A Project Report

on

AN AUTOMATED SYSTEM TO LIMIT COVID-19 USING FACIAL MASK DETECTION IN A SMART CITY NETWORK

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

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

17WH1A0542	Ms. K.SUMAVALLIKA
17WH1A0551	Ms. B.PRANEETHA
17WH1A0557	Ms. B.VARSHITHA VEENA

under the esteemed guidance of

Mr.K.RAJESH

Assistant Professor

Department of Computer Science and Engineering

BVRIT HYDERABAD

College of Engineering for Women

(NBA Accredited – EEE, ECE, CSE and IT)

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Bachupally, Hyderabad – 500090

May, 2021

DECLARATION

We hereby declare that the work presented in this project entitled **"AN AUTOMATED SYSTEM TO LIMITCOVID-19 USING FACIAL MASK DETECTION IN A SMART CITY NETWORK**" submitted towards completion of Project Work in IV year of B.Tech., CSE at 'BVRIT HYDERABAD College of Engineering For Women', Hyderabad is an authentic record of our original work carried out under the guidance of Mr.K.Rajesh, Assistant Professor, Department of CSE.

Sign. with date:
Ms.K.SUMAVALLIKA
(17WH1A0542)

Sign. with date:
Ms.B.PRANEETHA
(17WH1A0551)

Sign. with date:
Ms.B.VARSHITHA VEENA
(17WH1A0557)

BVRIT HYDERABAD

College of Engineering for Women

(NBA Accredited – EEE, ECE, CSE and IT)

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Bachupally, Hyderabad – 500090

Department of Computer Science and Engineering

Certificate

This is to certify that the Project Work report on "**AN AUTOMATED SYSTEM TO LIMIT COVID-19 USING FACIAL MASK DETECTION IN A SMART CITY NETWORK**" is a bonafide work carried out by Ms.K.SUMAVALLIKA(17WH1A0542); Ms. B.PRANEETHA(17WH1A0551) ; Ms.B.VARSHITHA VEENA(17WH1A0557) in the partial fulfillment for the award of B.Tech. degree in **Computer Science and Engineering, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Head of the Department
Dr. K.Srinivasa Reddy
Professor and HoD,
Department of CSE

Guide
Mr. K.Rajesh
Assisant Professor

External Examiner

Acknowledgements

We would like to express our sincere thanks to **Dr. K V N Sunitha, Principal**, **BVRIT HYDERABAD College of Engineering for Women**, for providing the working facilities in the college.

Our sincere thanks and gratitude to our **Dr.K.Srinivasa Reddy, Professor**, Department of CSE, **BVRIT HYDERABAD College of Engineering for Women** for all the timely support and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Mr.K.Rajesh, Assistant Professor**, Department of CSE**, BVRIT HYDERABAD College of Engineering for Women** for his constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank our Project Coordinator, all the faculty and staff of **CSE** Department who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

Ms.K.SUMAVALLIKA
(17WH1A0542)

Ms.B.PRANEETHA
(17WH1A0551)

Ms.B.VARSHITHA VEENA
(17WH1A0557)

Abstract

COVID-19 pandemic caused by the novel coronavirus is continuously spreading until now all over the world. The impact of COVID-19 has fallen on almost all sectors of development. The healthcare system is going through a crisis. Many precautionary measures have been taken to reduce the spread of this disease where wearing a mask is one of them. A system that restricts the growth of COVID-19 by finding out people who are not wearing any facial mask in a smart city network where all the public places are monitored with Closed-Circuit Television (CCTV) cameras.

A person with or without a mask is detected. A simplified approach to achieve this purpose using some basic Machine Learning packages like TensorFlow, Keras, OpenCV and Scikit-Learn. A deep learning architecture is trained on a dataset that consists of images of people with and without masks collected from various sources.Many algorithms are being devised using convolutional architectures to make the algorithm as accurate as possible. These convolutional architectures have made it possible to extract even the pixel details.The aim is to design a binary face classifier which can detect any face present in the frame irrespective of its alignment. We present a method to generate accurate face segmentation masks from any arbitrary size input image. This would be a useful tool to reduce the spread of this communicable disease for many countries in the world.

List of Contents

List of Figures

1. INTRODUCTION

COVID-19 pandemic caused by the novel coronavirus is continuously spreading until now all over the world. The impact of COVID-19 has fallen on almost all sectors of development. The healthcare system is going through a crisis. Many precautionary measures have been taken to reduce the spread of this disease where wearing a mask is one of them. A system that restricts the growth of COVID-19 by finding out people who are not wearing any facial mask in a smart city network where all the public places are monitored with Closed-Circuit Television (CCTV) cameras.

1.1 Objectives

The task is to find whether a person is with or without a mask. The aim is to design a binary face classifier which can detect any face present in the frame irrespective of its alignment. We present a method to generate accurate face segmentation masks from any arbitrary size input image. This would be a useful tool to reduce the spread of this communicable disease for many countries in the world.

1.2 Methodology

A simplified approach to achieve this purpose using some basic Machine Learning packages like TensorFlow, Keras, OpenCV and Scikit-Learn. A deep learning architecture is trained on a dataset that consists of images of people with and without masks collected from various sources.Many algorithms are being devised using convolutional architectures to make the algorithm as accurate as possible. These convolutional architectures have made it possible to extract even the pixel details.

1.2.1 Dataset

The facemask dataset consists of 3846 images. We collected a total of 1916 images of people with masks and 1930 images of people without a mask. For training purposes, 80% images of each class are used and the rest of the images are utilized for testing purposes.

Using facemask dataset to detect whether a facemask on a person

Since 2019, the whole world is being affected by coronavirus. It has become crucial to develop a system which is used to reduce the spread of coronavirus. In this project we proposed a system that uses CNN and mobilnetv2 models to detect whether a person is

wearing a mask or not. CCTV cameras are used to capture real-time video footage of different public places in the city. From that video footage, facial images are extracted and these images are used to identify the mask on the face. The learning algorithm Convolutional Neural Network (CNN) is used for feature extraction from the images then these features are learned by multiple hidden layers.

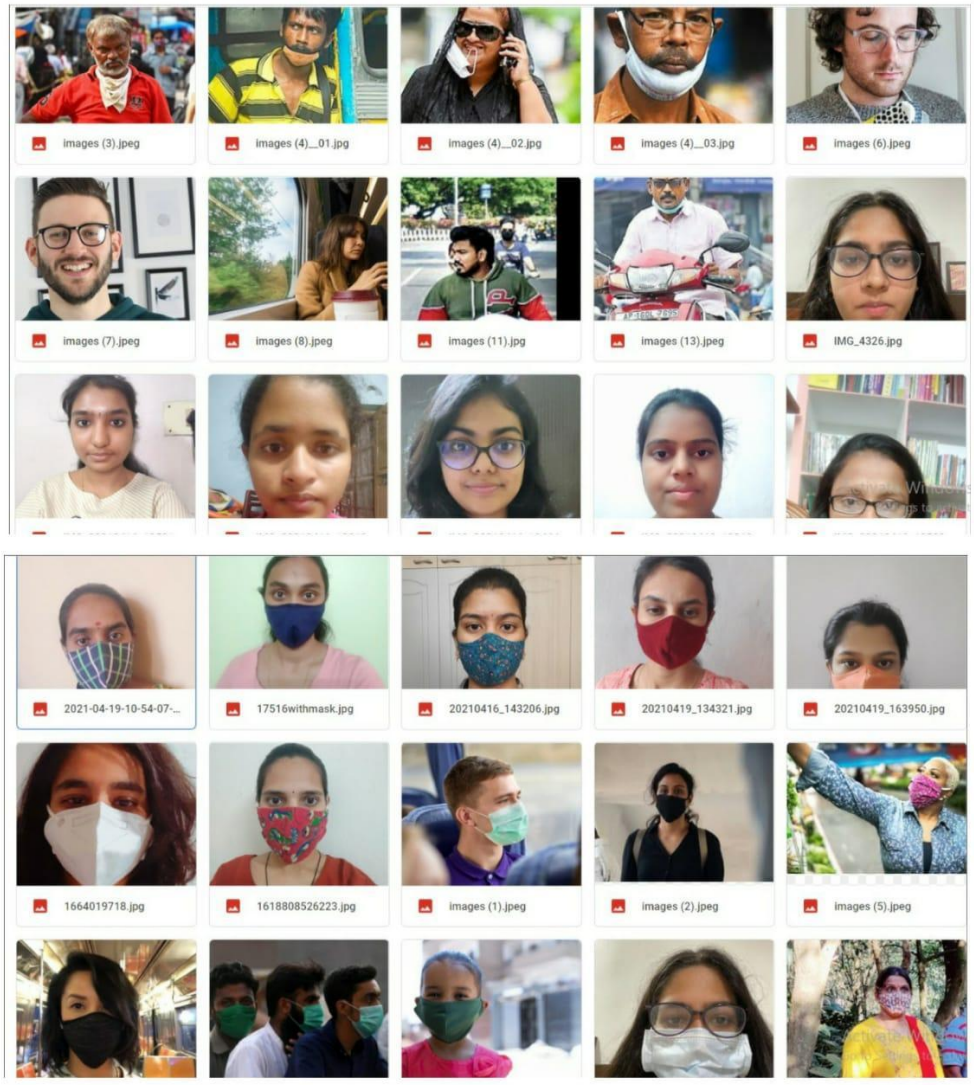


Fig 1.2.1: Dataset sample

Training the network

The network is trained using CNN and mobilenetv2. Pick the following configuration options: Input image resolution: 128,160,192, or 224px. Unsurprisingly, feeding in a higher resolution image takes more processing time, but results in better classification accuracy. We recommend 224 as an initial setting.The relative size of the model as a fraction of the largest MobileNet: 1.0, 0.75, 0.50, or 0.25. We recommend 0.5 as an initial setting. The smaller

models run significantly faster, at a cost of accuracy.

1.2.2 The proposed model

The proposed model consists of convolutional layers followed by a average pooling 2d layer. The final layer is fully connected MLP. ReLu activation function is applied to the output of every convolutional layer and fully connected layer. The first convolutional layer filters the input image with 32 kernels of size 3x3. After average pooling is applied, the output is given as an input for the second convolutional layer with 64 kernels of size 4x4. The last convolutional layer has 128 kernels of size 1x1 followed by a fully connected layer of 512 neurons. The output of this layer is given to the softmax function which produces a probability distribution of the four output class. The model is trained using adaptive moment estimation (Adam) with batch size of 32 for 20 epochs.

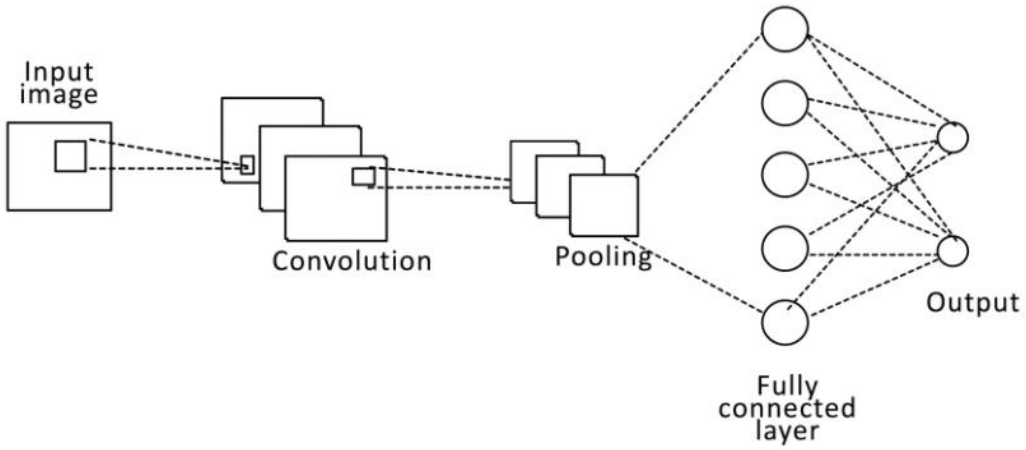


Fig 1.2.2: Convolution Neural Network

There are four CNN algorithm steps,

Convolution: The term convolution refers to the mathematical combination of two functions to produce a third function. It merges two sets of information. In the case of a CNN, the convolution is performed on the input data with the use of a filter or kernel to then produce a feature map.

Here are the three elements that enter into the convolution operation:

- Input image
- Feature detector

- Feature map

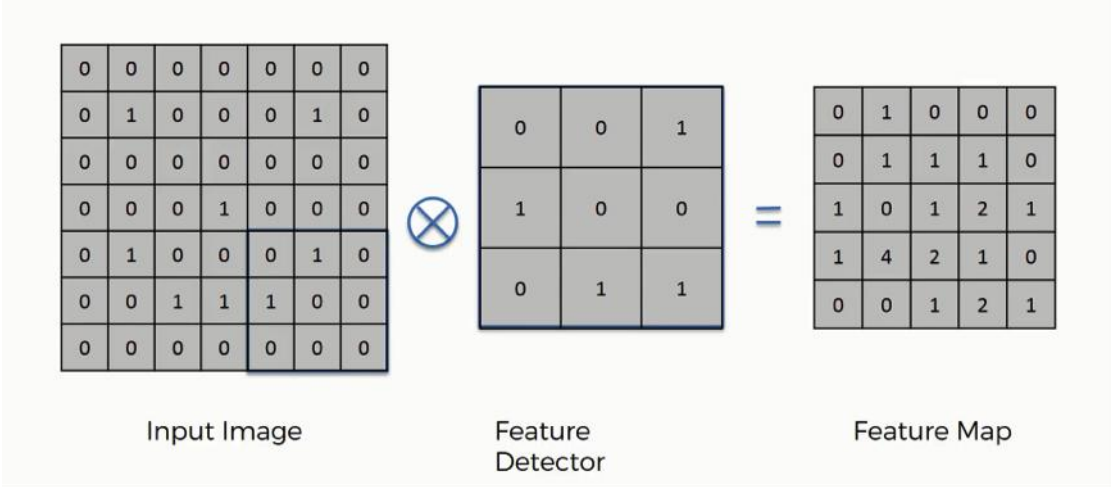


Fig 1.2.2.1: Convolution in CNN

Average pooling 2d: Average pooling involves calculating the average for each patch of the feature map. This means that each 2×2 square of the feature map is down sampled to the average value in the square.

Downsamples the input along its spatial dimensions (height and width) by taking the average value over an input window (of size defined by pool_size) for each channel of the input. The window is shifted by strides along each dimension.

The resulting output when using "valid" padding option has a shape (number of rows or columns) of: output_shape = math.floor((input_shape - pool_size) / strides) + 1 (when input_shape >= pool_size)

The resulting output shape when using the "same" padding option is: output_shape = math.floor((input_shape - 1) / strides) + 1

Arguments

- pool_size: integer or tuple of 2 integers, factors by which to downscale (vertical, horizontal). (2, 2) will halve the input in both spatial dimension. If only one integer is specified, the same window length will be used for both dimensions.
- strides: Integer, tuple of 2 integers, or None. Strides values. If None, it will default to pool_size.
- padding: One of "valid" or "same" (case-insensitive). "valid" means no padding. "same" results in padding evenly to the left/right or up/down of the input such that output has the same height/width dimension as the input.

- data_format: A string, one of channels_last (default) or channels_first. The ordering of the dimensions in the inputs. channels_last corresponds to inputs with shape (batch, height, width, channels) while channels_first corresponds to inputs with shape (batch, channels, height, width). It defaults to the image_data_format value found in your Keras config file at ~/.keras/keras.json. If you never set it, then it will be "channels_last".

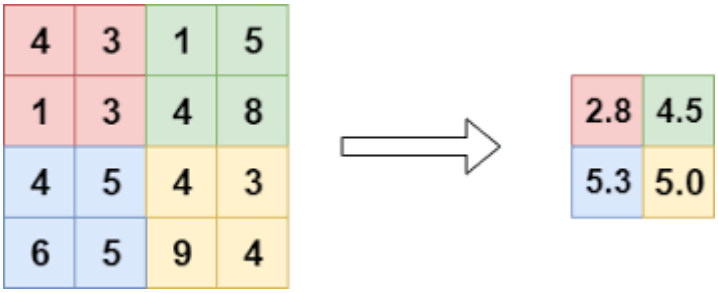


Fig 1.2.2.2:Average Pooling 2d in CNN

Flattening: Flattening is the process of converting all the resultant 2 dimensional arrays into a single long continuous linear vector.

Fig 1.2.2.3: Flattening in CNN

Full Connection: At the end of a CNN, the output of the last Pooling Layer acts as an input to the so-called Fully Connected Layer. There can be one or more of these layers ("fully connected" means that every node in the first layer is connected to every node in the second layer).

As you see from the image below, we have three layers in the full connection step:

- Input layer
- Fully-connected layer
- Output layer

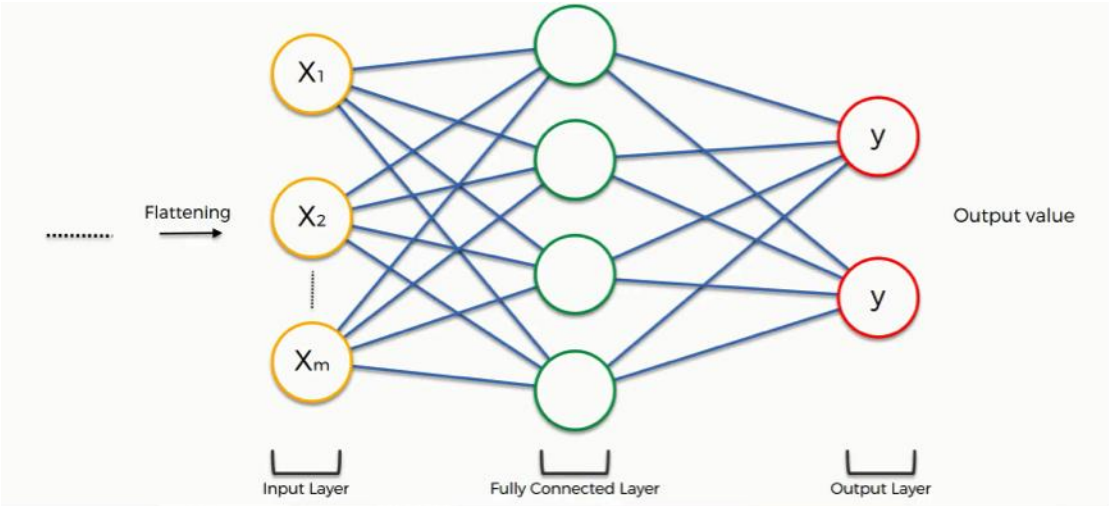


Fig 1.2.2.4: Full Connection in CNN

RELU

In a neural network, the activation function is responsible for transforming the summed weighted input from the node into the activation of the node or output for that input.

The rectified linear activation function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance.

In this tutorial, you will discover the rectified linear activation function for deep learning neural networks.

After completing this tutorial, you will know:

- The sigmoid and hyperbolic tangent activation functions cannot be used in networks with many layers due to the vanishing gradient problem.
- The rectified linear activation function overcomes the vanishing gradient problem, allowing models to learn faster and perform better.
- The rectified linear activation is the default activation when developing multilayer Perceptron and convolutional neural networks.

SOFTMAX

Softmax is a mathematical function that converts a vector of numbers into a vector of probabilities, where the probabilities of each value are proportional to the relative scale of each value in the vector.

The most common use of the softmax function in applied machine learning is in its use as an activation function in a neural network model. Specifically, the network is configured to output N values, one for each class in the classification task, and the softmax function is used to normalize the outputs, converting them from weighted sum values into probabilities that sum to one. Each value in the output of the softmax function is interpreted as the probability of membership for each class.

In this tutorial, you will discover the softmax activation function used in neural network models.

After completing this tutorial, you will know:

- Linear and Sigmoid activation functions are inappropriate for multi-class classification tasks.
- Softmax can be thought of as a softened version of the argmax function that returns the index of the largest value in a list.
- How to implement the softmax function from scratch in Python and how to convert the output into a class label.

1.3 Organization of Project

The technique which is developed takes the facial images that are extracted from the CCTV video footage and uses these images to identify the mask on the face.

We have three modules in our project.

- Data preprocessing
- Training the model
- Video detection

2. LITERATURE REVIEW

Many models have been developed to curb the spread of coronavirus in the past one year.A face mask detecting model named RetinaFaceMask combining with a cross-class object removal algorithm is proposed by a scholar Jiang. The developed model includes a one stage detector consisting of a feature pyramid network that results in slightly higher precision and recall than the baseline result. To reduce the shortage of datasets, they have applied transfer learning, a well-known deep learning technique. Another model was proposed to enforce the social distance using smart city and Intelligent Transportation System (ITS) during COVID-19 pandemic. This model described the deploying sensors in different places of the city to monitor the real-time movement of objects and offered a data-sharing platform. A noticeable contribution of a smart city in controlling the spread of coronavirus in South Korea is explained by Won Sonn and Lee. A time-space cartographer speeded up the contact tracking in the city including patient movement, purchase history, cell phone usages, and cell phone location. Real-time monitoring has been carried out on CCTV cameras in the hallways of residential buildings.

A remarkable pandemic control model without lockdown in a smart city has been outlined by Won Sonn and Lee. The patients have been interviewed and their past movement has been monitored. They have claimed that some patients tried to conceal their past mobility but a real-time tracking system found the exact information. Another model was proposed as a way to minimize the risk during COVID-19. This proposed model used the position of technology to track infected people. Drones and Robot technologies have been applied as medical personnel for providing adequate services to infected people. The continuous supply of essential materials and contactless logistic distribution of systems to society made the way to reduce the spread of coronavirus. ITS and real-time map reflection methods have been used to block the movement of vehicles during the pandemic. In addition, driverless vehicles have been used to monitor the scenarios across the city.

3. THEORETICAL ANALYSIS OF THE PROPOSED PROJECT

3.1 Requirements Gathering

Software requirements deal with software and hardware resources that need to be installed on a server which provides optimal functioning for the application. These software and hardware requirements need to be installed before the packages are installed. These are the most common set of requirements defined by any operation system. These software and hardware requirements provide a compatible support to the operation system in developing an application.

3.1.1 Software Requirements

Programming Language : Python 3.6
Dataset : Facemask dataset
Packages : Tensorflow, Keras, Numpy, Matplotlib, Scikit-learn, OpenCV, Imutils
Tool : Jupyter Notebook

3.1.2 Hardware Requirements

Operating System : Windows 10
Processor : Intel Core i5-2348M
CPU Speed : 2.30 GHz
Memory : 4 GB (RAM)

3.2 Technologies Description

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the

interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python modules

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. Such a file is called a *module*; definitions from a module can be *imported* into other modules or into the *main* module.

Facemask Dataset

The dataset for the experiment contains 3846 images. There are 1916 images of people wearing facemasks and 1930 images of people who are not wearing facemasks.

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and

production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

TensorFlow provides stable Python (for version 3.7 across all platforms)[1] and C APIs, and without API backwards compatibility guarantee: C++, Go, Java, JavaScrip[1] and Swift (archived and development has ceased). Third-party packages are available for C#, Haskell, Julia, MATLAB, R, Scala, Rust, OCaml, and Crystal.

Among the applications for which TensorFlow is the foundation, are automated image-captioning software, such as DeepDream

Keras

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

Up until version 2.3 Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML.As of version 2.4, only TensorFlow is supported. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet is also the author of the EXCeption deep neural network model.

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.

In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling.

Keras allows users to productize deep models on smartphones (iOS and Android), on the web, or on the Java Virtual Machine. It also allows use of distributed training of deep-learning models on clusters of Graphics processing units (GPU) and tensor processing units (TPU).

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make

working with image and text data easier to simplify the coding necessary for writing deep neural network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel.

In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization, and pooling. Keras allows users to productize deep models on smartphones (iOS and Android), on the web, or on the Java Virtual Machine. It also allows use of distributed training of deep-learning models on clusters of Graphics processing units (GPU) and tensor processing units.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image

Matplotlib

Matplotlib is a Python 2D plotting library which 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 shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample

plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Several toolkits are available which extend Matplotlib functionality. Some are separate downloads, others ship with the Matplotlib source code but have external dependencies.

- Basemap: map plotting with various map projections, coastlines, and political boundaries
- Cartopy: a mapping library featuring object-oriented map projection definitions, and arbitrary point, line, polygon and image transformation capabilities.
- Excel tools: utilities for exchanging data with Microsoft Excel
- GTK tools: interface to the GTK library
- Qt interface
- Mplot3d: 3-D plots
- Natgrid: interface to the natgrid library for gridding irregularly spaced data.
- matplotlib2tikz: export to Pgfplots for smooth integration into LaTeX documents
- Seaborn: provides an API on top of Matplotlib that offers sane choices for plot style and color defaults, defines simple high-level functions for common statistical plot types, and integrates with the functionality provided by Pandas.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.Scikit-learn is largely written in Python, and uses NumPy extensively for high-performance linear algebra and array operations. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing

- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis
- Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

Os

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

Notes on the availability of these functions:

- The design of all built-in operating system dependent modules of Python is such that as long as the same functionality is available, it uses the same interface; for example, the function os.stat(path) returns stat information about *path* in the same format (which happens to have originated with the POSIX interface).
- Extensions peculiar to a particular operating system are also available through the os module, but using them is of course a threat to portability.
- All functions accepting path or file names accept both bytes and string objects, and result in an object of the same type, if a path or file name is returned.
- On VxWorks, os.fork, os.execv and os.spawn*p* are not supported.

Time

The Python time module provides many ways of representing time in code, such as objects, numbers, and strings. It also provides functionality other than representing time, like waiting during code execution and measuring the efficiency of your code.

OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the

commercial products.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. The library is used extensively in companies, research groups and by governmental bodies.

OpenCV's deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

Imutils

A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying matplotlib images, sorting contours, detecting edges, and much more easier with OpenCv and both python2.7 and python3.

Jupyter Notebook

The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter.

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported

programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

A Jupyter Notebook document is a JSON document, following a versioned schema, containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through "Download As" in the web interface, via the nbconvert library[9] or "jupyter nbconvert" command line interface in a shell. To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

Anaconda Prompt

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPI as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator, as a graphical alternative to the command line interface (CLI).

The big difference between conda and the pip package manager is in how package dependencies are managed, which is a significant challenge for Python data science and the reason conda exists.

Before version 20.3, when pip installed a package, it automatically installed any dependent Python packages without checking if these conflict with previously installed packages. It would install a package and any of its dependencies regardless of the state of the existing installation. Because of this, a user with a working installation of, for example, Google Tensorflow, could find that it stopped working having used pip to install a different package that requires a different version of the dependent numpy library than the one used by Tensorflow. In some cases, the package would appear to work but produce different results in detail. While pip has since implemented consistent dependency resolution, this difference accounts for a historical differentiation of the conda package manager.

In contrast, conda analyses the current environment including everything currently installed, and, together with any version limitations specified (e.g. the user may wish to have

Tensorflow version 2,0 or higher), works out how to install a compatible set of dependencies, and shows a warning if this cannot be done.

Open source packages can be individually installed from the Anaconda repository, Anaconda Cloud (anaconda.org), or the user's own private repository or mirror, using the conda install command. Anaconda, Inc. compiles and builds the packages available in the Anaconda repository itself, and provides binaries for Windows 32/64 bit, Linux 64 bit and MacOS 64-bit. Anything available on PyPI may be installed into a conda environment using pip, and conda will keep track of what it has installed itself and what pip has installed.

Custom packages can be made using the conda build command, and can be shared with others by uploading them to Anaconda Cloud, PyPI or other repositories.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it is possible to create new environments that include any version of Python packaged with conda.

Python IDLE

Every Python installation comes with an Integrated Development and Learning Environment, which you'll see shortened to IDLE or even IDE. These are a class of applications that help you write code more efficiently. While there are many IDEs for you to choose from, Python IDLE is very bare-bones, which makes it the perfect tool for a beginning programmer.

Python IDLE comes included in Python installations on Windows and Mac. If you're a Linux user, then you should be able to find and download Python IDLE using your package manager. Once you've installed it, you can then use Python IDLE as an interactive interpreter or as a file editor.

The best place to experiment with Python code is in the interactive interpreter, otherwise known as a shell. The shell is a basic Read-Eval-Print Loop (REPL). It reads a Python statement, evaluates the result of that statement, and then prints the result on the screen. Then, it loops back to read the next statement.

The Python shell is an excellent place to experiment with small code snippets. You can access it through the terminal or command line app on your machine. You can simplify your workflow with Python IDLE, which will immediately start a Python shell when you open it.

Every programmer needs to be able to edit and save text files. Python programs are files with the .py extension that contain lines of Python code. Python IDLE gives you the ability to create and edit these files with ease.

Python IDLE also provides several useful features that you'll see in professional IDEs, like basic syntax highlighting, code completion, and auto-indentation. Professional IDEs are more robust pieces of software and they have a steep learning curve. If you're just beginning your Python programming journey, then Python IDLE is a great alternative.

4. DESIGN

4.1 Introduction

Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Once system requirements have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed, reviewed and documented. System design can be viewed from either a technical or project management perspective. From the technical point of view, design consists of four activities – architectural design, data structure design, interface design and procedural design.

The design for our project consists of two phases: Train mask detector where the model is trained based on the dataset provided and applying a face mask detector which is used to detect whether a person is wearing a mask or not.

4.2 Architecture Diagram

The architecture of a system is used to define the work flow of the system. In our project initially, images from the dataset are loaded and are trained using tensorflow and keras. From the CCTV footage the images of faces of people are extracted, then the trained model is used to detect whether a person is wearing a mask or not.

The learning model is based on CNN which is very useful for pattern recognition from images. The network comprises an input layer, several hidden layers and an output layer. The hidden layers consist of multiple convolution layers that learn suitable filters for important feature extraction from the given samples. The features extracted by CNN are used by multiple dense neural networks for classification purposes. The architecture contains three pairs of convolution layers each followed by one max pooling layer. This layer decreases the spatial size of the representation and thereby reduces the number of parameters. As a result,

the computation is simplified for the network. Then, a flatten layer reshapes the information into a vector to feed into the dense network. Three pairs of dense and dropout layers learn parameters for classification. The dense layer comprises a series of neurons each of them learn nonlinear features. The dropout layer prevents the network from overfitting by dropping out units. Finally, a dense layer containing two neurons distinguishes the classes.

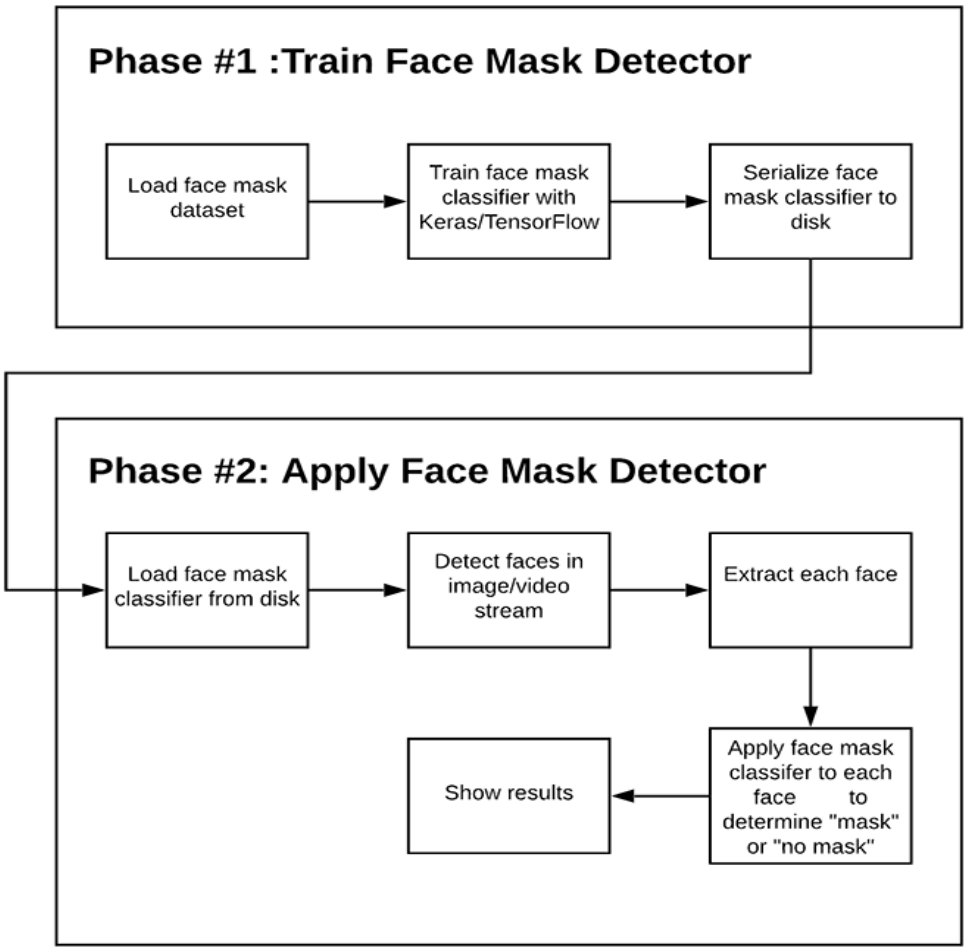


Fig 4.2.1: Architecture Diagram

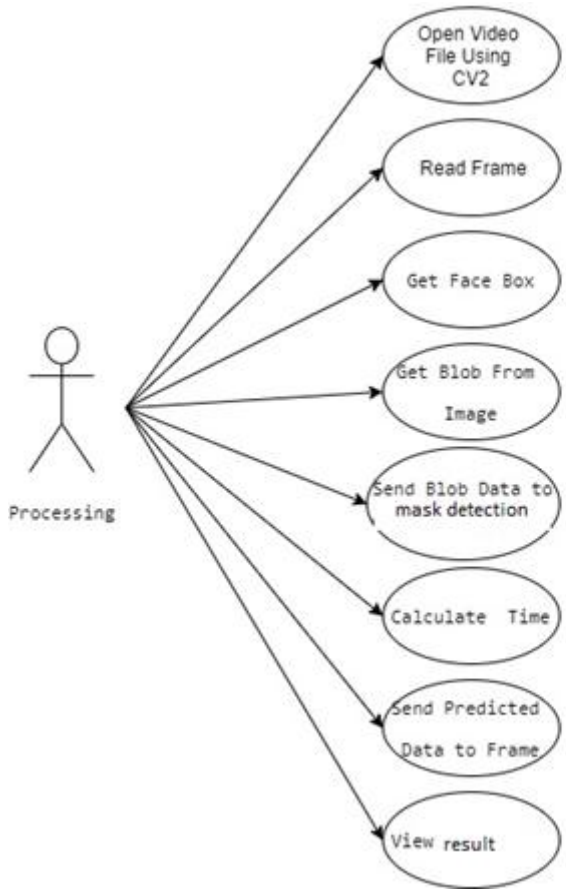
4.3 UML Diagrams

4.3.1 Use Case Diagram

Only static behavior is not sufficient to model a system rather dynamic behavior is more important than static behavior. In UML, there are five diagrams available to model the dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature, there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. Use case diagrams consist of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system.

Our project consists of two use case diagrams: one for training the model and the other for face mask detection.



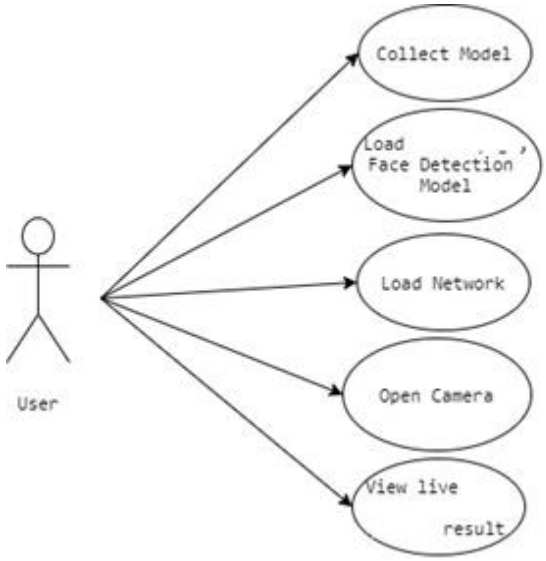


Fig 4.2.2: Use Case Diagrams of training the model and facemask detection

4.3.2 Sequence Diagram

Sequence Diagrams Represent the objects participating in the interaction horizontally and time vertically. The sequence diagram of our model contains the interaction between the following objects: user, system application, camera, image database and trained neural network.

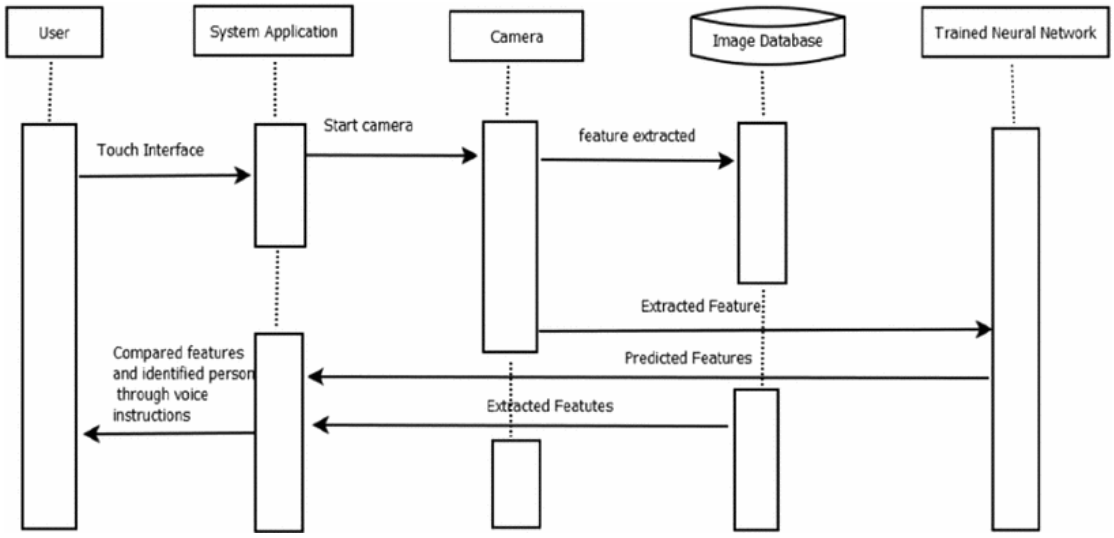


Fig 4.3.2: Sequence Diagram

4.3.3 Activity Diagram

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control. The activity diagram of our face mask detection model is given below.

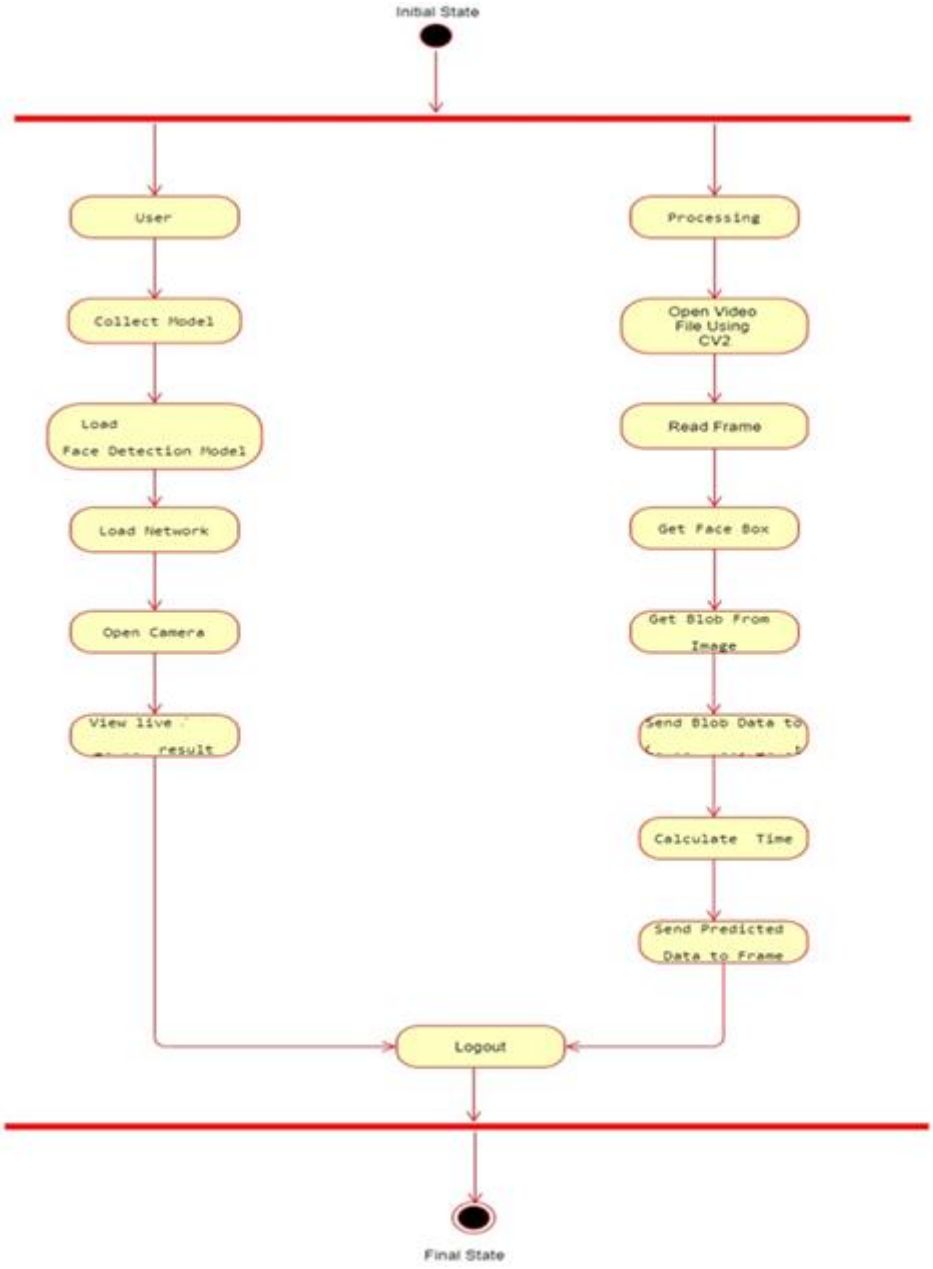


Fig 4.2.3: Activity Diagram

4.3.4 Class Diagram

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the system of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed. The class diagram of our facemask detection model consists of two classes they are user and processing

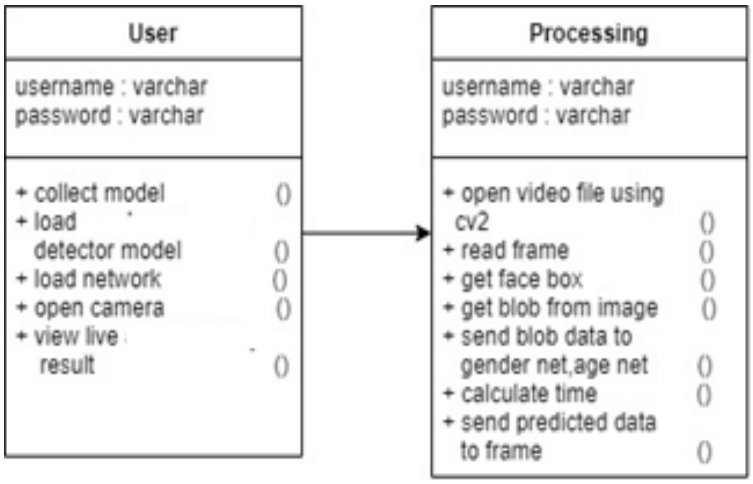


Fig 4.2.4: Class Diagram

5. IMPLEMENTATION

5.1 Coding

train_mask_detector.py

```
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 argparse
import os

INIT_LR = 1e-4
EPOCHS = 20
BS = 32
print("[INFO] loading images...")
imagePaths = list(paths.list_images("C://Users//Lenovo//Desktop//dataset"))

data = []
labels = []
```

```
for imagePath in imagePaths:
        label = imagePath.split(os.path.sep)[-2]
        image = load_img(imagePath, target_size=(224, 224))
        image = img_to_array(image)
        image = preprocess_input(image)
        data.append(image)
        labels.append(label)

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

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

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

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")

baseModel = MobileNetV2(weights="imagenet", include_top=False,
        input_tensor=Input(shape=(224, 224, 3)))
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)
for layer in baseModel.layers:
        layer.trainable = False
print("[INFO] compiling model...")
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
model.compile(loss="binary_crossentropy", optimizer=opt,
        metrics=["accuracy"])
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)
print("[INFO] evaluating network...")
predIdxs = model.predict(testX, batch_size=BS)
predIdxs = np.argmax(predIdxs, axis=1)
print(classification_report(testY.argmax(axis=1), predIdxs,
        target_names=lb.classes_))
print("[INFO] saving mask detector model...")
model.save("mask_detector1.model", save_format="h5")
```

detect_mask_video.py

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

```
import time
import cv2
import os
def detect_and_predict_mask(frame, faceNet, maskNet):
	(h, w) = frame.shape[:2]
		blob = cv2.dnn.blobFromImage(frame, 1.0, (400, 400),
			(140.0, 250.0, 150.0))
		faceNet.setInput(blob)
		detections = faceNet.forward()
		faces = []
		locs = []
		preds = []
		for i in range(0, detections.shape[2]):
			confidence = detections[0, 0, i, 2]
			if confidence > 0.5:
				box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
				(startX, startY, endX, endY) = box.astype("int")
				(startX, startY) = (max(0, startX), max(0, startY))
				(endX, endY) = (min(w - 1, endX), min(h - 1, endY))
				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)
				faces.append(face)
				locs.append((startX, startY, endX, endY))
		if len(faces) > 0:
			faces = np.array(faces, dtype="float32")
			preds = maskNet.predict(faces, batch_size=32)
		return (locs, preds)
print("[INFO] loading face detector model...")
prototxtPath = os.path.sep.join(["face", "deploy.prototxt"])
weightsPath = os.path.sep.join(["face",
```

```
        "res10_300x300_ssd_iter_140000.caffemodel"])
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)
print("[INFO] loading face mask detector model...")
maskNet = load_model("mask_detector.model")
print("[INFO] starting video stream...")
vs = VideoStream(src=0).start()
time.sleep(2.0)
while True:
        frame = vs.read()
        frame = imutils.resize(frame, width=500)
        (locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)
        for (box, pred) in zip(locs, preds):
                (startX, startY, endX, endY) = box
                (mask, withoutMask) = pred
                if mask > withoutMask:
                        label ="Mask"
                        color = (0, 255, 0)
                else:
                        label="No Mask"
                        color = (0, 0, 255)
                label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)
                cv2.putText(frame, label, (startX, startY - 10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
                cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)
        cv2.imshow("Frame", frame)
        key = cv2.waitKey(1) & 0xF
        if key == ord("q"):
                break
cv2.destroyAllWindows()
vs.stop()
```

5.2 TESTING

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as a system element and attendant costs associated with a software failure are motivating factors for we planned, through testing. Testing is the process of executing a program with the intent of finding an error. The design of tests for software and other engineered products can be as challenging as the initial design of the product itself.

There of basically two types of testing approaches.

One is Black-Box testing – the specified function that a product has been designed to perform, tests can be conducted that demonstrate each function is fully operated.

The other is White-Box testing – knowing the internal workings of the product ,tests can be conducted to ensure that the internal operation of the product performs according to specifications and all internal components have been adequately exercised.

White box and Black box testing methods have been used to test this package. The entire loop constructs have been tested for their boundary and intermediate conditions. The test data was designed with a view to check for all the conditions and logical decisions. Error handling has been taken care of by the use of exception handlers.

5.2.1 Testing Strategies

Testing is a set of activities that can be planned in advance and conducted systematically. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

Software testing is one element of verification and validation. Verification refers to the set of activities that ensure that software correctly implements a specific function. Validation refers to a different set of activities that ensure that the software that has been built is traceable to customer requirements.

The main objective of software is testing to uncover errors. To fulfill this objective, a series of test steps unit, integration, validation and system tests are planned and executed. Each test step is accomplished through a series of systematic test techniques that assist in the design of test cases. With each testing step, the level of abstraction with which software is considered is broadened.

Testing is the only way to assure the quality of software and it is an umbrella activity rather than a separate phase. This is an activity to be performed in parallel with the software effort and one that consists of its own phases of analysis, design, implementation, execution and maintenance.

UNIT TESTING:

This testing method considers a module as a single unit and checks the unit at interfaces and communicates with other modules rather than getting into details at statement level. Here the module will be treated as a black box, which will take some input and generate output. Outputs for a given set of input combinations are pre-calculated and are generated by the module.

SYSTEM TESTING:

Here all the pre-tested individual modules will be assembled to create the larger system and tests are carried out at system level to make sure that all modules are working in synchrony with each other. This testing methodology helps in making sure that all modules which are running perfectly when checked individually are also running in cohesion with other modules. For this testing we create test cases to check all modules at once and then generate test combinations of test paths throughout the system to make sure that no path is making its way into chaos.

MODULE TESTING

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

INTEGRATED TESTING

Testing is a major quality control measure employed during software development. Its basic function is to detect errors. Sub functions when combined may not produce more than it is desired. Global data structures can represent the problems. Integrated testing is a systematic technique for constructing the program structure while conducting the tests. To uncover errors that are associated with interfacing the objective is to make unit test modules and build a program structure that has been detected by design. In a non - incremental integration all the modules are combined in advance and the program is tested as a whole. Here errors will appear in an endless loop function. In incremental testing the program is constructed and tested in small segments where the errors are isolated and corrected.

Different incremental integration strategies are top – down integration, bottom – up integration, regression testing.

ACCEPTANCE TESTING

When that user finds no major problems with its accuracy, the system passes through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which eliminates wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

5.3 TEST CASES

Test Case Id	Test Case Name	Test Case Desc.	Test Steps			Test Case Status	Test Priority
			Step	Expected	Actual		
01	Upload the tasks dataset	Verify either file is loaded or not	If dataset is not uploaded	It cannot display the file loaded message	File is loaded which displays task waiting time	High	High
02	Upload live video	Verify either dataset loaded or not	If dataset is not uploaded	It cannot display dataset reading process completed	It can display dataset reading process completed	low	High
03	Preprocessing	Whether preprocessing on the dataset applied or not	If not applied	It cannot display the necessary data for further process	It can display the necessary data for further process	Medium	High
04	Prediction CNN model	Whether Prediction algorithm applied on the data or not	If not applied	CNN model is created	CNN model is created	High	High

05	Predictio n	Whether predicted data is displayed or not	If not displayed	It cannot view prediction containing face mask data	It can view prediction containing face mask or not	High	High
06	Noisy Records Chart	Whether the graph is displayed or not	If graph is not displayed	It does not show the variations in between clean and noisy records	It shows the variations in between clean and noisy records	Low	Medi um

5.4 ACCURACY

Our model produces an accuracy of 98.7% on the test data with a minimal amount of loss.

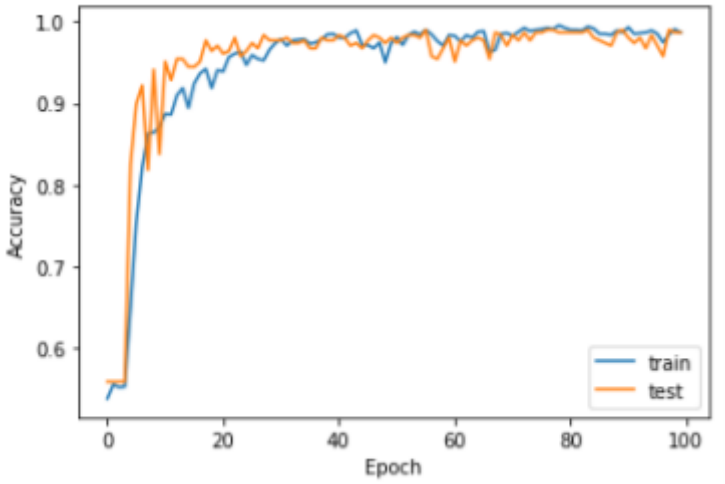


Fig 5.3.1: Accuracy for the developed system for train and testing phase

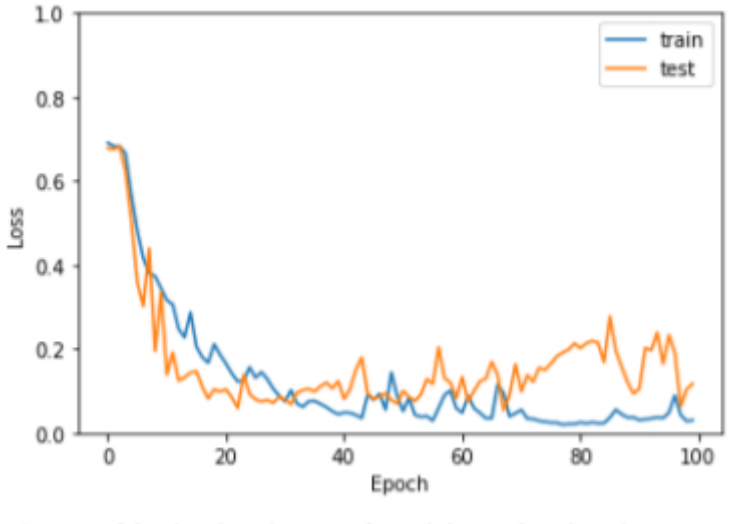


Fig 5.3.2: Loss for the developed system for train and testing phase

5.5 RESULTS

Face images extracted from video footage of the CCTV camera are provided as the input. The system detects whether the person is wearing a mask or not as output.

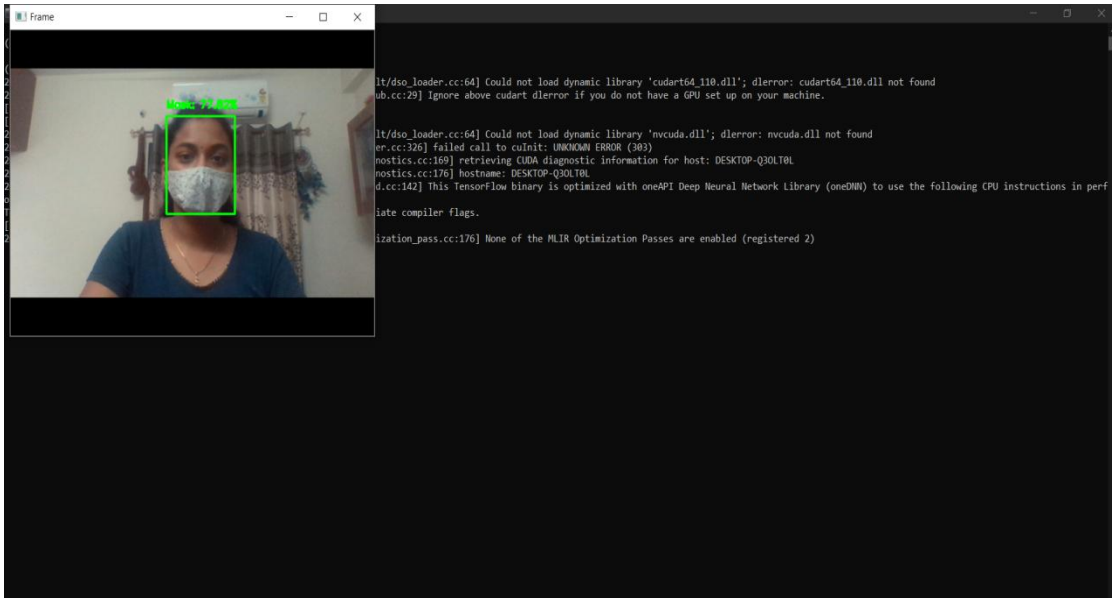


Fig 5.4.1: Output if the person is wearing a mask

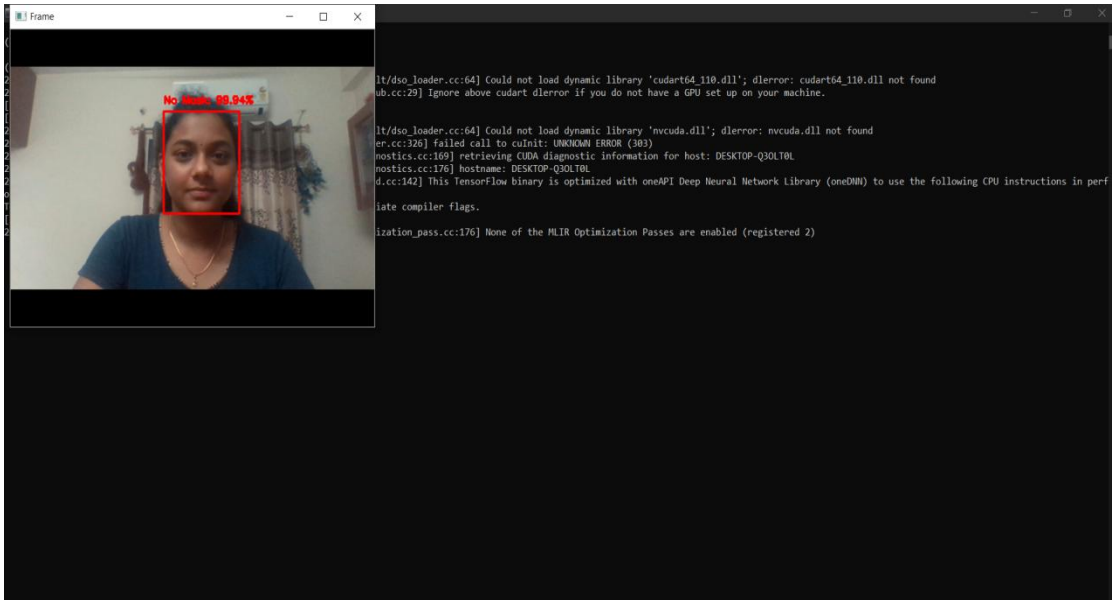


Fig 5.4.2: Output if the person is not wearing a mask

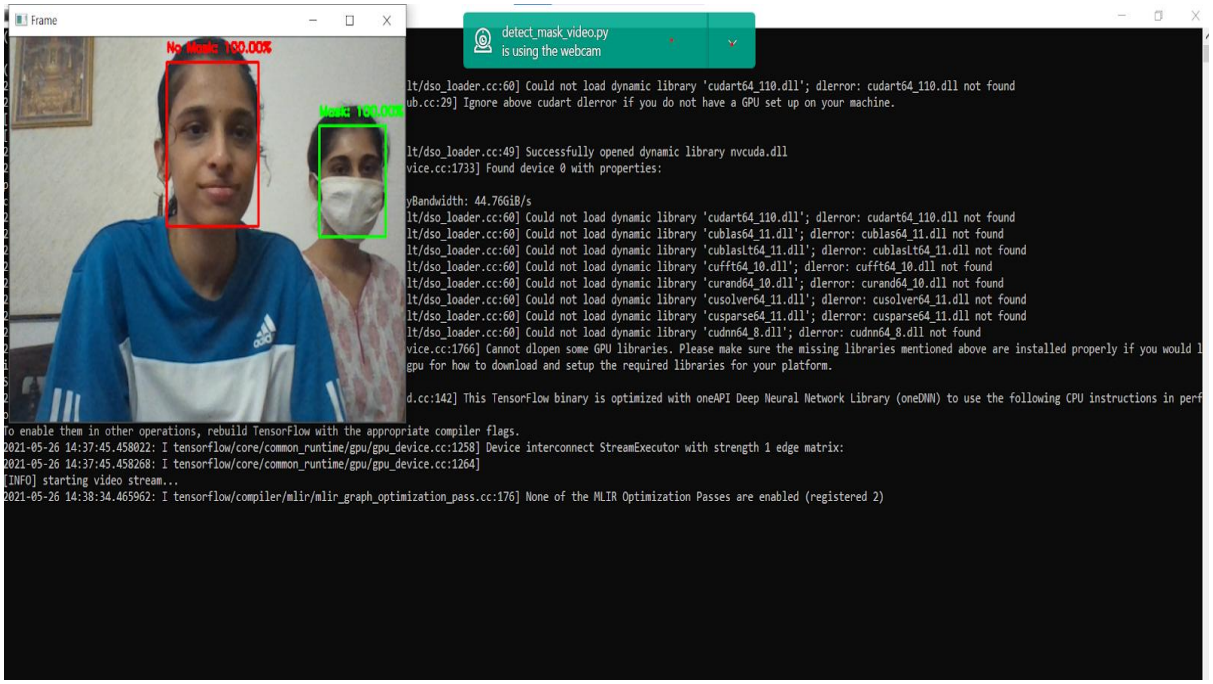


Fig 5.4.3 Output with two persons

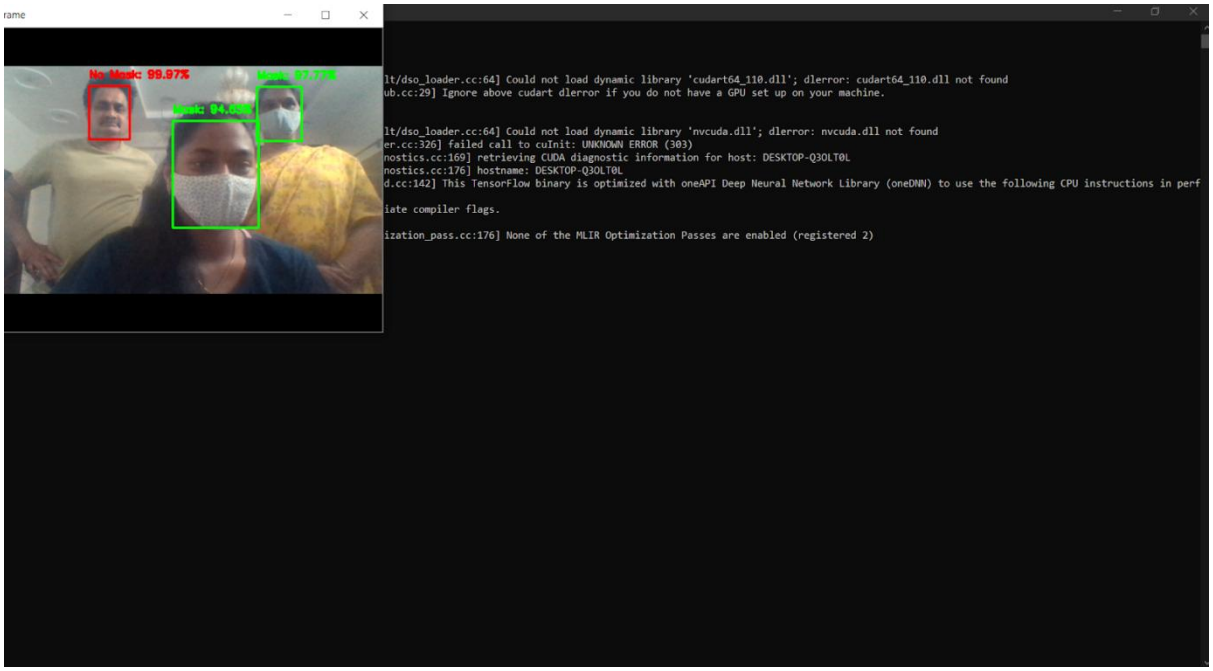


Fig 5.4.4 Output with three persons

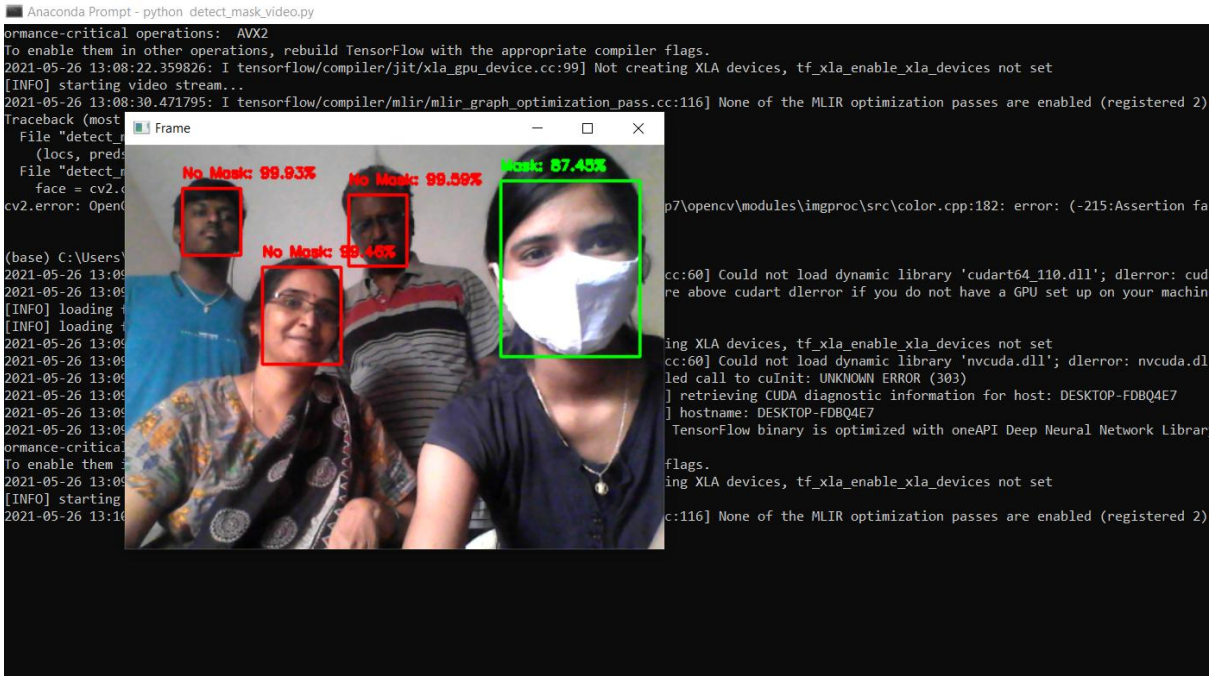


Fig 5.4.5 Output with multiple persons

6. CONCLUSION AND FURURE SCOPE

This project presents a system for a smart city to reduce the spread of coronavirus by detecting whether a person is wearing a mask or not. The motive of the work comes from the people disobeying the rules that are mandatory to stop the spread of coronavirus. The system contains a face mask detection architecture where a deep learning algorithm is used to detect the mask on the face. To train the model, labeled image data are used where the images were facial images with masks and without a mask. The proposed system detects a face mask with an accuracy of 98.7%.

The developed system faces difficulties in classifying faces covered by hands since it almost looks like the person wearing a mask. While any person without a face mask is traveling on any vehicle, the system cannot locate that person correctly. For a very densely populated area, distinguishing the face of each person is very difficult. For this type of scenario, identifying people without facemask would be very difficult for our proposed system. In order to get the best result out of this system, the city must have a large number of CCTV cameras to monitor the whole city as well as dedicated manpower to enforce proper laws on the violators.

8. REFERENCES

- Base Paper**:**
  https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9216386

- Dataset:
  https://drive.google.com/drive/folders/158ysxTE9E46Xbveo2c4z92Rd6NIwwzsz?usp=sharing

- Reference papers:
  https://www.worldometers.info/coronavirus
  https://www.irjet.net/archives/V7/i8/IRJET-V7I8530.pdf
  https://link.springer.com/article/10.1007/s11263-019-01247-4
  http://arxiv.org/abs/2005.03950