# Using AI for Face Mask Detector Alarm System - A Research Study

*A project submitted in partial fulfilment of the Requirements for the award of the degree of*

**Bachelor of Technology**
**in**
**COMPUTER SCIENCE AND ENGINEERING**

Submitted by:
**Mr. Venkatarami Reddy**
Roll No.: 12011041

Supervised by:
**Dr. Mukesh Mann**
Assistant Professor

**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY**
**SONEPAT-131201, HARYANA, INDIA**

# ACKNOWLEDGEMENTS

In the present world of competition there is a race of existence in which those are having will to come forward succeed. Project is like a bridge between theoretical and practical working. With this willing I joined this particular project. First of all, I would like to thank the supreme power the Almighty God who is obviously the one has always guided me to work on the one has always guided me to work on the right path of life. Without his grace this project could not become a reality. Next to him are my parents, whom I am greatly indebted for me brought up with love and encouragement to this stage. I am feeling oblige in taking the opportunity to sincerely thanks to **Dr. M. N. Doja** (Director of IIIT Sonepat) and special thanks to my worthy teacher of Computer Science **Dr. Mukesh Mann**. Moreover, I am highly obliged in taking the opportunity to sincerely thanks to all the staff members of CSE department for their generous attitude and friendly behaviour. At last but not the least I am thankful to all my teachers and friends who have been always helping and encouraging me though out the year. I have no valuable words to express my thanks, but my heart is still full of the favours received from every person

**Venkatarami Reddy (12011041)**

# SELF DECLARATION

I hereby declare that work contained in the research titled "**Using AI for Face Mask Detector Alarm System - A Research Study**" is original. I have followed the standards of project ethics to the best of my abilities. I have acknowledged all sources of information which I have used in the project.

Name: Venkatarami Reddy
Roll No.: 12011041

Department of Computer Science and Engineering,
Indian Institute of Information Technology,
Sonepat-131201, Haryana, India.

# CERTIFICATE

This is to certify that **Mr. Venkatarami Reddy** has worked on the research entitled "**Using AI for Face Mask Detector Alarm System - A Research Study**" under my supervision and guidance.

The contents of the project, being submitted to **Department of Computer Science and Engineering, IIIT, Sonepat,** for the award of the degree of **B. Tech in Computer Science and Engineering,** are original and have been carried out by the candidate himself. This project has not been submitted in full or part for the award of any other degree or diploma to this or any other university.

Dr. Mukesh Mann
Supervisor

Department of Compute Science and Engineering,
Indian Institute of Information Technology ,
Sonepat-131201, Haryana, India.

# ABSTRACT

Facial recognition, as a biometric system, is a crucial tool for the identification procedures. When using facial recognition, an individual’s identity is identified using their unique facial features. Biometric authentication system helps in identifying individuals using their physiological and behavioral features. Physiological biometrics utilize human features such as faces, irises, and fingerprints. In contrast, behavioral biometric rely on features that humans do, such as voice and handwritings. Facial recognition has been widely used for security and other law enforcement purposes. However, since **COVID-19** pandemic, many people around the world had to wear face masks. This thesis introduces a neural network system, which can be trained to identify people’s facial features while half of their faces are covered by face masks. The Convolution Neural Network (CNN) model using transfer learning technique has achieved remarkable accuracy even the original dataset is very limited. One large face mask detection dataset was first used to train the model, while the original much smaller Face Mask Detector dataset was used to adapt and fine-tune this model that was previously generated. During the training and testing phases, network structures, and various parameters were adjusted to achieve the best accuracy results for the actual small dataset.

*Keywords:* Biometrics, Facial Recognition, Face Mask Detection, Convolution Neural Network, Transfer learning.

Name of the student: **Venkatarami Reddy**

Roll No.: 12011041

Degree for which submitted: **B. Tech,** Department of **Computer Science and Engineering, IIIT Sonepat**

Project Title: **Using AI for Face Mask Detector Alarm System - A Research Study**

Name of the project Supervisor: **Dr. Mukesh Mann**

Month and Year of report submission: **October 2021**

| | |
|---|---|
| Acknowledgments | i |
| Self Declaration | ii |
| Certificate | iii |
| Abstract | iv |
| List of Abbreviations | v |
| List of Figures | vi |
| List of Tables and Equations | vii |
| Table of Contents | viii |

# List of Abbreviations

| | |
|---|---|
| **CNN** | **Convolutional Neural Network** |
| **FMD** | **Face Mask Detector** |
| **AI** | **Artificial Intelligence** |
| **ML** | **Machine Learning** |
| **FR** | **Face Recognition** |
| **DL** | **Deep Learning** |

# List of Figures

| ***Figure's No.*** | **Name of Figures** | ***Page Nos.*** |
|---|---|---|
| ***Figure 1*** | *Design of Face Mask Detector* | ***06*** |
| ***Figure 2*** | *People Wearing Face Masks* | ***07*** |
| ***Figure 3*** | *People not Wearing Face Masks* | ***07*** |
| ***Figure 4*** | *PYTHON Language* | ***08*** |
| ***Figure 5*** | *Jupyter Notebook* | ***08*** |
| ***Figure 6*** | *GitHub* | ***09*** |
| ***Figure 7*** | *TensorFlow* | ***09*** |
| ***Figure 8*** | *Keras* | ***10*** |
| ***Figure 9*** | *Matplotlib* | ***10*** |
| ***Figure 10*** | *Numpy* | ***10*** |
| ***Figure 11*** | *OpenCV* | ***11*** |
| ***Figure 12*** | *Fine-Tuning Stepa* | ***13*** |
| ***Figure 13*** | *Loading the Datasets* | ***14*** |
| ***Figure 14*** | *Training and Compiling* | ***14*** |
| ***Figure 15*** | *Accuracy Results* | ***16*** |
| ***Figure 16*** | *Plot of Results During Training* | ***16*** |
| ***Figure 17*** | *Testing the Trained Model* | ***17*** |
| ***Figure 18*** | *Detecting the Face Mask Using OpenCV* | ***18*** |
| ***Figure 19*** | *Mask Detector with no Mask* | ***19*** |
| ***Figure 20*** | *Mask Detector with Mask* | ***19*** |
| ***Figure 21*** | *Features of Face Mask Detector* | ***20-22*** |

## List of Tables

| ***Table No.*** | **Name of Figures** | ***Page Nos.*** |
|---|---|---|
| ***Table 1*** | *Epoch Number and Loss Percentage Results* | ***15*** |

## List of Equations

| ***Equation No.*** | **Name of Figures** | ***Page Nos.*** |
|---|---|---|
| ***Equation 1*** | *Supervised Learning Algorithm Equation* | ***12*** |

| TABLE OF CONTENTS | | | Pg. No |
|---|---|---|---|
| CHAPTER 01 | **Introduction** | | **01** |
| | **1.1** | Introduction | **01** |
| | **1.2** | Problem Outline | **02** |
| | **1.2.1** | Purpose of Project | **02** |
| | **1.2.2** | Project Description | **03** |
| | **1.2.3** | Project Objectives | **04** |
| | **1.3** | Summary | **04** |
| CHAPTER 02 | **Literature Review** | | **05** |
| | **2.1** | Face Mask Detector | **05** |
| | **2.2** | Face Detection Techniques | **05** |
| | **2.3** | Multi-Stage CNN Architecture for Face Mask Detection | **05** |
| | **2.4** | Real Time Face Mask Detection with Alarm System using DL | **05** |
| CHAPTER 03 | **Methodology** | | **06** |
| | **3.1** | System Design | **06** |
| | **3.2** | Datasets | **06** |
| | **3.3** | Development Platform Tools | **08** |
| | **3.4** | Libraries Used | **09** |
| | **3.5** | Summary | **11** |

| | | | |
|---|---|---|---|
| CHAPTER 04 | **Training and Testing** | | **12** |
| | **4.1** | Data PreProcessing | **12** |
| | **4.2** | Supervised Learning | **12** |
| | **4.3** | Transfer Learning | **13** |
| | **4.4** | Training and Fine Tuning | **13** |
| | **4.5** | Running Epochs and Accuracy | **15** |
| | **4.6** | Testing | **17** |
| CHAPTER 05 | **Results And Discussions** | | **19** |
| | **5.1** | Result on Testing | **19** |
| | **5.2** | Some Features of Face Mask Detector | **20** |
| | **5.3** | Multiple Mask Detection | **22** |
| | **5.4** | Summary | **23** |
| CHAPTER 06 | **Conclusion and Future Scope** | | **24** |
| | **6.1** | Limitations | **24** |
| | **6.2** | Scope of the Project | **25** |
| | **6.3** | Future Goals | **25** |
| | **6.4** | Conclusion | **27** |
| **References** | | | **26-27** |

# CHAPTER 01

# Introduction

## 1.1 Introduction

The trend of wearing face masks in public is rising due to the COVID-19 coronavirus epidemic all over the world. Before Covid-19, People used to wear masks to cover their health from air pollution. While other people are tone-conscious about their aesthetics, they hide their passions in the public to hide their faces. Further than five million cases were infected by COVID19 in lower than 6 months across 188 countries. The contagion spreads through close contact and in crowded and overcrowded areas. We can attack and predict new conditions by the help of new Technologies analogous as artificial intelligence, Iot, Big data, and Machine knowledge. In order to more understand infection rates might be drop through our fashion.

People are forced by laws to wear face masks intimately in numerous countries. These rules and laws were developed as an action to the exponential growth in cases and deaths in numerous areas. Still, the system of covering large groups of individualities is getting harder intimately areas. So we'll produce a robotization process for detecting the faces.

The distinctive face features of each existent in a face mask condition will give us a chance in identifiying the person fluently and in an automated way. Recent rapid-fire- fire- fire technological inventions in deep knowledge and computer vision have presented openings for development in numerous fields. As the main element of deep knowledge styles, deep neural networks (DNNs) have demonstrated superior performance in numerous fields, including object discovery, image type, image segmentation, and distancing discovery. One primary model of DNNs is Mobile Net, Mobile Net is a CNN armature model for Image Bracket and Mobile Vision. There are other models also but Mobile Net takes less computation power to apply transfer knowledge to. This makes it a perfect fit for Mobile bias, bedded systems and computers without GPU or low computational effectiveness with compromising significantly with the delicacyoftheresults.it is also swish suited for web cybersurfers as cybersurfers have limitation over calculation, graphic processing and storage.

### Mobile Net Architecture

- Mobile Nets for mobile and embedded vision applications is proposed, which are based on a streamlined architecture that uses depth wise separable convolutions to build light weight deep neural networks.
- Two simple global hyper-parameters that efficiently trade off between latency and accuracy are introduced.

The core layer of Mobile Net is depth wise separable filters, named as Depth wise Separable Convolution. The network structure is another factor to boost the performance. Finally, the width and resolution can be tuned to trade off between latency and accuracy.

So Having High Accuracy and validation accuracy we will use Mobile Net for training our model later in the section we will also compare accuracy, validation accuracy, loss, validation loss in CNN and Mobile Net.

## 1.2 Problem Outline

Coronavirus (COVID-19) is an emerging respiratory infectious disease caused by severe acute respiratory syndromecoronavirus 2 (SARS-CoV2). At present, COVID-19 has rapidly spread to the majority of countries worldwideaffecting more than 14.9 million individuals, and has caused 618,017 deaths, reported by the World HealthOrganization (WHO) on 23 July 2020. **To avoid globaltragedy, a practical and straightforward approach to precluding the spread of the contagion is urgently needed worldwide**. Former studies have proved the fact tha facemask-wearing is a very effective method in preventing the spread of respiratory viruses. For instance, theefficiencies of N95 and surgical masks in preventing the transmission of SARS are 91% and 68%, respectively Facemask- wearing can intrude airborne contagions and patches effectively, similar that these pathogens can no enter the respiratory system of another person. As a non- medicinal intervention, facemask- wearing is an effective and cheap system to decrease the rate of mortality and morbidity from respiratory virus andinfections. Since the outbreak of COVID-19, facemasks have been routinely used by the general public to reduce exposure to airborne pathogens in numerous countries. In addition to cases suspected of factual infection with COVID-19 being needed to wear facemasks for the forestallment of contagion spreading, healthy persons also need to wear facemasks to cover themselves from infection. Facemasks, when worn properly, will effectively prevent the forward instigation of micro-particles of the virus when expelled from a cough or sneeze, precluding complaint transmission. Still, the effectiveness of facemasks in containing the spread of airborne conditions in the general public has been lowered, substantially due to indecorous wearing. Thus, it's necessary to develop an automatic discovery approach for facemask- wearing conditions, which can contribute to particular protection and public epidemic forestallment. The distinctive facial characteristics of a face of each person and thealgoritms applied for detecting facemask- wearing conditions give us a chance for automatic identification.

### 1.2.1 Purpose of Project

COVID-19 has disintegrated normal life each over the Earth and disturbed humans' life drastically. Also, it has taken the lives of numerous people and numerous are floundering with its symptoms. This is a respiratory contagion and spread through respiration of COVID-19 Contagion through the nose while inhalation in the polluted terrain. Former studies have plant that facemask- wearing is precious in precluding the spread of respiratory contagions. For case, the edge of N95 and surgical masks in facemask- wearing conditions, which can contribute to particular protection and public epidemic forestallment.

Face mask Discovery refers to descry whether a person is wearing a mask or not. In fact, the matter is rear engineering of face discovery where the face is detected using different machine learning algorithms for the aim of security, authentication and surveillance. Face discovery is a crucial area in the field of Computer Vision and Pattern

Recognition. A significant body of exploration has contributed sophisticated to algorithms for face discovery in history. The primary exploration on face discovery was done in 2001 using the design of craft point and operation of traditional machine learning algorithms to train effective classifiers for discovery and recognition. The problems encountered with this approach include high complexity in point design and low discovery delicacy. In recent times, face discovery styles grounded on deep convolutional neural networks (CNN) have been extensively developed to ameliorate discovery performance.

This model can be installed in hospitals, shopping promenades, thoroughfares, roads, requests, seminaries, and numerous further public places to insure that everyone is wearing a mask and the one not wearing can be fluently linked using the model. But, as the days passed negligence in public has been increased, So we need find a new approach in precluding this type of geste in public and to help the spreading of COVID-19. So, it was necessary to develop an automatic discovery approach for face mask wearing conditions, which can contribute to particular protection and public epidemic forestallment.

### 1.2.2 Project Description

After the rout of the worldwide epidemic COVID-19, there arises a severe need for protection mechanisms, face masks being the primary bone. The introductory end of the design is to descry the presence of a face mask on mortal faces on live streamingvideo as well as on images.

The design is erected using OpenCV, Python library to apply machine literacy and deep literacy principles in Jupyter Notebook Editor. Keras Algorithm is used to make this deep literacy model to descry the face mask sensor. The model can work on real- time CCTV footage, which could be installed in crowded areas as well as hospitals, public places, promenades,etc. the model can prognosticate up to certain delicacy whether the person is wearing a mask or not.

The rapid worldwide spread of Coronavirus Disease 2019 (COVID-19) has resulted ina global pandemic. Correct facemask wearing is valuable for infectious disease control, but the effectiveness of facemasks has been diminished, mostly due to improper wearing. However, there have not been any published reports on the automatic identification of facemask-wearing conditions. In this study, we develop a new facemask-wearing condition identification method by combining image super- resolution and classification networks (SRCNet), which quantifies a three-category classification problem based on unconstrained 2D facial images. The proposed algorithm contains four main steps: Image pre-processing, facial detection and cropping, image super-resolution, and facemask-wearing condition identification.

### 1.2.3 Project Objectives

1. **Live Face Mask Detection:** To identify the person on a live video stream wearinga face mask with the help of computer vision and a deep learning algorithm.

2 **Max Accuracy:** To achieve maximum accuracy on detection, we need to have a data set which has images of all races and different features of men, women and children.

## 1.3 Summary

In this chapter, A brief introduction to Face Mask Detector is given. As the covid-19 virus came People don’t take safety measures like wearing masks To monitor that people are following this basic safety principle, a strategy should be developed. Aface mask detector system can be implemented to check this.

After that, the problem outline is discussed with the purpose of the project, its goals, and its description which tells about why we selected this project.
Project Objectives are discussed afterward which are Live Face Mask Detection and achieve Max Accuracy by taking large data set as much as possible

# CHAPTER 02

# Literature Review

## 2.1 Face Mask Detector

Single Shot Detector design is employed for object detection functions. during this system face mask detector may be deployed in several areas like searching malls, airports and different significant traffic places to observe the general public and to avoid the unfold of the unwellness by checking UN agency is following basic rules and UN agency isn't. It takes excessive time forknowledge loading in Google Colab Notebook. It failed to permit the access of digital camera that exhibit a hurdle in testing pictures and video stream. we've modelled a facemask detector mistreatment Deep learning. we have a tendency to ar processed system computationally economical mistreatment MobileNetV2 that makes it easier to Extract the information sets. we have a tendencyto use CNN design for higher performance. we are able to fix it in any quite cameras.

## 2.2 Face Detection Techniques

Artificial Human beings have not tremendous ability to identify different faces than machines, so automatic face detection system plays an important role in face recognition,headpose estimation etc.It has some problems like face occlusion,andnon uniform illumination. We use Neural Network to detect face in the Live video stream. Tensor flow is also used in this system . In existing they use Adaboost algorithm, we are using mob net CNN Architecture model in our proposed system. We will overcome all these problems in this paper.

## 2.3 Multi-Stage CNN Architecture for Face Mask Detection

This system consists of a dual-stage (CNN)architecture capable of detecting masked and unmasked faces and can be integrated with pre-installed CCTV cameras.This will help track safety violations, promote the use of face masks and ensure a safe working environment. Datasets were collected from public domain along with some data scraped from the internet. They use only pretrained datasets for detection. We can use any cameras to detect faces. It will be very useful for society and for peoples to prevent them from virus transmission. Here we use live video detection using open cv(python library)

## 2.4 Real Time Face Mask Recognition with System Using Deep Learning

This method provides a definite and accurate and very fast results for facemask detection. Raspberrypi which is primarily based real time mask recognition that captures the facial image. this method uses the field options of VGG-16 because the foundation network for face recognition.Deep learningtechniques area unit applied to construct a classifier that may collect image of an individual carryinga mask andno masks. Our projected study area unit uses the field features of CNN because the foundation network for face detection .It shows accuracy in detective work person carrying a mask not carrying a mask.

# CHAPTER 03

# Methodology

## 3.1 System Design

The major requirement for implementing this project using python programming language along with Deep learning, Machine learning, Computer vision and also with python libraries. The architecture consists of Mobile Net as the backbone, it can be used for high and low computation scenarios. We are using CNN Algorithm in our proposed system.

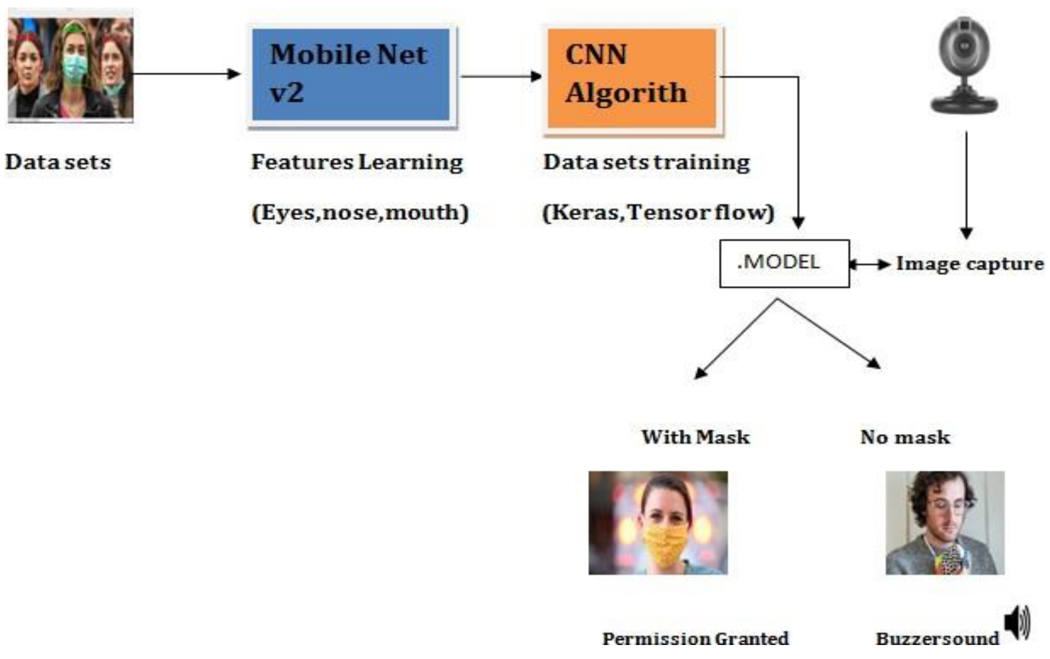


**Fig 1: Design of Face Mask Detector**

## 3.2 Datasets

This dataset, which is available on Kaggle, has three different classes which include People who wear masks, people who do not wear masks. The dataset consists of 3542 images of people wear masks, and 3846 images of people who do not wear a mask at all. The total number of images for this dataset comes up to 7388 face images. Figure 9 shows sample images of people who wear FM Figure 10 shows sample images of people who do not wear face masks. Figure 11 shows sample images of people who wear face masks but in an incorrect manner. In this dataset, people who wear face masks were labelled with class 0, while people who do not wear face masks were labelled with class 1 and people who wear face masks incorrectly were labelled with class 2. In this paper, this dataset was used for applying the transferlearning method. Figure 12 displays a distribution of the total number of different classes of the dataset. Likewise, figures 13, 14, and 15 present the distribution of each class image size

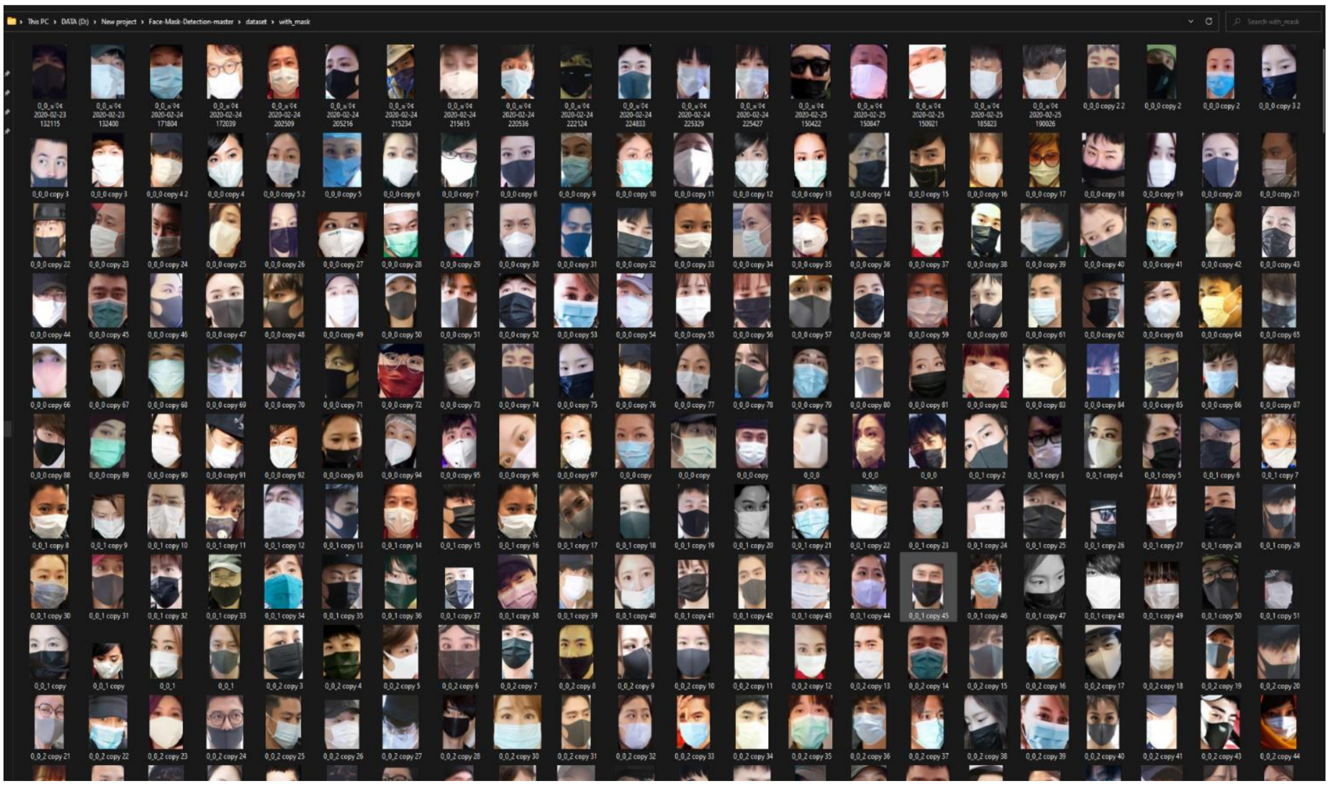

This PC › DATA (D:) › New project › Face-Mask-Detection-master › dataset › with_mask

Fig 2: People wearing Face Masks

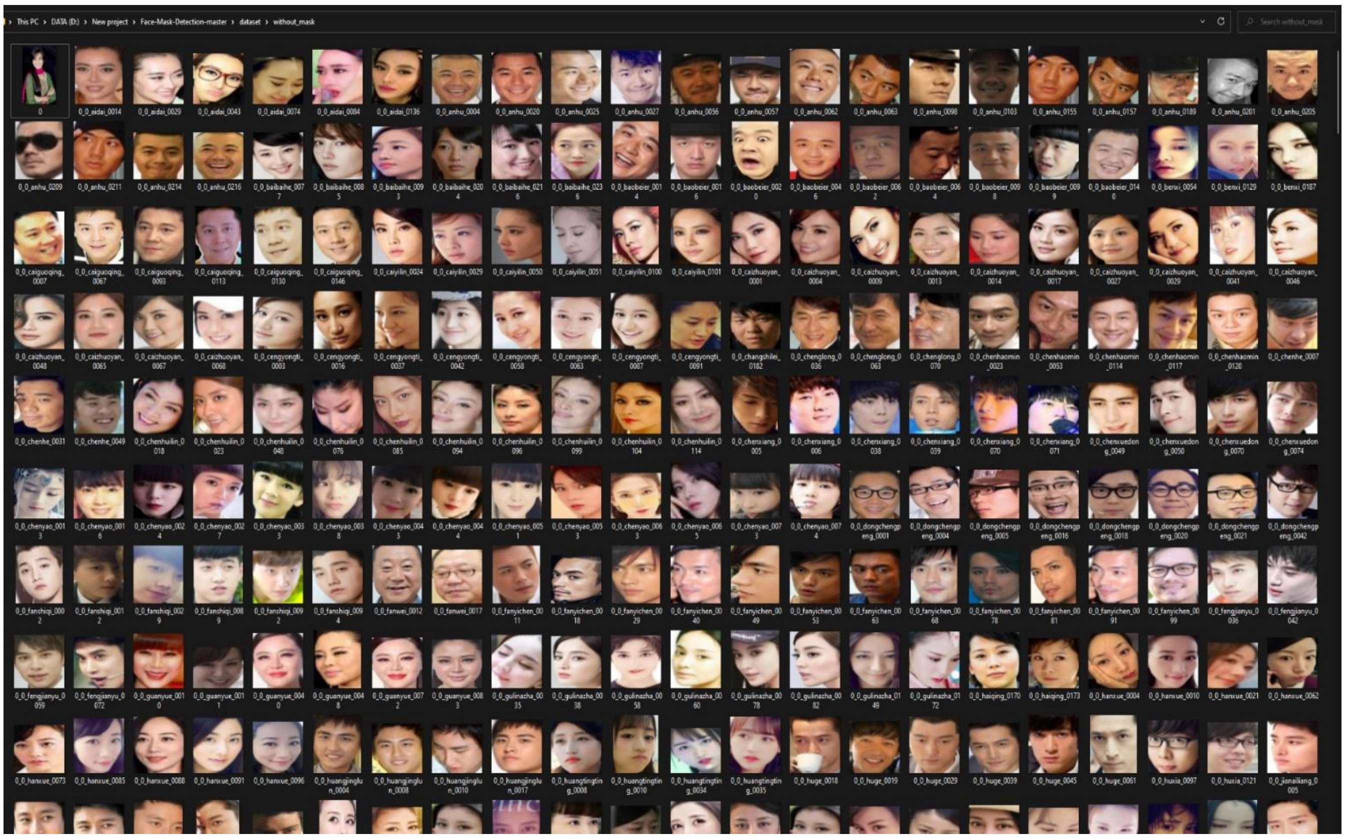

This PC › DATA (D:) › New project › Face-Mask-Detection-master › dataset › without_mask

Fig 3: People not wearing Face Masks

## 3.3 Development Platform Tools

The following development tools are needed for the development of this project

1) **PYTHON Language**

**Fig 4: PYTHON Language**

Python is an interpreted, high-level, and general-purpose programming language.Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

2) **Jupyter Notebook**

**Fig 5: Jupyter Notebook**

JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing,and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.

3) **GITHUB**

**Fig 6: GitHub**

GitHub, Inc. is a subsidiary of Microsoft which provides hosting for software development and version control using Git. It offers the distributed version control and source code management (SCM) functionality of Git, plus its own features. It provides access control and several collaboration features such as bugtracking, feature requests, task management, continuous integration, and wikis for every project.[3] Headquartered in California, it has been a subsidiaryof Microsoft since 2018.

## 3.4 Libraries Used

- **Tensorflow:** TensorFlow provides a collection of workflows to develop and train models using Python or JavaScript, and to easily deploy in the cloud, on-prem, in the browser, or on-device no matter what language you use.

**Fig 7: TensorFlow**

- **Keras:** Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides.

Fig 8: Keras

- **Matplotlib:** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

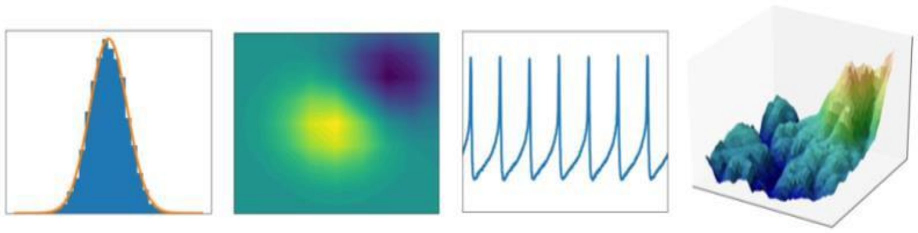


Fig 9: Matplotlib

Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web applicationservers, and various graphical user interface toolkits.

- **Numpy:** NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an opensource project and you can use it freely. NumPy stands for Numerical Python.

Fig 10: Numpy

- **Open-cv:** OpenCV was started at Intel in 1999 by Gary Bradsky, and the first release came out in 2000. Vadim Pisarevsky joined Gary Bradsky to manage Intel's Russian software OpenCV team. In 2005, OpenCV was used on Stanley, the vehicle that won the 2005 DARPA Grand Challenge. Later, its active development continued under the support of Willow Garage with

Gary Bradsky and Vadim Pisarevsky leading the project.OpenCV now supports a multitude of algorithms related to ComputerVision and Machine Learning and is expanding day by day.

OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.

OpenCV-Python is the Python API for OpenCV, combining the bestqualities of the OpenCV C++ API and the Python language

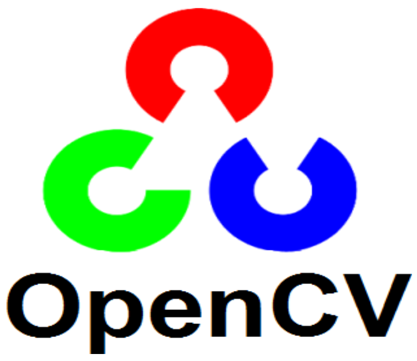


**Fig 11: OpenCV**

## 3.5 Summary

This chapter talks about the design part of the Face Mask Detector. And also about the Datasets to train the model.

This chapter also talks about the system architecture of FMD in which a brief discussion is there for the total design of the FMD.

This chapter also talks about the development tools we need to improve and perform the FMD also talks about the PYTHON libraries used in to train the model and detect the face mask.

# CHAPTER 04

# Training and Testing

## 4.1 Data Preprocessing

In Machine Learning, Performing data pre-processing is a very significant step that helps in Improving the quality of data to helpl the extraction of important understandings from the data. In data pre-processing, the data is getting cleaned and organized to become proper for building and training a model.

In this section we will convert all the images in the categories with and without mask into arrays to create a deep-learning model. Firstly with the help of keras.preprocessing.image we will load the images. Next we will set the height and width of the images (in our case 224 x 224px). Now with the help of img_to_array function we will convert the picture data into arrays. Here we create 2 arrays namely, data[ ] and label[ ]. In data[ ] we copy data of all the images and in label [ ] we add according to the category, with mask or without mask. With the help of binalizer() function from sklearn.model_selection we will convert the characters in label[ ] to binary values. Now the data[ ] and label[ ] should be converted into NumPy arrays as deep learning models only work with arrays. Next we will apply train_split_test function from sklearn.model_selection to train and test the images from data[ ] and label[ ]. We can change the percentage of images from the dataset to be used for training and testing.(In our case it is 80% and 20%)

## 4.2 Supervised Learning

Since supervised learning is that the best and most typical technique of machine learning nowadays, during this paper, supervised learning techniques were accustomed succeed the most effective performance results. supervised machine learning algorithms learn by example. The term supervised learning comes from the conception of coaching the dataset. The coaching dataset perpetually consists of input pictures, that also are perpetually combined with their correct outputs. throughout the coaching method, any patterns within the knowledge that relate to the chosen outputs are going to be examined by mistreatment the supervised learning formula. when coaching, a supervised learning formula can absorb new unidentified inputs and can be ready to determine the new inputs with their correct labels supported the preceding coaching knowledge. the most purpose of a supervised learning model is to presume the right label for recently conferred input file. A supervised learning formula as an easy equation that may be seen in Equation one. during this equation, Y is that the foretold output and x is that the input worth. This operate is employed in the main to attach input options to their foretold output that is formed by the model throughout the coaching method.

$$Y = f(x)$$

*Equation 1: Supervised Learning Algorithm Equation*

## 4.3 Transfer Learning

Transfer learning is one among the machine learning techniques that works by reusing a pre-trained model that was originally engineered for one more dataset. Transfer learning then reuses that model as a start line for a replacement sometimes smaller dataset. Transfer learning is applied for dashing up the model's coaching time and solve the matter of deficient information. Usually, whereas building the CNN models, tons of your time is spent on building and connecting convolutional layers. Transfer learning works by exploitation the previous neural network model todetermine edges within the earlier layers, structures within the middle layer, and high-level optionswithin the later layers. the first and middle layers are sometimes getting used, however the last layersar solely retrained. Loading the new information set and applying data resizing, shuffling and grayscale conversion, fine-tuning was applied to boost the accuracy.

## 4.4 Training and Fine-Tuning

During the transfer learning process, fine-tuning was applied on the new dataset with a very low learning rate. During this step, the previously constructed CNN model has been used except the output layer. Thus, we need to adjust the output layer to the number of target dataset classes for the new model. For example, in the approach, the initial pre-trained CNN model, had three classes. Also during transfer learning, the learning rate was lowered to 0.0001. Figure 12 presents the fine-tuning procedures.

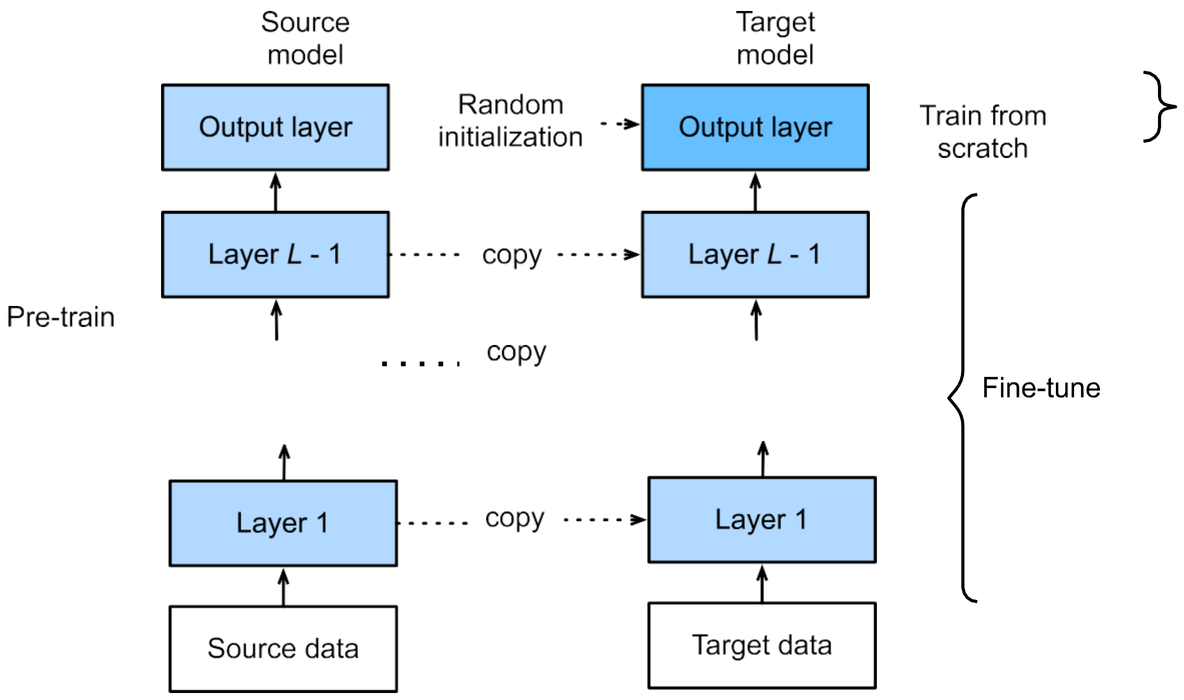
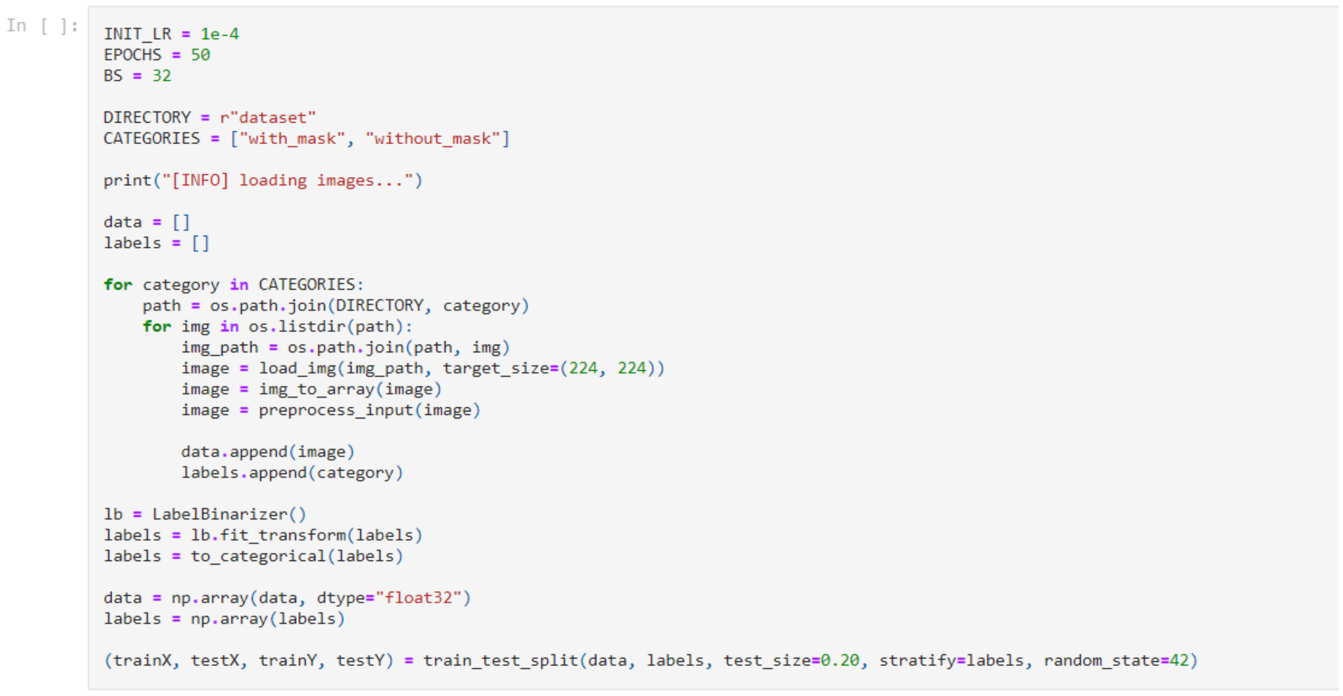


*Figure 12: Fine-tuning Steps*

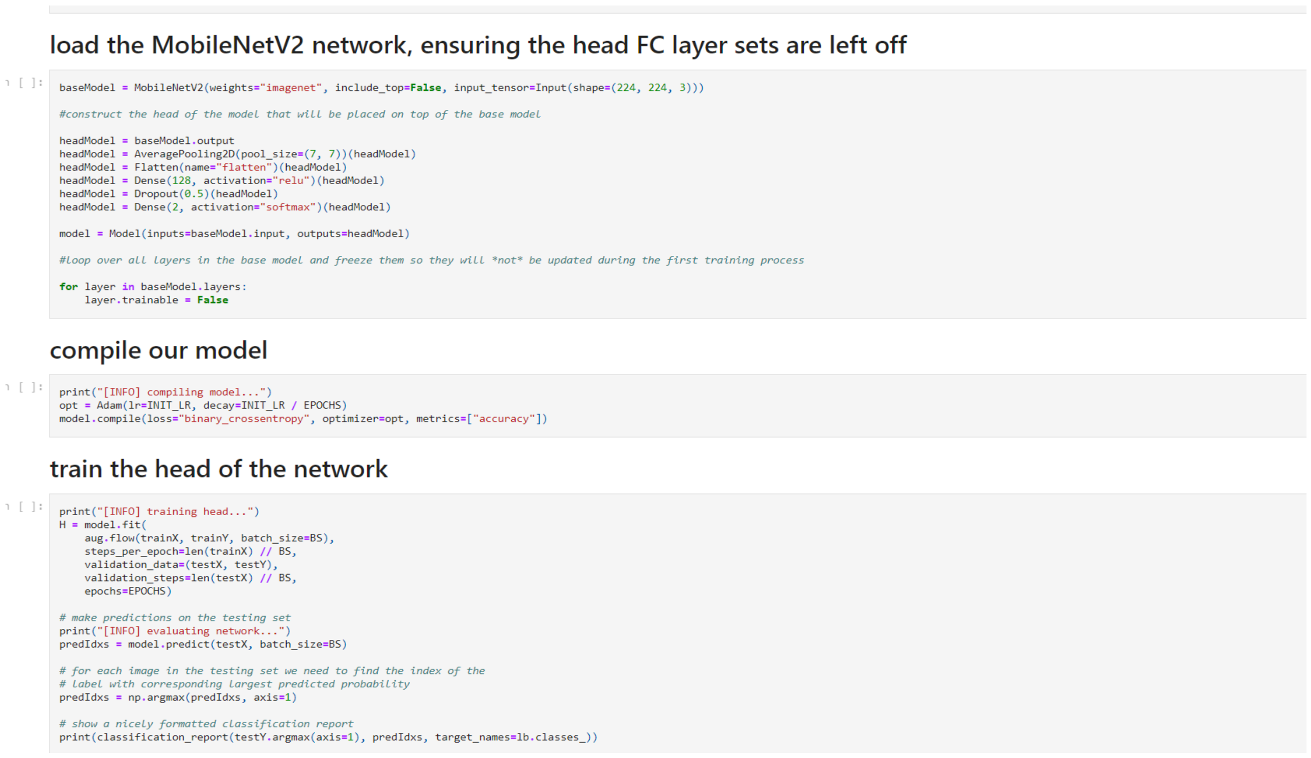
After completing the fine-tuning step, the training step took place using face mask detector dataset. During the training step, a batch size of 32 was applied to help the model faster and improve the accuracy performance.



In [ ]:

```
INIT_LR = 1e-4
EPOCHS = 50
BS = 32

DIRECTORY = r"dataset"
CATEGORIES = ["with_mask", "without_mask"]

print("[INFO] loading images...")

data = []
labels = []

for category in CATEGORIES:
    path = os.path.join(DIRECTORY, category)
    for img in os.listdir(path):
        img_path = os.path.join(path, img)
        image = load_img(img_path, target_size=(224, 224))
        image = img_to_array(image)
        image = preprocess_input(image)

        data.append(image)
        labels.append(category)

lb = LabelBinarizer()
labels = lb.fit_transform(labels)
labels = to_categorical(labels)

data = np.array(data, dtype="float32")
labels = np.array(labels)

(trainX, testX, trainY, testY) = train_test_split(data, labels, test_size=0.20, stratify=labels, random_state=42)
```

*Figure 13: Loading The datasets*



load the MobileNetV2 network, ensuring the head FC layer sets are left off

ı [ ]:

```
baseModel = MobileNetV2(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

#construct the head of the model that will be placed on top of the base model

headModel = baseModel.output
headModel = AveragePooling2D(pool_size=(7, 7))(headModel)
headModel = Flatten(name="flatten")(headModel)
headModel = Dense(128, activation="relu")(headModel)
headModel = Dropout(0.5)(headModel)
headModel = Dense(2, activation="softmax")(headModel)

model = Model(inputs=baseModel.input, outputs=headModel)

#loop over all layers in the base model and freeze them so they will *not* be updated during the first training process

for layer in baseModel.layers:
    layer.trainable = False
```

compile our model

ı [ ]:

```
print("[INFO] compiling model...")
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
model.compile(loss="binary_crossentropy", optimizer=opt, metrics=["accuracy"])
```

train the head of the network

ı [ ]:

```
print("[INFO] training head...")
H = model.fit(
    aug.flow(trainX, trainY, batch_size=BS),
    steps_per_epoch=len(trainX) // BS,
    validation_data=(testX, testY),
    validation_steps=len(testX) // BS,
    epochs=EPOCHS)

# make predictions on the testing set
print("[INFO] evaluating network...")
predIdxs = model.predict(testX, batch_size=BS)

# for each image in the testing set we need to find the index of the
# label with corresponding largest predicted probability
predIdxs = np.argmax(predIdxs, axis=1)

# show a nicely formatted classification report
print(classification_report(testY.argmax(axis=1), predIdxs, target_names=lb.classes_))
```

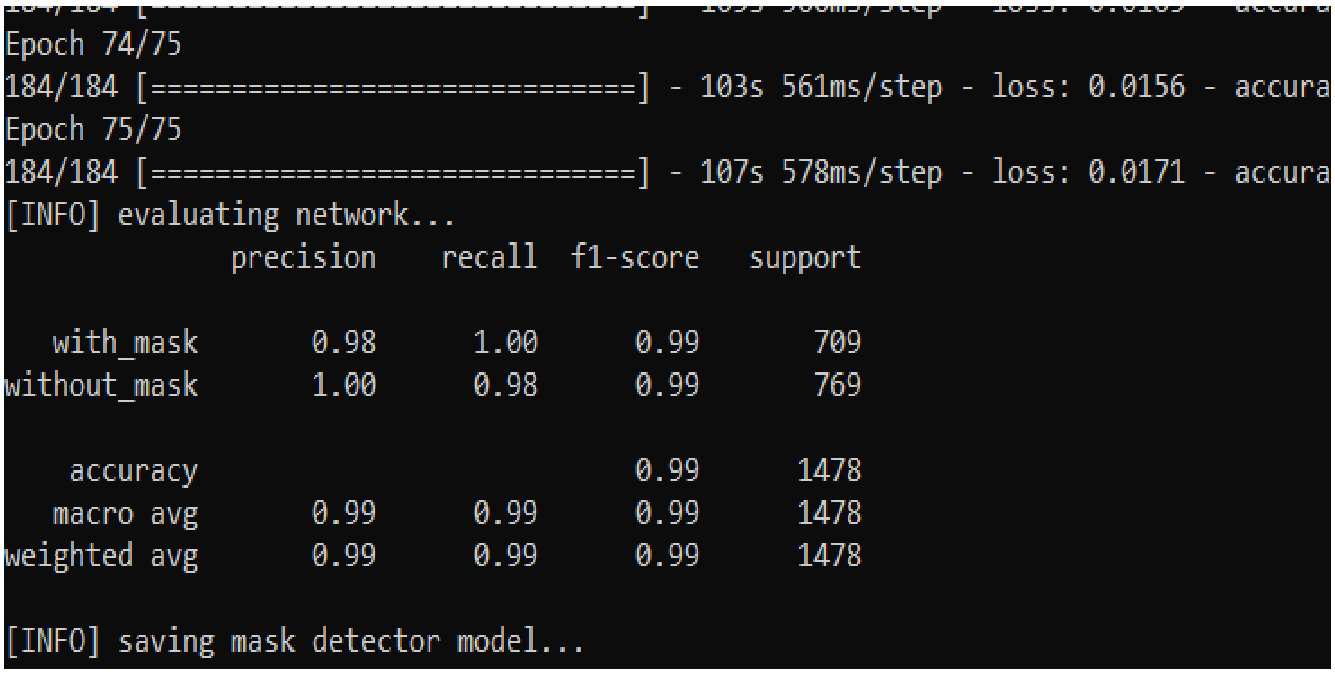
*Figure 14: Training and Compiling*

## 4.5 Running Epochs and Accuracy

An epoch refers to each cycle that a model takes through the full training dataset. For example, feeding the neural network with training data for more than one epoch, the result shouldbecome better in terms of predicting the given unseen data, which is the test data. In this approach,30 epochs were applied to achieve the be st accuracy results of predicting the test data. Table 1 shows the impact of epochs on the loss percentage.

| Epoch Number | Percentage of Loss | Percentage of Accuracy |
|---|---|---|
| **01** | **0.2419%** | **0.9316%** |
| **05** | **0.0612%** | **0.9826%** |
| **10** | **0.0434%** | **0.9871%** |
| **15** | **0.0399%** | **0.9883%** |
| **20** | **0.0347%** | **0.9889%** |
| **25** | **0.0277%** | **0.9901%** |
| **30** | **0.0272%** | **0.9910%** |
| **35** | **0.0253%** | **0.9915%** |
| **40** | **0.0227%** | **0.9910%** |
| **45** | **0.0211%** | **0.9920%** |
| **50** | **0.0204%** | **0.9929%** |
| **55** | **0.0220%** | **0.9922%** |
| **60** | **0.0224%** | **0.9917%** |
| **65** | **0.0178%** | **0.9934%** |
| **70** | **0.0162%** | **0.9939%** |
| **75** | **0.0171%** | **0.9940%** |

*Table 1: Epoch Number and Loss Percentage Results*



```
Epoch 74/75
184/184 [==============================] - 103s 561ms/step - loss: 0.0156 - accura
Epoch 75/75
184/184 [==============================] - 107s 578ms/step - loss: 0.0171 - accura
[INFO] evaluating network...
              precision    recall  f1-score   support

   with_mask       0.98      1.00      0.99       709
without_mask       1.00      0.98      0.99       769

    accuracy                           0.99      1478
   macro avg       0.99      0.99      0.99      1478
weighted avg       0.99      0.99      0.99      1478

[INFO] saving mask detector model...
```

*Figure 15: Accuracy Results*

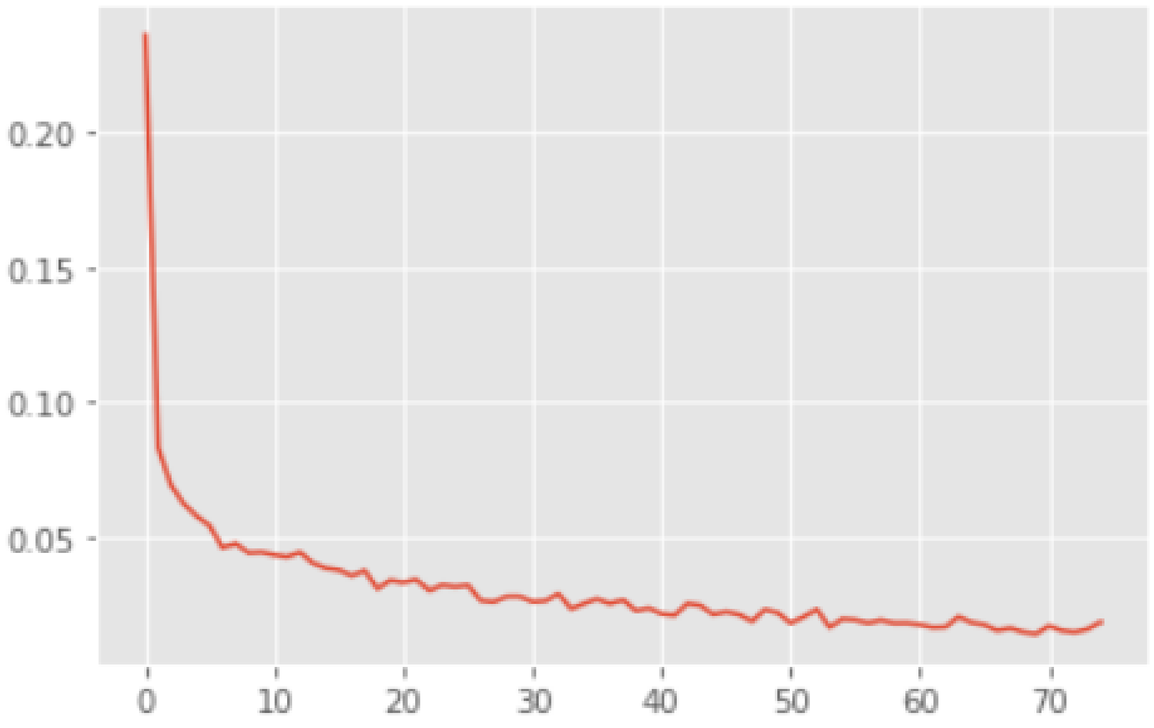


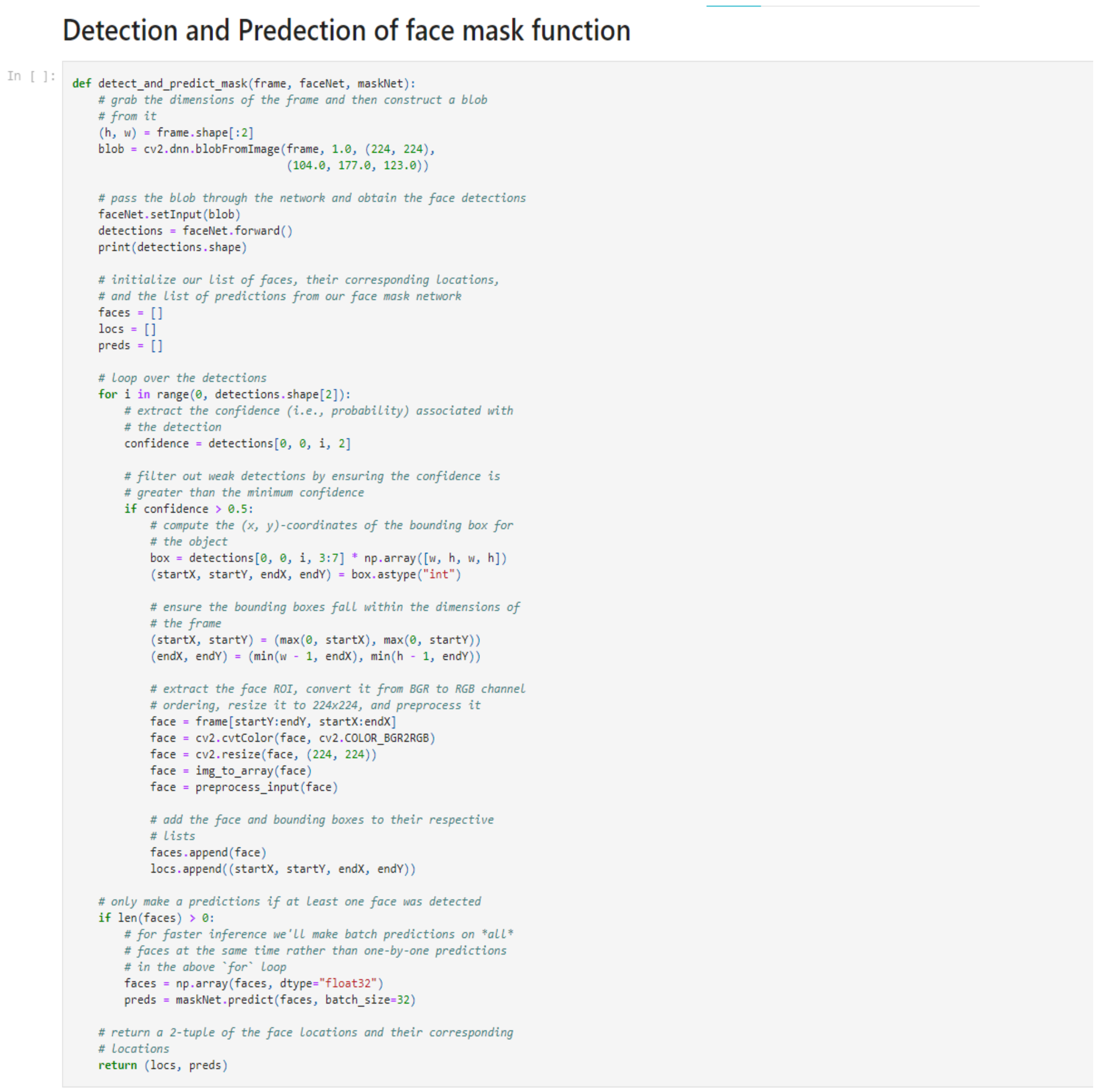
*Figure 16: Plot of results during Training*

## 4.6 Testing

For testing the face mask detector we need to load our pre-trained model from the output of Training file and then we use our caffemodel file to detect and predict the face mask.

### CaffeModel File:

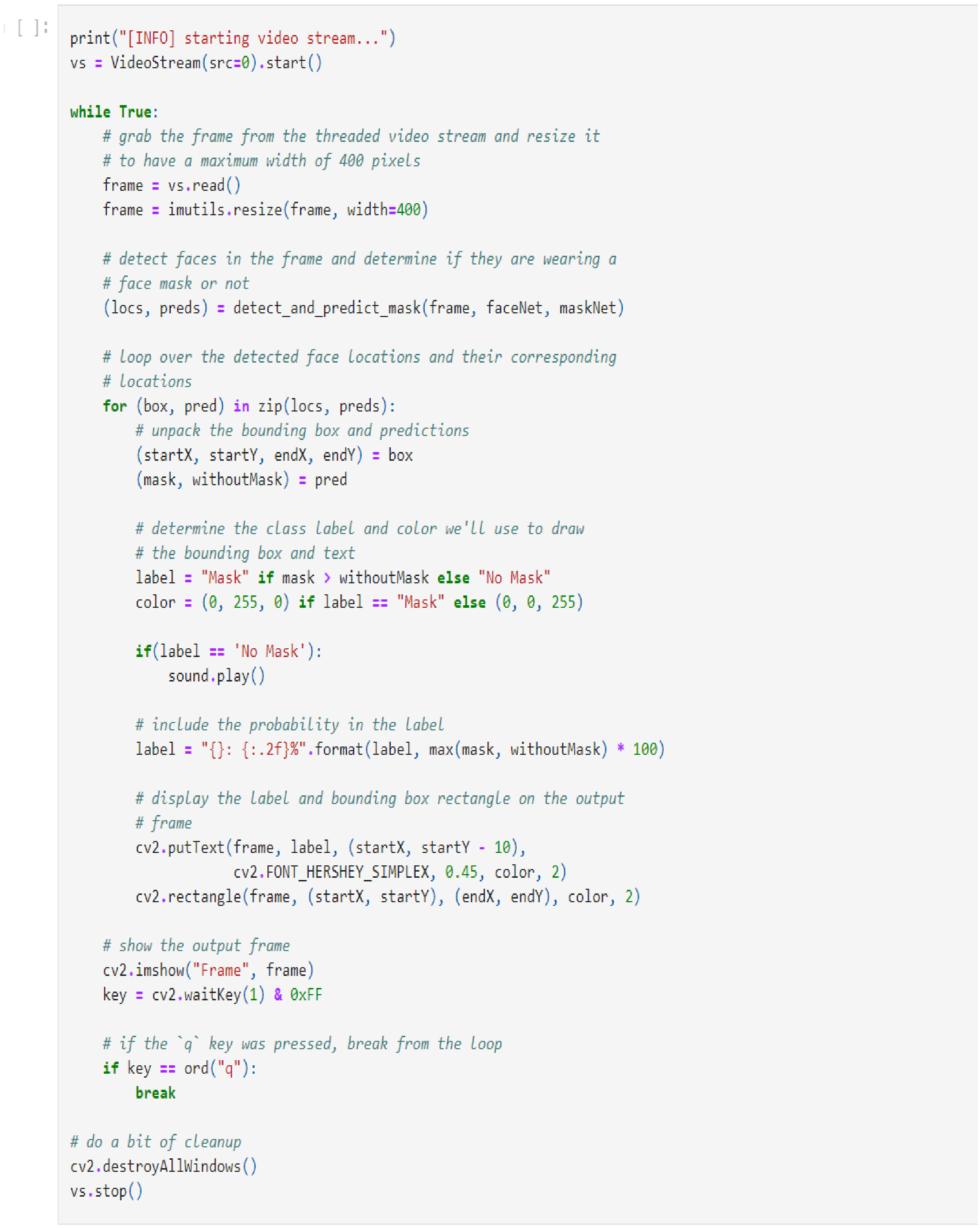
A CAFFEMODEL file is a machine learning model created by Caffe. It contains an image classification or image segmentation model that has been trained using Caffe. CAFFEMODEL files are created from .PROTOTXT files.



Detection and Predection of face mask function

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

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

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

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

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

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

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

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

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

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

*Figure 17: Testing the Trained model*



[ ]:

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

while True:
    # grab the frame from the threaded video stream and resize it
    # to have a maximum width of 400 pixels
    frame = vs.read()
    frame = imutils.resize(frame, width=400)

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

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

        # determine the class label and color we'll use to draw
        # the bounding box and text
        label = "Mask" if mask > withoutMask else "No Mask"
        color = (0, 255, 0) if label == "Mask" else (0, 0, 255)

        if(label == 'No Mask'):
            sound.play()

        # include the probability in the label
        label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)

        # display the label and bounding box rectangle on the output
        # frame
        cv2.putText(frame, label, (startX, startY - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
        cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)

    # show the output frame
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF

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

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

*Figure 18: Detecting the face mask using OpenCV*

# CHAPTER 05

# Results and Discussions

## 5.1 Result on Testing

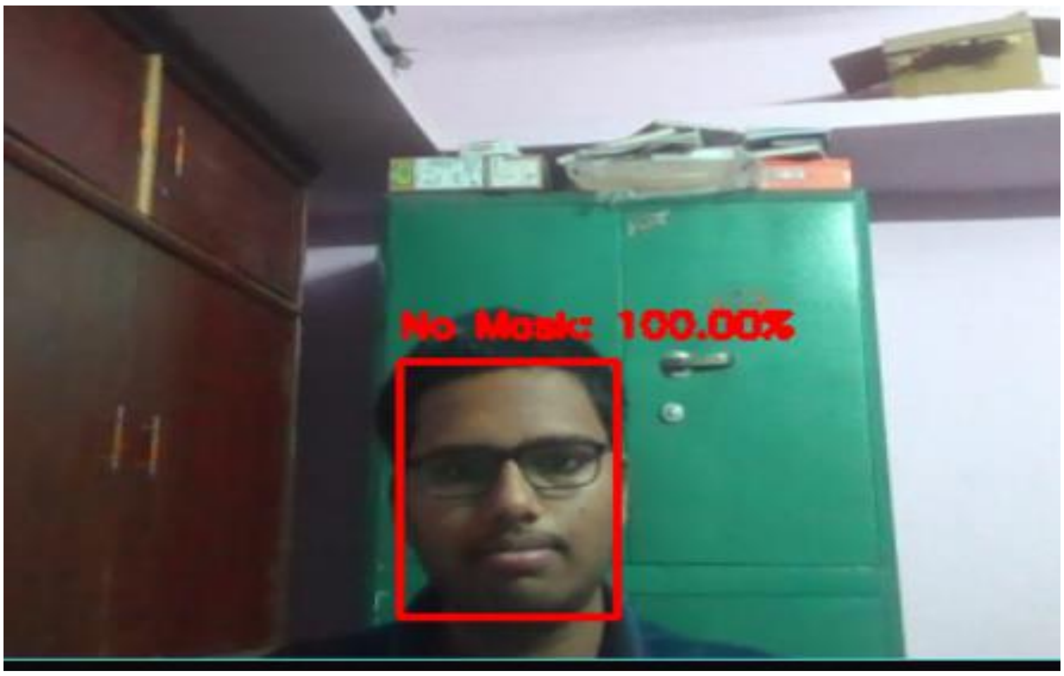


*Figure 19: Mask Detector Detecting with No Mask*

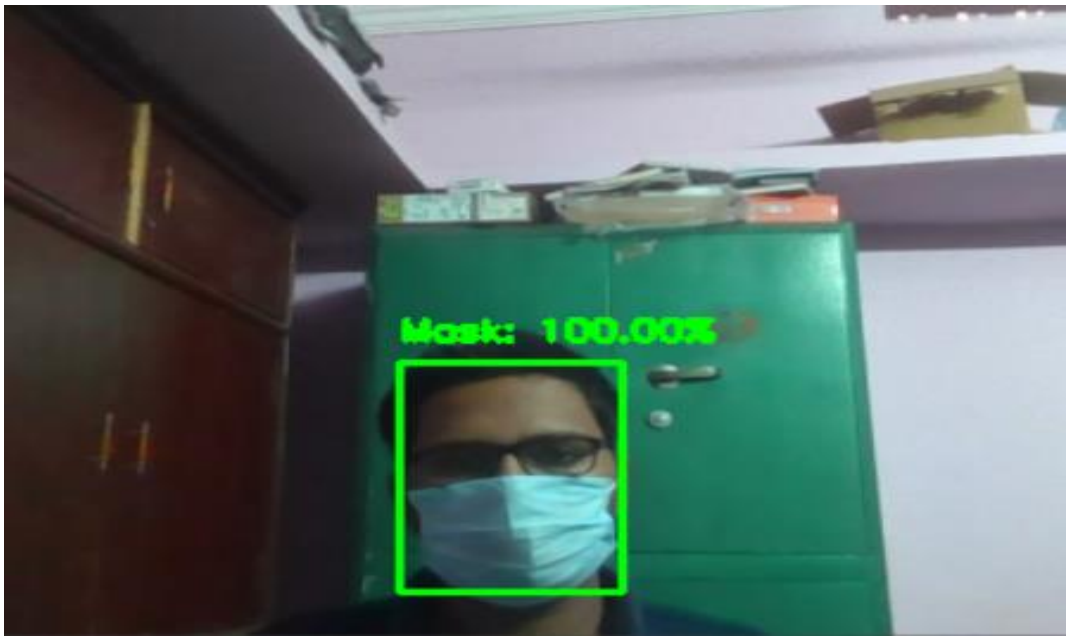
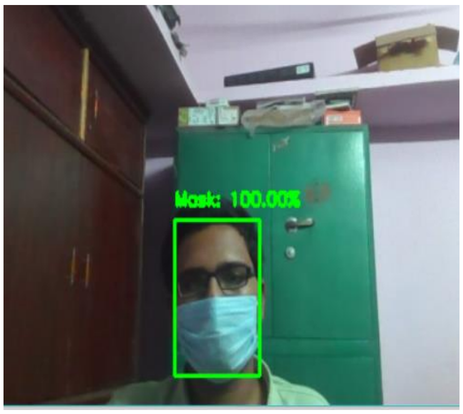
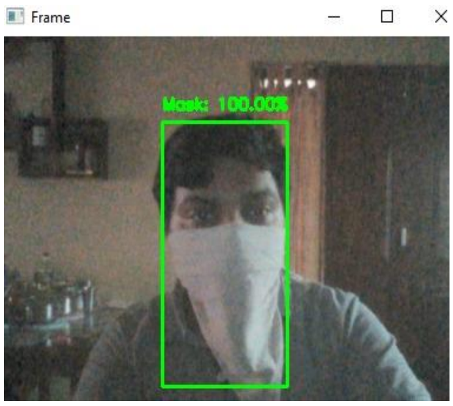
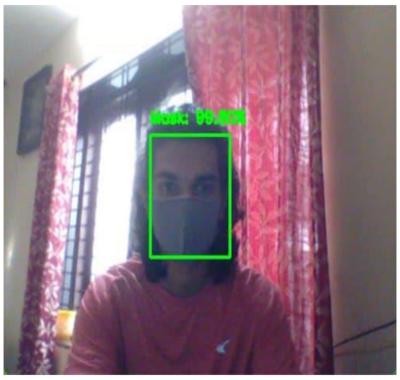


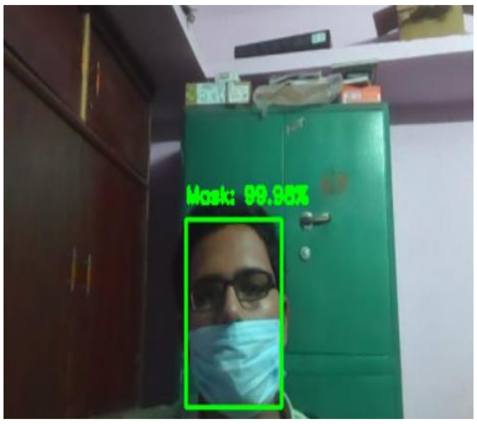
*Figure 20: Mask Detector Detecting with Mask*

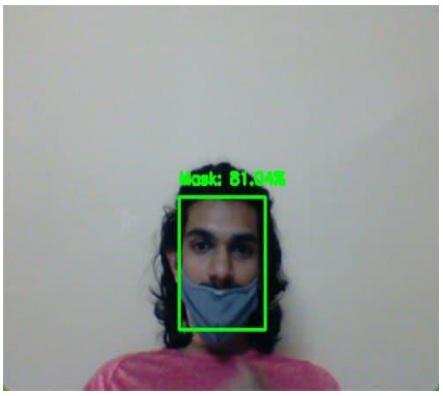
## 5.2 Some features of Face Mask Detector

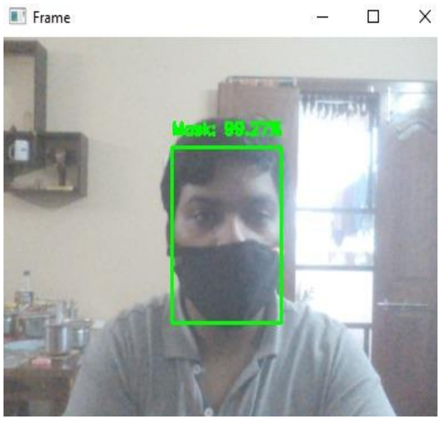


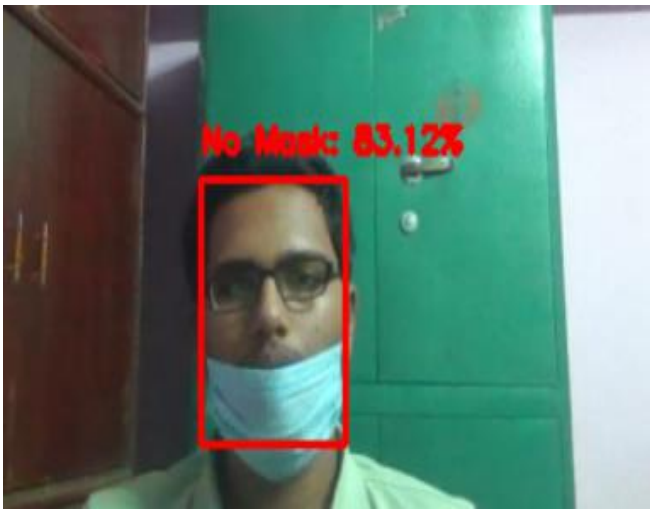


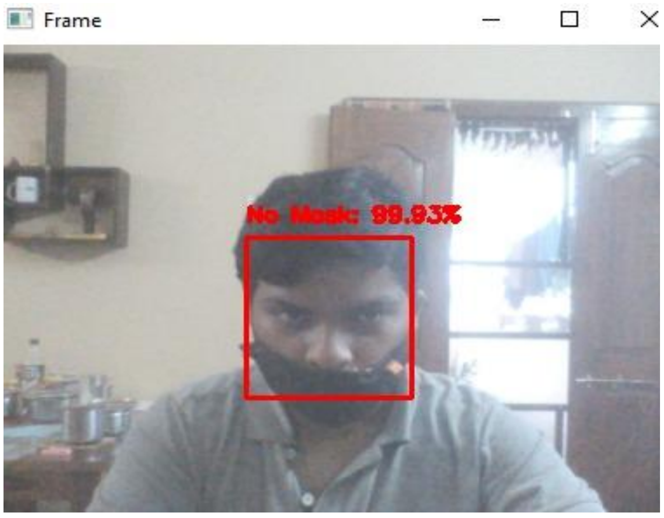


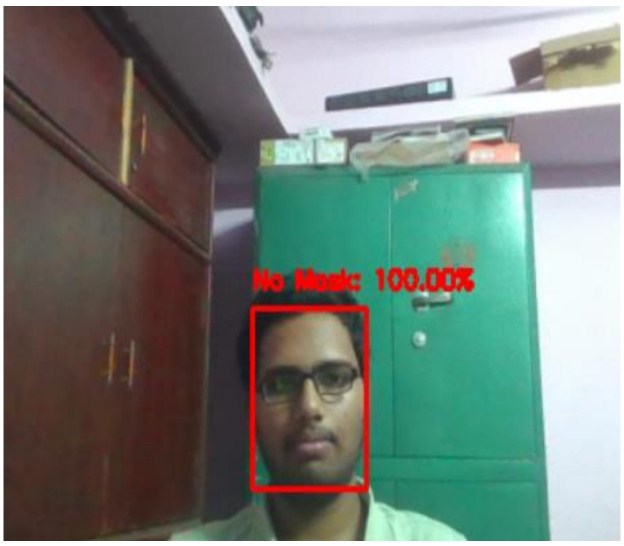


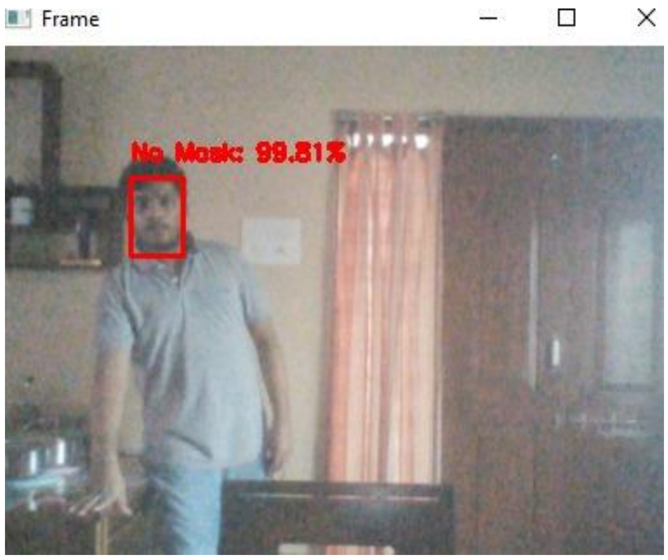


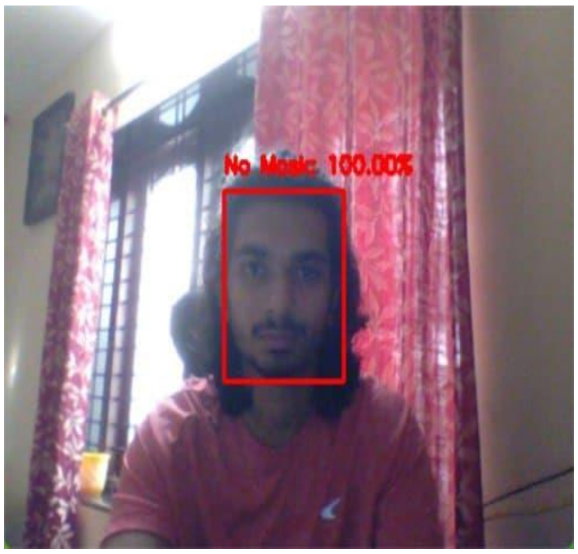


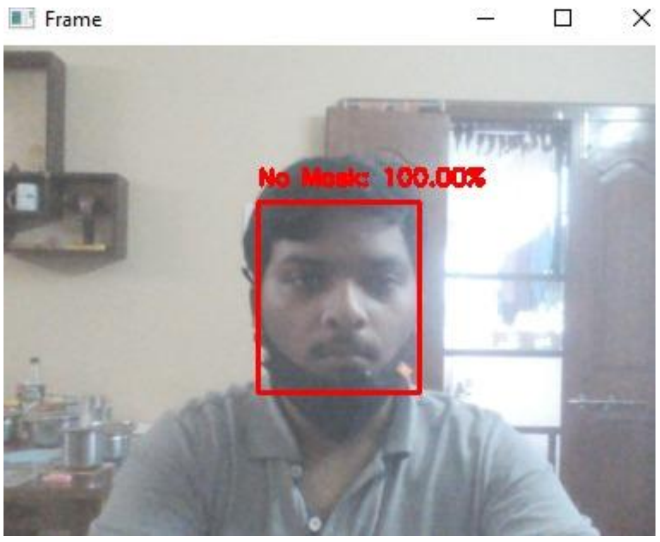


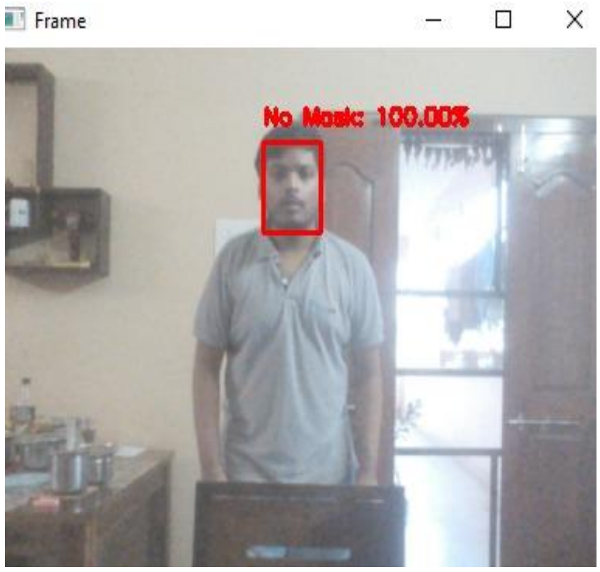


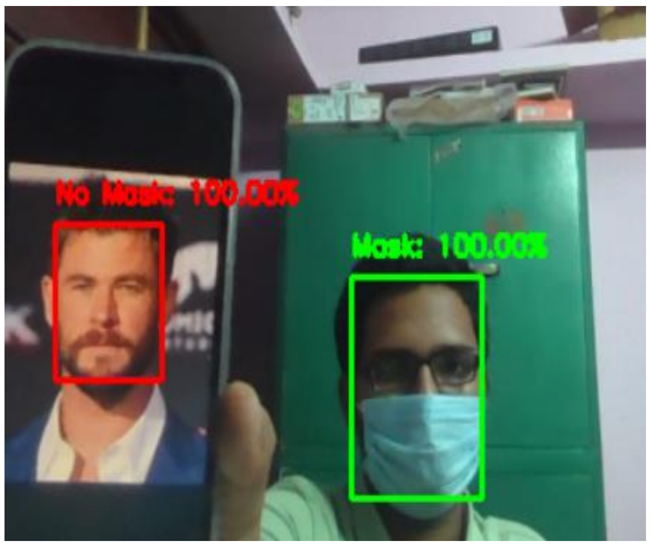

No Mask: 100.00%


Frame
No Mask: 99.81%

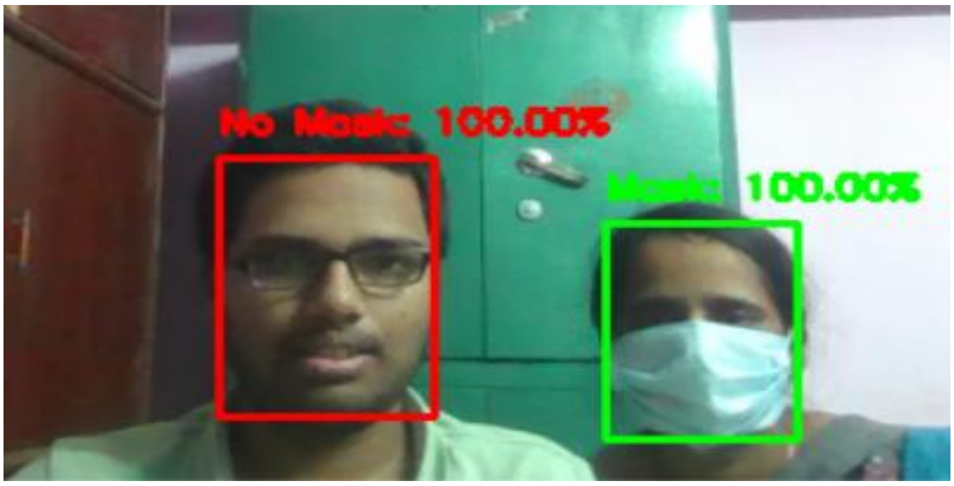

No Mask: 100.00%

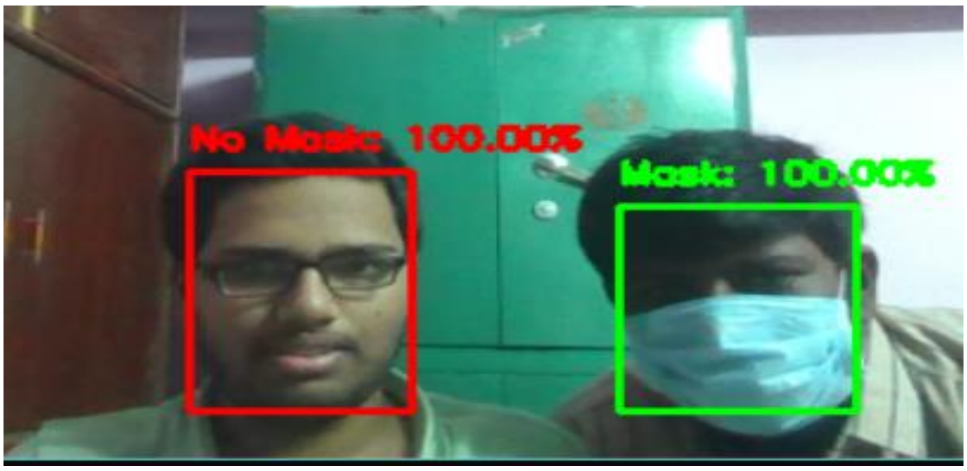

Frame
No Mask: 100.00%

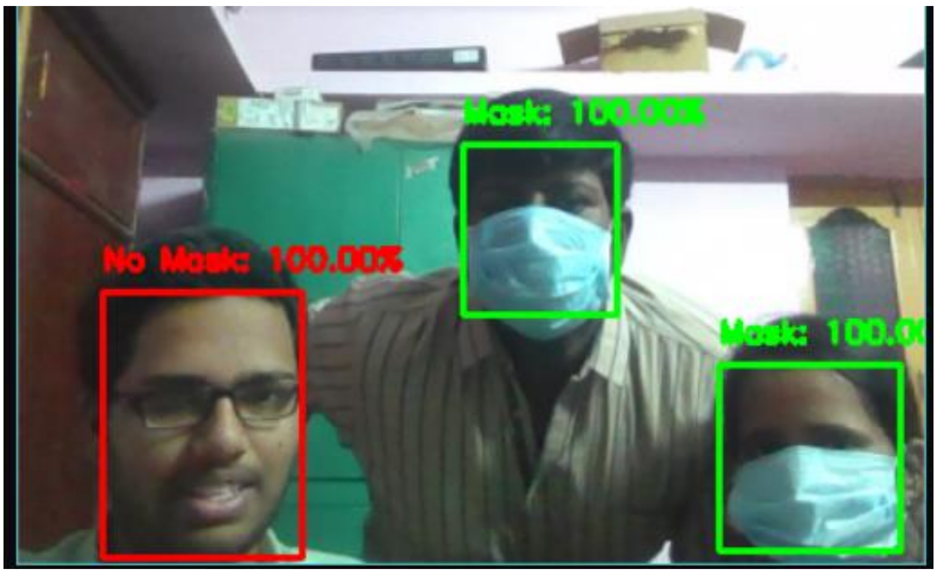

Frame
No Mask: 100.00%


No Mask: 100.00%
Mask: 100.00%

## 5.3 Multiple Mask Detector







**Fail Case :**

The one failure of face mask detector is it can not work on the very low quality camera.

## 5.4 Summary

Results on Testing from above trained model . Testing is done for this project as automation for Face Mask Detector was not possible.

# CHAPTER 06

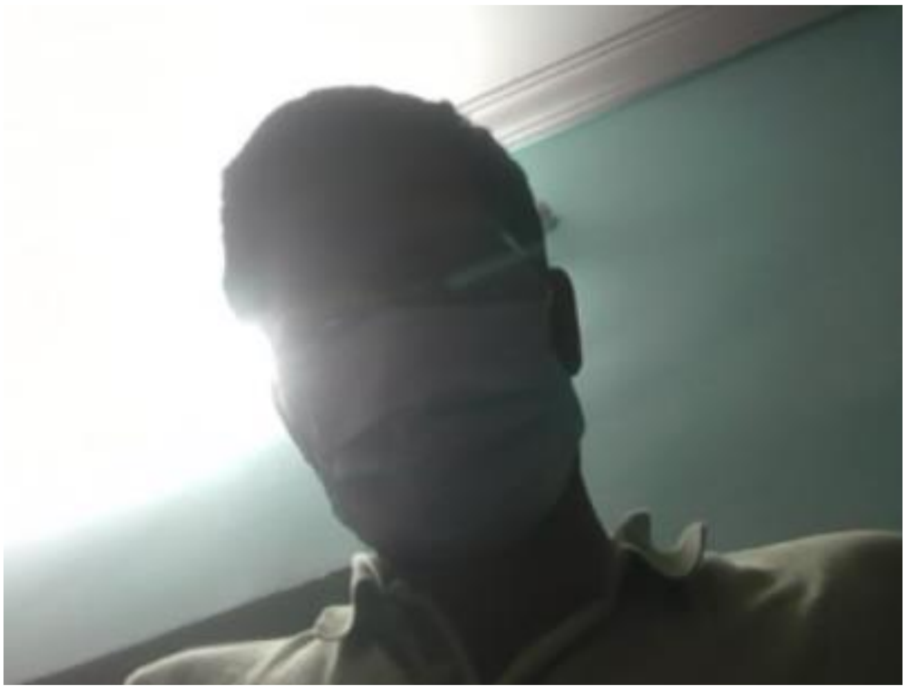
# Conclusion and Future Scope

## 6.1 Limitations

The identification of facemask- wearing conditions has multitudinous parallels to facial recognition. Still, the development of a facemask wearing condition identification network is challenging for several reasons. The limitations we have in the sector of datasets are one of themost challenging issues to face. The size of facemask- wearing condition datasets comparatively smaller than general facial recognition datasets. And the image quality of these datasets are not as high enough, compared to generally available facial recognition datasets. Likewise, the various performances of wearing facemasks erroneously largely increase the difficulty of identification. There are a numerous limitations too our study. Firstly, the dataset of Medical Masks which we used for face-mask wearing situation/condition identification is fairly small, and it can't cover all postures or surroundings. Also, the dataset does not contain video, where the identification result on a video aqueduct can't be tested. Another limitation isthe devilish data loading time in Jupyter Tablet while loading the dataset into it.

And one farther limitation in our Face Mask Sensor is low- quality cameras are not working properly. And the Camera should not be exposed to the light, it causes no discovery of mask. Let’s take a small illustration in the below image.

So also the camera is exposing farther light which is not fluently catch person’s face so that’s why the Mask Sensor is not detecting, since we didn’t train our model in that way.



So here the camera is exposing more light which is not clearly catch person’s face so that’s why the Mask Detector is not detecting, since we didn’t train our model in that way.

### Recommended Camera Settings:

**Quality**: highest (minimum 720p is recommended for good prediction)
**Bitrate**: Maximum Possible
**Profile**: Maximum possible
**I-Frame Interval (GOV):** 50

**Stream anti-aliasing**: off

**Exposure and Brightness**: Ensuring that the face is clearly seen (if the camera faces light source, the overexposed background is unacceptable)

**Shutter Speed**: Must not be too low (more than 1/50), because in such a case the blurring of the moving objects will occur).

## 6.2 Scope of the Project

All around the world nearly more than 50 countries have lately initiated wearing face masks mandatory People have to cover their faces in public, supermarkets, public transports, services, and stores. Retail companies frequently use software to count thenumber of people entering their stores. They may also like to measure prints on digital displays and promotional defenses. We're planning to ameliorate ourFace Mask Detection tool and release it as an open- source design. Our software can be equated to any being USB, IP cameras, and CCTV cameras to descry people without a mask. This discovery live videotape feed can be enforced in web and desktop operations sothat the driver can see notice dispatches. Software drivers can also get an image in case someone isn't wearing a mask. Here we are arranging an alarm system which will produce high intensity buzzer sounds when a person is not wearing a face-mask or properly not wearing a face-mask when they get detected by the system. This software can also be connected to the entrance gates and only people wearing face masks can go through the entrance.

## 6.3 Future Goals

- Integrate with the voice system
- Mask Detecting with very low-end camera
- Creating an Android Application.
- Upper Clothes Colour detection (like a blue shirt)
- Identity Detection

# References

[1] Chavez, S.; Long, B.; Koyfman, A.; Liang, S.Y. Coronavirus Disease (COVID-19): A primer for emergency physicians. Am. J. Emerg. Med. 2020. [CrossRef] [PubMed].

[2] Cowling, B.J.; Chan, K.H.; Fang, V.J.; Cheng, C.K.; Fung, R.O.; Wai, W.; Sin, J.; Seto, W.H.; Yung, R.; Chu, D.W.; et al. Facemasks and hand hygiene to prevent influenza transmission in households: A cluster randomized trial. Ann. Intern. Med. 2009, 151, 437–446. [CrossRef] [PubMed].

[3] Jefferson, T.; Del Mar, C.B.; Dooley, L.; Ferroni, E.; Al-Ansary, L.A.; Bawazeer, G.A.; van Driel, M.L.; Nair, S.; Jones, M.A.; Thorning, S.; et al. Physical interventions to interrupt or reduce the spread of respiratory viruses. Cochrane Database Syst. Rev. 2011, CD006207. [CrossRef] [PubMed].

[4] Sim, S.W.; Moey, K.S.; Tan, N.C. The use of facemasks to prevent respiratory infection: A literature review in the context of the Health Belief Model. Singap. Med. J. 2014, 55, 160–167. [CrossRef].

[5] Jefferson, T.; Foxlee, R.; Del Mar, C.; Dooley, L.; Ferroni, E.; Hewak, B.; Prabhala, A.; Nair, S.; Rivetti, A. Physical interventions to interrupt or reduce the spread of respiratory viruses: Systematic review. BMJ 2008, 336, 77–80. [CrossRef].

[6] Cristani, M.; Bue, A.D.; Murino, V.; Setti, F.; Vinciarelli, A. The Visual Social Distancing Problem. IEEE Access 2020, 8, 126876–126886. [CrossRef].

[7] Elachola, H.; Ebrahim, S.H.; Gozzer, E. COVID-19: Facemask use prevalence in international airports in Asia, Europe and the Americas, March 2020. Travel Med. Infect. Dis. 2020, 101637. [CrossRef].

[8] Lai, A.C.; Poon, C.K.; Cheung, A.C. Effectiveness of facemasks to reduce exposure hazards for airborne infections among general populations. J. R. Soc. Interface 2012, 9, 938–948. [CrossRef] [PubMed].

[9] R. Gambler, "Facial Recognition Technology," United States Government Accountability Office, September 2020. [Online]. Available: https://www.gao.gov/assets/gao-20-568.pdf

[10] Coolfire, "How U.S. Airports Use Facial Recognition," Coolfire Core Solutions, 29 November 2018. [Online].Available:https://www.coolfiresolutions.com/blog/airportfacial-recognition-technology/.

[11] P. Nagrath, R. Jain, A. Madan, R. Arora, P. Kataria and J. Hemanth, "SSDMNV2: A real time DNN-based face mask detection system using single shot multibox detector and MobileNetV2," Sustainable cities and society, March 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7775036/#sec0085.

[12] S. V. M. a. N. V. Dionisio, "Real-Time Facemask Recognition with Alarm System using Deep Learning," IEEE Control and System Graduate Research Colloquium (ICSGRC), 2020 [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9232610.

[13] S. R. B. a. J. S. Kumar, "Geometric shaped facial feature extraction for face recognition," 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7887965.

[14] T. P. o. A. PAI, "Understanding Facial Recognition Systems," 19 February 2020. [Online]. Available:https://www.partnershiponai.org/wp-content/uploads/2020/02/Understanding-Facial-Recognition-Paper_final.pdf

[15] R. Materese, "NIST Launches Studies into Masks' Effect on Face Recognition Software," NIST,4 August 2020. [Online]. Available:https://www.nist.gov/newsevents/news/2020/07/nist-launches-studies-masks-effect-face-recognition-software.

[16] M. Burugupalli, "IMAGE CLASSIFICATION USING TRANSFER LEARNING AND CONVOLUTION NEURAL NETWORKS,"July 2020.[Online]. Available:https://library.ndsu.edu/ir/bitstream/handle/10365/31517/Image%20Classification%20Using%20Transfer%20Learning%20and%20Convolution%20Neural%20Networks.pdf?sequence=1

[17] Z. Elhamraoui, "Fine-tuning in Deep Learning," Artificial Intellgence, 23 June 2020. [Online]. Available: https://ai.plainenglish.io/fine-tuning-in-deep-learning-909666d4c151\

[18] https://www.ijert.org/research/covid-19-facemask-detection-with-deep-learning-and-computer- vision-IJERTCONV9IS05017.pdf/

[19] https://www.scribd.com/document/541008885/Facial-Recognition-and-Face-Mask-Detection- Using-Machine-Learning

[20] https://digitalcommons.montclair.edu/cgi/viewcontent.cgi?article=1729