

A Project Report

On

Face Mask Detection Using Image Processing

Submitted in partial fulfilment of the requirement for the award of the degree of

Bachelor Of Technology

In

Electronics And Communication Engineering

Submitted by

C.SRI HARSHANADH REDDY

19191A0409

K.SUPRAJA

19191A0437

B.SUPRIYA

19191A0446

N.MADAN KUMAR

19191A0414

Under the Esteemed guidance of

K.RAVINDRA REDDY

Assistant Professor (Adhoc)

Department of ECE

JNTUA College of Engineering (Autonomous)

Pulivendula.



Department Of Electronics & Communication Engineering

Jawaharlal Nehru Technological University Anantapur College Of Engineering (Autonomous),
Pulivendula – 516 390, Y.S.R. (District), Andhra Pradesh, India

2017-2021

Jawaharlal Nehru Technological University Anantapur
College Of Engineering (Autonomous), Pulivendula
Department Of Electronics And Communication Engineering



CERTIFICATE

This is to certify that the project work entitles “ **Face Mask Detection Using Image Processing**” has been submitted by

C.SRI HARSHANADH REDDY	19191A0409
K.SUPRAJA	19191A0437
B.SUPRIYA	19191A0446
N.MADAN KUMAR	19191A0414

in partial fulfilment of the requirements for the award of the Bachelor of Technology in Electronics and Communication Engineering from JNTUA College of Engineering (Autonomous), Pulivendula during 2017-2021.

Signature of Project Guide:

Signature of Head of the Department:

K.RAVINDRA REDDY

Prof. R. RAMANA REDDY

Assistant professor (adhoc) in

Professor and Head,

Department of ECE,

Department of ECE,

JNTU college Of Engineering, Pulivendula.

JNTU College Of Engineering, Pulivendula

Jawaharlal Nehru Technological University Anantapur
College Of Engineering (Autonomous), Pulivendula
Y.S.R. Kadapa District (516390), Andhra Pradesh - India
Department Of Electronics And Communication Engineering



STUDENT DECLARATION

We hereby declare that the project entitled “**Face Mask Detection Using Image Processing**” is our own work and that to the best of our knowledge and belief it contains no material previously published or material which has been accepted for the award of any degree or diploma of any University or institute of higher learning.

C.SRI HARSHANADH REDDY	19191A0409
K.SUPRAJA	19191A0437
B.SUPRIYA	19191A0446
N.MADAN KUMAR	19191A0414

CONTENTS

I Abstract

II List Of Figures

III List Of Tables

Chapter No.	Chapter Name	Page No.
1	Introduction to machine learning	11
	1.1 Machine learning-Overview	
	1.1.1 Key Features	
	1.1.2 Advantages	
	1.1.3 Applications	
2	Literature Review	15
3	Existing System	16
4	Proposed System	17
5	Methodology	18
	5.1 Block Diagram	
	5.2 Flow Chart	
6	Software Requirements	20
	6.1 Python	
	6.1.1 Introduction	
	6.1.2 Python Downloading	
	6.1.3 Installing	
	6.1.4 verifying	
7	Libraries	26
	7.1 Command prompt	
	7.1.1 Access of command Prompt	
	7.1.2 Using of command Prompt	
	7.2 Libraries required	

	7.2.1 Pip upgrade	
	7.2.2 Numpy	
	7.2.3 OpenCV	
	7.2.4 Tensor flow	
	7.2.5 Imutils	
	7.2.6 Face_mask_detector	
8	Programs	35
	8.1 Source code for training	
	8.2 Source code for testing	
	8.3 Data set	
9	Working of the system	40
	9.1 working principle	
10	Result Analysis	43
	10.1 Results	
11	Conclusion And Future Enhancement	45
	11.1 Conclusion	
	11.2 Future Enhancement	
	References	46

Acknowledgement

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express our gratitude for all of them. The first person we would like to thank is our guide K.RAVINDRA REDDY, Assistant Professor (Adhoc), Department of Electronics and Communication Engineering J.N.T.U.A college of Engineering, Pulivendula. His wide knowledge and logical way of thinking have made a deep impression on us. He is a source of inspiration for innovative ideas and his kind of support is well known to all his students and colleagues.

We are grateful to Prof. R. RAMANA REDDY, Professor and Head of the Department of Electronics and Communication Engineering, J.N.T.U.A college of Engineering, Pulivendula who has extended his support for the success of this project.

We are grateful to Dr. SHAIK. TAJ MAHABOOB, Assistant Professor of the Department of Electronics and Communication Engineering, JNTUA college of Engineering, Pulivendula who has extended her support for the success of this project.

We wish to thank Prof. G. V. SUBBA REDDY, Vice Principal of the college, JNTUA college of Engineering, Pulivendula who has extended his support for the success of this project.

We wish to thank Prof. G. SANKARA SEKHAR RAJU, Principal of the college, JNTUA college of Engineering, Pulivendula who has extended his support for the success of this project

We wish to express our thanks to all staff members and our friends who have rendered their whole hearted support at all times for the successful completion of this project.

Finally, we acknowledge with gratitude the unflagging support and patience of our PARENTS for their guidance and encouragement during this dissertation work.

I

Abstract

During the time of this pandemic, there are some strict regulations that need to be followed to maintain the decorum of the city, state, or country. Since we can't always have the official authority on the lookout for some people not abiding by the rules, we can construct a face mask detection project that will enable us to figure out if a particular person is wearing a mask or not. During this time, with strict regulations of the lockdown, it would be a brilliant idea to implement this project to contribute to the upkeep of the society.

Hence, a project in which you can process images of an entire area or region by tracking people on the road or streets to analyse if they are wearing masks or not would be a spectacular idea. With the help of image processing algorithms and deep learning techniques, you can compute images of people who are wearing masks. The following Kaggle dataset for face mask detection would be a great starting point to analyse the training images for achieving an overall high accuracy.

One of the best ways to approach this problem would be to make use of transfer learning models such as ,VGG-16, face-net, RESNET-50,python,opencv and tensor flow and other similar architectures to see what method helps you to achieve the best results. As a starting point, we would highly recommend checking out one of my previous articles on smart face lock systems, where we construct some high-level face recognition systems. We can use a Arduino for faces with no mask and faces with a mask to solve this type of task.

II List of Figures

1. Block Diagram
2. Flow Chart
3. python browser page
4. Download specific version
5. saving python file
6. python file in window
7. Installing python Window
8. Installing window
9. Window of installed python
10. Pop up window
11. Checking for python version in command prompt
12. Opening command prompt window
13. Appearance of command prompt window
14. Window showing installed pip
15. Command prompt to install numpy
16. Installing numpy in command prompt
17. Showing numpy installed in command prompt
18. Command prompt for installing OpenCV
19. Installed OpenCV in command prompt
20. Installing tensor flow in command prompt
21. Installed tensor flow
22. Installed imutils in command prompt
23. Command Prompt for installing Face_mask_detector

24. Installing Face_mask_detector library
25. Installed Face_mask_detector library
26. Dataset Visualization
27. Dataset for Mask and no Mask
28. Working of the system
29. Detecting mask and no mask with percentage
30. Result with no mask
31. Result with mask
32. Result with no mask for multiple people
33. Result with mask for multiple people
34. Result of detecting and classifying with mask and no mask
35. Result of detecting all improper way of mask

III List of Tables

1. Percentage values for different classification of mask

Chapter 1

Introduction To Machine Learning

1.1 Machine learning - Overview

Machine learning is a field of study that looks at using computational algorithms to turn empirical data into usable models. The machine learning field grew out of traditional statistics and artificial intelligences communities. From the efforts of mega corporations such as Google, Microsoft, Facebook, Amazon, and so on, machine learning has become one of the hottest computational science topics in the last decade. Through their business processes immense amounts of data have been and will be collected. This has provided an opportunity to re-invigorate the statistical and computational approaches to autogenerate useful models from data.

Machine learning is a tool for turning information into knowledge. In the past 50 years, there has been an explosion of data. This mass of data is useless unless we analyse it and find the patterns hidden within. Machine learning techniques are used to automatically find the valuable underlying patterns within complex data that we would otherwise struggle to discover. The hidden patterns and knowledge about a problem can be used to predict future events and perform all kinds of complex decision making.

1.1.1 Key Features

Features are nothing but the independent variables in machine learning models. What is required to be learned in any specific machine learning problem is a set of these features (independent variables), coefficients of these features, and parameters for coming up with appropriate functions or models

- **Dataset:** A set of data examples, that contain features important to solving the problem.
- **Features:** Important pieces of data that help us understand a problem. These are fed in to a Machine Learning algorithm to help it learn.
- **Model:** The representation (internal model) of a phenomenon that a Machine Learning algorithm has learnt. It learns this from the data it is shown during training. The model is the output you get after training an algorithm. For example, a decision tree algorithm would be trained and produce a decision tree model.

Process:

- a) **Data Collection:** Collect the data that the algorithm will learn from.
- b) **Data Preparation:** Format and engineer the data into the optimal format, extracting important features and performing dimensionality reduction.
- c) **Training:** Also known as the fitting stage, this is where the Machine Learning algorithm actually learns by showing it the data that has been collected and prepared.
- d) **Evaluation:** Test the model to see how well it performs.
- e) **Tuning:** Fine tune the model to maximise it's performance.

1.1.2 Advantages

The advantages of Machine Learning span across every area of lifestyle and business. Here is a list of some of the advantages that Machine Learning has to offer –

➤ Easily identifies trends and patterns

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviours and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

➤ No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus software ; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

➤ Continuous Improvement

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

➤ Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

1.1.2 Applications

Popular Machine Learning Applications

➤ Social Media Features

Social media platforms use machine learning algorithms and approaches to create some attractive and excellent features. For instance, Facebook notices and records your activities, your chats, likes, and comments, and the time you spend on specific kinds of posts. Machine learning learns from your own experience and makes friends and page suggestions for your profile.

➤ Product Recommendations

One of the most popular and known applications of machine learning is Product Recommendation. Product recommendation is one of the stark features of almost every e-commerce website today, which is an advanced application of machine learning techniques. Using machine

learning and AI, websites track your behaviour based on your previous purchase, your searching pattern, your cart history, and make product recommendations.

➤ **Image Recognition**

Image Recognition is one of the most significant and notable Machine Learning and AI techniques: an approach for cataloguing and detecting a feature or an object in the digital image. This technique is being adopted for further analysis, such as pattern recognition, face detection, or face recognition.

➤ **Sentiment Analysis**

Sentiment analysis is one of the most necessary Applications of Machine Learning. Sentiment analysis is one of the most necessary Applications of Machine Learning. Sentiment analysis is a real-time machine learning application that determines the emotion or opinion of the speaker or the writer. For instance, if someone has written a review or email (or any form of a document), a sentiment analyser will instantly find out the actual thought and tone of the text. This sentiment analysis application can be used to analyse a review based website, decision-making applications, etc.

➤ **Automating Employee Access Control**

Organizations are actively implementing machine learning algorithms to determine the level of access employees would need in various areas, depending on their job profiles. This is one of the coolest applications of Machine Learning.

➤ **Marine Wildlife Preservation**

Machine learning algorithms are used to develop behaviour models for endangered cetaceans and other marine species, helping scientists regulate and monitor their populations.

➤ **Regulating Healthcare Efficiency and Medical Services**

Significant healthcare sectors are actively looking at using Machine Learning algorithms to manage better. They predict the waiting times of patients in the emergency waiting rooms across various departments of hospitals. The models use vital factors that help define the algorithm, details of staff at various times of day, records of patients, and complete logs of department chats and the layout of emergency rooms. Machine learning algorithms also come to play when detecting a disease, therapy planning, and prediction of the disease situation. This is one of the most necessary Applications of Machine Learning.

➤ **Predict Potential Heart Failure**

An algorithm designed to scan a doctor's free-form e-notes and identify patterns in a patient's cardiovascular history is making waves in medicine. Instead of a physician digging through multiple health records to arrive at a sound diagnosis, redundancy is now reduced with computers making an analysis based on available information.

➤ **Banking Domain**

Banks are now using the latest advanced technology machine learning has to offer to help prevent fraud & protect accounts from hackers. The algorithms determine what factors to consider to create a filter to keep harm at bay. Various sites that are unauthentic will be automatically filtered out and restricted from initiating transactions.

➤ **Language Translation**

One of the most common applications of Machine Learning is Language Translation .Machine learning plays a significant role in the translation of one language to another. We are amazed at how the websites can translate from one language to another effortlessly and gives contextual meaning as well. The technology behind the translation tool is called ‘machine translation.’ It has enabled the world to interact with people from all corners of the world; without it, life would not be as easy as it is now. It has provided a sort of confidence to travellers and business associates to safely venture into foreign lands with the conviction that language will no longer be a barrier.

Your model will need to be taught what you want it to learn. Feeding relevant back data will help the machine draw patterns and act accordingly. So it is imperative to provide relevant data and feed files to help the machine learn what is expected. In this case, with Machine Learning – it is what you want as a result depends on the contents of the files that are being recorded.

Chapter 2

Literature Review

Ms. R. Suganthalakshmi at all proposed

TITLE : Face Mask Detector

Single Shot Detector architecture is used for the object detection purpose. In this system face mask detector can be deployed in many areas like shopping malls, airports and other heavy traffic places to monitor the public and to avoid the spread of the disease by checking who is following basic rules and who is not. It takes excessive time for data loading in Google Colab Notebook. It did not allow the access of webcam which posed a hurdle in testing images and video stream. We have modeled a facemask detector using Deep learning. We are processed a system computationally efficient using MobileNetV2 which makes it easier to Extract the data sets. We use CNN architecture for better performance. We can fix it in any kind of cameras

TITLE :Face detection techniques: a review , 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 ,head- pose 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.

TITLE : 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)

TITLE : Real time face mask recognition with alarm system using deep learning

This process gives a precise and speedily results for facemask detection. Raspberry pi based real time face mask recognition that captures the facial image. This system uses the

architectural features of VGG-16 as the foundation network for face recognition. Deep learning techniques are applied to construct a classifier that will collect image of a person wearing a face mask and no masks. Our proposed study uses the architectural features of CNN as the foundation network for face detection. It shows accuracy in detecting person wearing a face mask and not wearing a face mask. This study presents a useful tool in fighting the spread of COVID-19 virus.

Chapter 3

Existing System

- Face recognition temperature screening system exists in which they display faces .

Existing System Uses:

- It will display faces as well as temperature in one screen.

Existing System Disadvantages:

- The price for facial recognition temperature screening system is very expensive.
- It is hard to implement .
- Hard to install.
- It is difficult to design.

Chapter 4

Proposed System

- The present designing system is efficient and machine learning based.
- Mask can be detected in public places at low cost .
- It is designed using python , opencv and tensor flow.
- It is easy to implement .

Chapter 5

Methodology

5.1 Block Diagram :

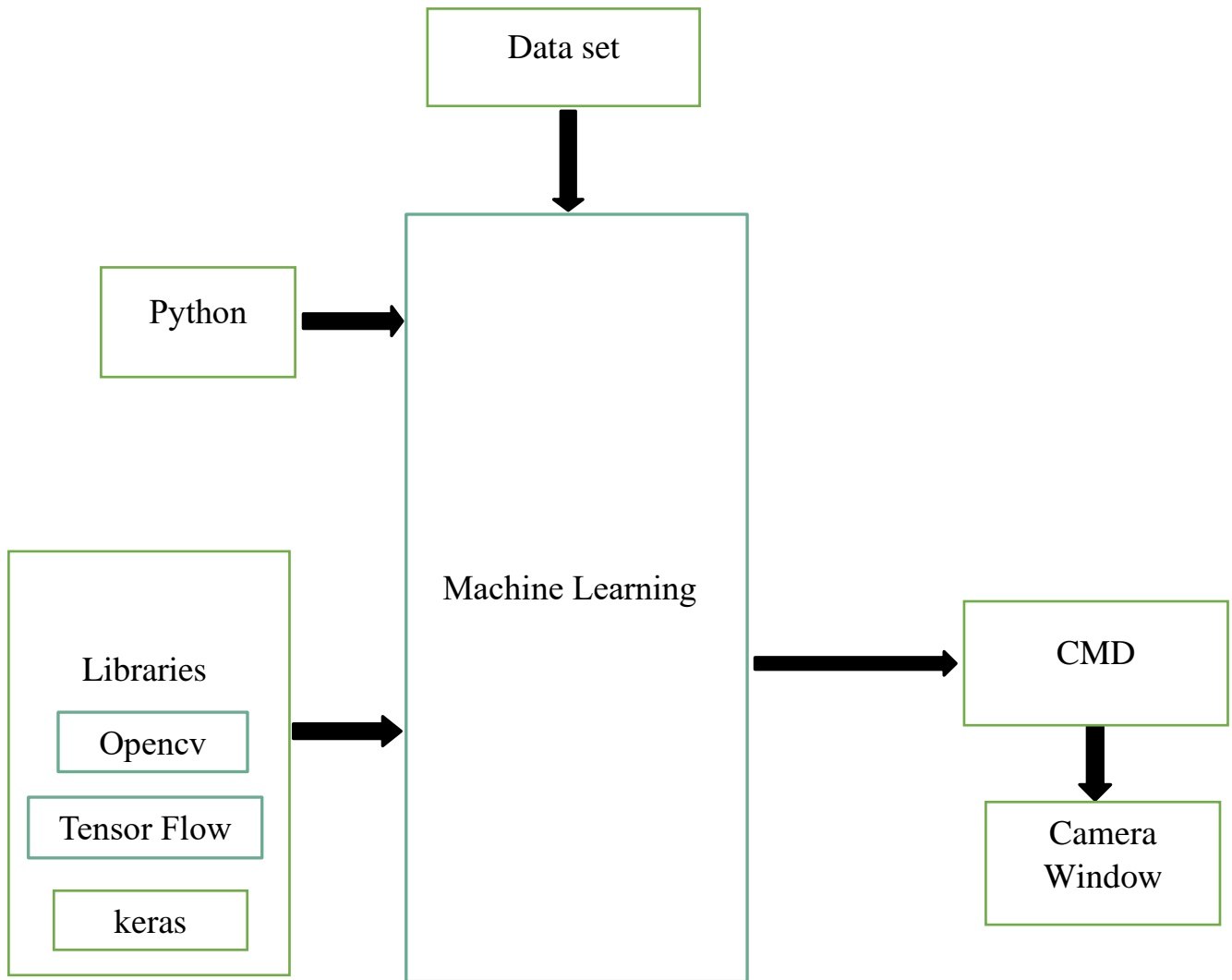


Fig .1 : Block diagram

5.2 Flow Chart :

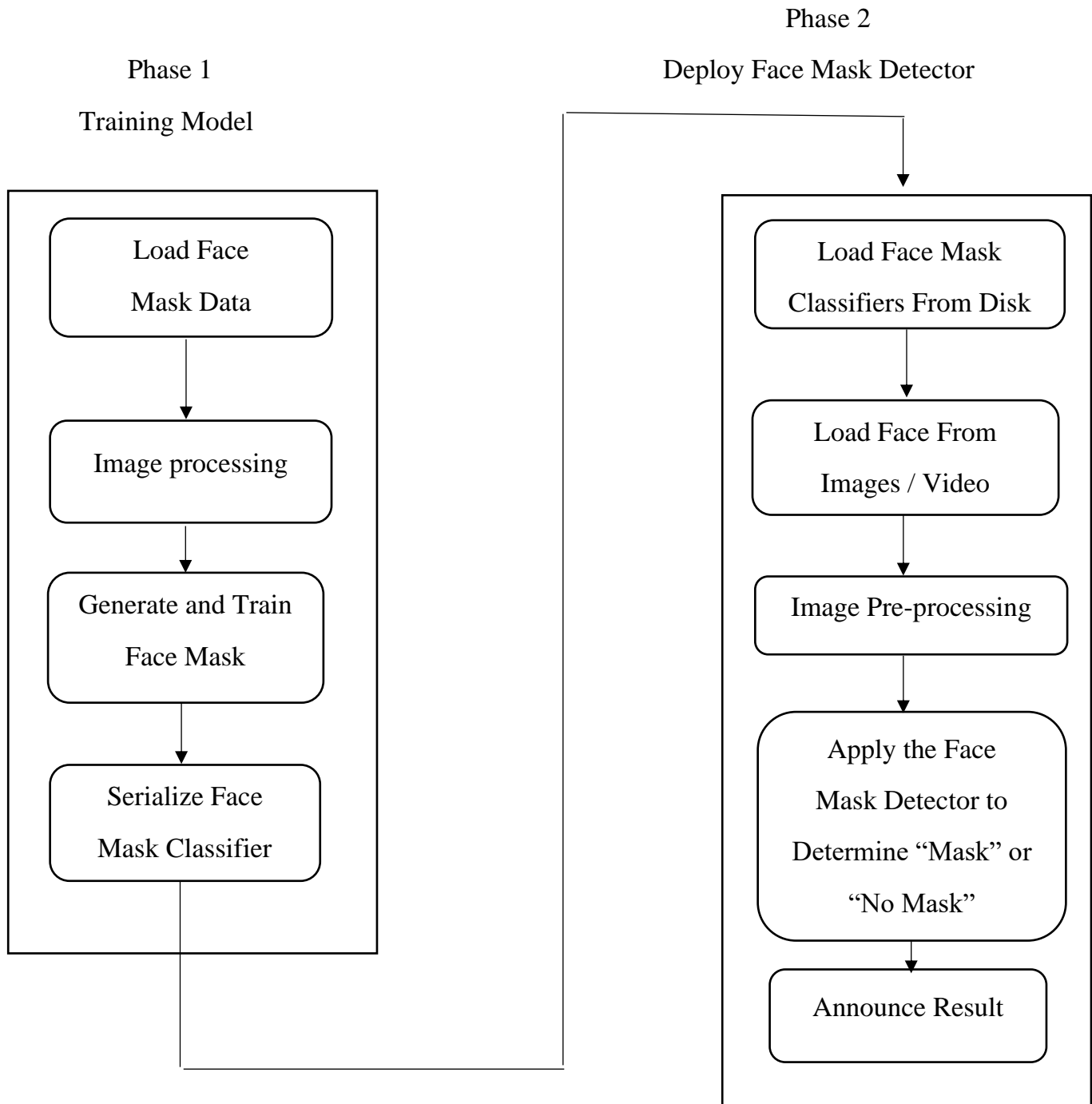


Fig 2 : Flow chart

Chapter 6

Software Requirements

6.1 Python

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.
- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

6.1.1 Introduction

Learning Python gives the programmer a wide variety of career paths to choose from. Python is an open-source (free) programming language that is used in web programming, data science, artificial intelligence, and many scientific applications. Learning Python allows the programmer to focus on solving problems, rather than focusing on syntax. Its relative size and simplified syntax give it an edge over languages like Java and C++, yet the abundance of libraries gives it the power needed to accomplish great things.

6.1.2 Python Downloading

You may want to print these instructions before proceeding, so that you can refer to them while downloading and installing Python. Or, just keep this document in your browser. You should read each step completely before performing the action that it describes.

This document shows downloading and installing Python 3.9.6 on Windows 10 in Summer 2021. You should download and install the latest version of Python. The current latest (as of Summer 2021) is Python 3.9.6.

Remember that you must install Python and then Eclipse as 64-bit applications.

Python: Version 3.9.6

The Python download requires about 25 Mb of disk space; keep it on your machine, in case you need to re-install Python. When installed, Python requires about an additional 90 Mb of disk space.

Downloading

1. Click windows.

The following page will appear in your browser.

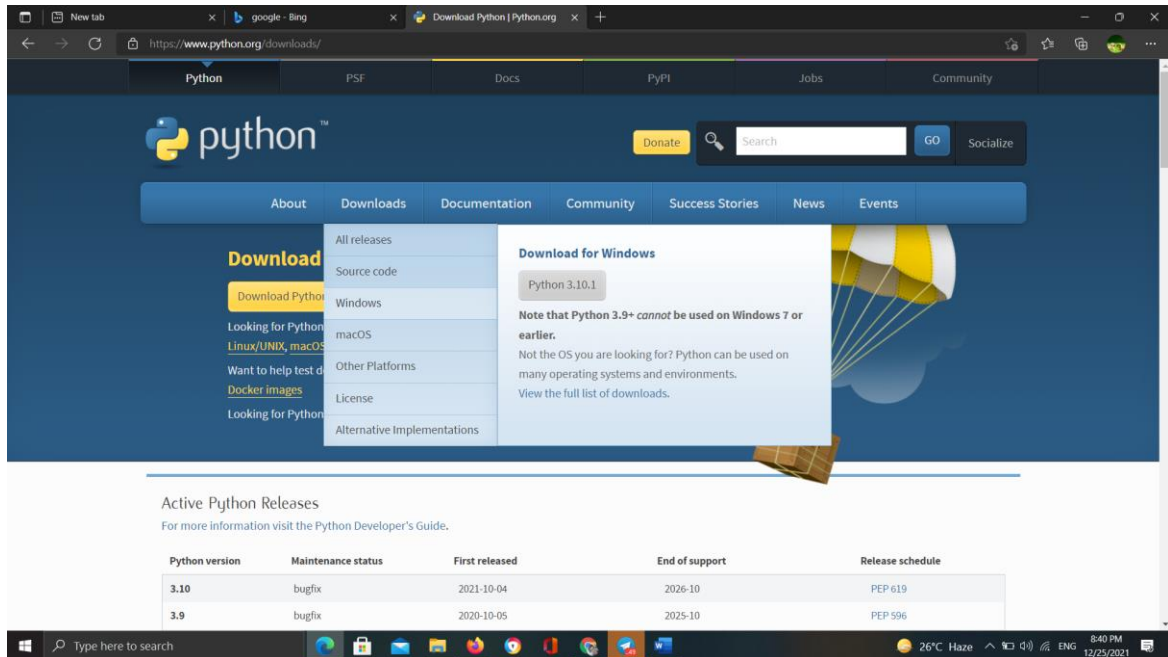


Fig 3 : python browser page

2. Click Python Download.

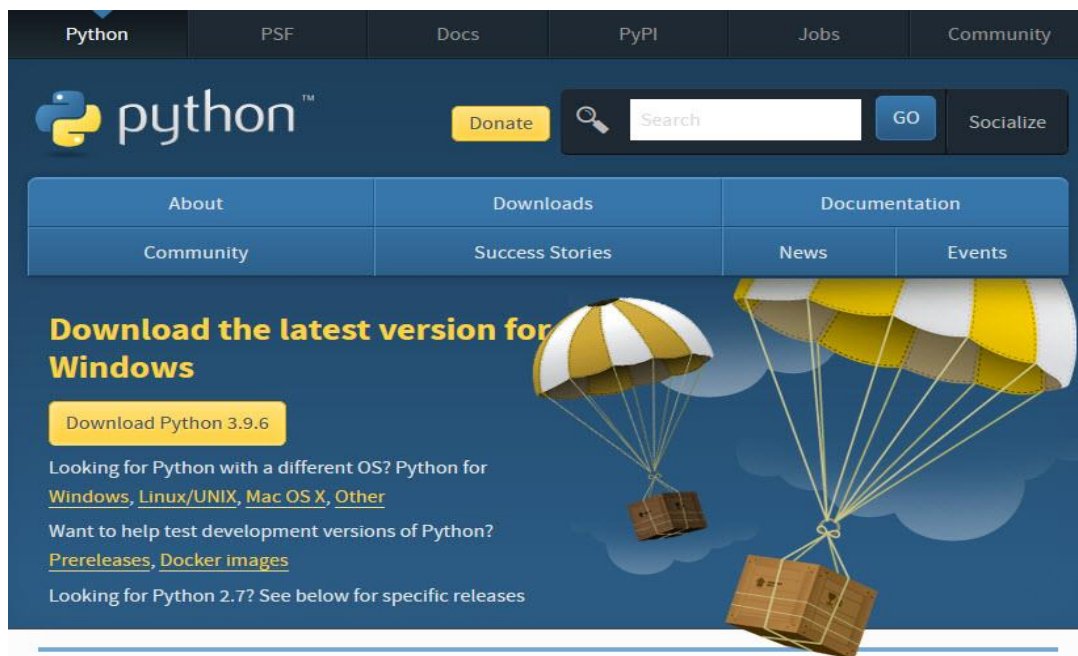


Fig 4 : Download specific version

3. Click the Download Python 3.9.6 button. The following pop-up window titled Opening python-3.96-amd64.exe will appear.



Fig 5 : saving python file

Click the Save File button.

The file named python-3.9.6-amd64.exe should start downloading into your standard download folder. This file is about 25 Mb so it might take a while to download fully if you are on a slow internet connection (it took me about 10 seconds over a cable modem).

The file should appear in your Downloads folder as



4. Move this file to a more permanent location, so that you can install Python (and reinstall it easily later, if necessary).
5. Feel free to explore this webpage further; if you want to just continue the installation, you can terminate the tab browsing this webpage.
6. Start the Installing instructions directly below.

6.1.3 Installing

1. Double-click the icon labeling the file python-3.9.6-amd64.exe.

A Python 3.9.6 (64-bit) Setup pop-up window will appear.

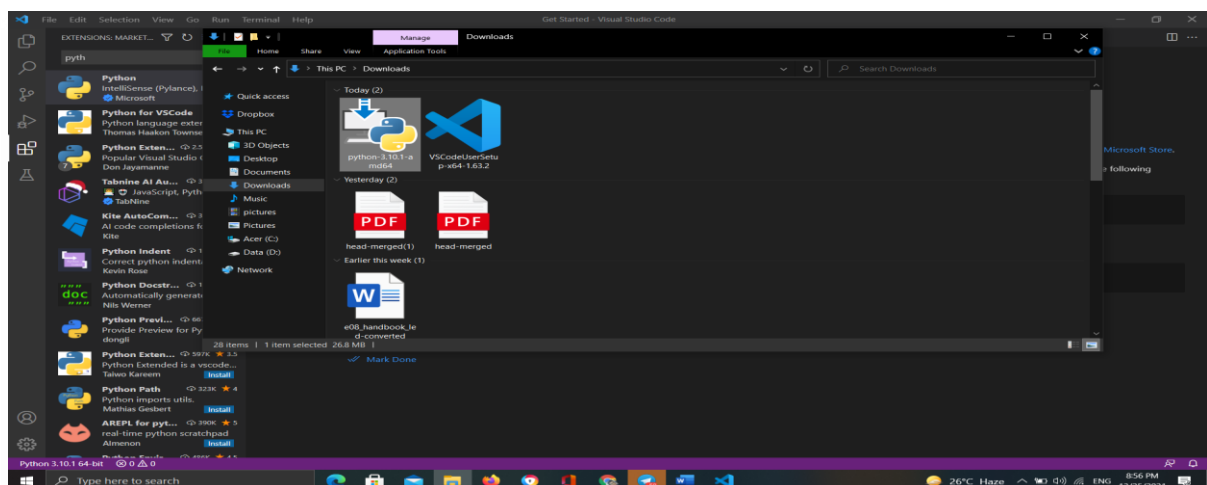


Fig 6 : Python file in window



Fig 7 : Installing python

Ensure that both the Install launcher for all users (recommended) and the Add Python 3.9 to PATH checkboxes at the bottom are checked: typically only first is checked by default.

If the Python Installer finds an earlier version of Python installed on your computer, the Install Now message may instead appear as Upgrade Now (and the checkboxes will not appear).

2. Highlight the Install Now (or Upgrade Now) message, and then click it.

When run, a User Account Control pop-up window may appear on your screen. I could not capture its image, but it asks, Do you want to allow this app to make changes to your device.

3. Click the Yes button.

A new Python 3.9.6 (64-bit) Setup pop-up window will appear with a Setup Progress message and a progress bar.

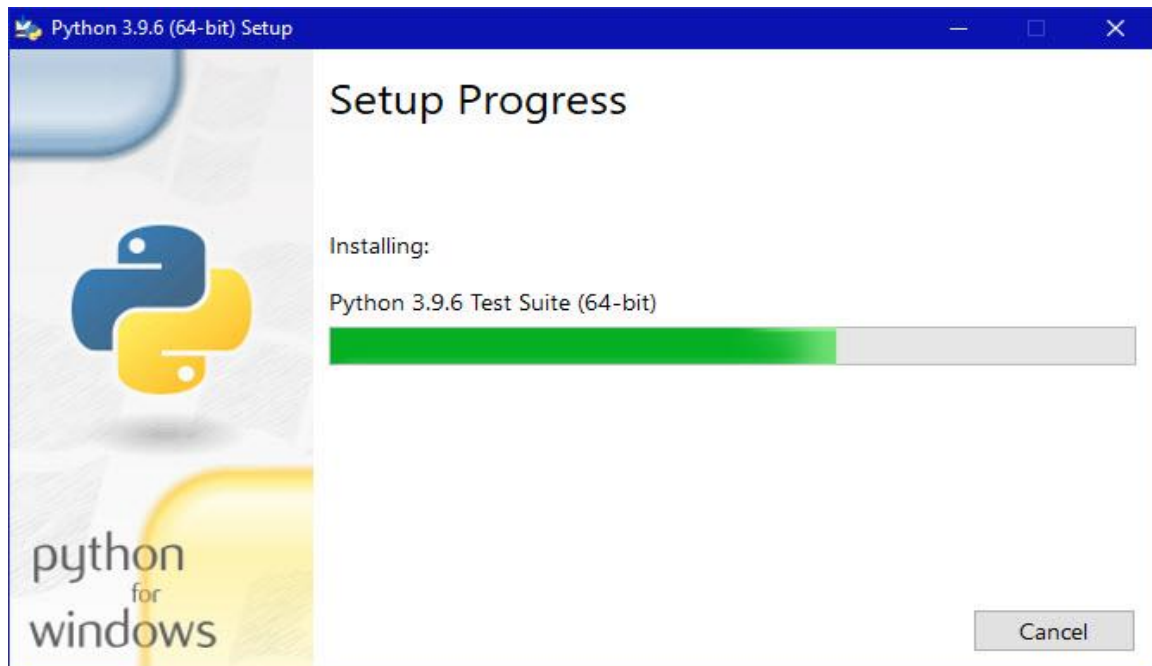


Fig 8 : Installing window

During installation, it will show the various components it is installing and move the progress bar towards completion. Soon, a new Python 3.9.6 (64-bit) Setup pop-up window will appear with a Setup was successfully message.

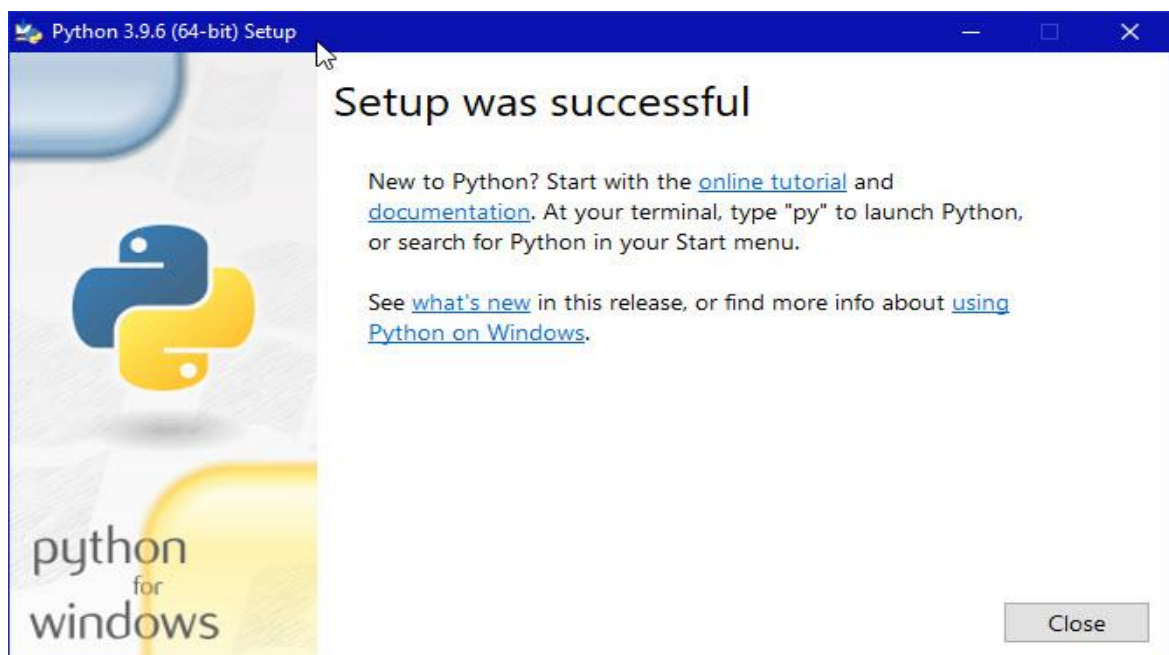


Fig 9 : Window of installed python

4. Click the Close button.

Python should now be installed.

6.1.4 Verifying

To try to verify installation,

1. Navigate to the directory `C:\Users\Pattis\AppData\Local\Programs\Python\Python39` (or to whatever directory Python was installed: see the pop-up window for Installing step 1).
2. Double-click the icon/file `python.exe`.

The following pop-up window will appear.

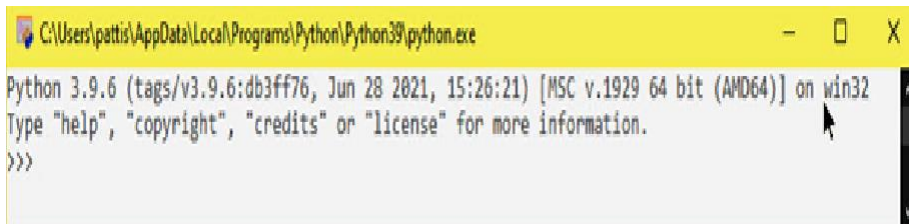


Fig 10 : Pop up window

A pop-up window with the title `C:\Users\Pattis\AppData\Local\Programs\Python\Python39\python.exe` appears, and inside the window; on the first line is the text `Python 3.9.6 ...` (notice that it should also say 64 bit). Inside the window, at the bottom left, is the prompt `>>>`: type `exit()` to this prompt and press enter to terminate Python.



Fig 11 : Checking for python version in command prompt

You should keep the file `python-3.9.6.exe` somewhere on your computer in case you need to reinstall Python (not likely necessary).

Chapter 7

Libraries

7.1 Command Prompt

Command Prompt is a command line interpreter application available in most Windows operating systems. It's used to execute entered commands. Most of those commands automate tasks via scripts and batch files, perform advanced administrative functions, and troubleshoot or solve certain kinds of Windows issues.

Command Prompt is officially called Windows Command Processor, but it's also sometimes referred to as the command shell or command prompt, or even by its filename, cmd.exe.

Command Prompt is sometimes incorrectly referred to as "the DOS prompt" or as MS-DOS. Command Prompt is a Windows program that emulates many of the command line abilities available in MS-DOS, but it's not MS-DOS.

Command is also an abbreviation for many other technology terms like centralized message distribution, colour monitor display, and common management database, but none of them have anything to do with Command Prompt.

7.1.1 Access of Command Prompt

There are several ways to open Command Prompt, but the "normal" method is via the Command Prompt shortcut located in the Start menu or on the Apps screen, depending on your version of Windows.

The shortcut is faster for most people, but another way to access Command Prompt is via the cmd Run command. You can also open cmd.exe from its original location:

Yet another method for opening Command Prompt in some versions of Windows is through the Power User Menu. However, you might see PowerShell there instead of Command Prompt depending on how your computer is set up. You can switch between Command Prompt and PowerShell from the Win+X menu.

Many commands can only be executed if you're running the Command Prompt as an administrator.

7.1.2 Using of Command Prompt

To use Command Prompt, you enter a valid Command Prompt command along with any optional parameters. Command Prompt then executes the command as entered and performs the task or function it's designed to perform in Windows.

For example, executing the following Command Prompt command in your Downloads folder would remove all MP3s from that folder. Commands must be entered into Command Prompt exactly. The wrong syntax or a misspelling could cause the command to fail or worse; it could

execute the wrong command or the right command in the wrong way. A comfort level with reading command syntax is recommended.

For example, executing the `dir` command will show a list of files and folders that exist at any specific location on the computer, but it doesn't actually do anything. However, change just a couple letters and it turns into the `del` command, which is how you delete files from Command Prompt!

Syntax is so important that with some commands, especially the delete command, adding even a single space can mean deleting entirely different data.

Here's an example where the space in the command breaks the line into two sections, essentially creating two commands where the files in the root folder (`files`) are deleted instead of the files in the subfolder (`music`).

The proper way to execute that command so as to remove files from the `music` folder instead is to remove the space so that the whole command is strung together correctly.

Don't let this scare you away from using Command Prompt commands, but definitely let it make you cautious.

7.2 Libraries Required

7.2.1 pip upgrade

Open the command prompt

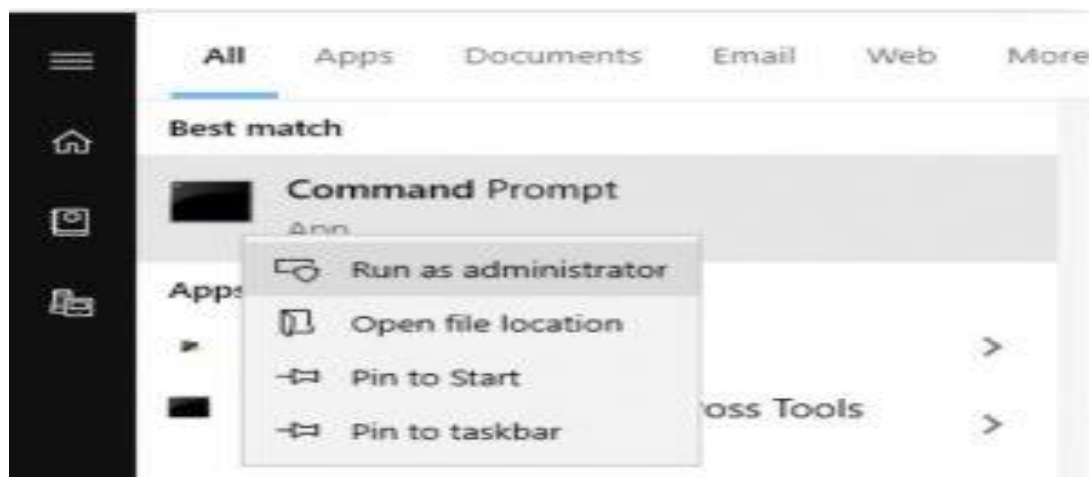


Fig 12 : Opening command prompt window

Then a command prompt should appear like this:

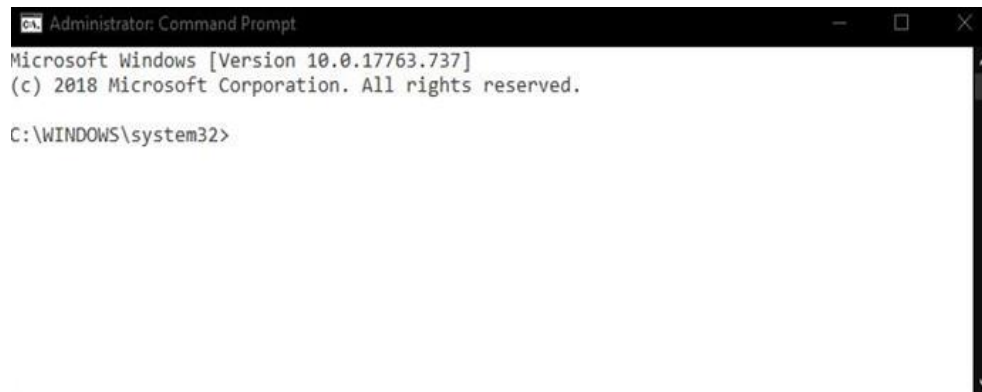


Fig 13 : Appearance of command prompt window

Easy method to upgrade PIP

1. Now that we have established your PIP version, you can easily upgrade to the latest version. With the Administrator : Command Prompt still open from the previous step, and still in the Python installation location given from the “where python” command, we can now type in the easy upgrade command and push ENTER:

```
python -m pip install --upgrade pip
```

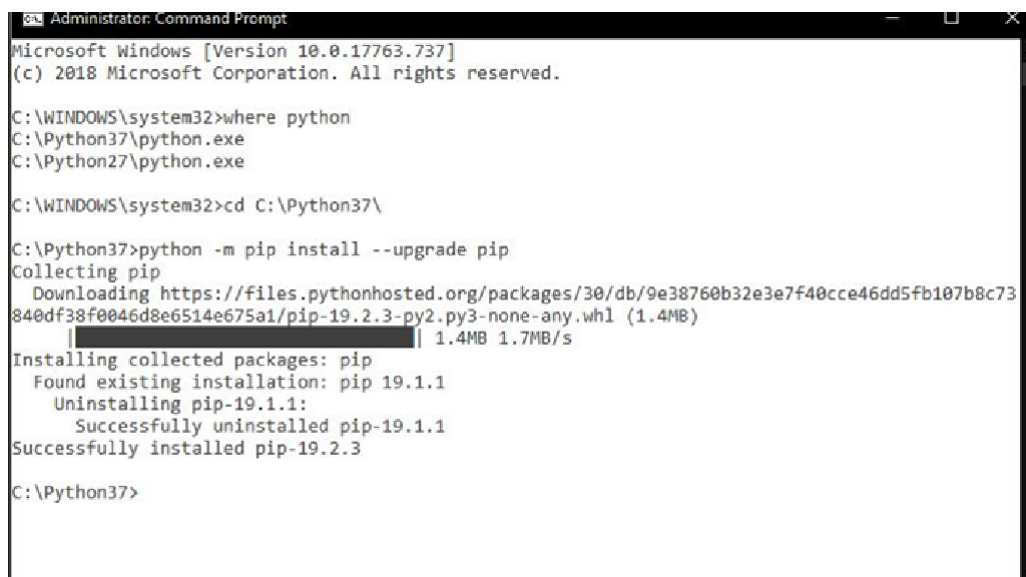


Fig 14 : Window showing installed pip

7.2.2 Numpy

1. Open the command prompt for the installation of numpy

```
C:\WINDOWS\system32>pip install numpy
Collecting pip
  Downloading pip-20.3.3-py2.py3-none-any.whl (1.5 MB)
    |██████████| 1.5 MB 251 kB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.2.3
    Uninstalling pip-20.2.3:
      Successfully uninstalled pip-20.2.3
  Successfully installed pip-20.3.3
C:\WINDOWS\system32>pip install numpy
```

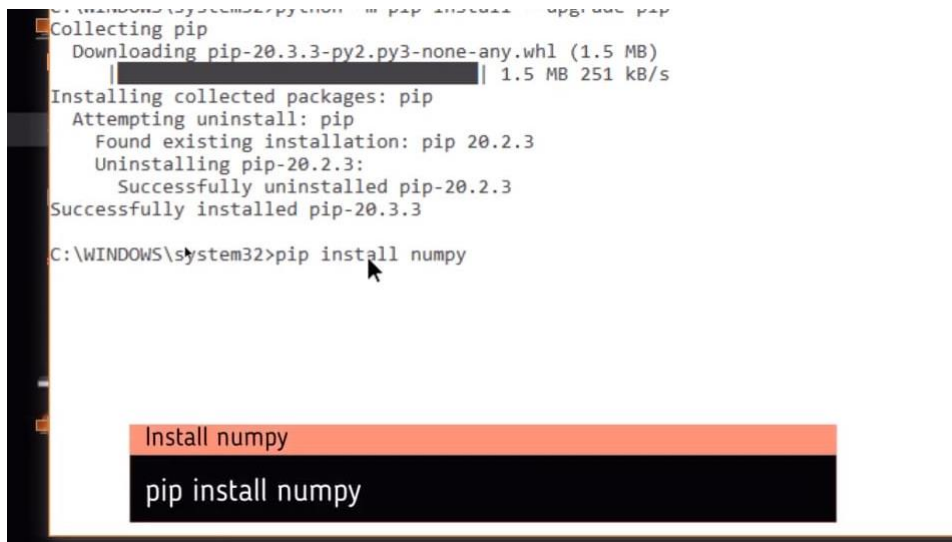


Fig 15 : Command prompt to install numpy

2. In the terminal, use the pip command to install numpy package.

```
C:\WINDOWS\system32>pip install numpy
Collecting pip
  Downloading pip-20.3.3-py2.py3-none-any.whl (1.5 MB)
    |██████████| 1.5 MB 251 kB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.2.3
    Uninstalling pip-20.2.3:
      Successfully uninstalled pip-20.2.3
  Successfully installed pip-20.3.3
C:\WINDOWS\system32>pip install numpy
Collecting numpy
  Downloading numpy-1.19.4-cp38-cp38-win_amd64.whl (13.0 MB)
    |██████████| 7.0 MB 22 kB/s eta 0:04:24
```

Fig 16 : Installing numpy in command prompt

3. Now the numpy is successfully installed

```

C:\WINDOWS\system32>python --version
Python 3.8.7

C:\WINDOWS\system32>python -m pip install --upgrade pip
Collecting pip
  Downloading pip-20.3.3-py2.py3-none-any.whl (1.5 MB)
    | 1.5 MB 251 kB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.2.3
    Uninstalling pip-20.2.3:
      *Successfully uninstalled pip-20.2.3
Successfully installed pip-20.3.3

C:\WINDOWS\system32>pip install numpy
Collecting numpy
  Downloading numpy-1.19.4-cp38-cp38-win_amd64.whl (13.0 MB)
    | 13.0 MB 384 kB/s
Installing collected packages: numpy
Successfully installed numpy-1.19.4

C:\WINDOWS\system32>

```

Fig 17 : Showing numpy installed in command prompt

7.2.3 OpenCV

1. Open the Command prompt to install OpenCV library .
2. In the terminal, use the pip command to install opencv_contrib_python_headless package.
3. Library will be downloaded successfully.

```

C:\WINDOWS\system32>pip install numpy
Collecting numpy
  Downloading numpy-1.19.4-cp38-cp38-win_amd64.whl (13.0 MB)
    | 13.0 MB 384 kB/s
Installing collected packages: numpy
Successfully installed numpy-1.19.4

C:\WINDOWS\system32>pip install opencv-contrib-python-headless
Collecting opencv-contrib-python-headless
  Downloading opencv_contrib_python_headless-4.5.1.48-cp38-cp38-win_amd64.whl (41
    | 41.2 MB 46 kB/s
Requirement already satisfied: numpy>=1.17.3 in e:\python38\lib\site-packages (fr
  9.4)
Installing collected packages: opencv-contrib-python-headless
Successfully installed opencv-contrib-python-headless-4.5.1.48

C:\WINDOWS\system32>

```

Fig 18 : Command prompt for installing OpenCV

- 4 .Check for the installed OpenCV

```
C:\WINDOWS\system32>pip install numpy
Collecting numpy
  Downloading numpy-1.19.4-cp38-cp38-win_amd64.whl (13.0 MB)
    | 13.0 MB 384 kB/s
Installing collected packages: numpy
Successfully installed numpy-1.19.4

C:\WINDOWS\system32>pip install opencv-contrib-python-headless
Collecting opencv-contrib-python-headless
  Downloading opencv_contrib_python_headless-4.5.1.48-cp38-cp38-win_amd64.whl (41
    | 41.2 MB 46 kB/s
Requirement already satisfied: numpy>=1.17.3 in e:\python38\lib\site-packages (fr
  9.4)
Installing collected packages: opencv-contrib-python-headless
Successfully installed opencv-contrib-python-headless-4.5.1.48

C:\WINDOWS\system32>
```

6. CHECK INSTALLED OPENCV
FACE MASK DETECTION IN WINDOWS

Fig 19 : Installed OpenCV in command prompt

7.2.4 Tensor flow

1. Open the Command prompt to install tensor flow library .
2. In the terminal, use the pip command to install tensor flow package.
3. library starts downloading.

```
Administrator: Command Prompt - pip install --upgrade tensorflow

C:\WINDOWS\system32>pip install --upgrade tensorflow
Collecting tensorflow
  Downloading tensorflow-2.4.0-cp38-cp38-win_amd64.whl (370.7 MB)
    | 370.7 MB 6.5 kB/s
WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)
ocolError('Connection aborted.', ConnectionResetError(10054, 'An existing connection
ost', None, 10054, None))': /simple/typing-extensions/
Requirement already satisfied: numpy~=1.19.2 in e:\python38\lib\site-packages (from
Collecting gast==0.3.3
  Downloading gast-0.3.3-py2.py3-none-any.whl (9.7 kB)
Collecting absl-py~=0.10
  Downloading absl_py-0.11.0-py3-none-any.whl (127 kB)
    | 127 kB 819 kB/s
Collecting astunparse~=1.6.3
  Downloading astunparse-1.6.3-py2.py3-none-any.whl (12 kB)
Collecting flatbuffers~=1.12.0
  Downloading flatbuffers-1.12-py2.py3-none-any.whl (15 kB)
Collecting google-pasta~=0.2
  Downloading google_pasta-0.2.0-py3-none-any.whl (57 kB)
    | 57 kB 185 kB/s
Collecting grpcio~=1.32.0
```

Fig 20 : Installing tensor flow in command prompt

4. Library will be downloaded successfully and check for installed tensor flow.


```
C:\WINDOWS\system32>pip install tensorflow
Collecting pip
  Downloading pip-20.3.3-py2.py3-none-any.whl (1.5 MB)
    | 1.5 MB 251 kB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.2.3
    Uninstalling pip-20.2.3:
      Successfully uninstalled pip-20.2.3
  Successfully installed pip-20.3.3
C:\WINDOWS\system32>pip install numpy
```

Fig 21 : Installed tensor flow

7.2.5 Imutils

1. Open the Command prompt to install Imutils library .
2. In the terminal, use the pip command to install tensor imutils package.
3. library starts downloading.
4. Library will be downloaded successfully and check for installed tensor flow

```
Administrator: Command Prompt
C:\WINDOWS\system32>pip install imutils
Collecting imutils
  Downloading imutils-0.5.3.tar.gz (17 kB)
Building wheels for collected packages: imutils
  Building wheel for imutils (setup.py) ... done
  Created wheel for imutils: filename=imutils-0.5.3-py3-none-any.whl size=25853 sha256
  ed2be3fe96f4ddf80e4ee5b5b1f5f5f
  Stored in directory: c:\users\yaser\appdata\local\pip\cache\wheels\c8\d6\0f\b0c3892
  2f943869
Successfully built imutils
Installing collected packages: imutils
Successfully installed imutils-0.5.3
C:\WINDOWS\system32>
```

Check installed tensorflow

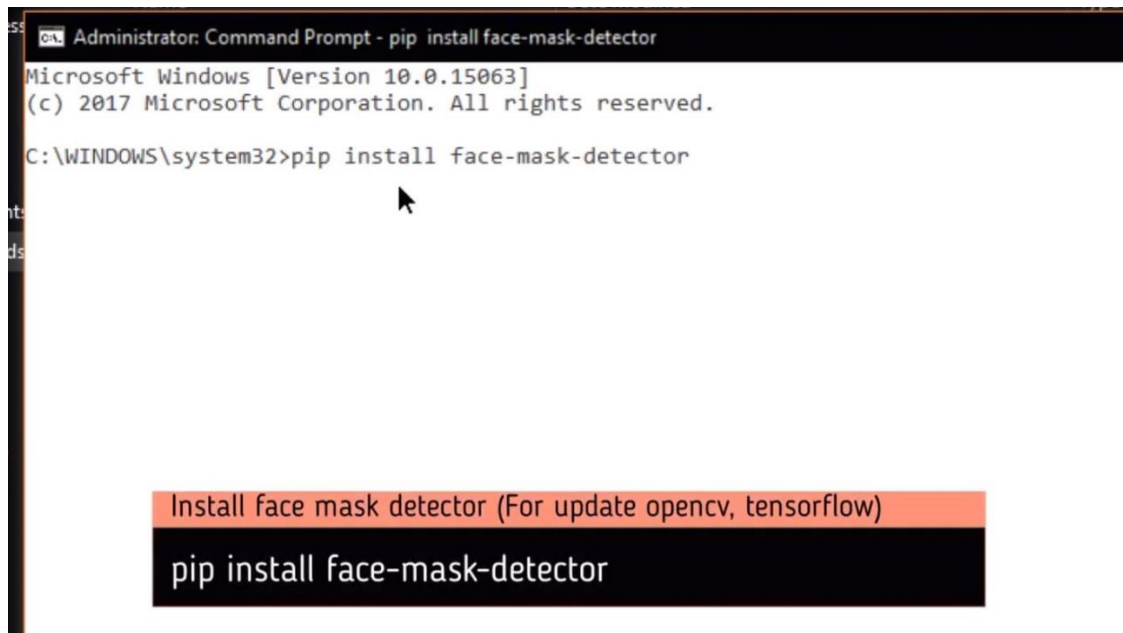
pip install imutils

Fig 22 : Installed imutils in command prompt

7.2.6 Face_mask_detector

1. Open the Command prompt to install **Face_mask_detector** library .

2. In the terminal, use the pip command to install tensor Face_mask_detector package.



```
Administrator: Command Prompt - pip install face-mask-detector
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

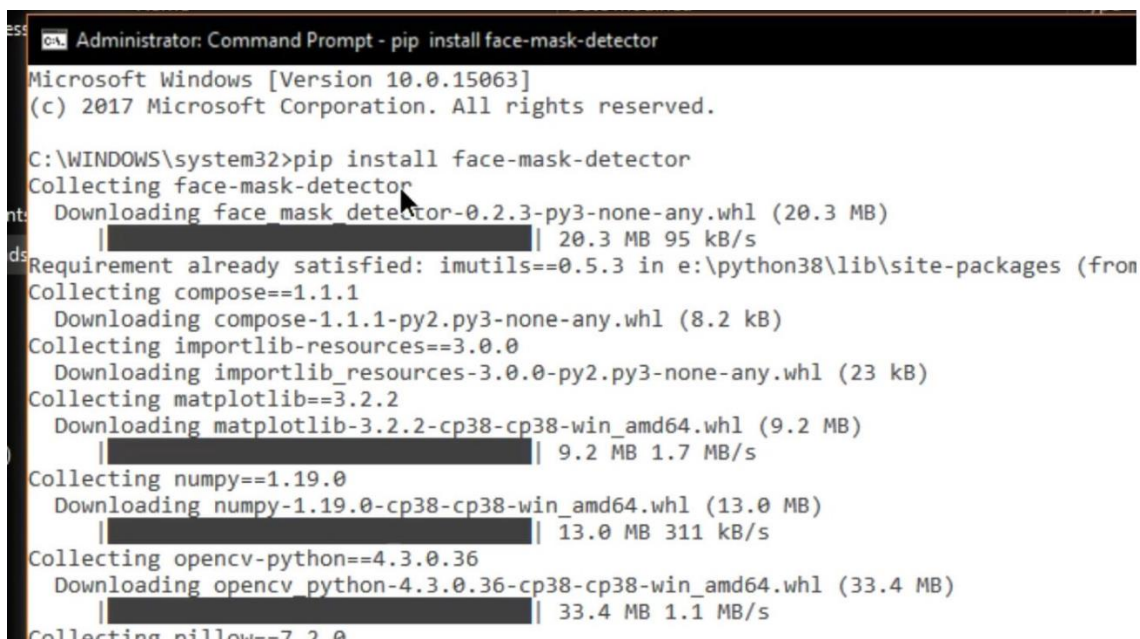
C:\WINDOWS\system32>pip install face-mask-detector
```

Install face mask detector (For update opencv, tensorflow)

```
pip install face-mask-detector
```

Fig 23 : Command Prompt for installing Face_mask_detector

3. library starts downloading.



```
Administrator: Command Prompt - pip install face-mask-detector
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>pip install face-mask-detector
Collecting face-mask-detector
  Downloading face_mask_detector-0.2.3-py3-none-any.whl (20.3 MB)
    | 20.3 MB 95 kB/s
Requirement already satisfied: imutils==0.5.3 in e:\python38\lib\site-packages (from
Collecting compose==1.1.1
  Downloading compose-1.1.1-py2.py3-none-any.whl (8.2 kB)
Collecting importlib-resources==3.0.0
  Downloading importlib_resources-3.0.0-py2.py3-none-any.whl (23 kB)
Collecting matplotlib==3.2.2
  Downloading matplotlib-3.2.2-cp38-cp38-win_amd64.whl (9.2 MB)
    | 9.2 MB 1.7 MB/s
Collecting numpy==1.19.0
  Downloading numpy-1.19.0-cp38-cp38-win_amd64.whl (13.0 MB)
    | 13.0 MB 311 kB/s
Collecting opencv-python==4.3.0.36
  Downloading opencv_python-4.3.0.36-cp38-cp38-win_amd64.whl (33.4 MB)
    | 33.4 MB 1.1 MB/s
Collecting pillow==7.2.0
```

Fig 24 : Installing Face_mask_detector library

4. Library will be downloaded successfully and check for installed tensor flow .



```
Administrator: Command Prompt - pip install --upgrade tensorflow
Collecting werkzeug>=0.11.15
  Downloading Werkzeug-1.0.1-py2.py3-none-any.whl (298 kB)
    | 136 kB 504 kB/s
Collecting wheel~=0.35
  Downloading wheel-0.36.2-py2.py3-none-any.whl (35 kB)
    | 298 kB 344 kB/s
Collecting wrapt~=1.12.1
  Downloading wrapt-1.12.1.tar.gz (27 kB)
Using legacy 'setup.py install' for termcolor, since package 'wheel' is not installed
Using legacy 'setup.py install' for wrapt, since package 'wheel' is not installed.
Installing collected packages: urllib3, pyasn1, idna, chardet, certifi, six, rsa, re
chertools, requests-oauthlib, google-auth, wheel, werkzeug, tensorboard-plugin-wit, p
th-oauthlib, absl-py, wrapt, typing-extensions, termcolor, tensorflow-estimator, ten
ssing, h5py, google-pasta, gast, flatbuffers, astunparse, tensorflow
  Running setup.py install for wrapt ... done
  Running setup.py install for termcolor ... done
```

Fig 25 : Installed Face_mask_detector library

Chapter 8

Programs

8.1 Source code for training :

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.utils import to_categorical
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
import os

# initialize the initial learning rate, number of epochs to train for,
# and batch size
INIT_LR = 1e-4
EPOCHS = 20
BS = 32

DIRECTORY = r"C:\Users\bhanu\OneDrive\Documents\Desktop\harsha\Face-Mask-Detection-master\dataset"
CATEGORIES = ["with_mask", "without_mask"]

# grab the list of images in our dataset directory, then initialize
# the list of data (i.e., images) and class images
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)

# perform one-hot encoding on the labels
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)

# construct the training image generator for data augmentation
aug = ImageDataGenerator(
    rotation_range=20,
    zoom_range=0.15,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.15,
    horizontal_flip=True,
    fill_mode="nearest")

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

# place the head FC model on top of the base model (this will become
# the actual model we will train)
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
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
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_))

# serialize the model to disk
print("[INFO] saving mask detector model...")
model.save("mask_detector.model", save_format="h5")

# plot the training loss and accuracy
N = EPOCHS
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), H.history["accuracy"], label="train_acc")
plt.plot(np.arange(0, N), H.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.savefig("plot.png")

```

8.2 Source code for testing :

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

sound = 5;
count = 0;

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

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

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

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

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

            # ensure the bounding boxes fall within the dimensions of
```

```

# the frame
(startX, startY) = (max(0, startX), max(0, startY))
(endX, endY) = (min(w - 1, endX), min(h - 1, endY))

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

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

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

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

# load our serialized face detector model from disk
prototxtPath = r"face_detector\deploy.prototxt"
weightsPath = r"face_detector\res10_300x300_ssd_iter_140000.caffemodel"
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

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

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

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

    # detect faces in the frame and determine if they are wearing a

```

```

# face mask or not
(locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)

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

    # determine the class label and color we'll use to draw
    # the bounding box and text
    label = "Mask" if mask > withoutMask else "No Mask"
    color = (0, 255, 0) if label == "Mask" else (0, 0, 255)
    sound = 1 if label == "Mask" else 0

    # 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)

count=(count+1)%20

# show the output frame
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF
if count==19:
    if sound == 1:
        playsound("D:\getIn.wav")
    if sound == 0:
        playsound("D:\wearMask.wav")
else:
    sound=5
# if the `q` key was pressed, break from the loop
if key == ord("q"):
    break

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

```


8.3 DATASET :

Dataset Visualization For the face mask detection, only a few datasets are available and most of them are either created artificially or full of noise with wrong labels. Hence, we created a practical dataset to train the model which yielded fruitful results at the end. The dataset comprising of images that are obtained from abundant data sources including MAFA, RMFD, CelebA and some images are downloaded from internet, cleaned the dataset by manual inspection. Finally, a dataset having 4000 images was created, with the label “with_mask” and “without_mask” and the distribution is visualized in Figure 1.

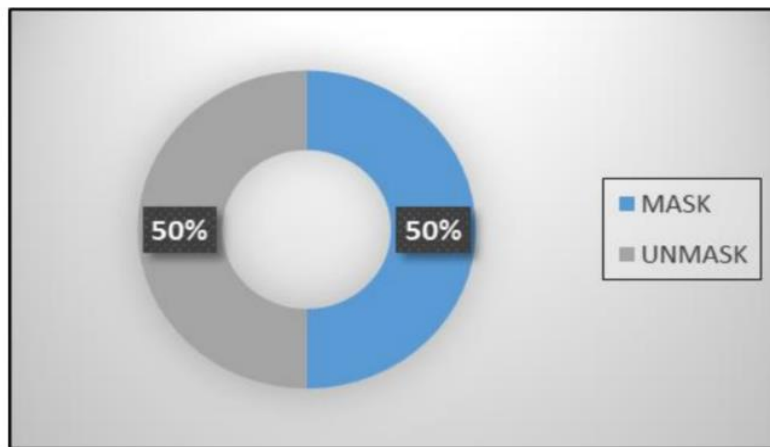


Fig 26 : Dataset Visualization

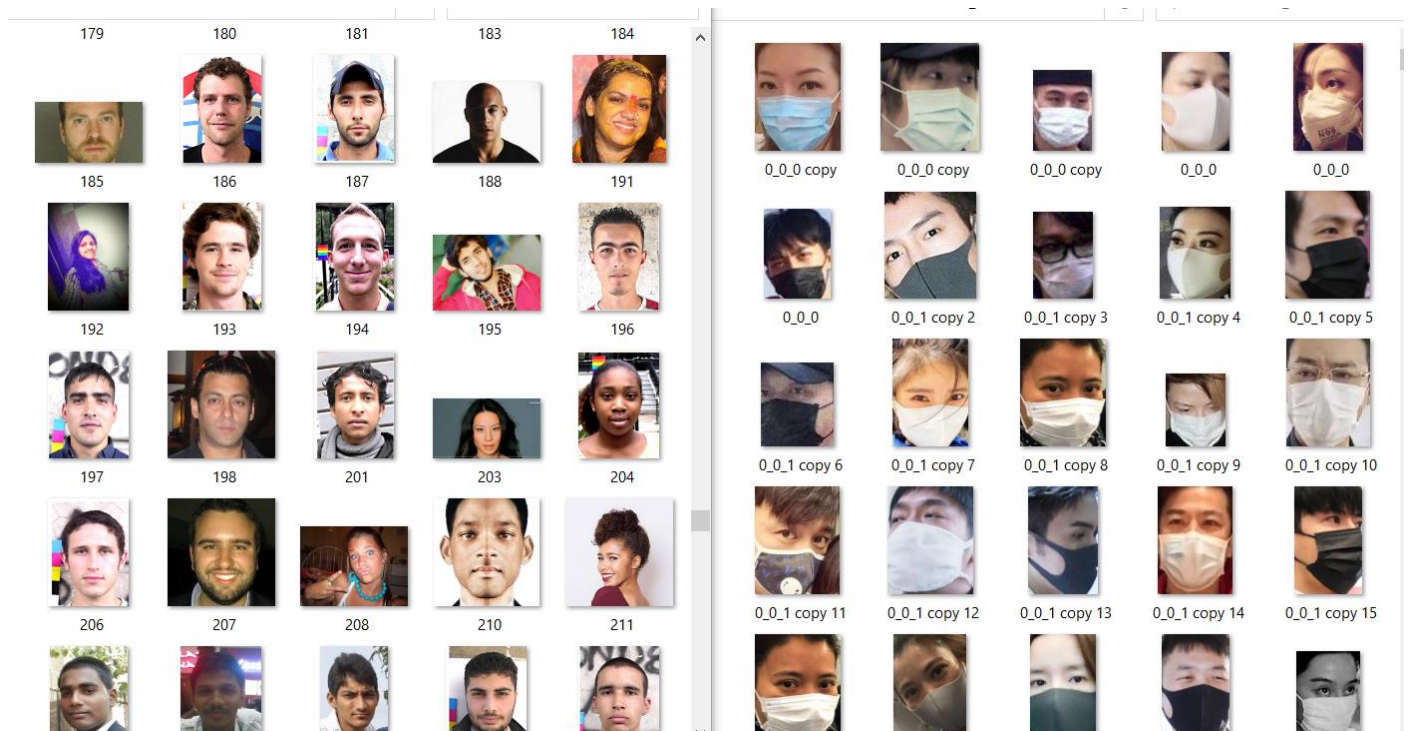


Fig 27 : Data set of mask and no mask

Chapter 9

Working of the system

9.1 Working principle :

In this project, we have developed a deep learning model for face mask detection using Python, Keras, and OpenCV. We developed the face mask detector model for detecting whether person is wearing a mask or not. We have trained the model using Keras with network architecture. Training the model is the first part of this project and testing using webcam using OpenCV is the second part.

The initial stage of the process starts with training the model using the appropriate dataset to predict in real time whether an individual is wearing mask or not. After the classifier is trained, a precise face detector model is essential for detection the faces so that it can classify masked and unmasked faces. For this task, an OpenCV's Single Shot Multibox Detector with ResNet-10 as its backbone, helps in real time detection of faces. The classifier uses MobileNetV2 pre-trained model to do predictions. The proposed model's work flow is illustrated in Figure 2.

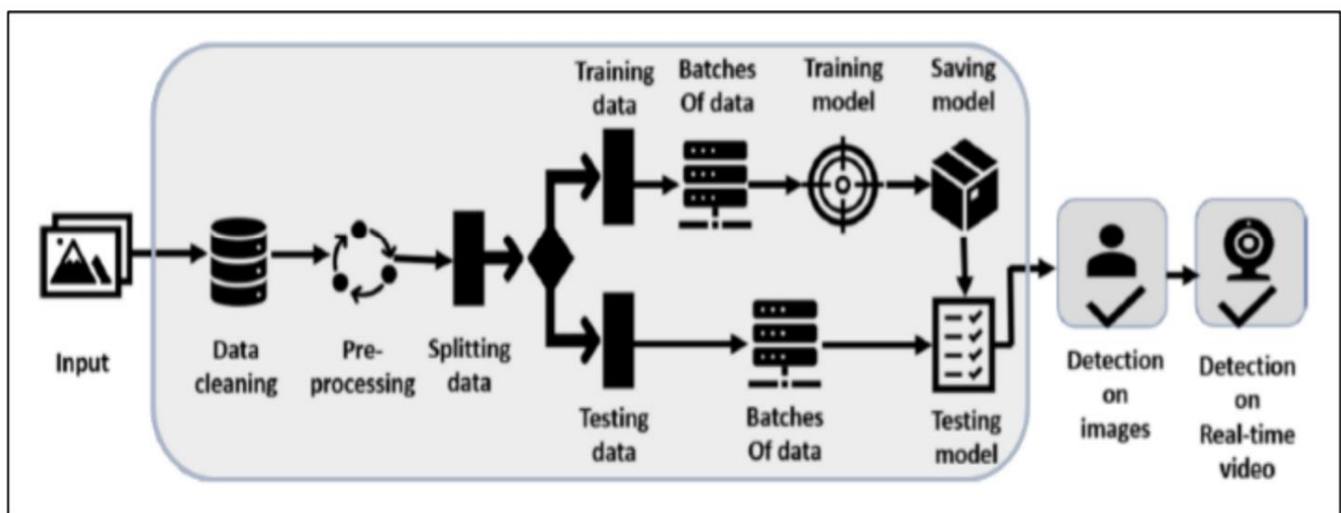


Fig 28 : Working of the system

The following table shows the percentages that are given to different classification of mask

	precision	recall	f1-score	support
with mask	0.95	0.99	0.97	500
without mask	0.99	0.95	0.97	500
accuracy			0.97	1000
macro avg	0.97	0.97	0.97	1000
weighted avg	0.97	0.97	0.97	1000

Table 1 : Percentage values for different classification of mask

In the given below figure depicts the predictions made on few images by the proposed model. The green rectangular box with an accuracy score depicts that the person is wearing mask, while the rectangular box with an accuracy score depicts that the person is no wearing masks.

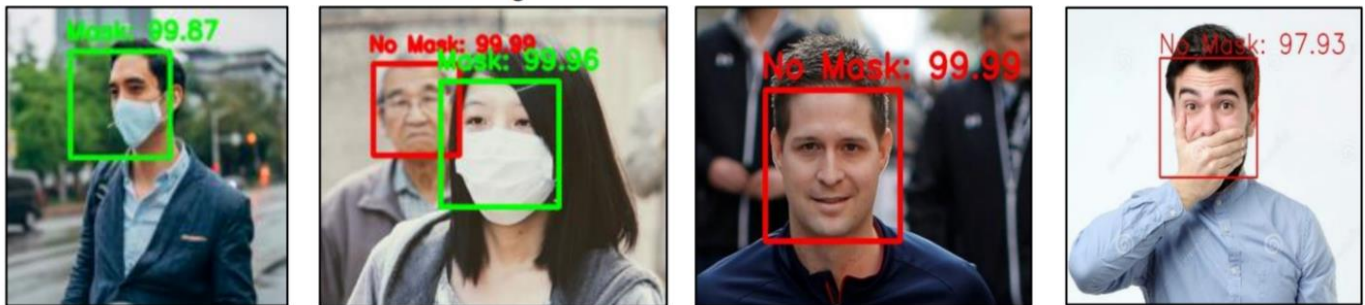


Fig 29 : Detecting mask and no mask with percentage

Chapter 10

Result Analysis

10.1 Results

To conduct experimental trails we have used a laptop customized with AMD Ryzen 5 3550H processor with 8GB RAM. The command prompt with Python 3.8.5 kernel is employed for the experimental trail development and implementation

1. Detecting the person face and indicating with red rectangle box that he is not wearing mask along with mask percentage.

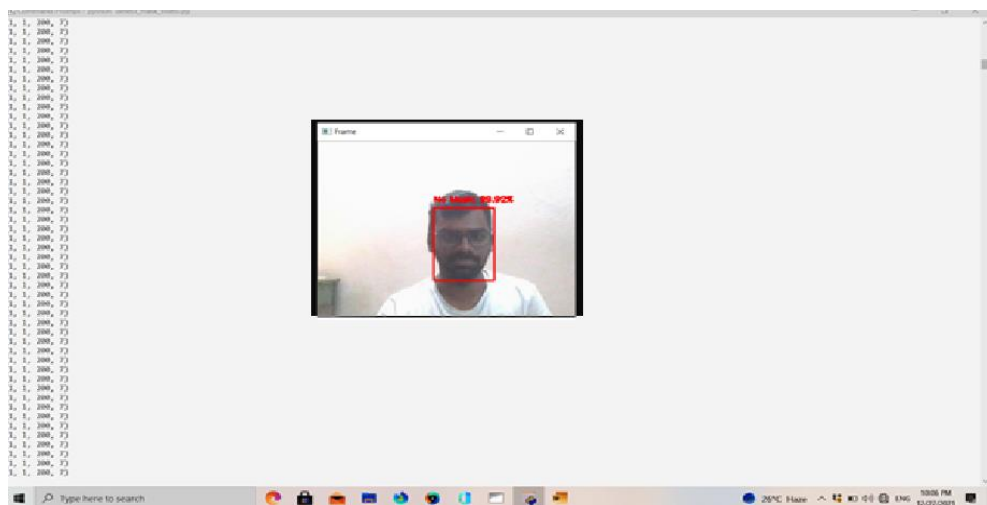


Fig 30 : Result with no mask

2. Detecting the person face and indicating with green rectangle box that he is wearing mask along with mask percentage.

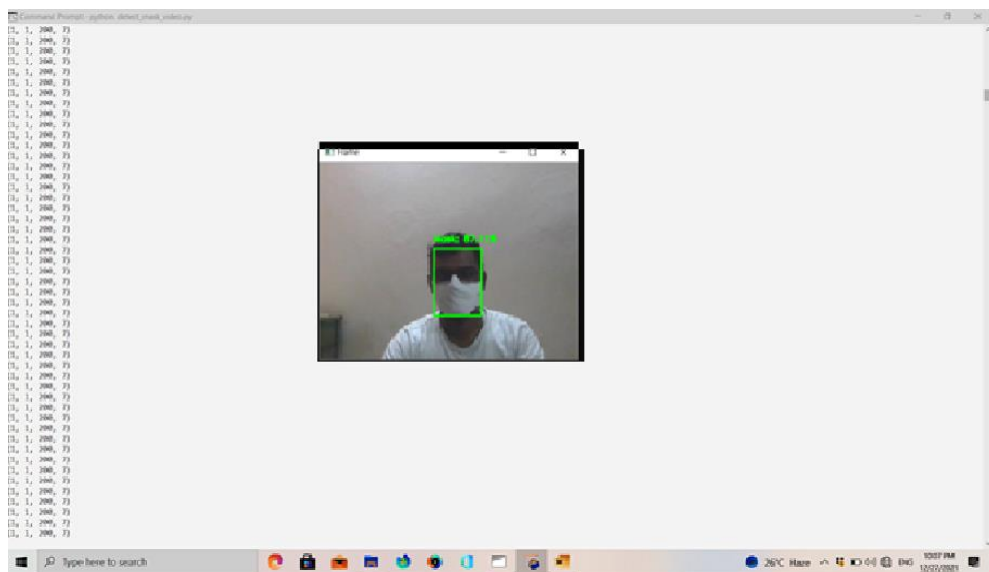


Fig 31 : Result with mask

3. Detecting the multiple faces and indicating with red rectangle box that they are not wearing mask along with mask percentage.

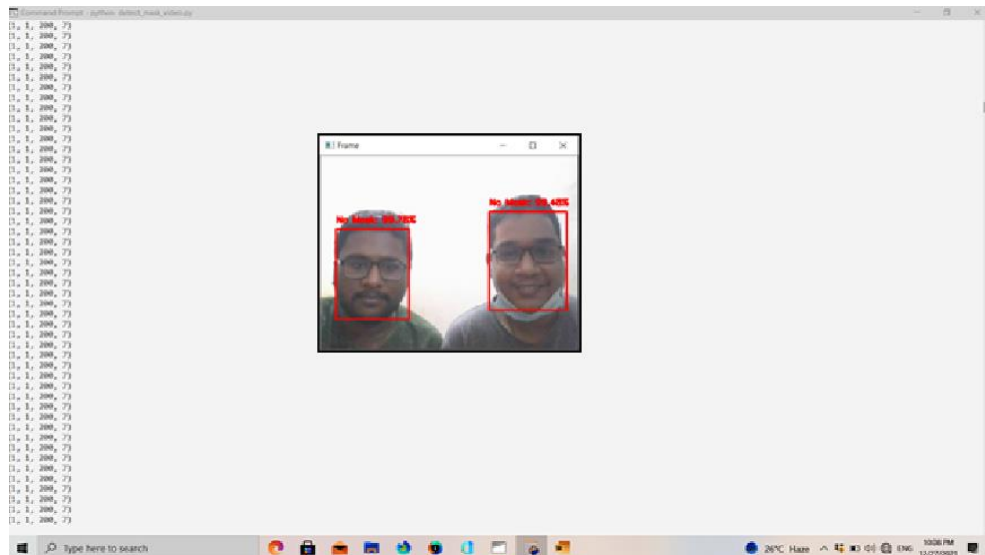


Fig 32 : Result with no mask for multiple faces

4. Detecting the multiple faces and indicating with green rectangle box that they are not wearing mask along with mask percentage.

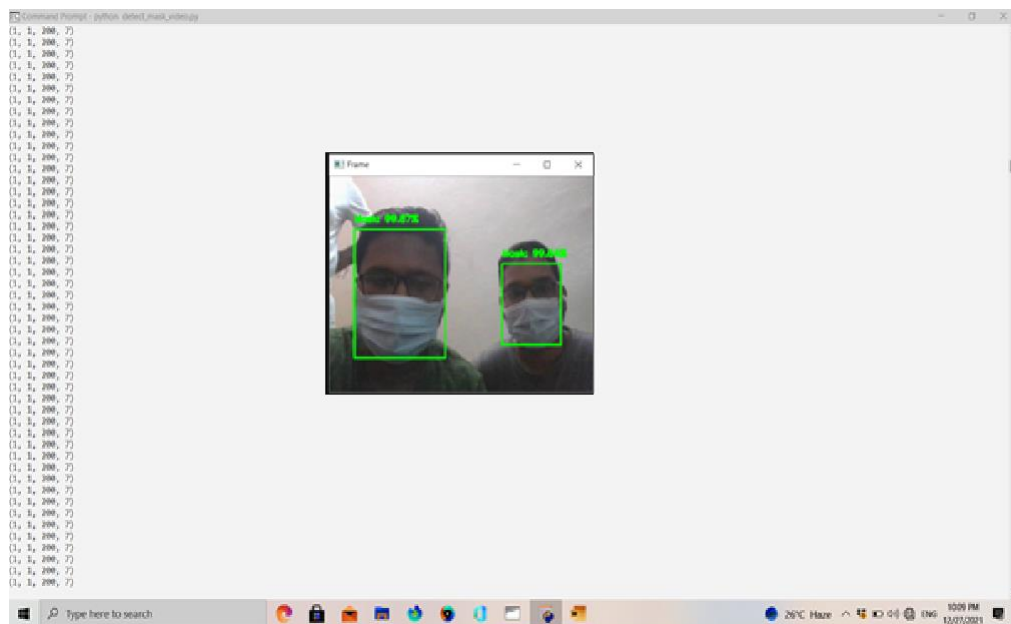


Fig 33 : Result with mask for multiple faces

5. Detecting the multiple faces and classifying that who is wearing mask and who are not with green and red colour rectangular boxes along with mask percentage.

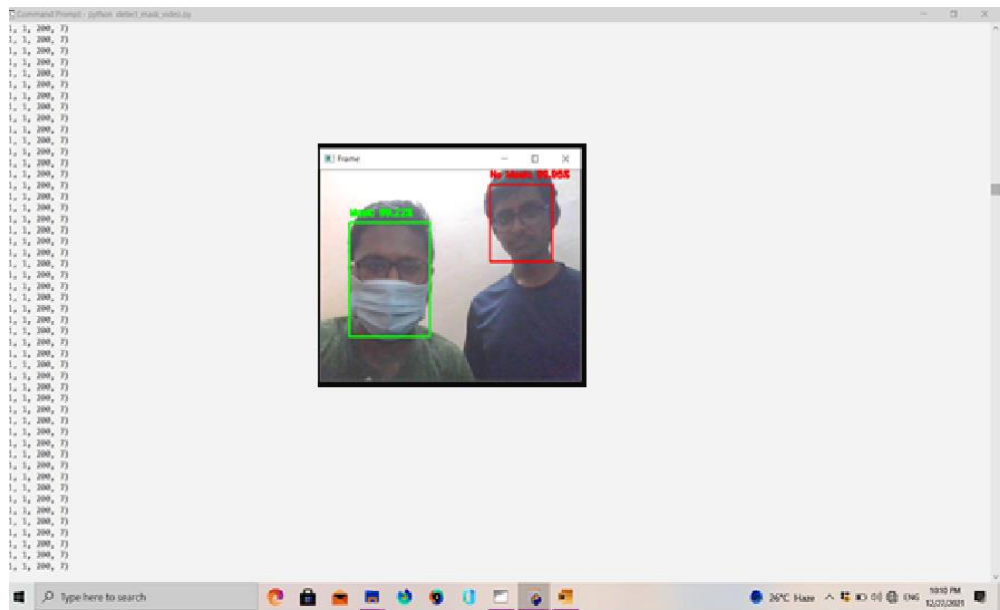


Fig 34 : Result of detecting and classifying with mask and no mask

6. Detecting the multiple faces and its is indicating with rectangular red boxes for improper mask wearing along with mask percentage.

7. It is specifying even person wearing cloth that to wear a original mask

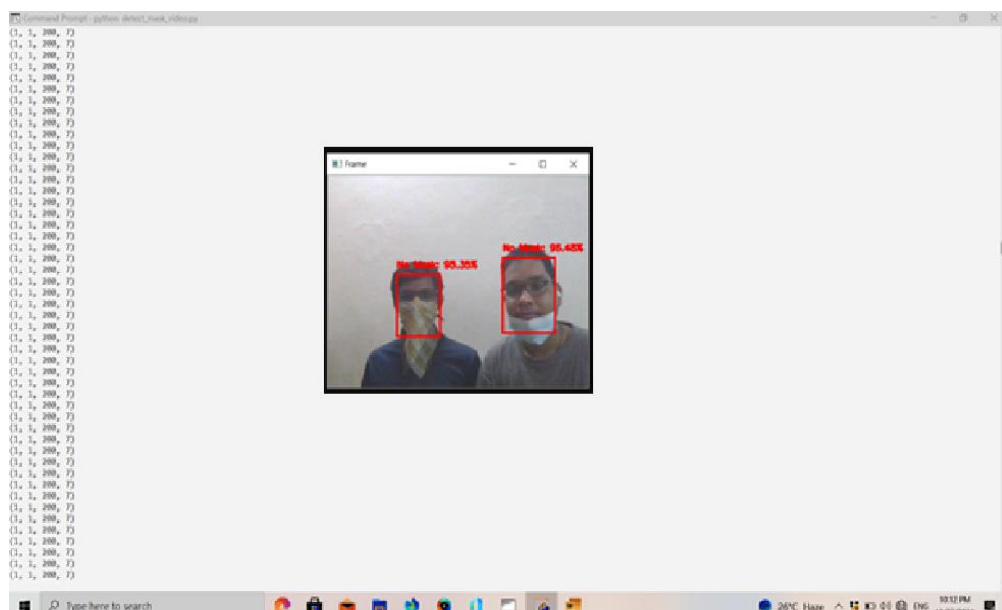


Fig 35 : Result for detecting all improper way of mask

Chapter 11

Conclusion And Future Enhancement

11.1 Conclusion :

In this project , our proposed method which is developed using OpenCV and Machine Learning with image classifier handled the limitations of the previous work and yielded fruitful results. The issues of wrong predictions have been successfully removed and thus we can notice that the model's accuracy has increased. This automated system will operate in an efficient manner to track people in public places where crowd monitoring is necessary. The new practical dataset created can be used for further advanced models for thermal screening and facial recognition.

11.2 Future Enhancement :

Deep learning is an important breakthrough in the AI field. It has recently shown enormous potential for extracting tiny features in image analysis. Due to the COVID-19 epidemic, some deep learning approaches have been proposed to detect patients infected with coronavirus. In this context, and unlike bacterial pneumonia, many other types of lung infections caused by viruses are called viral pneumonia. These viruses, such as the COVID-19, infect the lungs by blocking the oxygen flow, which can be life-threatening. This motivated researchers to develop many frameworks and schemes based on AI tools in the fight against this dangerous virus.

Some of the ideas that are useful :

1. Deep Learning Tools and CXR Image-Based COVID-19 Detection
2. Deep Learning Tools and CT Image-Based COVID-19 Detection
3. MobileNet Mask Model

References :

1. Paul Viola and Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, February – 2001, pp.I-511.
2. Shou Yang, Ping Luo, Chen Change Loy, Xiaoou Tang, "From Facial Parts Responses to Face Detection: A Deep Learning Approach", www.arXiv.org, September – 2015, arXiv: 1509.06451.
3. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", Advances in Neural Information Processing Systems, June - 2015, pp. 91 - 99.
4. Michael Optiz, Georg Waltner, Georg Poier, Horst Possegger and Horst Bischof, "Grid Loss: Detecting Occluded Faces", European Conference on Computer Vision, October - 2016, pp.386 - 402.
5. Mingjie Jiang, Xinqi Fan and Hong Yan, "RetinaMask: A Face Mask Detector", .org, May - 2020, arXiv: 2005.03950. IJARCCCE International Journal of Advanced Research in Computer and Communication Engineering Vol. 10, Issue 7, July 2021 DOI 10.17148/IJARCCCE.2021.10785 © IJARCCCE This work is licensed under a Creative Commons Attribution 4.0 International License 429 ISSN (O) 2278-1021, ISSN (P) 2319-5940
6. Mohamed Loey, Gunasekaran Manogaran, Mohamed Hamed N Taha and Nour Eldeen M Khalifad, "A Hybrid Deep Transfer Learning Model with Machine Learning methods for Face Mask Detection in the era of the COVID-19 Pandemic", Measurement, Jan - 2021, Vol.178, ArtNo.108288.
7. Sammy V. Militante and Nanette V. Dionisio, "Real-time Facemask Recognition with Alarm system using Machine Learning", 2020 11th IEEE Control and System Graduate Research Colloquium (ICSGRC), August – 2020, pp. 106 - 110.
8. Hoanh Nguyen, "Fast Object Detection Framework based on MobileNetV2 Architecture and Enhanced Feature Pyramid", Journal of Theoretical and Applied Information Technology, March – 2020, Vol.98, No.05.
9. Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt and Gang Hua, "A Convolutional Neural Network Cascade for Face Detection", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, June - 2015, pp.5325 - 5334.
10. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp.4510 - 4520.
11. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu and Alexander C Berg, "SSD: Single Shot MultiBox Detector", European Conference on Computer Vision, Springer, 2016, pp. 21-37.
12. Mohamed Loey, Gunasekaran Manogaran, Mohamed Hamed N Taha and Nour Eldeen M Khalifad, "Fighting against COVID-19: A Novel Machine Learning model based on YOLO-V2 with ResNet-50 for Medical Mask Detection", Sustainable Cities and Society, February - 2021, Vol. 65, Art No. 102600.

13. Bosheng Qin and Dongxiao Li, “Identifying Facemask-Wearing Condition Using Image Super-Resolution with Classification Network to Prevent COVID-19”, Sensors, September - 2020, Vol.20, No.18, pp.5236.
14. Zhongyuan Wang, Guangcheng Wang, Baojin Huang, Zhangyang Xiong,, Qi Hong, Hao Wu, Peng Yi, Kui Jiang, Nanxi Wang, Yingjiao Pei, Heling Chen, Yu Miao, Zhibing Huang and Jinbi Liang, “Masked Face Recognition Dataset and Application”, arXiv.org, March - 2020, preprint arXiv: 2003.09093.
15. Rajeev Ranjan, Vishal M Patel and Rama Chellappa, “Hyperface: A Deep Multi-Task Learning Framework for Face Detection, Landmark Localization, Pose Estimation, and Gender Recognition”, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), December – 2017, pp. 121 - 135.