**ATTENDANCE ENTRY USING FACE RECOGNITION**

by

Chava Sai Teja - 13BCE1032

Koutika Gourav - 13BCE1068

*A project report submitted to the*

**SCHOOL OF COMPUTING SCIENCES AND ENGINEERING**

*In partial fulfillment of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

*in*

**COMPUTER SCIENCE ENGINEERING**



**VIT UNIVERSITY, CHENNAI**

**Vandalur – Kelambakkam Road**

**Chennai – 600 127.**

**MAY 2017**



**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**DECLARATION**

We hereby declare that the project entitled “**Attendance Entry Using Face Recognition**” submitted by us to the School of Computing Science and Engineering, VIT University, Chennai Campus, Chennai in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out by us under the supervision of **Prof. Sivagami M**. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or university.

**Chava Sai Teja**

**(13BCE1032)**

**K.Gourav**

**(13BCE1068)**



**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING**

**CERTIFICATE**

The project report entitled “**Attendance Entry Using Face Recognition**” is prepared and submitted by **Chava Sai Teja (13BCE1032), K.Gourav (13BCE1068).**It has been found satisfactory in terms of scope, quality and presentation as partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** in VIT University, Chennai Campus, Chennai, India

**Prof. Sivagami M**

**Examined by:**

**Examiner 1 Examiner 2**

**ACKNOWLEDGEMENT**

We would like to express our deepest appreciation and gratitude to all those who provided us the possibility to complete this report. We give our special gratitude to our faculty guide, **Prof. Sivagami M**, whose contribution in stimulating suggestions and encouragement, helped us to coordinate our project and writing this report.

Furthermore we would also like to acknowledge with much appreciation the crucial role of our **Program Chair Dr. Ganesan R**, whose encouraging words enabled us to complete this project successfully. We would also like to thank **Dean of the Department, SCSE, Dr. VAIDEHI VIJAYAKUMAR** whose constant word of encouragement helped us immensely during the course of this Project. Last but not least, we take this opportunity to thank **Prof. Renta Chintala Bhargavi**, whose timely mails, and proper communication, facilitated smooth operation and completion of project in stipulated time.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Title** | **Page** |
|  | Cover Page | `1 |
|  | Declaration | 2 |
|  | Certificate | 3 |
|  | Acknowledgement | 4 |
|  | Table of Contents | 5 |
|  | List of Tables | 7 |
|  | List of Figures | 8 |
|  | List of Abrrevations | 9 |
|  | Abstract | 10 |
| 1 | Introduction | 11 |
|  | 1.1 Objective | 11 |
|  | 1.2 Background | 11 |
|  | 1.3 Motivation | 12 |
| 2 | Project Description and Goals | 13 |
|  | 2.1 Project Description | 13 |
|  | 2.2 Goals | 14 |
| 3 | Technical Specifications | 15 |
|  | 3.1 Hardware Specification | 15 |
|  | 3.2 Software Specification | 16 |
|  | 3.3 Data Flow | 17 |
| 4 | Engineering Design | 18 |
|  | 4.1 Design Approach | 18 |
|  | 4.1.1 Hardware Design | 18 |
|  | 4.1.2 Android Application |  |
|  | 4.2 Codes and Standards | 19 |
|  | 4.2.1 IEEE 802.11 | 19 |
|  | 4.3 Constraints, Alternatives and Trade offs | 19 |
|  | 4.3.1 Realistic Constraints | 19 |
|  | 4.3.2 Design Constraints | 19 |
|  | 4.3.3 Component Alternatives | 20 |
|  | 4.4 Trade off | 21 |
| 5 | Schedules, Tasks and Milestones | 22 |
|  | 5.1 Task Schedule | 22 |
| 6 | Project Demonstration | 26 |
|  | 6.1 Hardware Device | 26 |
|  | 6.2 Implementation of Method 1 | 26 |
|  | 6.3 Implementation of Method 2 | 29 |
|  | 6.4 Uploading the names into database | 31 |
| 7 | Cost Analysis | 34 |
| 8 | Conclusion | 35 |
| 9 | References | 36 |
|  | Appendix A - Source Code | 37 |

**List of Tables**

**Table No. Title Page**

4.1 Comparison of Alternatives of Raspberry pi 20

4.2 Comparison of Alternatives of USB Camera 21

5.1 Schedule 25

7.1 Component and Total Cost of the Project 34

**List of Figures**

**Figure No. Title Page**

2.1 Block Diagram of the Project 14

3.1 Data Flow of the Project 17

4.1 Overall Engineering Design 18

4.2 Figure of the Hardware 19

6.1 Figure of the prototype 26

6.2(a) Figure of Directory of training images 27

6.2(b) Figure of Directory of aligned images 27

6.2(c) Figure of CSV file 28

6.2(d) Figure of Directory of Embedding’s 28

6.2(e) Figure of Output 29

6.3(a) Figure of Directory of training images 29

6.3(b) Figure of YML file 30

6.3(c) Figure of Output 30

6.4(a) Figure of Home page in browser 31

6.4(b) Figure of Database 32

6.4(c) Figure of Android app 32

6.4(d) Figure of Android app 32

6.4(e) Figure of Android app 33

**List of Abbreviations**

**Abbreviation Expansion**

USB Universal Synchronous Bus

SD card Secure Digital Card

VGA Video Graphics Array

HDMI High Definition Multimedia Interface

LBPH Local Binary Patterns Histograms

SOC System On Chip

GPIO General Purpose Input Output

SQL Structured Query Language

HTML Hyper Text Markup Language

CSS Cascading Style Sheets

PHP Personal Home Page

CSV Comma Separated Values

**ABSTRACT**

Human face recognition has been widely used to be an important branch of biometrics verifications in many applications such as door control, video monitoring system, networks security and also human computer interactions. This project deals with automatic posting of student's attendance to the database. Present attendance systems are based on finger prints or manually noted down by a human. This consumes time based on the strength and also there are possibilities of proxies. The duration of a class is only 50min. So significant amount of time is lost in taking attendance by faculty. So the idea is to automate the posting of attendance system using face recognition technique with minimal hardware requirements. The proposed project uses raspberry pi and raspberry pi camera. Attendance entry can be made into smarter way by using face recognition technique.The camera is connected to raspberry pi and is placed in top right or top left corner of the room. The camera captures the image and pre-processing and face recognition is done in is done in raspberry pi and attendance is uploaded in database. The advantage that comes with attendance system using face recognition is avoiding proxies, real time proof for cross verifying attendance at any point of time and conserving time for faculty members from traditional attendance entry systems.

The project is expected to result in the development of a prototype device with capability of recognizing faces in the image. These recognized faces can be used to update attendance in the database by marking them as present and all the others as absent.

The desired objectives for the project are summarized as below:-

1. Todetect the faces in the captured image.

2. To train the classifier so that it recognizes faces in the image.

3. To recognize the faces in the captured image using the classifier.

**CHAPTER 1**

**INTRODUCTION**

* 1. **Objective**

The primary objective of this system is to present an automated system for human face recognition for institute to mark the attendance of their students. The system will record the attendance of the students in class room environment. The above system is fully automated and easily deployable.

* 1. **Background**

Face recognition is an easy task for humans. Experiments have shown, that even one to three day old babies are able to distinguish between known faces. So how hard could it be for a computer? It turns out we know little about human recognition to date. Are inner features (eyes, nose, mouth) or outer features (head shape, hairline) used for a successful face recognition? How do we analyze an image and how does the brain encode it? It was shown by [David Hubel](http://en.wikipedia.org/wiki/David_H._Hubel) and [Torsten Wiesel](http://en.wikipedia.org/wiki/Torsten_Wiesel), that our brain has specialized nerve cells responding to specific local features of a scene, such as lines, edges, angles or movement. Since we don’t see the world as scattered pieces, our visual cortex must somehow combine the different sources of information into useful patterns. Automatic face recognition is all about extracting those meaningful features from an image, putting them into a useful representation and performing some kind of classification on them.

Face recognition based on the geometric features of a face is probably the most intuitive approach to face recognition. One of the first automated face recognition systems was described in [[Kanade73]](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#kanade73): marker points (position of eyes, ears, nose, ...) were used to build a feature vector (distance between the points, angle between them, ...). The recognition was performed by calculating the euclidean distance between feature vectors of a probe and reference image. Such a method is robust against changes in illumination by its nature, but has a huge drawback: the accurate registration of the marker points is complicated, even with state of the art algorithms. Some of the latest work on geometric face recognition was carried out in [[Bru92]](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#bru92). A 22- dimensional feature vector was used and experiments on large datasets have shown, that geometrical features alone my not carry enough information for face recognition.

The Eigenfaces method described in [[TP91]](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#tp91) took a holistic approach to face recognition: A facial image is a point from a high-dimensional image space and a lower-dimensional representation is found, where classification becomes easy. The lower-dimensional subspace is found with Principal Component Analysis, which identifies the axes with maximum variance. While this kind of transformation is optimal from a reconstruction standpoint, it doesn’t take any class labels into account. Imagine a situation where the variance is generated from external sources, let it be light. The axes with maximum variance do not necessarily contain any discriminative information at all, hence a classification becomes impossible. So a class-specific projection with a Linear Discriminant Analysis was applied to face recognition in [[BHK97]](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#bhk97). The basic idea is to minimize the variance within a class, while maximizing the variance between the classes at the same time.

Recently various methods for a local feature extraction emerged. To avoid the high-dimensionality of the input data only local regions of an image are described, the extracted features are (hopefully) more robust against partial occlusion, illumation and small sample size. Algorithms used for a local feature extraction are Gabor Wavelets ([[Wiskott97]](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#wiskott97)), Discrete Cosinus Transform ([[Messer06]](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#messer06)) and Local Binary Patterns ([[AHP04]](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#ahp04)). It’s still an open research problem what’s the best way to preserve spatial information when applying a local feature extraction, because spatial information is potentially useful information.

* 1. **Motivation**

Taking attendance manually by traditional methods like checking and marking if each individual is present or not takes lot of time. This significant amount of time can be used for other purposes like taking class for the whole duration without allotting any time for attendance or can be used for refreshment activities.

Generally mistakes do happen while noting attendance. These mistakes can happen from both the sides(Teacher and Student) such as faculty marking attendance as absent for student who is present or students giving proxies for their friends. In the former case the student may later claim that he was present and faculty might not be in a situation to update the attendance because it was blocked by that time or faculty might not trust that student who was present on that day. The later case may encourage students to bunk classes which affects their academics.

These all problems motivated me to come up with a system which is robust and dynamic that eases the traditional processes of noting attendance.

**CHAPTER 2**

**PROJECT DESCRIPTION AND GOALS**

**2.1 Description**

This project proposes a prototype for automatically posting attendance to the database. The camera connected to the raspberry pi is placed in top left or top right or in the center based on the dimensions and elevation of class room. Camera captures the image and it is used for further processing by the raspberry pi. Face recognition is done through the following steps:

1. First, look at a picture and find all the faces in it
2. Second, focus on each face and be able to understand that even if a face is turned in a weird direction or in bad lighting, it is still the same person.
3. Third, be able to pick out unique features of the face that you can use to tell it apart from other people— like how big the eyes are, how long the face is, etc.
4. Finally, compare the unique features of that face to all the people you already know to determine the person’s name.

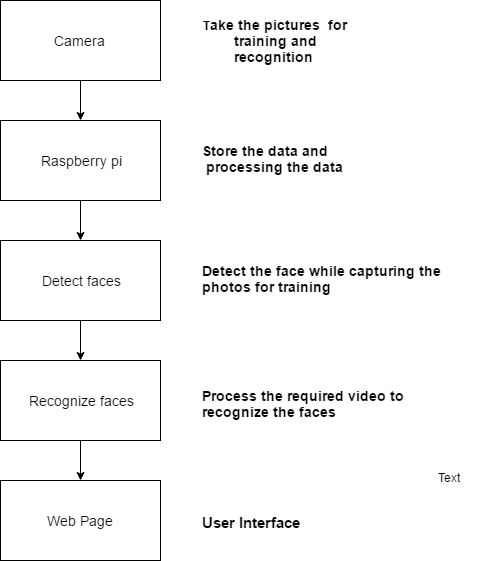


Fig. 2.1: Block Diagram of the Project

**2.2Goals**

To develop an automatic attendance posting system which is highly accurate and does its function within no time. The system should work in any kind of environments irrespective of constraints posed by illumination, age, emotion, pose and surroundings.

**CHAPTER 3**

**TECHNICAL SPECIFICATIONS**

**3.1 Hardware Specifications**

**3.1.1Raspberry pi**

* Soc: Broadcom BCM2836 (CPU,GPU, DSP, SDRAM)
* CPU: 900MHz quad-core ARM Cortex A7 (Armv7 instruction set)
* GPU: Broadcom Videocore IV @ 250 MHz
* More GPU info: OpenGL ES 2.0 (24GFLPS); 1080p30 MPEG-2 and VC-1decoder(with license); 1080p30 h.264/MPEG-4 AVC high-prfile decoder and encoder
* Memory: 1GB (shared with GPU)
* USB ports: 4
* Video input: 15-pin MIPI camera interface (CSI) connector
* Video outputs: HDMI, composite video (PAL and NTSC) via 3.5 mm jack
* Audio input: I2S
* Audio outputs: Analog via 3.5 mm jack; digital via HDMI and I2S
* Storage: Micro SD
* Network: 10/100Mbps Ethernet
* Peripherals: 17 GPIO plus specific functions, and HAT ID bus
* Power rating: 800 mA (4.0 W)
* Power source: 5V via MicroUSB or GPIO header
* Size: 85.60mm x 56.5 mm
* Weight: 45g

**3.1.2 Quantum PC Web Camera**

* Supported up to 25.0 mega pixels interpolated, the frame rate upto 30fps
* Excellent quality and fashionable style
* High-quality CMOS sensor
* Clear and sharp still picture and motion video capturing
* Ideally designed to work well with both laptops and desktop computers.
* Support Windows 8, 7, Windows XP, Vista
* Image Resolution 14 Mega Pixels (Interpolated) 6 white lights
* USB 2.0 Interface
* Inbuilt sensitive microphone.
* Image Sensor High Quality CMOS Sensor
* Potentiometer to switch on 6 lights when in dark
* Image Control Color saturation, brightness, sharpness and brightness is adjustable
* Snap shot switch for taking stil pictures
* Anti-flicker 50Hz,60Hz or outdoor
* Resolution Hardware: 500K pixels
* Image Quality RGB24 or I420
* Exposure Auto or manual
* Angle of View 58 o
* Interface USB2.0
* Frame Rate  30 fps (MAX)
* Lens
* f=6.0 F=2.0
* Focus Range 4cm to infinity

**3.2 Software Specifications**

1. Python with

* 1. Numpy - numpy is the fundamental package for scientific computing with Python.
  2. os - This module will be used to maneuver with image and directory names. First, we will use this module to extract the image names in the database directory and then from these names we will extract the individual number, which will be used as a label for the face in that image.
  3. sys - This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter.
  4. cv2 - **OpenCV** (*Open Source Computer Vision*) is a [library of programming functions](https://en.wikipedia.org/wiki/Library_(computing)) mainly aimed at real-time [computer vision](https://en.wikipedia.org/wiki/Computer_vision). The library is cross-platform and free for use under the [open-source](https://en.wikipedia.org/wiki/Open-source) [BSD license](https://en.wikipedia.org/wiki/BSD_license).
  5. MySQL dB - MySQL dB is a thread-compatible interface to the popular MySQL database server that provides the Python database API.

2. MySQL

MySQL is an [open-source](https://en.wikipedia.org/wiki/Open-source) [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS).

3. Sublime Text Editor

Sublime Text is a sophisticated text editor for code, markup and prose.

4. Apache Tomcat Server

Apache Tomcat, often referred to as Tomcat Server, is an open-source [Java Servlet Container](https://en.wikipedia.org/wiki/Servlet_container) developed by the [Apache Software Foundation](https://en.wikipedia.org/wiki/Apache_Software_Foundation) (ASF). Tomcat implements several [Java EE](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition) specifications including [Java Servlet](https://en.wikipedia.org/wiki/Java_Servlet), [Java Server Pages](https://en.wikipedia.org/wiki/JavaServer_Pages) (JSP), [Java EL](https://en.wikipedia.org/wiki/Unified_Expression_Language), and [Web Socket](https://en.wikipedia.org/wiki/WebSocket), and provides a "pure [Java](https://en.wikipedia.org/wiki/Java_(programming_language))" [HTTP](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) [web server](https://en.wikipedia.org/wiki/Web_server) environment in which [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) code can run.

5.PhpMyAdmin

PhpMyAdmin is a [free and open source](https://en.wikipedia.org/wiki/Free_and_open_source) tool written in [PHP](https://en.wikipedia.org/wiki/PHP) intended to handle the administration of [MySQL](https://en.wikipedia.org/wiki/MySQL) or [Maria DB](https://en.wikipedia.org/wiki/MariaDB) with the use of a browser. It can perform various tasks such as creating, modifying, deleting databases, tables, fields or rows, executing SQL statements, or managing users and permissions

**3.3 Data Flow**

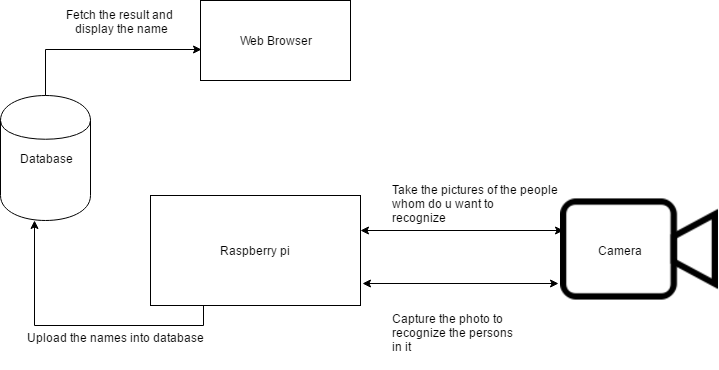


Fig. 3.1: Data Flow of the Project

**CHAPTER 4**

**DESIGN, APPROACH AND DETAILS**

**4.1 Design Approach**

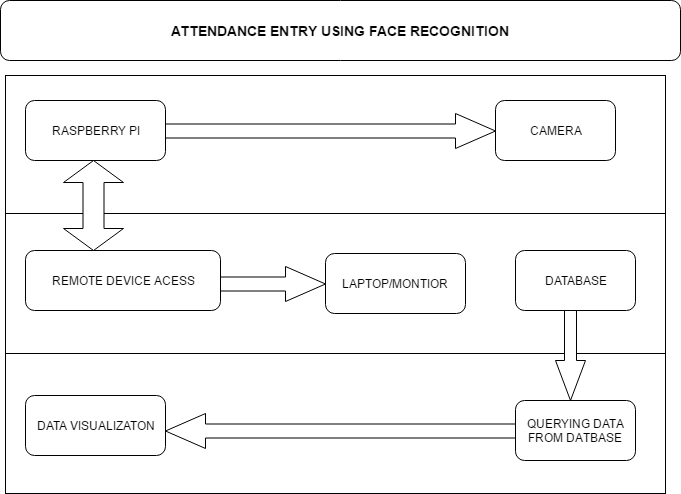
****

Fig.4.1: Overall Engineering Design

**4.1.1 Hardware Design**

Raspberry pi is connected to monitor via HDMI cable. The USB camera which is connected to the raspberry captures the images/video which can be seen over monitor. Face Recognition is done and attendance is updated to database. Attendance can also be seen over app.

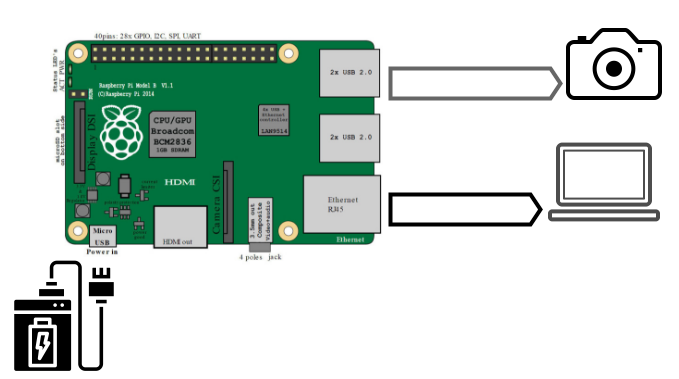
****

Fig. 4.2: Figure of the Hardware

**4.2Codes and Standards**

**4.2.1 IEEE 802.11**

**IEEE 802.3** is a [working group](https://en.wikipedia.org/wiki/Working_group) and a collection of [Institute of Electrical and Electronics Engineers](https://en.wikipedia.org/wiki/Institute_of_Electrical_and_Electronics_Engineers) (IEEE) standards produced by the working group defining the [physical layer](https://en.wikipedia.org/wiki/Physical_layer) and [data link layer](https://en.wikipedia.org/wiki/Data_link_layer)'s [media access control](https://en.wikipedia.org/wiki/Media_access_control) (MAC) of wired [Ethernet](https://en.wikipedia.org/wiki/Ethernet). This is generally a [local area network](https://en.wikipedia.org/wiki/Local_area_network) (LAN) technology with some [wide area network](https://en.wikipedia.org/wiki/Wide_area_network) (WAN) applications. Physical connections are made between nodes and/or infrastructure devices ([hubs](https://en.wikipedia.org/wiki/Network_hub), [switches](https://en.wikipedia.org/wiki/Network_switch), [routers](https://en.wikipedia.org/wiki/Router_(computing))) by various types of copper or [fiber cable](https://en.wikipedia.org/wiki/Optical_fiber).

802.3 is a technology that supports the [IEEE 802.1](https://en.wikipedia.org/wiki/IEEE_802.1) network architecture.

802.3 also defines LAN access method using [CSMA/CD](https://en.wikipedia.org/wiki/CSMA/CD).

**4.3 Constraints, Alternatives and Tradeoffs**

**4.3.1 Realistic Constraints**

* The Pose
* Illumination - This is a huge problem in image processing as it can effect lot of pixel values in image
* Emotions
* Age - With age there comes a lot of changes in face especially during teen age.
* Make up

**4.3.2 Design Constraint**

1. Due to cost constraints, the camera used is not expensive and therefore cannot provide clear picture.
2. Due to cost constraints, the raspberry pi used is not the latest version and therefore fast processing of image cannot be done.

**4.3.3 Component Alternatives**

**4.3.3.1 Raspberry pi**

|  |  |  |
| --- | --- | --- |
| Component | Advantages | Disadvantages |
| Beagle Bones | Faster Processor with large memory  Uses Linux Distribution  66 Digital I/O along with 7 Analog I/O ports | Much higher cost compared to raspberry pi |
| Raspberry pi 3 | 50 percent faster than pi 2. Has in-built Wi-Fi and Bluetooth | Higher cost when compared to raspberry pi 2. |
| Huawei’s HiKey 960 | Faster processor(Octa core)  High Ram - 3GB  High on board Storage - 32GB | Very high cost when compared to Raspberry pi 2, Raspberry pi 3 or Beagle bone. |

Table 4.1: Comparison of Alternatives of Raspberry pi

Among the available alternatives raspberry pi 3 would be the best choice taking cost into consideration. We don’t need processor that is fast enough to compute in 1-2 sec, similarly very slow processor would not be suitable. So Raspberry pi 3 seems to be the best suitable alternative available for this project.

**4.3.3.2 USB Camera**

|  |  |  |
| --- | --- | --- |
| Component | Advantages | Disadvantages |
| Raspberry pi Camera | The Rapsi camera connects directly to the GPU, and is capable of 1080p30 video encode, 5MP stills in pretty decent quality. Because its attached to the GPU, there is only a little impact on the CPU, leaving it available for other processig unlike webcams which use a lot more CPU. | Much higher cost compared to USB Camera |
| Other high end USB camera | Better quality picture for processing as well as high FPS. | Much higher cost when compared to both USB camera as well as raspberry pi camera. |

Table 4.2: Comparison of Alternatives of USB Camera

**4.3.3.3 Ethernet cable**

The alternative for Ethernet cable is use of Wi-Fi module. Advantages of using Wi-Fi module is that you can easily connect to all nearby Wi-Fi by configuring it whereas Ethernet cable involves wire. The disadvantage is that it is not fast when compared to Ethernet cable.

**4..4 Trade Off**

Camera :- Camera used is not of high quality due to cost constraints. For better accuracy, mutiple cameras can be placed at different locations in the class room.

**CHAPTER 5**

**SCHEDULE, TASKS AND MILESTONES**

**5.1 Task Schedule**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tasks | Starting Date | Completion  Date | Assignee | Description | Number of hours for the task(per day) |
| Project  Brainstorming | 2nd  November  2016 | 7th  November  2016 | Entire  team | No proper insight just a brief description | 5 |
| Report submission for review 0 | 7th  November  2016 | 8th  November  2016 | Entire  Team | Drafting report | 4 |
| Literature survey partial | 10th  November  2016 | 7th  January  2017 | Entire  Team | Searching  for papers | 3 |
| Literature  survey complete | 8th  January  2017 | 30th  January  2017 | Entire  Team | Identified  the  different implementation  for the project | 4 |
| Assessment of components based on the cost. | 31st  January  2017 | 10th  February  2017 | Entire  team | Identifying  different  components for the project | 3 |
| Final Selection of Component based on cost and requirements of the project | 11th  February  2017 | 14th  February  2017 | Entire  team | Finalizing the components  and ordering them. | 3 |
| Ordering the  Components | 14th  February  2017 | 17th  February  2017 | Entire  Team | Components ordered | 4 |
| Studying methods | 17th  February  2017 | 19th  February  2017 | Entire Team | Understanding different methods and their assumptions and effects on data. | 4 |
| Applying face detection technique | 17th  February  2017 | 22nd  February  2017 | Entire  Team | Applying viola jones |  |
| Report Submission for review 1 | 21st  February | 23rd  February | Entire team | Report Submission | 4 |
| Overcoming problems faced with voila jones | 24th February  2017 | 28th February  2017 | Entire team | Finding out and understanding other different methods to overcome the problems faced while using Voila jones | 4 |
| HOG Implementation | 1st March 2017 | 5th March 2017 | Entire Team | Implementing HOG for detection of faces | 5 |
| Overcoming the problem of faces in weird direction | 6th March 2017 | 10th March 2017 | Entire Team | Studying methods of how to tackle the problem when faces are in weird direction | 4 |
| Applying affine transformations | 11th March | 15th March | Entire Team | Applying rotation, scaling so that eyes and mouth are always in same position in image | 4 |
| Creating data set | 16th March | 18th March | Entire Team | Obtaining 20 images for each person to be recognized for training | 4 |
| Training | 19th March | 25th March | Entire Team | Training the classifier | 5 |
| Installing libraries | 26th March | 4th April | Entire Team | Dlib, OpenCV, scipy, pandas, os, sys | 5 |
| Report Submission for review 2 | 5th April | 7th April | Entire Team | Report Submission | 4 |
| Face recognition in raspberry pi | 8th April | 15th April | Entire Team | Finding different efficient methods of face recognition in raspberry pi | 5 |
| Data base integration | 15th April | 21st April | Entire Team | Connecting to the database and uploading the result | 4 |
| Final Report Draft | 22nd April | 27th April | Entire Team | Drafting report and corrections | 4 |

Table 5.1: Project Schedule

**CHAPTER 6**

**PROJECT DEMONSTRATION**

**6.1 Hardware Device**

The hardware setup comprises of raspberry pi, camera, SD card, Ethernet cable, vga to hdmi converter, power supply adapter. Ethernet cable is responsible for sharing the internet to the raspberry pi. Camera is required to capture the photos for training and recognition of faces.SD card is required so that raspberry pi os i.e. raspbian Jessie can be loaded into it. Vga to Hdmi converter is required to connect raspberry pi to the monitor so that we can get display of raspberry pi.



Fig. 6.1: Figure of the prototype

**6.2 Implementation of Method 1**

The first step is to capture the photos of the people for the training and recognition part. For this we need to create a folder of training images and sub folder, by naming the subfolder's name same asthe respective person's name and store the photos in the correct directory.

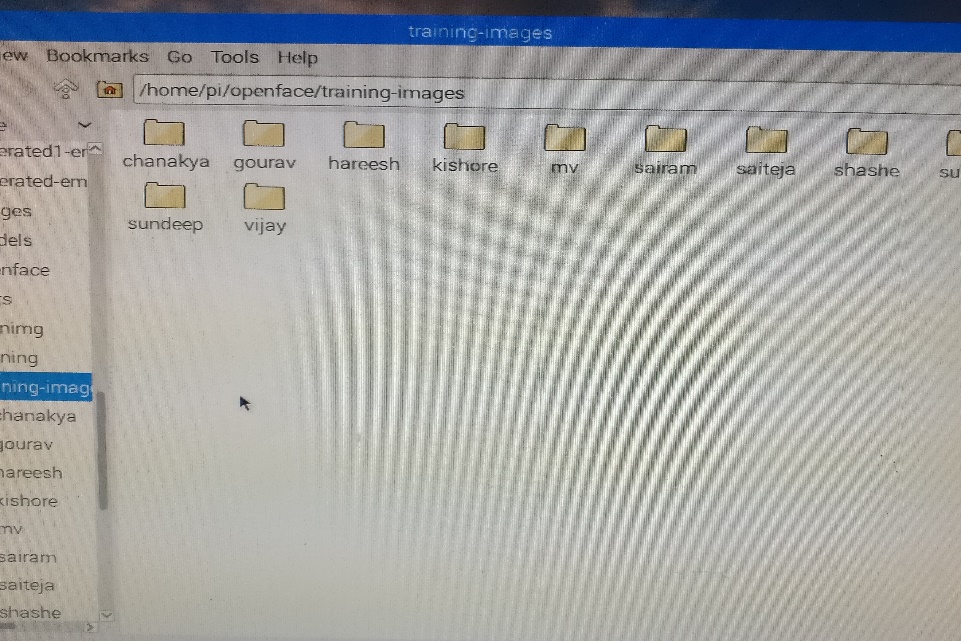


Fig. 6.2(a): Figure of Directory of training images

The second step is to align the faces. We have to deal with the problem that faces turned in different directions look totally different to a computer. To account for this, we will try to wrap each picture so that the eyes and lips are always in the sample place in the image. So in this we pass the directory of the training images so that it outputs a directory of aligned images.

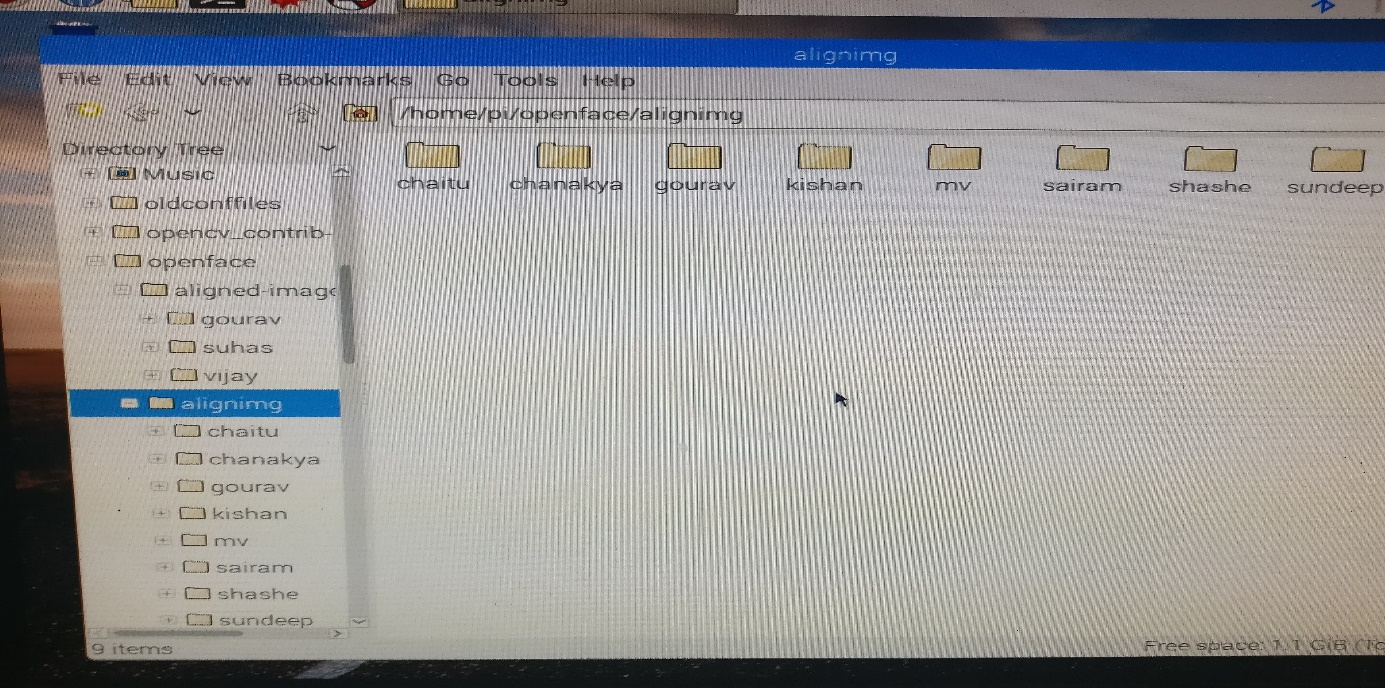


Fig. 6.2(b): Figure of Directory of aligned images

The simplest approach to face recognition is to directly compare the unknown face we found in Step 2 with all the pictures of the people that have already been stored. But this is a hectic process and it takes lot of time. The most reliable way is to measure the faces. Here we create with the help of a model embedding’s for the faces with the help of a model and we store them.

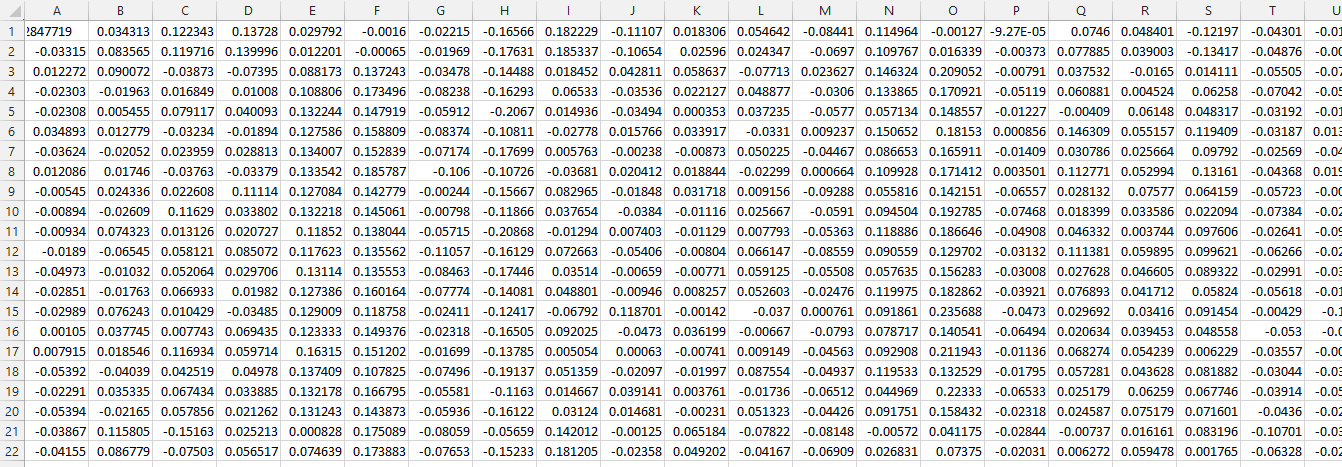


Fig. 6.2(c): Figure of CSV file

Then we need to train so that it can identify the faces. This file has the svm so that we can recognize the faces.

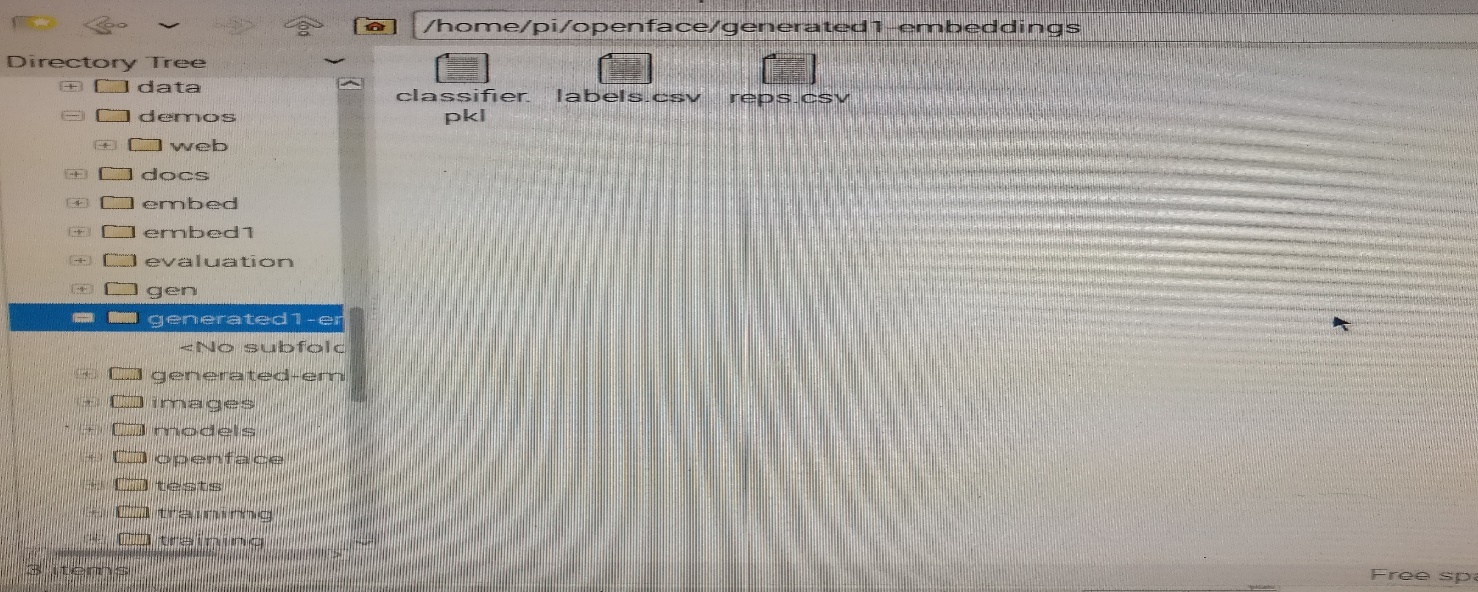


Fig. 6.2(d): Figure of Directory of Embedding’s

The last step in face recognition is to provide an image so that it can recognize the faces in it

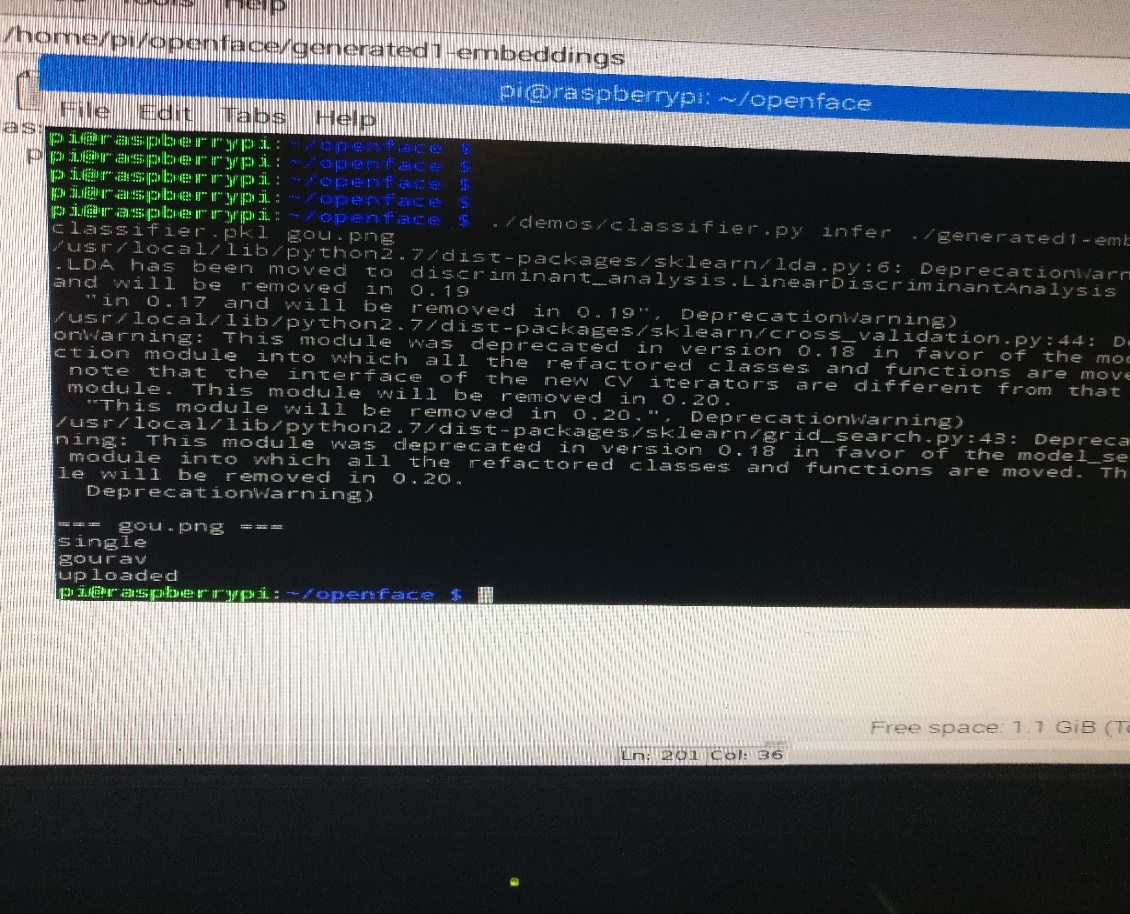


Fig. 6.2(e): Figure of Output

**6.3 Implementation of Method2**

The first step is to capture the photos of the people for the training and recognition part .In this we need to create a folder of training images named as dataset and we store the whole images are stored in that directory with respective of their user id

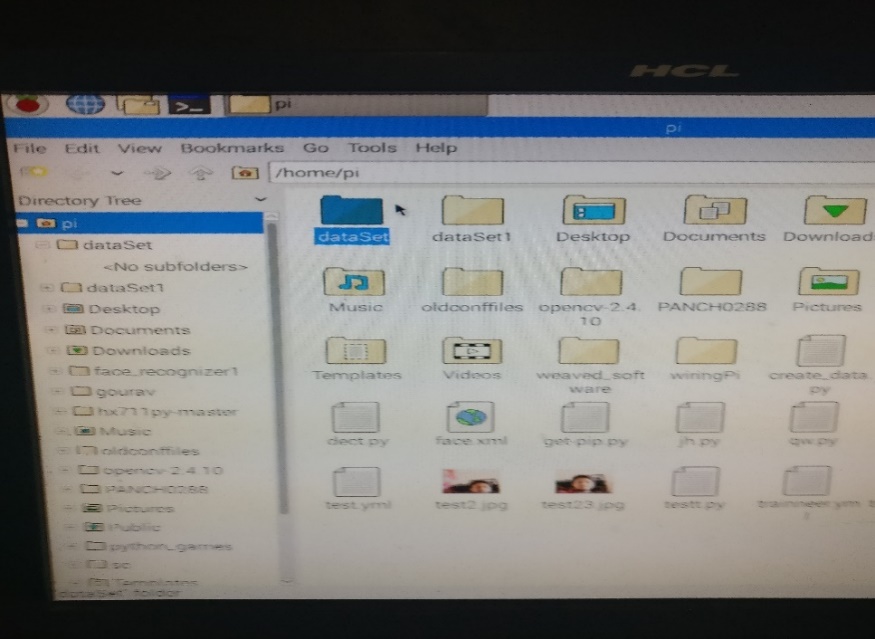


Fig. 6.3(a): Figure of Directory of training images

The second step is to train the images are passed as input it generates an yml file. For the training and recognition part LBPH face recognizer is used.LBPH analyzes each face in the training set separately and independently. LBPH is used to extract features from test face images and Manhattan distance to retrieve the correct match from the system's face database.

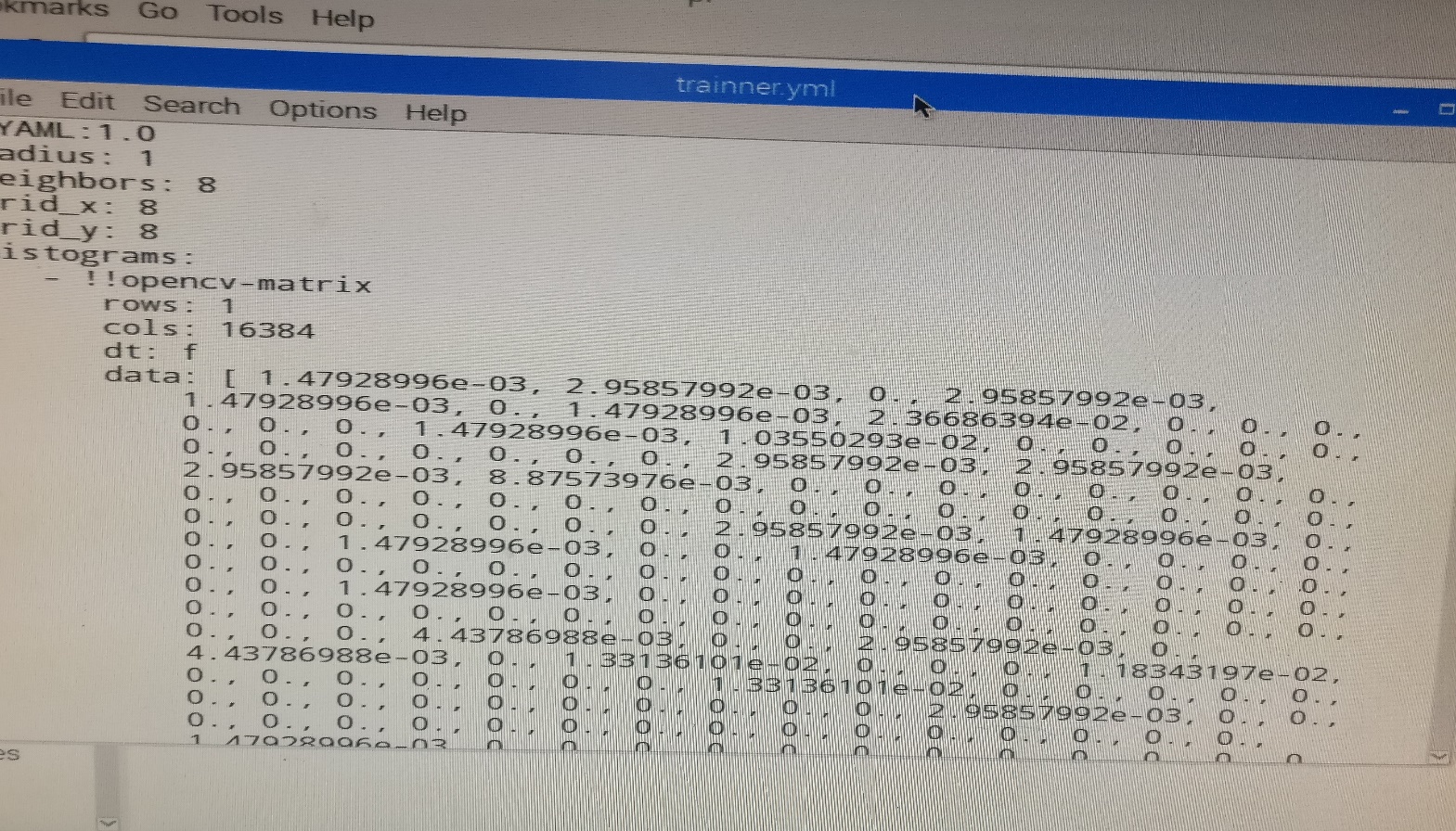


Fig. 6.3(b): Figure of YML file

The final step is to recognize the faces from the live streaming of video by camera

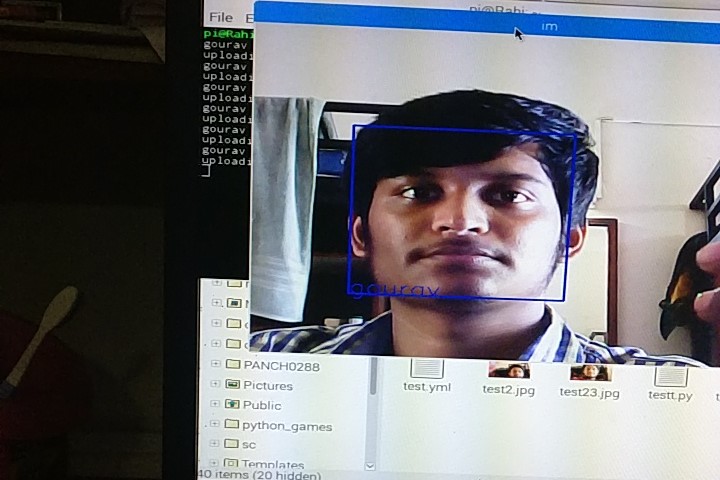


Fig. 6.3(c): Figure of Output

The algorithm 1 was not appropriate due to model which was used. The accuracy of output was very less compared to the algorithm 2

6.4 **Uploading the names into database**

The recognized names are uploaded into the MySQL database. With the help of the web application which was written in HTML and CSS or the android application the user can check the check the attendance of their class by providing the date and classidas input.

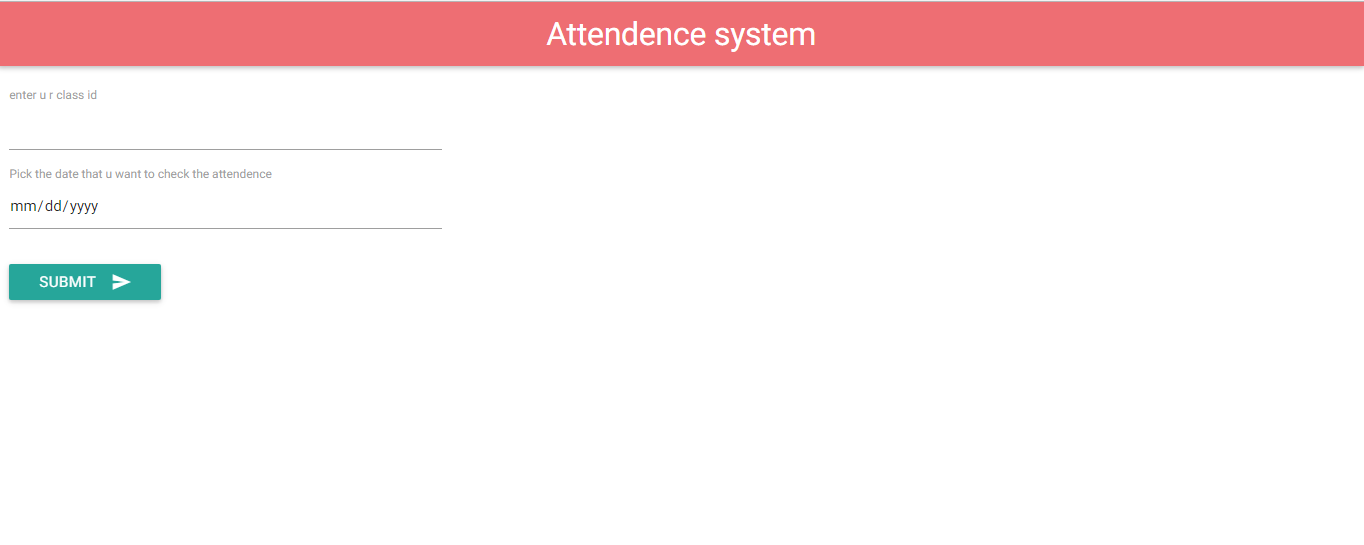


Fig. 6.4(a): Figure of Home page in browser

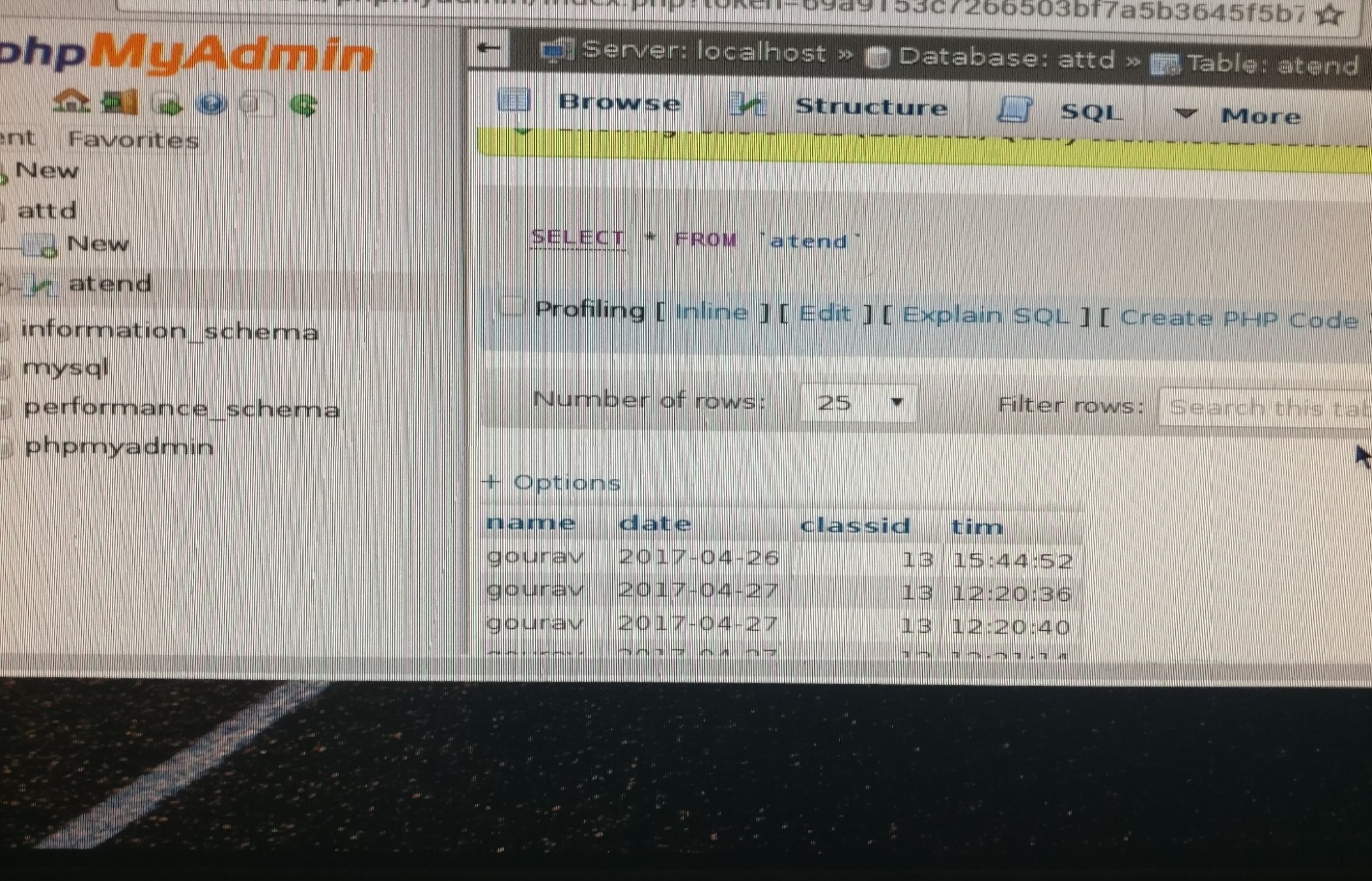


Fig. 6.4(b): Figure of Database

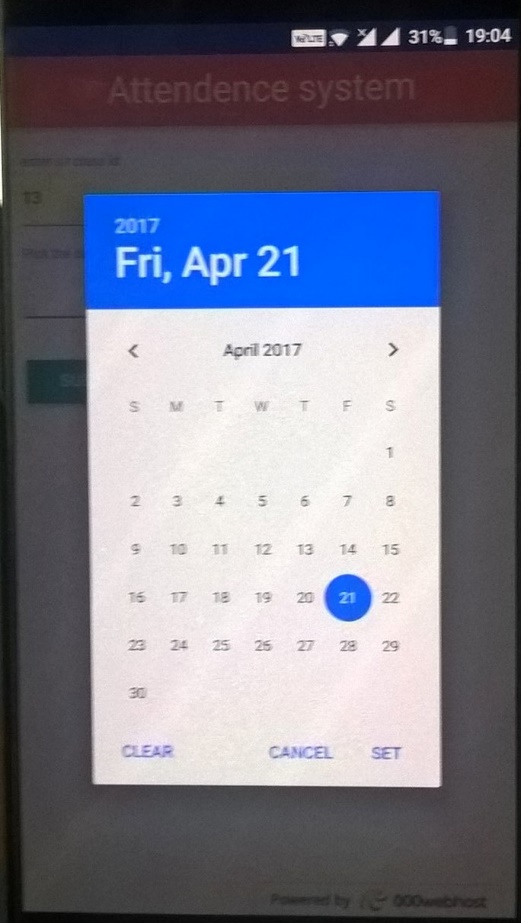
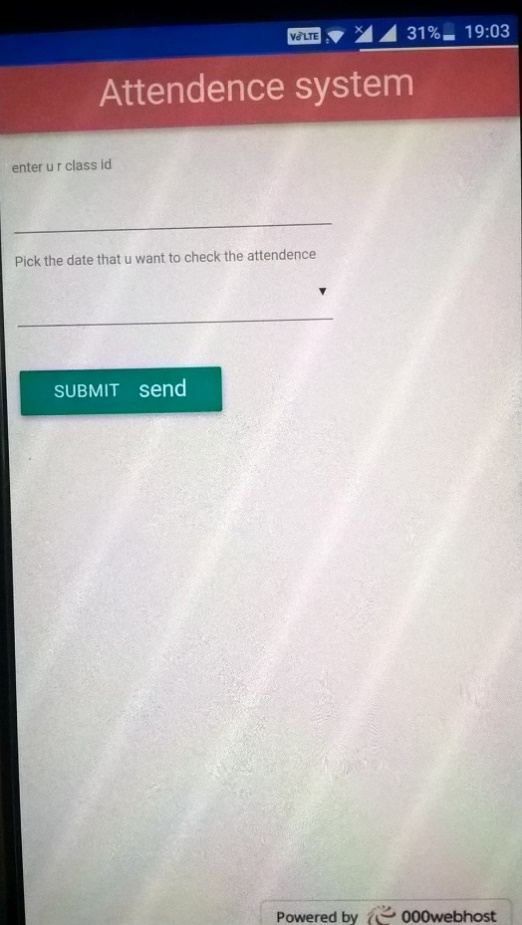


Fig. 6.4(c): Figure of Android app Fig. 6.4(d) Figure of Android app

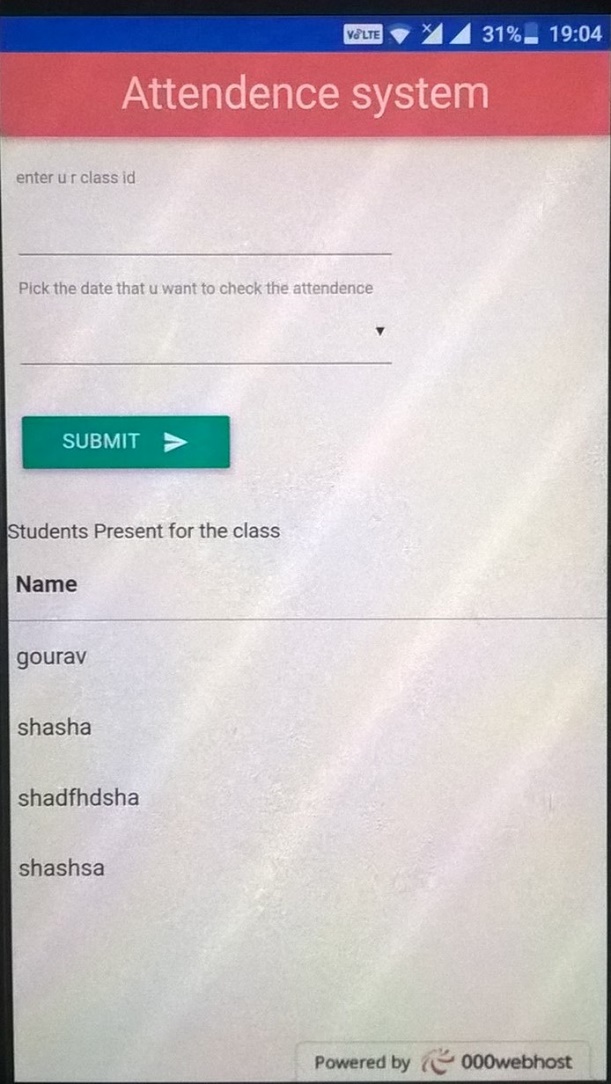


Fig. 6.4(e) Figure of Android app

**CHAPTER 7**

**COST ANALYSIS**

One of the objectives of this project is to design it with minimal hardware requirements as well as at a low cost. The following table clearly states the cost of each component and also the total cost.

|  |  |
| --- | --- |
| **Component** | **Cost** |
| SD Card | Rs 400 |
|  |  |
| Raspberry pi 2 | Rs. 2100 |
|  |  |
| Camera | Rs. 1000 |
|  |  |
| Ethernet cable | Rs. 100 |
|  |  |
| **Total:** | **Rs. 3600** |
|  |  |

Table 7.1: Component and Total Cost of the Project

**CHAPTER 8**

**CONCLUSION**

**8.1 Project Summary**

The aim of the project was to design a device which could automatically post the attendance of the students in the database. This was achieved by detecting the faces using HOG from the image captured by USB camera connected to raspberry pi followed by applying the transformations to the detected faces so that eyes and mouth are placed at same position in all the faces detected earlier followed by recognition of faces detected in the previous step followed by posting of attendance to the database.

**8.2 Project Conclusion**

Automatic posting of attendance to the database has been done successfully with minimal hardware requirements.

**8.3Future Scope** The system we have developed successfully accomplishes the task of marking the attendance in the classroom automatically. However, in order to develop a dedicated system which can be implemented in an educational institution, a very efficient algorithm which is insensitive to the lighting conditions of the classroom has to be developed. Also a camera of the optimum resolution has to be utilized in the system.

**CHAPTER 9**

**REFERENCES**

1. P. Viola and M. Jones, “Robust real-time face detection,” International Journal of Computer Vision, vol. 57, no. 2, pp. 137–154, 2004
2. D. Peleshko and K. Soroka . Research of Usage of Haar-like Features and AdaBoost Algorithm in Viola-Jones Method of Object Detection . In Proc of CADSM 2013 .
3. Adrian RhesaSeptianSiswanto, AntoSatriyoNugroho, MaulahikmahGalinium, “Implementation of Face Recognition Algorithm for Biometrics Based Time Attendance System”, IEEE Conference.
4. Faizan Ahmad, AaimaNajam and Zeeshan Ahmed, "Image-based Face Detection and Recognition: State of the Art", IJCSI International Journal of Computer Science Issues, Volume: 9, November-2012.
5. S. Paisitkriangkrai, C. Shen, J. Zhang. Face detection with effective feature extraction, in: Proceedings of the 10th Asian Conference on Computer Vision, Queenstown, New Zealand, 2010, pp.460-470.
6. Ashish Choudhary ,AbhishekTripathi ,Abhishek Bajaj ,MuditRathi and B.M Nandini "Automatic Attendance System Using Face Recognition". In International Journal of Modern Trends and Engineering.
7. ShubhiShriwastav, Dinesh Chandra Jain, " A Review on Face Recognition Attendance System". In International Journal of Computer Applications (0975 – 8887) Volume 143 – No.8, June 2016.
8. Dayanand S. Shilwant,Dr. A.R.Karwankar, "Student Monitoring By Face Recognition System". In International Journal of Electronics, Communication & Soft Computing Science and Engineering ISSN: 2277-9477, Volume 2, Issue 2.
9. MuthuKalyani.K, VeeraMuthu, "A SMART APPLICATION FOR AMS USING FACE RECOGNITION". In Computer Science & Engineering: An International Journal (CSEIJ), Vol. 3, No. 5, October 2013.
10. D. Nithya, "Automated Class Attendance System based on Face Recognition using PCA Algorithm". In International Journal of Engineering Research & Technology, Volume. 4 - Issue. 12 , December - 2015.
11. Mr. JiteshPadwal, Miss. SavitraPaharekari, Miss. ChaitaliJadhav, Miss. SurabhiNilangekar& Prof. UjwalaBodke, "Automated Attendance System Using NFC and Face Recognition". In Imperial Journal of Interdisciplinary Research (IJIR) Vol-3, Issue-1, 2017 ISSN: 2454-1362.
12. NirmalyaKar, MrinalKantiDebbarma, AshimSaha, and DwijenRudra Pal, " Study of Implementing Automated Attendance System Using Face Recognition Technique". In International Journal of Computer and Communication Engineering, Vol. 1, No. 2, July 2012.

**APPENDIX A**

**Source Code**

**Raspberry pi code for Method 1**

**Create dataset**

import cv2, sys, numpy, os

haar = 'haarcascade\_frontalface\_default.xml'

datasets = 'datasets'

sub\_data = 'Personname' #These are sub data sets of folder, for my faces I've used my name

path = os.path.join(datasets, sub\_data)

if not os.path.isdir(path):

os.mkdir(path)

(width, height) = (130, 100)

cade = cv2.CascadeClassifier(haar)

cam = cv2.VideoCapture(0)

# The program loops until it has 30 images of the face.

count = 1

while count < 31:

(\_, im) = cam.read()

gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY)

faces = cade.detectMultiScale(gray, 1.3, 4)

for (x,y,w,h) in faces:

cv2.rectangle(im,(x,y),(x+w,y+h),(255,0,0),2)

face = gray[y:y + h, x:x + w]

face1 = cv2.resize(face, (width, height))

cv2.imwrite('%s/%s.png' % (path,count), face1)

count += 1

key = cv2.waitKey(10)

if key == 27:

break

**Aligning the Faces**

import sys

importdlib

import cv2

importopenface

predictor\_model = "shape\_predictor\_68\_face\_landmarks.dat"

file = sys.argv[1]

detector = dlib.get\_frontal\_face\_detector()

pose\_predictor = dlib.shape\_predictor(predictor\_model)

aligner = openface.AlignDlib(predictor\_model)

# Load the image

image = cv2.imread(file)

# Run the HOG face detector on the image data

detected\_faces = detector(image, 1)

print("Found {} faces in the image file {}".format(len(detected\_faces), file))

# Loop through each face we found in the image

fori, rect in enumerate(detected\_faces):

# Detected faces are returned as an object with the coordinates

# of the top, left, right and bottom edges

print("- Face #{} found at Left: {} Top: {} Right: {} Bottom: {}".format(i, face\_rect.left(), face\_rect.top(), face\_rect.right(), face\_rect.bottom()))

# Get the the face's pose

landmarks = pose\_predictor(image, face\_rect)

# Use openface to calculate and perform the face alignment

alignedFace = aligner.align(534, image, face\_rect, landmarkIndices=openface.AlignDlib.OUTER\_EYES\_AND\_NOSE)

# Save the aligned image to a file

cv2.imwrite("aligned\_face\_{}.jpg".format(i), alignedFace)

**Trainer, Recognition and uploading into database code:**

import time

start = time.time()

import sys

importargparse

import cv2

import os

import pickle

import numpy as np

importopenface

importMySQLdb

db=MySQLdb.connect("localhost","root","raspberry","attdd")

c=db.cursor()

conn=sqlite3.connect('attd.db')

curs=conn.cursor()

fileDir = os.path.dirname(os.path.realpath(\_\_file\_\_))

modelDir = os.path.join(fileDir, '..', 'models')

dlibModelDir = os.path.join(modelDir, 'dlib')

openfaceModelDir = ('/home/pi')

defgetRep(img, multiple=False):

start = time.time()

bgr = cv2.imread(img)

ifbgr is None:

raise Exception("Unable to load image: {}".format(img))

rgb = cv2.cvtColor(bgrImg, cv2.COLOR\_BGR2RGB)

ifargs.verbose:

print(" + Original size: {}".format(rgb.shape))

ifargs.verbose:

print("Loading the image took {} seconds.".format(time.time() - start))

start = time.time()

if multiple:

bbs = align.getAllFaceBoundingBoxes(rgb)

else:

bb1 = align.getLargestFaceBoundingBox(rgb)

bbs = [bb1]

iflen(bbs) == 0 or (not multiple and bb1 is None):

raise Exception("Unable to find a face: {}".format(img))

ifargs.verbose:

print("Face detection took {} seconds.".format(time.time() - start))

reps = []

for bb in bbs:

start = time.time()

alignedFace = align.align(

args.imgDim,

rgbImg,

bb,

landmarkIndices=openface.AlignDlib.OUTER\_EYES\_AND\_NOSE)

ifalignedFace is None:

raise Exception("Unable to align image: {}".format(img))

ifargs.verbose:

print("Alignment took {} seconds.".format(time.time() - start))

print("This bbox is centered at {}, {}".format(bb.center().x, bb.center().y))

start = time.time()

rep = net.forward(alignedFace)

ifargs.verbose:

print("Neural network forward pass took {} seconds.".format(

time.time() - start))

reps.append((bb.center().x, rep))

sreps = sorted(reps, key=lambda x: x[0])

returnsreps

defadd\_attd(per):

print 'uploaded'

curs.execute("""INSERT INTO attd

VALUES (date('now'), (?))""",(per,))

c.execute("""INSERT INTO atend

VALUES (%s,CURRENT\_DATE(),%s,NOW())""",(per,cid))

conn.commit()

def train(args):

print("Loading embeddings.")

fname = "{}/labels.csv".format(args.workDir)

labels = pd.read\_csv(fname, header=None).as\_matrix()[:, 1]

labels = map(itemgetter(1),

map(os.path.split,

map(os.path.dirname, labels))) # Get the directory.

fname = "{}/reps.csv".format(args.workDir)

embeddings = pd.read\_csv(fname, header=None).as\_matrix()

le = LabelEncoder().fit(labels)

labelsNum = le.transform(labels)

nClasses = len(le.classes\_)

print("Training for {} classes.".format(nClasses))

ifargs.classifier == 'LinearSvm':

clf = SVC(C=1, kernel='linear', probability=True)

elifargs.classifier == 'GridSearchSvm':

print("""

Warning:

.

""")

param\_grid = [

{'C': [1, 10, 100, 1000],

'kernel': ['linear']},

{'C': [1, 10, 100, 1000],

'gamma': [0.001, 0.0001],

'kernel': ['rbf']}

]

clf = GridSearchCV(SVC(C=1, probability=True), param\_grid, cv=5)

elifargs.classifier == 'GMM': # Doesn't work best

clf = GMM(n\_components=nClasses)

elifargs.classifier == 'RadialSvm': # Radial Basis Function kernel

# works better with C = 1 and gamma = 2

clf = SVC(C=1, kernel='rbf', probability=True, gamma=2)

elifargs.classifier == 'DecisionTree': # Doesn't work best

clf = DecisionTreeClassifier(max\_depth=20)

elifargs.classifier == 'GaussianNB':

clf = GaussianNB()

elifargs.classifier == 'DBN':

fromnolearn.dbn import DBN

clf = DBN([embeddings.shape[1], 500, labelsNum[-1:][0] + 1

learn\_rate\_decays=0.9,

epochs=300,

ifargs.ldaDim> 0:

clf\_final = clf

clf = Pipeline([('lda', LDA(n\_components=args.ldaDim)),

('clf', clf\_final)])

clf.fit(embeddings, labelsNum)

fName = "{}/classifier.pkl".format(args.workDir)

print("Saving classifier to '{}'".format(fName))

with open(fName, 'w') as f:

pickle.dump((le, clf), f)

def infer(args, multiple=False):

with open(args.classifierModel, 'r') as f:

(le, clf) = pickle.load(f)

forimg in args.imgs:

print("\n=== {} ===".format(img))

reps = getRep(img, multiple)

iflen(reps) > 1:

print("List of faces in image from left to right")

for r in reps:

rep = r[1].reshape(1, -1)

bbx = r[0]

start = time.time()

predictions = clf.predict\_proba(rep).ravel()

maxI = np.argmax(predictions)

person = le.inverse\_transform(maxI)

confidence = predictions[maxI]

ifargs.verbose:

print("Prediction took {} seconds.".format(time.time() - start))

if multiple:

var=person

# print person

print 'multiple'

printvar

# print("Predict {} @ x={} with {:.2f} .".format(person, bbx,confidence))

#print("prdeictedperson".format(person))

#add(var)

add\_attd(var)

else:

var=person

print 'single'

printvar

# curs.execute("INSERT INTO attd(name,currentdate) VALUES('var',date('now'))")

add\_attd(var)

# print("Predict {} with {:.2f} ".format(person,confidence))

#print("prdeictedperson".format(person))

ifisinstance(clf, GMM):

dist = np.linalg.norm(rep - clf.means\_[maxI])

print(" + Distance from the mean: {}".format(dist))

**Code for Method 2**

**Data Set Creator**

import cv2

ca = cv2.VideoCapture(0)

detector=cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')

Id=raw\_input('enter your id')

num=0

while(True):

ret, im = ca.read()

gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY)

face = detector.detectMultiScale(gray, 1.3, 5)

for (x,y,w,h) in face:

cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)

num=num+1

cv2.imwrite("dataSet/User."+Id +'.'+ str(num) + ".jpg", gray[y:y+h,x:x+w])

cv2.imshow('frame',im)

#wait for 100 miliseconds

if cv2.waitKey(100) & 0xFF == ord('q'):

break

# break if the sample number is morethan 20

elifsampleNum>20:

break

cam.release()

**Training the dataset code**

import cv2,os

import numpy as np

from PIL import Image

recog = cv2.createLBPHFaceRecognizer()

detect= cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml");

defgetImagesAndLabels(path):

imagePaths=[os.path.join(path,f) for f in os.listdir(path)]

faceSamples=[]

Ids=[]

forimagePath in imagePaths:

pil=Image.open(imagePath).convert('L')

np1=np.array(pil,'uint8')

Id=int(os.path.split(imagePath)[-1].split(".")[1])

extract the face from the training image sample

fac=detect.detectMultiScale(np1)

for (x,y,w,h) in fac:

fasa.append(imageNp[y:y+h,x:x+w])

Ids.append(Id)

returnfasa,Ids

fasa,Ids = getImagesAndLabels('data')

recog.train(faces, np.array(Ids))

recog.save('trainner.yml')

**Recognition code**

import cv2

import numpy as np

recognizer = cv2.createLBPHFaceRecognizer()

recognizer.load('trainner/trainner.yml')

cascadePath = "haarcascade\_frontalface\_default.xml"

faceCascade = cv2.CascadeClassifier(cascadePath);

cid=sys.argv[1]

db=MySQLdb.connect(“localhost”,“root”,“raspberry”,“attd”)

co=db.cursor()

defadd\_attd(per):

co.execute(“SELECT COUNT(name) FROM aten WHERE name=’%s’ AND date=CURRENT\_DATE()”%(per))

dat=co.fetchone()

if(dat[0]==0):

co.execute(“““INSERT IGNORE INTO attend

values (%s,CURRENT\_DATE(),13,NOW())”””,(per))

db.commit()

cam = cv2.VideoCapture(0)

font = cv2.cv.InitFont(cv2.cv.CV\_FONT\_HERSHEY\_SIMPLEX, 1, 1, 0, 1, 1)

while True:

ret, im =cam.read()

gray=cv2.cvtColor(im,cv2.COLOR\_BGR2GRAY)

faces=faceCascade.detectMultiScale(gray, 1.2,5)

for(x,y,w,h) in faces:

cv2.rectangle(im,(x,y),(x+w,y+h),(225,0,0),2)

Id, conf = recognizer.predict(gray[y:y+h,x:x+w])

if(conf<50):

if(Id==1):

Id="gourav"

elif(Id==2):

Id="sundeep"

elif(Id==3):

Id="chanakya"

elif(Id==4):

Id="sairam"

elif(Id==4):

Id="mv"

else:

Id="Unknown"

cv2.cv.PutText(cv2.cv.fromarray(im),str(Id), (x,y+h),font, 255)

cv2.imshow('im',im)

if cv2.waitKey(10) & 0xFF==ord('q'):

break

cam.release()

cv2.destroyAllWindows()

**CODE of HTML page**

<html>

<head>

<title>The Materialize Range Example</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/materialize/0.97.3/css/materialize.min.css">

<script type="text/javascript" src="https://code.jquery.com/jquery-2.1.1.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/0.97.3/js/materialize.min.js"></script>

</head>

<nav>

<body>

<div class="nav-wrapper" center>

<a href="#" class="brand-logo center">Attendence system </a>

<ul id="nav-mobile" class="right hide-on-med-and-down">

</ul>

</div>

</nav>

<div class="row" center >

</div>

<div class="row" center>

<form class="col s8 m4 " method='post'>

<div class="row">

<label>enter u r class id</label>

<input type="text" name="cid" >

<label>Pick the date that u want to check the attendence</label>

<input type="date" name="da" class="datepicker" >

</div>

<button class="btn waves-effect waves-light" type="submit" name="action" >Submit

<i class="material-icons right">send</i>

</button>

</form>

</div>

<?php

if(isset($\_POST['action']))

{

if(!empty($\_POST['cid']) && !empty($\_POST['da']))

{

$host="localhost";

$user="id1432444\_gourav";

$password="1604gourav";

$database="id1432444\_attdd";

$connect=mysqli\_connect($host,$user,$password,$database);

if($connect)

{//echo "Connected to the server...!!";

}

else

die(mysqli\_error());

if (mysqli\_connect\_errno())

{

echo "Failed to connect to server: " . mysqli\_connect\_error();

}

$select = mysqli\_select\_db($connect,$database);

if($select)

{//echo "Selected Database...!!";

}

else

die(mysqli\_error($connect));

$cal\_date=$\_POST['da'];

$date=date('Y-m-d',strtotime($cal\_date));

$que=mysqli\_query($connect,"SELECT name FROM atend WHERE dat='$date'" );

echo "Students Present for the class";

echo"<br>";

if (mysqli\_num\_rows($que) >= 0)

{

echo" <table><thead><tr><th>Name</th></tr></thead><tbody>";

while($row = mysqli\_fetch\_array($que))

{

echo "<tr>";

echo "<td>";

echo $row['name'];

echo "</td>";

echo "<tr>";

}

echo "</tbody></table>";

}

}

mysqli\_close($connect);

}

?>

</body>

</html>