Human Body Temperature Screening and Mask Detection- COVID 19 Monitoring Solutions using TensorFlow

A project report submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR, ANANTAPURAMU

in partial fulfillment of the requirements for the award of degree of

BACHELOR OF TECHNOLOGY
In
ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

Ekkaluri Rajeswari	**- 17121A0473**
G M Keerthana	**- 17121A0476**
Gundigi Poojitha	**- 17121A0495**
Junju Pranavi	**- 17121A04A2**

Under the Guidance of

DR N. PADMAJA, M.Tech, Ph.D
Professor,Dept. of ECE

SREE VIDYANIKETHAN Engineering College (Autonomous)

Department of Electronics and Communication Engineering

SREE VIDYANIKETHAN ENGINEERING COLLEGE

(AUTONOMOUS)

Sree Sainath Nagar, A. Rangampet, Tirupathi-517102

(2017-2021)

SREE VIDYANIKETHAN ENGINEERING COLLEGE

(AUTONOMOUS)

SreeSainath Nagar, A.Rangampet - 517 102

VISION

To be one of the Nation's premier Engineering Colleges by achieving the highest order of excellence in Teaching and Research.

MISSION

- To foster intellectual curiosity, pursuit and dissemination of knowledge.
- To explore students' potential through academic freedom and integrity.
- To promote technical mastery and nurture skilled professionals to face competition in ever increasing complex world.

QUALITY POLICY

Sree Vidyanikethan Engineering College strives to establish a system of Quality Assurance to continuously address, monitor and evaluate the quality of education offered to students, thus promoting effective teaching processes for the benefit of students and making the College a Centre of Excellence for Engineering and Technological studies.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Vision

To be a centre of excellence in Electronics and Communication Engineering through teaching and research producing high quality engineering professionals with values and ethics to meet local and global demands.

Mission

- The Department of Electronics and Communication Engineering is established with the cause of creating competent professionals to work in multicultural and multidisciplinary environments.
- Imparting knowledge through contemporary curriculum and striving for development of students with diverse background.
- Inspiring students and faculty members for innovative research through constant interaction with research organizations and industry to meet societal needs.
- Developing skills for enhancing employability of students through comprehensive training process.
- Imbibing ethics and values in students for effective engineering practice.

B. Tech. (Electronics and Communication Engineering)

Program Educational Objectives

After few years of graduation, the graduates of B.Tech (ECE) will be:

PEO1. Enrolled or completed higher education in the core or allied areas of electronics and communication engineering or management.

PEO2. Successful entrepreneurial or technical career in the core or allied areas of electronics and communication engineering.

PEO3. Continued to learn and to adapt to the world of constantly evolving technologies in the core or allied areas of electronics and communication engineering.

Program Outcomes

On successful completion of the Program, the graduates of B.Tech. (ECE) Program will be able to:

PO1 **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2 **Problem analysis**: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3 **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4 **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5 **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6 **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7 **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8 **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9 **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10 **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11 **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12 **Lifelong learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes

On successful completion of the Program, the graduates of B. Tech. (ECE) will be able to

PSO1. Apply the knowledge of Electronics, Signal Processing, Communications, and VLSI & Embedded Systems to the solutions of real-world problems.

PSO2. Analyse, Design and Develop solutions in real time in the domains of Electronics, Signal Processing, Communications, and VLSI & Embedded Systems.

PSO3. Conduct investigations and address complex engineering problems in the domains of Electronics, Signal Processing, Communications, and VLSI & Embedded Systems.

PSO4. Apply appropriate techniques, resources, and modern tools to complex engineering systems and processes in the domains of Electronics, Signal Processing, Communications, and VLSI & Embedded Systems.

Department of Electronics and Communication Engineering

SREE VIDYANIKETHAN ENGINEERING COLLEGE

(AUTONOMOUS)

Sree Sainath Nagar, A. Rangampet, Tirupathi - 517102.

Certificate

This is to certify that the project report entitled

"Human Body Temperature Screening and Mask Detection- Covid19 Monitoring solutions using TensorFlow"

is the bonafide work done and submitted by

Ekkaluri Rajeswari - 17121A0473
G M Keerthana - 17121A0476
Gundigi Poojitha - 17121A0495
Junju Pranavi - 17121A04A2

in the Department of Electronics and Communication Engineering, ***Sree Vidyanikethan Engineering College,*** *A. Rangampet, affiliated to* **Jawaharlal Nehru Technological University Anantapur, Anantapuramu** *in partial fulfillment of the requirements for the award of the degree of* **Bachelor of Technology** *in* **Electronics and Communication Engineering** *during 2017-2021.*

GUIDE

Dr.N. PADMAJA, M.Tech., Ph.D

Professor,Dept. of ECE

HOD

Dr. N. GIREESH, M.Tech., Ph.D.

Professor & Head, Dept. of ECE.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

It has been an honor and pleasure to have **Dr. N. Padmaja, M.Tech., Ph.D, Professor,** Department of Electronics and Communication Engineering, Sree Vidyanikethan Engineering College, Tirupati as supervisor of this project work. In addition to his knowledge and vast experience, we could also enjoy his very positive and cooperation during the hardest moments of this project work.

We are thankful to **Dr. N. Gireesh, M.Tech., Ph.D, Professor & Head,** Department of Electronics and Communication Engineering, Sree Vidyanikethan Engineering College, for his guidance, sconstant encouragement and support.

We express my gratitude to beloved Principal **Dr. B.M. Satish, Ph.D**, of SVEC for providing all facilities in completing Project course successfully. Our heartful thanks to all my teachers in the department of ECE Sree Vidyanikethan Engineering College for their moral support and good wishes.

Finally, we would like to express Our sincere thanks to our parents, friends, one and all, those who guided, inspired and helped us in completion of our project work.

PROJECT BATCH: 21B06

Ekkaluri Rajeswari - 17121A0473
G M Keerthana - 17121A0476
Gundigi Poojitha - 17121A0495
Junju Pranavi - 17121A04A2

SREE VIDYANIKETHAN ENGINEERING COLLEGE

(AUTONOMOUS)

SreeSainath Nagar, A.Rangampet - 517 102

Department of Electronics and Communication Engineering

PROJECT TITLE: Temperature Check Up and Mask Detection – COVID 19 Monitoring Solutions Using TensorFlow

ABSTRACT: Covid-19 has upended societies and dramatically altered everyday life across the globe. Many constraints have been set on the people on maintaining social distancing, wearing a mask along with periodical temperature check at the entrance of malls, offices, super markets. This project proposes a model which detects whether a person wears a mask or not by training a Covid-19 face mask detector with tensor flow, Deep learning. This model also performs thermal scanning of a person and compares his/her body temperature with normal body temperature. If the measured temperature is comparable with the pre-defined value and the mask is detected, the person is allowed to enter otherwise the entry is denied with an alarm. The proposed model also finds the count of the total number of people who entered, got rejected.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

According to the World Health Organization (WHO)'s official Situation Report – 205, coronavirus disease 2019 (COVID-19) has globally infected over 20 million people causing over 0.7million deaths. Individuals with COVID-19 have had a wide scope of symptoms reported – going from mellow manifestations to serious illness. Respiratory problems like shortness of breath or difficulty in breathing is one of them. Elder people having lung disease can possess serious complications from COVID-19 illness as they appear to be at higher risk. Some common human coronaviruses that infect public around the world are 229E, HKU1, OC43, and NL63. Before debilitating individuals, viruses like 2019-nCoV, SARS-CoV, and MERS-CoV infect animals and evolve to human coronaviruses. Persons having respiratory problems can expose anyone (who is in close contact with them) to infective beads. Surroundings of a tainted individual can cause contact transmission as droplets carrying virus may withal arrive on his adjacent surfaces.

To curb certain respiratory viral ailments, including COVID-19, wearing a clinical mask is very necessary. The public should be aware of whether to put on the mask for source control or aversion of COVID-19. Potential points of interest of the utilization of masks lie in reducing vulnerability of risk from a noxious individual during the "pre-symptomatic" period and stigmatization of discrete persons putting on masks to restraint the spread of virus. WHO stresses on prioritizing medical masks and respirators for health care assistants. Therefore, face mask detection has become a crucial task in present global society.

Face mask detection involves in detecting the location of the face and then determining whether it has a mask on it or not [1]. The issue is proximately cognate to general object detection to detect the classes of objects. Face identification categorically deals with distinguishing a specific group of entities that is Face. It has numerous applications, such as autonomous driving, education, surveillance, and so on. This paper presents a simplified approach to serve the above purpose using the basic Machine Learning (ML) packages such as TensorFlow, Keras, OpenCV and Scikit-Learn.

1.1 Introduction to Embedded Systems

An embedded system can be defined as a computing device that does a specific focused job. Appliances such as the air-conditioner, VCD player, DVD player, printer, fax machine, mobile phone etc. are examples of embedded systems. Each of these appliances will have a processor and special hardware to meet the specific requirement of the application along with the embedded software that is executed by the processor for meeting

that specific requirement. The embedded software is also called “firm ware”. The desktop/laptop computer is a general purpose computer. You can use it for a variety of applications such as playing games, *word* processing, accounting, software development and so on. In contrast, the software in the embedded systems is always fixed listed below:

Embedded systems do a very specific task, they cannot be programmed to do different things. Embedded systems have very limited resources, particularly the memory. Generally, they do not have secondary storage devices such as the CDROM or the floppy disk. Embedded systems have to work against some deadlines. A specific job has to be completed within a specific time. In some embedded systems, called real-time systems, the deadlines are stringent. Missing a deadline may cause a catastrophe-loss of life or damage to property. Embedded systems are constrained for power. As many embedded systems operate through a battery, the power consumption has to be very low. Some embedded systems have to operate in extreme environmental conditions such as very high temperatures and humidity.

1.2 Overview of Embedded System Architecture

Every embedded system consists of custom-built hardware built around a Central Processing Unit (CPU). This hardware also contains memory chips onto which the software is loaded.

The software residing on the memory chip is also called the ‘firmware’.

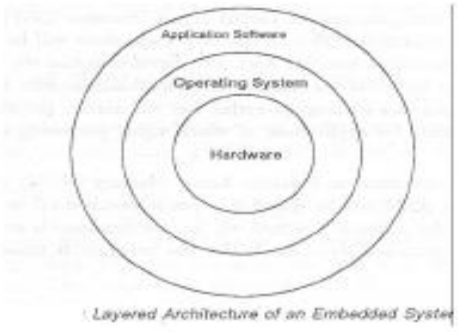

Fig 1.1 Layered Architecture of an Embedded System

The embedded system architecture can be represented as a layered architecture as shown in Fig.

The operating system runs above the hardware, and the application software runs above the operating system. The same architecture is applicable to any computer including a desktop computer. However, there are significant differences. It is not compulsory to have an operating system in every embedded system. For small appliances such as remote control units, air conditioners, toys etc., there is no need for an operating system and you can write only the software specific to that application.

For applications involving complex processing, it is advisable to have an operating system. In such a case, you need to integrate the application software with the operating system and then transfer the entire software on to

the memory chip. Once the software is transferred to the memory chip, the software will continue to run for a long time you don't need to reload new software.

Now, let us see the details of the various building blocks of the hardware of an embedded system. As shown in Fig. the building blocks are;

- Central Processing Unit (CPU)
- Memory (Read-only Memory and Random Access Memory)
- Input Devices
- Output devices
- Communication interfaces
- Application-specific circuitry

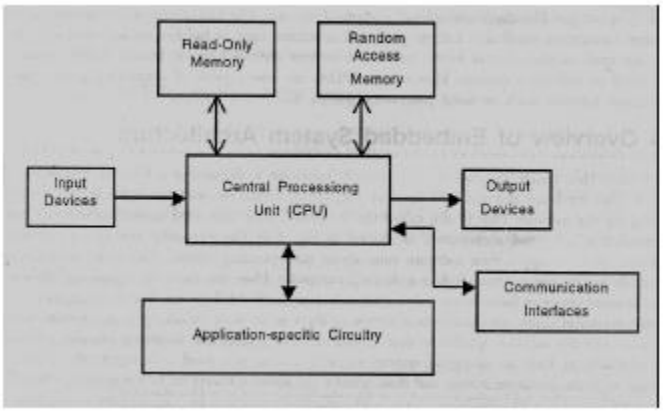

Fig 1.2 Building Blocks of Hardware of an Embedded system

1.2.1 Central Processing Unit (CPU):

The Central Processing Unit (processor, in short) can be any of the following: microcontroller, microprocessor or Digital Signal Processor (DSP). A micro-controller is a low-cost processor. Its main attraction is that on the chip itself, there will be many other components such as memory, serial communication interface, analog-to digital converter etc. So, for small applications, a micro-controller is the best choice as the number of external components required will be very less. On the other hand, microprocessors are more powerful, but you need to use many external components with them. D5P is used mainly for applications in which signal processing is involved such as audio and video processing.

1.2.2 Memory:

The memory is categorized as Random Access 11emory (RAM) and Read Only Memory (ROM). The contents of the RAM will be erased if power is switched off to the chip, whereas ROM retains the contents even if the power is switched off. So, the firmware is stored in the ROM. When power is switched on, the processor reads the ROM; the program is program is executed.

1.2.3 Input devices:

Unlike the desktops, the input devices to an embedded system have very limited capability. There will be no keyboard or a mouse, and hence interacting with the embedded system is no easy task. Many embedded systems will have a small keypad-you press one key to give a specific command. A keypad may be used to input only the digits. Many embedded systems used in process control do not have any input device for user interaction; they take inputs from sensors or transducers 1'fnd produce electrical signals that are in turn fed to other systems.

1.2.4 Output devices:

The output devices of the embedded systems also have very limited capability. Some embedded systems will have a few Light Emitting Diodes (LEDs) to indicate the health status of the system modules, or for visual indication of alarms. A small Liquid Crystal Display (LCD) may also be used to display some important parameters.

1.2.5 Communication interfaces:

The embedded systems may need to, interact with other embedded systems at they may have to transmit data to a desktop. To facilitate this, the embedded systems are provided with one or a few communication interfaces such as RS232, RS422, RS485, Universal Serial Bus (USB), IEEE 1394, Ethernet etc.

1.2.6 Application-specific circuitry:

Sensors, transducers, special processing and control circuitry may be required fat an embedded system, depending on its application. This circuitry interacts with the processor to carry out the necessary work. The entire hardware has to be given power supply either through the 230 volts main supply or through a battery. The hardware has to design in such a way that the power consumption is minimized.

1.3 Following are the advantages of Embedded Systems:

1. They are designed to do a specific task and have real time performance constraints which must be met.
2. They allow the system hardware to be simplified so costs are reduced.
3. They are usually in the form of small computerized parts in larger devices which serve a general purpose.

1.4 Application Areas

Nearly 99 per cent of the processors manufactured end up in embedded systems. The embedded system market is one of the highest growth areas as these systems are used in very market segment- consumer electronics, office automation, industrial automation, biomedical engineering, wireless communication, data communication, telecommunications, transportation, military and so on.

Consumer appliances

At home we use a number of embedded systems which include digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air-conditioner, VCO player, video game consoles, video recorders etc. Today's high-tech car has about 20 embedded systems for transmission control, engine spark control, air-conditioning, navigation etc. Even wristwatches are now becoming embedded systems. The palmtops are powerful embedded systems using which we can carry out many general-purpose tasks such as playing games and word processing.

Office automation: The office automation products using em embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

Industrial automation: Today a lot of industries use embedded systems for process control. These include pharmaceutical, cement, sugar, oil exploration, nuclear energy, electricity generation and transmission. The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc., and then take appropriate action based on the monitored levels to control other devices or to send information to a centralized monitoring station. In hazardous industrial environment, where human presence has to be avoided, robots are used, which are programmed to do specific jobs. The robots are now becoming very powerful and carry out many interesting and complicated tasks such as hardware assembly.

Medical electronics: Almost every medical equipment in the hospital is an embedded system. These equipments include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, colonscopy, endoscopy etc. Developments in medical electronics have paved way for more accurate diagnosis of diseases.

Computer networking: Computer networking products such as bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded systems which implement the necessary data communication protocols. For example, a router interconnects two networks. The two networks may be running different protocol stacks. The router's function is to obtain the data packets from incoming pores, analyze the packets and send them towards the destination after doing

necessary protocol conversion. Most networking equipments, other than the end systems (desktop computers) we use to access the networks, are embedded systems

Telecommunications: In the field of telecommunications, the embedded systems can be categorized as subscriber terminals and network equipment. The subscriber terminals such as key telephones, ISDN phones, terminal adapters, web cameras are embedded systems. The network equipment includes multiplexers, multiple access systems, Packet Assemblers Dissemblers (PADs), satel1ite modems etc. IP phone, IP gateway, IP gatekeeper etc. are the latest embedded systems that provide very low-cost voice communication over the Internet.

Wireless technologies: Advances in mobile communications are paving way for many interesting applications using embedded systems. The mobile phone is one of the marvels of the last decade of the 20th century. It is a very powerful embedded system that provides voice communication while we are on the move. The Personal Digital Assistants and the palmtops can now be used to access multimedia services over the Internet. Mobile communication infrastructure such as base station controllers, mobile switching centers are also powerful embedded systems.

Insemination: Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc. are all embedded systems. Test equipment such as oscilloscope, spectrum analyzer, logic analyser, protocol analyser, radio communication test set etc. are embedded systems built around powerful processors. Thank to miniaturization, the test and measuring equipment are now becoming portable facilitating easy testing and measurement in the field by field-personnel.

Security: Security of persons and information has always been a major issue. We need to protect our homes and offices; and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative businesses nowadays. Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Encryption devices are nearly 99 per cent of the processors that are manufactured end up in~ embedded systems. Embedded systems find applications in.every industrial segment-consumer electronics, transportation, avionics, biomedical engineering, manufacturing, process control and industrial automation, data communication, telecommunication, defence, security etc. Used to encrypt the data/voice being transmitted on communication links such as telephone lines. Biometric systems using fingerprint and face recognition are now being extensively used for user authentication in banking applications as well as for access control in high security buildings.

Finance: Financial dealing through cash and cheques are now slowly paving way for transactions using smart cards and ATM (Automatic Teller Machine, also expanded as Any Time Money) machines. Smart card, of the size of a credit card, has a small micro-controller and memory; and it interacts with the smart card reader! ATM machine and acts as an electronic wallet. Smart card technology has the capability of ushering in a cashless society.

CHAPTER-2

DIGITAL IMAGE PROCESSING

Digital Image Processing Tutorial provides basic and advanced concepts of Image Processing. Our Digital Image Processing Tutorial is designed for beginners and professionals both.

Digital Image Processing is used to manipulate the images by the use of algorithms. For processing digital images the most common software that used widely is Adobe Photoshop.

Our Digital Image Processing Tutorial includes all topics of Digital Image Processing such as introduction, computer graphics, signals, photography, camera mechanism, pixel, transaction, types of Images, etc.

2.1 Digital Image Processing (DIP)

Digital Image Processing (DIP) is a software which is used to manipulate the digital images by the use of computer system [2]. It is also used to enhance the images, to get some important information from it.

Fig 2.1 Image of DIP

For example: Adobe Photoshop, MATLAB, etc.

It is also used in the conversion of signals from an image sensor into the digital images.

A certain number of algorithms are used in image processing.

Digital Image Processing

Digital Image Processing is a software which is used in image processing. For example: computer graphics, signals, photography, camera mechanism, pixels, etc.

Digital Image Processing provides a platform to perform various operations like image enhancing, processing of analog and digital signals, image signals, voice signals etc.

It provides images in different formats.

2.2 Digital Image Processing allows users the following tasks

- **Image sharpening and restoration:** The common applications of Image sharpening and restoration are zooming, blurring, sharpening, grayscale conversion, edges detecting, Image recognition, and Image retrieval, etc.
- **Medical field:** The common applications of medical field are Gamma-ray imaging, PET scan, X-Ray Imaging, Medical CT, UV imaging, etc.
- **Remote sensing:** It is the process of scanning the earth by the use of satellite and acknowledges all activities of space.
- **Machine/Robot vision:** It works on the vision of robots so that they can see things, identify them, etc.

2.3 Characteristics of Digital Image Processing

- It uses software, and some are free of cost.
- It provides clear images.
- Digital Image Processing do image enhancement to recollect the data through images.
- It is used widely everywhere in many fields.
- It reduces the complexity of digital image processing.
- It is used to support a better experience of life.

2.4 Advantages of Digital Image Processing

- Image reconstruction (CT, MRI, SPECT, PET)
- Image reformatting (Multi-plane, multi-view reconstructions)
- Fast image storage and retrieval
- Fast and high-quality image distribution.
- Controlled viewing (windowing, zooming)

2.5 Disadvantages of Digital Image Processing

- It is very much time-consuming.
- It is very much costly depending on the particular system.
- Qualified persons can be used.
- Proposed methodology

2.6 Step 1: Data Visualization

In the first step, let us visualize the total number of images in our dataset in both categories.

We can see that there are *690* images in the '*yes*' class and *686* images in the '*no*' class.

```
The number of images with facemask labelled 'yes': 690
The number of images with facemask labelled 'no': 686
```

Step 2: Data Augmentation

In the next step, we augment our dataset to include more number of images for our training. In this step of *data augmentation*, we rotate and flip each of the images in our dataset. We see that, after data augmentation, we have a total of *2751* images with *1380* images in the '*yes*' class and '*1371*' images in the '*no*' class.

```
Number of examples: 2751
Percentage of positive examples: 50.163576881134134%, number of pos examples: 1380
Percentage of negative examples: 49.836423118865866%, number of neg examples: 1371
```

Step 3: Splitting the data

In this step, we split Our data into the training set which will contain the images on which the CNN model will be trained and the test set with the images on which our model will be tested [3].

In this, we take split_size *=0.8*, which means that *80%* of the total images will go to the *training set* and the remaining *20%* of the images will go to the *test set.*

```
The number of images with facemask in the training set labelled 'yes': 1104
The number of images with facemask in the test set labelled 'yes': 276
The number of images without facemask in the training set labelled 'no': 1096
The number of images without facemask in the test set labelled 'no': 275
```

After splitting, we see that the desired percentage of images have been distributed to both the training set and the test set as mentioned above.

Step 4: Building the Model

In the next step, we build our *Sequential CNN model* with various layers such as *Conv2D, MaxPooling2D, Flatten, Dropout* and *Dense*. In the last Dense layer, we use the '*softmax*' function to output a vector that gives the *probability* of each of the two classes.

Sequential Model for Face Mask detection

Here, we use the 'adam' optimizer and 'binary_crossentropy*'* as our loss function as there are only two classes. Additionally, you can even use the *MobileNetV2* for better accuracy.

CHAPTER 3

LITERATURE SURVEY

OS by using NVIDIA JETSON TX1 controller board. The experimental results show that the proposed method capable to detect and distinguish objects in the different lighting condition, E. Susanto, R. Rudiawan, D. Analia, S. Pamungkas and H. Soebakti, " The deep learning development for real-time ball and goal detection of barelang-FC&quot, 2017 International Electronics Symposium on Engineering Technology and Applications (IES-ETA), pp. 146-151, 2017.

Studies of object detection have recently attracted increased interest. One of the applications of object detection is robotics. This paper present the real-time object detection integrated to humanoid robot soccer. In order to enhance the vision to detect ball and goal, the You Only Look Once (YOLO) methods is used as deep-learning object detection method. The real-time experiments have been carried out in LINUX with interference from other objects, also from the different angle of capturing an image.

C. Liu, Y. Tao, J. Liang, K. Li and Y. Chen, " Object detection based on YOLO network & quot;, 2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC), pp. 799-803, 2018.

Object detection based on the deep learning has achieved very good performances. However, there are many problems with images in real-world shooting such as noise, blurring and rotating jitter, etc. These problems have an important impact on object detection. Using traffic signs as an example, we established image degradation models which are based on YOLO network and combined traditional image processing methods to simulate the problems existing in real-world shooting. After establishing the different degradation models, we compared the effects of different degradation models on object detection. We used the YOLO network to train a robust model to improve the average precision (AP) of traffic signs detection in real scenes.

W. Yang and Z. Jiachun, "Real-time face detection based on YOLO", 2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII), pp. 221-224, 2018.

As a target detection system, YOLO has a fast detection speed and is suitable for target detection in real-time environment. Compared with other similar target detection systems, it has better detection accuracy and faster detection time. This paper is based on YOLO network and applied to face detection. In this paper, YOLO target detection system is applied to face detection. Experimental results show that the face detection method based on YOLO has stronger robustness and faster detection speed. Still in a complex environment can guarantee the high detection accuracy. At the same time, the detection speed can meet real-time detection requirements

D. Garg, P. Goel, S. Pandya, A. Ganatra and K. Kotecha, "A deep learning approach for face detection using YOLO", *2018 IEEE Punecon*, pp. 1-4, 2018.

Deep learning is nowadays a buzzword and is considered a new era of machine learning which trains the computers in finding the pattern from a massive amount of data. It mainly describes the learning at multiple levels of representation which helps to make sense on the data consisting of text, sound and images. Many organizations are using a type of deep learning known as a convolutional neural network to deal with the objects in a video sequence. Deep Convolution Neural Networks (CNNs) have proved to be impressive in terms of performance for detecting the objects, classification of images and semantic segmentation. Object detection is defined as a combination of classification and localization. Face detection is one of the most challenging problems of pattern recognition. Various face related applications like face verification, facial recognition, clustering of face etc. are a part of face detection. Effective training needs to be carried out for detection and recognition. The accuracy in face detection using the traditional approach did not yield a good result. This paper focuses on improving the accuracy of detecting the face using the model of deep learning. YOLO (You only look once), a popular deep learning library is used to implement the proposed work. The paper compares the accuracy of detecting the face in an efficient manner with respect to the traditional approach. The proposed model uses the convolutional neural network as an approach of deep learning for detecting faces from videos. The FDDB dataset is used for training and testing of our model. A model is finetuned on various performance parameters and the best suitable values are taken into consideration. It is also compared the execution of training time and the performance of the model on two different GPUs.

Table 3.1 Literature Survey

Year	Author	Description	Publication
2010	S. S. Mohamed, N. M. Tahir, R. Adnan	• Background subtraction is a widely used approach for detecting moving objects in videos from static cameras • Describes an efficient background subtraction technique for extracting the moving objects from a scene. Gaussian mixture models (GMM) gives best results than other segmentation methods.	Background modelling and background subtraction performance for object detection
2015	R Girshick	• State of the art object detection networks depend on region proposal algorithms to hypothesize object locations • Fast R-CNN have reduced the running time of these detection networks, exposing region proposal computation as a bottleneck	Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
2017	Shiming Ge, Jia Li, Qiting Ye, ZhaoLuo	• Introduces a dataset, denoted as MAFA and LLE-CNNs for masked face detection • This model contains three modules • two pre-trained CNNs, Incorporation of Embedding module (using locally linear embedding (LLE) algorithm) and Verification module	Detecting Masked Faces in the Wild with LLE-CNNs
2019	Selahattin, Sahaj Singh, Judy Qiu	• Cars are equipped with multiple cameras which captures video	A Fast Video Image Detection using

		streams that can be used for detection and predictive tasks to increase race safety • New dataset is created and compared it with three different Single Shot Multibox Detector models from TensorFlow Detection Model Zoo.	TensorFlow Mobile Networks for Racing Cars
2020	Adrian Rosebrock	• Includes two phase mask detector • Training the model using Keras/TensorFlow on datasets • Implemeting face mask detector in real-time video streams with OpenCV	COVID-19: Face Mask Detector with OpenCV, Keras/TensorFlow, and Deep Learning
2020	Isack Farady, Chih-Yang-Lin, Amornthep Rojanasarit, Kanatip, Fityanul	• Deep learning object detection network to detect and capture the temperature of a specific point inside a predicted bounding box • Includes two modules for the RetinaNet model to detect three categories of mask-wearing positions and the temperature of the head	Mask Classification and Head Temperature Detection Combined with Deep Learning Networks

CHAPTER 4

SOFTWARE REQUIREMENTS

Python is a simple, general purpose, high level, and object-oriented programming language.

Python is an interpreted scripting language also. *Guido Van Rossum* is known as the founder of Python programming.

Our Python tutorial includes all topics of Python Programming such as installation, control statements, Strings, Lists, Tuples, Dictionary, Modules, Exceptions, Date and Time, File I/O, Programs, etc. There are also given Python interview questions to help you better understand Python Programming.

4.1 Python

Python is a general purpose, dynamic, high-Level and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

Fig 4.1 Picture of Python Programming Language

Python is *easy to learn* yet powerful and versatile scripting language, which makes it attractive for Application Development.

Python's syntax and *dynamic typing* with its interpreted nature make it an ideal language for scripting and rapid application development.

Python supports *multiple programming pattern*, including object-oriented, imperative, and functional or procedural programming styles.

Python is not intended to work in a particular area, such as web programming. That is why it is known as *multipurpose* programming language because it can be used with web, enterprise, 3D CAD, etc.

We don't need to use data types to declare variable because it is *dynamically typed* so we can write a=10 to assign an integer value in an integer variable.

Python makes the development and debugging *fast* because there is no compilation step included in Python development, and edit-test-debug cycle is very fast.

4.2 Python 2 vs. Python 3

In most of the programming languages, whenever a new version releases, it supports the features and syntax of the existing version of the language, therefore, it is easier for the projects to switch in the newer version. However, in the case of Python, the two versions Python 2 and Python 3 are very much different from each other.

A list of differences between Python 2 and Python 3 are given below:

1. Python 2 uses **print** as a statement and used as print "something" to print some string on the console. On the other hand, Python 3 uses **print** as a function and used as print ("something") to print something on the console.
2. Python 2 uses the function raw-input() to accept the user's input. It returns the string representing the value, which is typed by the user. To convert it into the integer, we need to use the int() function in Python. On the other hand, Python 3 uses input() function which automatically interpreted the type of input entered by the user. However, we can cast this value to any type by using primitive functions (int(), str(), etc.).
3. In Python 2, the implicit string type is ASCII, whereas, in Python 3, the implicit string type is Unicode.

4. Python 3 doesn't contain the xrange() function of Python 2. The xrange() is the variant of range() function which returns a x range object that works similar to Java iterator. The range() returns a list for example the function range(0,3) contains 0, 1, 2.
5. There is also a small change made in Exception handling in Python 3. It defines a keyword **as** which is necessary to be used. We will discuss it in Exception handling section of Python programming tutorial.

4.3 Python History

Python was invented by **Guido van Rossum** in 1991 at CWI in Netherland. The idea of Python programming language has taken from the ABC programming language or we can say that ABC is a predecessor of Python language.

There is also a fact behind the choosing name Python. Guido van Rossum was a fan of the popular BBC comedy show of that time, **"Monty Python's Flying Circus"**. So, he decided to pick the name **Python** for his newly created programming language.

Python has the vast community across the world and releases its version within the short period.

4.4 Reasons for learning python

Python provides many useful features to the programmer. These features make it most popular and widely used language. We have listed below few-essential feature of Python.

- Easy to use and Learn
- Expressive Language
- Interpreted Language
- Object-Oriented Language
- Open Source Language
- Extensible
- Learn Standard Library
- GUI Programming Support
- Integrated
- Embeddable
- Dynamic Memory Allocation
- Wide Range of Libraries and Frameworks

4.5 Usage of python in various areas

Python is a general-purpose, popular programming language and it is used in almost every technical field. The various areas of Python use are given below.

- Data Science
- Date Mining
- Desktop Applications
- Console-based Applications
- Mobile Applications
- Software Development
- Artificial Intelligence
- Web Applications
- Enterprise Applications
- 3D CAD Applications
- Machine Learning
- Computer Vision or Image Processing Applications.
- Speech Recognitions

4.6 OPENCV USING PYTHON

OpenCV provides basic and advanced concepts of OpenCV. It is an open-source library for the computer vision. It provides the facility to the machine to recognize the faces or objects. In this tutorial we will learn the concept of OpenCV using the Python programming language.

OpenCV includes all topics of Read and Save Image, Canny Edge Detection, Template matching, Blob Detection, Contour, Mouse Event, Gaussian blur and so on.

4.6.1 OpenCV

Fig 4.2 Picture of OpenCV

OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc.

In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos [4].

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, and three-dimension model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify and different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and acts accordingly.

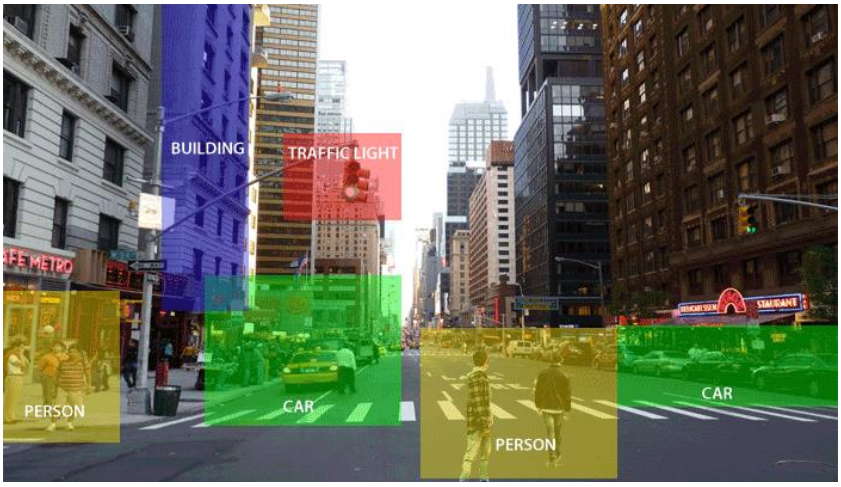

Fig 4.3 OpenCV for Computer Vision

Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency. There are a two main task which are defined below:

- **Object Classification** - In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories.
- **Object Identification** - In the object identification[5], our model will identify a particular instance of an object - for example, parsing two faces in an image and tagging one as Virat Kohli and other one as Rohit Sharma.

Fig 4.4 Picture of Object Identification

4.6.2 History

OpenCV stands for Open Source Computer Vision Library, which is widely used for image recognition or identification. It was officially launched in 1999 by Intel. It was written in C/C++ in the early stage, but now it is commonly used in Python for the computer vision as well.

The first alpha version of OpenCV was released for the common use at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and between 2001 and 2005, five betas were released. The first 1.0 version was released in 2006.

The second version of the OpenCV was released in October 2009 with the significant changes. The second version contains a major change to the C++ interface, aiming at easier, more type-safe, pattern, and better implementations. Currently, the development is done by an independent Russian team and releases its newer version in every six months.

4.6.3 Working of OpenCV

Here is, how computers perform image recognition.

Human eyes provide lots of information based on what they see. Machines are facilitated with seeing everything, convert the vision into numbers and store in the memory. Here the question arises how computer convert images into numbers. So the answer is that the pixel value is used to convert images into numbers. A pixel is the smallest unit of a digital image or graphics that can be displayed and represented on a digital display device.

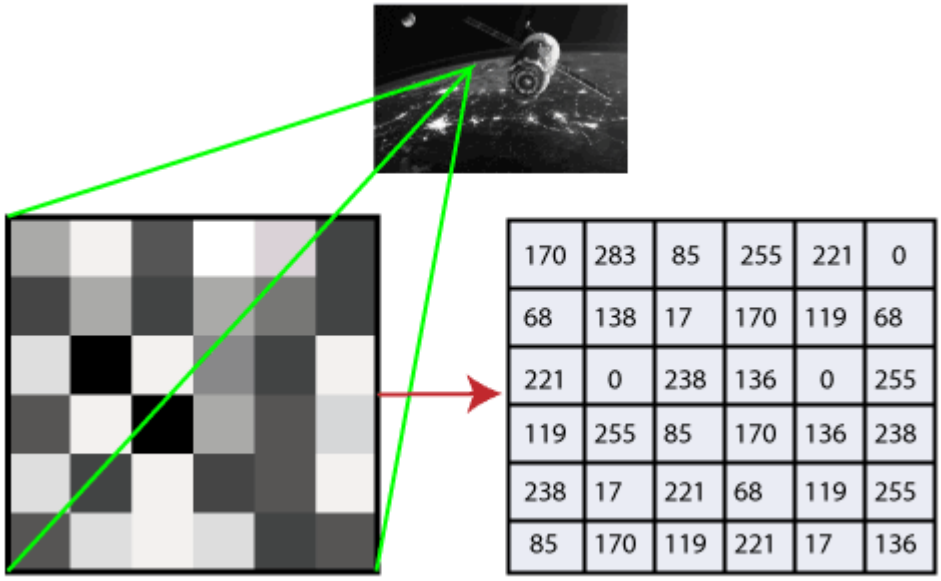

Fig 4.5 Conversion of Images to numbers

The picture intensity at the particular location is represented by the numbers. In the above image, we have shown the pixel values for a grayscale image consist of only one value, the intensity of the black colour at that location.

There are two common ways to identify the images. They are

- Grayscale
- RGB

Grayscale

Grayscale images are those images which contain only two colors black and white. The contrast measurement of intensity is black treated as the weakest intensity, and white as the strongest intensity. When we use the grayscale image, the computer assigns each pixel value based on its level of darkness.

RGB

An RGB is a combination of the red, green, blue color which together makes a new color. The computer retrieves that value from each pixel and puts the results in an array to be interpreted.

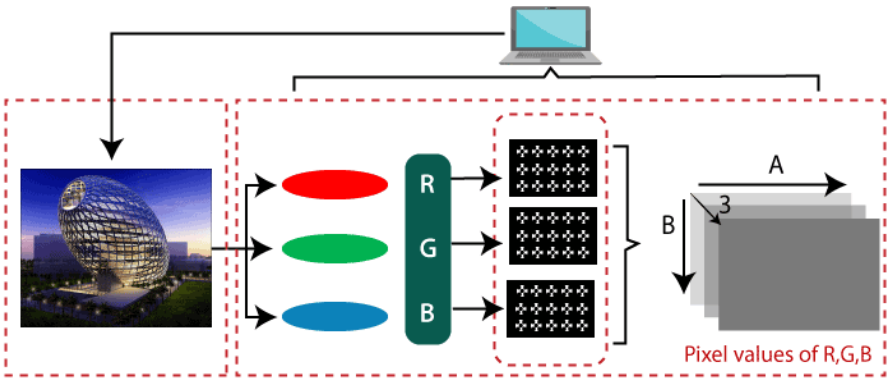

Fig 4.6 Picture explaining RGB

CHAPTER 5

COMPUTER VISION AND COMPUTER GRAPHICS

5.1 Computer Vision

Computer vision is a scientific field which deals with how computers can be made as high level devices which understand digital images and videos. In terms of engineering, it is an automate task that the human visual system can do. Computer vision has methods for acquiring, processing, analysing and understanding the digital image. The most important task is to extract high dimensional data from the real world which can produce numerical or symbolic information.

As a scientific discipline, computer vision is related to the theory of artificial system which can extract information from images. The image data is used in the form of video sequences which can be seen by a human.

Applications of Computer vision is as follows:

1. Robotics
2. Medicine
3. Security
4. Transportation
5. Industrial Automation

5.2 Computer Graphics

In computer graphics, pictures and films are created using computers. Usually, it refers to computer generated image data which is created to help specialized graphical hardware and software. It is a very vast developed area of computer science. In 1960, *Verne Hudson* and *William Fetter* of Boeing invented computer graphics. Computer graphics also includes user interface design, sprite graphics, vector graphics, 3D modelling, shaders, and computer vision. Computer graphics is also responsible to display image data in a meaningful manner. Computer graphics is also used for processing data which is received from the physical world.

Applications of Computer graphics is as follows:

- Computer Aided Design (CAD)
- Presentation Graphics
- 3d Animation
- Education and training
- Graphical User Interfaces

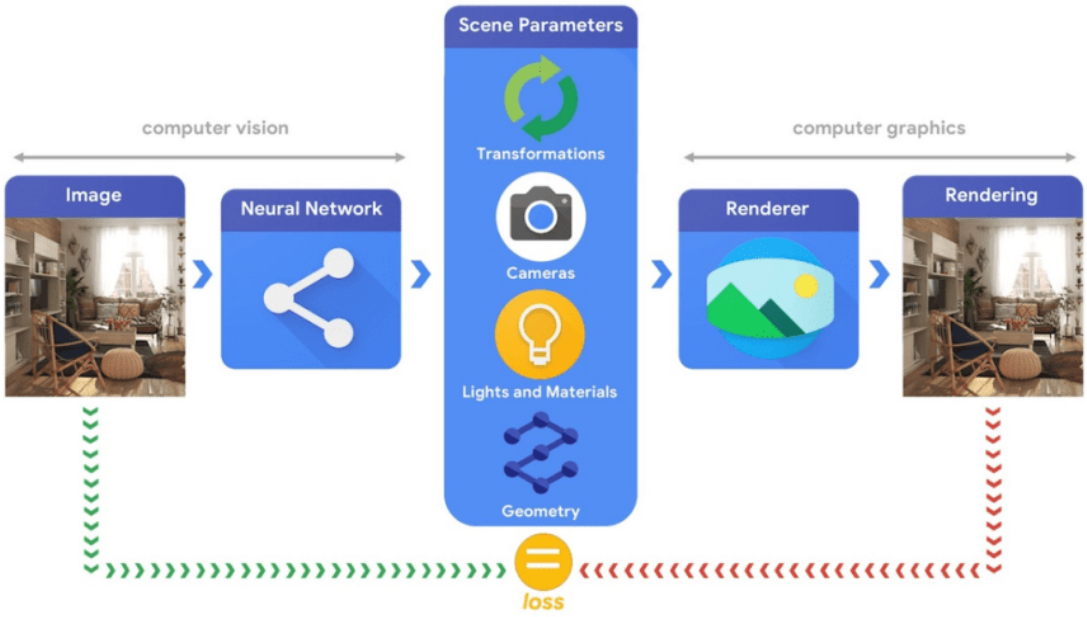

Fig 5.1 Computer Graphics and Computer Vision is equivalent to TensorFlow Graphics

5.3 Computer Vision using OpenCV

- OpenCV is available for free of cost.
- Since the OpenCV library is written in C/C++, so it is quit fast. Now it can be used with Python.
- It requires less RAM to usage, it maybe of 60-70 MB.
- Computer Vision is portable as OpenCV and can run on any device that can run on C.

5.4 Computer Vision

Computer vision is concerned with modelling and replicating human vision using computer software and hardware. Formally if we define computer vision then its definition would be that computer vision is a discipline that studies how to reconstruct, interrupt and understand a 3d scene from its 2d images in terms of the properties of the structure present in scene.

It needs knowledge from the following fields in order to understand and stimulate the operation of human vision system.

- Computer Science
- Electrical Engineering
- Mathematics
- Physiology
- Biology
- Cognitive Science

5.5 Computer Vision Hierarchy

Computer vision is divided into three basic categories that are as following:

Low-level vision: includes process image for feature extraction.

Intermediate-level vision: includes object recognition and 3D scene Interpretation

High-level vision: includes conceptual description of a scene like activity, intention and behaviour.

5.6 Related Fields of Computer Vision

Computer Vision overlaps significantly with the following fields:

Image Processing: it focuses on image manipulation.

Pattern Recognition: it studies various techniques to classify patterns.

Photogrammetry: it is concerned with obtaining accurate measurements from images.

5.7 Computer Vision Vs Image Processing

Image processing studies image to image transformation. The input and output of image processing are both images.

Computer vision is the construction of explicit, meaningful descriptions of physical objects from their image. The output of computer vision is a description or an interpretation of structures in 3D scene.

5.8 Applications

- Robotics
- Medicine
- Security
- Transportation
- Industrial Automation

Robotics Application

- Localization-determine robot location automatically
- Navigation
- Obstacles avoidance
- Assembly (peg-in-hole, welding, painting)
- Manipulation (e.g. PUMA robot manipulator)
- Human Robot Interaction (HRI): Intelligent robotics to interact with and serve people

Medicine Application

- Classification and detection (e.g. lesion or cells classification and tumor detection)
- 2D/3D segmentation
- 3D human organ reconstruction (MRI or ultrasound)
- Vision-guided robotics surgery

Industrial Automation Application

- Industrial inspection (defect detection)
- Assembly
- Barcode and package label reading
- Object sorting
- Document understanding (e.g. OCR)

Security Application

- Biometrics (iris, finger print, face recognition)
- Surveillance-detecting certain suspicious activities or behaviours

Transportation Application

- Autonomous vehicle
- Safety, e.g., driver vigilance monitoring

5.9 Computer Graphics and its Applications

Computer graphics are graphics created using computers and the representation of image data by a computer specifically with help from specialized graphic hardware and software. Formally we can say that Computer graphics is creation, manipulation and storage of geometric objects (modelling) and their images (Rendering).

The field of computer graphics developed with the emergence of computer graphics hardware. Today computer graphics is use in almost every field. Many powerful tools have been developed to visualize data. Computer graphics field become more popular when companies started using it in video games. Today it is a multibillion dollar industry and main driving force behind the computer graphics development. Some common applications areas are as following:

Fig 5.2 Applications of Computer Graphics

- Computer Aided Design (CAD)
- Presentation Graphics
- 3d Animation
- Education and training
- Graphical User Interfaces

Computer Aided Design

- Used in design of buildings, automobiles, aircraft and many other product
- Use to make virtual reality system.

Presentation Graphics

- Commonly used to summarize financial, statistical data
- Use to generate slides

3d Animation

- Used heavily in the movie industry by companies such as Pixar, DresmsWorks

- To add special effects in games and movies.

Education and training

- Computer generated models of physical systems
- Medical Visualization
- 3D MRI
- Dental and bone scans
- Stimulators for training of pilots etc.

Graphical User Interfaces

- It is used to make graphical user interfaces objects like buttons, icons and other components.

CHAPTER 6

ALGORITHMS FOR OBJECT DETECTION

This part of the topic highlights the algorithm used for object detection as well as object tracking.

6.1 YOLO Architecture

The Yolo algorithm stands for You Only Look Once, this algorithm is a state of art, which works on a real-time system, build on deep learning for solving various Object Detection as well as Object Tracking problems. The architecture of Yolo can be observed from the below Fig 8.1.

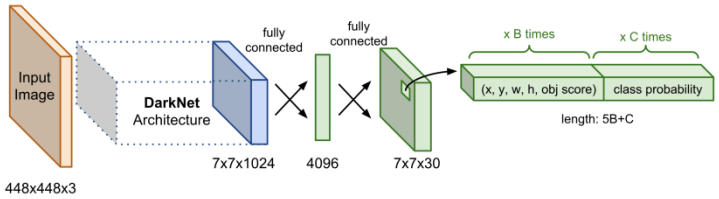

Fig 6.1 YOLO Architecture

It can be observed from the above figure that the architecture contains the Input image layers which are responsible for taking the inputs that would be passed to further layers, input can be any image depending upon the use cases. Along the input layer comes the DarkNet Architecture, this is an open-source neural network for which framework is created with the help of C & CUDA, this framework features YOLO for object detection & object tracking.

Further, the architecture consists of the flattened layer which is densely connected with the convolutional layer which is also densely connected to pass the data from each node to other nodes in the architecture, similarly, this is passed to the output layer which gives 4-part values, those 4 parts describe the predicted value for the bounding box, denoted by x, y, w, h, along with the object detection score plus the probability of the predicted class. This YOLO is part of the One-Shot object detector family which is accurate & fast, there is also a Two-Shot object detector.

Two-Shot object detectors which are popular are R-CNN, Fast R-CNN, and Faster R-CNN, these algorithms are accurate in obtaining the results based on certain use cases but are slow as compared to that of Yolo, You Only Look Once is an algorithm that looks at the image at a single glance and based on that look predicts the bounding boxes related to certain classes, classes can be anything ranging from Dog to Car, or Gun to Tanks,

this special feature makes Yolo stand out from others. Different types of object detectors based on a shot can be observed in Fig 8.2 below.

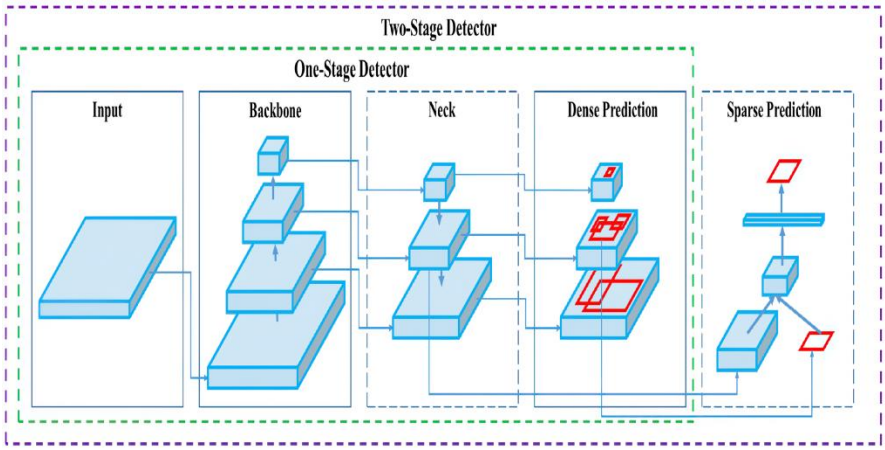

Fig 6.2 Different Types of Detector

From the above figure, we can find out different components, there are 4 different types of components

Input The input to the detector can be an image or video based on the use cases specified in the research.

Backbone The backbone of the object detector contains models, these models can be ResNet, DenseNet, VGG.

Neck The neck in the detector acts as an extra layer, which goes in parallel to the backbone & the head.

Head The head is the network that is in charge of the detection of objects based on bounding boxes.

Experimental Results

The experimental results section for this project details the results obtained after doing various observations and forming final outputs. This project focuses on social distancing detection & face mask detection for the events of Covid-19, Fig 8.3 explains the architecture for calculating the distance between objects and shows the flow of how the output is getting generated with the use of Yolo Version 4.

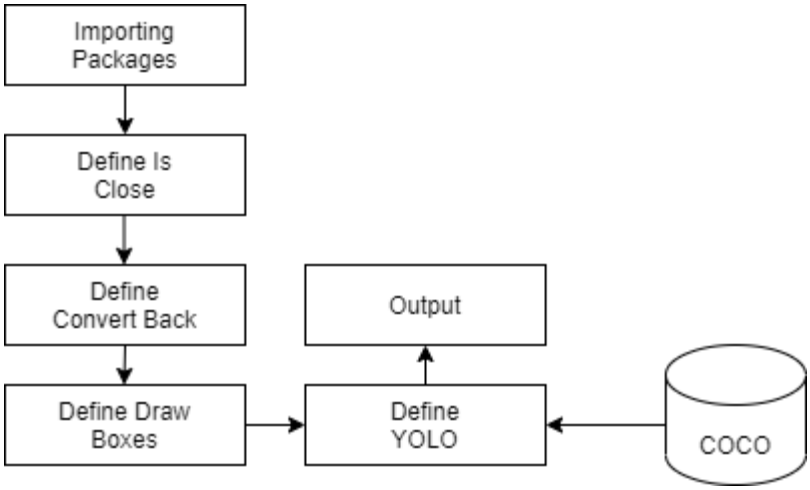

Fig 6.3 YOLO Darknet Architecture

The below Fig 8.4 is the architecture for the analysis of face masks on objects, the objects over here is the person on which the detection is performed with the help of custom datasets. The custom dataset is trained for 3 different categories (Good, None & Bad) depending upon the annotations provided, it bounds the boxes with respective classes. The difference between object detection and object tracking is the use of a tracker (in Yolo DeepSort) which helps in keeping a track of an object by assigning an Id.

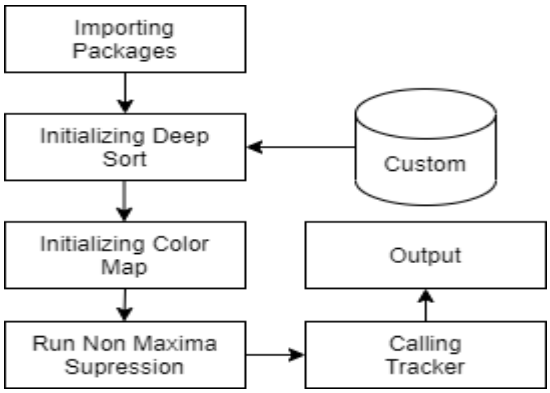

Fig 6.4 YOLO Deepsort Architecture

Below are the examples of what datasets have been used for training purposes. It can be observed from Fig 8.5, which shows the detection for a person based on the COCO dataset. This dataset contains a large number of classes ranging from Cat to Car to Person and so on.

Fig 6.5 COCO Dataset Sample

Similarly, Fig 8.6 below shows the custom dataset used for face mask detection. This custom dataset contains 600 images with annotations made for every object present in the frame. The need for creating a custom dataset was because the COCO dataset doesn't contain classes for face mask detection [6].

Fig 6.6 Custom Dataset Sample

Based on the above figure, the annotation was created for different classes present in the frame, it can be observed from Fig 8.7, it contains 2 different classes (0 & 2). The classes use for face mask detection are 0 for Good, 1 for None & 2 for Bad respectively.

```
# Image 1
0 0.0655 0.7021276595744681 0.121 0.3037417461482025
2 0.42275 0.06456346294937637 0.0405 0.06603081438004402
0 0.321 0.05979457079970653 0.058 0.06969919295671313
0 0.053 0.02090975788701394 0.054 0.04035216434336023
0 0.2575 0.17351430667644901 0.054 0.08143800440205429
0 0.346 0.18525311812179016 0.068 0.10051357300073367
2 0.5875 0.20763022743947177 0.041 0.07043286867204696
2 0.55 0.264490095377843 0.06 0.08437270726338958
0 0.68725 0.14343360234776228 0.0565 0.0946441672780631
0 0.825 0.1966250917094644 0.047 0.08070432868672046
0 0.77575 0.4787234042553192 0.0835 0.10491562729273661
0 0.9315 0.3547322083639032 0.061 0.08290535583272193
0 0.49525 0.41709464416727804 0.0685 0.11225238444607484
```

Fig 6.7 Annotations for Objects based on Images

Similarly, the other annotation file was created based on Person Object Detection for creating bounding boxes based on objects detected in the frame. It can be observed from Fig 8.8 below, which contains a single class (0 for Person), the output goal for social distancing is to detect the person in a frame, and based on distance between the other object, measurement is calculated. For calculating the distance between objects, the Euclidean Distance formula is used.

```
#Image 1 - Person Annotations
0 0.3016129032258065 0.41244239631336405 0.0935483870967742 0.1382488479262673
0 0.2161290322580645 0.6278801843317973 0.11612903225806452 0.1774193548387097
0 0.6653225806451613 0.37672811059907835 0.07903225806451612 0.12672811059907835
0 0.7879032258064517 0.619815668202765 0.09838709677419355 0.14746543778801843
```

Fig 6.8 Annotations of Objects based on Images

Below is the training graph plotted for the training of custom dataset. The custom dataset used in this research is related to face mask, the epoch for which it was trained is 4000 Epochs, it can be observed from Fig 8.9, the loss vs the epochs were getting reduced after 1200 Epochs and remained constant throughout the last epoch. This explains that the training loss was minimized till 1200 and thereafter it was constant, which means that the training epochs should've been set around 2000, because the more number of iterations present in training the data, the more computing power is needed for performing [8].

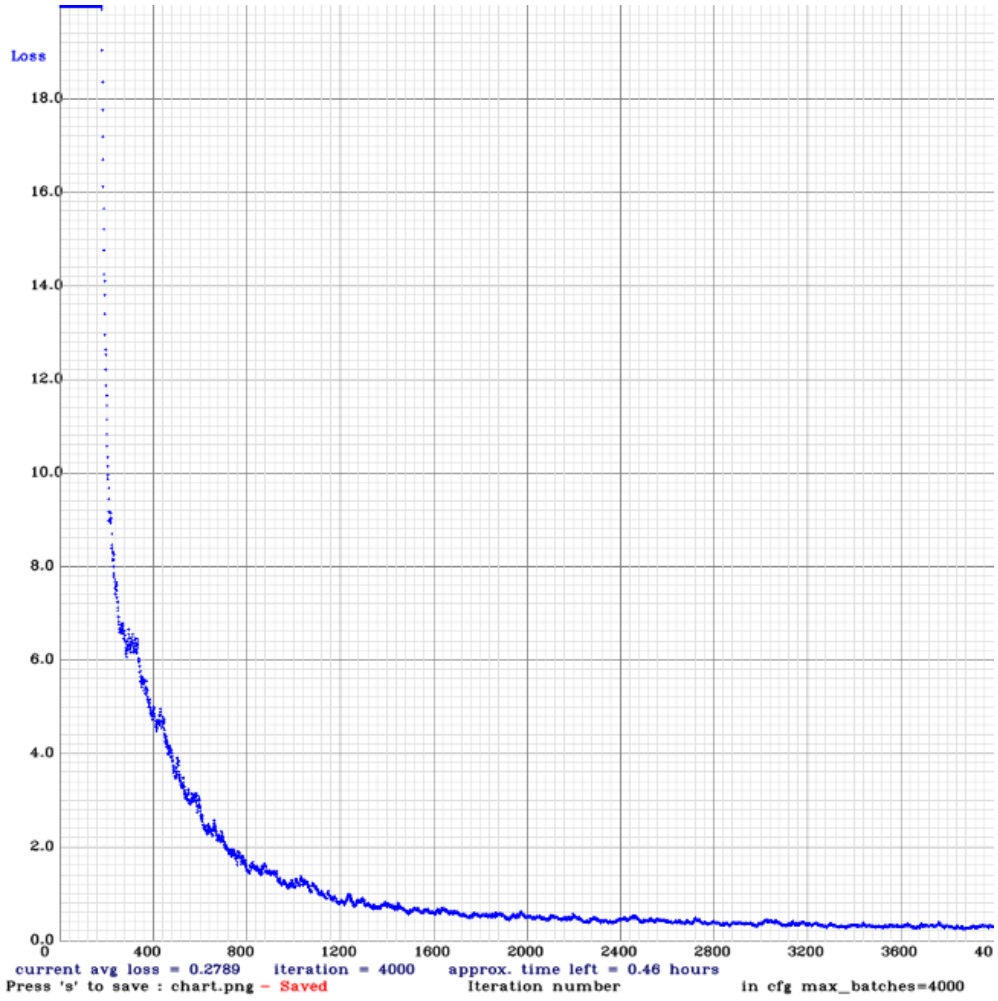

Fig 6.9 Iteration Graph for Training Custom Data

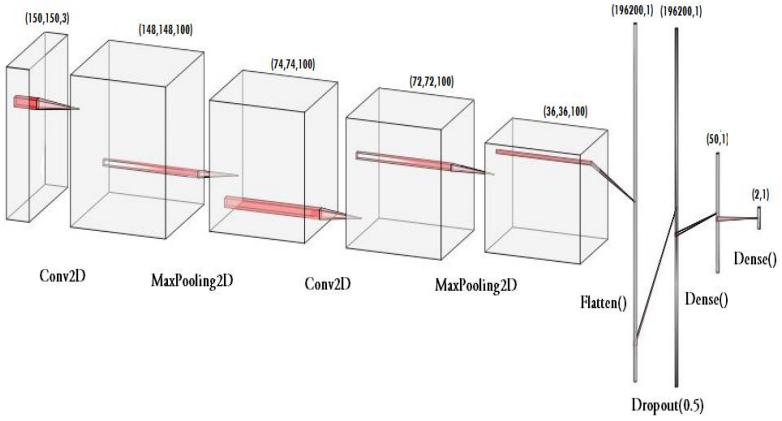

Fig 6.10 CNN Model for Face Mask

6.2 Step 5: Pre-Training the CNN model

After building our model, let us create the '*train_generator*' and '*validation_generator*' to fit them to our model in the next step. We see that there are a total of *2200* images in the *training set* and *551* images in the *test set [7]*.

```
Found 2200 images belonging to 2 classes.

Found 551 images belonging to 2 classes.
```

Step 6: Training the CNN model

This step is the main step where we fit our images in the training set and the test set to our sequential model we built using *keras* library. We have trained the model for *30 epochs* (iterations). However, we can train for more number of epochs to attain higher accuracy lest there occurs *over-fitting*.

```
history=model.fit_generator(train_generator,
                    epochs=30,
                    validation_data=validation_generator,
                    callbacks=[checkpoint])>>Epoch 30/30
220/220 [==============================] - 231s 1s/step - loss:                0.0368 - acc: 0.9886 -
val_loss: 0.1072 - val_acc: 0.9619
```

We see that after the 30th epoch, our model has an accuracy of *98.86%* with the training set and an accuracy of *96.19%* with the test set. This implies that it is well trained without any over-fitting.

Step 7: Labeling the information

After building the model, we label two probabilities for our results, ***['0'** as **'without_mask'** and **'1'** as **'with_mask']***. We are also setting the boundary rectangle color using the RGB values. ***['RED'** for **'without_mask'** and **'GREEN'** for **'with_mask']***

```
labels_dict={0:'without_mask',1:'with_mask'}
color_dict={0:(0,0,255),1:(0,255,0)}
```

Step 8: Importing the face detection program

After this, we intend to use it to detect if we are wearing a face mask using our PC's webcam. For this, first, we need to implement face detection. In this, we are using the *Haar Feature-based Cascade Classifiers* for detecting the features of the face.

```
face_clsfr=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
```

This cascade classifier is designed by *OpenCV* to detect the *frontal face* by training thousands of images. The .xml file for the same needs to be downloaded and used in detecting the face. I have uploaded the file in my GitHub repository.

Step 9: Detecting the Faces with and without Masks

In the last step, we use the OpenCV library to run an infinite loop to use our web camera in which we detect the face using *Cascade Classifier*.

The code `webcam = cv2.VideoCapture(0)` denotes the usage of webcam.

The model will predict the possibility of each of the two classes `([without_mask, with_mask])`. Based on which probability is higher, the label will be chosen and displayed around our faces.

CHAPTER 7

HARDWARE EQUIPMENT AND ITS DESCRIPTION

For the hardware we are using Arduino UNO and mlx90614 sensor to detect the temperature.

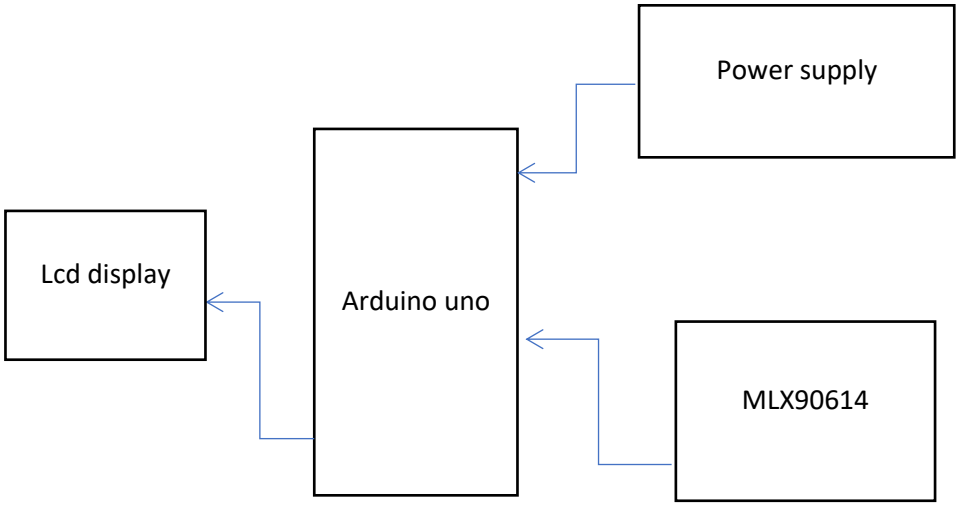

Fig 7.1: Block diagram for detecting the temperature

7.1 ARDUINO UNO:

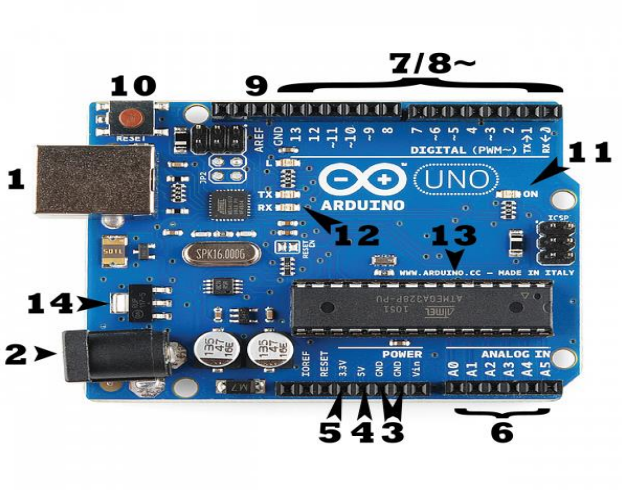

Fig 7.2: Arduino UNO

Power (USB / Barrel Jack)

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply (like this) that is terminated in a barrel jack. In the picture above the USB connection is labeled **(1)** and the barrel jack is labeled **(2)**.

The USB connection is also how you will load code onto your Arduino board. More on how to program with Arduino can be found in our Installing and Programming Arduino tutorial.

NOTE: Do NOT use a power supply greater than 20 Volts as you will overpower (and thereby destroy) your Arduino. The recommended voltage for most Arduino models is between 6 and 12 Volts.

Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjuction with a breadboard and some wire. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

- **GND (3)**: Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- **5V (4) & 3.3V (5)**: As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
- **Analog (6)**: The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
- **Digital (7)**: Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
- **PWM (8)**: You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have a tutorial on PWM, but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).

- **AREF (9)**: Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Reset Button

Just like the original Nintendo, the Arduino has a reset button **(10)**. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

Power LED Indicator

Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' **(11)**. This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

TX RX LEDs

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear – once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs . These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

Main IC

The black thing with all the metal legs is an IC, or Integrated Circuit . Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC's from the ATMEL company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

Voltage Regulator

The voltage regulator is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says – it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn

away an extra voltage that might harm the circuit. Of course, it has its limits, so don’t hook up your Arduino to anything greater than 20 volts.

7.1.1 Micro Controller in Arduino

ATMEGA328P is high performance, low power controller from Microchip. ATMEGA328P is an 8-bit microcontroller based on AVR RISC architecture. It is the most popular of all AVR controllers as it is used in ARDUINO boards.

Atmega328 is an Atmel microcontroller, which is used in Arduino UNO board. Here's its image: Here are few of its features: Atmega328 has 28 pins in total. It has 3 Ports in total which are named as Port B, Port C and Port D

Actually Arduino UNO is a Single Micro-controller board. And the name of this Micro Controller is ATmega328p which is a product of ATmel. 32 - represents it's flash memory capacity that is 32KB. 8 - represents it's cpu type that is of 8 bit. p - simply denotes that it needs less power to work than it earlier version.

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button.

Fig 7.3 ATmega328

7.1.2 Features

These features consist of advanced RISC architecture,

- Good performance,
- Low power consumption,
- Real timer counter having separate oscillator,
- 6 PWM pins,
- Programmable Serial USART,
- Programming lock for software security,
- Throughput up to 20 MIPS etc.

7.1.3 PIN DIAGRAM

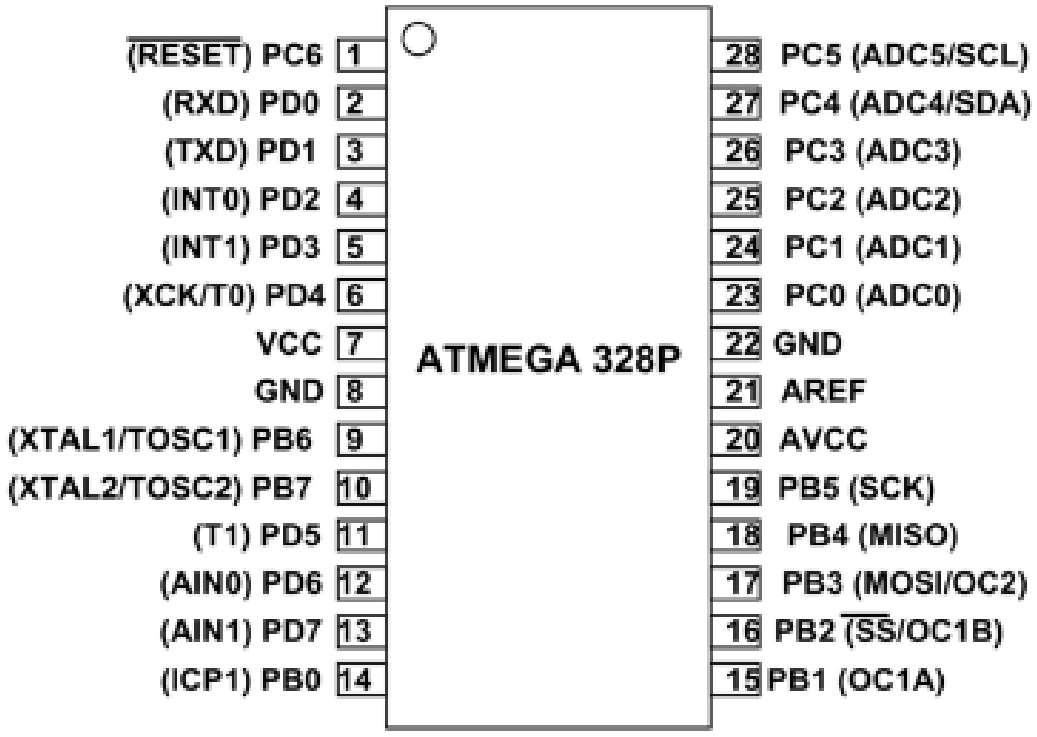

Fig 7.4 pin diagram of ATmega 328p

The Atmega328 is a very popular microcontroller chip produced by Atmel. It is an 8-bit microcontroller that has 32K of flash memory, 1K of EEPROM, and 2K of internal SRAM. The Atmega328 is one of the microcontroller chips that are used with the popular Arduino Duemilanove boards.

ATMEGA328P is high performance, low power controller from Microchip. ATMEGA328P is an 8-bit microcontroller based on AVR RISC architecture. It is the most popular of all AVR controllers as it is used in ARDUINO boards. It is an 8-bit and 28 Pins AVR Microcontroller, manufactured by Microchip, follows RISC Architecure and has a flash type program memory of 32KB. It has an EEPROM memory of 1KB and its SRAM memory is of 2KB. ... It also has 3 builtin Timers, two of them are 8 Bit timers while the third one is 16-Bit Timer.

The ATmega328/P is a low-power CMOS 8-bit microcontroller based on the AVR® enhanced RISC (reduced instruction set computer) architecture. In Order to maximize performance and parallelism, the AVR uses Harvard architecture – with separate memories and buses for program and data

1: Atmega328P and Atmega328 are the same every sense architecturally.

2: Atmega328P just consumes lower power than Atmega328.

Look up the numbers in the datasheet. This means that the 328P is manufactured in a finer process than the 328Actually Arduino UNO is a Single Micro-controller board. And the name of this Micro Controller is ATmega328p which is a product of ATmel. 32 – represents it's flash memory capacity that is 32KB. 8 represent its CPU type that is of 8 bit. p – simply denotes that it needs less power to work than it earlier version.

7.1.4 TYPES OF MEMORY

The ATmega328P has three types of memory hardware:

- 32 KB of In-System Programmable (ISP) Flash program memory.
- 2 KB of SRAM (Static Random-Access Memory)
- 1 KB of EEPROM (Electrically Erasable Programmable Read-Only Memory)

ISP MEMORY

In-system programming (ISP), also called in-circuit serial programming (ICSP), is the ability of some programmable logic devices, microcontrollers, and other embedded devices to be programmed while installed in a complete system, rather than requiring the chip to be programmed prior to installing it into the system.

SRAM

SRAM (static RAM) is random access memory (RAM) that retains data bits in its memory as long as power is being supplied. Unlike dynamic RAM (DRAM), which stores bits in cells consisting of a capacitor and a

transistor, SRAM does not have to be periodically refreshed.

EEPROM

EEPROM (electrically erasable programmable read-only memory) is user-modifiable read-only memory (ROM) that can be erased and reprogrammed (written to) repeatedly through the application of higher than normal electrical voltage. Unlike EPROM chips, EEPROMs do not need to be removed from the computer to be modified.

7.2 DESCRIPTION OF ON-BOARD HARDWARE COMPONENTS

7.2.1 Power Supply

The input to the circuit is applied from the regulated power supply. The a.c. input i.e., 230V from the mains supply is step down by the transformer to 12V and is fed to a rectifier. The output obtained from the rectifier is a pulsating d.c voltage. So in order to get a pure d.c voltage, the output voltage from the rectifier is fed to a filter to remove any a.c components present even after rectification. Now, this voltage is given to a voltage regulator to obtain a pure constant dc voltage.

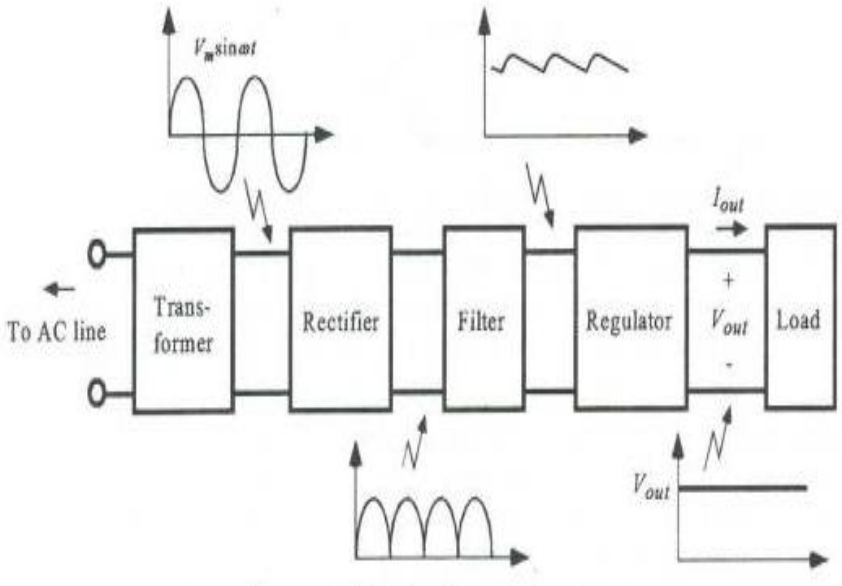

Fig 7.5 Circuit diagram for hardware component

Transformer:

Usually, DC voltages are required to operate various electronic equipment and these voltages are 5V, 9V or 12V. But these voltages cannot be obtained directly. Thus the a.c input available at the mains supply i.e., 230V is to be brought down to the required voltage level. This is done by a transformer. Thus, a step down transformer is employed to decrease the voltage to a required level.

Rectifier:

The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification.

Filter:

Capacitive filter is used in this project. It removes the ripples from the output of rectifier and smoothens the D.C. Output received from this filter is constant until the mains voltage and load is maintained constant. However, if either of the two is varied, D.C. voltage received at this point changes. Therefore a regulator is applied at the output stage.

Voltage regulator:

As the name itself implies, it regulates the input applied to it. A voltage regulator is an electrical regulator designed to automatically maintain a constant voltage level. In this project, power supply of 5V and 12V are required. In order to obtain these voltage levels, 7805 and 7812 voltage regulators are to be used. The first number 78 represents positive supply and the numbers 05, 12 represent the required output voltage levels.

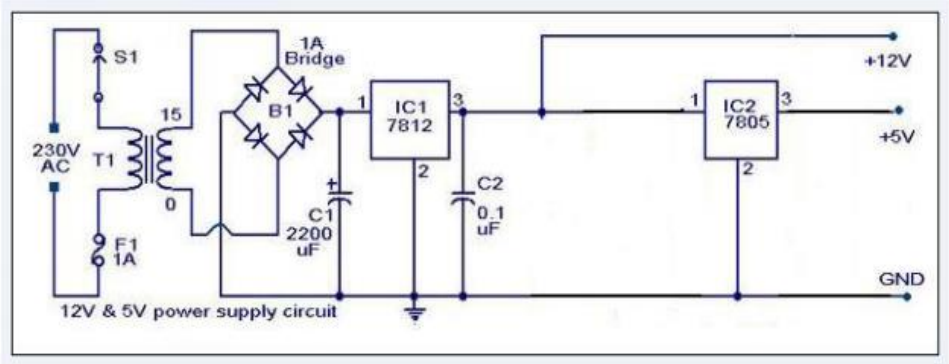

Fig 7.6 Power supply circuit diagram

7.3 LCD DISPLAY

LCD refers to 'Liquid Crystal Display'. A liquid crystal display or LCD draws its definition from its name itself. It is combination of two states of matter, the solid and the liquid. LCD uses a liquid crystal to produce a visible image. Liquid crystal displays are super-thin technology display screen that are generally used in laptop computer screen, TVs, cell phones and portable video games. LCD's technologies allow displays to be much thinner when compared to cathode ray tube (CRT) technology.

Liquid crystal display is composed of several layers which include two polarized panel filters and electrodes. LCD technology is used for displaying the image in notebook or some other electronic devices like mini computers. Light is projected from a lens on a layer of liquid crystal. This combination of colored light with the grayscale image of the crystal (formed as electric current flows through the crystal) forms the colored image. This image is then displayed on the screen.

An LCD is either made up of an active matrix display grid or a passive display grid. Most of the Smartphone's with LCD display technology uses active matrix display, but some of the older displays still make use of the passive display grid designs. Most of the electronic devices mainly depend on liquid crystal display technology for their display. The liquid has a unique advantage of having low power consumption than the LED or cathode ray tube [9].

Liquid crystal display screen works on the principle of blocking light rather than emitting light. LCD's requires backlight as they do not emits light by them. We always use devices which are made up of LCD's displays which are replacing the use of cathode ray tube. Cathode ray tube draws more power compared to LCD's and are also heavier and bigger.

7.3.1 Construction of LCD:

Simple facts that should be considered while making an LCD:

The basic structure of LCD should be controlled by changing the applied current.

We must use a polarized light.

Liquid crystal should able be to control both of the operation to transmit or can also able to change the polarized light.

As mentioned above that we need to take two polarized glass pieces filter in the making of the liquid crystal. The glass which does not have a polarized film on the surface of it must be rubbed with a special polymer

which will create microscopic grooves on the surface of the polarized glass filter. The grooves must be in the same direction of the polarized film. Now we have to add a coating of pneumatic liquid phase crystal on one of the polarized filter of the polarized glass. The microscopic channel cause the first layer molecule to align with filter orientation. When the right angle appears at the first layer piece, we should add a second piece of glass with the polarized film. The first filter will be naturally polarized as the light strikes it at the starting stage.

Thus the light travels through each layer and guided on the next with the help of molecule. The molecule tends to change its plane of vibration of the light in order to match their angle. When the light reaches to the far end of the liquid crystal substance, it vibrates at the same angle as that of the final layer of the molecule vibrates. The light is allowed to enter into the device only if the second layer of the polarized glass matches with the final layer of the molecule.

The principle behind the LCD's is that when an electrical current is applied to the liquid crystal molecule, the molecule tends to untwist. This causes the angle of light which is passing through the molecule of the polarized glass and also cause a change in the angle of the top polarizing filter. As a result a little light is allowed to pass the polarized glass through a particular area of the LCD. Thus that particular area will become dark compared to other. The LCD works on the principle of blocking light. While constructing the LCD's, a reflected mirror is arranged at the back. An electrode plane is made of indium-tin oxide which is kept on top and a polarized glass with a polarizing film is also added on the bottom of the device. The complete region of the LCD has to be enclosed by a common electrode and above it should be the liquid crystal matter.

Next comes to the second piece of glass with an electrode in the form of the rectangle on the bottom and, on top, another polarizing film. It must be considered that both the pieces are kept at right angles. When there is no current, the light passes through the front of the LCD it will be reflected by the mirror and bounced back. As the electrode is connected to a battery the current from it will cause the liquid crystals between the common-plane electrode and the electrode shaped like a rectangle to untwist. Thus the light is blocked from passing through. That particular rectangular area appears blank.

7.3.2 Advantages of an LCD's:

- LCDs consumes less amount of power compared to CRT and LED
- LCDs are consist of some microwatts for display in comparison to some mill watts for LED's
- LCDs are of low cost
- Provides excellent contrast

- LCDs are thinner and lighter when compared to cathode ray tube and LED

7.3.3 Disadvantages of an LCD's:

1. Require additional light sources
2. Range of temperature is limited for operation
3. Low reliability
4. Speed is very low
5. LCD's need an AC drive

7.3.4 Applications of Liquid Crystal Display:

Liquid crystal technology has major applications in the field of science and engineering as well on electronic devices.

- Liquid crystal thermometer
- Optical imaging
- The liquid crystal display technique is also applicable in visualization of the radio frequency waves in the waveguide
- Used in the medical applications

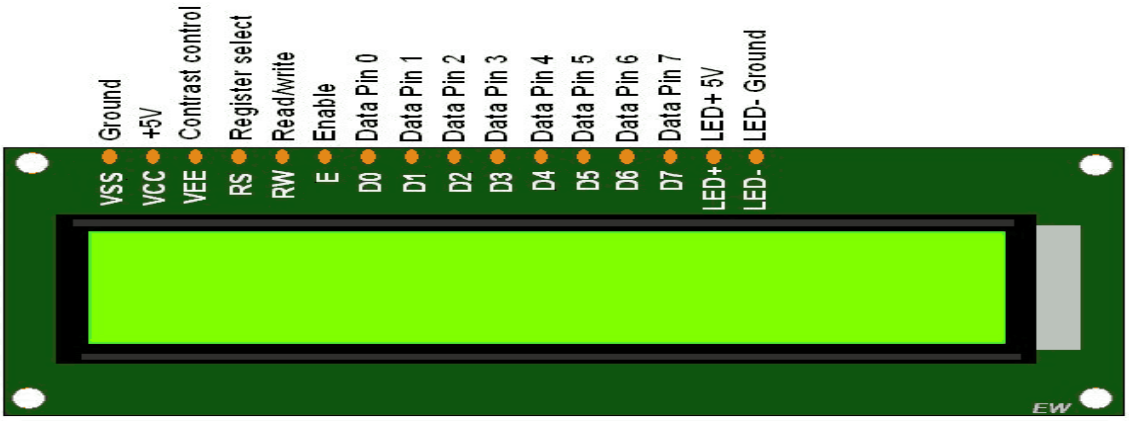

Fig 7.7 LCD Display

Below is the table of description of pins in the LCD

Table 7.1 Pins in LED

Pin No:	**Pin Name:**	**Description**
1	Vss (Ground)	Ground pin connected to system ground
2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contrast.
4	Register Select	Connected to Microcontroller to shit between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledgement
7	Data Pin 0	
8	Data Pin 1	
9	Data Pin 2	

10	Data Pin 3	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free.
11	Data Pin 4	
12	Data Pin 5	
13	Data Pin 6	
14	Data Pin 7	
15	LED Positive	Backlight LED pin positive terminal
16	LED Negative	Backlight LED pin negative terminal

Features of 16×2 LCD module

Operating Voltage is 4.7V to 5.3V

Current consumption is 1mA without backlight

Alphanumeric LCD display module, meaning can display alphabets and numbers

Consists of two rows and each row can print 16 characters.

Each character is built by a 5×8pixel box

Can work on both 8-bit and 4-bit mode

It can also display any custom generated characters

Available in Green and Blue Backlight

16x2 Display Equivalents

Dot Matrix LED Display, 7-Segment LED Display, OLED Display, TFT LCD Screen Display

7.3.5 Brief Description on LCD modules

LCD modules are very commonly used in most embedded projects, the reason being its cheap price, availability and programmer friendly. Most of us would have come across these displays in our day-to-day life, either at PCO's or calculators. The appearance and the pinouts have already been visualized above now let us get a bit technical.

16×2 LCD is named so because; it has 16 Columns and 2 Rows. There are a lot of combinations available like, 8×1, 8×2, 10×2, 16×1, etc. but the most used one is the 16×2 LCD. So, it will have (16×2=32) 32 characters in total and each character will be made of 5×8 Pixel Dots. A Single character with all its Pixels is shown in the below picture.

Now, we know that each character has (5×8=40) 40 Pixels and for 32 Characters we will have (32×40) 1280 Pixels. Further, the LCD should also be instructed about the Position of the Pixels. Hence it will be a hectic task to handle everything with the help of MCU, hence an Interface IC like HD44780is used, which is mounted on the backside of the LCD Module itself. The function of this IC is to get the Commands and Data from the MCU and process them to display meaningful information onto our LCD Screen. You can learn how to interface an LCD using the above mentioned links. If you are an advanced programmer and would like to create your own library for interfacing your Microcontroller with this LCD module then you have to understand the HD44780 IC is working and commands which can be found its datasheet.

Display data RAM (DDRAM)

Pin No.	Name	Description
1	**VSS**	Power supply (GND)
2	**VCC**	Power supply (+5V)
3	**VEE**	Contrast adjust
4	**RS**	0 = Instruction input 1 = Data input
5	**R/W**	0 = Write to LCD module 1 = Read from LCD module
6	**EN**	Enable signal
7	**D0**	Data bus line 0 (LSB)
8	**D1**	Data bus line 1
9	**D2**	Data bus line 2
10	**D3**	Data bus line 3
11	**D4**	Data bus line 4
12	**D5**	Data bus line 5
13	**D6**	Data bus line 6
14	**D7**	Data bus line 7 (MSB)
15	**LED+**	Back Light VCC
16	**LED-**	Back Light GND

Fig 7.8 DDRAM-Display Data RAM

Display data RAM (DDRAM) stores show data represented in eight-bit individual codes. Its all-encompassing capacity is 80 X eight bits, or 80 characters. The area in display data RAM (DDRAM) that isn't utilized for showcase can be utilized as advanced records RAM. So whatever you send on the DDRAM is actually displaced on LCD. For LCDs like 1x16, only sixteen characters are visible, so anything you write after sixteen chars is written in DDRAM but is not seen to the user.

CGROM - Character Generator ROM

The character Generator ROM Generates 5 x 8 dot or 5 x 10 dot character patterns from 8-bit individual codes. It can generate 208 5 x 8 dot individual styles and 32 5 x 10 dot character styles.

CGRAM - character Generator RAM

As clean from the name, CGRAM is used to create custom characters in liquid crystal display. Inside the character generator RAM, the user can rewrite individual patterns through software program. For 5 x 8 dots, 8 individual patterns may be written, and for 5 x 10 dots, four-character patterns may be written.

BF - Busy Flag

Busy Flag is a status indicator flag for lcd. When we send a command or records to the liquid crystal show for processing, this flag is set (ie., BF =1) and as quickly as instruction is finished efficaciously this flag is cleared (BF = zero). This is beneficial in producing and exact amount of delay for the liquid crystal display processing to read Busy Flag, the circumstance RS = 0 and R/W = 1 should be met and The MSB of the liquid crystal show facts bus (D7) act as busy flag. When BF = 1 manner liquid crystal display is busy and will no longer take delivery of subsequent command or records and BF = zero approach liquid crystal display is ready for the subsequent command or records to procedure.

Instruction Register (IR) and Data Register (DR)

There are two eight-bit registers in HD44780 controller practise and information sign in. Guidance sign up corresponds to the take a look at in in which you send commands to liquid crystal display e.g. Lcd shift command, liquid crystal show clear, lcd deal with and lots of others. And information join up is used for storing statistics this is to be displayed on liquid crystal display. When send the enable signal of the liquid crystal display is said, the records at the pins is latched in to the information sign in and data is then moved routinely to the DDRAM and therefore is displayed at the liquid crystal display. Data Register is not only used for sending records to DDRAM however additionally for CGRAM, the address in which you want to send the records, is determined through the instruction you send to liquid crystal display.

Commands and Instruction set

Only the instruction register (IR) and the data register (DR) of the liquid crystal display can be managed by way of using the MCU. Earlier than starting the internal operation of the liquid crystal display, control records is temporarily stored into those registers to permit interfacing with numerous MCUs, which carry out at different speeds, or diverse peripheral control devices. The inner operation of the liquid crystal dispaly is

decided through indicators dispatched from the MCU. These signlas, which include register selection signal (RS), read/write signal (R/W), and the data bus (DB0 to DB7), make up the liquid crystal display instructions. There are 4 classes of instructions that:

- Designate lcd capabilities, collectively with display format, data length, and many others.
- Set internal RAM addresses
- Perform data transfer with internal RAM

Below is a brief listing of beneficial commands which can be used frequently while running at the lcd.

No.	Instruction	Hex	Decimal
1	Function Set: 8-bit, 1 Line, 5x7 Dots	0x30	48
2	Function Set: 8-bit, 2 Line, 5x7 Dots	0x38	56
3	Function Set: 4-bit, 1 Line, 5x7 Dots	0x20	32
4	Function Set: 4-bit, 2 Line, 5x7 Dots	0x28	40
5	EntryMode	0x06	6
6	Display off Cursor off (clearing display without clearing DDRAM content)	0x08	8
7	Display on Cursor on	0x0E	14
8	Display on Cursor off	0x0C	12
9	Display on Cursor blinking	0x0F	15
10	Shift entire display left	0x18	24
12	Shift entire display right	0x1C	30
13	Move cursor left by one character	0x10	16
14	Move cursor right by one character	0x14	20
15	Clear Display (also clear DDRAM content)	0x01	1
16	Set DDRAM address or cursor position on display	0x80+add	128+add
17	Set CGRAM address or set pointer to CGRAM location	0x40+add	64+add

Fig 7.9 Commands of LCD

Sending commands to LCD:

To send commands we in reality want to pick out the command register. The whole thing is identical as we've executed inside the initialization recurring. But we will summarize the common steps and put them in a single subroutine. Following are the steps:

- Flow facts to liquid crystal display port
- Select command register
- Pick out write operation
- Send enable signal
- Await liquid crystal display to system the command

Sending Information to LCD:

To send data we clearly want to pick the data register. The whole lot is identical because the command recurring. Following are the steps:

- Pass facts to liquid crystal display port
- Choose data register
- Choose write operation
- Send enable signal
- Look forward to liquid crystal display to process the data.

7.4 I²C Adapter

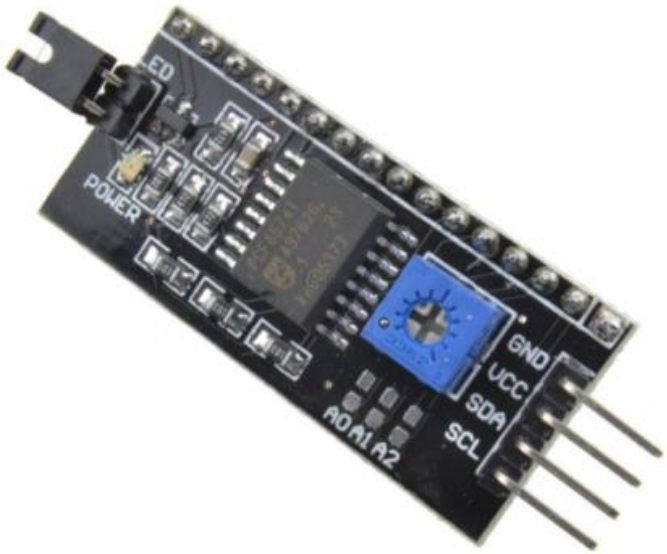

Fig 7.10 I2C adapter

This is a RoHS compliant I2C Serial LCD Daughter board that can be connected to a standard 16×2 or 20×4 Character Display Module that supports 4-bit mode. All Character Modules sold on our site support 4-bit mode, and nearly all commercially available 16×2 and 20×4 line character modules support it too.

This board has a PCF8574 I2C chip that converts I2C serial data to parallel data for the LCD display. There are many examples on the internet for using this board with Arduino. Do a search for "Arduino LCD PCF8574". The I2C address is 0x3F by default, but this can be changed via 3 solder jumpers provided on the board. This allows up to 3 LCD displays to be controlled via a single I2C bus (giving each one it's own address)

Specifications and Features:

1. 5V power supply.
2. Serial I2C control of LCD display using PCF8574.
3. Backlight can be enabled or disabled via a jumper on the board.
4. Contrast control via a potentiometer.
5. Can have 8 modules on a single I2C bus (change address via solder jumpers)address, allowing.
6. Size ：41.6 x 19.2 mm.

7.5 MLX90614

Fig 7.11 MLX90614

The MLX90614 ESF is an Infra-Red thermometer for non-contact temperature measurements. Both the IR sensitive thermopile detector chip and the signal conditioning ASIC are integrated into the same TO-39 can. The Integrated MLX90614 GY-906 is a low noise amplifier, 17-bit ADC, and powerful DSP unit thus achieving high accuracy and resolution of the thermometer.

The user can configure the digital output to be PWM. As a standard, the 10-bit PWM is configured to continuously transmit the measured temperature in the range of -20 to 120 °C, with an output resolution of 0.14 °C.

Applications:

1. High precision non-contact temperature measurements
2. Thermal Comfort sensor for Mobile Air Conditioning control system
3. The temperature sensing element for residential, commercial and industrial building air conditioning Windshield defogging
4. The automotive blind angle detection
5. Industrial temperature control of moving parts
6. Temperature control in printers and copiers
7. Home appliances with temperature control
8. Healthcare
9. Livestock monitoring
10. Movement detection
11. Multiple zone temperature control – up to 127 sensors can be read via common 2 wires

12. Thermal relay/alert
13. Body temperature measurement

Features:

Works from 3.3V to 5V input, Module has power regulator IC built-in.

1. Standard I2C interface with built 2x pull up resistors
2. When measuring the temperature, please maintain a measuring distance of 1 cm
3. Small size, low cost
4. Easy to integrate
5. Factory calibrated in a wide temperature range: -40 to 125 °C for sensor temperature and -70 to 380 °C for object temperature.
6. High accuracy of 0.5°C, over the wide temperature range (0+50 C for both Ta and To)
7. Medical accuracy of 0.1°C in a limited temperature range available on request
8. The measurement resolution of 0.01°C
9. SMBus compatible digital interface for fast temperature readings and building sensor networks
10. Customizable PWM output for continuous reading
11. Simple adaptation for 8 to 16V applications

Working:

contact less temperature sensors which utilizes Laser or IR to calculate the temperature of an object. The MLX90614 is one such sensor that uses IR energy to detect the temperature of an object.

Fig 7.12 Sensor Top view

MLX90614 sensor is manufactured by Melexis Microelectronics Integrated system, it has two devices embedded in it, one is the infrared thermopile detector (sensing unit) and the other is a signal conditioning DSP device (computational unit). It works based on Stefan-Boltzmann law which states that all objects emit IR energy and the intensity of this energy will be directly proportional to the temperature of that object. The sensing unit in the sensor measures how much IR energy is emitted by a targeted object and the computational unit converts it into temperature value using a 17-bit in-built ADC and outputs the data through I2C communication protocol. The sensor measures both the object temperature and ambient temperature to calibrate the object temperature value. The features of MLX90614 sensor is given below, for more details refer the MLX90614 Datasheet.

MLX90614 Infrared Temperature Sensor Features:

- Operating Voltage: 3.6V to 5V
- Object Temperature Range: -70°C to 382.2°C
- Ambient Temperature Range: -40°C to 125°C
- Resolution/Accuracy: 0.02°C

7.6 Measurement distance between the Sensor and the Object

One question that is not directly answered by the datasheet is the measuring distance between the sensor and the object. The value of this distance is given by the term **Field of View (FOV)**, for our sensor the field of view is about 80° [10].

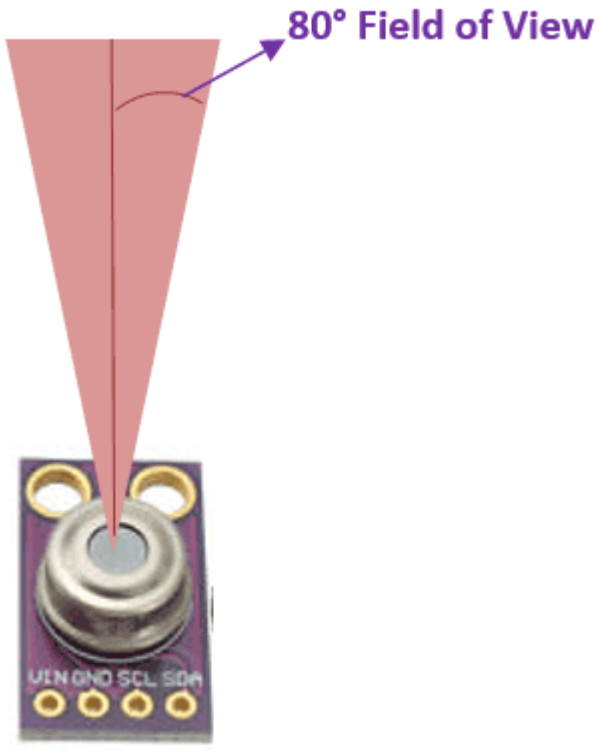

Fig 7.13 Measurement of distance between sensor and object

You can think of the sensing range to be in a conical shape from the point of sensor as show above. So, as we go far from the measuring object the sensing area increase by two folds. Meaning for every 1cm we move away from the object the sensing area grows by 2cm. In our thermal gun we have placed a laser diode on top of the sensor to know where the sensing area of the sensor is currently pointing at. I found that the values were reliable if the gun is pointed at 2cm away from the object and the accuracy goes down as we move away.

CHAPTER 8

FINANCE AND PROJECT MANAGEMENT

8.1 PROJECT MANAGEMENT

Our team main objective is to take on the project that relates the current pandemic. So we have decided to work on Mask Detection and Human Body Temperature Screening. Our project includes both simulation and Hardware, we made use of both software and hardware components like Open CV, Tensor Flow Python-PyCharm, IR sensor, LCD display

8.2 PROJECT TIME MANAGEMENT

We have segregated our whole project duration into 2 stages. Stage-1 deals with Simulation and Stage-2 deals with Hardware.

Stage-1: Software Implementation

We have divided our Stage-1 of our Project into many phases as listed below.

- **Phase-1 (2 weeks) :**

  We spent on searching the area on which our project has to be done. And finally decided to do our project related to Health Care Sector.

- **Phase-2 (2 weeks) :**

  As our project deals with current societal problem, we spent these 2 weeks on researching many articles, referring magazines on how to carry out the project.

- **Phase-3 (1 weeks) :**

  In this span of 2 weeks, we decided what tools has to be used in the project and what technologies has to be involved in the project.

  Below are the tools and technologies used in Mask Detection.

  - Tensor Flow, Machine Learning
  - OpenCV
  - Computer Vision
  - Python-PyCharm

- **Phase-4 (3 weeks) :**

  We started implementing our project making use of required Neural Networks (CNN) from OpenCV. The programming language we used in the project is Python. And we have used TensorFlow for training our proposed model such that it detects mask against sample datasets.

- **Phase-5 (2 weeks) :**

  We decided not to limit our project for sample datasets and hence we extended it to detect live video streams.

Stage-2: Hardware Implementation

- **Phase-1 (1 week) :**

  We have used this 1 week to research about the hardware components that has to be used in the project.

- **Phase-2 (2 weeks):**

  In order to detect temperature, we have used Infrared Sensor, Arduino Board. To display the results, we used LCD Display.

- **Phase-3 (1 week) :**

  Spent on the analysing the results and arriving at conclusions. The rest of the time in this phase is spent on Manuscript submissions and final analysis of the work.

8.3 FINANCE MANAGEMENT:

Below, we have listed the price of the hardware components that we have used in our project.

COMPONENTS USED	PRICE OF THE COMPONENT (IN Rs)
MLX90614	1200
ARDUINO UNO	750
LCD DISPLAY	350

CHAPTER 9

RESULTS AND CONCLUSION

9.1 Results:

Fig 9.1 Input-1

Above image consists of two persons wearing mask which we have given the image as an input.

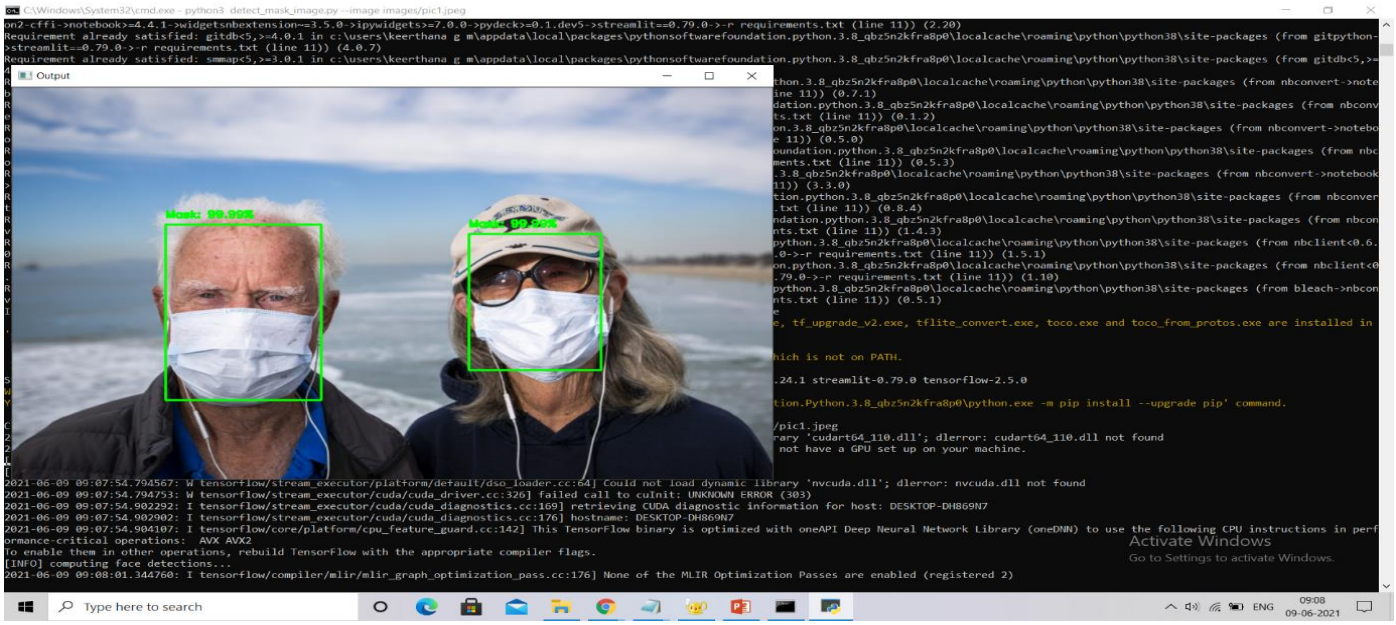

Fig 9.2 Output-1

As the two persons are wearing mask, our developed model detected those two masks the percentage of wearing mask.

Fig 9.3 Input-2

In the above image, two persons are not wearing mask whereas one person wears a mask.

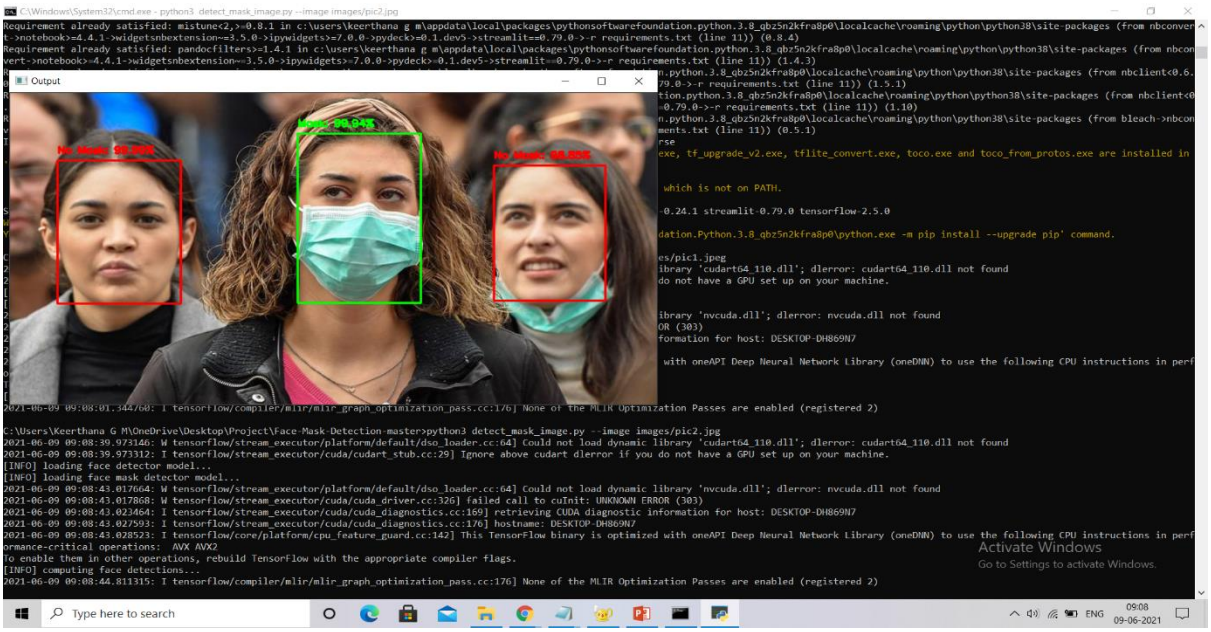

Fig 9.4 Output-2

These images clearly show how the proposed model detects whether the person is wearing a mask or not with more than 98% accuracy. The green colour indicates the presence of mask whereas the red indicates the absence of mask both with around 98-99.9% accuracy.

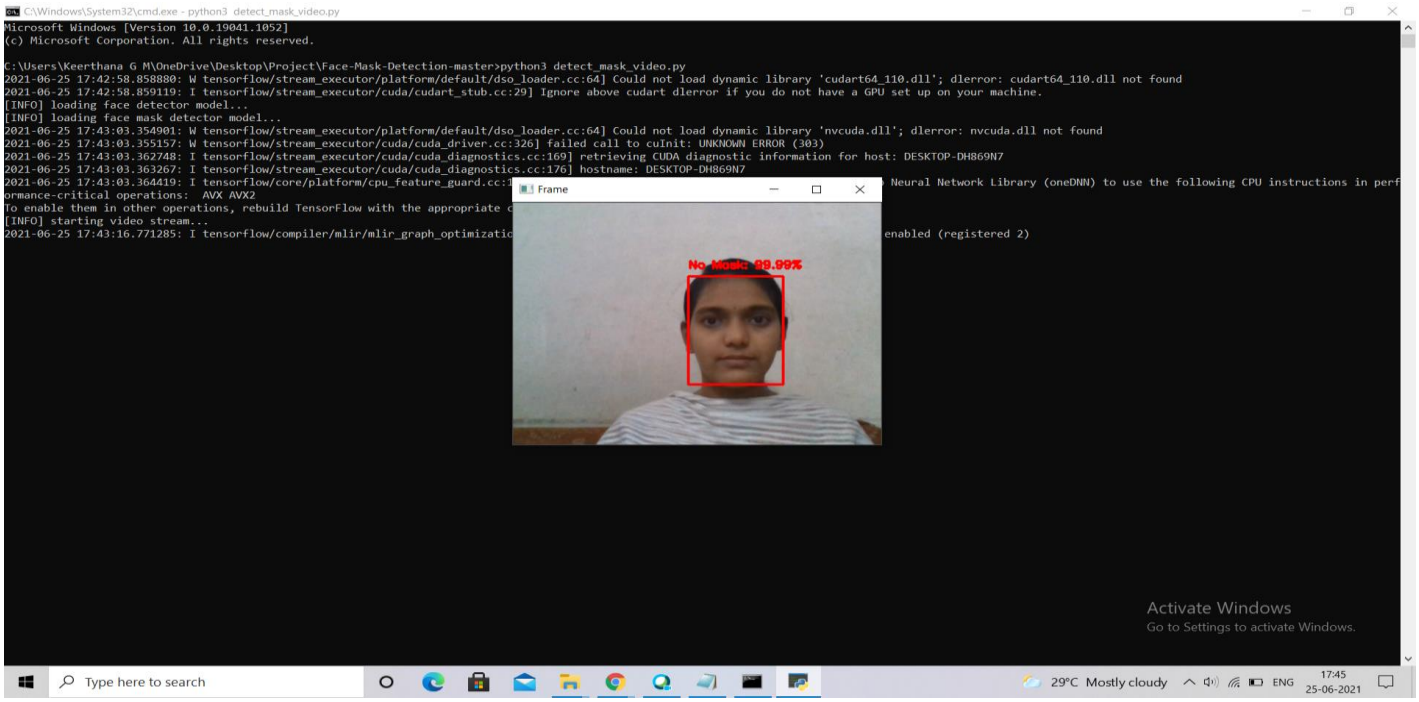

Fig 9.5 Face mask detection during Video streaming- without Mask

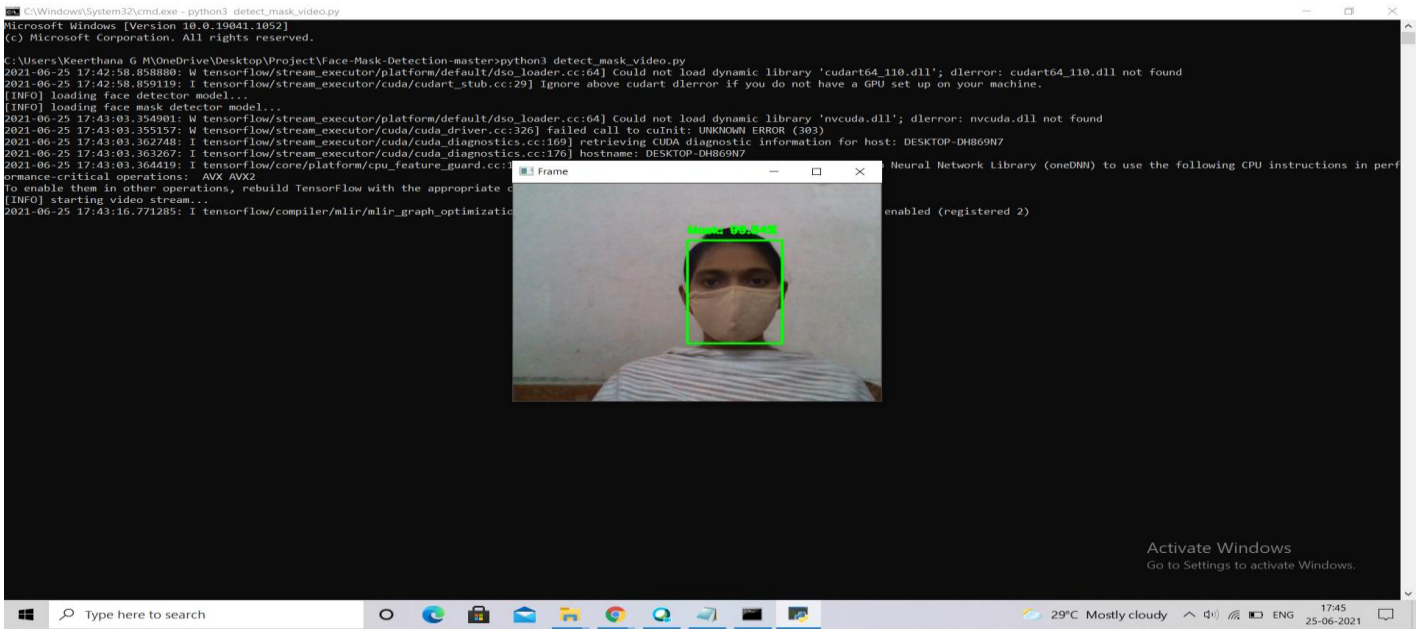

Fig 9.6 Face mask detection during Video streaming- with Mask

These are the screenshots that depict how the model detects whether the person wears a mask or not during live video streaming. These results are also around 98-99% accurate.

9.2 CONCLUSION AND FUTURE SCOPE

9.2.1 CONCLUSION

By the development of face mask detector, it was detected if the person wore a face mask and allowed their entry. The accuracy of the model was achieved and the optimization of the model was a continuous process and a highly accurate solution was built. The model also measured the temperature of the person and made sure that the person was not allowed, if the temperature exceeded the threshold value. Thus it could prevent people from Virus Transmission.

This proposed system will operate in an efficient manner in the current situation where there is a high need to track human body temperature without the actual contact. We have addressed in depth, the identification of face masks that prevent the virus spread along with temperature detection. This solution has the potential to significantly reduce violations by real-time interventions, so the proposed system would improve public safety through saving time and helping to reduce the spread of coronavirus. This solution can be used in places like temples, shopping complexes, metro stations, airports etc.

9.2.2 FUTURE SCOPE

This model can be developed further and many more sub-modules can be added.

Coughing and Sneezing Detection: Chronic coughing and sneezing is one of the key symptoms of COVID-19 infection as per WHO guidelines and also one of the major routes of disease spread to non-infected public. Deep-learning based approach can be proved handy here to detect & limit the disease spread by enhancing our proposed solution

Temperature Screening: Elevated body temperature is another key symptom of COVID-19 infection, at present scenario thermal screening is done using handheld contactless IR thermometers where health worker need to come in close proximity with the person needed to be screened which makes the health workers vulnerable to get infected and also it is practically impossible to capture temperature for each and every person in public places, the proposed use-case can be equipped with thermal cameras based screening to analyze body temperature of the people in public places that can add another helping hand to enforcement agencies to tackle the pandemic effectively.

REFERENCES

[1] M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud and J. -H. Kim, "An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 2020, pp. 1-5, doi: 10.1109/IEMTRONICS51293.2020.9216386.

[2] S. A. Sanjaya and S. Adi Rakhmawan, "Face Mask Detection Using MobileNetV2 in The Era of COVID-19 Pandemic," 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI), 2020, pp. 1-5, doi: 10.1109/ICDABI51230.2020.9325631.

[3] T. Meenpal, A. Balakrishnan and A. Verma, "Facial Mask Detection using Semantic Segmentation," 2019 4th International Conference on Computing, Communications and Security (ICCCS), 2019, pp. 1-5, doi: 10.1109/CCCS.2019.8888092.

[4] R. Suganthalakshmi, A. Hafeeza, P. Abinaya, A.Ganga Devi, 2021, Covid-19 Facemask Detection with Deep Learning and Computer Vision, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ICRADL – 2021 (Volume 09 – Issue 05),

[5] Z. Zhao, P. Zheng, S. Xu and X. Wu, "Object Detection With Deep Learning: A Review," in IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 11, pp. 3212-3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.

[6] Z. Wang and T. S. Kim, "Learning to Recognize Masked Faces by Data Synthesis," 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2021, pp. 036-041, doi: 10.1109/ICAIIC51459.2021.9415252.

[7] S. Sakshi, A. K. Gupta, S. Singh Yadav and U. Kumar, "Face Mask Detection System using CNN," 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), 2021, pp. 212-216, doi: 10.1109/ICACITE51222.2021.9404731.

[8] A. Chavda, J. Dsouza, S. Badgujar and A. Damani, "Multi-Stage CNN Architecture for Face Mask Detection," 2021 6th International Conference for Convergence in Technology (I2CT), 2021, pp. 1-8, doi: 10.1109/I2CT51068.2021.9418207.

[9] Petre, V., & Oancea, C. (2018). Educational Platform for LCD Displays. *2018 International Conference and Exposition on Electrical And Power Engineering (EPE)*, 0042-0045.

[10] A. M. Kassim, H. I. Jaafar, M. A. Azam, N. Abas and T. Yasuno, "Performances study of distance measurement sensor with different object materials and properties," 2013 IEEE 3rd International Conference on System Engineering and Technology, 2013, pp. 281-284, doi: 10.1109/ICSEngT.2013.6650185.

APPENDIX

Algorithm for Face Mask Detection for image as an input

```
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
import numpy as np
import argparse
import cv2
import os
def mask_image():
	# construct the argument parser and parse the arguments
	ap = argparse.ArgumentParser()
	ap.add_argument("-i", "--image", required=True,
		help="path to input image")
	ap.add_argument("-f", "--face", type=str,
		default="face_detector",
		help="path to face detector model directory")
	ap.add_argument("-m", "--model", type=str,
		default="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"])
	net = cv2.dnn.readNet(prototxtPath, weightsPath)

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

	# load the input image from disk, clone it, and grab the image spatial
	# dimensions
	image = cv2.imread(args["image"])
	orig = image.copy()
	(h, w) = image.shape[:2]

	# construct a blob from the image
	blob = cv2.dnn.blobFromImage(image, 1.0, (300, 300),
		(104.0, 177.0, 123.0))

	# pass the blob through the network and obtain the face detections
	print("[INFO] computing face detections...")
	net.setInput(blob)
	detections = net.forward()
```

```
# 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 = image[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)
                face = np.expand_dims(face, axis=0)

                # pass the face through the model to determine if the face
                # has a mask or not
                (mask, withoutMask) = model.predict(face)[0]

                # 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(image, label, (startX, startY - 10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
                cv2.rectangle(image, (startX, startY), (endX, endY), color, 2)

# show the output image
cv2.imshow("Output", image)
cv2.waitKey(0)
if __name__ == "__main__": mask_image()
```

Algorithm for Face Mask Detection for video as an input

```
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
import smtplib
from PIL import ImageGrab
import cv2
SUBJECT = "DETECTED WITHOUT FACEMASK "
TEXT = "Visitor's violated Face Mask Policy. See in the camera to recognize user. A Person has been detected without a face mask. Please Alert the authorities."


def sendMail(noofways):
    #to send the count of people without masks through email
    message = 'Subject: {}\n\n{}'.format(SUBJECT, str(noofways)+"  "+TEXT)
    mail = smtplib.SMTP('smtp.gmail.com', 587)
    mail.ehlo()
    mail.starttls()
    mail.login('ram22aws@gmail.com','Laksh@3223')
    mail.sendmail('ram22aws@gmail.com','ram22aws@gmail.com',message)
    print("email sent")
    mail.close()


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="face_detector",
        help="path to face detector model directory")
    ap.add_argument("-m", "--model", type=str,
        default="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()
time.sleep(2.0)
noofpersons=0

# 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
        #print(box,pred)
        (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)
        if label == "No Mask":
            noofpersons += 1

        # 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"):
            sendMail(noofpersons)
            break

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

SREE VIDYANIKETHAN ENGINEERING COLLEGE
(AUTONOMOUS)
SreeSainath Nagar, A.Rangampet - 517 102

Department of Electronics and Communication Engineering

PROJECT TITLE: Temperature Check Up and Mask Detection – COVID 19 Monitoring Solutions

ABSTRACT: Covid-19 has upended societies and dramatically altered everyday life across the globe. Many constraints have been set on the people on maintaining social distancing, wearing a mask along with periodical temperature check at the entrance of malls, offices, super markets. This project proposes a model which detects whether a person wears a mask or not by training a Covid-19 face mask detector with tensor flow, Deep learning. This model also performs thermal scanning of a person and compares his/her body temperature with normal body temperature. If the measured temperature is comparable with the pre-defined value and the mask is detected, the person is allowed to enter otherwise the entry is denied with an alarm. The proposed model also finds the count of the total number of people who entered, got rejected.

PROJECT BATCH: 21B06

Ekkaluri Rajeswari - 17121A0473
G M Keerthana - 17121A0476
Gundigi Poojitha - 17121A0495
Junju Pranavi - 17121A04A2

GUIDE: Dr. N. Padmaja, M.Tech, Ph.D
Professor,Dept. of ECE

POs Attained:

	Program Outcomes												Program Specific Outcomes			
	Po1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
Temperature check and Mask detection Covid19 Monitoring Solutions	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√

Signature of the Guide

Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV

Arjya Das
Department of Information TechnologyJadavpur University
Kolkata, India
arjyadas1999@gmail.com

Mohammad Wasif Ansari
Department of Information Technology
Jadavpur University
Kolkata, India
razamoeezraza@gmail.com

Rohini Basak
Department of Information TechnologyJadavpur University
Kolkata, India
visitrohinihere@gmail.com

***Abstract*—COVID-19 pandemic has rapidly affected our day- to-day life disrupting the world trade and movements. Wearinga protective face mask has become a new normal. In the near future, many public service providers will ask the customersto wear masks correctly to avail of their services. Therefore,face mask detection has become a crucial task to help global society. This paper presents a simplified approach to achievethis purpose using some basic Machine Learning packages like TensorFlow, Keras, OpenCV and Scikit-Learn. The proposed method detects the face from the image correctly and then identifies if it has a mask on it or not. As a surveillance task performer, it can also detect a face along with a mask in motion. The method attains accuracy up to 95.77% and 94.58% respectively on two different datasets. We explore optimizedvalues of parameters using the Sequential Convolutional Neural Network model to detect the presence of masks correctly withoutcausing over-fitting.**

Keywords—Coronavirus, Covid-19, Machine Learning, FaceMask Detection, Convolutional Neural Network, TensorFlow

I. Introduction

According to the World Health Organization (WHO)'s official Situation Report – 205, coronavirus disease 2019 (COVID-19) has globally infected over 20 million people causing over 0.7million deaths [1]. Individuals with COVID-19 have had a wide scope of symptoms reported – going from mellow manifestations to serious illness. Respiratory problems like shortness of breath or difficulty in breathing is one of them. Elder people having lung disease can possess serious complications from COVID-19 illness as they appear to be at higher risk [2]. Some common human coronaviruses that infect public around the world are 229E, HKU1, OC43, and NL63. Before debilitating individuals, viruses like 2019-nCoV, SARS-CoV, and MERS-CoV infect animals and evolve to human coronaviruses [3]. Persons having respiratory problems can expose anyone (who is in closecontact with them) to infective beads. Surroundings of a tainted individual can cause contact transmission as droplets carrying virus may withal arrive on his adjacent surfaces [4].

To curb certain respiratory viral ailments, including COVID-19, wearing a clinical mask is very necessary. The public should be aware of whether to put on the maskfor source control or aversion of COVID-19. Potential points of interest of the utilization of masks lie in reducing vulnerability of risk from a noxious individual during the "pre-symptomatic" period and stigmatization of discrete persons putting on masks to restraint the spread of virus. WHO stresses on prioritizing medical masks and respirators for health care assistants[4]. Therefore, face mask detection has become a crucial task in present global society.

Face mask detection involves in detecting the location of the face and then determining whether it has a mask on itor not. The issue is proximately cognate to general object detection to detect the classes of objects. Face identification categorically deals with distinguishing a specific group of entities i.e. Face. It has numerous applications, such as autonomous driving, education, surveillance, and so on [5]. This paper presents a simplified approach to serve the above purpose using the basic Machine Learning (ML) packages such as TensorFlow, Keras, OpenCV and Scikit-Learn.

The rest of the paper is organized as follows: Section II explores related work associated with face mask detection. Section III discusses the nature of the used dataset. Section IV presents the details of the packages incorporated to build the proposed model. Section V gives an overview of our method. Experimental results and analysis are reported in section VI. Section VII concludes and draws the line towards future works.

II. Related Work

In face detection method, a face is detected from an image that has several attributes in it. According to [21],research into face detection requires expression recognition, face tracking, and pose estimation. Given a solitary image,the challenge is to identify the face from the picture. Face detection is a difficult errand because the faces changein size, shape, color, etc and they are not immutable. It becomes a laborious job for opaque image impeded by some other thing not confronting camera, and so forth. Authorsin [22] think occlusive face detection comes with two major challenges: 1) unavailability of sizably voluminous

2020 IEEE 17th India Council International Conference (INDICON) | 978-1-7281-6916-3/20/$31.00 ©2020 IEEE | DOI: 10.1109/INDICON49873.2020.9342585

datasets containing both masked and unmasked faces, and 2) exclusion of facial expression in the covered area. Utilizing the locally linear embedding (LLE) algorithm and the dictionaries trained on an immensely colossal pool of masked faces, synthesized mundane faces, several mislaid expressions can be recuperated and the ascendancy of facial cues can be mitigated to great extent. According to the work reported in [11], convolutional neural network (CNNs) incomputer vision comes with a strict constraint regarding the size of the input image. The prevalent practice reconfigures the images before fitting them into the network to surmount the inhibition.

Here the main challenge of the task is to detect the face from the image correctly and then identify if it has a mask on it or not. In order to perform surveillance tasks, the proposed method should also detect a face along with a mask in motion.

III. Dataset

Two datasets have been used for experimenting the current method. Dataset 1 [16] consists of 1376 images in which 690 images with people wearing face masks and the rest686 images with people who do not wear face masks. Fig. 1 mostly contains front face pose with single face in the frame and with same type of mask having white color only.

Fig. 1. Samples from Dataset 1 including faces without masks and with masks

Dataset 2 from Kaggle [17] consists of 853 images and its countenances are clarified either with a mask or without a mask. In fig. 2 some face collections are head turn, tilt and slant with multiple faces in the frame and different types of masks having different colors as well.

Fig. 2. Samples from Dataset 2 including faces without masks and with masks

IV. Incorporated Packages

A. TensorFlow

TensorFlow, an interface for expressing machine learning algorithms, is utilized for implementing ML systems into fabrication over a bunch of areas of computer science, including sentiment analysis, voice recognition, geographic information extraction, computer vision, text summarization, information retrieval, computational drug discovery and flaw detection to pursue research [18]. In the proposed model, the whole Sequential CNN architecture (consists of several layers) uses TensorFlow at backend. It is also used to reshape the data (image) in the data processing.

B. Keras

Keras gives fundamental reflections and building units for creation and transportation of ML arrangements with high iteration velocity. It takes full advantage of the scalability and cross-platform capabilities of TensorFlow. The core data structures of Keras are layers and models [19]. All the layers used in the CNN model are implemented using Keras. Along with the conversion of the class vector to the binary class matrix in data processing, it helps to compile the overallmodel.

C. OpenCV

OpenCV (Open Source Computer Vision Library), an open-source computer vision and ML software library, is utilized to differentiate and recognize faces, recognize objects, group movements in recordings, trace progressive modules, follow eye gesture, track camera actions, expel red eyes from pictures taken utilizing flash, find comparative pictures from an image database, perceive landscape and set up markers to overlay it with increased reality and so forth [20]. The proposed method makes use of these features of OpenCV in resizing and color conversion of data images.

V. The Proposed Method

The proposed method consists of a cascade classifier and a pre-trained CNN which contains two 2D convolution layers connected to layers of dense neurons. The algorithm for face mask detection is as follows:

Algorithm 1: Face Mask Detection

Input: Dataset including faces with and without masks
Output: Categorized image depicting the presence of face mask

```
for each image in the dataset do
    Visualize the image in two categories and label them
    Convert the RGB image to Gray-scale image
    Resize the gray-scale image into 100 x 100
    Normalize the image and convert it into 4 dimensional array
end
for building the CNN model do
    Add a Convolution layer of 200 filters
    Add the second Convolution layer of 100 filters
    Insert a Flatten layer to the network classifier
    Add a Dense layer of 64 neurons
    Add the final Dense layer with 2 outputs for 2 categories
end
Split the data and train the model
```

A. Data Processing

Data preprocessing involves conversion of data froma given format to much more user friendly, desired and meaningful format. It can be in any form like tables, images, videos, graphs, etc. These organized information fit in with an information model or composition and captures relationship between different entities [6]. The proposed method deals with image and video data using Numpy and OpenCV.

a) Data Visualization: Data visualization is the pro- cess of transforming abstract data to meaningful representa- tions using knowledge communication and insight discovery through encodings. It is helpful to study a particular pattern in the dataset [7].

The total number of images in the dataset is visualized in both categories – 'with mask' and 'without mask'.

The statement *categories=os.listdir(data_path)* categorizes the list of directories in the specified data path. The variable *categories* now looks like: ['with mask', 'without mask']

Then to find the number of labels, we need to distinguish those categories using *labels=[i for i in range(len(categories))]*. It sets the labels as: [0, 1]

Now, each category is mapped to its respective label using *label_ dict=dict(zip(categories,labels))* which at first returns an iterator of tuples in the form of zip object where theitems in each passed iterator is paired together consequently. The mapped variable *label dict* looks like: {'with mask': 0, 'without mask': 1}

b) Conversion of RGB image to Gray image: Modern descriptor-based image recognition systems regularly work on grayscale images, without elaborating the method used to convert from color-to-grayscale. This is because the color- to-grayscale method is of little consequence when using robust descriptors. Introducing nonessential information could increase the size of training data required to achieve good performance. As grayscale rationalizes the algorithm and diminishes the computational requisites, it is utilized for extracting descriptors instead of working on color images instantaneously [8].

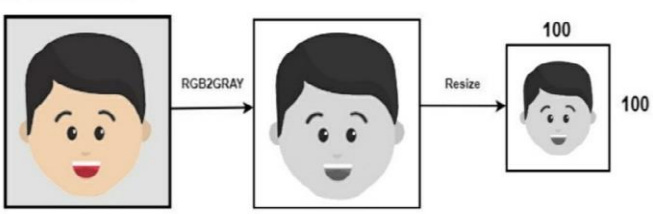

Fig. 3. Conversion of a RGB image to a Gray Scale image of 100x100 size

We use the function *cv2.cvtColor(input_image, flag)* for changing the color space. Here flag determines the type of conversion [9]. In this case, the flag *cv2.COLOR_BGR2GRAY*is used for gray conversion.

Deep CNNs require a fixed-size input image. Therefore we need a fixed common size for all the images in the dataset. Using *cv2.resize()* the gray scale image is resizedinto 100 x 100.

c) Image Reshaping: The input during relegation of an image is a three-dimensional tensor, where each channel has a prominent unique pixel. All the images must have identically tantamount size corresponding to 3D feature tensor. How- ever, neither images are customarily coextensive nor their corresponding feature tensors [10]. Most CNNs can only accept fine-tuned images. This engenders several problems throughout data collection and implementation of model. However, reconfiguring the input images before augmenting them into the network can help to surmount this constraint. [11].

The images are normalized to converge the pixel range between 0 and 1. Then they are converted to 4 di- mensional arrays using *data=np.reshape(data,(data.shape[0], img size,img size,1)*) where 1 indicates the Grayscale image. As, the final layer of the neural network has 2 outputs – with mask and without mask i.e. it has categorical representation, the data is converted to categorical labels.

B. Training of Model

a) Building the model using CNN architecture: CNN has become ascendant in miscellaneous computer vision tasks[12]. The current method makes use of Sequential CNN.

The First Convolution layer is followed by Rectified Linear Unit (ReLU) and MaxPooling layers. The Convolution layer learns from 200 filters. Kernel size is set to 3 x 3 which specifies the height and width of the 2D convolution window. As the model should be aware of the shape of the input expected, the first layer in the model needs to be provided with information about input shape. Following layers canperform instinctive shape reckoning [13]. In this case, *in-put shape* is specified as *data.shape[1:]* which returns the dimensions of the data array from index 1. Default paddingis "valid" where the spatial dimensions are sanctioned totruncate and the input volume is non-zero padded. The activation parameter to the Conv2D class is set as "relu".It represents an approximately linear function that possesses all the assets of linear models that can easily be optimized with gradient-descent methods. Considering the performance and generalization in deep learning, it is better compared to other activation functions [14]. Max Pooling is used to reducethe spatial dimensions of the output volume. Pool size is set to 3 x 3 and the resulting output has a shape (number of rows or columns) of: shape of output = (input shape_ - pool size _ _ + 1) / strides), where strides has default value (1,1) [15].

As shown in fig, 4, the second Convolution layer has 100 filters and Kernel size is set to 3 x 3. It is followed by ReLu and MaxPooling layers. To insert the data into CNN, the long vector of input is passed through a Flatten layer which transforms matrix of features into a vector that can be fed into a fully connected neural network classifier. To reduce

overfitting a Dropout layer with a 50% chance of setting inputs to zero is added to the model. Then a Dense layer of 64 neurons with a ReLu activation function is added. The final layer (Dense) with two outputs for two categories uses the Softmax activation function.

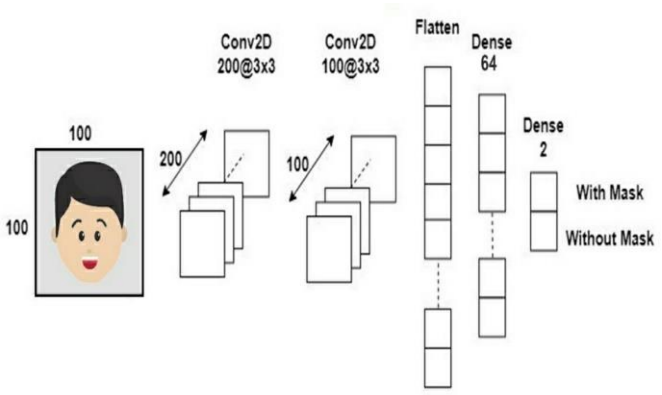

Fig. 4. Convolutional Neural Network architecture

The learning process needs to be configured first withthe compile method [13]. Here "adam" optimizer is used. *categorical_crossentropy* which is also known as multiclass log loss is used as a loss function (the objective that the model tries to minimize). As the problem is a classification problem, metrics is set to "accuracy".

b) Splitting the data and training the CNN model: After setting the blueprint to analyze the data, the model needs to be trained using a specific dataset and then to be tested against a different dataset. A proper model and optimized *train test split* help to produce accurate results while making a prediction. The test_size is set to 0.1 i.e. 90% data of the dataset undergoes training and the rest 10% goes for testing purposes. The validation loss is monitored using *ModelCheckpoint*. Next, the images in the training set and the test set are fitted to the Sequential model. Here, 20% of the training data is used as validation data. The model is trained for 20 epochs (iterations) which maintains a trade-off between accuracy and chances of overfitting. Fig. 5 depicts visual representation of the proposed model.

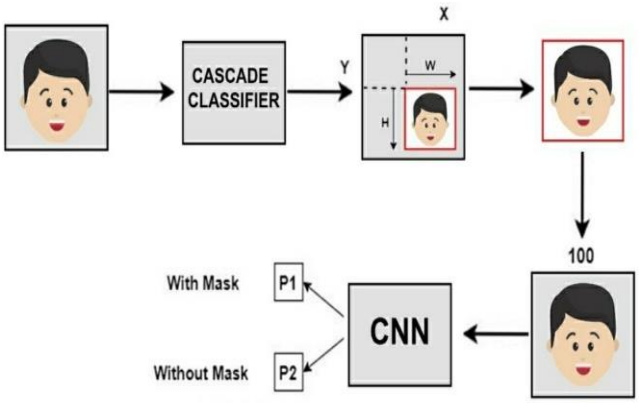

Fig. 5. Overview of the Model

VI. Result And Analysis

The model is trained, validated and tested upon two datasets. Corresponding to dataset 1, the method attains accuracy up to 95.77% (shown in fig. 7). Fig. 6 depicts how this optimized accuracy mitigates the cost of error. Dataset 2 is more versatile than dataset 1 as it has multiple faces in the frame and different types of masks having different colors as well. Therefore, the model attains an accuracy of 94.58% on dataset 2 as shown in Fig. 9. Fig. 8 depicts the contrast between training and validation loss corresponding to dataset 2. One of the main reasons behind achieving this accuracy lies in *MaxPooling*. It provides rudimentary translation invariance to the internal representation along with the reduction in the number of parameters the modelhas to learn. This sample-based discretization process down-samples the input representation consisting of image, by reducing its dimensionality. Number of neurons has the optimized value of 64 which is not too high. A much higher number of neurons and filters can lead to worse performance.The optimized filter values and pool size help to filter outthe main portion (face) of the image to detect the existenceof mask correctly without causing over-fitting.

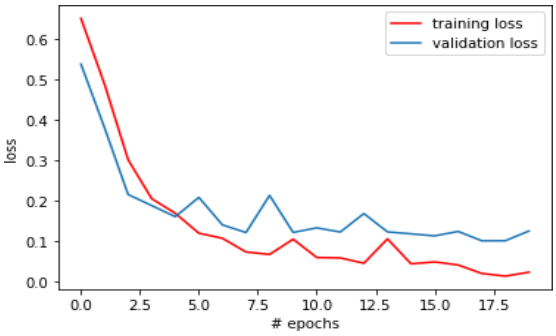

Fig. 6. # epochs vs loss corresponding to dataset 1

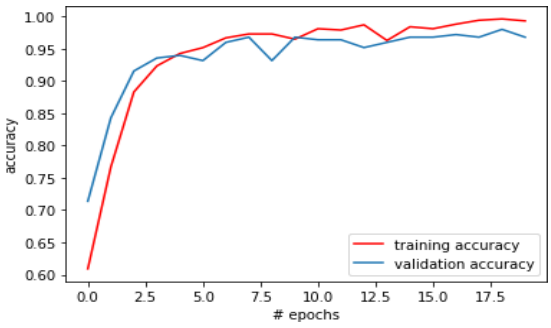

Fig. 7. # epochs vs accuracy corresponding to dataset 1

The system can efficiently detect partially occluded faces either with a mask or hair or hand. It considers the occlusion degree of four regions – nose, mouth, chin and eye to differentiate between annotated mask or face covered by hand. Therefore, a mask covering the face fully including nose and chin will only be treated as "with mask" by the model.

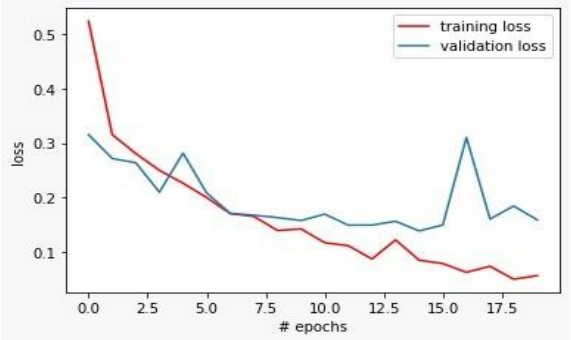

Fig. 8. # epochs vs loss corresponding to dataset 2

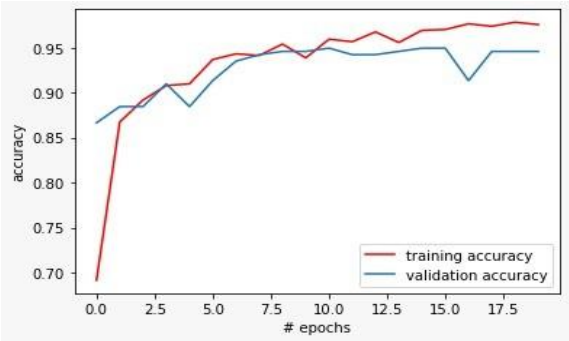

Fig 9. # epochs vs accuracy corresponding to dataset 2

The main challenges faced by the method mainly compriseof varying angles and lack of clarity. Indistinct moving faces in the video stream make it more difficult. However, followingthe trajectories of several frames of the video helps to createa better decision – “with mask” or “without mask”.

VII. CONCLUSIONS

In this paper, we briefly explained the motivation of the work at first. Then, we illustrated the learning and perfor- mance task of the model. Using basic ML tools and simplifiedtechniques the method has achieved reasonably high accuracy.It can be used for a variety of applications.Wearing a mask may be obligatory in the near future, considering the Covid-19crisis. Many public service providers will ask the customers towear masks correctly to avail of their services. The deployed model will contribute immensely to the public health care system. In future it can be extended to detect if a person is wearing the mask properly or not. The model can be further improved to detect if the mask is virus prone or not i.e. the type of the mask is surgical, N95 or not.

REFERENCES

[1] W.H.O., "Coronavirus disease 2019 (covid-19): situation report, 205".2020 "Coronavirus Disease 2019 (COVID-19) – Symptoms",Centers for Disease Control and Prevention, 2020. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html. 2020.

[2] "Coronavirus — Human Coronavirus Types — CDC", Cdc.gov, 2020. [Online]. Available: https://www.cdc.gov/coronavirus/types.html. 2020.

[3] W.H.O., "Advice on the use of masks in the context of COVID-19: interim guidance", 2020.

[4] M. Jiang, X. Fan and H. Yan, "RetinaMask: A Face Mask detector", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/2005.03950.2020.

[5] B. Suvarnamukhi and M. Seshashayee, "Big Data Concepts and Techniques in Data Processing", International Journal of Computer Sciences and Engineering, vol. 6, no. 10, pp. 712-714, 2018. Available:10.26438/ijcse/v6i10.712714.

[6] F. Hohman, M. Kahng, R. Pienta and D. H. Chau, "Visual Analyticsin Deep Learning: An Interrogative Survey for the Next Frontiers," in IEEE Transactions on Visualization and Computer Graphics, vol. 25, no. 8, pp. 2674-2693, 1 Aug. 2019, doi: 10.1109/TVCG.2018.2843369.

[7] C. Kanan and G. Cottrell, "Color-to-Grayscale: Does the Method Matter in Image Recognition?", PLoS ONE, vol. 7, no. 1, p. e29740, 2012. Available: 10.1371/journal.pone.0029740.

[8] Opencv-python-tutroals.readthedocs.io. 2020. Changing Colorspaces — Opencv-Python Tutorials 1 Documentation. [online] Availableat:https://opencv-python-tutroals.readthedocs.io/en/latest/py tutorials/ py imgproc/py colorspaces/py colorspaces.html. 2020.

[9] M. Hashemi, "Enlarging smaller images before inputting into convolu- tional neural network: zero-padding vs. interpolation", Journal of Big Data, vol. 6, no. 1, 2019. Available: 10.1186/s40537-019-0263-7 . 2020.

[10] S. Ghosh, N. Das and M. Nasipuri, "Reshaping inputs for con- volutional neural network: Some common and uncommon methods", Pattern Recognition, vol. 93, pp. 79-94, 2019. Available: 10.1016/j.patcog.2019.04.009.

[11] R. Yamashita, M. Nishio, R. Do and K. Togashi, "Convolutional neuralnetworks: an overview and application in radiology", Insights intoImaging, vol. 9, no. 4, pp. 611-629, 2018. Available: 10.1007/s13244- 018-0639-9.

[12] "Guide to the Sequential model - Keras Documentation", Faroit.com, 2020. [Online]. Available: https://faroit.com/keras-docs/1.0.1/getting- started/sequential-model-guide/. 2020.

[13] Nwankpa, C., Ijomah, W., Gachagan, A. and Marshall, S., 2020. Activation Functions: Comparison Of Trends In Practice And Research For Deep Learning. [online] arXiv.org. Available at: https://arxiv.org/abs/1811.03378. 2020.

[14] K. Team, "Keras documentation: MaxPooling2D layer", Keras.io, 2020. [Online]. Available: https://keras.io/api/layers/pooling layers/ max pooling2d/. 2020.

[15] "prajnasb/observations", GitHub, 2020. [Online]. Available: https://github.com/prajnasb/observations/tree/master/experiements/data. 2020.

[16] "Face Mask Detection", Kaggle.com, 2020. [Online]. Available: https://www.kaggle.com/andrewmvd/face-mask-detection. 2020.

[17] "TensorFlow White Papers", TensorFlow, 2020. [Online]. Available: https://www.tensorflow.org/about/bib. 2020.

[18] K. Team, "Keras documentation: About Keras", Keras.io, 2020. [On- line]. Available: https://keras.io/about. 2020.

[19] "OpenCV", Opencv.org, 2020. [Online]. Available: https://opencv.org/.2020.

[20] D. Meena and R. Sharan, "An approach to face detection and recognition," 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE), Jaipur, 2016, pp. 1-6, doi: 10.1109/ICRAIE.2016.7939462.

[21] S. Ge, J. Li, Q. Ye and Z. Luo, "Detecting Masked Faces in the Wild with LLE-CNNs," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 426-434, doi: 10.1109/CVPR.2017.53.

BIODATA

NAME	**:**	E. Rajeswari
FATHER NAME	**:**	E. Venkata Ramana
DATE OF BIRTH	**:**	23rd June 2000
NATIONALITY	**:**	INDIAN

CONTACT DETAILS

Contact No.	:	8309954879
Email	:	erajeswari128@gmail.com
Contact Address	:	D.No 27/127, Moolasagaram, Nandyal, Kurnool, 518501

BIODATA

NAME	**:**	G M Keerthana
FATHER NAME	**:**	G Munaswamy
DATE OF BIRTH	**:**	23rd September, 1999
NATIONALITY	**:**	INDIAN

CONTACT DETAILS

Contact No.	:	9515231970
Email	:	gmkeerthana246@gmail.com
Contact Address	:	D. No: 10-827/B, Seshapeeran street, Chittoor

BIODATA

NAME	**:**	G. Poojitha
FATHER NAME	**:**	G. Thippaiah
DATE OF BIRTH	**:**	22nd June, 2000
NATIONALITY	**:**	INDIAN

CONTACT DETAILS

Contact No.	**:**	7893196222
Email	**:**	gundigipoojitha@gmail.com
Contact Address	**:**	D No: 13-1-525-309, Lecturers colony, 1st Cross, Ananthapur

BIODATA

NAME	**:**	Junju Pranavi
FATHER NAME	**:**	Junju Srinivasulu
DATE OF BIRTH	**:**	12th September 1999
NATIONALITY	**:**	INDIAN

CONTACT DETAILS

Contact No.	:	9603157949
Email	:	pranavijunju@gmail.com
Contact Address	:	Door no: 3-4-20, Buddhi vari street, kovur- 524137, Nellore(dt)