ATTENDANCE DETECTION USING RASPBERRY PI

A Mini Project Report

Submitted to the APJ Abdul Kalam Technological University

in partial fulfillment of requirements for the award of degree

Bachelor of Technology

in

Electronics and Communication Engineering

by

Erin Edward George. (KME20EC013)

Hana V.K. (KME20EC015)

Anupama P.T. (KME20EC008)

N.Z Fadhiya. (KME20EC027)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

KMEA ENGINEERING COLLEGE

KERALA

July 2023

DEPT. OF ELECTRONICS & COMMUNICATION ENGINEERING

KMEA ENGINEERING COLLEGE

2022-2023

CERTIFICATE

This is to certify that the report entitled **ATTENDANCE DETECTION USING RASPBERRY PI** submitted by **Erin Edward George**, **Hana V.K**, **Anupama P.T**, **N Z Fadhiya** to the APJ Abdul Kalam Technological University in partial fulfillment of the B.Tech. degree in Electronics and Communication Engineering is a bonafide record of the project work carried out by them under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Mr. Binish M.C
(Project Guide)
Assistant Professor
Dept. of ECE
KMEA Engineering College

Ms. Abini A. R.
(Project Coordinator)
Assistant Professor
Dept. of ECE
KMEA Engineering College

Dr. Shahin M. Abdulla
HOD
Dept. of ECE
KMEA Engineering College

External Examiner

DECLARATION

We hereby declare that the mini project report **'Attendance Detection Using Raspberry PI'**, submitted for partial fulfillment of the requirements for the award of the degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of **Asst. Prof. Binish M.C** . This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the sources. We also declare that we have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed as the basis for the award of any degree, diploma, or similar title of any other University.

Erin Edward George

Hana V.K

Anupama P.T

N.Z Fadhiya

Ernakulam

25-07-2023

ABSTRACT

Smartphone based face attendance system is the need in today's digital world and age for universities and schools as smartphones with very good picture quality provided by its camera makes such systems really affordable and practical as far as implementation is concerned. This project presents a modern attendance system that uses facial recognition and automates the attendance system. It is built on a Raspberry Pi computer with a camera and a screen. When a student's face is captured by the camera, the system uses the Local Binary Patterns algorithm to recognize them. If the student is recognized then their attendance is marked or updated to a text file. The system achieves an accuracy rate of 70 percentage with a dataset of 5 person images. Overall, this system simplifies attendance tracking and improves security.

ACKNOWLEDGEMENT

We take this opportunity to express our deepest sense of gratitude and sincere thanks to everyone who helped us to complete this work successfully. We express our sincere thanks to Dr. Shahin M.Abdulla, Head of Department, Electronics and Communication Engineering, KMEA Enigineering College for providing us with all the necessary facilities and support. We would like to express our sincere gratitude to the **Asst. Prof. Abini M.A**, department of Electronics and Communication Engineering, KMEA Enigineering College Ernakulam for the support and co-operation. We would like to place on record our sincere gratitude to our project guide **Asst. Prof. Binish M.C,** Assistant Professor, Electronics and Communication Engineering, KMEA Enigineering Collegefor the guidance and mentorship throughout this work. Finally I thank my family, and friends who contributed to the succesful fulfilment of this mini project.

Erin Edward George.

Hana V.K.

Anupama P.T.

N.Z Fadhiya.

Contents

List of Figures

List of Abbreviations

LBPH- Local binary pattern histogram

SQL-Structured query language

HOG-Histogram of oriented gradients

SVM-Support vector machine

BP-Local binary pattern

DCNN-Deep convolutional neural network

LMS-Learning management system

PTZ- Pan-tilt-zoom

KNN-K-nearest neighbors algorithm

MySQL-My structured query language.

CNN-Convolutional neural network

CCTV-Closed-circuit television

PCA-Prompt corrective Action

LDA-Linear discriminant analysis

ANN-Artificial neural networks

PDBNN-Probabilistic decision-based neural network

R-CNN-Regions with convolutional neural networks

NMS-Network management station

RGB-Red, Green, Blue

Chapter 1

INTRODUCTION

Attendance tracking is a fundamental task in various domains, including educational institutions, workplaces, and events. The traditional manual methods of recording attendance are time-consuming and prone to errors.To address these challenges, automated systems utilizing computer vision techniques have emerged as efficient and accurate solutions.This project focuses on the combination of a Raspberry Pi, camera module, and the application of Haar cascade frontal face detection and Linear Block Pattern Histogram (LBPH) for multiple attendance detection.The Raspberry Pi, a versatile single-board computer, offers a cost-effective and flexible platform for diverse applications.Coupled with a camera module, the Raspberry Pi becomes capable of capturing images or recording video, providing a powerful foundation for attendance tracking systems.Haar cascade frontal face detection is a widely used technique for detecting faces in images or video frames.By employing a trained classifier that utilizes Haar-like features, this technique can efficiently identify facial patterns and features.Haar cascade frontal face detection excels in real-time face localization, making it an ideal choice for attendance tracking. To accurately recognize and track multiple individuals, face recognition algorithms are crucial.One such algorithm is LBPH, which analyzes facial features and extracts patterns for individual identification. LBPH constructs histograms of facial regions and compares them against a database of known faces, enabling reliable face recognition in attendance tracking scenarios. By integrating the Raspberry Pi, camera module, Haar cascade frontal face detection, and LBPH, a comprehensive attendance tracking system can be developed. The camera module captures images or video frames, which are processed using Haar cascade frontal face detection to detect and localize faces accurately.Subsequently, LBPH analyzes the recognized faces to identify individuals and record their attendance.It is important to consider the advantages and limitations of these techniques to ensure the system's reliability and performance.Factors such as varying lighting conditions, pose variations, and the quality of training data can influence the accuracy of face detection and recognition algorithms.In this project, our aim is to explore the integration of a Raspberry Pi, camera module, Haar cascade frontal face detection, and LBPH for multiple attendance detection.We will provide a concise overview of the advantages and disadvantages of these techniques and evaluate the results obtained from the system.Through this research, we aim to contribute to the development of efficient and accurate attendance tracking systems using computer vision techniques.

1.1 Problem Statement

The traditional methods of manual attendance tracking are prone to errors, time-consuming, and inefficient, necessitating the need for automated systems. The problem at hand is to develop an automated attendance tracking system using computer vision techniques, specifically by integrating a Raspberry Pi, camera module, Haar cascade frontal face detection, and LBPH for multiple attendance detection. The challenges with manual attendance tracking methods include the risk of human error in recording attendance, the time required to manually mark attendance for a large number of individuals, and the inability to track attendance in real-time.These challenges can result in inaccurate attendance records, administrative burden, and potential loss of productivity.

To address these challenges, the proposed solution involves leveraging computer vision techniques to automate the attendance tracking process. The Raspberry Pi serves as the foundation for the system, offering computational power and flexibility.The camera module captures images or video frames, which are processed using Haar cascade frontal face detection to accurately detect and locate faces.The LBPH algorithm is then applied to recognize and identify multiple individuals, allowing for reliable attendance marking.

The problem statement involves designing and implementing a system that can overcome the limitations of manual attendance tracking methods by automating the process using computer vision techniques.The system should be capable of accurately detecting and recognizing multiple faces, even in real-time scenarios, and recording attendance efficiently. Additionally, the system should address challenges such as varying lighting conditions, pose variations, and the quality of training data to ensure reliable and accurate attendance tracking.

The successful implementation of this automated attendance tracking system would result in improved accuracy, reduced administrative burden, and real-time tracking capabilities.It would provide educational institutions, workplaces, and event organizers with an efficient and reliable method for attendance management, leading to increased productivity and streamlined operations.

1.2 Objectives

The objective of this research project is to develop an automated attendance tracking system by integrating a Raspberry Pi, camera module, Haar cascade frontal face detection, and LBPH for multiple attendance detection.The system aims to overcome the limitations of manual attendance tracking methods by leveraging computer vision techniques.The specific objectives are as follows:

Firstly, the project aims to design and develop a functional attendance tracking system that utilizes the Raspberry Pi as the computational platform and a camera module for image or video capture.This involves setting up the hardware components, configuring the software environment, and establishing the communication between the Raspberry Pi and the camera module.

Secondly, the system will incorporate Haar cascade frontal face detection to accurately detect and localize faces within the captured images or video frames.The objective is to achieve robust face detection performance, even in challenging scenarios with varying lighting conditions and pose variations.

Thirdly, the project will implement the LBPH algorithm for face recognition.This involves analyzing the detected faces, extracting facial features, and comparing them with a database of known faces for identification.The goal is to develop a reliable face recognition component capable of accurately recognizing multiple individuals.

Next, the system will mark the attendance of recognized individuals based on their identification through the face recognition component. The objective is to create an efficient and accurate attendance marking mechanism that can handle real-time or near real-time processing requirements.

Furthermore, the project will conduct thorough testing and evaluation of the developed system.The objective is to assess the system's accuracy, efficiency, and robustness.This evaluation will involve testing the system under various lighting conditions, different facial poses, and diverse datasets to measure its reliability in different scenarios.

Additionally, the research will focus on optimization and refinement of the system.The objective is to identify areas for improvement, such as enhancing face detection and recognition accuracy, reducing processing time, and optimizing the hardware configuration.Necessary modifications will be implemented to enhance the overall performance of the system.

Lastly, the project will validate the developed attendance tracking system by comparing its results with manual attendance records or alternative attendance tracking methods.The objective is to assess the system's effectiveness in accurately recording attendance and reducing administrative burden.

Overall, the objective is to contribute to the development of an automated attendance tracking system that offers accurate, efficient, and reliable attendance recording using computer vision techniques.The system aims to streamline attendance management in educational institutions, workplaces, and events, leading to improved productivity and efficiency.

Chapter 2

LITERATURE REVIEW

Sayan Seal *et al.* (2020) proposed an approach towards the development of an automated attendance system using face detection and recognition [1].The steps can be broken down to dataset collection, and face detection using two different approaches namely Viola Jones Method (HAAR Cascade method) and Histogram of Oriented Gradient (HOG), face recognition methods include (i) deep-learning based approaches such as Inception V3 Model, Siamese Neural Networks and (ii) non-deep learning approaches, such as Principal Components Analysis using Eigen Faces, Local Binary Pattern Histogram (LBPH), SQL(structred query language) database for storing the data. An accurate strong classifier can be trained by the Cascade function by taking a weighted average of the decisions made by the weak classifiers.Each weak classifier detects one of the many defining features of the object.However, this method is extremely sensitive to parameters, scaling and minimum neighbours as a whole.The HOG face detector has been used as a possible solution to these problems.The HOG extracts the features by a detector window and is fed to a linear Support Vector Machine for object or non-object classification. Another advantage of HOG is that the threshold can easily be specified as the sizes of the actual faces are much bigger compared to that of an image being displayed on a mobile screen when viewed from a particular distance.The recognition process has three stages, identification, recognition, and categorization. It has an interface, made in Angular, for automatic marking of attendance from group photos and Structed Query Language database for storing data. High resolution cameras and non-frontal face detection can make the system more accurate.

Harikrishnan J *et al.* (2019) proposed a Vision-face recognition monitoring system for surveillance using deep learning technology and computer vision [2]. The steps involved are Face detection and gathering of data, training the recognizer, facial recognition, and attendance management in Excel. The image pre-processing steps involve the conversion of the input image to a grayscale image of 8bits, bicubic interpolation to find the position of two nearby points and orientation at those positions, Thresholding using Torezo to enhance facial features, Morphological closing to remove the noise and close the disjoint areas in the image, followed by Erosion which removes white noises by discarding pixels near the boundary. Face detection is done by Haar Cascade Method (cv2.CascadeClassifier). Each feature like eyes, nose, and lips has a threshold value which identifies the face and nonface images and is classified. The training operation is performed by a function contained in OpenCV. The recognizer used here is Local Binary Patterns Histograms (LBPH), a 3x3 matrix window which is moved along the image.The neighbouring pixel values are compared with the centre pixel.If the values measured are greater than or equal to the central pixel value then that pixel value is read as 1 and the rest as 0. The values are read in clockwise order, forming a binary value which is the local binary pattern of that specific area. This is done to the entire image and LBPs are obtained.The binary value obtained is then compared with the trained histogram value set and the best matching histogram is returned with the label of the person. Using single snap mode, function snap() is used to capture the face and marks attendance in the Excel sheet.Results tested of 74 per cent accuracy and can be configured with Raspberry Pi.

Pradeesh N *et al.* (2019) proposed fast and reliable group attendance marking system using face recognition in classrooms [3]. Attendance is marked using a fixed camera in a classroom. Some factors to be considered are, bright environment to get fine details, not using hight quality cameras as it increases computation time, accuracy of the system. For a fast and reliable system attendance is marked in groups, as it doesn't take much time taking attendance individually. Proposed solution involves using deep convolutional neural network (DCNN), a pretrained ResNet V1 model built in using dlib C++ library. The details of the students are stored in AMPLE, an existing LMS (Learning Management System). The AMPLE convert student images to Embedding, a vector size of 128 for easier storage and processing speed. K-nearest neighbour (KNN) classifier is used for classification task, an initial step to train student images and creates a classifier for future use. To reduce error and increase accuracy, maximum of 15 students are arranged to face the camera. When a face is detected, the classifier compares it with the threshold distance to the face distance, if the face distance is greater than the threshold distance, then it is considered as an unknown face and is sent for manual verification. Improvements can be made by using PTZ cameras, training Resnet using low light and low resolution images.

Omar Abdul Rahman Salim *et al.* (2018) [4] proposed a system in which a door will open once the attendance of the person is noted, thereby letting him or her into the room of the event. This is a web-based system where the attendance is stored in MySQL database. Initially, the captured face or image is cut and cropped to the ROI or Region of Interest. This is done using Haar Feature-based Cascade algorithm.The proposed methodology in this paper is LBP- Linear Block Pattern, with the aid of which we compare the facial features with the datasets. If the person is identified the servo motors may rotate, opening the door to the the lecture hall or the room of the event. The training of the data set was done by a minimum of 12 images of each person with different facial expressions. The reason to specifically choose LBP as the methodology is because it can update the classifier model with new images in an exuberant way.

Payal Patil *et al.* (2020) proposed the comparative analysis of facial recognition models using video for a real-time attendance monitoring system [5]. It uses the Viola -jones (haar cascade) method and CNN algorithm for face detection and recognition respectively. Viola-jones is a machine learning object detection method used to identify image-based or video-based features that use numerical information extracted from the image. CNN is used to extract a feature from a given image by reducing the image for easy processing without losing its features. The procedure starts by capturing the image from a 10-second video then the haar cascade detects the face, and extracts the face from the frame for augmentation. Then it is stored in a single folder with the student's name. This database is a passed-to-face net model. In the recognizer part of the system (CNN), first input the video from a webcam or CCTV in which multiple faces is detected from a single frame, which is passed through a classifier for recognition using CNN. Then names of students will display on corresponding faces with a bounded box. The array of recognized face pass to the system and is marked in an Excel sheet. The resultant cascade method has better accuracy rather than others. This method detects the maximum number of faces without using augmentation

Allan Jason C. Areceo *et al.* (2020) proposed the E-Attendance Checker through Facial Recognition using a Histogram of Oriented Gradients(HOG) with Support Vector Machine(SVM)[6]. It involves a design of an e-attendance checker that was established using HOG and SVM algorithms for face detection and recognition respectively. HOG along with SVM was applied to examine the effect of luminance of the surrounding the facial orientation of the student and so as their distance from the camera in the facial detection and recognition. The process flows from taking an image and then implementing face detection via HOG which transpires and ends with the face recognition using the SVM algorithm. Its output was shown in an online database that includes the names of the students and the date and time the attendance was taken. It was found that the system has an accuracy of 95.65 percentage and can detect and recognize up to 37 students. In this system, it is suggested that the classroom should have a luminance level of about 217.39 lux or higher to achieve a better accuracy performance of the system. This system has a drawback in accordance with the given current covid 19 situation, as the accuracy of the system will be affected. The accuracy of the system will increase for which the students will have social distancing lessening the interference of the faces where as the accuracy will also decrease as students will now be wearing face masks that will affect the face detection and recognition of the system. With this it is best to add more data or training images to overcome the effect of the face mask.

Khem Puthea *et al.* (2017) proposed the attendance marking system based on face recognition[7]. This system implements LBP (local binary pattern) to recognize the human face and SVM (support vector machine) to find the similarities of the images stored in the database with the image acquired by the camera inside the classroom. The facial recognition process has many steps such as capture, extraction, comparison, and matching. Capture is the way to snap a picture during the enrollment of the system. Extraction is used for finding specific features from the face. In comparison, new input is used for comparison with the database. The last step is matching: the system will try to find the matching of the new face with the registered face based on the extraction and comparison process. There are various approaches for facial recognition such as geometrical approach, PCA based, LDA-based, local binary pattern, and active appearance model. Principle component analysis (PCA) is a statistical method used in facial recognition for feature extraction and used to remove redundant information. The main goal of PCA is keeping the original information of the data, minimizing original laws information, and improving the face analysis. The result of the study shows that the PCA algorithm is incredibly effective in extensive databases. PCA has better performance in the system of attendance management based on facial recognition than the manual attendance system which is time-consuming. ANN and PCA have been integrated to solve a blocking issue of the attendance management system based on facial recognition. However, this system still has issues with system performance and accuracy in recognizing human faces. Future work will use fast PCA with back-propagation to resolve this problem.

Shang-Hung Lin *et al.* (1997) proposed a face recognition system based on probabilistic decision-based neural networks (PDBNN)[8]. The PDBNN face recognition system consists of three modules which are face detection, eye localizer, and face recognition. First, the face detector detects the position of the face in the image. The eye localizer then determines the position of the eyes to generate feature vectors. Finally, the third module is face recognition. PDBNN can be used effectively for all three modules.PDBNN also gets its architectural features from decision-based neural network(PDBNN). The main problem and difficulty in face detection is to consider the many variations of the face image. There are many ways to deal with these such as variables spatial image literals, correlation patterns and view-based property fields, etc. The input images from the camera are converted to 8-bit greyscale image of size 640*480 by SIV digitizer board and are stored in the frame buzzer. If the PDBNN face detector detects a face then the eye localizer is activated to locate the eyes.A sub-image of a size 140*100 is extracted from the face region and is fed to the face recognizer for verification.The processed image is then compared with the trained image for face detection.Edge filtering and histogram modification techniques can be applied to recondition the facial images. Sometimes the image is processed through edge filtering and histogram techniques which reduces the image size and increases storage capacity. The system is built to overcome the disadvantages of low lighting, wearing glasses and can detect variations of face features and cluttered backgrounds.

Jack Febrain Rusdi *et al.* (2020) proposed student attendance using face recognition technology[9].The technologies used in recording student attendance include face recognition technology through computer vision and Matlab. Some face recognition methods are computer vision, both with deep learning and the use of convolution neural networks and Raster R-CNN Matlab.The four stages in attendance-taking system are face taking module, training module, camera module and attendance module. The system training and testing are carried out by CNN library functions. The images stored in the library are used as references for predicting student's faces from the input. More reference images will produce higher accurate results. The camera in a classroom detects objects in the shape of faces is compared with the face library for predictions. The accuracy and performance of the attendance system is influenced by algorithms, completeness of the library, facial recognition aids, camera resolution, processor and memory strength, number of object image libraries

Kaipeng Zhang *et al.* (2016) proposed joint face detection and alignment using multitask cascaded convolutional networks[10]. To increase efficiency, haar-like features and AdaBoost classifiers is used to train cascaded classifiers. It shows real-time accuracy with advanced features. The CNN method has a high face detection time cost. due to that cascaded-CNN is used. The disadvantage of CNN is that there is no filter in the convolution process and face detection for binary classification is a difficult task. The proposed network (P-Net) is used for the candidate facial window and for bounding box regression vectors. After that non maximum suppression is obtained to connect the overlapped candidates. All candidates are then sent to another CNN called (R-Net) which rejects false candidates and then generates NMS with bounding box regression. Face classification, face marker localization, multi-site training, and online complex model mining are used for the training of images.

Hemantkumar Rathod *et al.* (2017) proposed machine learning-based approach on automatic attendance system[11]. Here, face detection and face recognition is done using cameras installed in the classroom to find different faces. These faces are then edited and cropped to reduce the size of the image before they are converted to grayscale images and these faces are compared to the database. thereafter it displays the result and attendance is marked. The student photos used to create the document were taken from a variety of angles and gestures. Each student with 8 to 10 images is taken and converts this RGB image to a grayscale image and then it is sized in to 112*92 pixels to reduce time. The image folder is named as a database with subfolders having different faces of a single person is after named to the corresponding subfolder. HOG (Histogram of Oriented Gradients) is often used in computer vision. This method checks for dynamic changes, content quality, and more. To use HOG, the image is divided into small regions and then the gradient direction of the histogram is calculated. Arrange all the gradients into angular bins. Each pixel represents the weight of the gradient relative to its angle. This histogram is then converted into blocks, also known as histogram groups or normalized groups , which represent the descriptor. The Viola-Jones algorithm was used to capture images. When the camera is installed in the classroom, the algorithm detects the face and then it is cropped for eye detection. This modified image is saved in the test file. Compare this with the ”database” folder behind the ”test” folder. On the basis of supervised learning object recognition is processed. After training with certain objects (faces), a decision can be made using a support vector machine (SVM). After comparison, the classifier is used to extract the face with the best similarity from the stored database file. These extracted images are used for marking the attendance. The overall system is implemented using MATLAB.

A.S. Hasban *et al.* (2019) proposed students face recognition using Raspberry Pi[12]. Here, face detection focuses on frontal behavior and facial recognition is driven by images captured by the Raspberry Pi camera. The camera takes pictures of the students and keeps them on database file. The unknown face is detected in the template and it is not saved to main file. To observe faces from different angles on the Raspberry Pi camera, Face recognition is divided into three steps: data collection, extraction and training, testing or face recognition. This also focuses to provide less noise, better image recognition, random detection, computations, etc. after this image is extracted and changed into a vector with some parts of the data zone (which includes the student’s image). Then it is moved in to document and calculation is done to get more information from the image. The main device used here is a raspberry pi 2, and a CSI camera with a USB port is used to connect all other devices. The algorithm used is the Haar cascade classifier. OpenCV is the installed library and the interpreter used here is python3. Different classes are used for face, eyes, and smile. Parameters used by the classifier include scaleFactor, minNeighbors, and minSize. scaleFactor determines how much to reduce the image size for each image. The minNeighbors parameter specifies how many neighbors each student rectangle can have. The minSize parameter represents the minimum rectangular face size to be considered. Here with the raspberry pi 2, 56 percent recognition accuracy and 120ms normal operating time is achieved in the Linux phase. Raspberry Pi night vision can be used for better photos at night. Thus, the attendance status of students can be recorded by this method and the names of absent students are marked on the screen.

D. Sri Sai Mahesh *et al.* (2020) proposed Facial Detection And Recognition System on Raspberry Pi with Enhanced Security[13]. The facial detection algorithm used here is the haar cascade classifier and the face recognition algorithm are Eigenface, Fisherface, Telegram and LBPH. Here raspberry pi 3 with the passive infrared sensor is used. Large data, redundant images may trouble the detection of image because of that the image is converted into grayscale and feature extraction is obtained. Feature extraction is done by a sampling of image. then the image is compared with the extracted image with the help of HAAR and AdaBoost features. LBP is a technique which covers the image into small parts of zones and features get extracted in each part of the image pixels in to binary pattern. Then it is combined together to obtain a picture. The nearest neighbor (KKL) is placed according to the histogram matching method and face recognition is done after extraction of the LBPH feature. Here a security system is used, that compares the image with trained image and if the face detected is right it provides access to open the door. Telegram Bot API is used to send instructions to a raspberry pi. It is applicable in the field of banking, home security, etc. The main aim of this is to provide security to the real world. It opens the door for the authorized person by facial detection and output would be displayed in the screen, if the wrong person tries to enter the door then the image will be stored in a system database and an alert message will be passed to the user.

Seema Singh *et al.* (2019) proposed Facial Recognition Machine Learning Algorithms on Raspberry Pi[14]. In this face is detected by using face biometrics which carries three steps, first training the system with captured images, secondly categorizing them into different classes, thirdly storing these classified images to the database system. HOG and Haar-Cascade are the two algorithms used. Haar classification is applied on Raspberry Pi and the final image can be captured at the output. To increase the image-capturing speed integral image technique is used. The selection process is done through AdaBoost that converts to classifiers and all these classified images are then cascaded to increase the speed of detection with exact image in the output. Haar-Cascade algorithm compares the light and darker part of the image. The HOG model represents a histogram with tonal distribution of image data. The intensity of the image is shown by image gradient which provides image extraction with help of sobel filters. The steps involved in developing histogram gradient is preprocessing, producing gradient image and generating 8*8 cells for histogram gradients. Haar algorithm provides speed, precision and accuracy on the detection of image when connected with raspberry pi and HOG model is not working properly with a raspberry pi so it is replaced by Intel i5 processors for better real-time exposure. HOG provides more accurate results in facial recognition with better result as compared to Haar. But both of these methods vary according to their feature specifications

Chapter 3

METHODOLOGY

3.1 Block Diagram

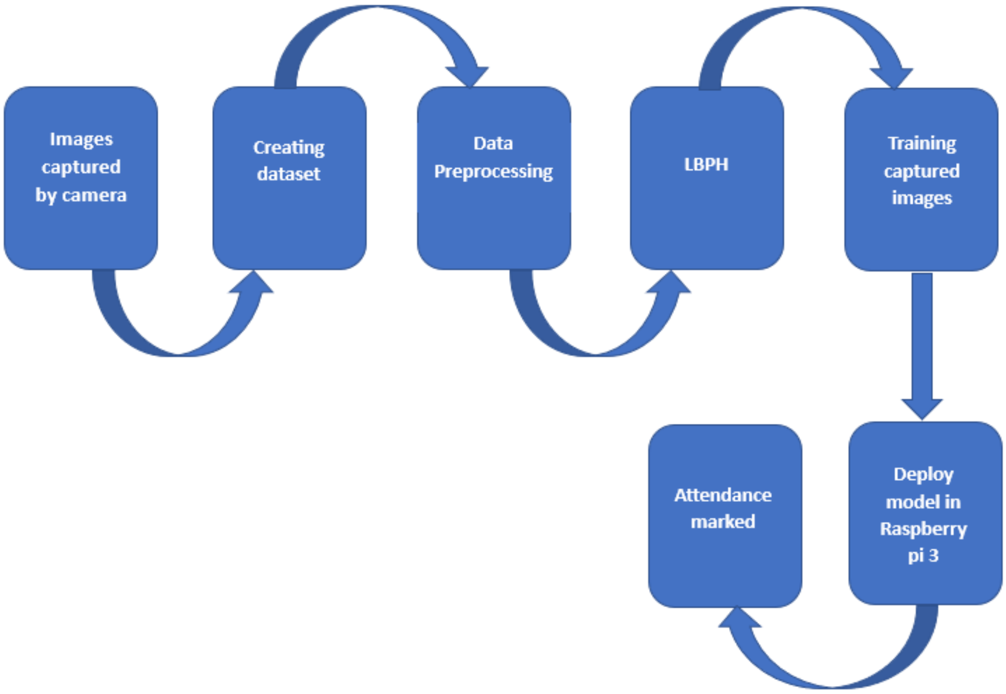

Figure 3.1: Block Diagram of experimental design

1. Image Captured by Camera: This block represents the camera module connected to the Raspberry Pi. The camera captures images of individuals or a video stream containing frames, which serve as input for the attendance system.

2. Creating Dataset: This block represents the process of creating a dataset of

images for training the attendance system. It involves capturing multiple images of individuals under different conditions, such as different lighting, angles, and facial expressions. These images are used to build a dataset that will be used for training the face recognition model.

3. Data Preprocessing: This block represents the preprocessing step applied to the captured images before feeding them to the face recognition model. Preprocessing techniques can include resizing, cropping, normalization, or applying filters to enhance the quality and consistency of the images. These preprocessing steps ensure that the images are in a suitable format and condition for the subsequent face recognition process.

4. LBPH : This block represents the LBPH algorithm used for face recognition. The LBPH algorithm analyzes the preprocessed images and extracts local binary patterns to create histograms that represent the unique facial features of individuals. These histograms serve as templates for recognizing and comparing faces.

5. Training Captured Images: This block represents the training phase of the face recognition model. The captured and preprocessed images, along with their corresponding labels, are used to train the LBPH model. The model learns to recognize and differentiate individuals based on the patterns and features extracted during training.

6. Deploy Model in Raspberry Pi 3: This block represents deploying the trained face recognition model on the Raspberry Pi 3. The model, after being trained on a separate computer, is transferred and installed on the Raspberry Pi 3 for real-time face recognition.

7. Attendance Marked: This block represents the final step of marking attendance based on recognized faces. The deployed face recognition model processes the live video stream captured by the camera on the Raspberry Pi 3. It compares the detected faces with the learned representations of faces from the training phase. Based on the recognition results, the attendance system marks the presence of individuals and updates the attendance records accordingly.

3.2 Proposed Methodology

3.2.1 Haar Cascade frontal face detection

The objective of this research project is to develop an automated attendance tracking system by integrating a Raspberry Pi, camera module, Haar cascade frontal face detection, and LBPH for multiple attendance detection. The system aims to overcome the limitations of manual attendance tracking methods by leveraging computer vision techniques. The specific objectives are as follows:

Firstly, the project aims to design and develop a functional attendance tracking system that utilizes the Raspberry Pi as the computational platform and a camera module for an image or video capture. This involves setting up the hardware components, configuring the software environment, and establishing the communication between the Raspberry Pi and the camera module.

Secondly, the system will incorporate Haar cascade frontal face detection to accurately detect and localize faces within the captured images or video frames. The objective is to achieve robust face detection performance, even in challenging scenarios with varying lighting conditions and pose variations.

Thirdly, the project will implement the LBPH algorithm for face recognition. This involves analyzing the detected faces, extracting facial features, and comparing them with a database of known faces for identification. The goal is to develop a reliable face recognition component capable of accurately recognizing multiple individuals.

Next, the system will mark the attendance of recognized individuals based on their identification through the face recognition component. The objective is to create an efficient and accurate attendance marking mechanism that can handle real-time or near-real-time processing requirements.

Furthermore, the project will conduct thorough testing and evaluation of the developed system. The objective is to assess the system's accuracy, efficiency, and robustness. This evaluation will involve testing the system under various lighting conditions, different facial poses, and diverse datasets to measure its reliability in different scenarios.

Additionally, the research will focus on optimization and refinement of the system.

The objective is to identify areas for improvement, such as enhancing face detection and recognition accuracy, reducing processing time, and optimizing the hardware configuration. Necessary modifications will be implemented to enhance the overall performance of the system.

Lastly, the project will validate the developed attendance tracking system by comparing its results with manual attendance records or alternative attendance tracking methods. The objective is to assess the system's effectiveness in accurately recording attendance and reducing administrative burden.

Overall, the objective is to contribute to the development of an automated attendance tracking system that offers accurate, efficient, and reliable attendance recording using computer vision techniques. The system aims to streamline attendance management in educational institutions, workplaces, and events, leading to improved productivity and efficiency.

3.2.2 Local Binary Pattern Histogram

LBPH is a face recognition algorithm commonly used in computer vision applications. It operates by analyzing local patterns in the texture of a face image to create a representation known as a histogram. The LBPH algorithm follows the following steps:

1. Image Preprocessing: The input face image is preprocessed to enhance its quality and normalize variations in illumination and pose. Common preprocessing techniques include grayscale conversion, histogram equalization, and alignment.

2. Local Binary Patterns Extraction: The face image is divided into small regions or patches, such as cells or pixels. For each region, the Local Binary Pattern (LBP) operator is applied. The LBP operator compares the pixel values of a central pixel with its neighboring pixels and encodes the result as a binary pattern. This process is performed for each pixel in the region, resulting in a binary code.

3. Histogram Creation: After extracting the LBP codes for each region, a histogram is constructed by counting the occurrences of different LBP patterns within the region. The histogram captures the distribution of local texture patterns within the face image.

4. Face Representation: The histograms from all the regions are concatenated to

form a single feature vector, representing the face. This feature vector encodes the texture information of the face based on the local patterns.

5. Face Recognition: During the recognition phase, the LBPH algorithm compares the feature vector of the input face with a database of known face feature vectors. The comparison is typically done using distance metrics such as Euclidean distance or cosine similarity. The input face is classified as the individual whose face feature vector is most similar to it.

LBPH is known for its simplicity and efficiency, making it suitable for scenarios with limited training data or when computational resources are constrained. However, it may be less robust to variations in pose, illumination, and facial expressions compared to more advanced deep learning-based approaches. Nonetheless, LBPH can provide reliable face recognition results in many practical applications.LBPH has been widely used in attendance tracking, access control systems, and surveillance applications, enabling accurate recognition of individuals based on their facial texture patterns.

3.3 Flowchart

There are three main parts of the Automatic Face Recognition and Detection system which include Database, Training Module, and Testing Module.

A. Database: Database is the first and most important part of Automatic Face Recognition and Detection system. The database consists of lots of images of students whose attendance needs to be marked. When the camera opens, then the Haar Cascade classifier is loaded on image. Using frontal face Haar Cascade classifier, the face is detected from the image. If the face is detected then the system takes images of that person and stores image at assigned location. The system takes 2-3 minutes for making database of one person. Thus, it consumes less time of user. Once the database is saved the next step goes to the training phase.

B. Training Model: Training Model is the second important part of Automatic Face Recognition and Detection system. Here we are going to train our recognizer using a pre-set label Image database. Here the gray scale images which are stored into database are converted into matrix using Numpy. The Numpy is a library /general purpose array processing package which provides a high performance multidimensional array object, and tools for working with these arrays. In the next step, Haar Cascade Classifier and Recognizer are loaded on these gray scale images. The Haar Cascade detects the face from the image and the recognizer helps the system to perform the operation of feature extraction from images. The OpenCV provides 16 Haar Cascade to detect face, eyes, object, and text. Haar Cascade use adaboost algorithm to detect facial organs like eyes, nose and mouth. Using Haar wavelet it draws rectangular frame around the face. After the feature extraction step, the data is stored into a file at an assigned location.

C. Testing Model: Testing Model is the third important part of Automatic Face Recognition and Detection system. In Testing Model the real-time image is compared with the real-time video and recognize the authorized person. Here the real time color image which is captured by camera is converted into gray scale image. Using Haar Cascade and recognizer system detects the face from the image and performs feature extraction operation on images. The next step is to calculate the degree of similarity and other parameters between real-time image and real-time video. Here system compares the face between real-time image and real-time video. In this automatic face recognition system, the angle of face, surrounding light conditions, distance between person and camera and facial expressions are also considered while recognition. If the degree of similarity and other parameters between real-time image and real-time video are matched then the authorized person's name is displayed on rectangular window, otherwise unknown person is displayed on rectangular window.

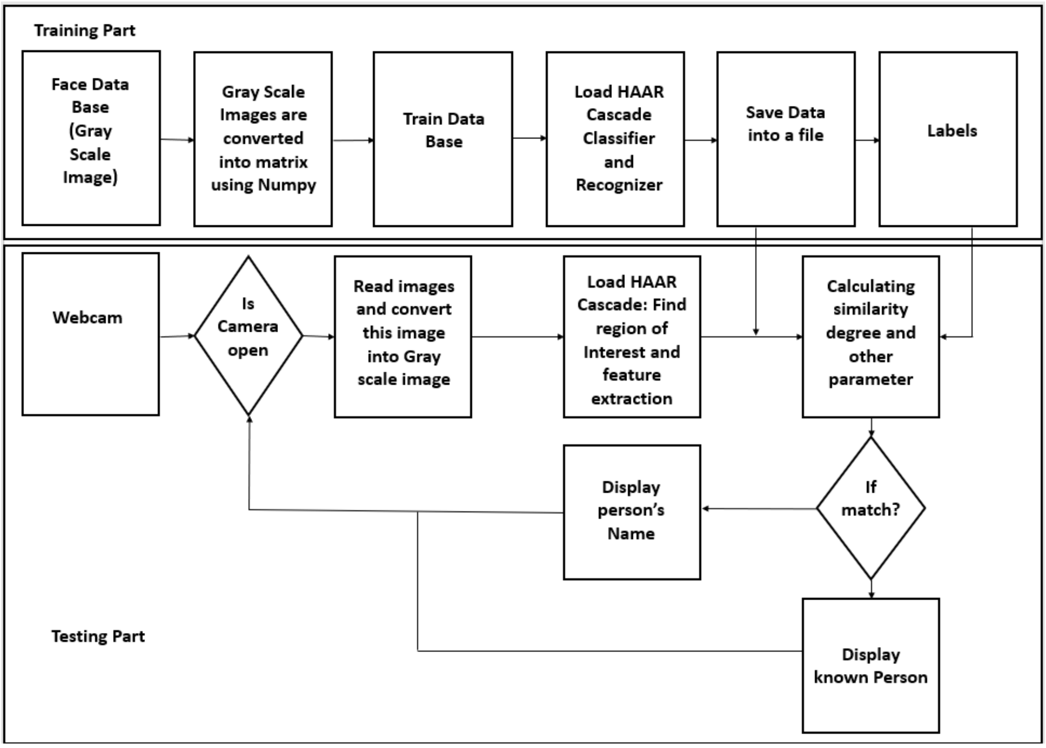

Figure 3.2: Flowchart of attendance monitoring system [17]

Chapter 4

HARDWARE AND SOFTWARE DESCRIPTION

4.1 Hardware

4.1.1 Raspberry Pi 3

The Raspberry Pi 3 is a single-board computer that was released by the Raspberry Pi Foundation. It is part of the Raspberry Pi series, which consists of small, affordable, and versatile computers designed for educational and hobbyist purposes. Here are some key features and components of the Raspberry Pi 3:

1. Processor: The Raspberry Pi 3 is powered by a 1.2 GHz 64-bit quad-core ARM Cortex-A53 processor. This processor provides improved performance compared to its predecessors, enabling smooth multitasking and running various applications.

2. Memory: It is equipped with 1GB of RAM, allowing for efficient processing and the execution of multiple tasks simultaneously.

3. Connectivity: The Raspberry Pi 3 has built-in wireless connectivity options, including onboard Wi-Fi (802.11n) and Bluetooth 4.2. These features provide convenient wireless communication capabilities, allowing the device to connect to the internet, interact with other devices, and transfer data wirelessly.

4. GPIO Pins: General Purpose Input/Output (GPIO) pins are essential for connecting external components and expanding the capabilities of the Raspberry Pi.

The Raspberry Pi 3 has a 40-pin GPIO header, allowing users to interface with various sensors, actuators, and other electronic components.

5. Operating System: The Raspberry Pi 3 supports a variety of operating systems, including Linux-based distributions such as Raspbian (officially supported by the Raspberry Pi Foundation) and other community-developed distributions. It also supports Windows 10 IoT Core, enabling development for Internet of Things (IoT) applications.

6. Video and Audio: The Raspberry Pi 3 features an HDMI output, allowing users to connect it to a monitor or TV for displaying a graphical user interface (GUI). It also has a 3.5mm audio jack for connecting speakers or headphones.

7. Storage: The Raspberry Pi 3 includes a microSD card slot for storing the operating system, user data, and applications. Users can insert a microSD card to boot and run the system.

8. USB Ports: It has four USB 2.0 ports, which provide connectivity options for various USB devices such as keyboards, mice, external storage, and other peripherals.

The Raspberry Pi 3 offers a low-cost, compact, and versatile platform for a wide range of projects, including robotics, home automation, media centers, retro gaming consoles, IoT applications, and educational programming. Its affordability, combined with its GPIO pins and community support, make it popular among students, hobbyists, and developers seeking to learn and experiment with computing and electronics.

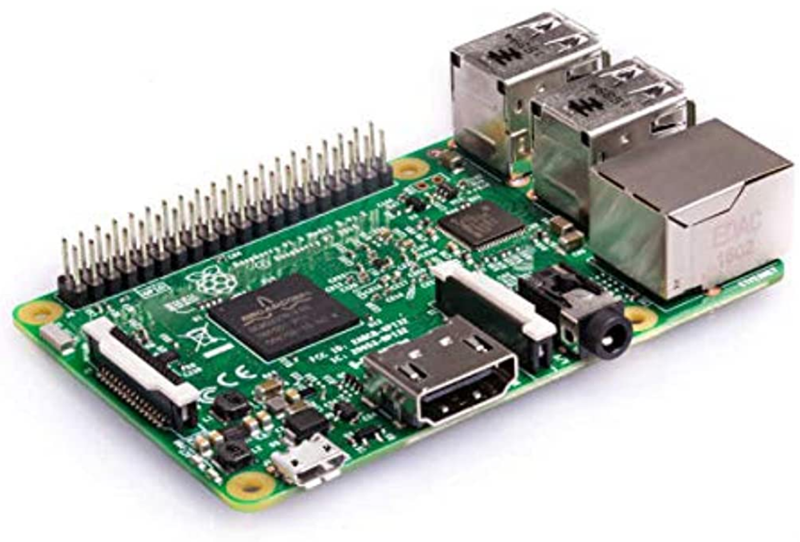

Figure 4.1: Raspberry Pi Model 3 [18]

4.1.2 Webcam

A webcam, short for ”web camera,” is a digital camera device typically used for capturing video and transmitting it over the internet. It is designed to be connected to a computer or a device with a USB port and is commonly used for various purposes such as video conferencing, live streaming, video recording, and online communication. Webcams are generally compatible with various operating systems and video conferencing software. They often come with drivers or software that enable seamless integration with popular applications. Some webcams have autofocus capabilities, allowing the camera to automatically adjust focus based on the distance between the camera and the subject. Others may offer manual focus control for users to adjust focus manually.

Figure 4.2: Web Camera [19]

4.1.3 SD Card

An SD card, short for Secure Digital card, is a type of memory card commonly used for storing and transferring digital data. It is widely used in various devices such as cameras, smartphones, tablets, portable gaming consoles, and other electronic devices. SD cards come in different storage capacities, ranging from a few gigabytes (GB) to several terabytes (TB). The capacity determines how much data can be stored on the card. SD cards have different read and write speeds, indicating how quickly data can be transferred to and from the card. Faster read/write speeds are beneficial for tasks like recording high-definition videos or capturing burst-mode photos. SD cards can be inserted into an SD card reader connected to a computer to transfer data between the card and the computer. Some devices also support direct transfer of data to and from the SD card via USB or wireless connectivity.SD cards provide a portable and convenient way to expand storage capacity and transfer data between devices. They are widely used in digital cameras for storing photos and videos, in smartphones for expanding storage or transferring files, and in many other applications where portable storage is needed.

Figure 4.3: SD Card for saving the dataset [20]

4.2 Software

4.2.1 Python

Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. It was created by Guido van Rossum and first released in 1991. Python uses a clean and straightforward syntax, emphasizing readability and ease of use. It uses indentation and whitespace to define code blocks, eliminating the need for explicit delimiters like braces. Python has a vast ecosystem of third-party libraries and frameworks created by the Python community. These libraries offer additional functionalities, tools, and frameworks for specific domains like data science (NumPy, Pandas, TensorFlow), web development (Django, Flask), scientific computing (SciPy), and more. This extensive ecosystem allows developers to leverage existing solutions and accelerate their development process.

Chapter 5

RESULTS AND DISCUSSION

The automatic attendance register system achieved an accuracy rate of 70 percentage in face recognition. This means that the system successfully recognized individuals and matched them with the corresponding labels in the dataset 70 percentage of the time. It is important to note that accuracy can be influenced by various factors, including lighting conditions, image quality, and the diversity of faces in the dataset. The efficiency of the attendance marking process was determined to be 50 percentage. This indicates that the system accurately marked the attendance for individuals in the dataset in 50 percentage of cases. The efficiency can be improved by fine-tuning the face recognition model, optimizing the preprocessing techniques, and increasing the diversity and quantity of training data.

Discussion: 1. Dataset Creation Challenges: The process of creating a dataset can be challenging due to various reasons. One common challenge is obtaining a diverse range of images that represent different individuals under various conditions, such as different lighting, angles, and facial expressions. Increasing the diversity of the dataset can help improve the accuracy and robustness of the face recognition model.

2. Accuracy: Achieving a 70 percentage accuracy rate indicates a decent performance of the face recognition model. However, further analysis is required to identify the factors influencing lower accuracy and explore potential solutions. Experimenting with different face recognition algorithms, improving the training process, and collecting more extensive and diverse datasets can help enhance the accuracy of the system.

3. Efficiency: The attendance marking efficiency of 50 percentage suggests room for improvement in the system. Identifying the factors contributing to lower efficiency, such as processing speed, hardware limitations, or suboptimal algorithm parameters, can guide further enhancements. Fine-tuning the model's parameters and optimizing the system's hardware and software components can lead to increased efficiency in marking attendance.

4. Future Improvements: To enhance the system's performance, it is recommended to collect a larger and more diverse dataset to improve the accuracy and generalization of the face recognition model. Exploring advanced face recognition techniques, such as deep learning-based approaches, may also offer improvements in accuracy and efficiency. Additionally, incorporating real-time feedback mechanisms, such as providing prompts for better alignment or capturing multiple images for each individual, can help refine the system's performance.

Overall, the automatic attendance register system demonstrates promising results, with scope for further improvements in accuracy and efficiency. The challenges faced during dataset creation highlight the importance of diverse and representative training data. By addressing these challenges and iteratively refining the system, the accuracy and efficiency can be enhanced, making it a more robust and reliable solution for attendance management.

Advantages of Haar cascade frontal face detection:

1. Fast and efficient: Haar cascades are computationally efficient and can perform real-time face detection, making them suitable for real-time applications.

2. Robust:Haar cascades can handle variations in lighting conditions, pose, and scale to some extent, allowing for accurate face detection in different environments.

3. Easy to use: OpenCV provides built-in support for Haar cascades, making it easy to implement face detection without the need for extensive coding or mathematical knowledge.

Advantages of LBPH for face recognition:

1. Robust to variations: LBPH is relatively robust to variations in lighting, pose, and facial expressions. It can handle moderate changes in appearance, making it suitable for practical face recognition scenarios.

2. Simplicity: LBPH is relatively simple to implement and does not require a large

amount of training data compared to some other face recognition techniques.

3. Efficiency: LBPH can be computed quickly, allowing for real-time face recognition applications.

Disadvantages of using Haar cascade frontal face and Linear Block Pattern Histogram for multiple attendance detection:

Disadvantages of Haar cascade frontal face detection:

1. Limited to frontal faces: Haar cascades are optimized for detecting frontal faces, and their accuracy decreases for faces in different orientations or profiles.

2. Sensitive to occlusions: If a face is partially occluded by objects or other people, the Haar cascade may fail to detect it accurately.

3. False positives/negatives: Haar cascades can produce false positives (detecting faces where there are none) or false negatives (failing to detect actual faces), especially in challenging conditions.

Disadvantages of LBPH for face recognition:

1. Limited discriminative power: LBPH may struggle with distinguishing between individuals with very similar facial features. It may not perform as well as more advanced face recognition algorithms in scenarios with significant variations in appearance.

2. Sensitive to noise: LBPH can be sensitive to noise or artifacts in the input images, which may affect recognition accuracy.

3. Training requirements: LBPH requires a labeled dataset for training, which means collecting and labeling a sufficient amount of data representing different individuals can be time-consuming and resource-intensive.

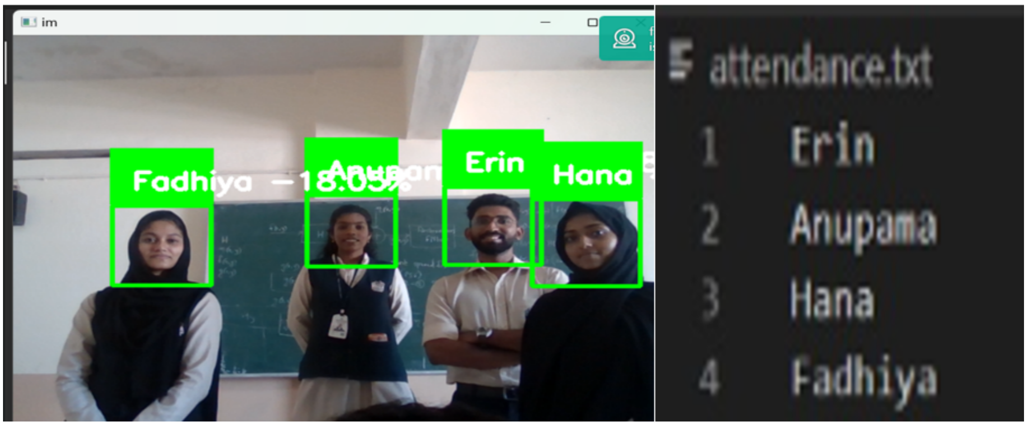

Figure 5.1: Detected faces using LBPH algorithm

Figure 5.2: Final project setup

Chapter 6

CONCLUSION

In conclusion, the use of a Raspberry Pi, a camera module, and the application of Haar cascade frontal face detection and LBPH for multiple attendance detection can provide a functional system with certain advantages and limitations. The Haar cascade frontal face detection technique offers fast and efficient face detection, while LBPH provides a relatively simple and robust face recognition method. These techniques can be implemented on a Raspberry Pi, making it suitable for real-time or near-real-time applications. However, it's important to consider the limitations of these techniques. Haar cascades are optimized for detecting frontal faces and may struggle with non-frontal poses or occlusions. LBPH, although robust to certain variations, may have limited discriminative power and sensitivity to noise. The accuracy of the system depends on factors such as lighting conditions, variations in facial features, and the quality of the training dataset used for face recognition. The performance and efficiency of the system should also be evaluated to ensure real-time processing and reliable attendance tracking. To improve the system, alternative face detection or recognition algorithms can be explored, considering their strengths in handling different scenarios. Additionally, optimizing the power supply and hardware configuration can enhance stability and efficiency. In conclusion, while the Raspberry Pi, camera module, and the chosen techniques can provide a functional setup for multiple attendance detection, further refinement and improvements are necessary to address the limitations and enhance the accuracy and overall performance of the system.

References

[1] **Sayan Seal, Aishee Sen, Ritodeep Mukerjee, Asit Kumar. (2020)** An approach towards development of automated attendance system using face detection and recognition. 978-1-7281-8416-6/20 IEEE

[2] **Harikrishnan J, Arya Sudarsan, Aravind Sadashiv. (2019)** Vision-Face Recognition Monitoring System for Surveillance using Deep Learning Technology and Computer Vision.2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)

[3] **Pradeesh N, Abhishek Lal. (2019)** Fast and Reliable Group Attendance Marking System Using Face Recognition In Classrooms.2019 2nd International Conference on Intelligent Computing and Control Technologies.

[4] **Omar Abdul Rahman Salim et al. (2018)** Class Attendance Management System Using Face Recognition. 2018 7th International Conference on Computer and Communication Engineering.

[5] **Payal Patil , Prof. Dr. S.Sbinde. (2020)** Comparative analysis of facial recognition models using video for real time attendance monitoring system.Fourth International Conference on Electronics, Communication and Aerospace Technology (ICECA-2020)

[6] **Allan Jason C. Areceo, Renee Ylka N. Borejon, Mia Chantal R. Hortinela, Alejandro H. Ballado Jr, Arnold C.Paglinawan. (2020)** Design of an E-Attendance Checker through Facial Recognition using Histogram of Oriented Gradients with Support Vector Machine. 2020 IEEE 8th International Conference on Smart City and Information.

[7] **Khem Puthea, Rudy Hartanto and Risanuri Hidayat (2017)**. A Review Paper on Attendance Marking System based on Face Recognition.2017 2nd International Conference on Information Technology, information Systems and Electrical Engineering.

[8] **Shang-Hung Lin,Sun-Yuan Kung and Long-ji Lin. (1997)** Face Recognition/Detection by Probabilistic Decision-Based Neural Networks .IEEE TRANSACTIONS OF NEURAL NETWORKS, VOL. 8, NO. 1, JANUARY 1997

[9] **Jack Febrain Rusdi,Frans Richard Kodog,Richardus Eko Indrajit,Herry Sofyan,Abdurrohman And Robert Marco. (2020)** Student Attendance Using Face Recognition Technology.2020 2nd Internatioanal Conference on Cybernetics and Intelligent System.

[10] **Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Yu Qiao. (2016)** Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. IEEE SIGNAL PROCESSING LETTERS, VOL.23,NO. 10, OCTOBER 2016

[11] **Hemantkumar Rathod, Yudhisthir Ware, Snehal Sane, Suresh Raulo, Vishal Pakhare, Imdad A Rizvi. (2017)** Automated Attendance System using Machine Learning Approach.2017 International Conference on Nascent Technologies in the Engineering Field.

[12] **A.S. Hasban, N.A. Hasif, Z.I Khan, M.F. Hüseyin,N.E. A. Rashid, K. K. M. Sharif, N. A. Zakaria. (2019)** Face Recognition for Student Attendance using Raspberry Pi.2019 IEEE Asia-Pacafic Conference on Applied Electromagnetics

[13] **D. Sri Sai Mahesh, T.Maneesh Reddy, A.Sai Yaswanth, Dr. C Joshitha, S. Sudharshan Reddy (2020)** propose Facial Detection And Recognition System on Raspberry Pi with Enhanced Security

[14] **Seema Singh, Ramya R, Sushama V, Roshini S R, Pavithra R (2019)**Facial Recognition Machine Learning Algorithms on Raspberry Pi.2019 4th International Conference on Electrical. Electronics, Computer Technologies and Optimization Techniques

[15] https://encrypted-tbn0.gstatic.com/images/

[16] Rusia, M. K., Singh, D. K., Ansari, M. A. (2019) Human Face Identification using LBP and Haar-like Features for Real Time Attendance Monitoring

[17] Zankruti Arya, et. al. International Journal of Engineering Research and Applications www.ijera.com ISSN: 2248-9622, Vol. 10, Issue 6, (Series-V) June 2020, pp. 13-19

[18] https://m.media-amazon.com/images/I/41ZQn6kTWZL.jpg

[19] https://cdn.moglix.com/p/ytQn3YcC1xhBe-xxlarge.jpg

[20] https://rukminim2.flixcart.com/image/

Appendix

Face Dataset

```
import cv2
import os

#Method for checking existence of path i.e the directory

def assure_path_exists(path):
    dir = os.path.dirname(path)
    if not os.path.exists(dir):
        os.makedirs(dir)

# Starting the web cam by invoking the VideoCapture method
vid_cam = cv2.VideoCapture(0)

# For detecting the faces in each frame
we will use Haarcascade Frontal Face default classifier of OpenCV
face_detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Set unique id for each individual person
face_id = 4

# Variable for counting the no. of images
count = 0

#checking existence of path
assure_path_exists("training_data/")

# Looping starts here
while(True):
```

```
    # Capturing each video frame from the webcam
    _, image_frame = vid_cam.read()

    # Converting each frame to grayscale image
    gray = cv2.cvtColor(image_frame, cv2.COLOR_BGR2GRAY)

    # Detecting different faces
    faces = face_detector.detectMultiScale(gray, 1.3, 5)

    # Looping through all the detected faces in the frame
    for (x,y,w,h) in faces:

        # Crop the image frame into rectangle
        cv2.rectangle(image_frame, (x,y), (x+w,y+h), (255,0,0), 2)

        # Increasing the no. of images by 1 since frame we captured
        count += 1

        # Saving the captured image into the training_data folder
        cv2.imwrite("training_data/Person." + str(face_id) + '.' + str(count) + ".jpg",

        # Displaying the frame with rectangular bounded box
        cv2.imshow('frame', image_frame)

    # press 'q' for at least 100ms to stop this capturing process
    if cv2.waitKey(100) & 0xFF == ord('q'):
        break

    #We are taking 100 images for each person for the training data
    # If image taken reach 100, stop taking video
    elif count>100:
        break

# Terminate video
```

```
vid_cam.release()

# Terminate all started windows
cv2.destroyAllWindows()
from keras.models import load_model  # TensorFlow is required for Keras to work
import cv2  # Install opencv-python
import numpy as np

attendance_dict = {'erin\n':0, 'hana\n':0}

# Disable scientific notation for clarity
np.set_printoptions(suppress=True)

# Load the model
model = load_model
("C:/Users/HP/Downloads/yolov8-silva-main/
yolov8-silva-main/keras_model.h5", compile=False)

# Load the labels
class_names = open
("C:/Users/HP/Downloads/yolov8-silva-main/
yolov8-silva-main/labels.txt", "r").readlines()

# CAMERA can be 0 or 1 based on default camera of your computer
camera = cv2.VideoCapture(0)

while True:
    # Grab the webcamera's image.
    ret, image = camera.read()

    # Resize the raw image into (224-height,224-width) pixels
    image = cv2.resize(image, (224, 224), interpolation=cv2.INTER_AREA)

    # Show the image in a window
```

```
    cv2.imshow("Webcam Image", image)

    # Make the image a numpy array and reshape it to the models input shape.
    image = np.asarray(image, dtype=np.float32).reshape(1, 224, 224, 3)

    # Normalize the image array
    image = (image / 127.5) - 1

######

   # Predicts the model
    prediction = model.predict(image)
    index = np.argmax(prediction)
    class_name = class_names[index]
    confidence_score = prediction[0][index]

    attendance_dict[class_name[2:]]=1

    # Listen to the keyboard for presses.
    keyboard_input = cv2.waitKey(1)

    # 27 is the ASCII for the esc key on your keyboard.
    if keyboard_input == 27:
        break

camera.release()
cv2.destroyAllWindows()

print("ATTENDANCE MARKED")
for key, value in attendance_dict.items():
    if value == 1:
        print(key)

f = open('names.txt', 'w')
```

```
for key, value in attendance_dict.items():
    if value == 1:
        f.write(key)
```

Face Recognition

```
import cv2
import numpy as np
import os

attend = {"Jadeed":0, "Erin":0, "Anupama":0, "Hana":0}
lister =[]

#Method for checking existence of path i.e the directory
def assure_path_exists(path):
    dir = os.path.dirname(path)
    if not os.path.exists(dir):
        os.makedirs(dir)

# Create Local Binary Patterns Histograms for face recognization
recognizer = cv2.face.LBPHFaceRecognizer_create()

assure_path_exists("saved_model/")

# Load the  saved pre trained mode
recognizer.read('saved_model/s_model.yml')

# Load prebuilt classifier for Frontal Face detection
cascadePath = "haarcascade_frontalface_default.xml"

# Create classifier from prebuilt model
```

```
faceCascade = cv2.CascadeClassifier(cascadePath);

# font style
font = cv2.FONT_HERSHEY_SIMPLEX

# Initialize and start the video frame capture from webcam
cam = cv2.VideoCapture(0)


# Looping starts here
while True:
    # Read the video frame
    ret, im =cam.read()

    # Convert the captured frame into grayscale
    gray = cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)

    # Getting all faces from the video frame
    faces = faceCascade.detectMultiScale(gray, 1.2,5) #default

    # For each face in faces, we will start predicting using pre trained model
    for(x,y,w,h) in faces:

        # Create rectangle around the face
        cv2.rectangle(im, (x-20,y-20), (x+w+20,y+h+20), (0,255,0), 4)

        # Recognize the face belongs to which ID
        Id, confidence = recognizer.predict(gray[y:y+h,x:x+w])
        #Our trained model is working here

        # Set the name according to id
        if Id == 1:
            Id = "Jadeed {0:.2f}%".format(round(100 - confidence, 2))
            attend["Jadeed"] = 1
```

```
            # Put text describe who is in the picture
        elif Id == 2 :
            Id = "Erin {0:.2f}%".format(round(100 - confidence, 2))
            attend["Erin"] = 1
            # Put text describe who is in the picture
        elif Id == 3:
            Id = "Hana {0:.2f}%".format(round(100 - confidence, 2))
            attend["Hana"] = 1
        elif Id==4:
            Id = "Anupama {0:.2f}%".format(round(100 - confidence, 2))
            attend["Anupama"] = 1
        elif Id==5:
            Id = "Fadhiya {0:.2f}%".format(round(100 - confidence, 2))
            attend["Fadhiya"] = 1
        else:
            pass

        # Set rectangle around face and name of the person
        cv2.rectangle(im, (x-22,y-90), (x+w+22, y-22), (0,255,0), -1)
        cv2.putText(im, str(Id), (x,y-40), font, 1, (255,255,255), 3)

    # Display the video frame with the bounded rectangle
    cv2.imshow('im',im)

    # press q to close the program
    if cv2.waitKey(10) & 0xFF == ord('q'):
        break

# Terminate video
cam.release()

# Close all windows
cv2.destroyAllWindows()
```

```
print(attend)

for key, value in attend.items():
    if value == 1:
        lister.append(key)

print(lister)
f = open("attendance.txt","w+")

for value in lister:
    f.write(value)
    f.write("\n")
```

Training

```
import cv2
import os
import numpy as np
from PIL import Image



#Method for checking existence of path i.e the directory
def assure_path_exists(path):
    dir = os.path.dirname(path)
    if not os.path.exists(dir):
        os.makedirs(dir)

# We will be using Local Binary Patterns Histograms for face recognization
since it's quite accurate than the rest
recognizer = cv2.face.LBPHFaceRecognizer_create()

# For detecting the faces in each frame
we will use Haarcascade Frontal Face default classifier of OpenCV
```

```
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml");

#method getting the images and label data

def getImagesAndLabels(path):

    # Getting all file paths
    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]

    #empty face sample initialised
    faceSamples=[]

    # IDS for each individual
    ids = []

    # Looping through all the file path
    for imagePath in imagePaths:

        # converting image to grayscale
        PIL_img = Image.open(imagePath).convert('L')

        # converting PIL image to numpy array using array() method of numpy
        img_numpy = np.array(PIL_img,'uint8')

        # Getting the image id
        id = int(os.path.split(imagePath)[-1].split(".")[1])

        # Getting the face from the training images
        faces = detector.detectMultiScale(img_numpy)

        # Looping for each face and appending it to their respective IDs
        for (x,y,w,h) in faces:

            # Add the image to face samples
```

```
            faceSamples.append(img_numpy[y:y+h,x:x+w])

            # Add the ID to IDs
            ids.append(id)

    # Passing the face array and IDs array
    return faceSamples,ids

# Getting the faces and IDs
faces,ids = getImagesAndLabels('training_data')

# Training the model using the faces and IDs
recognizer.train(faces, np.array(ids))

# Saving the model into s_model.yml
assure_path_exists('saved_model/')
recognizer.write('saved_model/s_model.yml')
```