A PROJECT STAGE-II REPORT

on

FACE MASK DETECTION

A Dissertation submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND INSTRUMENTATION ENGINEERING

By

D. AKHILESH	**19BD1A1003**
J. HARSHA	**19BD1A1020**
K. ROHITH REDDY	**19BD1A1023**
K. JAYANTH ADITHYA	**19BD1A1029**

Under the esteemed guidance of

MR. L. AMARENDER REDDY

Assistant professor

Department of EIE

Department of Electronics and Instrumentation Engineering

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY

Approved by A.I.C.T.E Affiliated to JNTU, Hyderabad

NARAYANGUDA, HYDERABAD, TELANGANA-500029

2022-2023

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
Approved by A.I.C.T.E, Affiliated to JNTU, Hyderabad
3-5-1026, Narayanaguda, Hyderabad-500029

CERTIFICATE

This is to certify that the project entitled "**FACE MASK DETECTION**" being submitted by

D. AKHILESH	**19BD1A1003**
J. HARSHA	**19BD1A1020**
K. ROHITH REDDY	**19BD1A1023**
K. JAYANTH ADITHYA	**19BD1A1029**

students of **Keshav Memorial Institute of Technology, JNTU** in partial fulfilment of the requirements for the award of **"Bachelor of Technology"** in **"Electronics and Instrumentation Engineering"** as a specialization is a record of bonafide work carried out by them under my guidance and supervision in the academic year 2022-2023.

INTERNAL GUIDE	**HEAD OF THE DEPARTMENT**
MR. L. AMARENDER REDDY	**MRS. SWAPNA BANDARI**
Assistant Professor	**Assistant Professor & H.O.D.**
Department of EIE	**Department of EIE**

Submitted for the project viva voice examination held on_____

INTERNAL EXAMINAR **EXTERNAL EXAMINAR**

ACKNOWLEDGEMENT

We take this opportunity to thank all the people who have rendered their full support to our project work.

We render our thanks to **Dr. Maheshwar Dutta**, B.E, M.Tech, Ph.D., principal who encouraged us to do the project.

We are grateful to **Mr. Neil Gogte**, Director for facilitating all the amenities required for carrying out this project.

We express our sincere gratitude to **Mr. S. Nitin**, Director and **Dr. Jayaprakash**, Dean for providing an excellent environment in the college.

We are also thankful to **MRS. SWAPNA BANDARI**, Head of Department for providing us with the time and amenities to make the project a success within the given schedule.

We are also thankful to **MR. L. AMARENDER REDDY** as our internal guide for his valuable guidance and encouragement were given to us throughout the project work.

We would like to thank the entire **EIE Department** faculty who helped us directly and indirectly in the completion of project.

DECLARATION

We,

Hereby declare that the project report entitled "**FACE MASK DETECTION**" done in partial fulfilment for the award of the Degree of Bachelor of Technology in Electronics and Instrumentation Engineering affiliated to Jawaharlal Nehru Technological University, Hyderabad. We have not submitted this report to any other University or organization for award of any other degree.

STUDENT NAME	**ROLL NUMBER**
D. AKHILESH	**19BD1A1003**
J. HARSHA	**19BD1A1020**
K. ROHITH REDDY	**19BD1A1023**
K. JAYANTH ADITHYA	**19BD1A1029**

INDEX

LIST OF FIGURES

LIST OF TABLES

Chapter 1
INTRODUCTION

The spread of COVID- 19 pandemic disease has created the most crucial global health crisis of the world that has deep impact on humanity and the way the user perceives our world and our everyday lives. A novel corona virus has resulted in person-to-person transmission but as far as we move, the transmission of the novel corona causing corona virus disease 2019 (COVID-19) can also be a form a symptomatic carrier with no COVID-19 symptoms. Till now there is no report about any clinically approved antiviral medicines or vaccine that are effective against COVID-19 virus COVID-19. It has spread rapidly across the world bringing massive health, economic, environmental and social challenges to the entire human population. The trend of wearing face masks in public is rising due to the COVID- 19 corona virus epidemic all over the world. Before Covid-19, People used to wear masks to protect their health from air pollution. While other people are self-conscious about their looks, they hide their emotions from the public by hiding their faces.

Scientists proofed that wearing face masks works on impeding COVID-19 transmission. COVID-19 (known as corona virus) is the latest epidemic virus that hit the human health in the last century. In 2020, the rapid spreading of COVID-19 has forced the World Health Organization to declare COVID- 19 as a global pandemic. More than five million cases were infected by COVID-19 in less than 6 months across 188 countries. The virus spreads through close contact and in crowded and overcrowded areas.

The monitoring process involves the finding of anyone who isn't sporting a face mask. To monitor that people are following this basic safety principle, a strategy should be developed. A face mask detector system can be implemented to check this. Face mask detection means to identify whether a person is wearing a mask or not. The first step to recognize the presence of a mask on the face is to detect the face, which makes the strategy divided into two parts: to detect faces and to detect masks on those faces. Face detection is one of the applications of object detection and can be used in many areas like security, biometrics, law enforcement and more. There are many detector systems developed around the world and being implemented. However, all this science needs optimization; a better, more precise detector, because the world cannot afford any more increase in corona cases.

Here the user introduces a mask face detection model that's supported Artificial intelligence using image process techniques. The planned model may be detecting the mask with image and real time detection people wearing mask or not wearing a mask. The user has introduced a comparison between them to seek out the foremost appropriate algorithm program that achieved the very best accuracy.

The novel coronavirus covid-19 had brought a new normal life. India is struggling to get out of this virus attack and the government implemented lockdown for the long way. Lockdown placed a pressure on the global economy. So the government gave relaxations in lockdown. Declared by the WHO that a potential speech by maintaining distance and wearing a mask is necessary. The biggest support that the government needs after relaxation is social distancing and wearing of masks by the people. But many people are getting out without a face mask this may increase the spread of covid-19. Economic Times India has stated that " Survey Shows that 90 percent Indians are aware, but only 44 percent wearing a mask ". This survey clearly points that people are aware but they are not wearing the mask due to some discomfort in wearing and carelessness. This may result in the easy spreading of covid-19 in public places.

The world health organisation has clearly stated that until vaccines are found the wearing of masks and social distancing are key tools to reduce spread of virus. So, it is important to make people wear masks in public places. In densely populated regions it is difficult to find the persons not wearing the face mask and warn them. Hence, we are using image processing techniques for identification of persons wearing and not wearing face masks. In real time images are collected from the camera and it is processed in Raspberry Pi embedded development kit. The real time images from the camera are compared with the trained dataset and detection of wearing or 1 not wearing a mask is done. The trained dataset is made by using machine learning technique which is the deciding factor of the result. The algorithm created by means of using a trained dataset will find the persons with and without wearing face masks.

The Internet of Things (IOTs) can be used for connecting objects like smartphones, Internet TVs, laptops, computers, sensors and actuators to the Internet where the devices are linked together to enable new forms of communication between things and people, and between things themselves. Intimation messages are sent to authority persons by means of using IOT.

Chapter 2
LITERATURE SURVEY

This chapter is set out to provide the overall analysis of various research approaches deployed and several options and achievements attained by different writers on our specified field of study and applications. The content here will take account of concepts of various researchers that have already made extensive research on the tools and applications with their collective efforts which has backed to the growth of technology.

J. Barabas et al [1] proposed a system for automated testing of temperature and hygienic standards. An algorithm is used for mask detection utilizing both neural networks (NN) and feature vector description based on a histogram of oriented gradients (HOG) approach. The device is trained with several image sets, and these are stored in Caffe framework format and processed in the OpenCV Deep Neural Network module. Proper temperature is measured using the MLX90614 infrared thermometer. Raspberry Pi processor is the core of the whole system which enables real-time processing of all image and sensor data.

Sammy et al [2] suggested a deep learning approach to detect facemask and social distance. The image data sets are stored in JPEG format. Classification of images is done using open-source software using python and OpenCV. A CNN model is created using Tensor flow and Keras module along with the VGG-16 network model. The system is also designed to give the count of people violating the physical distancing.

Arjya Das et al [3] introduced a technique to identify hybrid facial masks using basic machine learning packages. The expected system acts as the surveillance task performer which detects mask even in motion. Values of parameters are optimized using the Sequential CNN model to detect the presence of facemasks accurately without over-fitting. The incorporated ML packages include TensorFlow, Keras, OpenCV are used for reshaping, compile the overall model along with resizing and colour conversion of data images.

Suresh K et al [4]. This paper approaches a simplified way to achieve facemask detection. The mask is extracted and fed as an input into a convolutional neural network. The real-time automated detection has been done by MobileNet and OpenCV. The datasets are divided into categories that provide an advantage to improve variants.

Anirudh Lodh et al [5] designed a system that is used for face recognition and person identification. This proposed method uses MobileNet_V2 neural networks as a classification algorithm. With increased accuracy and less time complexity, the proposed system detects multiple faces from all angles. For training the system, a Keras/TensorFlow library named MobileNet_V2 is used. In addition to this, mails will be sent to the people not wearing masks through python scripts.

Samuel Ady Sanjaya et al [6] proposed an effective solution for face mask recognition using machine learning. This system is used to build a face recognition model which collects, pre-processes, and testing the data. In this paper, the model performs iterations on the testings until it reaches maximum accuracy and this model is evaluated as the next step. For pre-processing input, MobileNetV2 is used.

Mohamed Loey et al [7] gave three face mask datasets in their paper which are selected for investigation. The proposed model performs feature extraction using Resnet50, classification of face masks using decision trees, Support Vector Machine (SVM), and ensemble algorithm. The datasets are Real-World Masked Face Dataset (RMFD), the Simulated Masked Face Dataset (SMFD), and the Labelled Faces in the Wild (LFW). These classifiers can achieve up to 100% testing accuracy.

Md. Rafiuzzaman Bhuiyan et al [8] has addressed a system based on deep learning to classify facial masks using the YOLOv3 algorithm. The image acquisition followed by annotation is used to train the system. Input is an image passed into the YOLOv3 model. This object detector divides the input into a grid that analyses the target object's features.

Rehman et al [9] proposed a system that restrict the growth of COVID-19 by finding out people who are not wearing any facial mask in a smart city network where all the public places are monitored with Closed-Circuit Television (CCTV) cameras. While a person without a mask is detected, the corresponding authority is informed through the city network. A deep learning architecture is trained on a dataset that consists of images of people with and without masks collected from various sources. The trained architecture achieved 98.7% accuracy on distinguishing people with and without a facial mask for previously unseen test data.

Deore et al [10] use a simple HOG feature extractor along with the Viola-Jones algorithm. It is a combination of Haar feature selection, integral image creation, Adaboost training, and cascading classifiers. It classifies first the mouth, nose, and ears. If a person's mouth is not detected it is assumed that he is wearing a mask.

A face mask detecting model named Retina Face Mask combining with a cross- class object removal algorithm is proposed by Jiang et al. [11]. The developed model includes one stage detector consisting feature pyramid network that results in slightly higher precision and recall than the baseline result. For reducing the shortage of datasets, they have applied transfer learning, a well-known deep learning technique.

COVID-19 pandemic caused by novel coronavirus is continuously spreading until now all over the world. The impact of COVID-19 has been fallen on almost all sectors of development. The healthcare system is going through a crisis. Many precautionary measures have been taken to reduce the spread of this disease where wearing a mask is one of them. In this paper, we propose a system that restrict the growth of COVID-19 by finding out people who are not wearing any facial mask in a smart city network where all the public places are monitored with Closed-Circuit Television (CCTV) cameras. While a person without a mask is detected, the corresponding authority is informed through the city network. A deep learning architecture is trained on a dataset that consists of images of people with and without masks collected from various sources. The trained architecture achieved 98.7% accuracy on distinguishing people with and without a facial mask for previously unseen test data. It is hoped that our study would be a useful tool to reduce the spread of this communicable disease for many countries in the world.

Recognition from faces is a popular and significant technology in recent years. Face alterations and the presence of different masks make it too much challenging. In the real-world, when a person is uncooperative with the systems such as in video surveillance then masking is further common scenarios. For these masks, current face recognition performance degrades. An abundant number of researches work has been performed for recognizing faces under different conditions like changing pose or illumination, degraded images, etc. Still, difficulties created by masks are usually disregarded. The primary concern to this work is about facial masks, and especially to enhance the recognition accuracy of different masked faces. A feasible approach has been proposed that consists of first detecting the facial regions.

Table 1: Literature Survey

SR NO	TITLE	ADVANTAGES	DRAWBACKS
1.	Performance Evaluation of Intelligent Face Mask Detection System with various Deep Learning Classifiers.	The system is evaluated with different classifiers. Different classifiers like MobileNetV2, RESNET50, VGG16, each of them was compared with a few Optimizers like ADAM, ADAGRAD, SGD to yield the maximum accuracy. ADAM gave the maximum accuracy.	The best framework might be executed alongside interfacing with caution and alarming frameworks soon. This framework might be incorporated with a framework which can coordinate with a framework actualizing social distancing which can make it a healthy framework that can welcome emotional effect on the spread.
2.	The Covid-19 Facemask Detection by using Deep Learning and Computer Vision.	The system comprises of MobileNet as the spine which can be very well utilized for high and low calculation situations. In order to extract more robust features, learning is used to gain weights from a similar task face detection, which is trained on large datasets. The proposed method accomplishes state-of-art results on public face mask dataset. By the advancement of face mask detection, it can be detected whether a person is wearing face mask or not and permit their entry would be of great help to the society.	In future studies, a more extensive facemask detection and wearing dataset including images and videos will be collected and labelled with more details in order to improve the performance.

3.	A Cascade Framework for face mask detection.	A deep-learning based algorithm for the face mask detection is proposed. This is an algorithm based on a recently planned CNN course the structure comprises of about three CNNs. Additionally, another dataset called "masked face dataset" is proposed which have 160 images for training and 40 images for testing. To defeat the overfitting issue due to lack of training samples, the model is pre-trained with wider face dataset, and finetuned them with masked face training set. This masked face detection algorithm is assessed on the masked face testing set, it is set to achieve satisfactory results.	The system can be used in CCTV footages to identify whether a person is wearing a mask correctly so that he does not pose any hazard to others.
4.	A Multi-Stage CNN Architecture for Face Mask Detection.	A two-stage face mask detector was introduced. First stage uses pre-trained RetinaFace model for face detection, after comparing its performance with Dlib and MTCNN. Second stage uses NASNetMobile based model for classifying faces as "masked" or "unmasked". Besides, centroid tracking is used to improve performance on video streams.	At present, the model gives 5 FPS inference speed on a CPU. We plan to improve this up to 15 FPS, making our model deployable for CCTV cameras, without need of a GPU. Stage 1 and stage 2 models can be easily replaced with improved models that would give better accuracy and lower latency.
5.	Using a deep learning	In this work, methodology for	Mass screening is little difficult in

	framework to detect face masks from the video footages of a camera or any video recording devices.	recognizing face masks from videos is proposed. A profoundly successful face identification model is utilized for getting facial pictures and signals. A particular facial classifier is constructed utilizing deep learning for the errand of deciding the presence of a face mask in the facial pictures distinguished. The subsequent methodology is strong and is assessed on a custom dataset got for this work. The proposed approach was discovered to be successful as it depicted high exactness, review, and precision esteems on the picked dataset which contained videos with fluctuating occlusions and facial angles.	crowded places like railway stations, bus stops, streets, schools, colleges, etc. so the system yielding better accuracy can be created.
6.	A convolutional neural network (CNN) to detect face masks using tensor flow and keras.	A deep convolutional neural network (CNN) is utilized to extricate highlights from input pictures. Keras is utilized for actualizing CNN additionally Dlib and OpenCV for adjusting faces on information pictures. Face acknowledgment execution is assessed utilizing a dataset.	Facial acknowledgment is a powerful apparatus that can help law makers perceive lawbreakers and software organizations are utilizing the innovation to help clients access their innovation. This is an innovation that can be additionally evolved to be utilized in different avenues like ATMs, getting to private records, or other sensitive materials. This can make

			other safety efforts, for example, passwords and keys obsolete. Another way that innovators are looking forward to execute facial acknowledgment inside subways and other transportation outlets. They are hoping to use this innovation to use faces as credit cards to pay for our transportation charge.
7.	A Review on Face Mask Detection using (CNN)	By the improvement of AI and image processing analysis present strategies for mask detection. By utilizing image processing analysis and AI technique is utilized for finding out mask detection. Face mask identification can be done through different strategies. Essentially convolutional neural network technique is utilized quickly. The precision and decision making are exceptionally high in CNN contrasted with others.	Facial mask detection and non-masked face detection accuracy provided high variations.
8.	An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network	In this work, a framework is proposed that confine the development of Coronavirus by discovering individuals who are not wearing any facial mask in a smart city network where all the	The created framework faces challenges in arranging faces covered by hands since it nearly resembles the individual wearing a mask and while any individual without a face mask is going on

		public spots are checked with CCTV cameras. While an individual without a mask is identified, the corresponding authority is informed through the city network. A deep learning design is prepared on a dataset that comprises of pictures of individuals with and without masks gathered from different sources. The trained architecture accomplished 98.7% precision on recognizing individuals with and without a facial mask for previously unseen test data.	any vehicle, the framework can't find that individual accurately. For in a thickly populated zone, recognizing the face of every individual is troublesome. For this sort of situation, recognizing individuals without face mask would be very hard for the proposed framework. To get the best result out of this framework, the city should have an enormous number of CCTV cameras to screen the entire city as well as dedicated manpower to uphold appropriate laws on the violators. Since the data about the violator is sent through SMS, the framework fails when there is an issue in the network.
9.	Face mask Recognition Dataset and Application	It is urgent to improve the acknowledgment execution of the current face recognition innovation on the masked faces. Most current progressed face acknowledgment approaches are planned dependent on deep learning, which rely upon an enormous face sample. To this end, this work proposes three sorts of masked face datasets, including (MFDD), (RMFRD) and Simulated Masked Face Recognition Dataset (SMFRD).	MFDD, RMFRD and SMFRD datasets are built, and built up a state-of-the-art algorithm that is very dependent on these datasets. The calculation will serve the utilizations of contactless face authentication in local area access, campus management, and various enterprise resumption scenarios. This exploration has contributed logical and innovative capacity to the avoidance and control of Covid epidemics and the resumption of creation in industry. Moreover,

		Among them, to the best of our knowledge, RMFRD is as of now the world's biggest real-world masked face dataset and these datasets are freely accessible to industry and the academia, based on which different applications on masked faces can be created.	because of the continuous event of haze weather, individuals will frequently wear covers, and the requirement for face recognition with mask will continue for quite a while.
10.	Fighting against Covid: A novel deep learning model based on YOLO-V2 with ResNet-50 for medical face mask detection	The target of this paper is to comment on and confine the clinical face mask objects and all things are considered as pictures. Wearing a clinical face mask in open territories and public places, ensure individuals from COVID-19 transmission among them. The accomplished results	As a further study, it is intended to distinguish a kind of masked faces in picture and video based on deep learning models.

Chapter 3
PROPOSED SYSTEM

3.1 Block Diagram

With the assistance of Computer Vision and PI Raspberry, attempts were made in the proposed system to check individuals without and with masks automatically. This module detects the person's face, determines whether the person wears a mask or is not wearing a mask, and alert the person if the mask is not used. The emphasis is on how to find the person on an image/video stream with face masks by using the OpenCV, Tensor Flow, Keras and PyTorch library, using machine vision and a deeper learning algorithm.

Approach

1. To Train Deeper using learning model
2. To apply mask detector into pictures/ live video stream.

The proposed system takes input which is person’s face, processes it and determines whether the person is wearing mask or not. After running the code, video stream will start and a square box will be drawn over the face. This system monitors continuously, and when a person is identified without a mask then the audible buzzer activates. If mask detected, then system checks temperature. If temperature is within the normal range, the access for the automatic door will be granted. Otherwise, the audible buzzer actives which means the access is denied.

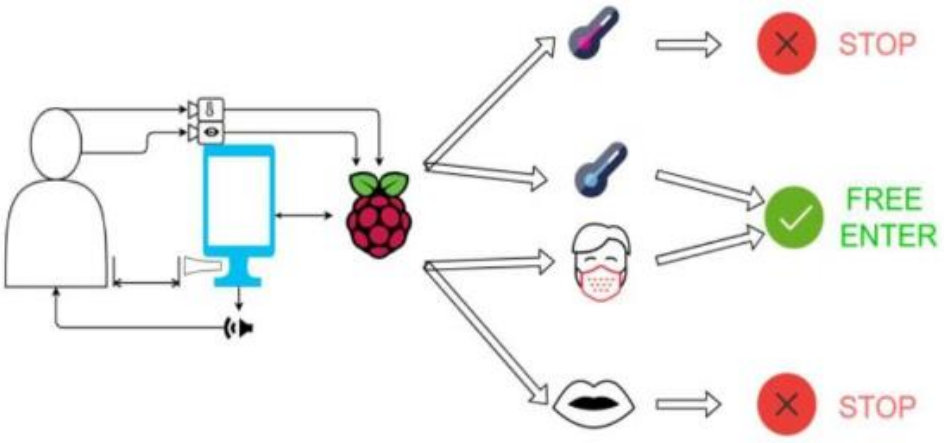

Figure 1: Mask Detection and Temperature checking mode

The proposed system has 3 blocks. First block is for mask detection which includes webcam. Second block is for temperature measurement which includes MLX90614 sensor

and IR sensor. Third block is output, this includes buzzer, LCD module and DC motor. Figures representing prototype model are shown below.

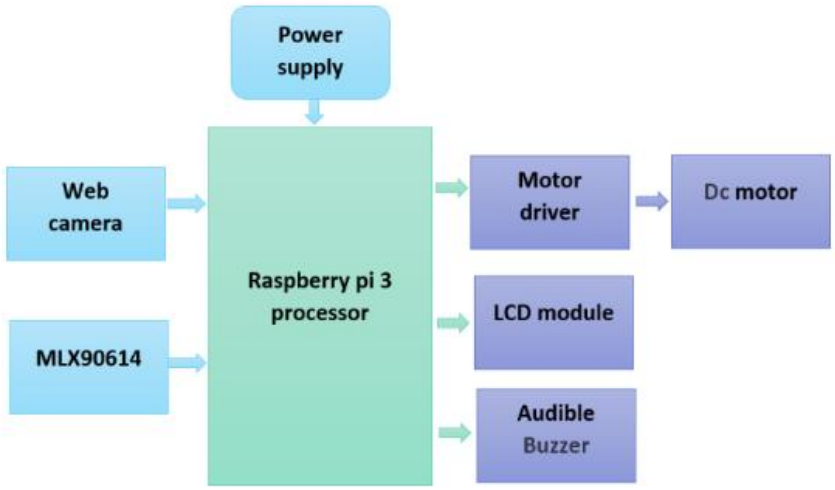

Figure 2: Block Diagram

This paper explains an efficient system called as Face mask detection alert system using raspberry Pi which detects whether the person have worn a face mask or not. The proposed structure of the face mask detection and alert system performs following tasks:

A) Face Detection (using Pi camera input).

B) Mask Detection (if person has worn face mask or not).

We perform face detection using CNN i.e., a structure of SSD (Single Shot Detector) and another model called as MobileNetV2 to detect people in a video frame. We perform this face mask detection system on a Raspberry Pi 4. The proposed algorithm for face mask detection system consists of: pre-processing, training the CNN, face mask detection. The dataset we are using consists of images with different sizes, colours and orientations.

Therefore, the pre-processing step is to convert all the pictures into grayscale because we'd like to make certain that colour shouldn't be a problem for detecting masks. After that, we need to have all the images in the same size before applying it to the neural network.

Training the CNN: We used the structure of SSD. The backbone network is lite. The total model only has 1.01M parameters. Input size of the model is 260x260 and the backbone network (BN Network) only has 8 convolution layers. The total model has only 24 layers with the location and classification layers counted and we have also used another CNN architecture called MobileNetV2. We merge the Backbone network to Conv layers in order to accelerate

the inference speed. The last step is to recognize whether the person is wearing the mask or not using the model trained.

The proposed system will focus on enhancing the prediction by increasing its accuracy and detection probability. This setup has its own camera module through which it monitors facemask and it has a non-contact temperature sensor to read the body temperature and allows the person if they clear the COVID-19 protocols. A hybrid system model using classical and deep learning for facial mask recognition and detection will be implemented. The system utilises face mask detection dataset which will be consisting of images with mask and images without mask, OpenCV to detect faces from a livestream through the Webcam in real-time. The image dataset will be using to build the face mask detection system. The system will be implemented with deep learning using python, OpenCV, Keras and Tensor flow. The main goal of the system is to identify whether the detected individual on the video or image is wearing a mask or is without the mask with the approach of computer vision and machine/deep learning. Here image dataset is loaded from Keras and then the images are converted into an array, later Mobile Net is used to pre-process input image and to append image to the data list. In the proposed system the main contribution includes person face identification and face mask detection. These both are done in Real Time with the help of MobileNet and OpenCV. A square box is been displayed on every person's face with the colour of red and green where red indicates the person is not wearing a mask and green indicates a person is wearing a mask.

The developed system model will be tested on the real time video streaming and with a set of images. The result of the person from the video displays a person with a square bound box. This system monitors continuously, and whenever a person is identified without a mask then the audible buzzer activates.

For implementation of mask detection using an OpenCV and pi camera interfaced to the Raspberry Pi. When the user switches on the kit then web camera capture the images. In case that image does not contain mouth and nose, it means that person is wearing mask properly and corresponding door will be opened.

The temperature sensor measures person’s temperature using contactless IR sensor. The persons pass one by one. In case the temperature exceeds average value or mask not detected or correctly aligned, the raspberry pi3 processor generates signal to lock the door and gives the audible alert through buzzer. Otherwise, the door is opened to enter. The main controlling

device of the project is Raspberry pi3 processor. Here we are using DC motor as door. The status of the project will be displayed on LCD module.

The system will be used in offices, hospital, airports, banks, sports venues, entertainment industries, hospitality places and densely populated places. The system shall support the society through offering lower spread of COVID-19 and time saving. The system will be effectively implemented during this current condition where lockdowns has been eased to allow public gatherings, mall shopping, church gathering and reopening of schools. This automation of checks will minimize the manpower for inspections at public gatherings hence can be used at any situation and period.

3.2 Software

3.2.1 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

3.2.2 History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

3.2.3 Features of Python

Python's features include −

- Easy-to-learn − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- Easy-to-read − Python code is more clearly defined and visible to the eyes.
- Easy-to-maintain − Python's source code is fairly easy-to-maintain.
- A broad standard library − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- Extendable − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases − Python provides interfaces to all major commercial databases.
- GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- Scalable − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

3.2.4 Thonny

An integrated development environment (IDE) facilitates computer programmers by integrating fundamental tools (e.g., code editor, compiler, and debugger) into a single software package. Users do not need to install the language's compiler/interpreter on their machines; an IDE provides the environment itself. Thonny is a free, dedicated IDE for Python designed for beginners.

3.2.4.1 Features of Thonny

- Easy to get started: Thonny comes with Python 3.10 built in, so just one simple installer is needed and you are ready to learn programming. (You can also use a separate Python installation, if necessary.) The initial user interface is stripped of all features that may distract beginners.

- No-hassle variables: Once you are done with hello-worlds, select *View* → *Variables* and see how your programs and shell commands affect Python variables.

- Simple debugger: Just press Ctrl+F5 instead of F5 and you can run your programs step-by-step, no breakpoints needed. Press F6 for a big step and F7 for a small step. Steps follow program structure, not just code lines.

- Step through expression evaluation: If you use small steps, then you can even see how Python evaluates your expressions. You can think of this light-blue box as a piece of paper where Python replaces subexpressions with their values, piece-by-piece.

- Faithful representation of function calls: Stepping into a function call opens a new window with separate local variables table and code pointer. Good understanding of how function calls work is especially important for understanding recursion.

- Highlights syntax errors: Unclosed quotes and parentheses are the most common beginners' syntax errors. Thonny's editor makes these easy to spot.

- Explains scopes: Highlighting variable occurrences reminds you that the same name doesn't always mean the same variable and helps spotting typos. Local variables are visually distinguished from global.

- Mode for explaining references: Variables are initially presented according to simplified model (name → value) but you can switch to more realistic model (name → address/id → value).

- Code completion: Students can explore APIs with the help of code completion.

- Beginner friendly system shell: Select Tools → Open system shell to install extra packages or learn handling Python on command line. PATH and conflicts with other Python interpreters are taken care of by Thonny.

- Simple and clean pip GUI: Select Tools → Manage packages for even easier installation of 3rd party packages.

3.2.4.2 Advantages of Thonny

There are many advantages over any other IDE; they are as follow:

1. Cross-platform support:

   - Windows
   - Linux
   - Mac

2. Light-weight

3. Robust Architecture

4. Freeware: Free of Cost- probably the best feature of all for all the programmers out there, even more for the organizations.

5. Many users will use it or might have used it for desktop applications only, but it also provides great tool support for Web Technologies like; HTML, CSS, JSON.

There are a few things that one can find a bit odd compared with so many features. It mainly helps the front-end developers as compared with the back-end developers. But as per some user's opinions, it is equally helpful. It supports most of the languages used by most of the programmers, but other languages might have to download, or extensions may have to be used

for them. Along with this common zoom-in, zoom-out brightness, theme selection features too are made available.

3.2.5 Libraries

1. OpenCV
2. TensorFlow

3.2.5.1 OpenCV

OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human.

When it integrated with various libraries, such as NumPy, python can process the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it is free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS, and Android. When OpenCV was designed the focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.

Look at the following images

Figure 3: Original Image

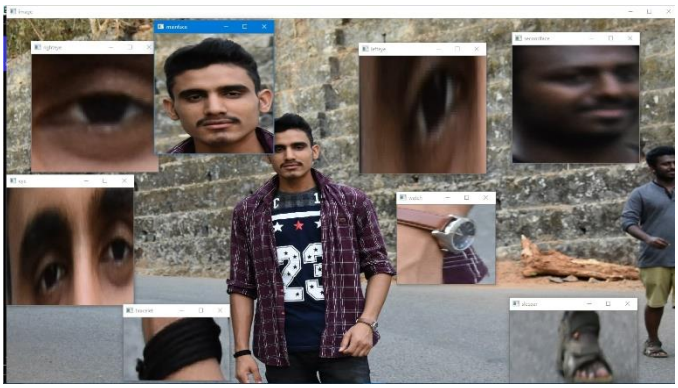

Figure 4: Classification of Image

From the above original image, lots of pieces of information that are present in the original image can be obtained. Like in the above image there are two faces available and the person(I) in the images wearing a bracelet, watch, etc. so by the help of OpenCV we can get all these types of information from the original image. It is the basic introduction to OpenCV we can continue the Applications and all the things in our upcoming articles.

Applications of OpenCV: There are lots of applications which are solved using OpenCV, some of them are listed below

- face recognition
- Automated inspection and surveillance
- number of people – count (foot traffic in a mall, etc)
- Vehicle counting on highways along with their speeds
- Interactive art installations
- Anomaly (defect) detection in the manufacturing process (the odd defective products)
- Street view image stitching
- Video/image search and retrieval
- Robot and driver-less car navigation and control
- object recognition
- Medical image analysis
- Movies – 3D structure from motion
- TV Channels advertisement recognition

Open CV Functionality

- Image/video I/O, processing, display (core, imgproc, highgui)
- Object/feature detection (objdetect, features2d, nonfree)
- Geometry-based monocular or stereo computer vision (calib3d, stitching, videostab)
- Computational photography (photo, video, superres)
- Machine learning & clustering (ml, flann)
- CUDA acceleration (gpu)

Image processing is a method to perform some operations on an image, in order to get an enhanced image and or to extract some useful information from it. If we talk about the basic definition of image processing then "Image processing is the analysis and manipulation of a digitized image, especially in order to improve its quality."

An image may be defined as a two-dimensional function f(x, y), where x and y are spatial(plane) coordinates, and the amplitude of fat any pair of coordinates (x, y) is called the intensity or grey level of the image at that point.

In another word An image is nothing more than a two-dimensional matrix (3-D in case of colored images) which is defined by the mathematical function f(x, y) at any point is giving the pixel value at that point of an image, the pixel value describes how bright that pixel is, and what color it should be. Image processing is basically signal processing in which input is an image and output is image or characteristics according to requirement associated with that image.
Image processing basically includes the following three steps:

1. Importing the image
2. Analysing and manipulating the image
3. Output in which result can be altered image or report that is based on image analysis

3.2.5.2 TensorFlow

TensorFlow is an open-source end-to-end platform for creating Machine Learning applications. It is a symbolic math library that uses dataflow and differentiable programming to perform various tasks focused on training and inference of deep neural networks.

It allows developers to create machine learning applications using various tools, libraries, and community resources. Currently, the most famous deep learning library in the world is Google's TensorFlow. Google product uses machine learning in all its products to improve the search engine, translation, image captioning or recommendations.

To give a concrete example, Google users can experience a faster and more refined search experience with AI. If the user types a keyword in the search bar, Google provides a recommendation about what could be the next word.

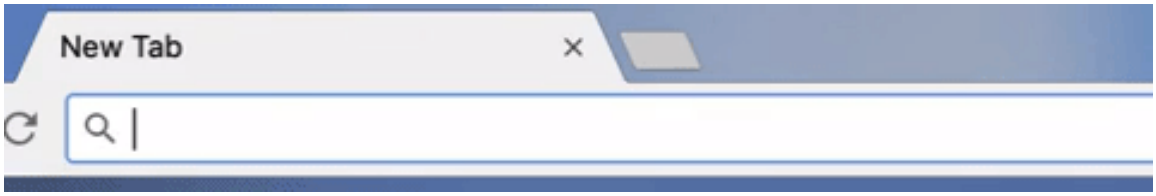

Figure 5: TensorFlow Example

Google wants to use machine learning to take advantage of their massive datasets to give users the best experience. Three different groups use machine learning:

- Researchers
- Data Scientists
- Programmers

They can all use the same toolset to collaborate with each other and improve their efficiency.

Google does not just have any data; they have the world's most massive computer, so Tensor Flow was built to scale. TensorFlow is a library developed by the Google Brain Team to accelerate machine learning and deep neural network research.

It was built to run on multiple CPUs or GPUs and even mobile operating systems, and it has several wrappers in several languages like Python, C++ or Java.

TensorFlow enables you to build dataflow graphs and structures to define how data moves through a graph by taking inputs as a multi-dimensional array called Tensor. It allows you to

construct a flowchart of operations that can be performed on these inputs, which goes at one end and comes at the other end as output.

TensorFlow architecture works in three parts:

- Pre-processing the data
- Build the model
- Train and estimate the model

It is called TensorFlow because it takes input as a multi-dimensional array, also known as tensors. You can construct a sort of flowchart of operations (called a Graph) that you want to perform on that input. The input goes in at one end, and then it flows through this system of multiple operations and comes out the other end as output. This is why it is called TensorFlow because the tensor goes in it flows through a list of operations, and then it comes out the other side. TensorFlow hardware, and software requirements can be classified into

Development Phase: This is when you train the mode. Training is usually done on your Desktop or laptop.

Run Phase or Inference Phase: Once training is done TensorFlow can be run on many different platforms. You can run it on

- Desktop running Windows, macOS or Linux
- Cloud as a web service
- Mobile devices like iOS and Android

You can train it on multiple machines then you can run it on a different machine, once you have the trained model. The model can be trained and used on GPUs as well as CPUs. GPUs were initially designed for video games. In late 2010, Stanford researchers found that GPU was also very good at matrix operations and algebra so that it makes them very fast for doing these kinds of calculations. Deep learning relies on a lot of matrix multiplication. TensorFlow is very fast at computing the matrix multiplication because it is written in C++. Although it is implemented in C++, TensorFlow can be accessed and controlled by other languages mainly, Python.

Finally, a significant feature of TensorFlow is the Tensor Board. The Tensor Board enables to monitor graphically and visually what TensorFlow is doing.

3.3 Hardware

3.3.1 LCD

Stands for "Liquid Crystal Display." LCD is a flat panel display technology commonly used in TVs and computer monitors. It is also used in screens for mobile devices, such as laptops, tablets, and smart phones.

LCD displays do not just look different from bulky CRT (Cathode Ray Tube) monitors, the way they operate is significantly different as well. Instead of firing electrons at a glass screen, an LCD has a backlight that provides light source to individual pixels arranged in a rectangular grid. Each pixel has a RGB (Red, Green, and Blue) sub-pixel that can be turned on or off. When all of a pixel's sub-pixels are turned off, it appears black.

When all the sub-pixels are turned on 100%, it appears white. By adjusting the individual levels of red, green, and blue light, millions of color combinations are obtained.

3.3.1.1 LCD Construction

An LCD screen includes a thin layer of liquid crystal material sandwiched between two electrodes on glass substrates, with two polarizers on each side. A polarizer is an optical filter that lets light waves of a specific polarization pass through while blocking light waves of other polarizations. The electrodes need to be transparent, so the most popular material is ITO (Indium Tin Oxide).

As LCD can't emit light itself, normally a backlight is placed behind an LCD screen in order to be seen during the dark environment. The light sources for backlight can be LED (Light Emitting Diode) or CCFL (Cold Cathode Fluorescent Lamps).

The LED backlight is most popular. Of course, if you like to have a color display, a layer of color filter can be made into an LCD cell. The color filter consists of RGB color. You can also add a touch panel in front of an LCD.

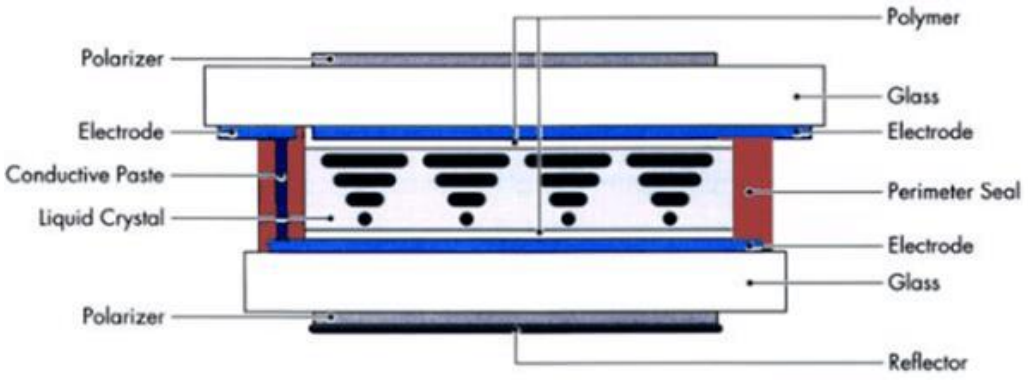

Figure 6: LCD Display Structure

3.3.1.2 Working of LCD

The first LCD panel technology in mass production is called TN (Twisted Nematic). The principle behind the LCDs is that when an electrical field is not applied to the liquid crystal molecules, the molecules twist 90 degrees in the LCD cell.

When the light either from ambient light or from the backlight passes through the first polarizer, the light is polarized and twisted with the liquid crystal molecular layer. When it reaches the second polarizer, it is blocked. The viewer sees the display is black.

When an electric field is applied to the liquid crystal molecules, they are untwisted. When the polarized light reaches the layer of liquid crystal molecules, the light passes straight through without being twisted. When it reaches the second polarizer, it will also pass through, the viewer sees the display is bright. Because LCD technology uses electric fields instead of electric current (electron passes through), it has low power consumption.

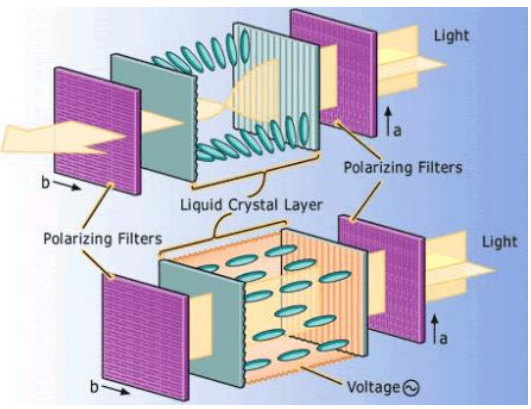

Figure 7: Working of LCD

3.3.1.3 Basics of LCD

The most basic LCD introduced above is called passive matrix LCDs which can be found mostly in low end or simple applications like, calculators, utility meters, early time digital watches, alarm clocks etc. Passive matrix LCDs have a lot of limitations, like the narrow viewing angle, slow response speed, dim, but it is great for power consumption.

In order to improve upon the drawbacks, scientists and engineers developed active-matrix LCD technology. The most widely used is TFT (Thin Film Transistor) LCD technology. Based on TFT LCD, even more modern LCD technologies are developed. The best known is IPS (In Plane Switching) LCD. It has super wide viewing angle, superior image picture quality, fast response, great contrast, less burn-in defects etc.

IPS LCDs are widely used in LCD monitors, LCD TVs, IPhone, pads etc. Samsung even revolutionized the LED backlighting to be QLED (quantum dot) to switch off LEDs wherever light is not needed to produce deeper blacks.

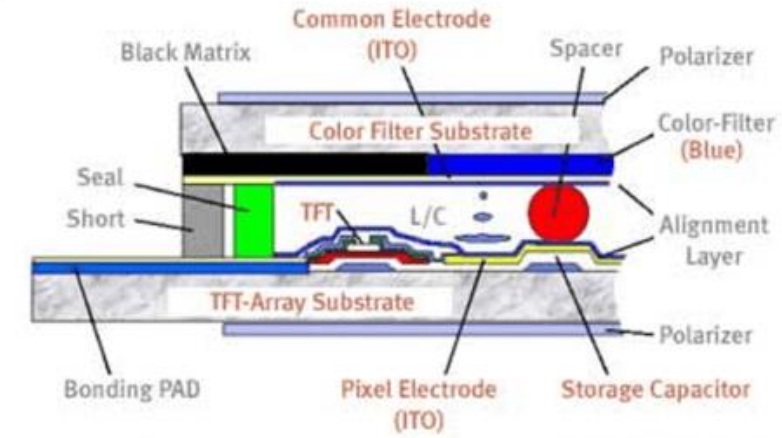

Figure 8: Active TFT Color Display

3.3.2 Raspberry pi Board:

RASPBERRY PI 4: is a development board in PI series. It can be considered as a single boardcomputer that works on LINUX operating system. The board not only has tons of features it also has terrific processing speed making it suitable for advanced applications. PI board is specifically designed for hobbyist and engineers who are interested in LINUX systems and IoT (Internet of Things).

How to Use RASPBERRY PI 4: as mentioned earlier pi is simply a computer on a single board so it cannot be used like Arduino development boards. for the pi to start working we needto first install operating system. this feature is similar to our pc. the pi has dedicated os for it; any other os will not work.

We will discuss the programming of PI in step by step below.

1. Take the 16GB micro-SD card and dedicate it specifically for PI OS.
2. Choose and Download OS software.
3. Format the SD card and install OS on to the SD memory card using convenient methods.
4. Take the SD card after OS installation and insert it in PI board.
5. Connect monitor, keyboard, and mouse
6. Power the board with micro-USB connector
7. Once the power is tuned ON the PI will run on the OS installed in the memory card and will start from boot.
8. Once all drivers are checked the PI will ask for authorization, this is set by default andcan be changed.
9. After authorization you will reach desktop where all application program development starts.

On the PI you can download application programs required for your use and can directly install as you do for your PC. After that you can work on developing required program and get the PI run the developed programs.

GPIO and the 40-pin Header: A powerful feature of the Raspberry Pi is the row of GPIO (general-purpose input/output) pins along the top edge of the board. A 40-pin GPIO header is found on all current Raspberry Pi boards (unpopulated on Raspberry Pi Zero, Raspberry Pi Zero W and Raspberry Pi Zero 2 W). Prior to the Raspberry Pi 1 Model B+ (2014), boards comprised a shorter 26-pin header. The GPIO header on all boards (including the Raspberry Pi 400) have a 0.1" (2.54mm) pin pitch

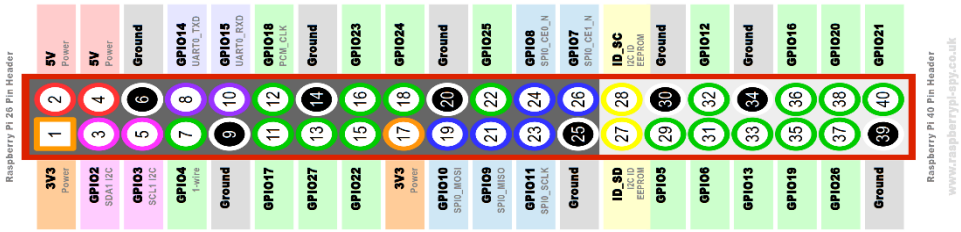

Figure 9: GPIO and 40-pin Header pins on Raspberry Pi

Raspberry Pi 4: is a development board in PI series. It can be considered as a single board computer that works on LINUX operating system. The board not only has tons of features it also has terrific processing speed making it suitable for advanced applications. PI board is specifically designed for hobbyist and engineers who are interested in LINUX systems and IoT(Internet of Things).

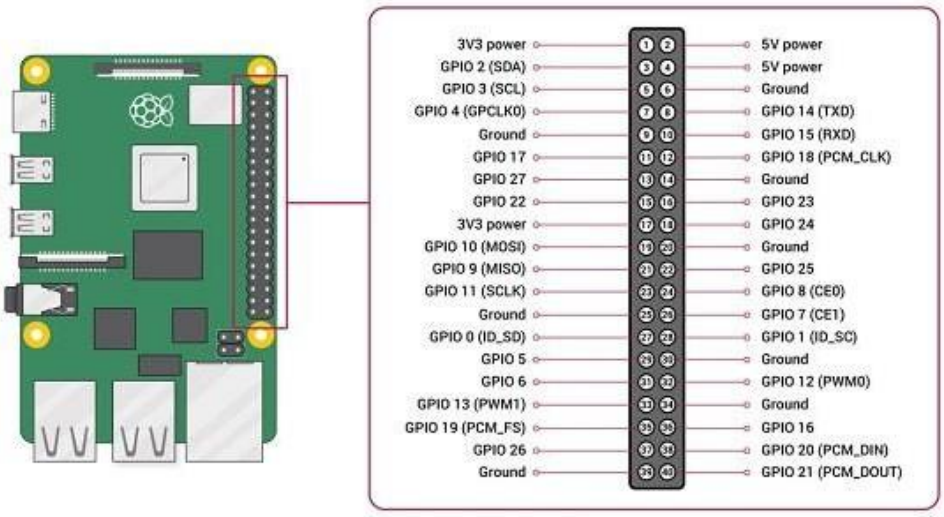

Figure 10: GPIO Pins of Raspberry Pi in Software

Any of the GPIO pins can be designated (in software) as an input or output pin and used for a wide range of purposes.

Voltages: Two 5V pins and two 3.3V pins are present on the board, as well as a number of ground pins (0V), which are unconfigurable. The remaining pins are all general purpose 3.3V pins, meaning outputs are set to 3.3V and inputs are 3.3V-tolerant.

Outputs: A GPIO pin designated as an output pin can be set to high (3.3V) or low (0V).

Inputs: A GPIO pin designated as an input pin can be read as high (3.3V) or low (0V). This is made easier with the use of internal pull-up or pull-down resistors. Pins GPIO2 and GPIO3 have fixed pull-up resistors, but for other pins this can be configured in software.

As well as simple input and output devices, the GPIO pins can be used with a variety of alternative functions, some are available on all pins, others on specific pins.

- PWM (pulse-width modulation)
    - Software PWM available on all pins
    - Hardware PWM available on GPIO12, GPIO13, GPIO18, GPIO19
- SPI
    - SPI0: MOSI (GPIO10); MISO (GPIO9); SCLK (GPIO11); CE0 (GPIO8), CE1(GPIO7)
    - SPI1: MOSI (GPIO20); MISO (GPIO19); SCLK (GPIO21); CE0 (GPIO18);CE1 (GPIO17); CE2 (GPIO16)
- I2C
    - Data: (GPIO2); Clock (GPIO3)
    - EEPROM Data: (GPIO0); EEPROM Clock (GPIO1)
- Serial
    - TX (GPIO14); RX (GPIO15)

GPIO pinout:

A handy reference can be accessed on the Raspberry Pi by opening a terminal window and running the command pinout. This tool is provided by the GPIO Zero Python library, which is installed by default on the Raspberry Pi OS desktop image, but not on Raspberry Pi OS Lite.

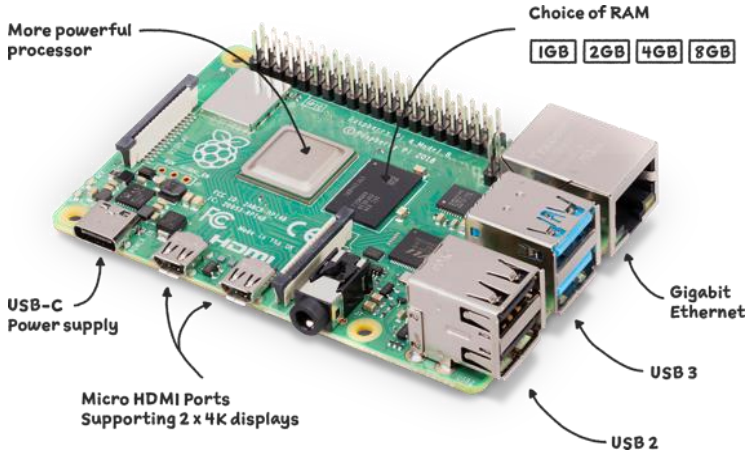

Figure 11: Raspberry Pi 4 Board

Table 2: Raspberry Pi-4 Pin Configuration

Pin group	Pin name	Description
POWER SOURCE	+5V, +3.3V, GND and Vin	+5V -power output +3.3V -power output GND – GROUND pin
COMMUNICATION INTERFACE	UART Interface(RXD, TXD) [(GPIO15,GPIO14)]	UART (Universal Asynchronous Receiver Transmitter) used for interfacing sensors and other devices
UART (Universal Asynchronous Receiver Transmitter) used for interfacing sensors and other devices	SPI (Serial Peripheral Interface) used for communicating with other boards or peripherals.	
TWI Interface (SDA, SCL) x 2 [(GPIO2, GPIO3)] [(ID_SD, ID_SC)]	WI (Two Wire Interface) Interface can be used to connect peripherals.	
INPUT OUTPUT PINS	26 I/O	Although these some pins have multiple functions they can be considered as I/O pins.
PWM	Hardware PWM availableon GPIO12, GPIO13, GPIO18, GPIO19	These 4 channels can provide PWM (Pulse Width Modulation) outputs. Software PWM available on all pins

Table 3: Raspberry Pi 4 Technical Specifications

Microprocessor	Broadcom BCM2837 64bit Quad Core Processor
Processor Operating Voltage	3.3V
Raw Voltage input	5V, 2A power source
Maximum current through each I/O pin	16mA
Maximum total current drawn from all I/O pins	54mA
Flash Memory (Operating System)	16Gbytes SSD memory card
Internal RAM	1Gbytes DDR2
Clock Frequency	1.2GHz
GPU	Dual Core Video Core IV® Multimedia Co-Processor. Provides Open GLES 2.0, hardware-accelerated Open VG, and 1080p30 H.264 high- profile decode. Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure.
Ethernet	10/100 Ethernet
Wireless Connectivity	BCM43143 (802.11 b/g/n Wireless LAN and Bluetooth 4.1)
Operating Temperature	-40ºC to +85ºC

Table 4: Board Connectors

Name	**Description**
Ethernet	Base T Ethernet Socket
USB	2.0 (Four sockets)
Audio Output	3.5mm Jack and HDMI
Video output	HDMI
Camera Connector	15-pin MIPI Camera Serial Interface (CSI-2)
Display Connector	Display Serial Interface (DSI) 15-way flat flex cable connector with two data lanes and a clock lane.
Memory Card Slot	Push/Pull Micro SDIO

3.3.3 Servo Motor:

Figure 12: Servo Motor

A servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a servo mechanism**.** If motor is powered by a DC power supply, then it is called DC servo motor, and if it is AC-powered motor then it is called AC servo motor. For this tutorial, we will be discussing only about the DC servo motor working. Apart from these major classifications, there are many other types of servo motors based on the type of gear arrangement and operating characteristics.

A servo motor usually comes with a gear arrangement that allows us to get a very high torque servo motor in small and lightweight packages. Due to these features, they are being used in many applications like toy car, RC helicopters and planes, Robotics, etc.

Servo motors are rated in kg/cm (kilogram per centimeter) most hobby servo motors are rated at 3kg/cm or 6kg/cm or 12kg/cm. This kg/cm tells you how much weight your servo motor can lift at a particular distance. For example: A 6kg/cm Servo motor should be able to lift 6kg if the load is suspended 1cm away from the motors shaft, the greater the distance the lesser the weight carrying capacity. The position of a servo motor is decided by electrical pulse and its circuitry is placed beside the motor.

3.3.3.1 Servo motor working mechanism

It consists of three parts:

1. Controlled device
2. Output sensor
3. Feedback system

It is a closed-loop system where it uses a positive feedback system to control motion and the final position of the shaft. Here the device is controlled by a feedback signal generated by comparing output signal and reference input signal.

Here reference input signal is compared to the reference output signal and the third signal is produced by the feedback system. And this third signal acts as an input signal to the control the device. This signal is present if the feedback signal is generated or there is a difference between the reference input signal and reference output signal. So, the main task of servomechanism is to maintain the output of a system at the desired value at presence of noises.

A servo consists of a Motor (DC or AC), a potentiometer, gear assembly, and a controlling circuit. First, we use gear assembly to reduce RPM and to increase torque of the motor. Say at initial position of servo motor shaft, the position of the potentiometer knob is such that there is no electrical signal generated at the output port of the potentiometer. Now an electrical signal is given to another input terminal of the error detector amplifier.

Now the difference between these two signals, one comes from the potentiometer and another comes from other sources, will be processed in a feedback mechanism and output will be provided in terms of error signal. This error signal acts as the input for motor and motor starts rotating. Now motor shaft relates to the potentiometer and as the motor rotates so the potentiometer and it will generate a signal. So as the potentiometer's angular position changes, its output feedback signal changes. After sometime the position of potentiometer reaches at a position that the output of potentiometer is same as external signal provided. At this condition, there will be no output signal from the amplifier to the motor input as there is no difference between external applied signal and the signal generated at potentiometer, and in this situation motor stops rotating.

Interfacing hobby Servo motors like s90 servo motor with MCU is very easy. Servos have three wires coming out of them**.** Out of which two will be used for Supply (positive and negative) and one will be used for the signal that is to be sent from the MCU. An MG995 Metal Gear Servo Motor which is most used for RC cars humanoid bots etc. The picture of MG995 is shown below:

Figure 13: MG995 Metal Gear Servo Motor

The color coding of your servo motor might differ hence check for your respective datasheet. All servo motors work directly with your +5V supply rails but we have to be careful on the amount of current the motor would consume if you are planning to use more than two servo motors a proper servo shield should be designed.

All motors have three wires coming out of them. Out of which two will be used for Supply (positive and negative) and one will be used for the signal that is to be sent from the MCU.

Servo motor is controlled by PWM (Pulse with Modulation) which is provided by the control wires. There is a minimum pulse, a maximum pulse and a repetition rate. Servo motor can turn 90 degrees from either direction form its neutral position. The servo motor expects to see a pulse every 20 milliseconds (ms) and the length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the motor turn to the 90° position, such as if pulse is shorter than 1.5ms shaft moves to 0° and if it is longer than 1.5ms than it will turn the servo to 180°.

Servo motor works on PWM (Pulse width modulation) principle, means its angle of rotation is controlled by the duration of applied pulse to its Control PIN. Basically, servo motor is made up of DC motor which is controlled by a variable resistor (potentiometer) and some gears. High speed force of DC motor is converted into torque by Gears. We know that WORK= FORCE X DISTANCE, in DC motor Force is less and distance (speed) is high and in Servo, force is High and distance is less. The potentiometer is connected to the output shaft of the Servo, to calculate the angle and stop the DC motor on the required angle.

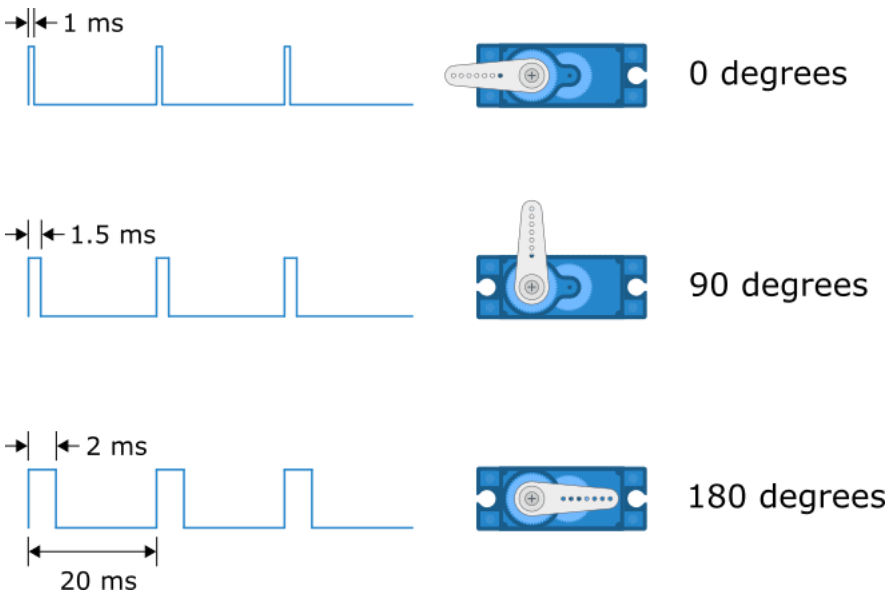

Figure 14: Angular rotation of servo motor

Servo motor can be rotated from 0 to 180 degrees, but it can go up to 210 degrees, depending on the manufacturing. This degree of rotation can be controlled by applying the Electrical Pulse of proper width, to its Control pin. Servo checks the pulse in every 20 milliseconds. The pulse of 1 ms (1 millisecond) width can rotate the servo to 0 degrees, 1.5ms can rotate to 90 degrees (neutral position) and 2 ms pulse can rotate it to 180 degrees.

All servo motors work directly with your +5V supply rails but we must be careful about the amount of current the motor would consume if you are planning to use more than two servo motors a proper servo shield should be designed.

3.3.4 MLX90614 Non-Contact IR Temperature Sensor:

The MLX90614 is a Contactless Infrared (IR) Digital Temperature Sensor that can be used to measure the temperature of a particular object ranging from -70° C to 382.2°C. The sensor uses IR rays to measure the temperature of the object without any physical contact and communicates to the microcontroller using the I2C protocol.

Figure 15: MLX90614 IR Temperature Sensor

Pin No.	Pin Name	Description
1	Vdd (Power supply)	Vdd can be used to power the sensor, typically using 5V
2	Ground	The metal can also act as ground
3	SDA – Serial Data	Serial data pin used for I2C Communication
4	SCL – Serial Clock	Serial Clock Pin used for I2C Communication

Figure 16: MLX90614 Pinout Configuration

3.3.4.1 Working principle

As mentioned earlier, the MLX90614 sensor can measure the temperature of an object without any physical contact with it. This is made possible with a law called Stefan-Boltzmann Law, which states that all objects and living beings emit IR Energy and the intensity of this emitted IR energy will be directly proportional to the temperature of that object or living being. So, the MLX90614 sensor calculates the temperature of an object by measuring the amount of IR energy emitted from it.

The MLX90614 Temperature sensor is manufactured by a company called Melexis. The sensor is factory calibrated and hence it acts like a plug and play sensor module for speeding up development processes.

The MLX90614 consists of two devices embedded as a single sensor, one device acts as a sensing unit and the other device acts as a processing unit. The sensing unit an Infrared Thermopile Detector called MLX81101 which senses the temperature and the processing unit is a Single Conditioning ASSP called MLX90302 which converts the signal from the sensor to digital value and communicates using I2C protocol. The MLX90302 has a low noise amplifier, 17-bit ADC and a powerful DSP which helps the sensor to have high accuracy and resolution.

3.3.5 Raspberry pi cam

Figure 17: Raspberry pi camera module

This 5-megapixel sensor with OV5647 camera module is capable of 1080p video and still images that connect directly to your Raspberry Pi. This is the plug-and-play-compatible latest version of the Raspbian operating system, making it perfect for time-lapse photography, recording video, motion detection and security applications. Connect the included ribbon cable to the CSI (Camera Serial Interface) port on your Raspberry Pi, and you are good to go!

The board itself is tiny, at around 25mm x 23mm x 9mm and weighing in at just over 3g, making it perfect for mobile or other applications where size and weight are important. The sensor has a native resolution of 5 megapixel, and has a fixed focus lens on board. In terms of still images, the camera is capable of 2592 x 1944-pixel static images, and supports 1080p30, 720p60 and 640x480p90 video.

Note: This module is only capable of taking pictures and video, not sound.

3.3.5.1 Features

- Compatible with Raspberry Pi 4 Model B/3B+/3B/2B/Zero Wireless
- 5 Megapixel OV5647 Camera
- Camera specifications
- Static Images Resolution: 2592×1944

- Supported Video Resolution: 1080p/30 fps, 720p/ 60fps and 640 x480p 60/90 video recording
- Aperture (F): 1.8
- Visual Angle: 65 degrees
- Dimension: 24mmx23.5mmx8mm
- Weight: 3g
- Interface: CSI connector
- Supported OS: Raspbian (latest version recommended)

3.3.5.2 Preparation

Product List

- 1 x Camera Module
- 1 x 15cm FFC Cable for Raspberry Pi 4B/3B/2B/1B+

Required Preparation

- Raspberry Pi 2B/3A+/3B/3B+/4B/zero/zero w
- Raspberry Pi power adaptor
- Network Cable (it is for RPi system installation and you can ignore it if you have finished.)
- Card Reader: (it is for RPi system installation and you can ignore it if you have finished.)
- SD card (a recommended 8g and higher one) (it is for RPi system installation and you can ignore it if you have finished.)

3.3.5.3 How to use Camera module with Pi

The Pi camera module when purchased comes along with a ribbon cable, this cable must be connected to the CSI (Camera Serial Interface) port of the Pi**.** This port can be found near the HDMI port just connect the cable to it as shown below.

Figure 18: Interfacing camera module with Pi

After interfacing the hardware, we must configure the Pi to enable Camera. Use the command "sudo raspi-config" to open the configuration window. Then under interfacing options enable camera. Finally reboot the Pi and your camera module is ready to use. Then, you can make the Pi to take photos or record videos using simple python scripts.

3.3.6 Buzzer:

An audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.

Figure 19: Buzzer Pin Configuration

The pin configuration of the buzzer is shown below. It includes two pins namely positive and negative. The positive terminal of this is represented with the '+' symbol or a longer terminal. This terminal is powered through 6Volts whereas the negative terminal is represented with the '-' symbol or short terminal and it is connected to the GND terminal.

The specifications of the buzzer include the following.

- Color is black
- The frequency range is 3,300Hz
- Operating Temperature ranges from – 20° C to +60°C
- Operating voltage ranges from 3V to 24V DC
- The sound pressure level is 85dBA or 10cm
- The supply current is below 15mA

3.3.6.1 Working

The working principle of a buzzer depends on the theory that, once the voltage is given across a piezoelectric material, then a pressure difference is produced. A piezo type includes piezo crystals among two conductors.

Once a potential disparity is given across these crystals, then they thrust one conductor & drag the additional conductor through their internal property. So, this continuous action will produce a sharp sound signal.

The circuit diagram of the water level indicator using the buzzer is shown below. This circuit is used to sense or detect the water level within the tank or washing machine or pool, etc. This circuit is very simple to design using few components such as a transistor, buzzer, 300K variable resistor, and power supply or 9V battery.

A buzzer is an efficient component to include the features of sound in our system or project. It is an extremely small & solid two-pin device thus it can be simply utilized on breadboard or PCB. So, in most applications, this component is widely used.

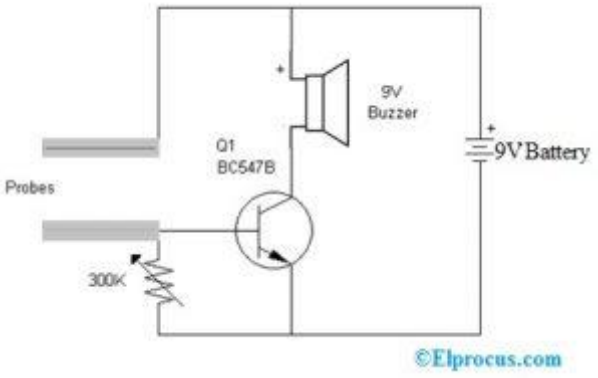

Figure 20: Water level Circuit using Buzzer

Once the two probes of the circuit are placed in the tank, it detects the level of water. Once the water level exceeds the fixed level, then it generates a beep sound through a buzzer connected to the circuit. This circuit uses a BC547B NPN transistor however we can also use any general-purpose transistor instead of using 2N3904/2N2222.

This water level sensor circuit working is very simple and the transistor used within the circuit works as a switch. Once the two probes notice the water level within the tank, then the transistor turns ON & the voltage begins flowing throughout the transistor to trigger the buzzer.

Chapter 4
RESULT AND DISCUSSION

4.1 Working

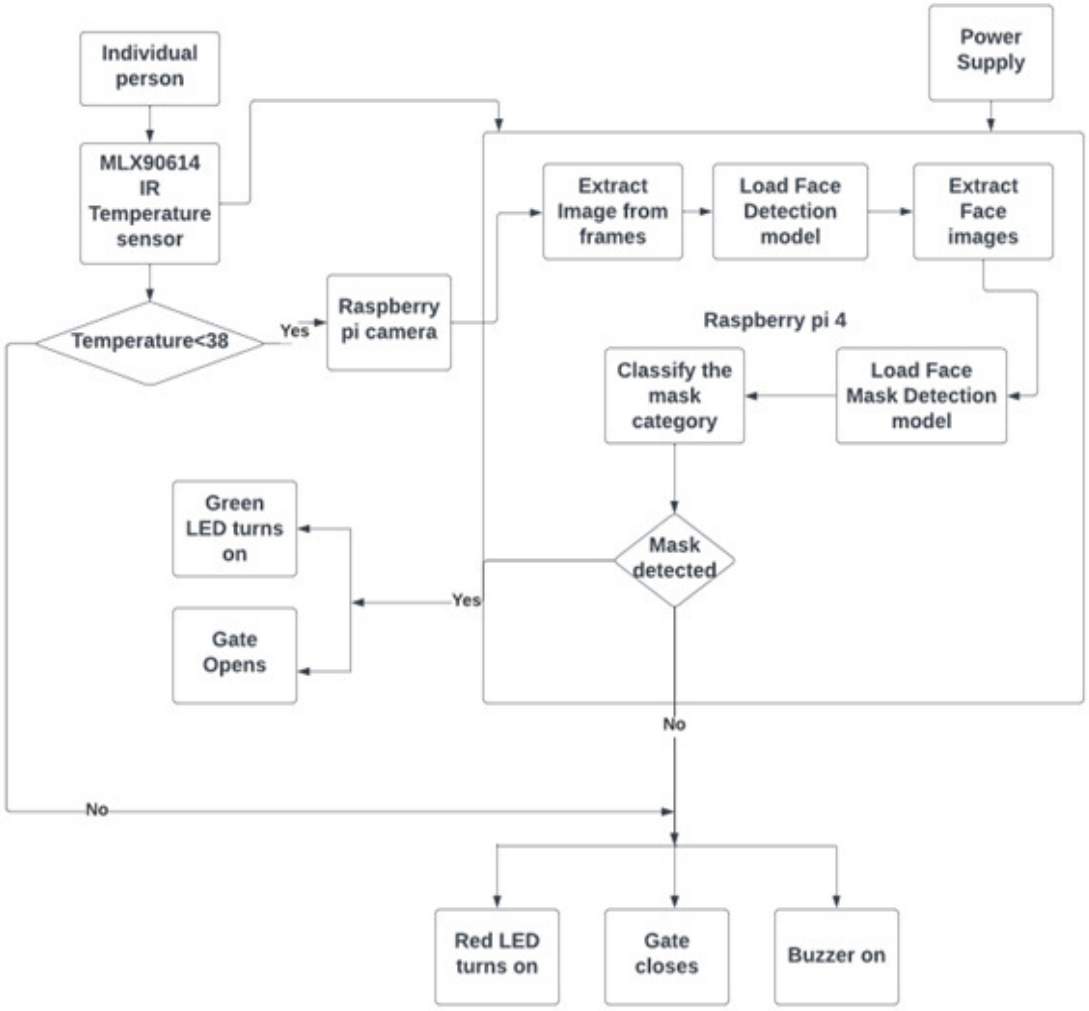

Figure 21: Workflow Diagram

At first, a model with hundreds of pictures, both masks and not masks, is trained here. The model distinguishes between faces with/without mask.

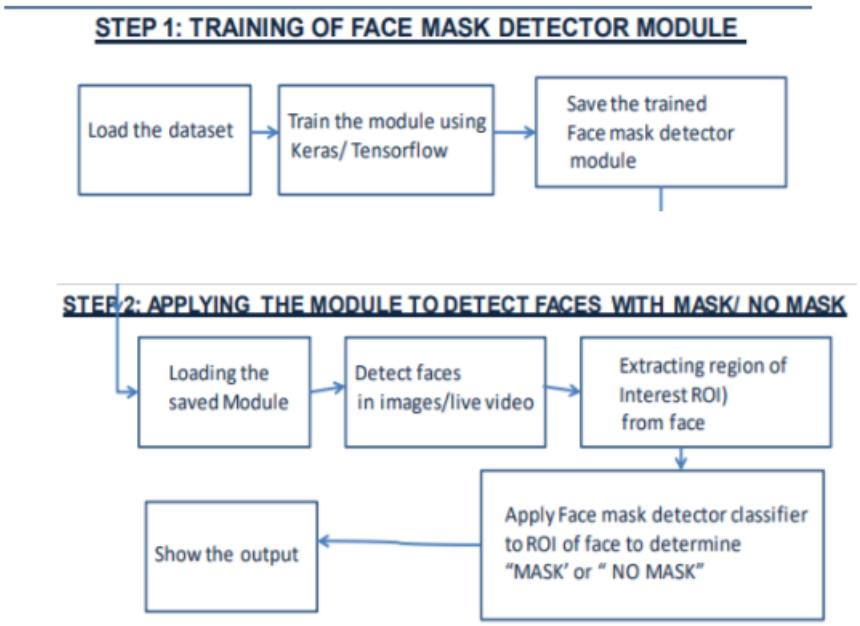

Figure 22: Training steps of data set

Before facial recognition, a database of diverse people was created, which includes their faces, unique characteristics, and other personal details. These are extracted from the database and stored in a pre-configured folder. We will use the features from these images to improve our classification systems. We will also compare these features to the ones that were already stored in a database.

This system can identify people wearing a facemask on a video stream by using various deep learning and computer vision techniques such as Opencv, Keras, TensorFlow, etc.

The MLX90614 is s non-contact infrared thermometer which is used to determine the temperature of a person's body. Then into a Raspberry Pi data is been feeded through the thermometer and which then displayed on an LCD screen.

The green LED will be turned on if the person is wearing a face mask and their temperature is below the threshold. The red LED will be activated if the mask is not worn. If the temperature is above the threshold, the buzzer will be turned on and the individual's face will be identified. If the temperature is below and mask is worn, then gate will open.

We used a dataset that covered all potential test cases to build this system. Selenium and numerous other extensions were used to scrape photographs from the internet of people wearing masks and people who were not wearing masks in order to construct a dataset. This data collection was split into two parts.

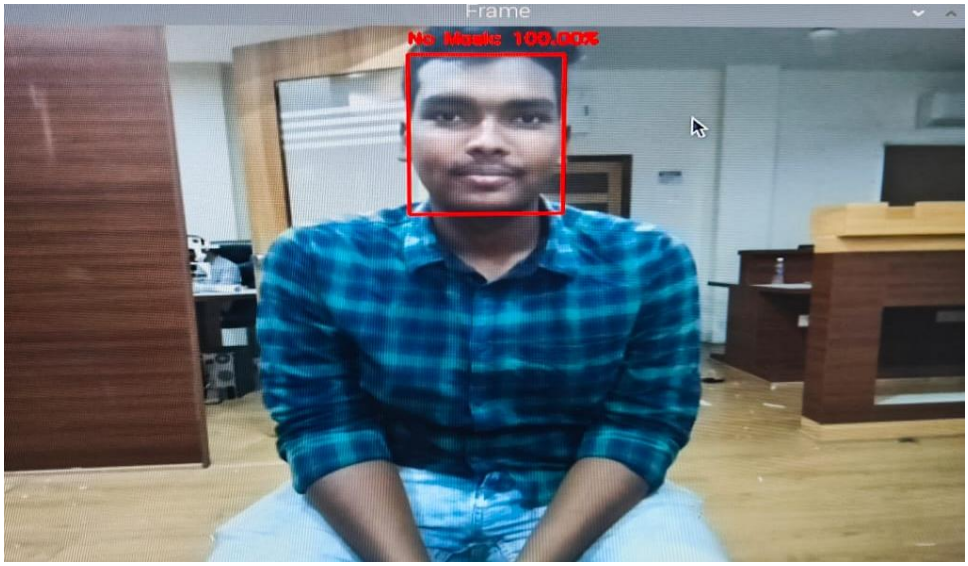

Figure 23: Image showing person without mask

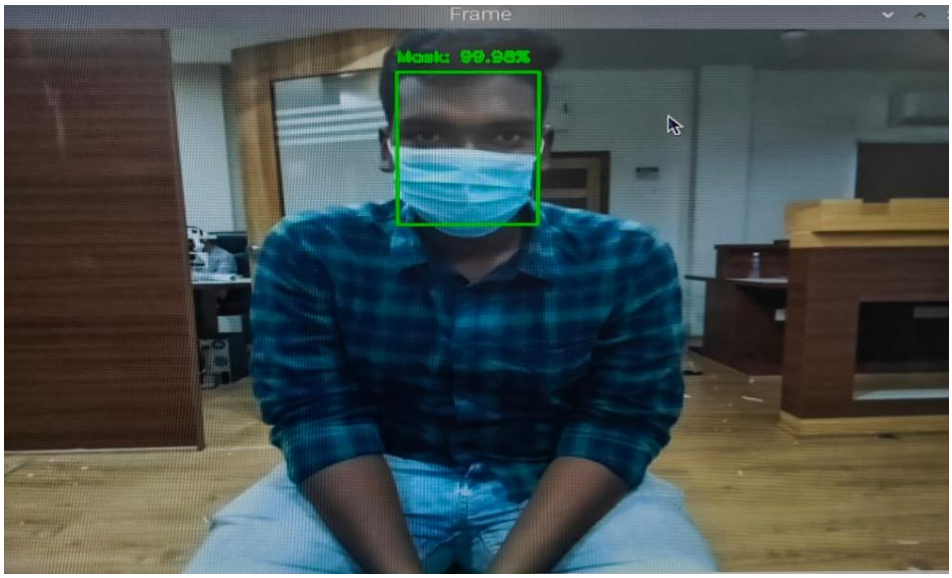

Figure 24: Image showing people with mask

There are two sections, namely the Training set and the Validation set, which contain 80 percent and 20 percent photos, respectively.

Using Deep Learning and Computer Vision techniques and libraries such as OpenCV, this system will identify people wearing a facemask on an image/video stream.

Others include Keras and Tensor Flow. Our focus will be on to load our face mask detection dataset from disc part. The images we downloaded are in a variety of formats. Sizes and resolutions are available in a variety of formats. Consequently, crop and resize the image, RGB color transformation of the source image (256 x 256) filtering.

The MLX90614 ESF is a non-contact infrared thermometer that is used to determine a person's body temperature. The data from the thermometer is sent to a Raspberry Pi, which displays it on screen. The green LED will glow, and the relay will activate the motor. So that gate will open to enter him/her into a respective building.

4.1.1 Algorithm for Face Mask Detection

It is divided into two phases:

i) TRAINING PHASE: In this phase we will generate a face detection model using classification algorithm. This phase is further divided into two stages:

a) Data Pre-processing

b) Data Training

The training includes the following process –

- Loading Image Data
- Pre-processing
- Loading the MobileNetV2 classifier
- Building a new fully connected (FC) head

a) Data Pre-processing: -

This stage works on image data available in dataset. In this face mask detection project, dataset is a classified collection on image data of peoples wearing mask and not wearing mask generated via facial land marking artificially.

The Pre-processing process includes: -

- Loading image dataset.
- Resizing dataset images to 224 x 224 pixels.
- Conversion to array format using numpy array.
- Scaling the pixel intensities in the input image.
- Ensuring training data is in numpy array format.
- Segmentation image data for training and testing.
- Data augmentation.

b) Data Training: -

This stage will work on training the preprocessed image data using classification algorithm. Algorithm used in this is MobileNetV2 to generate a base model for testing purpose.

The Data Training process includes:-

- Fine tuning of the data.
- Loading of MobileNet with pre-trained image Net weights.
- Construction of new FC head.
- Adam optimizer for model compilation.
- Model generation.

ii) DEPLOYMENT PHASE:

This phase will do the rural time detection of face mask using the trained model from the training phase using classification algorithm:

- Load the trained models.
- Get the real-time image data.
- Compare real time data with trained data.
- Show the results using OpenCV.

4.2 Code

Final_project.py

```
# import the necessary packages
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from imutils.video import VideoStream
import numpy as np
import imutils
import time
import cv2
import os
from smbus2 import SMBus
from mlx90614 import MLX90614
# from RPi_GPIO_i2c_LCD import lcd
from rpi_lcd import LCD
import Adafruit_DHT
import RPi.GPIO as GPIO
import time
import os,sys
import spidev # To communicate with SPI devices
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
#*******************Defining all use pin ********************************#
## Address of backpack
# i2c_address = 0x27

## Initialize display
# LCD = lcd.HD44780(i2c_address)
LCD1 = LCD()
I2C_BUS = 1
```

```
SENSOR_ADDRESS = 0x5A
# Initialize the I2C bus
bus = SMBus(I2C_BUS)

# Initialize the MLX90614 sensor
sensor = MLX90614(bus, address=SENSOR_ADDRESS)

green_led_pin = 29
Red_led_pin = 33
servo_pin=31
buzzer_pin = 36
GPIO.setup(servo_pin,GPIO.OUT)

pwm = GPIO.PWM(servo_pin,50) # 50 Hz (20 ms PWM period)
pwm.start(7) # start PWM by rotating to 90 degrees
pwm.ChangeDutyCycle(0)

GPIO.setup(Red_led_pin , GPIO.OUT)
GPIO.setup(green_led_pin , GPIO.OUT)
GPIO.setup(buzzer_pin, GPIO.OUT)
s=1
flag='c'

#***********Read Temperature code start ***************#
# Start SPI connection
def Temp():
    temperature = sensor.get_object_1()
    return temperature

#***********Read Temperature code end ***************#
```

```
#******************detect function *********************#
def detect_and_predict_mask(frame, faceNet, maskNet):
	# Grab the dimensions of the frame and then construct a blob from it
	(h, w) = frame.shape[:2]
	blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),(104.0, 177.0, 123.0))

	# Pass the blob through the network and obtain the face detections
	faceNet.setInput(blob)
	detections = faceNet.forward()
	#print(detections.shape)

	# Initialize our list of faces, their corresponding locations,
	# and the list of predictions from our face mask network
	faces = []
	locs = []
	preds = []

	# loop over the detections
	for i in range(0, detections.shape[2]):
		# Extract the confidence (i.e., probability) associated with the detection
		confidence = detections[0, 0, i, 2]

		# Filter out weak detections by ensuring the confidence is
		# greater than the minimum confidence
		if confidence > 0.5:
			# Compute the (x, y)-coordinates of the bounding box for
			# the object
			box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
			(startX, startY, endX, endY) = box.astype("int")
```

```
                # Ensure the bounding boxes fall within the dimensions of
                # the frame
                (startX, startY) = (max(0, startX), max(0, startY))
                (endX, endY) = (min(w - 1, endX), min(h - 1, endY))

                # extract the face ROI, convert it from BGR to RGB channel
                # ordering, resize it to 224x224, and preprocess it
                face = frame[startY:endY, startX:endX]
                face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
                face = cv2.resize(face, (224, 224))
                face = img_to_array(face)
                face = preprocess_input(face)

                # add the face and bounding boxes to their respective lists
                faces.append(face)
                locs.append((startX, startY, endX, endY))

        # only make a prediction if at least one face was detected
        if len(faces) > 0:
                # for faster inference we will make batch predictions on *all*
                # faces at the same time rather than one-by-one predictions
                # in the above `for` loop
                faces = np.array(faces, dtype="float32")
                preds = maskNet.predict(faces, batch_size=32)

        # return a 2-tuple of the face locations and their corresponding locations
        return (locs, preds)


#******************detect function end *********************************#
```

```
# load our serialized face detector model from disk
prototxtPath = r"/home/facemask/Downloads/deploy.prototxt"
weightsPath = r"/home/facemask/Downloads/res10_300x300_ssd_iter_140000.caffemodel"
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

# load the face mask detector model from disk
maskNet = load_model("/home/facemask/Downloads/mask_detector.model")

# initialize the video stream
print("[INFO] starting video stream...")
vs = VideoStream(0).start()

#*****************software function *************************#
def software():
    s=1
    flag='c'

    while True:
        # grab the frame from the threaded video stream and resize it
        # to have a maximum width of 400 pixels

        frame = vs.read()
        frame = imutils.resize(frame, width=400)

        # detect faces in the frame and determine if they are wearing a face mask or not
        (locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)

        # loop over the detected face locations and their corresponding locations
        for (box, pred) in zip(locs, preds):
            # unpack the bounding box and predictions
            (startX, startY, endX, endY) = box
            (mask, withoutMask) = pred
```

```
        # determine the class label and color we'll use to draw the bounding box and text
        label = "Mask" if mask > withoutMask else "No Mask"
        color = (0, 255, 0) if label == "Mask" else (0, 0, 255)
           # include the probability in the label
        label1 = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)
        # display the label and bounding box rectangle on the output frame
        cv2.putText(frame, label1, (startX, startY - 10),
           cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
        cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)
        # show the output frame
        cv2.imshow("Frame", frame)
        cv2.waitKey(1000)
        if(label == "Mask"):
           print("mask detected")
           flag='a'
           s=0
        else:
           print("mask not detected")
           flag='b'
           s=0

     if s == 0:
        break
  return flag
#**************software function end **********************#

#**************Main code **********************#
LCD1.text("WELCOME TO FACE",1)
LCD1.text("MASK DETECTION",2)
time.sleep(5)
while(1):
  LCD1.clear()
```

```
LCD1.text("Please Scan",1)
LCD1.text("Your Temperature",2)
time.sleep(5)
temp  = Temp()
itemp=int(temp)
LCD1.clear()
LCD1.text("Your Body Temp: ",1)
LCD1.text(str(itemp),2)
time.sleep(5)
print(temp)

LCD1.clear()
LCD1.text("Please Scan ",1)
LCD1.text("Your Face ",2)
time.sleep(1)
if(temp<35):
    flag=software()
    if((flag == 'a') ):
        LCD1.clear()
        LCD1.text("Gate Open",1)
        time.sleep(2)
        GPIO.output(green_led_pin,GPIO.HIGH)  #LED ON
        #GPIO.output(Red_led_pin,GPIO.LOW)  #LED OFF
        #pwm.start(7)
        #pwm.ChangeDutyCycle(0)
        pwm.ChangeDutyCycle(2.0) # rotate to 0 degrees
        time.sleep(1)
        pwm.ChangeDutyCycle(7.0) # rotate to 90 degrees
        time.sleep(1)
        pwm.ChangeDutyCycle(0)
        # stops the pwm on 13
        #pwm.stop()
```

```
        GPIO.output(green_led_pin,GPIO.LOW)

        elif(flag == 'b'):
            LCD1.clear()
            LCD1.text("Please wear mask",1)
            #GPIO.output(green_led_pin,GPIO.LOW)  #LED OFF
            GPIO.output(Red_led_pin,GPIO.HIGH)  #LED ON
            GPIO.output(buzzer_pin,GPIO.HIGH)  #Buzzer On
            time.sleep(2)
            GPIO.output(buzzer_pin,GPIO.LOW)  #Buzzer OFF
            GPIO.output(Red_led_pin,GPIO.LOW)
    else:
        LCD1.clear()
        LCD1.text("High Body Temperature ",1)
        #GPIO.output(green_led_pin,GPIO.LOW)  #LED OFF
        GPIO.output(Red_led_pin,GPIO.HIGH)  #LED ON
        GPIO.output(buzzer_pin,GPIO.HIGH)  #Buzzer On
        time.sleep(0.5)

        GPIO.output(buzzer_pin,GPIO.LOW)  #Buzzer OFF
        GPIO.output(Red_led_pin,GPIO.LOW)
            #cv2.putText(img,n,(x,y),font,1,(255,100,100),2) #show image
        #cv2.imshow("output",img) #show image
        # 27 is ASCII value of escapes key
        #It is use to run thread which waiting for ESC button because of this image will show on
window
        #if you not use this function then it will not load image
    if((cv2.waitKey(2) == 27)):
            LCD1.clear()
            pwm.stop()
            break;
```

```
# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()
```

Output:

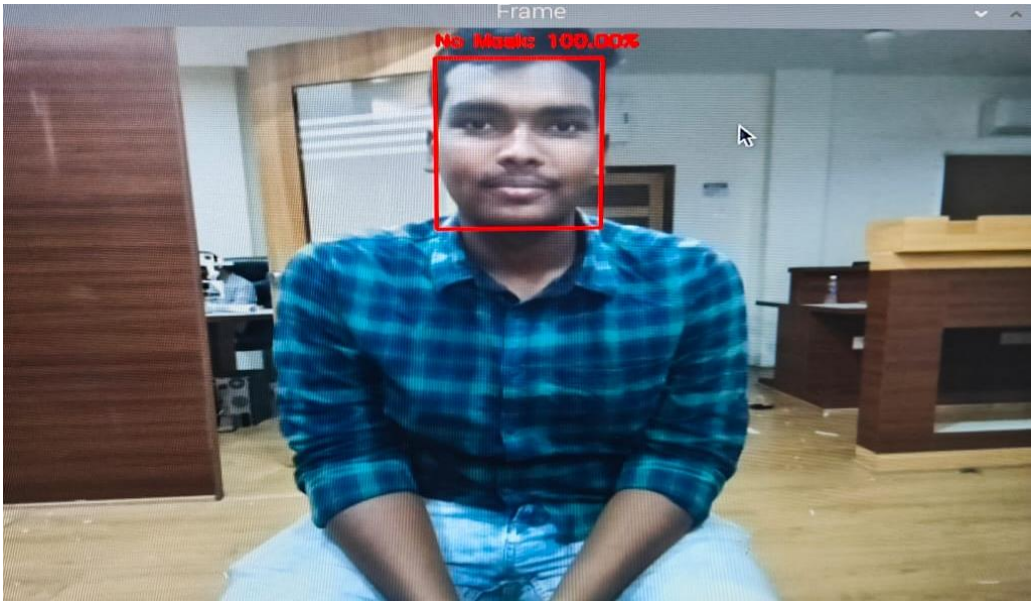

Figure 25: Without mask

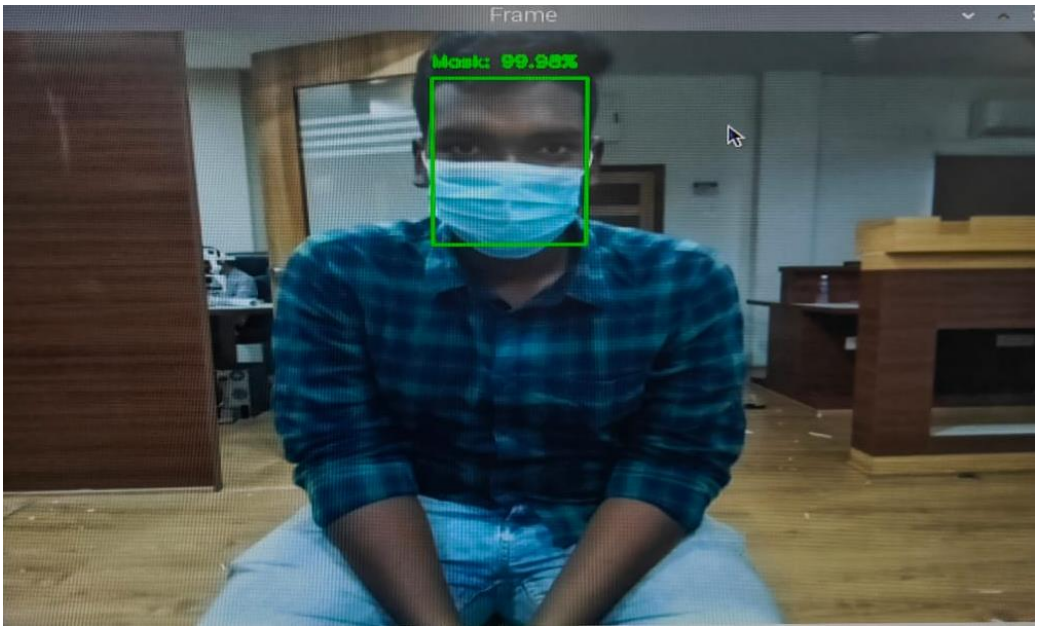

Figure 26: With Mask

4.3 Advantages

Using a face mask detection system has several advantages, particularly in the context of public health and safety. Here are some detailed explanations of these advantages:

1. Ensuring Compliance with Mask Mandates: Face mask mandates have been implemented in many places to prevent the spread of infectious diseases, such as COVID-19. A face mask detection system can help enforce these mandates by identifying individuals who are not wearing masks or are not wearing them properly. This ensures that people follow the regulations and reduces the risk of infection transmission.

2. Automating Monitoring Processes: Manual monitoring of mask compliance can be time-consuming and resource-intensive, especially in crowded areas. A face mask detection system automates this process by using computer vision algorithms to analyze video footage or images in real-time. It can quickly and accurately identify whether a person is wearing a mask, eliminating the need for constant human supervision.

3. Improved Safety in High-Risk Environments: Face mask detection systems are particularly useful in high-risk environments, such as hospitals, airports, and public transportation. These places often have a higher likelihood of disease transmission, and ensuring mask compliance becomes crucial. By using a detection system, potential outbreaks can be identified early, and appropriate actions can be taken to mitigate the risks.

4. Prompt Alerting and Intervention: Face mask detection systems can be integrated with alert systems to notify authorities or staff when individuals are not wearing masks. This enables quick intervention, such as reminding the person to wear a mask or denying entry to certain areas until compliance is met. Prompt alerting ensures a swift response to potential violations, reducing the chance of virus transmission.

5. Data Collection and Analysis: Face mask detection systems can collect data on mask compliance rates, patterns, and trends over time. This information is valuable for public health officials and policymakers in assessing the effectiveness of mask mandates, identifying areas with low compliance, and making data-driven decisions. It can also be used for research purposes, such as studying behavioral patterns related to mask-wearing.

6. Reducing Confrontations and Enhancing Customer Experience: In settings where mask mandates are in place, conflicts can arise when individuals refuse to wear masks or do not

comply. A face mask detection system reduces the need for direct confrontations by automatically identifying non-compliant individuals. This leads to a smoother and more positive customer experience, as staff can focus on providing assistance rather than enforcement.

7. Scalability and Cost-Effectiveness: Face mask detection systems can be deployed in various settings, from small businesses to large public spaces, and can scale as needed. Once implemented, the system operates continuously without the need for constant human supervision, making it cost-effective in the long run. It offers a reliable solution that can be easily integrated with existing security or surveillance infrastructure.

In summary, a face mask detection system offers advantages such as ensuring compliance with mask mandates, automating monitoring processes, improving safety in high-risk environments, prompt alerting and intervention, data collection and analysis, reducing confrontations, and scalability. By leveraging technology to enforce mask-wearing, these systems contribute to public health efforts and help mitigate the spread of infectious diseases.

4.4 Disadvantages

While using a face mask detection system offers several advantages, it is important to consider potential disadvantages as well. Here are some detailed explanations of the disadvantages:

1. Privacy Concerns: Face mask detection systems typically rely on video surveillance or image processing technologies to identify individuals wearing masks. This can raise concerns about privacy, as people may feel uncomfortable with constant monitoring and potential storage of their facial data. It is crucial to implement appropriate data protection measures and ensure transparency regarding the use and storage of collected data to address these concerns.

2. Technical Limitations and False Positives: Face mask detection systems rely on computer vision algorithms that analyze images or video footage to determine mask compliance. However, these systems may encounter technical limitations, such as difficulties in accurately detecting masks due to lighting conditions, occlusions, or different types of masks. This can lead to false positives or false negatives, where individuals are incorrectly identified as non-compliant or compliant, respectively.

3. Adaptability to New Mask Designs: As mask designs evolve over time, face mask detection systems may encounter challenges in adapting to new types of masks. The system's algorithms may not be trained or optimized for detecting specific mask variations, which can result in decreased accuracy. Regular updates and training of the system with new mask designs can help mitigate this issue.

4. Reliance on Visual Detection Only: Face mask detection systems primarily rely on visual detection of masks. However, this approach may not be foolproof, as it cannot detect certain scenarios such as people wearing transparent masks, face shields, or masks with inadequate coverage. Additionally, the system may not be able to identify if a mask is worn correctly, leaving room for potential loopholes.

5. Potential Bias and Discrimination: Face mask detection systems can inadvertently introduce bias and discrimination if not developed and trained with diverse datasets. If the system is biased towards specific racial or ethnic groups, it can result in disproportionate targeting or false identifications, leading to unequal treatment. It is crucial to ensure that the system is trained on diverse datasets and regularly audited to address bias-related concerns.

6. Implementation Costs and Maintenance: Implementing a face mask detection system requires upfront investment in hardware, software, and infrastructure, including cameras, sensors, and data processing capabilities. Additionally, ongoing maintenance and updates may be necessary to keep the system accurate and effective. These costs should be considered in the decision-making process.

7. Overreliance on Technology: While face mask detection systems offer automation and efficiency, overreliance on technology can lead to complacency in human monitoring and decision-making. It is important to strike a balance between technology and human intervention to ensure the accuracy and fairness of mask detection, especially in situations where contextual understanding and judgment are required.

In conclusion, face mask detection systems have potential disadvantages such as privacy concerns, technical limitations and false positives, adaptability to new mask designs, reliance on visual detection only, potential bias and discrimination, implementation costs and maintenance, and overreliance on technology. These challenges need to be addressed and carefully managed to ensure the ethical and effective implementation of such systems.

4.5 Applications

Using a face mask detection system has various applications across different industries and sectors. Here are some detailed explanations of these applications:

1. Public Health and Safety: Face mask detection systems are particularly valuable in public health and safety settings. They can be implemented in hospitals, clinics, and healthcare facilities to ensure compliance with mask-wearing protocols among staff and visitors. These systems help reduce the risk of infection transmission and contribute to a safer healthcare environment.

2. Transportation Hubs: Airports, train stations, bus terminals, and other transportation hubs can benefit from face mask detection systems. These systems can be integrated with access control systems to identify individuals who are not wearing masks properly or not at all. This helps enforce mask mandates, improving safety for travelers and staff.

3. Retail and Commercial Spaces: Face mask detection systems can be used in retail stores, shopping malls, and commercial buildings to ensure compliance with mask requirements. By automatically detecting non-compliant individuals, the system can alert store staff or security personnel to intervene and enforce mask-wearing policies. This helps create a safer shopping and working environment.

4. Educational Institutions: Schools, colleges, and universities can implement face mask detection systems to monitor mask compliance among students, teachers, and staff. These systems can be integrated with existing security systems or attendance management systems to ensure that individuals entering the premises are wearing masks. It promotes a healthy and safe learning environment.

5. Hospitality and Entertainment Venues: Hotels, restaurants, theaters, and other entertainment venues can use face mask detection systems to enforce mask-wearing protocols among guests and employees. By automatically detecting non-compliance, the system can alert the management to take appropriate actions, such as reminding guests to wear masks or denying entry if necessary. It helps maintain a safe and enjoyable experience for everyone.

6. Manufacturing and Industrial Settings: Face mask detection systems can be deployed in manufacturing plants, warehouses, and industrial facilities to ensure that employees follow safety protocols, including wearing masks.

7. Public Spaces and Events: Face mask detection systems can be used in public spaces, such as parks, museums, and tourist attractions, to ensure mask compliance among visitors. They can also be implemented in event venues, stadiums, and concert halls to enforce mask mandates during gatherings. This helps reduce the risk of virus transmission in crowded settings.

In summary, the applications of face mask detection systems span across public health, transportation, retail, education, hospitality, manufacturing, and public spaces. These systems help enforce mask mandates, improve safety, and contribute to the overall well-being of individuals in various settings. By automating mask compliance monitoring, they offer efficient and reliable solutions to promote public health and safety.

Chapter 5
CONCLUSION

5.1 Conclusion

In conclusion, implementing a face mask detection project offers several advantages and applications in various industries and sectors. By utilizing computer vision algorithms and technology, these systems help ensure compliance with mask-wearing protocols, improve public health and safety, and mitigate the spread of infectious diseases. They automate the monitoring process, provide prompt alerts and interventions, and contribute to data collection and analysis for informed decision-making.

However, it is important to address potential disadvantages such as privacy concerns, technical limitations, bias, and implementation costs. By carefully managing these challenges and striking a balance between technology and human intervention, face mask detection systems can be effective tools in promoting a safer and healthier environment for individuals in different settings.

As the technology are blooming with emerging pandemic. So, we have created novel face mask detector which can possibly contribute to public health care department. The face mask detection is trained on CNN model and we have used OpenCV, Tensor Flow and python to detect whether person is wearing a mask or not. The model was tested with real time video as well as images and a promising accuracy is achieved and the optimization of the model is continuous process. In this project, MobileNetV2 model and a structure of SSD has been used for creating an efficient masked face recognition system.

We have benchmarked this approach on a well-known dataset. Our approach tested on those datasets shows better recognition rates. So, MobileNetV2 / structure of SSD model trained on masked and non-masked images gives better accuracy for simple masked face recognition.

A face mask detection project offers a valuable solution for ensuring compliance with mask mandates and promoting public health and safety. By leveraging computer vision algorithms and surveillance technology, these systems can automatically detect individuals who are not wearing masks or are not wearing them properly.

The advantages of implementing a face mask detection system include ensuring compliance with mask mandates, automating monitoring processes, improving safety in high-risk environments, prompt alerting and intervention, data collection and analysis, reducing confrontations, and scalability. These benefits contribute to mitigating the spread of infectious diseases and creating a safer environment for individuals.

However, it is essential to consider the potential disadvantages of face mask detection systems, such as privacy concerns, technical limitations, adaptability to new mask designs, reliance on visual detection only, potential bias and discrimination, implementation costs, and overreliance on technology. These challenges need to be addressed and carefully managed to ensure ethical and effective implementation.

Overall, a face mask detection project can be a valuable tool in various industries and sectors, including public health, transportation, retail, education, hospitality, manufacturing, and public spaces. By integrating these systems into existing infrastructure and workflows, organizations can enforce mask mandates, improve safety protocols, and contribute to the well-being of individuals in a post-pandemic world.

5.2 Future scope

The future scope for face mask detection projects is promising and encompasses several potential developments and advancements. Here are some key areas of future scope:

1. Advanced Detection Algorithms: There is room for further improvement and refinement of the detection algorithms used in face mask detection systems. Future research can focus on developing more accurate and robust algorithms that can handle challenging scenarios such as low lighting conditions, occlusions, and diverse mask designs. These advancements can enhance the overall performance and reliability of the system.

2. Integration with Emerging Technologies: Face mask detection projects can benefit from integration with emerging technologies such as artificial intelligence (AI), machine learning (ML), and deep learning. These technologies can enable more sophisticated and context-aware detection capabilities, leading to higher accuracy and better adaptability to evolving mask designs and usage patterns.

3. Real-time Analytics and Insights: The future scope for face mask detection projects includes leveraging real-time analytics and insights. By collecting and analyzing data on mask compliance rates, patterns, and trends, organizations can gain valuable insights into the effectiveness of mask mandates, identify areas of improvement, and make informed decisions regarding public health policies and interventions.

4. Mask Quality Assessment: In addition to detecting mask presence, future face mask detection systems may incorporate features to assess the quality of masks being worn. This could involve analyzing factors such as mask fit, filtration efficiency, and material composition. Such capabilities can help ensure that individuals are using masks that provide optimal protection and meet relevant quality standards.

5. Integration with Access Control and Security Systems: Face mask detection systems can be integrated with access control and security systems to enhance overall safety measures. This integration can enable seamless monitoring of mask compliance during access control processes, helping to prevent unauthorized entry and potential security breaches.

6. Multimodal Biometric Authentication: Combining face mask detection with other biometric authentication methods, such as facial recognition or fingerprint scanning, can offer enhanced security and identity verification. This multimodal approach can provide a more robust and accurate system for access control and identification purposes while simultaneously ensuring mask compliance.

7. Mobile and Wearable Applications: With the proliferation of mobile devices and wearable technologies, the future scope for face mask detection projects includes the development of mobile applications or wearable devices that can detect mask compliance. These portable solutions can enable individuals to monitor their own mask usage and receive alerts or reminders when necessary, fostering personal accountability.

8. Integration with Smart City Infrastructure: Face mask detection systems can be integrated into smart city infrastructure to create safer and more efficient urban environments. By combining face mask detection with other smart technologies, such as surveillance cameras, sensors, and data analytics, cities can proactively manage mask compliance in high-traffic areas, public transportation, and public space

REFERENCES

[1] Barabas, J., R. Zalman, and M. Kochlan. "Automated evaluation of COVID-19 riskfactors coupled with real-time, indoor, personal localization data for potential disease identification, prevention and smart quarantining." 2020 43rd International Conference on Telecommunications and Signal Processing (TSP). IEEE, 2020.

[2] Sammy v. militante, Nanettev.dionisio “Real time face mask recognition with alarmsystem using deep learning”, 2020 11th IEEE control and system graduate research colloquium.

[3] Arjya Das, Mohammad Wasif Ansari, and Rohini Basak “Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV”, 2020.

[4] K Suresh, MB Palangappa, S Bhuvan “Face Mask Detection by using Optimistic Convolutional Neural Network”, 2021 6th International Conference on Inventive Computation Technologies (ICICT).

[5] Anirudh Lodh, Utkarsh Saxena, Ajmal khan, Anand Motwani, Shakkeera L and Sharmasth Vali Y “Prototype for Integration of Face Mask Detection and Person Identification Model – COVID-19”, 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA).

[6] Samuel Ady Sanjaya, Suryo Adi Rakhmawan “Face Mask Detection Using MobileNetV2 in The Era of COVID-19 Pandemic”, 2020 International Conference onData Analytics for Business and Industry. [7] Mohamed Loey, Gunasekaran Manogaran, Mohamed Hamed N. Taha and Nour Eldeen M. Khalifa: A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the covid-19 pandemic. Measurement p.108288 (2020).

[8] Md. Rafiuzzaman Bhuiyan, Sharun Akter Khushbu and Md. Sanzidul “A Deep Learning Based Assistive System to Classify COVID-19 Face Mask for Human Safety with YOLOv3”, 2020 11th International Conference on Computing, Communication and Networking Technologies.

[9] Mohammad Marufur Rahman, Md. Motaleb Hossen Manik, Md. Milon Islam, Saifuddin Mahmud, Jong Hoon Kim, “An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network”, IEEE Xplore, 18 October 2020.

APPENDIX