

IOT BASED FACE MASK DETECTION

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

HEENA PRAVEEN (Reg. No - 40110452)
SHAIK YASEEN AHMAD (Reg. No - 40111469)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Category-1 University by UGC

Accredited with Grade "A++" by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHISALAI,

CHENNAI - 600119

April - 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Category - I University by UGC

Accredited "A++" by NAAC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Heena Praveen (40110452)** and **Shaik Yaseen Ahmad (40111469)** who carried out the Project Phase work entitled "IOT BASED FACE MASK DETECTION" under my supervision from November 2023 to April 2024.

Internal Guide

SONIA JENIFER RAYEN M.Tech., Ph.D

HEAD OF THE DEPARTMENT
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Dr. L. LAKSHMANAN, M.E., Ph.D
(Jeppiaar Nagar, Rajiv Gandhi Salai,
Chennai - 600 119.)

Submitted for Viva Voce Examination held on 16/4/2024

Wai
16/04/24

Internal Examiner

16/4/24

External Examiner

DECLARATION

I, **HEENA PRAVEEN (40110452)**, hereby declare that the Project Report entitled "**IOT BASED FACE MASK DETECTION**" done by me under the guidance of **Dr.Sonia Jenifer Rayen M.Tech., Ph.D**, is submitted in partial fulfillment of the requirements for the award of **Bachelor of Engineering degree in Computer Science and Engineering**.

DATE: 15/4/2024

PLACE: Chennai



SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey our thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr.Sonia Jenifer Rayen M.Tech., Ph.D**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

With the world's population growing, distractions can cause dangerous accidents that are . Several studies have been done in an According to data obtained by the World Health Organization, the global pandemic of COVID-19 has severely impacted the world and has now infected more than eight million people worldwide. The rise of COVID-19 pandemic has had a lasting impact in many countries worldwide since 2019. Face-mask detection had been significant progress in the Image processing and deep learning fields studies. Many face detection models have been designed using different algorithms and techniques. The proposed approach in this project is developed to avoid mask-less people from entering the desired places (i.e., Mal by detecting face-mask using deep learning, TensorFlow, and OpenCV and sending a signal to Arduino device that connected to the sensors and counts the peoples while entering and exiting. This helps to maintain the social distancing inside those places and limit the Total number of persons. If the total number of people count increased, the Arduino micro controller will give the buzzer alarm to notify the person. In This project, we detect the faces in real-time and identifies whether the person wears a mask or not. According to data obtained by the World Health Organization, the global pandemic of COVID19 has severely impacted the world and has now infected more than eight million people worldwide. The rise of COVID-19 pandemic has had a lasting impact in many countries worldwide since 2019. Face-mask detection had been significant progress in the Image processing and deep learning fields studies. Many face detection models have been designed using different algorithms and techniques. The proposed approach in this project is developed to avoid mask-less people from entering the desired places (i.e., Mall, University, Office, ...etc.) face-mask using deep learning, TensorFlow, Keras, and OpenCV and sending a signal to Arduino device that connected to the sensors and counts the peoples while entering and exiting.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	viii
1.	INTRODUCTION	1
2.	LITERATURE SURVEY	9
	2.1 Inferences from literature Survey	9
	2.2 Open problems in Existing System	10
3.	REQUIREMENTS ANALYSIS	
	3.1 Feasibility studies / Analysis of the project	15
	3.2 Software Requirements Specification	17
4.	DESCRIPTION OF PROPOSED SYSTEM	
	4.1 Selected Methodology or process model	19
	4.2 Architecture/ Overall Design of Proposed Model/System	20
	4.3 Description of Software for Implementation And testing plan of the Proposed Mo	21
	4.4 Project Management Plan	23
	4.5 Transition/Software to Operation Plan	23
5.	RESULT	29
6.	SUMMARY AND CONCLUSION	33
	REFERENCES	34

APPENDIX

A. Source Code	38
B. Screenshots	48
C. Research paper	52

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
4.1	Database and input image	11
4.2	System Architecture of face-mask Detection	12
4.3	Images without mask (Kaggle 2022)	13
4.4	Images with mask (Kaggle 2022)	14
4.5	The training of data of data face masks and Non masks	15
4.6	Arduino Uno	12
4.7	LCD Display Board	13
4.8	Relay Module	14
4.9	Infrared Sensors	14
5.1	Real Time Analysis	25
5.2	Green Signal	26
5.3	Python code for fragment	27
5.4	Total percent wore mask (output)	28

CHAPTER 1

INTRODUCTION

We are in a world of digital transformation. In every aspect technology is one common thing people depend upon. If we look back in the 1970s, people used to count visitor manually by counting them or they used a manual tally counter. But today we can see that many methods have been introduced to count people without the need of any human presence. The sensors and cameras will simplify our job counting the people. We just need to program them to perform the required task. The primary method of counting visitors include hiring people to stand and manually count the number of guests or workers who enter or exit from the venue or location. Even the tally counters are not user friendly and don't have many advantages. Therefore, these methods prove to be unreliable and come at a great cost. The demand for the electronic device which can control the room appliances has a great surge such that it can be implemented in many real time applications like in hotels, living room, garage and so on. The model can track the number of persons entering and leaving the room and also switches the lights and fans on and off if the room is engaged or vacant respectively. By employing this device in the room reduces the laborious works to search for the switch to light the room at once you entered. The persons entering the room through the entrance will be sensed by the Infrared Sensors (shortly IR sensors) and the signal sensed is sent to the Arduino UNO for processing and controlling the count in the room and also explicitly monitors the lights and fan in the room. Prior to the coronavirus disease 2019 (covid-19) pandemic, there was no concrete evidence supporting the use of community masks to reduce the spread of respiratory infections. Masks are primarily intended to prevent the wearer from spreading the viral droplets (source control). Covid-19 and other respiratory infections spread primarily through inhalation of respiratory aerosols produced by coughing, sneezing, talking, or breathing. The virus propagates and migrates down the respiratory tract and may lead to pneumonia, acute respiratory distress syndrome (ARDS) and even death. Scientists have attempted to build automated facial mask recognition systems in public locations to ensure the use of face masks in common areas.

Following the COVID-19 epidemic, other researchers developed their own techniques for monitoring face masks in common areas.

Employing image processing algorithms, Surveillance systems are utilized for monitoring of public spaces in order to guarantee that no one's face is visible in crowded locations. Deep learning-based approaches for object identification and imagery analytics have been increasingly popular over the years. Most of the past research has been conducted using convolutional neural network models. There are two instances in which current face mask detection algorithms are unable to reliably identify the masks. When there are many people in a single image or video frame, it is difficult to precisely identify all of the faces “with mask and without mask.” In our nation, ladies wear half-faced veils that serve the same purpose as face masks, but the current methods do not identify them as face masks. How to construct a more efficient and accurate classification approach is a key aspect for the implementation of facial mask detection techniques in mobile environment. However, several deep learning models are costly and time-consuming in their evaluation steps, making them unsuitable for mask detection in the facial image paradigm in a mobile environment. In order to overcome the shortcomings of the existing approach, the suggested method makes use of Depth wise Separable Convolutions with Mobile Net for mask detection in facial images. Depth-wise separable convolution (DSC) was first proposed in [1] and is now widely used in image processing for classification tasks

CHAPTER 2

LITERATURE SURVEY

Anjali K, Reshmi Anand, Sradha D Prabhu, Geethu R.S. There is a rapid increase in the spread of the CovidVirus.

This has resulted in serious health issues among people. There is a dire need for a proper system to detect COVID symptoms and Alert Officials. The proposed solution is an IoT based multiple Health Monitoring System to detect symptoms of COVID and alert officials. It consists of a contactless temperature sensor, Oximeter, Automatic Hand Sanitizer, Camera module, Room Occupancy limiter and a Social

Distancing module. It is a contactless method for patient evaluation. The Internet of Things enables devices to connect to the internet and communicate with each other. Data from these IoT-connected devices will help to identify the scenario and take necessary.

Contactless Temperature Sensor, Oximeter, and Heartbeat rate sensors are used to measure temperature, Oxygen level, and Heart rate respectively. NodeMCU reads sensor values, ESP32 camera module capture picture, and stores for future reference. The sensors measure the data. Normal temperature reading is from 97°F to 99°F and normal SpO2 reading is at least 95% [1]. If the measured value crosses the normal value, those details are immediately sent to health workers with a picture of the individual. The proposed solution also consists of a module to maintain Social Distancing and also an automatic Hand Sanitizing module. As the entire setup is automated, it reduces time and labour and also ensures satisfactory results.

Keywords: Internet of Things, Covid, Smart Healthcare, Arduino, NodeMCU, Oximeter, Sensor, Automatic Hand Sanitizer, Temperature Sensor, Social distancing, Data collection

Constantinescu and Rainer Stiefelhagen The current COVID-19 pandemic is having a major impact on our daily lives.

Social distancing is one of the measures that has been implemented with the aim of slowing the spread of the disease, but it is difficult for blind people to comply with this. In this paper, we present a system that helps blind people to maintain physical distance to other persons using a combination of RGB and depth cameras. We use a real-time semantic segmentation algorithm on the RGB camera to detect where persons are and use the depth camera to assess the distance to them; then, we provide audio feedback through bone-conducting headphones if a person is closer than 1.5 m. Our system warns the user only if persons are nearby but does not react to non-person objects such as walls, trees or doors; thus, it is not intrusive, and it is possible to use it in combination with other assistive devices. We have tested our prototype system on one blind and four blindfolded persons, and found that the system is precise, easy to use, and amounts to low cognitive load.

Keywords: computer vision for the visually impaired; social distancing; semantic segmentation.

Mustafa Musa Jaber, Thamer Alameri , Mohammed Today, COVID-19-patient health monitoring and management are major public health challenges for technologies.

This research monitored COVID-19 patients by using the Internet of Things. IoT-based collected real-time GPS helps alert the patient automatically to reduce risk factors. Wearable IoT devices are attached to the human body, interconnected with edge nodes, to investigate data for making health condition decisions. This system uses the wearable IoT sensor, cloud, and web layers to explore the patient's health condition remotely. Every layer has specific functionality in the COVID-19 symptoms' monitoring process. The first layer collects the patient health information, which is transferred to the second layer that stores that data in the cloud. The network examines health data and alerts the patients, thus helping users take immediate actions. Finally, the web layer notifies family members to take

appropriate steps. This optimized deep-learning model allows for the management and monitoring for further analysis.

Keywords: COVID-19; cloud computing; deep learning; healthcare data; IoT sensors; wearable sensors.

Nidal Nasser ,Qazi Emad-ul-Haq, Muhammad Imran Coronavirus (COVID-19) is a very contagious infection at has drawn the world's attention.

Modelling such diseases can be extremely valuable in predicting their effects. Although classic statistical modelling may provide adequate models, it may also fail to understand the data's intricacy. An automatic COVID-19 detection system based on computed tomography (CT) scan or X-ray images is effective, but a robust system design is challenging. In this study, we propose an intelligent healthcare system that integrates IoT- cloud technologies. This architecture uses smart connectivity sensors and deep learning (DL) for intelligent decision-making from the perspective of the smart city. The intelligent system tracks the status of patients in real time and delivers reliable, timely, and high-quality healthcare facilities at a low cost. COVID-19 detection experiments are performed using DL to test the viability of the proposed system. We use a sensor for recording, transferring, and tracking healthcare data. CT scan images from patients are sent to the cloud by IoT sensors, where the cognitive module is stored. The system decides the patient status by examining the images of the CT scan. The DL cognitive module makes the real-time decision on the possible course of action. When information is conveyed to a cognitive module, we use a state-of-the-art classification algorithm based on DL, i.e., ResNet50, to detect and classify whether the patients are normal or infected by COVID-19. We validate the proposed system's robustness and effectiveness using two benchmark publicly available datasets (Covid-Chestxray dataset and Chex-Pert dataset). At first, a dataset of 6000 images is prepared from the above two datasets. The proposed system was trained on the collection of images from 80% of the datasets and tested with 20% of the data. Cross-validation is performed using a tenfold cross-validation technique for performance evaluation. The results indicate that the proposed system gives an accuracy of 98.6%, a sensitivity of 97.3%, a

specificity of 98.2%, and an F1score of 97.87%. Results clearly show that the accuracy, specificity, sensitivity, and F1-score of our proposed method are high. The comparison shows that the proposed system performs better than the existing state-of-the-art systems. The proposed system will be helpful in medical diagnosis research and healthcare systems. It will also support the medical experts for COVID-19 screening and lead to a precious second opinion.

Keywords: Coronavirus , Internet of things, Cloud computing, Detection, Machine learning ,Deep learning.

R. Dhruv Seshadri , Evan V. Davies , Ethan R. Harlow, Jeffrey J. Hsu,Shanina C. Knighton, Timothy A. Walker , James E. Voos and Colin

The COVID-19 pandemic has brought into sharp focus the need to harness and leverage our digital infrastructure for remote patient monitoring. As current viral tests and vaccines are slow to emerge, we see a need for more robust disease detection and monitoring of individual and population health, which could be aided by wearable sensors. While the utility of this technology has been used to correlate physiological metrics to daily living and human performance, the translation of such technology toward predicting the incidence of COVID-19 remains a necessity. When used in conjunction with predictive platforms, users of wearable devices could be alerted when changes in their metrics match those associated with COVID-19. Anonymous data localized to regions such as neighbourhoods or zip codes could provide public health officials and researchers a valuable tool to track and mitigate the spread of the virus, particularly during a second wave. Identifiable data, for example remote monitoring of cohorts (family, businesses, and facilities) associated with individuals diagnosed with COVID-19, can provide valuable data such as acceleration of transmission and symptom onset. This manuscript describes clinically relevant physiological metrics which can be measured from commercial devices today and highlights their role in tracking the health, stability, and recovery of COVID-19+ individuals and front-line workers. Our goal disseminating from this paper is to initiate a call to action among front-line workers and engineers toward developing digital health platforms for monitoring and managing this pandemic.

Keywords: wearable sensors, COVID-19, pandemic, predictive analytics, remote patient monitoring.

XINQI FAN , MINGJIE JIANG, AND HONG YAN , (Fellow, IEEE) Coronavirus disease 2019 has seriously affected the world.

One major protective measure for individuals is to wear masks in public areas. Several regions applied a compulsory mask-wearing rule in public areas to prevent transmission of the virus. Few research studies have examined automatic face mask detection based on image analysis. In this paper, we propose a deep learning based single-shot light-weight face mask detector to meet the low computational requirements for embedded systems, as well as achieve high performance. To cope with the low feature extraction capability caused by the light-weight model, we propose two novel methods to enhance the model's feature extraction process. First, to extract rich context information and focus on crucial face mask related regions, we propose a novel residual context attention module. Second, to learn more discriminating features for faces with and without masks, we introduce a novel auxiliary task using synthesized Gaussian heat map regression. Ablation studies show that these methods can considerably boost the feature extraction ability and thus increase the mask detection performance. Comparison with other models shows that the proposed model achieves state-of-the-art results on two public datasets, the AIZOO and Moxa3K face mask datasets. In particular, compared with another light-weight you only look once version 3 tiny model, the mean average precision of our model is 1.7% higher on the AIZOO dataset, and 10.47% higher on the Moxa3K dataset. Therefore, the proposed model has a high potential to contribute to public health care and get against the coronavirus disease 2019 pandemic.

Keywords: Face mask detection, residual context attention, synthesized Gaussian heat map regression, coronavirus disease 2019.

Ziwei Song , Kristie Nguyen , Tien Nguyen , Catherine Cho and Jerry Gao – boyencers

According to the World Health Organization (WHO), wearing a face mask is one of the most effective protections from airborne infectious diseases such as COVID-19. Since the spread of COVID-19, infected countries have been enforcing strict mask regulation for indoor businesses and public spaces. While wearing a mask is a requirement, the position and type of the mask should also be considered in order to increase the effectiveness of face masks, especially at specific public locations.

However, this makes it difficult for conventional facial recognition technology to identify individuals for security checks. To solve this problem, the Spartan Face Detection and Facial Recognition System with stacking ensemble deep learning algorithms is proposed to cover four major issues: Mask Detection, Mask Type Classification, Mask Position Classification and Identity Recognition. CNN, AlexNet, VGG16, and Facial Recognition Pipeline with FaceNet are the Deep Learning algorithms used to classify the features in each scenario. This system is powered by five components including training platform, server, supporting frameworks, hardware, and user interface. Complete unit tests, use cases, and results analytics are used to evaluate and monitor the performance of the system. The system provides cost-efficient face detection and facial recognition with masks solutions for enterprises and schools that can be easily applied on edge-devices.

Keywords: COVID-19; masked face; facial recognition; deep learning; ensemble model.

2.1 INFERENCES FROM LITREATURE SURVEY

However, in mainly most of the discussed model, there are some flaws found. There is no regulatory authority to restrict invalid voters from casting a vote, and it is not secure from quantum attack. Their model is not accurate and did not care about voter's integrity. Moreover, their scheme using Distributed consensus in which testimonies (data and facts) can be organized into cartels because fewer people keep the network active, a "51%" attack becomes easier to organize. This

attack is potentially more concentrated and did not discuss scalability and delays in electronic voting, which are the main concerns about the blockchain voting system. They have used the Multichain framework, a private blockchain derived from Bitcoin, which is unsuitable for the nationwide voting process. As the authors mentioned, their system is efficient for small and medium-sized voting environments only. Furthermore, Solidity does not support elliptic curve cryptography, they used an external library to do the computation. After the library was added, the voting contract became too big to be stored on the blockchain. Since it has occurred throughout the history of the Bitcoin network, OVN is susceptible to a denial-of-service attack. Table 3 shows the main comparison of selected electronic voting schemes based on blockchain. The properties described above are the bare minimum for any voting solution.

But all the technologies mentioned above are useless if there is no trust in the system itself. A voting system needs to be fully verifiable to earn this trust, i.e., everyone involved can ensure that the system complies with the stated properties. Ensuring verifiability can be split into two tasks: personal, when the voter can verify that their ballot is correctly recorded and tallied; and universal, when everyone can prove that the system as a whole works precisely. This entails the inputs and outputs of the voting protocol stages being published and proof of correct execution. For example, mix-networks rely on proof of correct shuffling (a type of zero-knowledge proof), while proof of correct decryption is also used in mix-networks and threshold decryption.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

Real-time data collection is critical in the field of human health. This paper outlines a method for tracking a human being's heartbeat rate per sec and detecting normal body temperature from a distance. The information was gathered from a group of volunteers, and the device was tested with sensors created by the research team. The Arduino microcontroller is designed to send data over the

XBee wireless network to a remote PC station for display and storage. It reduces device power consumption by activating the sensor with a remote-control command from the receiving PC. The performance of a Wireless Sensor Network in rising applications including weapon sensor ships, medical applications, habitat monitoring, and seismic surveillance was investigated. WSN has recently concentrated on domestic sequences and market applications. The efficiency of the PIC-created WSN models is demonstrated in this project. To establish sensing phenomena, the normal nodes of temperature sensors were used for networks. Here, findings show that the time setting has a significant impact on the sensor node's efficiency. The purpose of this document is to identify and briefly explain the critical factors and issues that affect WSN output. Controlling laboratory measurements and clinical trials limit the realism and duration of various tests.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

To give a clear view of existing methods, a hierarchical representation is outlined in. According to the used features, all methods are divided into two classes: Handcrafted Feature-based methods, Neural Network-based methods. Handcrafted Feature-based methods are also usually regarded as conventional methods. They can be further classified as two categories in accordance with the number of detectors: single-detector methods and multiple-detector methods. Most detectors depend on AdaBoost algorithm. Different detectors, for example, face detector, facial mask detector, nose detector, mouth detector, nose and mouth detector, and eye detector, are selected or combined together. According to the number of stages, the methods can be classified as three categories: single-stage methods, two-stage methods, and multi-stage methods. For single-stage methods, they are mainly implemented by transfer learning of object detection algorithms. For example, YOLO series methods: YOLO, YOLOv2, YOLOv3, YOLOv4, YOLOv5, and corresponding to tiny versions. For two-stage methods, they can be further divided into three kinds referring to the use of neural network: Neural Network + Neural Network, Neural Network + Hand-crafted Feature, and Hand-crafted Feature + Neural Network. Twostage methods consist of two parts.

Project Description and Objectives: The main motive of this project is to build a advance level of key logger. - The key logger is not to record only the keystroke but also to record the work of the mouse also.

Market Analysis: This key logger is used are used by: Parents Organization Hackers

Technical Feasibility: the technical resources to record the keystroke is python interface to the run the application. - the technical resource to display the recorded keystroke is Linux terminal.

Operational Feasibility: it will be useful for the monitoring purpose of an individual by an authorized person of the organization or by the parent

Legal and Ethical Feasibility: The punishment for hacking is given in Section 66 of the IT Act, 2000. The Section also enumerates the essential ingredients of hacking. These areas follows: There must be an intention to cause wrong to a person. - Evaluating the ethical implications of a key logger attack is important because it involves serious ethical and legal concerns. Key loggers are typically associated with malicious intent, and their use can lead to various ethical issues and potential harm.

Risk Analysis: Key logger attacks, like many forms of cyber attacks, come with several potential risks and consequences, both for the attacker and the victim. - Mitigating the high-impact risks associated with a key logger attack requires a multi-faceted approach that focuses on preventing, detecting, and responding to such attacks..

Decision-Making: This project is to used only for monitoring the individual by an authorized person Remember, a feasibility study and risk analysis are essential tools for responsible project management. They help ensure that projects are well-planned and have a higher chance of success while minimizing potential negative impacts.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

- Operating System - Windows or Linux or FreeBSD or MacOS
- Platform - x86 – 64
- Language - Python, Arduino

- Libraries - TensorFlow, OpenCV, Scikit-learn, NumPy
- CNN Model – MobileNetV2

Python: Python is a scripting language that is high-level, interpreted, interactive, and object-oriented. Python is intended to be extremely readable. It comes only with English terms rather than punctuation, and it has fewer syntactical structures than other languages.

Easy-to-learn: Python has a small number of keywords, a basic structure, and a well-defined syntax. This enables the pupil to swiftly learn the language.

Easy-to-read: Python code is more clearly defined and visible to the eyes.

Easy-to-maintain: Python's source code is relatively simple to maintain.

- λ **A broad standard library** - The majority of Python's library is portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- λ **Interactive Mode** - Python features an interactive mode that enables for interactive testing and debugging of code snippets.
- λ **Portable** - Python can operate on a broad range of hardware devices and has the same interface across them all.
- λ **Extendable** - The Python interpreter may be extended using low-level modules. These modules allow programmers to enhance or adapt their tools to make them more efficient.
- λ **Databases** - Python provides interfaces to all major commercial databases.
- λ **GUI Programming** - Python can construct and port GUI programmes to numerous system calls, libraries, and windows systems, including Windows MFC, Macintosh, and Unix's X Window system.
- Scalable** - Python provides a better structure and support for large programs than shell scripting.

Oracle VM Virtual Box is a free and open-source virtualization software that allows you to run multiple operating systems on a single computer. With Virtual Box, you can create virtual machines that run different operating systems like Windows, Linux, and macOS on your host machine. Virtual Box enables you to install and run different operating systems as guests, allowing you to test software, run applications or perform tasks on different environments without affecting your primary operating

system. The virtual machines created can have their own settings, including virtual hardware components such as network adapters, hard disks, and CPUs.

Ubuntu: As of my knowledge cutoff date of September 2021, there was no release of Ubuntu 22.04. However, Ubuntu 22.04 is expected to be released in April 2022 according to Ubuntu's release cycle. Ubuntu 22.04 is expected to be a Long Term Support (LTS) release, meaning that it will receive support and updates for a period of 5 years. It is likely to come with new features and improvements, as well as updated software packages. Some of the expected changes and features in Ubuntu 22.04 may include:

- Updated software packages, including the latest versions of the Linux kernel, GNOME desktop environment, and other key software components.
- Improved performance and stability, with optimizations for faster boot times and better system responsiveness.
- Enhanced security features, including updated encryption and authentication mechanisms, and improved security hardening.
- Improved support for hybrid and cloud-native environments, with better integration with Kubernetes and other cloud platforms.

3.3 PROJECT MANAGEMENT PLAN

A Project Management Plan (PMP) is a comprehensive document that outlines how a project will be planned, executed, monitored, and controlled. It serves as a guide for the entire project team and stakeholders, ensuring that everyone has a common understanding of the project's objectives, scope, timeline, resources, and responsibilities. Here's a typical structure for a Project Management Plan:

Project Introduction: The Keylogger Implementation Project is a software development initiative aimed at creating a keylogger application. This project has a specific focus on keyloggers that are intended for legitimate and ethical purposes within an organization or specific context. A keylogger, in this context, is a software tool designed to record keystrokes on a computer or mobile device. The primary objective is to develop a keylogger that enhances security, productivity, and user experience within the designated context.

Communication Plan: Effective communication with stakeholders is crucial for the success of any project, including a Keylogger Implementation Project. Here's a plan on how project information can be communicated to stakeholders:

CHAPTER 4

4.1. DESCRIPTION OF PROPOSED SYSTEM

The approach of this project is to make use of internet and use all appliances in the best Way possible , I tried to make use of device for sensing which will make human efforts less. The demand for the electronic device which can control the room appliances has a great surge such that it can be implemented in many real time applications like in hotels, living room, garage and so on. The model can track the number of persons entering and leaving the room and also switches the lights and fans on and off if the room is engaged or vacant respectively. By employing this device in the room reduces the laborious works to search for the switch to light the room at once you entered. The persons entering the room through the entrance will be sensed by the Infrared Sensors (shortly IR sensors) and the signal sensed is sent to the Arduino UNO for processing and controlling the count in the room and also explicitly monitors the lights.

4.1.1. HARDWARE USED -

A. Arduino UNO:

Arduino UNO is an interactive board which receive various signals as input, it operates different operations based the signal given to Arduino UNO. It has 13 digital pins and 6 analogue pins. A typical Arduino UNO is given in the figure





Fig. 4.1 Arduino UNO

Digital pins are connected to the digital output and the analogue pins are used to get basic input signals. It has a 32kb of flash memory to store the program that is coded for the operation. It is connected to the system using a B-type cable. The software used to code is Arduino UNO 1.8.8 as the latest version.

B. LCD – display board:

Liquid Crystal Display board of dimension 16x2 where it can display of 32 characters in two lines each of 16 characters. It has 16 external holes for connections as shown in the figure 2. These output holes can be connected to the bread board with the help of male to male breakable pins. It seeks a voltage of 5V from the source provided to the Arduino UNO board. It has three ground connections.





Fig .4.2 LCD Display board

C. 5v – relay module:

Relay module is a magnetically operated switch which closes the external circuit based on signal received from the Arduino UNO board. The relay seeks voltage from the source to induce a magnetic field which attracts a metal strip and closes the external circuit; at once it receives the positive signal from the Arduino UNO.

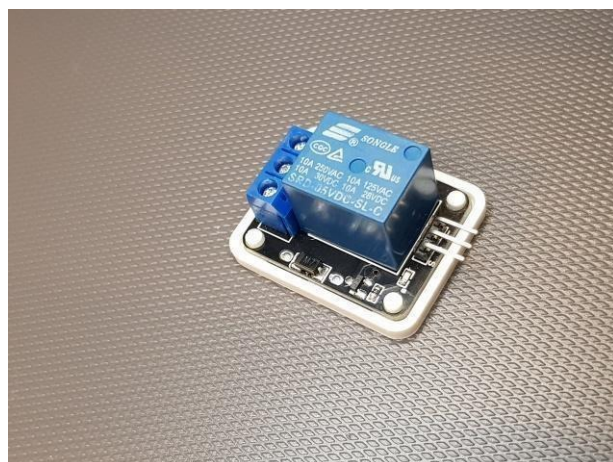


Fig.4.3 Relay Module

There are three external pins available in the relay module to connect to the external circuit as given in the figure 3. The middle point is always the ground, the positive line of the external circuit can be to the either of the two pins based on the normally open and normally closed.

D. Infrared sensors:

Obstacle detection sensor generates a signal when any object encounters it. It has transmitter and receiver at its end. They send analogue signals to the Arduino UNO for processing. It has two extrapolation facing outwards they are receiver and transmitter as shown in the figure 4. Black coloured knob is the transmitter whereas the white coloured is receiver. It has three pins to get connected they are for ground, Vcc and sending output signal.



Fig.4.4 Infrared Sensors

E. Power source:

Power source for the working of Arduino UNO can be given through the 9V DC battery with the help of power cord whereas the alternative one is the connection of Arduino UNO to the laptop using B type cable. During developing stages it will be easier to connect with B type cable, but as an end product it is advised to use DC battery. It requires a maximum of 5V, in the real time model the power can be given from the AC socket by transforming it to pulsating DC and rectifying to DC.

F. Potentiometer:

It is the device which is used to provide variable resistance to the circuit. In the LCD board, the third pin is facilitated for contrast adjustment by providing the necessary resistance to it along with the input voltage. It has three external pins to get connected to the circuit, one pin to the resistor terminal and other two pins are connected to both ground and VCC.

4.1.2. SOFTWARE REQUIREMENTS—

Arduino IDE

Python IDLE

Programming Language—

Embedded 'C'

Python

4.1.3. SELECTED METHODOLOGY OR PROCESS MODEL

In Proposed System, it has now been proved that not wearing a mask will not only affect our health but will also affect others health if they come in close contact with us and as per World Health Organization and Indian Govt Laws to protect others and ourselves this proposed model will use the camera to monitor if people are wearing a mask or not. This method is been designed by Python that is connected by a USB to an Arduino UNO microcontroller. It requires to have PySerial, which is a Python API module uses for reading and writing serial data from Arduino or any other Microcontroller.

4.1.4.METHODOLOGY-

In one embodiment is a method of providing postural feed back comprising: receiving by a microprocessor at repeated intervals data from a tri-axial accelerometer, the microprocessor and the tri-axial accelerometer comprising a sensor device attached to a user, the sensor device further comprising memory, an actuator, and a power Source; normalizing by the microprocessor the received accelerometer data; determining by the microprocessor a postural description of the user based on the normalized received accelerometer data; and triggering by the microprocessor the actuator to output sensory feedback based on the postural description of the user. In another embodiment is a postural feedback apparatus comprising: a sensor device configured to be attached on a user, the sensor device comprising: a tri- axial accelerometer; an actuator, and a microprocessor configured to receive data from the tri-axial accelerometer about movement of the user; normalize the received accelerometer data; determine a postural description of the user based on the normalized received accelerometer data; and trigger the actuator to output sensory feedback based on the postural description of the user.

A. Search Strategy

For selecting the primary articles, the major databases such as IEEE Xplore, Springer Link, ACM digital library, Science Direct, and Google Scholar were searched for related articles. The articles were searched using several keywords related to 'COVID-19' (e.g. 'CoV2' or 'Coronavirus' or 'COVID-19') and keywords related to the machine learning, artificial intelligence and deep learning, for example, ('Machine learning' and 'CoV2' or 'Coronavirus' or 'COVID-19'), ('Artificial intelligence' and 'CoV2' or 'coronavirus' or 'COVID-19'), ('Deep learning' and 'CoV2' OR 'coronavirus' or 'COVID-19'), ('Machine learning' AND 'influenza' and 'CoV2' or 'Coronavirus' or ('Machine learning' and 'lockdown' and 'pandemic' and 'CoV2' or 'coronavirus' or 'COVID-19')).

B. Inclusion and Exclusion Criteria

The criteria for selecting an article were as follows: (a) The article is written in English, and (b) the article adopts, develops, or proposes AI and ML techniques,

algorithms, systems, methods, or applications to fight the COVID-19 pandemic. In addition, we used the following exclusion criteria: (a) duplicate articles that are found through several scholarly databases; (b) articles that are not focused on our research objectives; (c) articles that are not written in English; and (d) earlier versions of any article that has been published on the same set of data exploring the same objective.

C. Study Selection

The search results comprised 913 articles from the stated databases. The Prisma flowchart in shows the article selection process in different phases following the exclusion inclusion criteria. From 913 articles, the study selection process was initiated by removing duplicated articles, and a total of 619 were excluded during the first round of screening. Next, articles written in English were screened based on their abstract, and in some cases the introduction, to assess their compatibility with the inclusion criteria.

D. Data Extraction

Data extraction was conducted to explore different AI and ML techniques, algorithms, and systems and their implications for fighting the COVID-19 pandemic. Thus, the selected articles were read meticulously to collect and extract research data and construct the review article. The entire research process was supervised and monitored by two senior researchers with expertise in AI, ML, and health informatics to ensure the reliability and quality of this review article.

4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

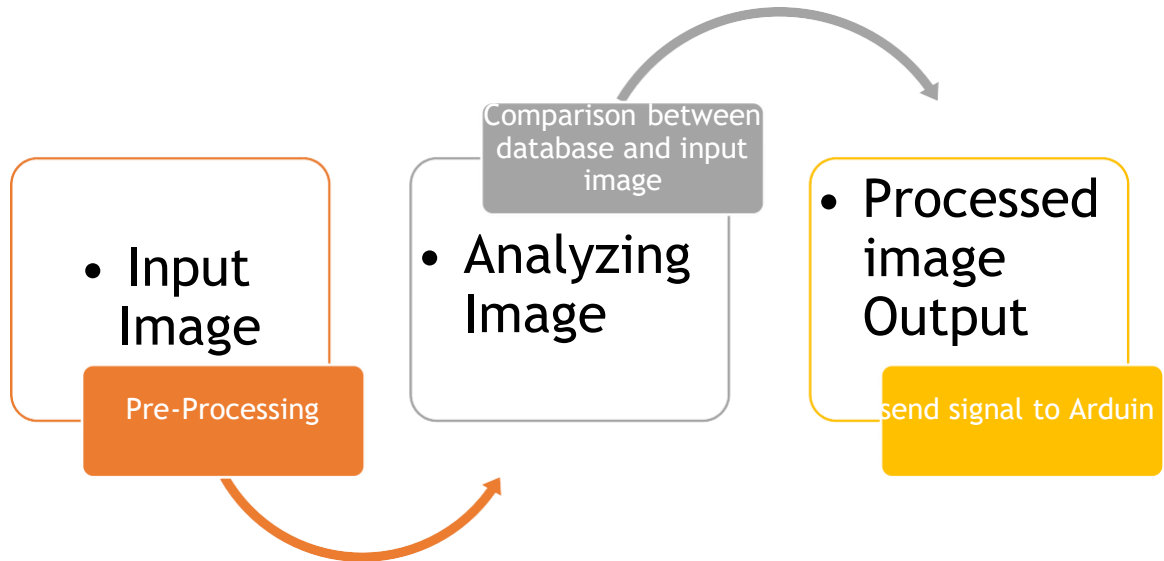


Fig.4.5: Database and input image

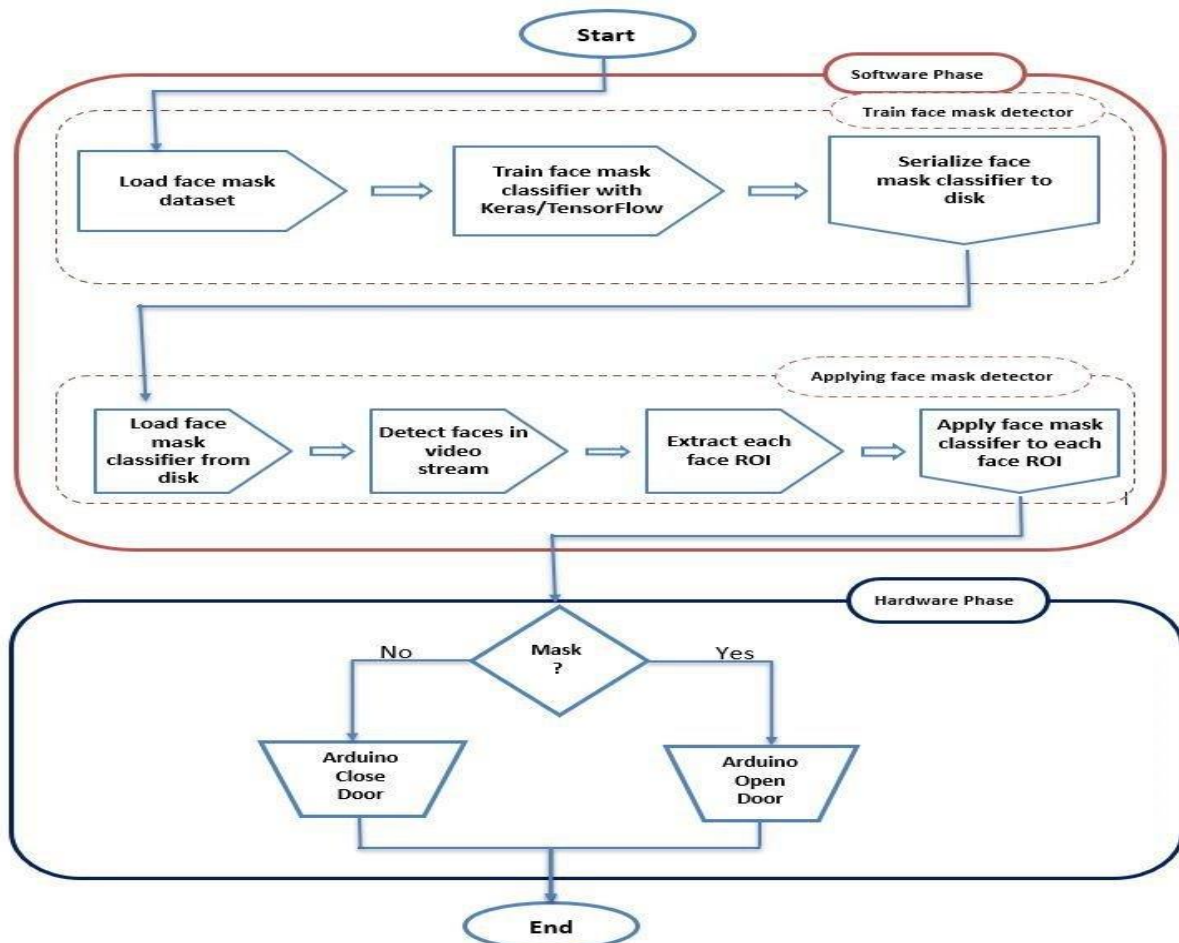


Fig.4.6 Flow Diagram

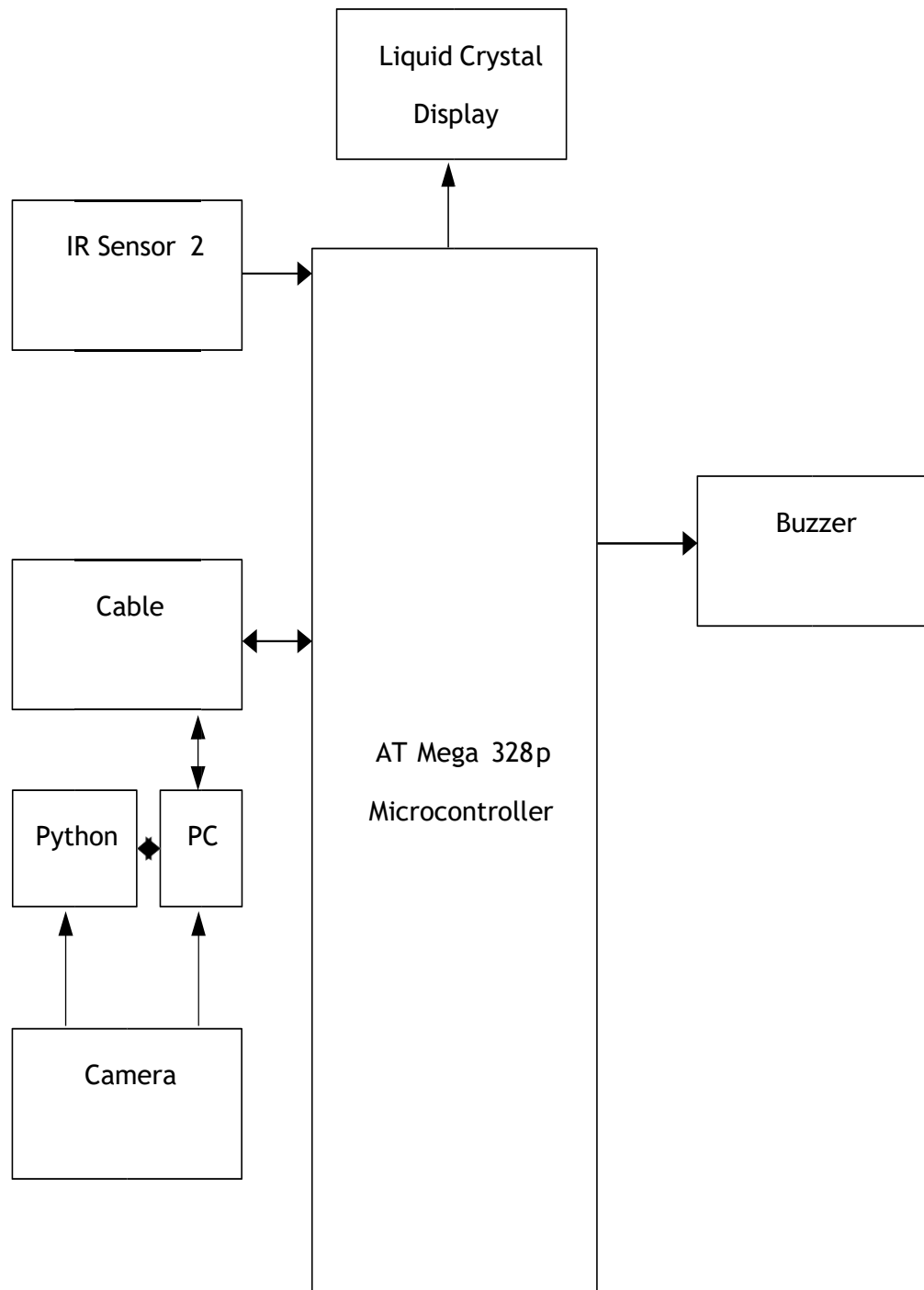


Fig 4.7: System Architecture of face-mask Detection

The block diagram of the proposed system has been shown in the above figures. The camera captures the image of the person inside the car and sends that to the HOG model to train and it detects each feature from the face using facial landmark technique in the system. The next step in the process would be as it starts to find the position and condition of each feature to analyze, it should detect whether the person is sleeping or not. If any of the features especially the eye and the head pose/state of the person are detected to be abnormal then automatically the system starts to produce the siren sound. This is where we will get the final judgment of the state of the driver, whether he is concentrated or distracted.

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

Two image datasets were collected from the kaggle website for the model's training and testing purposes. (Kaggle 2022.) Following that, image dataset was split into two groups: one Images_with_mask and one Images_ without_mask. More images have also been added to the dataset, which were taken with the laptop's default camera. Using these datasets, it is possible to develop a model that can distinguish between people wearing masks and people who are not wearing masks. These datasets contain around 1400 images from the folders "Images_with_mask" and "Images_without_mask". Images with mask and without mask can be seen in Figures 4.3.1 and 4.3.2.



Fig.4.8. Images without mask (Kaggle 2022)

Different types of people's facial images are displayed above the image figure. These images were taken from kaggle's website. The dataset also contained images taken by author using his computer's default camera and added to the dataset. These images were utilized to achieve the purpose of model training.



Fig.4.9 Images with mask (kaggle 2022)

The datasets also contain images with mask. These have also been collected from kaggle website. For data training, pictures were taken from different viewpoints and by different people. Author included a couple of his own photos in the image datasets, which he took with the computers default camera.

Data set comprises 7553 RGB images in mask and without mask in the two folders. Images are also referred as an mask and mask-free images. Mask pix are 3725 face photo and 3828 faceless mask images. N order to educate a custom face mask detector , we want to break our challenge into distinct levels, every with its very own redetection dataset from the disk, schooling a model(the usage of keras/Tener waft) on this dataset, and serialise the masks detector to disk. 2> Deployment: When the face mask detector is skilled, we are able to then flow on to the loading the masks detector, acting face detection, after which it classifying each face as with mask or without masks.

- 1 Eyes
- 2 Eyebrows
- 3 Nose
- 4 Mouth
- 5 Jawline To use facial landmarks to construct a data set of face carrying masks.

4.4 PROJECT MANAGEMENT PLAN

Single shot detector structure can be used for object detection. In this device, face masks detector can be evolved in lots of regions like shopping branch stores , airports and heavy website site visitors locations to screen the general public and to avoid the spread of the disease with the resource of checking who's following primary rule and who isn't always. It takes excessive time in load Google Colab Notebook.We re processed a device computationally efficient the usage of theMobileNetV2 which makes smooth to Extract the Data set.

1. We can use CNN structure for higher performance.
2. We can repair it any type of cameras
3. Each image is divided into three bins Of 1*1,2*2,3*3 size.
4. We can use that for VGG-f for v-support regression.

5. We can use MobileNet model for predict the class of images and the fast..

4.5 TRANSITION/ SOFTWARE TO OPERATIONS PLAN

To recognize human faces autonomously, it uses sophisticated deep learning algorithm relying on machine vision. A multiscale rapid RCNN procedure depending on upper or lower layers (UPL-RCNN) is suggested to purely recognize a range of human faces. The +e network is made up of components that perform spatial affine transformations and feature region constituents(ROI).

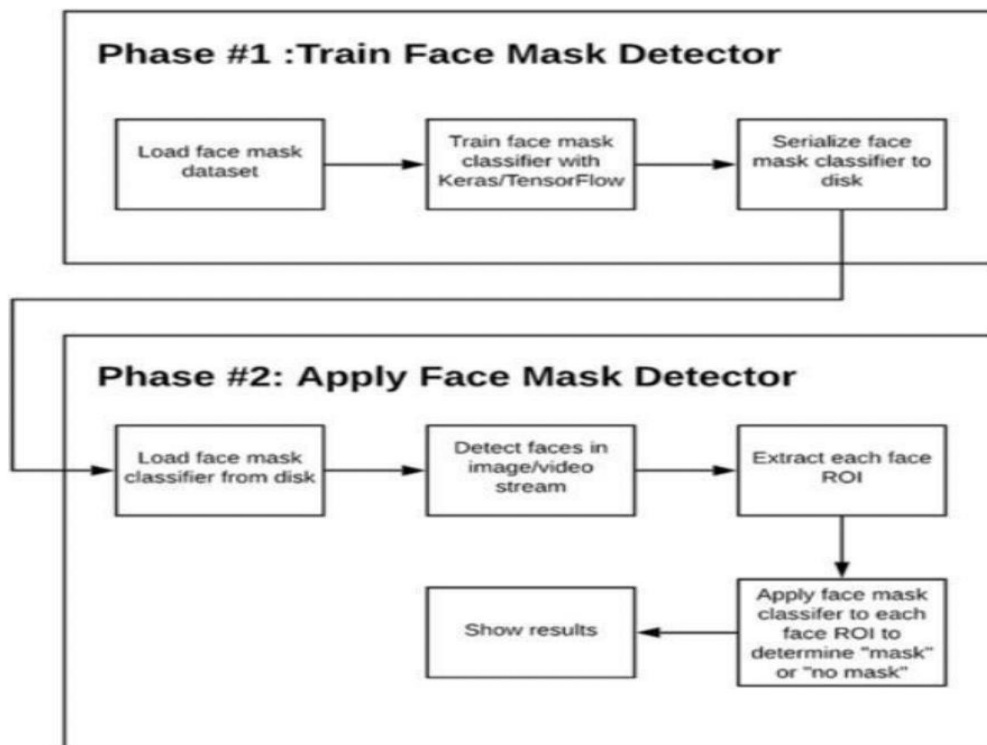


Fig.4.5.1 The training of data of face masks and non-masks

CHAPTER 5

RESULTS

It is tested and experimented with a model, where the working is based on the IR sensor in the following cases:

A. Case (i):

The room appliances are in off state, if there is nobody in the room which implies that energy is getting conserved instead of wasting the fig.5 gives a pictorial representation of the circuit connection and the external appearance with the LCD display.

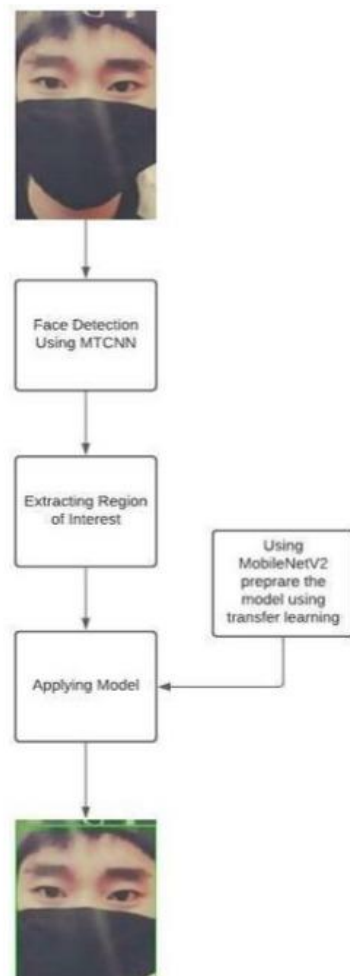


Fig. 5.1 Real Time Analysis

B. Case (ii):

The room appliances are switched on as the person enters the room.

The pictorial representation is given and the circuit connection is shown in the figure 5.1



Fig.5.2 Green Signal

C. Case (iii):

The room appliances remains switched on even when the person count increases in the room.

SCREENSHOT OF RESULT

```
def detect_and_predict_mask(frame, faceNet, maskNet):
    # 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))
```

```

13
14 # Hard code the values of your server and ip address here.
15 ip_address = "109.74.200.23"
16 port_number = "8080"
17 # Time interval in seconds for code to execute.
18 time_interval = 10

```

Fig.5.3 Fragment of code in python file

Name	Status	3% CPU	57% Memory	2% Disk	0% Network
Apps (6)					
Google Chrome (12)		0.1%	638.1 MB	0.1 MB/s	0 Mbps
keylogger (3)		0%	22.9 MB	0 MB/s	0 Mbps
Task Manager		0.2%	23.2 MB	0.1 MB/s	0 Mbps
Windows Explorer		0.1%	55.5 MB	0.1 MB/s	0 Mbps
Windows Security		0%	28.3 MB	0 MB/s	0 Mbps
WPS Office (32 bit) (9)		1.9%	222.8 MB	0 MB/s	0 Mbps
Background processes (64)					
Antimalware Service Executable		0%	205.5 MB	0 MB/s	0 Mbps
Application Frame Host		0%	6.8 MB	0 MB/s	0 Mbps
COM Surrogate		0%	1.0 MB	0 MB/s	0 Mbps
COM Surrogate		0%	2.7 MB	0 MB/s	0 Mbps
Component Package Support S...		0%	1.3 MB	0 MB/s	0 Mbps

Fig.5.4 final percent those who wore mask
(output)

CHAPTER – 6

SUMMARY AND CONCLUSION

From study of reference papers we came into a conclusion for making a very efficient and compact device that can detect people count and mask detection. The vision system is setup by individual subsystems face detection tracking subsystem based on adaptive skin detector, and a simple and fast motion predictor is composed for face tracking. The second is the face recognition system based on algorithms such as PCA and recognition procedures. It is robust and efficient to recognise many people on line in different views and unknown scene. This study proposes the MobileNetbased Depthwise Separable Convolution Neural Network (DS-CNN) for mask detection in facial images. We compare our findings to the original convolutional filters on specific datasets. The suggested system outperformed current classical convolutions in experiments, according to the results. The suggested technique is also contrasted with previous work on a motivated baseline method. Our findings (Acc. = 93.14, Pre. = 92, recall = 92, F-score = 92) show that the proposed method produces the highest overall performance across a variety of assessment metrics. The approach requires extra processing to generate visualizations and, owing to dataset constraints, cannot discriminate between right and erroneous mask usage. Our future aim is to create face mask recognition datasets with different mask wearing states, or employ zero shot learning to make the design identify erroneous mask wearing states.

REFERENCES

- [1] Dr. K.S. Tiwari, Supriya Bhagat, Nikita Patil, Priya Nagare, IOT Based Driver Drowsiness Detection & Health Monitoring System.
- [2] Rateb Jabbar, Mohammed Shinoy, Mohamed Kharbeche, Khalifa Al-Khalifa, MoezKrichen, Kamel Barkaoui, Driver Drowsiness Detection Model using CNN Techniques for Android App.
- [3] Md. Yousuf Hossain, Fabian Parsia George, IOT based Real-time Drowsy Driving Detection System for the Prevention of Road Accidents.
- [4] Mika Sunagawa, Shin-ichi-Shikii, WataruNakai, Makoto Mochizuki, Koichi Kusukame, and Hiroki Kitajima, Comprehensive Drowsiness Level Detection Model Combining Multimodal Information.
- [5] Monagi H. Alkinani, Wazir Zada Khan, and Quratulain Arshad, Detecting Human Driver Inattentive and Aggressive Driving Behaviour using Deep Learning: Recent Advances, Requirements, and Open Challenges.
- [6] Joao Ruivo Paulo, Gabriel Pires, and Urbano J. Nunes, Cross-Subject Zero Calibration Driver's Drowsiness Detection: Exploring Spatiotemporal Image Encoding of EEG Signals for CNN Classification.
- [7] Naveen Senniappan Karuppusamy, Bo-Yeong Kang, Multimodal System to Detect Driver Fatigue using EEG, Gyroscope, and Image Processing.
- [8] Yaocong Hu, Mingqi Lu, Chao Xie, and Xiabo Lu, Driver Drowsiness Recognition via 3D Conditional GAN and Two-level AttentionBiLSTM.
- [9] CeerthiBala U.K, Sarath TV, Internet of things based Intelligent Drowsiness

Alert System, Fifth International Conference on Communication and Electronics Systems (ICCES2020).

[10] S Sethi, M Kathuria and T Kaushik, "Face mask detection using deep learning: An approach to reduce risk of Coronavirus spread", *PubMed Central(PMC)* - U.S.

[11] B Varshini, Yogesh HR, Pash SD, S Maaz, V Madhumita and A Sasi, "IoT-Enabled smart doors for monitoring body temperature and face mask detection.

[12] S Teboulbi, S Messaoud, Hajjaji MA and A Mtibaa, "Real-Time Implementation of AI-Based Face Mask Detection and SocialDistancing Measuring System for COVID-19 Prevention

[13] K Kalla, VK Bokka, CS Devarapalli, Gaggala VV and Pitta RS, "Face Mask Detection Using Machine Learning.

[14] GKJ Hussain, R Priya, S Rajarajeswari, P Prasanth and N Niyazuddeen, "The Face Mask Detection Technology for Image Analysis in the Covid-19 Surveillance System.

[15] Vinod BG and A Teja's, "Implementation of Automatic Contactless Temperature Sensing and Door Access", *International Journal of Advanced Research in Computer and Communication Engineering*.

□

[16] A Das, MW Ansari and R Basak, "Covid-19 Face Mask Detection Using TensorFlow Keras and OpenCV", *2020 IEEE 17th India Council International Conference (INDICON)*, 2020.

[17] Militante SV and Dionisio NV, "Deep Learning Implementation of Facemask and Physical Distancing Detection with AlarmSystems", *Vocational Education and Electrical Engineering*.

[18] P Nagrath, R Jain, A Madan, A Arora, P Kataria and A Hemanth, "SSDMNV2: A real-time DNN-based face mask detection system using single shot multibox detector andMobileNetV2", 2020.

[19] V Vinitha and V Velantina, "COVID-19 facemask detection with deep learning and computer vision", *IRJET*, 2020.

[20] SA Sanjaya and SA Rakhmawan, "Face Mask DetectionUsing MobileNetV2 in The Era of COVID-19 Pandemic", *2020International Conference on Data Analytics for Business andIndustry:Way Towards a Sustainable Economy (ICDABI)*, pp. 1-5, 2020.

[21] Abhishek Madankar, Minal Patiil and Prabhakar Khandait, "Automated Waste Segregation System and its approach towards generation of Ethanol", *2019 5th International Conference on AdvancedComputing & Communication Systems (ICACCS)*, 2019.

[22] T Meenpal, A Balakrishnan and A Verma, "Facial Mask Detection using Semantic Segmentation", *2019 4th International Conference on Computing Communications and Security (ICCCS)*, 2019.

[23] G Deora, R Godhula and V Udpikar, "Study of Masked Face Detection Approach in Video Analytics", *IEEE Conference on Advances in Signal Processing*, 2016.

[24] M.L. Keote and P.T. Karule, "Design and implementation of energy efficient Adiabatic ECRL and basic gates.

APPENDIX –

Source Code

FACE MASK PROGRAM -

```
# import the necessary packages from
tensorflow.keras.applications.mobilenet_v2 import
preprocess_input from
tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model from
imutils.video import VideoStream import numpy as np import
argparse import imutils import time import cv2 import os
```

```
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, (300,
300),
    (104.0, 177.0, 123.0))

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

    # 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 >
args["confidence"]:
            # 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 predictions if at least one face
was detected        if len(faces) > 0:

        # for faster inference we'll 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)

# construct the argument parser and parse the
arguments ap = argparse.ArgumentParser()
ap.add_argument("-f", "--face", type=str,
default="Datasets",      help="path to face
detector model directory") ap.add_argument("-
m", "--model", type=str,
default="Datasets/mask_detector.model",

```

```

help="path to trained face mask detector model")
ap.add_argument("-c", "--confidence", type=float,
default=0.5,      help="minimum probability to
filter weak detections") args =
vars(ap.parse_args())

# load our serialized face detector model from disk
print("[INFO] loading face detector model...")
prototxtPath = os.path.sep.join([args["face"],
"deploy.prototxt"]) weightsPath =
os.path.sep.join([args["face"],
"res10_300x300_ssd_iter_140000.caffemodel"])
faceNet = cv2.dnn.readNet(prototxtPath,
weightsPath)

# load the face mask detector model from disk
print("[INFO] loading face mask detector
model...") maskNet =
load_model(args["model"])

# initialize the video stream and allow the camera sensor to
warm up print("[INFO] starting video stream...") vs =
VideoStream(src=0).start()

# loop over the frames from the video
stream 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)
cv2.imshow("Live", frame)

```

```

        # 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

            cv2.rectangle(frame, (startX, startY), (endX, endY),
            (0,255,0), 2)

        # show the output
        frame
        cv2.imshow("Live",
        frame)      key =
        cv2.waitKey(30) &
        0xFF

        # if the `q` key was pressed, break
        from the loop      if key == ord("q"):
            break

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

```

PEOPLE COUNT -

```
#include<LiquidCrystal.h>
```

```
LiquidCrystal lcd( 12, 11, 10, 9, 8,7);
```

```
#define splash splash1
```

```
#define ir1 6
```

```
#define ir2 5
```

```
#define buz 2
```

```
#define buz_ 3
```

```
int ir1_r, ir2_r, ir1_Stat, ir2_Stat;
```

```
int max_allowed = 5;
```

```
float rst,
```

```
setDisp =
```

```
3; int
```

```
per_tot_Sta
```

```
tus;
```

```
void setup()
```

```
{
```

```
    Serial.begin(115200);
```

```
    pinMode(ir1, INPUT);
```

```
    pinMode(ir2, INPUT);
```

```
    pinMode(buz,
```

```
    OUTPUT);
```

```
    pinMode(buz_,
```

```
    OUTPUT);
```

```
digitalWrite(buz, LOW);  
digitalWrite(buz_, LOW);  
LcDSet();
```

```
Serial.println("Ready");
```

```
}
```

```
void LcDSet() {
```

```
    lcd.begin(16, 2);
```

```
    splash(0, "Covid  
Safe"); splash(1,  
"RoOm");
```

```
    delay(2000);  
    lcd.clear();
```

```
}
```

```
void loop()
```

```
{
```

```
    getSensorOUT()
```

```
    ;
```

```
    getSensorIN();
```

```
    set_Disp(3);
```

```
}
```

```
void showDisplay() {
```

```

    lcd.setCursor(0, 0);
    lcd.print("Person No's:
"); lcd.setCursor(13, 0);
    lcd.print(per_tot_Status);

}

```

```

void getSensorIN() {

    ir1_r =
digitalRead(ir1);
    ir2_r =
digitalRead(ir2);

    if (ir1_r == 0 and ir1_Stat == 0) {

per_tot_Status+
+;    ir1_Stat =
1;

    }
    else if (ir1_r == 1 and ir1_Stat == 1) {

        ir1_Stat = 0;

        splash(1, "");
        Serial.println("RST");
    }
}

```

```

    if (per_tot_Status > max_allowed) {    per_tot_Status =
max_allowed;
//    splash(0,
"Not");
splash(1, "Not-
Allowed");

    Serial.println("Maximum Allowed No of Person's Already
Present in the Room \nVisit Later
\nThank You!!!");
digitalWrite(buz,
HIGH);
delay(50);
digitalWrite(buz,
LOW);
delay(70);
digitalWrite(buz,
HIGH);
delay(50);
digitalWrite(buz,
LOW);
delay(70);

}
if (per_tot_Status < 0) {

    per_tot_Status = 0;

}

}

void getSensorOUT() {

```

```

    ir1_r =
digitalRead(ir1);
    ir2_r =
digitalRead(ir2);

    if (ir2_r == 0 and ir2_Stat == 0) {

per_tot_Status-
-;    ir2_Stat =
1;

    }
    else if (ir2_r == 1 and ir2_Stat == 1) {

        ir2_Stat = 0;

    }

    if (per_tot_Status > max_allowed) {

        per_tot_Status = max_allowed;
    }
    if (per_tot_Status < 0) {

        per_tot_Status = 0;

    }

}

void set_Displacement(float val) {

```

```

setDisp += 0.1;

if (setDisp >= val) {

showDisplay(
);  setDisp
= 0;

}

}

```

PEOPLE COUNT – Screenshots

```

import the necessary packages
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from imutils.video import VideoStream
import numpy as np
import imutils
import time
import cv2
import os

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 predictions if at least one face was detected
if len(faces) > 0:
    # for faster inference we'll 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)

# load our serialized face detector model from disk

```

```

prototxtPath = r"face_detector\deploy.prototxt"
weightsPath = r"face_detector\res10_300x300_ssd_iter_140000.caffemodel"
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

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

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

# loop over the frames from the video stream
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
        label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)

        # display the label and bounding box rectangle on the output
        # frame
        cv2.putText(frame, label, (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)
    key = cv2.waitKey(1) & 0xFF

    # if the `q` key was pressed, break from the loop
    if key == ord("q"):

```

```
        break

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