VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590014, KARNATAKA

A PROJECT REPORT
ON

"REAL TIME FACE MASK DETECTION USING DEEP LEARNING"

Submitted in Partial Fulfillment for the award of the degree Of

BACHELOR OF ENGINNERING

IN

INFORMATION SCIENCE & ENGINNERING

Submitted By

Aditya N	**1SG17IS003**
Akshay P	**1SG17IS008**
Arpita Prakash Paramaj	**1SG17IS012**
Deepak K	**1SG17IS027**

Under the Guidance of

Dr. Asha P N
Associate Professor, Dept. of ISE

Department of Information Science and Engineering
SAPTHAGIRI COLLEGE OF ENGINEERING
14/5, Chikkasandra, Hesarghatta Main Road
Bengaluru-560057

Karnataka State Council for Science and Technology
Near J.N. Tata Auditorium, Indian Institute of Science Campus, Bengaluru, Karnataka 560012

Academic Year 2020-21

44th SERIES OF STUDENT PROJECTS PROGRAMME: 2020 - 2021

ABSTRACT

Covid-19 has affected the world seriously and is causing a global health crisis. The major protection method for people is to wear masks in public areas. Many public service providers require customers to use the services only if they wear masks properly. However, only a few research studies regarding face mask detection based on Image analysis. In this paper, we propose Real time Face Mask Detection, which is a highly-accuracy and efficient face mask detector. A composite model using classical and deep machine learning for face mask detection will be proposed. A Real-Time face mask detection consists of with mask and without mask images as datasets, we are using OpenCV for real-time face mask detection from a live stream via our webcam in streamlit module of Python. We are using datasets to build a COVID-19 Real-Time face mask detector with computer vision using Python, OpenCV, Tensor Flow Keras and Streamlit (for Framework). Our aim is to find whether a person on image or video stream is wearing a face mask or not by the help of deep learning and computer visions. Many algorithms are being devised using convolutional architectures to make the algorithms as accurate as possible. All these convolutional architectures have made it possible to extract even the pixel details. We aim to design a Real time face mask detection using computer vision techniques. We are presenting a method for generating accurate face segmentation masks from any arbitrary size input image. Starting from the RGB image of any size, the method uses Predefined Training Weights of VGG – 16(Visual Geometry Group) Architecture for the purpose of feature extraction and pre trained model has been used to detect the humans.

ACKNOWLEDGEMENT

Any achievement does not depend solely on the individual efforts but on the guidance, encouragement and co-operation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped us in carrying out this project work. We would like to take this opportunity to thank them all.

We would like to express our immense gratitude to **Dr. H Ramakrishna,** Principal, Sapthagiri College of Engineering Bangalore, for his help and inspiration during the tenure of the course.

We also extend our sincere thanks to **Dr. H R Ranganatha**, Professor and Head, Department of Information Science and Engineering, Sapthagiri College of Engineering, for their constant support.

We are also thankful to Project Coordinator **Prof. Prerana Chaithra**, Associate Professor, Department of Information Science and Engineering, Sapthagiri College of Engineering, for her valuable suggestions and guidance in our work.

We would like to express our heartfelt gratitude to **Dr. Asha P N,** Associate Professor, Departmentof Information Science and Engineering, Sapthagiri College of Engineering, for her timely advice on the project and regular assistance throughout the work.

We also extend our sincere thanks to all the **Faculty members** and **supporting staff**, Department of Information Science and Engineering, Sapthagiri College of Engineering, for their constant support andencouragement.

We are grateful to **The Karnataka State Council for Science and Technology**, Bangalore, Karnataka for considering and accepting our project proposal and giving us a chance to being in research, thereby for all financial assistances. Without this, it would have been impossible to carry out our project.

ADITYA N (1SG17IS003)

AKSHAY P (1SG17IS008)

ARPITA PRAKASH PARAMAJ (1SG17IS012)

DEEPAK K (1SG17IS027)

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Overview

Face detection have been emerged as a very interesting problem in image processing and computer vision. It has a confidential range of applications from facial motion capture to face recognition which at the start needs the face to be detected with a very good accuracy. Face detection is more common in today's days because it not only used on images but also in video applications like real time surveillance and detecting faces in videos. More accuracy classification of images is possible now with the advancements of convolution networks. After face detection Pixel level information is required which most face detection methods will fail to provide. But obtaining pixel level details has been one of the challenging parts in semantic segmentation. Here Semantic segmentation is the process of assigning a label to each pixel of an image. In our case the labels are either face or no face. Semantic segmentation is used to separate the face by classifying each pixel of the image as face or a background.

This project proposes a model for face detection using semantic segmentation in an image by classifying each pixel as face with mask and face without mask i.e. beneficially creating a binary classifier and then detecting particular segmented area. The model works very well only for images having frontal faces. The pandemic situation reports 96 of WHO presented that COVID-19 has globally infected over 2.7 million people and caused over 180,000 deaths. In addition, there are several similar large scale serious respiratory diseases, such as sever acute respiratory syndrome (SARS) and the Middle East respiratory syndrome (MERS), which occurred in the past few years. Meanwhile Mask wearing can reduce the transmission of virus. Many public service providers require the customers to use the services if and only if they wear masks. Therefore, detection of face mask has become a major computer vision and image processing task to help the global society.

Social Distancing, the term that has taken the world by storm and is transforming the way we live. Social distancing has become a mantra around the world, transcending languages and cultures. This way of living has been forced upon us by the fastest growing pandemic the world has ever seen – COVID-19. As per the World Health Organization (WHO), COVID-19 has so far

infected almost 4 million people and claimed over 230K lives globally. Around 213 countries have been affected so far by the deadly virus. The biggest cause of concern is that COVID-19 spreads from person to person through contact or if you're within close proximity of an infected person. Given how densely populated some areas are, this has been quite a challenge. We here are building a tool that can potentially detect where each person is in realtime, and return a bounding box that turns red if the distance between two people is dangerously close. This can be used by governments to analyse the movement of people and alert them if the situation turns serious. Some of the outcomes are:

- Reduces number of infections and transmission of virus.
- Safe guard work-spaces and working environment.
- Improve business operational efficiency & improves productivity.
- Better decision making through analytics and visualization.
- Real-time capturing and effective results.

1.2 Problem Statement

Covid-19 has affected the world seriously and the major protection method for people is to wear masks in the pandemic situation to protect themselves so a real time face mask detection model is implemented. The aim is to identify if a person is wearing a mask or not with both browsing photos and webcam and displays appropriate results with some messages.

1.3 Objectives

To build an object detection model to detect whether a person is wearing mask or not from the images and video streams with the help of image processing, computer vision and deep learning algorithm. Using Streamlit module for the frontend framework and providing the results for the face mask detection. The accuracy of the model is expected to be more than 90%.

- Review results through a web browser and give the percentage.
- Alert system, popups, voice system are integrated.
- Digital outputs to a Covid-19 analysis link.

1.4 Limitations

- Training and testing above 1500 image dataset are not capable in normal CPU as it need huge Epochs that is 50% and more.
- We require GPU machines to get high accurate and performable model, otherwise it causes either over fitting or under fitting ML Model.

1.5 Literature Survey

[1] An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network COVID-19 pandemic caused by novel coronavirus is continuously spreading until now all over the world. The impact of COVID-19 has been 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. In this paper, we propose a system that restrict 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. While a person without a mask is detected, the corresponding authority is informed through the city network. A deep learning architecture is trained on a dataset that consists of images of people with and without masks collected from various sources. The trained architecture achieved 98.7% accuracy on distinguishing people with and without a facial mask for previously unseen test data. It is hoped that our study would be a useful tool to reduce the spread of this communicable disease for many countries in the world.

We proposed an automated smart framework for screening persons who are not using a face mask in this paper. In the smart city, all public places are monitored by CCTV cameras. The cameras are used to capture images from public places; then these images are feed into a system that identifies if any person without face mask appears in the image. If any person without a face mask is detected then this information is sent to the proper authority to take necessary actions. The block diagram of the developed framework is depicted in Figure. 1. All the blocks of the developed system are described as follows.

[2] According to data obtained by the World Health Organization, the global pandemic of COVID-19 has severely impacted the world and has now infected more than eight million people

worldwide. Wearing face masks and following safe social distancing are two of the enhanced safety protocols need to be followed in public places in order to prevent the spread of the virus. To create safe environment that contributes to public safety, we propose an efficient computer vision-based approach focused on the real-time automated monitoring of people to detect both safe social distancing and face masks in public places by implementing the model on raspberry pi4 to monitor activity and detect violations through camera. The proposed system helps to ensure the safety of the people at public places by automatically monitoring them whether they maintain a safe social distance, and also by detecting whether or not and individual wears face mask. This section briefly describes the solution architecture and how the proposed system will automatically functions in an automatic manner to prevent the coronavirus spread. The proposed system uses a transfer learning approach to performance optimization with a deep learning algorithm and a computer vision to automatically monitor people in public places with a camera integrated with a raspberry pi4 and to detect people with mask or no mask. We also do fine tuning, which is another form of transfer learning, more powerful than just the feature extraction.

The proposed system is a deep learning solution that uses OpenCV and TensorFlow, to train the model. We combine the deep learning MobileNetV2 modal with the SSD framework for a fast and efficient deep learning solution for real-time human detection in video streams and use a triangular similarity technique to measure distance between persons detected by camera in real time in public places and comprises customized data collection to resolve a face mask detection model with variance in the types of face masks worn by the public in real time by means of a transfer of learning[20] to a pre-trained SSD face detector.

[3] Face Detection has evolved as a very popular problem in Image processing and Computer Vision. Many new 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. We aim 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. Beginning from the RGB image of any size, the method uses Predefined Training Weights of VGG – 16 Architecture for feature extraction. Training is performed through Fully Convolutional Networks to semantically segment

out the faces present in that image. Gradient Descent is used for training while Binomial Cross Entropy is used as a loss function. Further the output image from the FCN is processed to remove the unwanted noise and avoid the false predictions if any and make bounding box around the faces. Furthermore, proposed model has also shown great results in recognizing non-frontal faces. Along with this it is also able to detect multiple facial masks in a single frame. Experiments were performed on Multi Parsing Human Dataset obtaining mean pixel level accuracy of93.884 % for the segmented face masks.

We propose a method of obtaining segmentation masks directly from the images containing one or more faces indifferent orientation. The input image of any arbitrary size is resized to 224 × 224 × 3 and fed to the FCN network for feature extraction and prediction. The output of the network is then subjected to post processing. Initially the pixel values of the face and background are subjected to global thresholding. After that it passed through median filter to remove the high frequency noise and then subjected to Closing operation to fill the gaps in the segmented area. After this bounding box is drawn around the segmented area.

The feature extraction and prediction are performed using pre-defined training weights of VGG 16 architecture. The basic VGG-16 architecture is depicted in Figureure 2. Our proposed model consists of a total of 17 convolutional layers and 5 Max pooling layers. The initial image size which is fed to the model is 224×224×3. As the image is processed through the layers for feature extraction its passed through convolutional layers and max pooling layers.

[4] The COVID - 19 pandemic is devastating mankind irrespective of caste, creed, gender, and religion. Until a vaccine is discovered, we should do our bit to constrain the expanse of the coronavirus. Using a face mask can undoubtedly help in managing the spread of the virus. COVID - 19 face mask detector uses or owns Facemasknet, deep learning techniques to successfully test whether a person is with wearing a face mask or not. The manuscript presents three-class classification namely person is wearing a mask, or improperly worn masks or no mask detected. Using our deep learning method called Facemasknet, we got an accuracy of 98.6 %. The Facemasknet can work with still images and also works with a live video stream. Cases in which the mask is improperly worn are when the nose and mouth are partially covered. Our face mask identifier is least complex in structure and gives quick results and hence can be used in CCTV footages to detect whether a person is wearing a mask perfectly so that he does not pose any danger

to others. Mass screening is possible and hence can be used in crowded places like railway stations, bus stops, markets, streets, mall entrances, schools, colleges, etc. By monitoring the placement of the face mask on the face, we can make sure that an individual wears it the right way and helps to curb the scope of the virus.

Face mask detection has been accomplished by adopting Deep Learning techniques. We have designed our project into two phases: training face mask detector and implementing face mask detector. Figure 2 depicts the training and detection phase of our face mask detect or model. The dataset is loaded for the model to be trained and the model is serialized in the training phase. Further, the trained model is loaded, the faces are detected in images and video streams and then the region of interest (ROI) is extracted. Finally, the face mask detector is applied and the images or faces in the video streams are classified as with a mask, improperly worn mask, without a mask. The green and yellow rectangular frame individually interpret the detected face and mask. The dataset consisted of 15 images of improperly worn masks, 10 masked images, and 10 images without a mask. We have used Matlab programming to build our facemask detector model.

[5] The corona virus COVID-19 pandemic is causing a global health crisis so the effective protection methods is wearing a face mask in public areas according to the World Health Organization (WHO). The COVID-19 pandemic forced governments acrossthe world to impose lockdownsto prevent virustransmissions. Reportsindicate that wearing facemasks while at work clearly reduces the risk of transmission. An efficient and economic approach of using AI to create a safe environment in a manufacturing setup. A hybrid model using deep and classical machine learning for face mask detection will be presented. A face mask detection dataset consists of with mask and without mask images , we are going to use OpenCV to do real-time face detection from a live stream via our webcam. We will use the dataset to build a COVID-19 face mask detector with computer vision using Python, OpenCV, and Tensor Flow and Keras. Our goal isto identifywhether the person on image/video stream is wearing a face mask or not with the help of computer vision and deep learning.

The proposed system focuses on how to identify the person on image/video stream wearing face mask with the help of computer vision and deep learning algorithm by using the OpenCV, Tensor flow, Keras and PyTorch library

The flow to identify the person in the webcam wearing the face mask or not. The process is two-fold. One is to identify the faces in the webcam and second is to classify the faces based on the mask.

We used OpenCV, tensor flow, Keras , Pytorch and CNN to detect whether people were wearing face masks or not. The models were tested with images and real-time video streams. The accuracy of the model is achieved and, the optimization of the model is a continuous process and we are building a highly accurate solution by tuning the hyper parameters

1.6 Organization of the Report

CHAPTER 1 Introduction	Introduces the concept of Machine Learning, Deep Learning, Streamlit concept and Computer Vision Techniques, along with the technical advancements and present applications and features. Purpose and objective of current system are also specified.
CHAPTER 2 Analysis	Previous work in the current field has been analyzed and reviewed to see the possible shortcomings that are overcome in the proposed system.
CHAPTER 3 Design	The functional and non-functional requirements of the system are detailed here along with the details of the specific software and modules that has been utilized.
CHAPTER 4 Implementation	The architecture and algorithm of the system has been explained along with how the flow of the control is to be done among the techniques.
CHAPTER 5 Testing	Each specific module of the system is implemented and the systemis integrated to perform as a whole, along with use case diagrams, and class diagrams and UML diagrams.

CHAPTER 2

ANALYSIS

2.1 Existing System

2.1.1 Description

The corona virus is causing a huge cause in the health crisis and the effective way to stay from it is wearing a mask in public and crowded areas according to the World Health Organization (WHO). The government has imposed lockdown to reduce the effect of virus. An efficient way to save the environment was to use Machine learning models. To detect face masks using images, there are machines which just process the image and provides an outcome of person wearing a mask or not wearing a mask. There is no proper software where an image browsing and live stream video is implemented. In the Streamlit there is no option for adding a live webcam option.

2.1.2 Drawbacks

The existing system has the following drawbacks:

- There are suitable models which process the detection without much accuracy and efficiency.
- After extracting the images, it consumes time to give results.

2.2 Proposed System

2.2.1 Description

The proposed system focuses on how to identify the person on image/video stream wearing face mask with the deep learning algorithm by using OpenCV, Tensorflow and Keras libraries. A web frame named Streamlit will be added with responsive features to implement Graphical User Interface. Our aim is to find whether a person on an image or video stream is wearing a face mask or not by the help of deep learning. Many algorithms are being devised using convolutional architectures to make the algorithms as accurate as possible.

- **First option** is image browsing where an image is uploaded, tested and resulted in if a person is wearing a mask or not. If there is a green dimensional the bounding box with high confidence score then a person is wearing a mask and outputs animation of balloon pops up. If there is red dimensional bounding box with high confidence score then a person is not wearing a mask and gives an alert "Please wear a mask and protect yourself and your family from Covid-19"
- **Second option** is Live stream video, in live stream video it will give live results of a person wearing a mask or not with high confidence score. If there is a green dimensional bounding box with high confidence score then a person is wearing a mask and if there is red dimensional bounding box with high confidence score then a person is not wearing a mask.
- **Third option** is voice alert which tells us about the rules to be followed when we got out.
- **Fourth option** gives the status of the world like how many cases Covid has been reported, displays the place with highest covid cases.

2.2.2 Advantages

- Safe guard work-spaces and working environment.
- Reduces number of infections and transmission of viruses.
- Improve business operational efficiency & productivity.
- Adherence to social-distancing & protection compliance.
- Can be used to deal with the specific hazards in construction, laboratory environments.

2.3 Requirements Specifications

Requirements specifications are a central artifact in most software and hardware development processes. They capture the goals of the software and hardware to be developed and constitute the connection between the customer/user and the developers. Many call the specifications the determining part for project success or failure.

2.3.1. Functional Requirements

Functional requirements are a set of statements that define the functions that the system should provide, how the system should react to particular input and how the system should behave in particular situations. In some cases, the functional requirements may also specify what the system should not do. A function is described as a set of inputs, the behaviour, and outputs. The functional requirements of the proposed system are as follows:

- Open the Streamlit using the anaconda prompt.
- After the Steamlit opens, options can be viewed such as

1. Browsing of images
2. Live video stream.
3. Voice Speech
4. Covid-19 status around the world.

- If the browsing of photos option is used then, pictures can be browsed from your system where the picture can consist of a person wearing a mask or not.
- After the picture is uploaded, the picture will be processed for testing and then gives the accurate results.
- After the testing process, it will detect whether a person is wearing a mask or not with a confidence score.
- If the person is wearing a mask then an animation is shown up with confidence score, if the person is not wearing a mask then an alert is shown with confidence score.
- If Live video stream option is used then, the system will automatically take the laptop web camera and will detect if the person is wearing mask or not with a confidence score.

- Voice Speech: Here information related to covid-19 is given.

- Covid status: This link will provide us status of covid situation in and around the like how many cases covid has been reported, displays the place with highest covid cases.

2.3.2 Non - Functional Requirements

Non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Non-Functional Requirements are the constraints on the services or function offered by the system. Non-functional necessities are those who don't directly have an effect on the functioning of the system however have an effect on, the performances of the system. The non-functional requirements of the proposed system are as follows:

1. **Accuracy** is defined as the percentage of correct predictions for the test data.
2. **Performance** defines how fast a software system or its particular piece responds to certain users' actions under certain workload.
3. **Efficiency** the user concern for how fast the system functions, how efficiently the system takes in inputs and processes outputs, and how much can be processed at a time.
4. **Usability** requirements are documented expectations and specifications designed to ensure that a product, service, process or environment is easy to use.
5. **Maintainability** is how easy it is for a system to be supported, changed, enhanced, and restructured over time.

2.3.3 Hardware Requirements

Device	PC, Laptop
Wi-Fi Connectivity	2.4 and 5 GHz
Operating System	Windows 10
Camera	Required to a good camera or a min. of 5 megapixel

- **Web-Camera:**

A webcam is a video camera that feeds or streams an image or video in real time to or through a computer network, such as the Internet. Webcams are typically small cameras that sit on a desk, attach to a user's monitor, or are built into the hardware.

Figure 2.3.3: Web-Camera

2.3.4 Software Requirements

- **Windows:** Microsoft Windows, commonly referred to as Windows, is a group of several proprietary graphical operating system families, all of which are developed and marketed by Microsoft. Each family caters to a certain sector of the computing industry.

Figure 2.3.4 Windows

- **Anaconda Navigator**: It is a free and open source distribution of python for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux and macOS.
- **Jupiter notebook**: Project Jupyter is a non-profit organization created to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages".
- **User Interface**: Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science.
- **Visual Studio Code**: It is a dual-licensed source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.
- **Google Colab**: It is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.
- **Programming Language**: Python 3.x
- **Python Libraires**: NumPy, OpenCv, tensorflow, keras, Os, MobileNetV2, Caffee.

CHAPTER 3

DESIGN

3.1 System Design

System designs is the process of defining the architecture, Modules, Interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of the systems theory to productdevelopment.

3.1.1 Design Overview

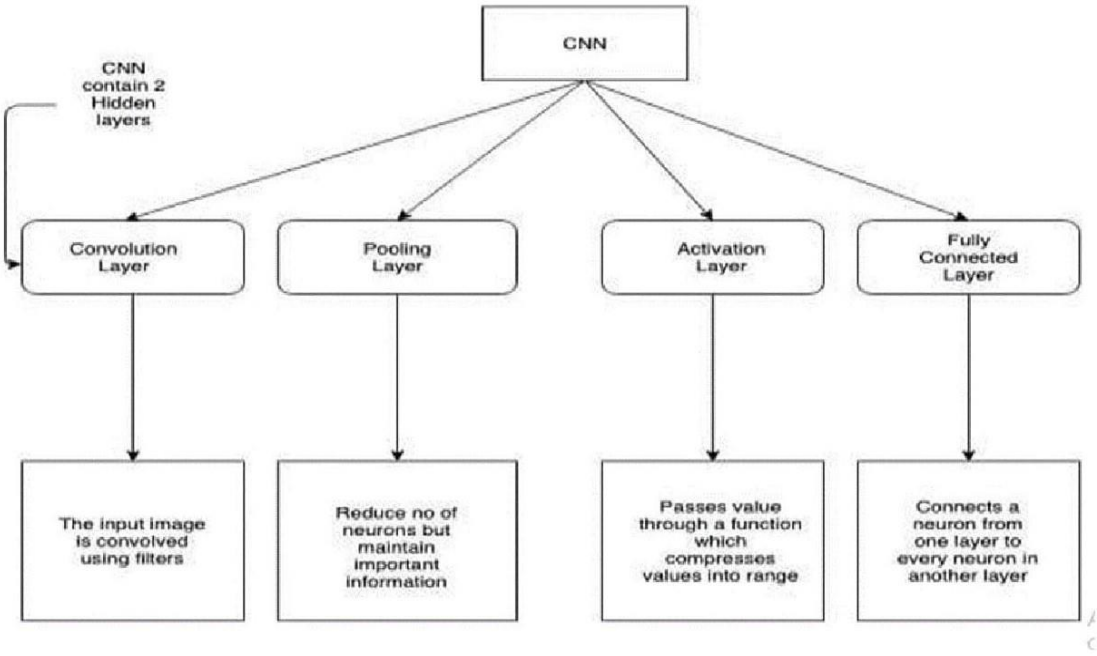


Figure 3.1: CNN Block Diagram

As shown in Figure. 3.1, the model of CNN configuration for bird identification utilized a stack of convolutional layers comprising an input layer, two FC layers, and one final output softmax layer. Each convolutional layer comprised (a) 5 × 5 convolution, (b) BN, (c) ReLU activation, and (d) pooling layers. This section explains how to construct an optimized CNN model, why the parameters and hyper parameters must betuned before training, the total

number of convolutional layers, the size of the kernels for all relative convolutional layers, and the likelihood of retaining the anode during dropout regularization for the dataset.

3.2 Detailed Design

3.2.1 Use Case diagram

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionalityof a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system. Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities.

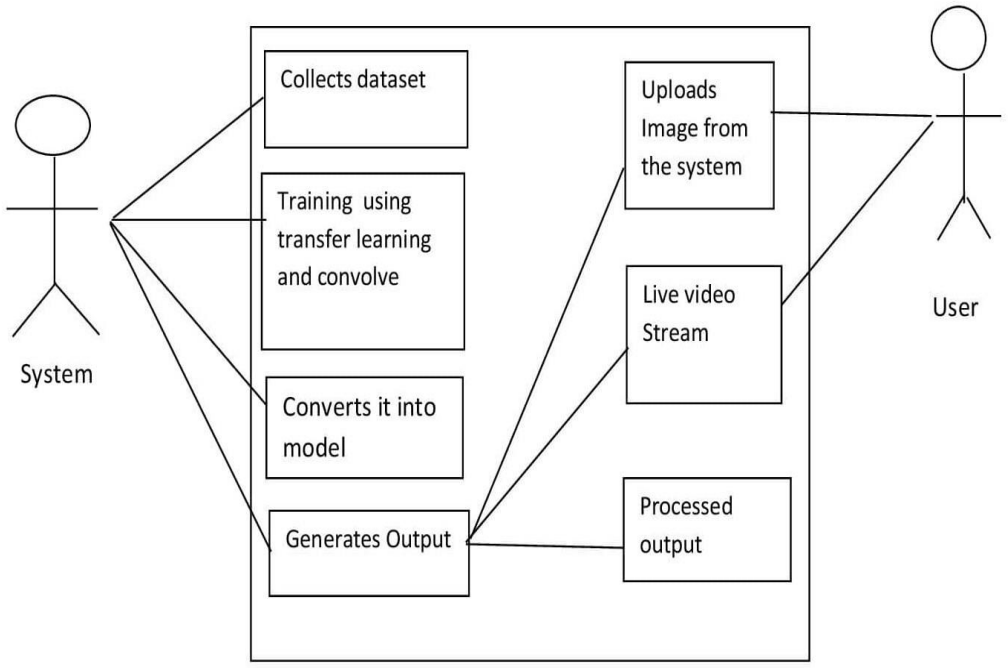


Figure 3.2.1 Use Case Diagram

The Figure. 3.2 depicts the use case diagram of the system where the user just has to upload an image eitherfrom the system or can upload it from any website as well. Then rest all the work will be done by the system where the system initializes by extracting the features, then pre-

processes the image, then segmentation of that image, then recognition and then generation of score sheet**.**

3.2.2 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not onlyused for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class andalso the constraints imposed on the system. The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints.It is also known as a structural diagram. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction. Below is Figure 3.2.2.

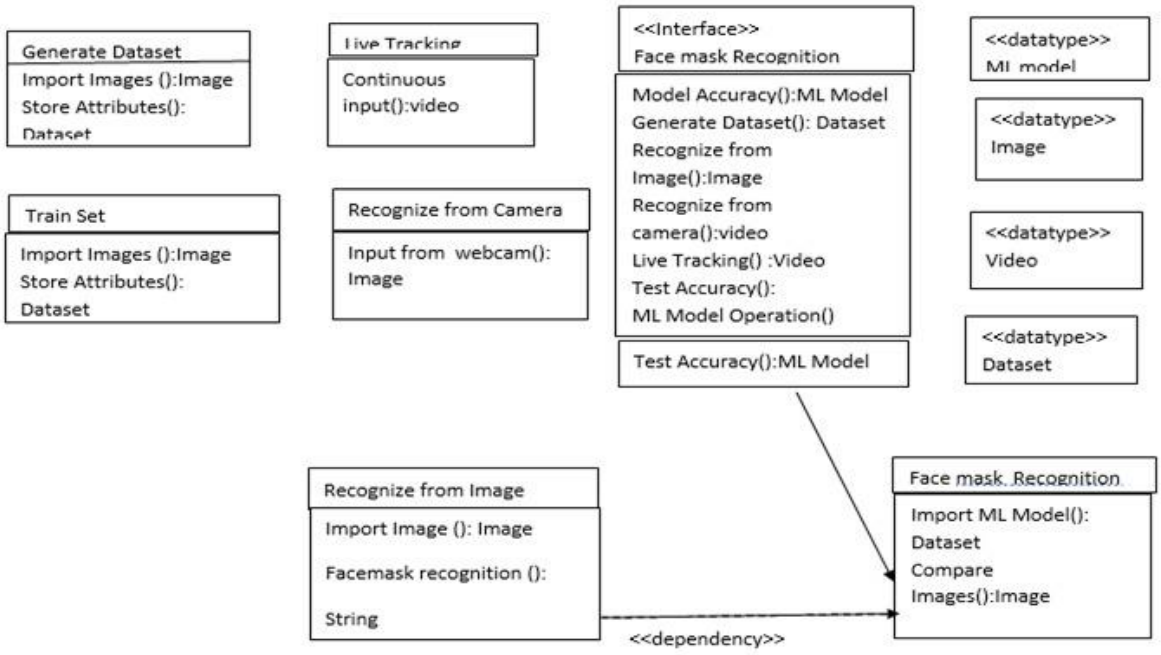


Figure 3.2.2 Class Diagram

Figure 3.2.3 helps us to know about the class diagram of the image detection and classification process. Thedifferent classes are:

- **Face mask Detection**: Is an abstract class. The abstract class is created to call out the class anywhere with the help of its instance. The class consist of functions such as camera Recognition(), trainSet(), testAccuracy() and live Tracking() whereeach function performs a particular task. cameraRecognition() tries to capture the image which the user wants and sends it directly as the input image. trainSet() is a function of type dataset which contains the information totrain the model. testAccuracy() is a function which is responsible for the accuracy with which the model predictsthe output of the given input data.
- **Generate Dataset**: Is the class which contains two function importImages() and storeAttributes(). The function is responsiblefor importing the images from the user and making the dataset of it and the other is responsible for storing the attribute of the images which will try to identify the particular image respectively.
- **Image:** A class which is responsible for containing the image which is the input.
- **Data Set**: A class which contains the dataset of the form both the training and the testing.
- **Live Tracking**: This class is responsible how the camera works and how the pictures taken from the camera work. The module has an association with the LiveTracking class where the LiveTracking helps to identify the input image. Facemask Recognition consist of Recognize from camera() and Recognize from video() as the function which is of type Image and video respectively. The function Recognize from camera() consist of taking the picture as input and Recognize from video() tries to find the information present in video image in real time scene.
- **Face Mask Recognition** : A class responsible for extracting the features of the image taken or of the image which is contained in the dataset. The class has the dependency of the class Face mask Recognition which is responsible for sending the image for image extraction process. The function present in the class i.e Recognize from Images() and Recognize from video() functions import imagesfor image extraction process and recognizes them.
- **TrainSet** : A class which is responsible for training the model according to the input which it has received, form thedata which it has collected from the image extraction process. Function such as ImportImages() and storeAttributes() helps the class in this process.

- **Recognize from image**: A class responsible for the detection of the object given in the frame of reference. The class has associationwith the image Recognition class where the class helps the Face mask Detection class to identify the images. The TrainSet class also helps the ObjectDetection Class as the class contains all the learned features of the images inthe dataset. The function importDataset() and compareImages() helps the class to generate the output.
- **TestAccuracy**: The class which tries to predict the accuracy of the model. It has association with the Face mask Recognition class where the class helps the TestAccuracy class to predict the accuracy of the model and the output which it has generated. The function testAccuracy() and operation() helps the class to predict the accuracy of the output generated. To test the accuracy, the score sheet is generated where the comparison of the image with all the known classification of the human face wearing mask is being done and detect if person is wearing mask or not.

3.2.3 State Chart Diagram

A state chart diagram is a type of diagram used in computer science and related fields to describe the behaviourof systems. State diagrams require that the system described is composed of a finite number of states; sometimes, thisis indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics. State diagrams are used to give an abstract description of the behaviour of a system. This behaviour is analysed and represented by a series of events that can occur in one or more possible states. Hereby "each diagram usually represents objects of a single class and track the different states of its objects through the system".

A state chart diagram shows the behaviour of classes in response to external stimuli. Specifically, a statediagram describes the behaviour of a single object in response to a series of events in a system. Sometimes it's also known as a Harel state chart or a state machine diagram. This UML diagram models the dynamic flow of control from state to state of a particular object within a system.

State chart diagrams are good at describing the behaviour of an object across several use cases. State diagrams are not very good at describing behaviour that involves a number [Grab your

reader's attention with a great quote from the document or use this space to emphasize a key point. To place this text box anywhere on the page, just drag it.]of objects collaborating. As such, it is useful to combine state diagrams with other techniques. For instance, interaction diagrams are good at describing the behaviour of several objects in a single use case.

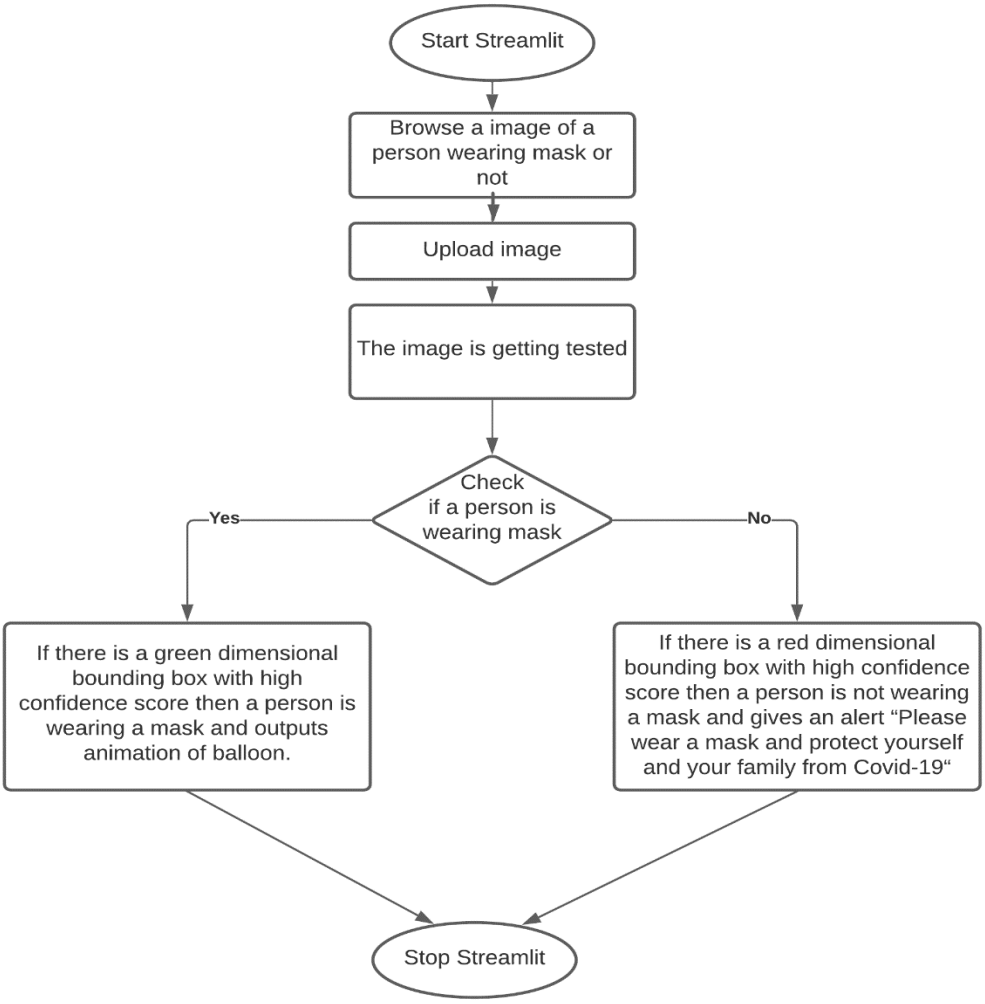


Figure 3.2.3(a)Image State Chart Diagram

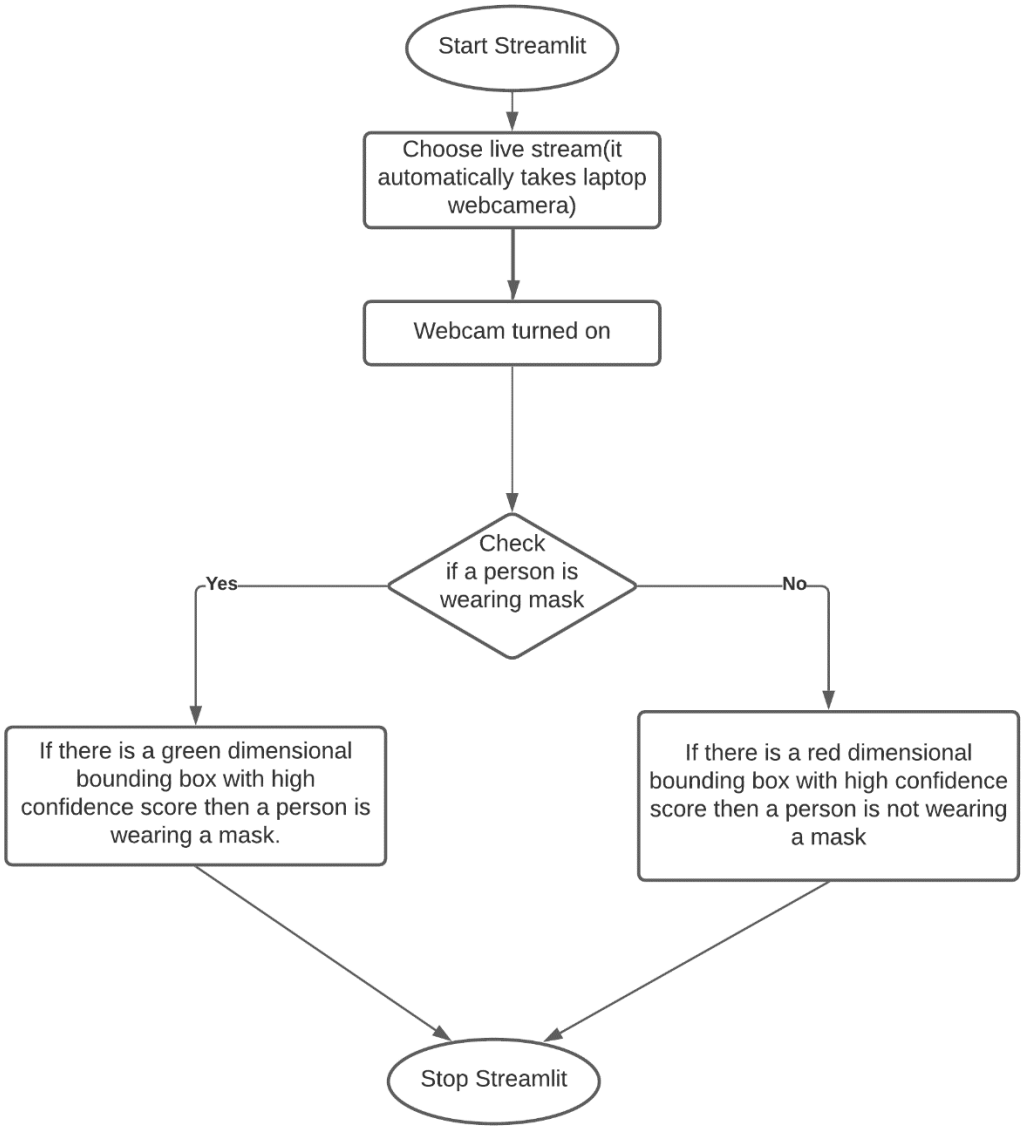


Figure 3.2.3(b): Video State Chart Diagram

3.2.4 Sequence Diagram

A sequence diagram shows object interaction arranged in time sequence. It depicts the objects and classesinvolved in the scenario and the sequence of messages exchanged between the object needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realization in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages aresent and when. They are a popular dynamic modelling solution in UML because they specifically focus on lifelines, or the processes and objects that live simultaneously.

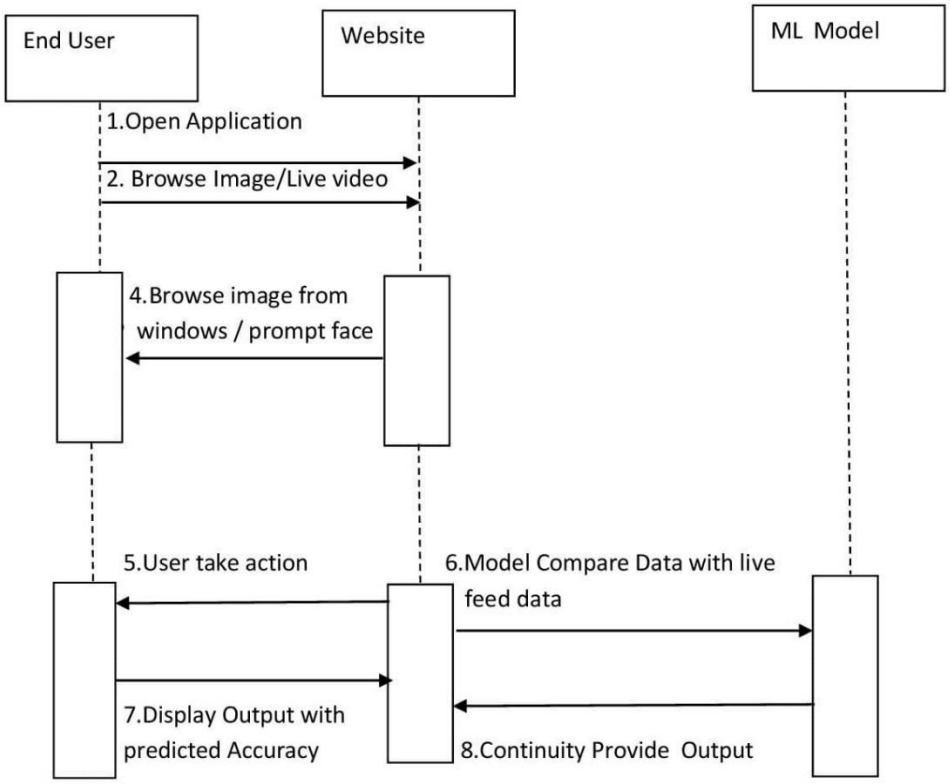


Figure 2.3.4 Sequence Diagram

3.2.5 Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activitycan be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviourof the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart.Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

Next in Figure 3.2.5 the activity diagram of the system is given where step by step procedures are followed todetect the species of the bird. Here the user is supposed to upload an image, if any error is occurred or any interruption happens, the process will stop right at that moment and the user is supposed to restart procedure. If the image is successfully uploaded, then the system will take the required steps to identify the species of the birdappearing in the image. The process involves pre-processing, feature extraction, segmentation, recognition and at last output is generated by the generation of score sheet.

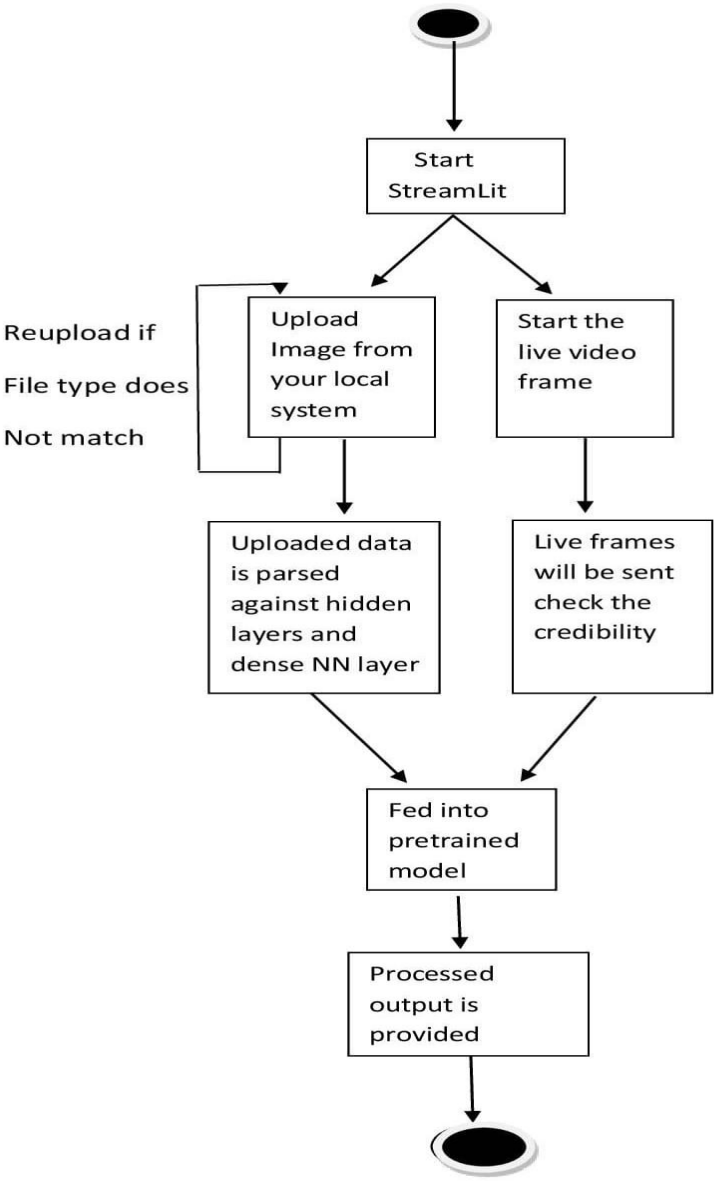


Figure 3.2.5: Activity Diagram

CHAPTER 4

IMPLEMENTATION

4.1 Main Execution File

ui.py file

```
import streamlit as st
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
import numpy as np
import cv2
import os
import webbrowser
from PIL import Image
import tkinter as tk

st.set_option('deprecation.showfileUploaderEncoding', False)
@st.cache(hash_funcs={cv2.dnn_Net: hash})
def load_face_detector_model():
    prototxt_path = os.path.sep.join(
        ["face_detector", "deploy.prototxt"])
    weight_path = os.path.sep.join(
        ['face_detector', 'res10_300x300_ssd_iter_140000.caffemodel'])

    net = cv2.dnn.readNet(prototxt_path, weight_path)

    return net
```

```
@st.cache(allow_output_mutation=True)
def load_mask_model():
        mask_model = load_model("mask_detector.model")
        return mask_model

net = load_face_detector_model()
model = load_mask_model()
confidence_selected = st.sidebar.slider('Select a confidence range', 0.0, 0.1, 0.5, 0.1)

def detect_mask(image):
        image = cv2.imdecode(np.fromstring(image.read(), np.uint8), 1)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        orig = image.copy()
        (h, w) = image.shape[:2]
        blob = cv2.dnn.blobFromImage(image, 1.0, (300, 300), (104.0, 177.0, 123.0))
        net.setInput(blob)
        detection = net.forward()

        for i in range(0, detection.shape[2]):
                confidence = detection[0, 0, i, 2]
                if confidence > confidence_selected:
                        box = detection[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 = image[startY:endY, startX:endX]
                        face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB
                        face = cv2.resize(face, (224, 224))
                        face = img_to_array(face)
                        face = preprocess_input(face)
                        face = np.expand_dims(face, axis=0)
```

```
                (mask, withoutMask) = model.predict(face)[0]
                label = "Mask" if mask > withoutMask else "No Mask"
                color = (0, 255, 0) if label == "Mask" else (255, 0, 0)
                label = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)
                cv2.rectangle(image, (startX, startY), (endX, endY), color, 6)
        return image, label


def about():
    st.write(
                '''
        **This project workflow follow 3 phases**

        1. Loading the Face Mask Classifier model
        2. Detect Faces in the image
        3. Extract each Face Region of Interest(ROI)
        4. Apply face mask classifier to each face ROI to determine 'mask' or 'No mask'
    ''')
    st.write(
        '''
    **Guide**:- **Dr**. **Asha P N**
    **Project Done By** :
    1SG17IS003-*Aditya N*
    1SG17IS008-*Akshay P*
    1SG17IS012-*Arpita Prakash Paramaj*
    1SG17IS027-*Deepak K*
    ''')
```

```
def main():
    st.title("Face Mask Detector App :mask:")
    st.write("*Using the Python, Tensorflow and OpenCV*")
    activities = ['Home', 'About']
    choice = st.sidebar.selectbox("Hey, What do want to do", activities)
    if choice == "Home":
        st.write("Go to the about Page to learn more about this Project")
        image_file = st.file_uploader("Upload image", type=['jpeg', 'jpg', 'png','jfif'])
        if st.button('Mask Detecting Webcam '):
            os.system('python D:\Face-Mask-Detector-Using-OpenCV-and-Python-master\detect_mask_video.py')
        if st.button('Click here to listen the alert audio'):
            st.audio("https://www.cdc.gov/coronavirus/2019-ncov/downloads/communication/PSA_English_How_to_Wear_a_Mask.mp3")
        url = 'https://www.covid19india.org/'
        if st.button('Click here to know the updates'):
            webbrowser.open_new_tab(url)

        if image_file is not None:
            st.sidebar.image(image_file, width=140)
            if st.button("Process"):
                with st.spinner('Your photo is getting processed 😁...'):
                    image, label = detect_mask(image_file)
                    if label[:4]=="Mask":
                        st.image(image, width=200)
                        st.success(label)
                        st.balloons()
                    elif label[:7]=="No Mask":
                        st.image(image, width=200)
                        st.error(label)
```

```
                                        st.write('Please wear a *Mask* and protect yourself
                                        and your family From Covid-19')

    elif choice == 'About'
            about()



if __name__ == "__main__":
    main()
```

4.2 Execution File for Video

detect_mask_video.py

```
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, (300, 300),(104.0, 177.0, 123.0))
    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 > args["confidence"]:
                        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)

ap = argparse.ArgumentParser()
ap.add_argument("-f", "--face", type=str, default="face_detector",
         help="path to face detector model directory")
ap.add_argument("-m", "--model", type=str, default="mask_detector.model",
        help="path to trained face mask detector model")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
        help="minimum probability to filter weak detections")
args = vars(ap.parse_args())
```

```
print("[INFO] loading face detector model...")
prototxtPath = os.path.sep.join([args["face"], "deploy.prototxt"])
weightsPath = os.path.sep.join([args["face"],"res10_300x300_ssd_iter_140000.caffemodel"])
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)
print("[INFO] loading face mask detector model...")
maskNet = load_model(args["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=400)
	(locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)
	for (box, pred) in zip(locs, preds):
		(startX, startY, endX, endY) = box
		(mask, withoutMask) = pred
		label = "Mask" if mask > withoutMask else "No Mask"
		color = (0, 255, 0) if label == "Mask" else (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) & 0xFF
	if key == ord("q"):
	break
cv2.destroyAllWindows()
vs.stop()
```

CHAPTER 5

TESTING

The philosophy behind testing is to find errors. The common view of testing is that it is performed to prove that there are no errors in a program. However, it is virtually impossible to prove that no program will be free and clear of errors. Therefore, the most useful approach and practical approach is with the understanding that testing is the process of executing a program with explicit intention of finding errors that is, making the program fail.

5.1 Unit Testing

Unit testing is the testing of each module. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as 'module testing'. The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields

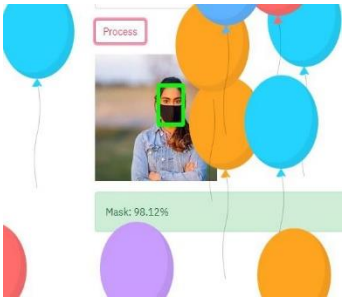
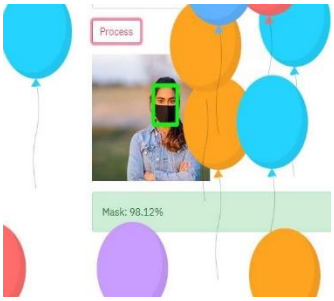
In our there are 2 modules to check one for browsing of images and live video.

Test Case ID	**Objective/Input**	**Expected Output**	**Actual Output**	**Status (Pass/Fail)**
01	Browsing through captured Pictures/Images	Detect and display the output with desired colored borders, percentage through detected image	Displays output with varying percentage (percentage varied is nearest value)	**PASS**
02	Browsing through Web Camera/External Camera (live)	Detects and displays the desired colored border and percentage on screen through live video	Displays output on screen with desired colored borders with varying percentage nearest to accurate value	**PASS**

5.2 Integration Testing

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance.

In our system we are checking if image of a person with a mask is detected then animation is displayed, if the image of person with no mask is detected then an alert is displayed and if wrong image is uploaded.

Test CaseID	Objective/Input	Expected Output	Actual Output	Status (Pass/Fail)
1				PASS
2				PASS
3				PASS

5.3 System Testing

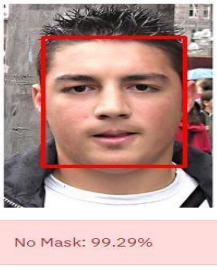
System testing is the type of testing to check the behaviour of a complete and fully integrated software product based on the software requirements specification (SRS) document. The main focus of this testing is to evaluate Business / Functional / End-user requirements.

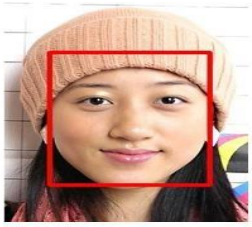
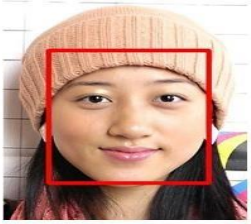
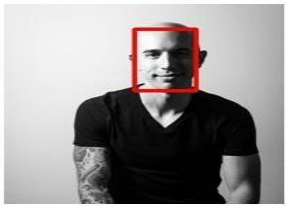
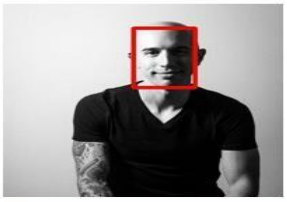
This is black box type of testing where external working of the software is evaluated with the help of requirement documents & it is totally based on Users point of view. For this type of testing do not required knowledge of internal design or structure or code.

This testing is to be carried out only after System Integration Testing is completed where both Functional & Non-Functional requirements are verified.

In the integration testing testers are concentrated on finding bugs or defects on integrated modules. But in the Software System Testing testers are concentrated on finding bugs/defects based on software application behaviour, software design and expectation of end user.

a) In Software Development Life Cycle the System Testing is perform as the first level of testing where the System is tested as a whole.
b) In this step of testing check if system meets functional requirement or not.
c) System Testing enables you to test, validate and verify both the Application Architecture and Business requirements.
d) The application or System is tested in an environment that particularly resembles the effective production environment where the application/software will be lastly deployed.
System as whole is connected and tested.

Test CaseID	Objective/Input	Expected Output	Actual Output	Status (Pass/Fail)
1				PASS

2				**PASS**
3				**PASS**
4				**PASS**
5				**PASS**
6				**FAIL**

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENTS

6.1 Conclusion

The Real Time Face Mask Detection Using Deep Learning for detecting humans wearing face mask and not wearing is developed in python programming language by making use of most of module present in the language. The idea behind using python programming language was to make the code look simple in order to make ease the difficulty level in modifying the code and maximize the use of the different module present in the python programming language. The system can be used efficiently by almost every company who cannot afford to manual verification systems due to their high price.

As the technology are blooming with emerging trends the availability so we have novel face mask detector which can possibly contribute to public healthcare. The architecture consists of MobileNet as the backbone and it can be used for high and low computation scenarios. In order to extract more robust features, we utilize transfer learning to adopt weights from a similar task face detection, which is trained on a very large dataset. We used tensor flow, Keras, OpenCV and CNN to detect whether people are wearing face masks or not. The models had been tested with images and real-time video streams using a webcam. The accuracy of the model is achieved and the optimization of the model is a continuous process and we are building a highly accurate solution by tuning the hyper parameters. This specific model could be used as a use case for edge analytics.

We can provide the superpower to ML models by providing the IT Ops teams support which will lead to continuous training, testing, managing, monitoring, deploying Machine Learning Models in Production and to govern their use in production environments. We will be solving one of the major problem which is being faced by Machine learning Models by Building an End-to-End fully automated model with the OPS which we can call as MLOPS [Mops] with the Mops we can automate the entire Process from the time of data collection, data pre-processing, building, testing like analysing metrics such as R2 Score, Precision, Recall, Accuracy and deploying Our Machine Learning Mode

6.2 Future Enhancements

As we have more manual work from the training to testing phase, we are trying to achieve automation. Working on Building a deep Learning Docker Container with all the ML modules implemented inside it. Along with that to achieve Complete automation we are working with Jenkins to improve our model accuracy with frequent code Deployments.

Replacement of laptop with a compact model with camera attached for video capture the plan is to replace laptop with a workable model/device such as Raspberry Pi which needs to be installed at place of inspection/ site. Raspberry Pi models are compact in size. This will reduce installation complexities by 90%. This device/ model will be attached to a laptop only for execution of commands with help of LAN cable. As such commands can be run by a remote operator also and the requirement of laptop present at the site is negated.

Removal of laptop for all purpose. Execution only with help of model There will not be any constraint of laptop for executing any command to run the model either at the site or at any remote location. The device which is comparable to the size of a credit card with attached camera and loaded with necessary algorithms/code would meet the requirements. There would not be any obligation of trained/ semi trained individuals to run the commands. A normal sentry/ operator should be able to operate the model. These can be done by hosting system in cloud and reduce latency for end user along with removal of above constraints.

SNAPSHOTS

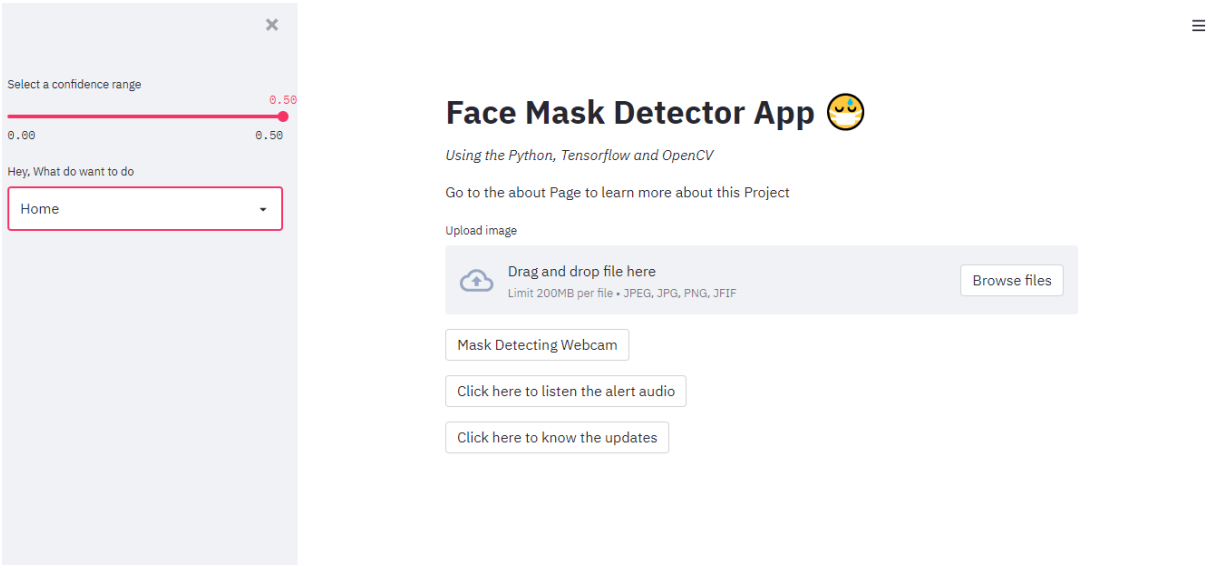


Figure A: Main page

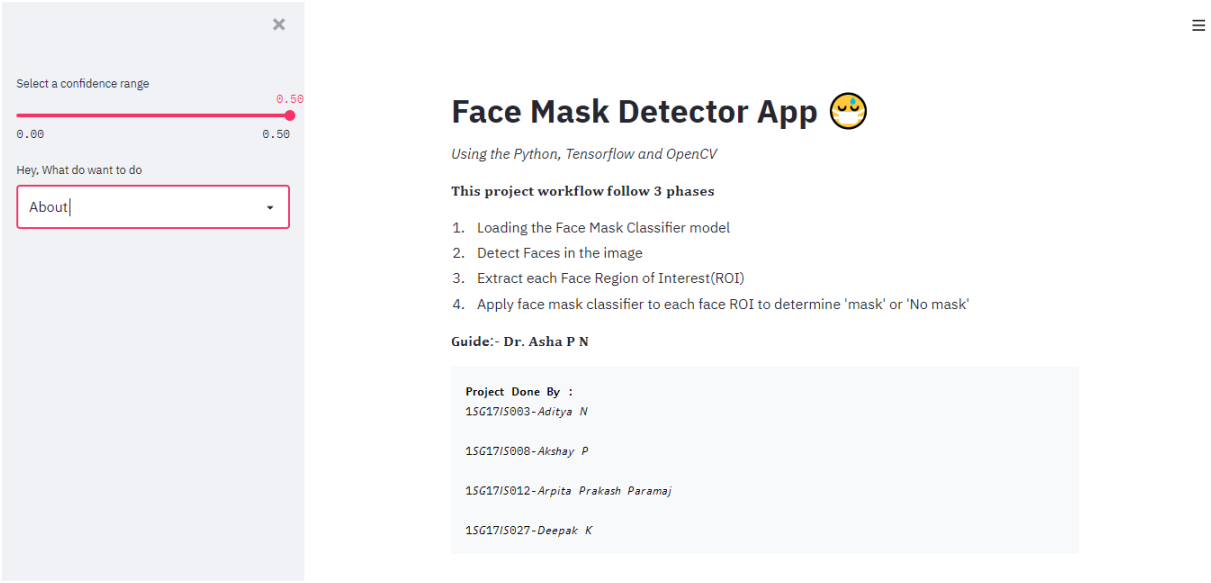


Figure B: About page

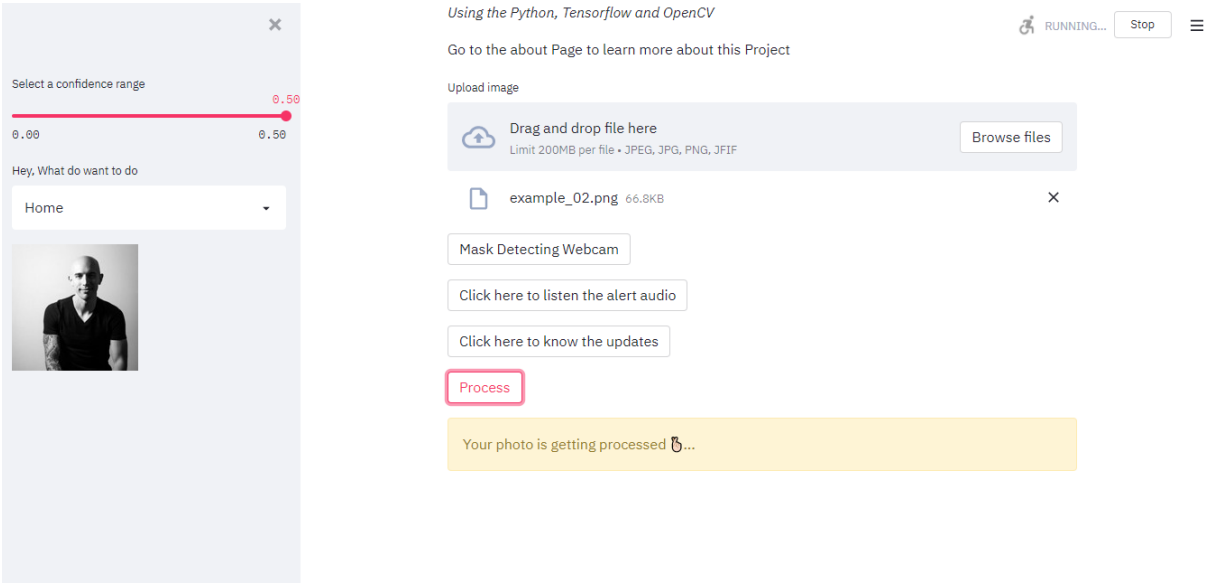


Figure C: Browsing of images option used and selecting a no mask image

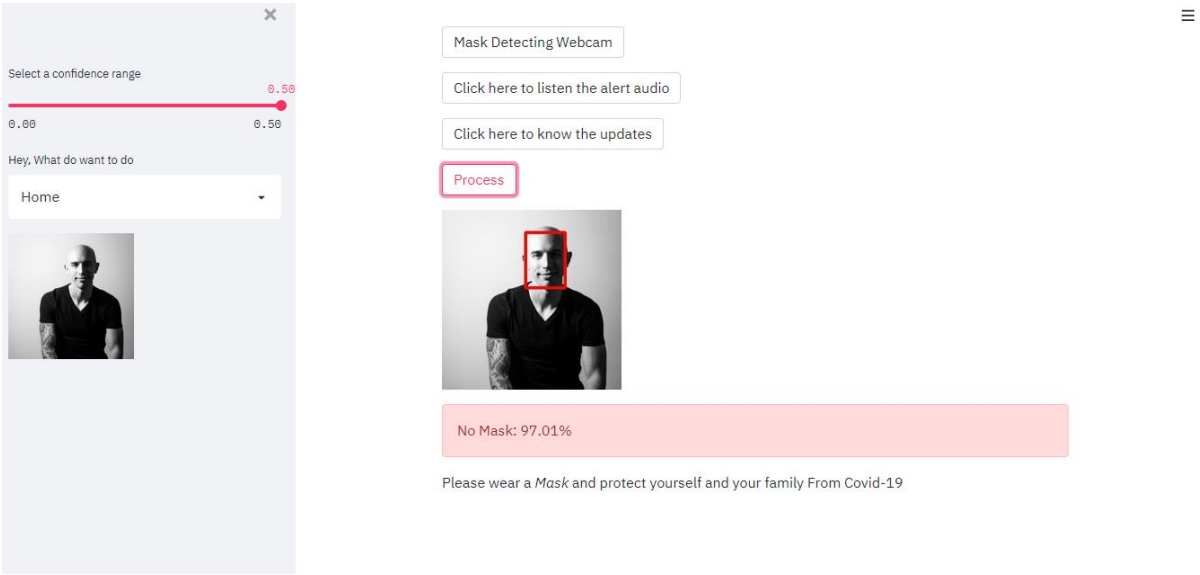


Figure D: After processing the image, output driven as No Mask with a message.

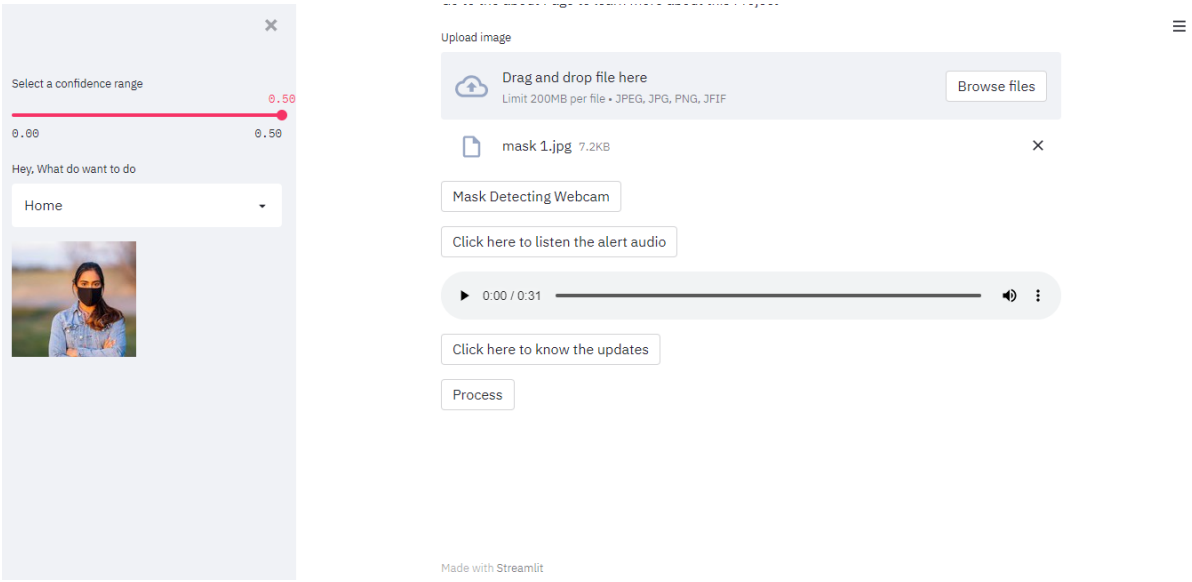


Figure E: Browsing of images option used and selecting a no mask image

Figure F: Mask image processed and output driven as Mask with a balloon animation

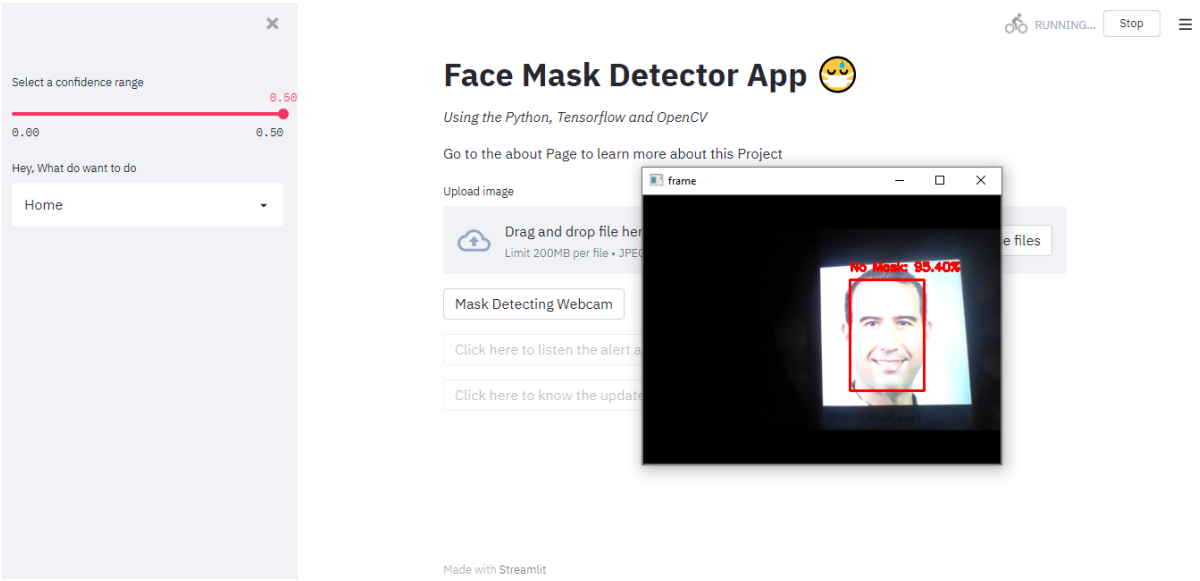


Figure G: Live video stream option used; output driven as No mask

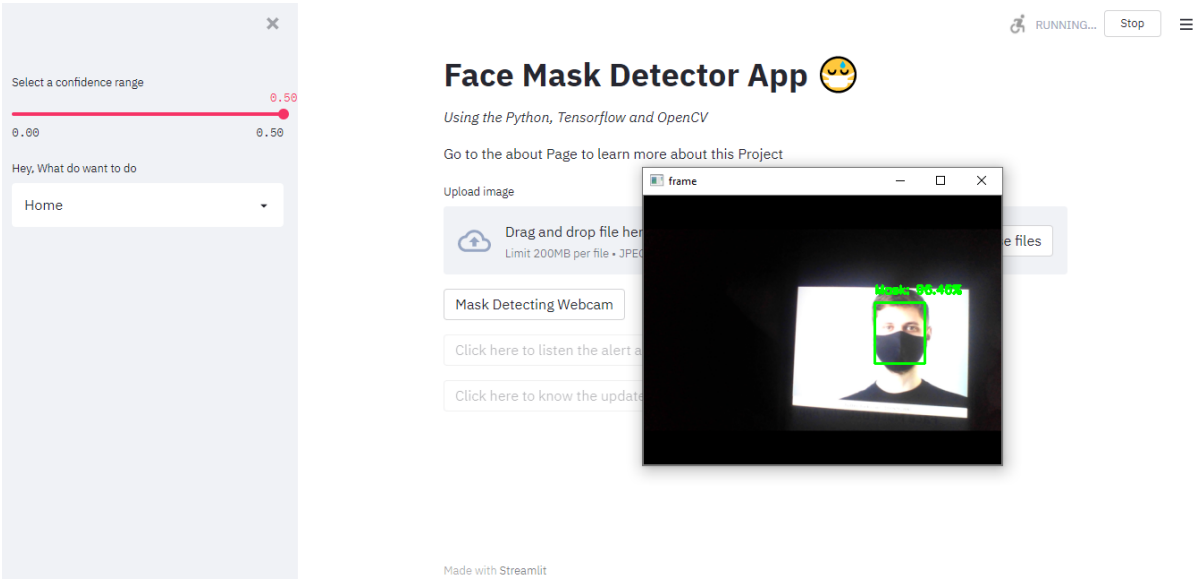


Figure H: Live video stream option used; output driven as Mask

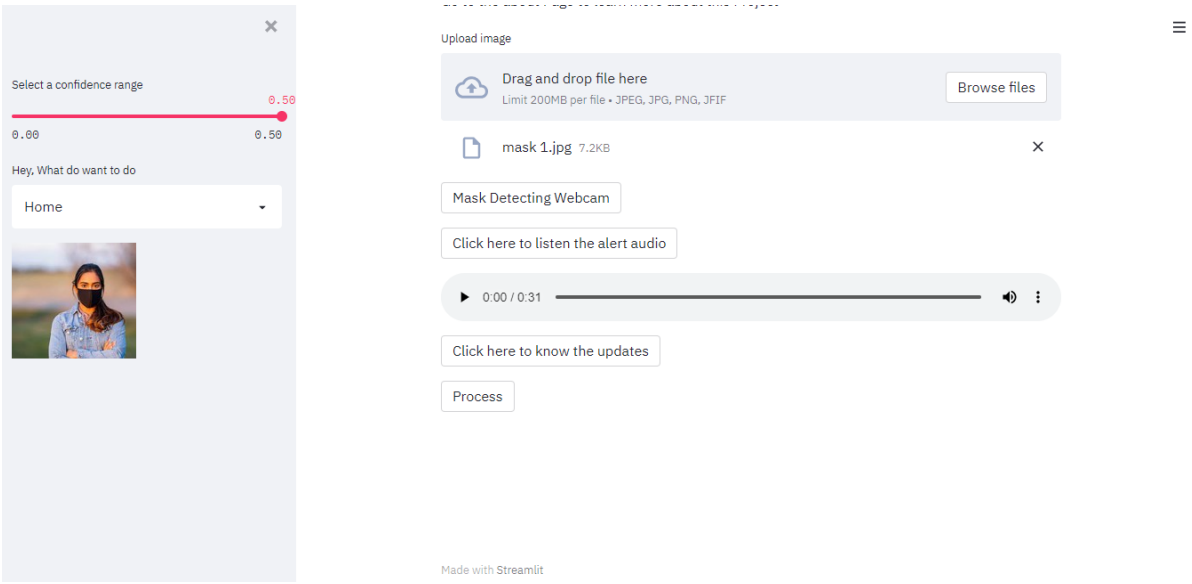


Figure I: Audio Option

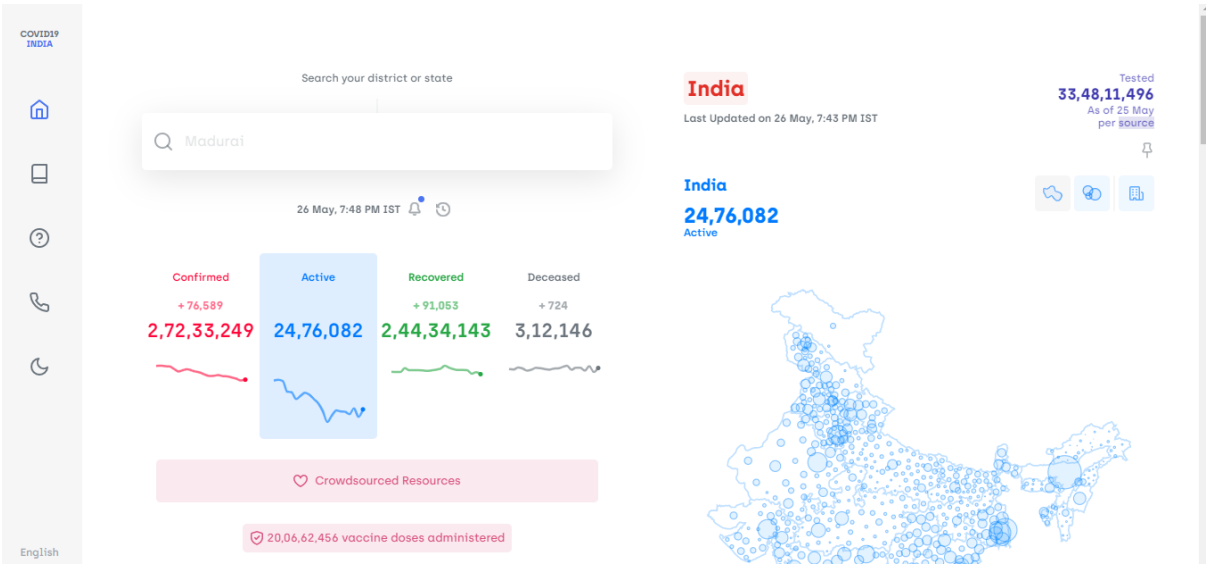


Figure J: Covid-19 status option

ANNEXURE A

Glossary ❖

- With the help of this project we will be able to propose a Real Time Face Mask Detection using Deep Learning.
- Here we import several packages from python libraries whose instances can be used to achieve the required tasks based on the input given by the user. Each module has its own functionality that helps the system to integrate them with help of few more modules and run and give desired output. The different module used in the system are:
    1. OpenCV
    2. Numpy
    3. StreamLit
    4. OS
    5. Keras
    6. MobileNetV2
    7. Caffee with face mask detection
    8. Python3.8
    9: Tensorflow
- With StreamLit as user Interface: We use 4 features in this streamlit. They are:

    1. Image browsing: If the person is wearing a mask then an animation is shown up with confidence score, if the person is not wearing a mask then an alert is shown with confidence score.

    2. Live Video Stream: If Live video stream option is used then, the system will automatically take the laptop web camera and will detect if the person is wearing mask or not with a confidence score.

    3. Voice Speech: Here information related to covid-19 is given.

    4. Covid status: This link will provide us status of covid situation in and around the like how many cases covid has been reported, displays the place with highest covid cases.

ANNEXURE B

Acronyms

WHO	World Health Organization
SARS	Sever Acute Respiratory Syndrome
MERS	Middle East Respiratory Syndrome
ML	Machine Learning
SSD	Solid-State Drive
VGG	Visual Geometry Group
ROI	Region of Interest
CNN	Convolution Neural Network
BN	Batch Normalization
FC	Fully Connected
RELU	Rectified Linear Activation Function
UML	Unified Model Language
SRS	Software Requirements Specification

ANNEXURE C

Language Description

Python is a high-level, interpreted, interactive and object-oriented scripting language. It was created by Guido van Rossum, and released in 1991. Python was designed for readability, and has some similarities to the English language with influence from mathematics. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses. Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose. Python has syntax that allows developers to write programs with fewer lines than some other programming languages. Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick. Python can be treated in a procedural way, an object-orientated way or a functional way. Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc). Python can be used on a server to create web applications, alongside software to create workflows, connect to database systems, handle big data and perform complex mathematics, rapid prototyping or for production-ready software development. It provides very high-level dynamic data types and supports dynamic type checking. It supports automatic garbage collection. It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java. The biggest strength of Python is huge collection of standard library which can be used for the following – Machine Learning. ¬ GUI Applications (like Kivy, Tkinter, PyQt etc

BIBLIOGRAPHY

[1] Mohammad Marufur Rahman, Md. Motaleb Hossen Manik, Md. Milon Islam, Saifuddin Mahmud Jong-Hoon Kim An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network.

[2] Shashi Yadav(2020) "Deep Learning based Safe Social Distancing and Face Mask Detection in Public Areas for COVID-19 Safety Guidelines Adherence"

[3] Toshanlal Meenpal, Ashutosh Balakrishnan, Amit Verma (2019) "Facial Mask Detection using Semantic Segmentation"

[4] Madhura Inamdar · Ninad Mehendale "Real-time face mask identification using Facemasknet deep learning network"

[5] Vinitha, Velantina "COVID-19 FACEMASK DETECTION WITH DEEP LEARNING AND COMPUTER"

[6] Retina Face Mask: A Face-mask Detector by Mingjie Jiang , Xinqi Fan ,Hong Yan

[7] Detecting Masked Faces in the Wild with LLE-CNNs by Shiming Ge1 , Jia Li , Qiting Ye , Zhao Luo.

[8] P. A. Rota, M. S. Oberste, S. S. Monroe, W. A. Nix, R. Campagnoli, J. P. Icenogle, S. Penaranda, B. Bankamp, K. Maher, M.-h. Chen et al., "Characterization of a novel coronavirus associated with severe acute respiratory syndrome," science, vol. 300, no. 5624, pp. 1394–1399, 2003.

[9] Z. A. Memish, A. I. Zumla, R. F. Al-Hakeem, A. A. Al-Rabeeah, and G. M. Stephens, "Family cluster of middle east respiratory syndrome coronavirus infections," New England Journal of Medicine, vol. 368, no. 26, pp. 2487–2494, 2013.

[10] Performance Evaluation of Intelligent Face Mask Detection System with various Deep Learning Classifiers by C.Jagadeeswari, M.Uday Theja

[11] Streamlit Documentation Release 0.72.0 for Streamlit web framework.