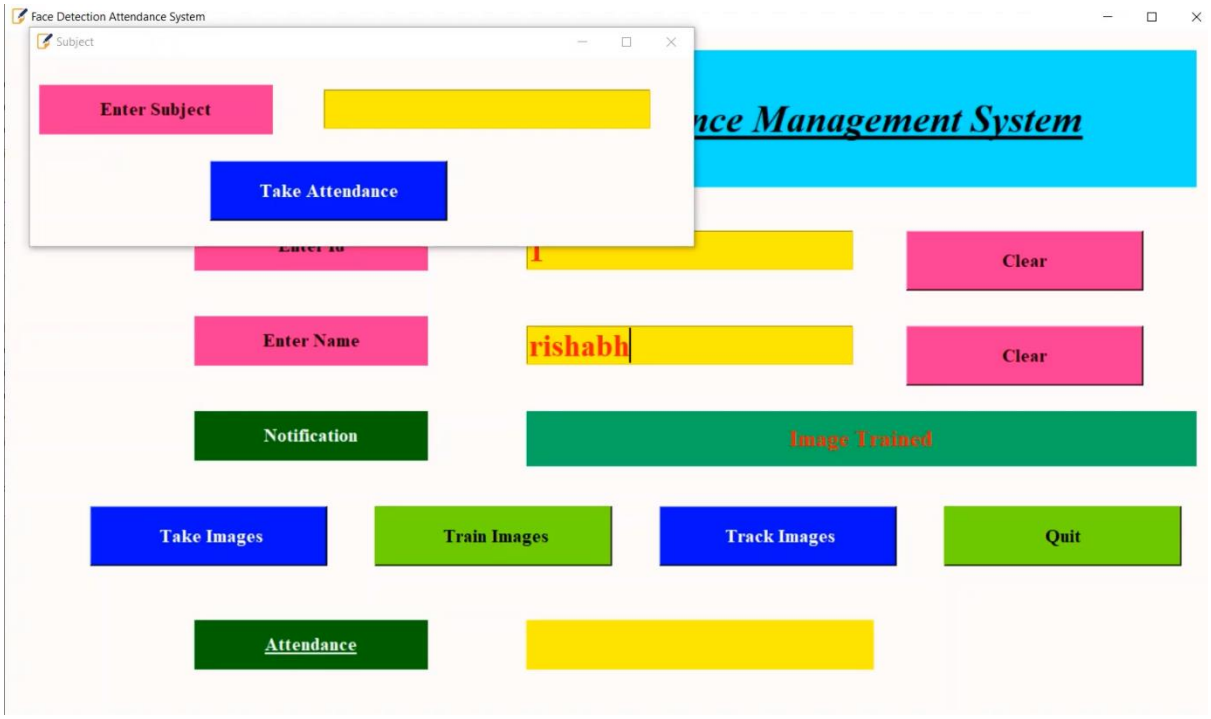
Step 6: After scanning your face the system will close the webcam and show you the notification in the GUI.



Step 7: Then click on "Train Images" to train your image after scanning all the faces of the students.

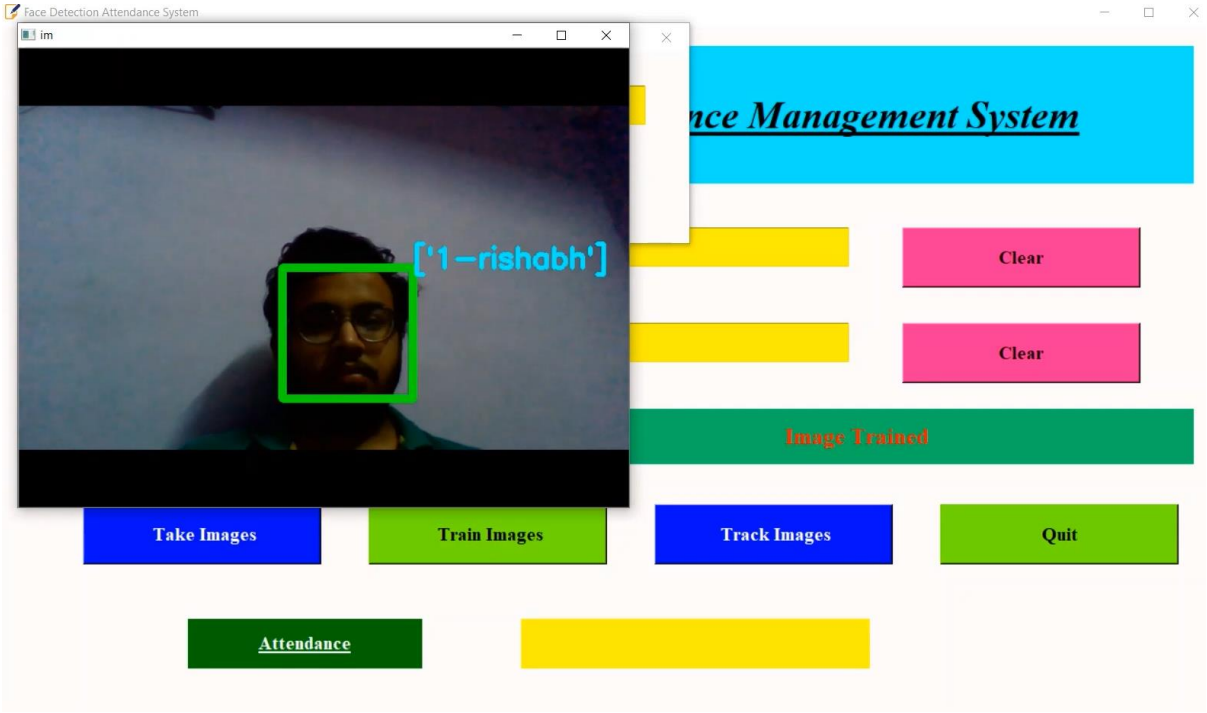


Step 8: Now Click on “Track Images” to take attendance.

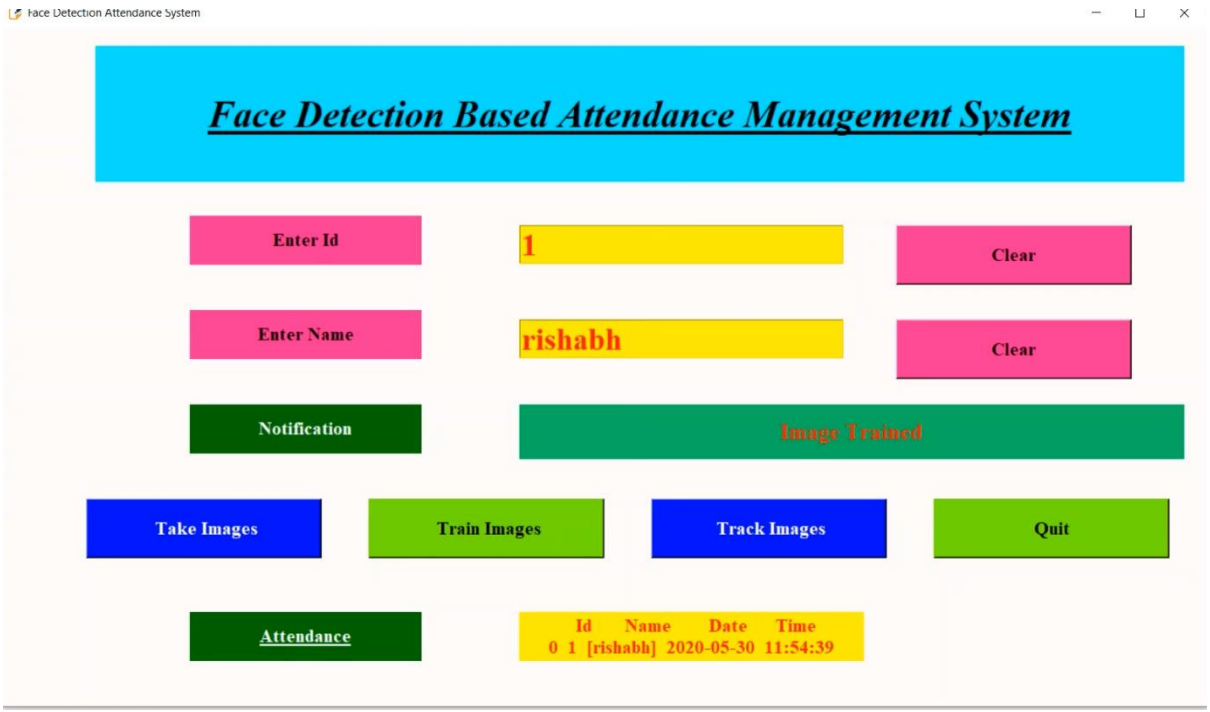


Step 9: Now Enter the Subject name in the box and click on the “Take Attendance” button.

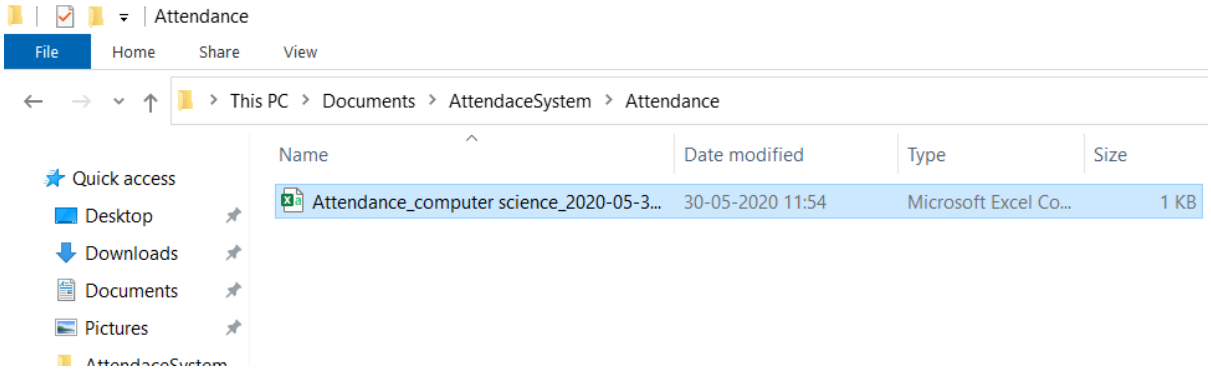
Step 10: Now webcam scans the images against the data and shows their names on their face. (Try to maintain space between two students)



Step 11: After scanning all the faces GUI Will show the list of students available.



Step 12: This system also saves the attendance list in the ‘AttendaceSystem\Attendance’ with date and time.



CHAPTER 9

SOURCE CODE SNAPSHOTS

GUI

```
window = tk.Tk()
window.title("Face Detection Attendance System")
window.geometry('1280x720')
window.iconbitmap('Logo.ico')
dialog_title='QUIT'
dialog_user='Are you sure?'
window.configure(background='snow')
window.grid_rowconfigure(0, weight=1)
window.grid_columnconfigure(0,weight=1)

message = tk.Label(window, text = "Face Detection Based Attendance Management System" , bg="cyan", fg="black", width=50, height=3, font=('times',30,'italic b
message.place(x=100,y=20)

lbl=tk.Label(window, text= "Enter Id", width=20, height = 2, fg="black", bg="deep pink", font=('times',15, 'bold'))
lbl.place(x=200,y=200)
txt= tk.Entry(window,width=20, bg="yellow", fg="red", font=('times', 25, 'bold'))
txt.place(x=550,y=210)

lbl2 =tk.Label(window, text="Enter Name", width= 20, fg="black", bg="deep pink", height=2, font=('times',15, 'bold'))
lbl2.place(x=200, y=300)
txt2= tk.Entry(window,width=20, bg="yellow", fg="red", font=('times', 25, 'bold'))
txt2.place(x=550,y=310)

lbl3 =tk.Label(window, text="Notification", width= 20, fg="white", bg="Green", height=2, font=('times',15, 'bold'))
lbl3.place(x=200, y=400)

message = tk.Label(window, text="", bg="yellow", fg="red", width = 30, height= 2, activebackground = "yellow", font=('times',15, 'bold'))
message.place(x=550,y=400)

lbl3 =tk.Label(window, text="Attendance", width= 20, fg="white", bg="Green", height=2, font=('times',15, 'bold underline'))
lbl3.place(x=200, y=620)

message2= tk.Label(window, text=" ", bg="yellow", fg="red", width = 30, height= 2, activebackground = "yellow", font=('times',15, 'bold'))
message2.place(x=550,y=620)
```

```
clearButton= tk.Button(window, text="Clear", command=clear, fg="black", bg="deep pink", width=20, height=2, activebackground="Red", font=('times',15, 'bold')
clearButton.place(x=950,y=210)

clearButton2= tk.Button(window, text="Clear", command=clear2, fg="black", bg="deep pink", width=20, height=2, activebackground="Red", font=('times',15, 'bold
clearButton2.place(x=950,y=310)

takeImg= tk.Button(window, text=" Take Images", command=TakeImages, fg="white", bg= "blue2", width=20, height=2, activebackground="Red", font=('times',15, 'b
takeImg.place(x=90,y=500)

trainImg= tk.Button(window, text=" Train Images", command=TrainImages, fg="black", bg= "lawn green", width=20, height=2, activebackground="Red", font=('times
trainImg.place(x=390,y=500)

trackImg= tk.Button(window, text=" Track Images", command=subject, fg="white", bg= "blue2", width=20, height=2, activebackground="Red", font=('times',15, 'bo
trackImg.place(x=690,y=500)

quitWindow= tk.Button(window, text="Quit", command=window.destroy, fg="black", bg= "lawn green", width=20, height=2, activebackground="Red", font=('times',15
quitWindow.place(x=990,y=500)

window.mainloop()
```

Capture images

attendancesystem.py - C:\Users\risha\Documents\AttendaceSystem\attendancesystem.py (3.8.3)

File Edit Format Run Options Window Help

```
def TakeImages():
    Id = (txt.get())
    Name = (txt2.get())
    if(Id.isnumeric() and Name.isalpha()):
            cam = cv2.VideoCapture(0)
            detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
            sampleNum = 0
            while (True):
                ret, img = cam.read()
                gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                faces = detector.detectMultiScale(gray, 1.3, 5)
                for (x, y, w, h) in faces:
                    cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                    # incrementing sample number
                    sampleNum = sampleNum + 1
                    # saving the captured face in the dataset folder
                    cv2.imwrite("TrainingImages\ " + Name + "." + Id + '.' + str(sampleNum) + ".jpg",gray[y:y + h, x:x + w])
                    cv2.imshow('Frame', img)
                # wait for 100 miliseconds
                if cv2.waitKey(100) & 0xFF == ord('q'):
                    break
                # break if the sample number is morethan 100
                elif sampleNum > 70:
                    break
            cam.release()
            cv2.destroyAllWindows()
            res = "Images Saved for Id : " + Id + " Name : " + Name
            row = [Id, Name]
            with open('StudentDetails\studentDetails.csv', 'a+',newline='') as csvFile:
                   writer = csv.writer(csvFile)
                   writer.writerow(row)
            csvFile.close()
            message.configure(text=res, bg="SpringGreen3", width=50, font=('times', 18, 'bold'))
    else:
        if(Id.isnumeric()):
            res="Enter Numeric Value Only!!"
            message.configure(text=res)
        elif(Name.isalpha()):
            res="Enter Character Name Only!!"
            message.configure(text=res)
        else:
            res="You have not Enter any Value!!"
            message.configure(text=res)
```

Train Images

```
def TrainImages():
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    harcascadePath="haarcascade_frontalface_default.xml"
    detector=cv2.CascadeClassifier(harcascadePath)
    faces,Id= getImagesAndLabels("TrainingImages")
    recognizer.train(faces,np.array(Id))
    recognizer.save("TrainingImageLabel\Trainner.yml")
    res = "Image Trained"#+",".join(str(f) for f in Id)
    message.configure(text=res)

def getImagesAndLabels(path):
    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
    faces=[]
    Ids=[]
    for imagePath in imagePaths:
        pilImage=Image.open(imagePath).convert('L')
        imageNp=np.array(pilImage, 'uint8')
        Id= int(os.path.split(imagePath)[-1].split(".")[1])
        faces.append(imageNp)
        Ids.append(Id)
    return faces,Ids
```

Mark Attendance

```
def TrackImages():
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    recognizer.read("TrainingImageLabel\Trainner.yml")
    harcascadePath="haarcascade_frontalface_default.xml"
    facesCascade= cv2.CascadeClassifier(harcascadePath)
    df=pd.read_csv("StudentDetails\studentDetails.csv")
    cam= cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names= ['Id', 'Name', 'Date', 'Time']
    attendance= pd.DataFrame(columns = col_names)
    fps=0
    while True:
        ret,im = cam.read()
        gray= cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
        faces= facesCascade.detectMultiScale(gray, 1.3, 5)
        for(x,y,w,h) in faces:
            Id,conf=recognizer.predict(gray[y:y+h, x:x+w])
            fps=fps+1
            if (conf<50):
                ts=time.time()
                date=datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa=df.loc[df['Id'] == Id]['Name'].values
                tt=str(Id) + "-" +aa
                attendance.loc[len(attendance)] = [Id,aa,date,timeStamp]
                cv2.rectangle(im, (x, y), (x + w, y + h), (0, 260, 0), 7)
                cv2.putText(im, str(tt), (x + h, y), font, 1, (255, 255, 0,), 4)
            else:
                Id='Unknown'
                tt=str(Id)
                if(conf>75):
                    noOfFile=len(os.listdir("ImagesUnknown"))+1
                    cv2.imwrite("ImagesUnknown\Image"+str(noOfFile)+".jpg", im[y:y+h,x:x+w])
                cv2.rectangle(im, (x, y), (x + w, y + h), (0, 25, 255), 7)
                cv2.putText(im, str(tt), (x + h, y), font, 1, (0, 25, 255), 4)
        attendance=attendance.drop_duplicates(['Id'],keep='first')
        cv2.imshow('im',im)
        if cv2.waitKey(100) & 0xFF == ord('q'):
                        break
        elif fps > 70:
                    break
    ts = time.time()
    date =datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
```

```
        for(x,y,w,h) in faces:
            Id,conf=recognizer.predict(gray[y:y+h, x:x+w])
            fps=fps+1
            if (conf<50):
                ts=time.time()
                date=datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa=df.loc[df['Id'] == Id]['Name'].values
                tt=str(Id) + "-" +aa
                attendance.loc[len(attendance)] = [Id,aa,date,timeStamp]
                cv2.rectangle(im, (x, y), (x + w, y + h), (0, 260, 0), 7)
                cv2.putText(im, str(tt), (x + h, y), font, 1, (255, 255, 0,), 4)
            else:
                Id='Unknown'
                tt=str(Id)
                if(conf>75):
                    noOfFile=len(os.listdir("ImagesUnknown"))+1
                    cv2.imwrite("ImagesUnknown\Image"+str(noOfFile)+".jpg", im[y:y+h,x:x+w])
                cv2.rectangle(im, (x, y), (x + w, y + h), (0, 25, 255), 7)
                cv2.putText(im, str(tt), (x + h, y), font, 1, (0, 25, 255), 4)
        attendance=attendance.drop_duplicates(['Id'],keep='first')
        cv2.imshow('im',im)
        if cv2.waitKey(100) & 0xFF == ord('q'):
                        break
        elif fps > 70:
                    break
    ts = time.time()
    date =datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
    timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
    Hour, Minute ,Second=timeStamp.split(":")

    currentSubject=sub1.get()
    fileName="Attendance/Attendance_" +currentSubject+"_"+date+"_" + Hour+ "_"+ Minute+ "_" + Second+ ".csv"
    attendance=attendance.drop_duplicates(['Id'],keep='first')
    attendance.to_csv(fileName, index=False)
    sub.destroy()
    cam.release()
    cv2.destroyAllWindows()
    res=attendance
    message2.configure(text=res)
```

CHAPTER 10
CONCLUSION, APPLICATIONS AND FUTURE SCOPE

Conclusion

The FDAS can be proven as an efficient system for classroom attendance. By using this system, the chances of fake attendance and proxies can be reduced. There are lots of biometrics systems that can be used for managing attendance, but face recognition has the best performance. So, we need to implement a reliable and efficient attendance system for classroom attendance which can work for multiple face recognition at one time. We found the solution for light intensity problem and head pose problem for which we can use the Illumination Invariant algorithm. Also, to implement this system, no specialized hardware is required. A camera device and a standalone PC with high storage capacity are sufficient for constructing this system.

Applications

1. Institutions

Institutions have the traditional way of marking attendance called out each student name to check if they are present. This method of the roll call is time-consuming and tedious. By using Facial Recognition, the process of taking attendance can be significantly improved to save time and provide a hassle-free way to automatically mark attendance. Since the number of students in an institution is more, using an automated system improves the productivity and standard of the college.

2. Companies

In most companies, employees have the practice of using their biometrics or ID card to log their time of entry and exit. During peak hours the number of people entering and exiting the office is generally high. This causes congestion in the workplace and people queue up to and await their turn. Facial Recognition systems provide a more convenient way of managing this process of attendance. Employees don’t have to worry about logging their time as it’s an automatic process. The system will note the time of entry and exit when an employee enters or exits the office.

3. Prison

In prisons, every day headcount of the prisoners is done to check if all inmates are present. Using facial recognition to automate this process of doing headcount increases the efficiency and reliability is improved. The security also increases as tabs can be kept on each prisoner at all times.

Future Scope

The system can be enhanced in the future by updating internal marks for each of the students along with their attendance marks and the collected data can be hence uploaded in the respective portals. A complete database of the students will be maintained securely. Also, the push messages could be sent to the parents once the attendance of their ward is marked absent. An analysis system can be added to the system to save time.

Limitation

- As it is a webcam-based project It may not be accurate always.
- Some space is required between students otherwise it will not detect faces properly.
- It does not the analysis of the attendance.

REFERENCES

[1] Ajinkya Patil, Mridang Shukla, “Implementation of Class Room Attendance System Based on Face Recognition in Class”, IJAET (International Journal of Advances in Engineering and Technology), Vol. 7, Issue 3, July 2014.

[2] Naveed Khan Baloch, M. Haroon Yousaf, Waqar Ahmad, M. Iran Baig, “Algorithm for Efficient Attendance Management: Face Recognition based Approach”, IJCSI, Vol. 9, Issue 4, July 2012.

[3] Ahonen, T., Hadid, A. and Pietikäinen, M. (2006), Face Description with Local Binary Patterns: Application to Face Recognition. IEEE Trans. Pattern Analysis and Machine Intelligence 28(12):2037-2041.

[4] Hadid, A. and Pietikäinen, M. (2009), Combining Appearance and Motion for Face and Gender Recognition from Videos. Pattern Recognition 42(11):2818-2827.

[5] Li, S.Z., Chu, R.F., Liao, S.C. and Zhang, L.(2007), Illumination Invariant Face Recognition Using Near-Infrared Images. IEEE Trans. Pattern Analysis and Machine Intelligence 29(4):627-639.

[6] Pietikäinen, M., Hadid, A., Zhao, G. and Ahonen, T. (2011), Computer Vision Using Local Binary Patterns, Springer, 207 p.

[7] Zhao, G. and Pietikäinen, M. (2007), Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions. IEEE Trans. Pattern Analysis and Machine Intelligence 29(6:915-928.

[8]. IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 1, July 2012 ISSN (Online): 1694-0814

[9]. Javier Ruiz Del Solar, Rodrigo Verschae, and Mauricio Correa. Face recognition in unconstrained environments: A comparative study. In ECCV Workshop on Faces in RealLife Images: Detection, Alignment, and Recognition, Marseille, France, October 2008.

[10]. Kyungnam Kim “Face Recognition using Principle Component Analysis”, Department of Computer Science, University of Maryland, College Park, MD 20742, USA.

[11]. Osuna, E., Freund, R. and Girosit, F. (1997). "Training support vector machines: an application to face detection." 130-136

[12]. W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld,“Face recognition: A literature survey,” ACM Computing Surveys, 2003, vol. 35, no. 4, pp. 399-458.

[13]. Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. Computer Vision and Image Understanding (CVIU), 110(3):346–359.

[14]. H.K.Ekenel and R.Stiefelhagen,Analysis of local appearance based face recognition: Effe cts of feature selection and feature normalization. In CVPR Biometrics Workshop, New York, USA, 2006

[15] T.Kanade, “Picture processing by computer complex and recognition of human faces,” technical report, Dept. Information Science, Kyoto Univ., 1973.

[16] V. Blanz and T. Vetter, “Face recognition based on fitting a 3D morphable model,” IEEE Trans. On Pattern Analysis and Machine Intelligence, vol. 25, no. 9, September 2003

BIBLIOGRAPHY

https://lpuin-my.sharepoint.com/:u:/g/personal/vishal_11615139_lpu_in/ESZ0oMNjXq1Jlaq2yCUAlt0B R7YNkLoPyOzfSfHtnK23Cw?e=jR6fIk

https://www.instructables.com/id/Real-time-Face-Recognition-an-End-to-end-Project/

https://pdfs.semanticscholar.org/b690/8248f52c917c8202e5bb1d39a19c0e018813.pdf

https://towardsdatascience.com/face-recognition-for-beginners-a7a9bd5eb5c2

https://www.tutorialspoint.com/

https://www.pyimagesearch.com/2018/06/11/how-to-build-a-custom-face-recognition- dataset/

https://medium.com/dataseries/face-recognition-with-opencv-haar-cascade-a289b6ff042a

https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b

https://www.ilovepdf.com/download/k2cls85mn39jn81ycsp1q3bdlA9k1s177xc0cykxjy407m8fz102dp7x10l5zb077t17gklst7yd604dvp1h4fr1vxs2jpzf5nvblnAxrgsc24b9bvyhssb3yyt5wbd04nd645h8zpnnqqqgxl7lws3dz1dcqcwvw1fq8xt4y53lzcls0c61/28o

https://www.ijitee.org/wp-content/uploads/papers/v8i6/F3965048619.pdf