Mask Detection using Machine Learning

J-comp Final Report

submitted by

SIDDHARTH DAHIYA

18BCE0446

&

ASHISH POUDEL

18BCE2446

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

May 2021

ACKNOWLEDGEMENTS

We take immense pleasure in thanking **Dr. G. Viswanathan**, our beloved Chancellor, VIT University and respected Dean, **Dr. K. Ramesh Babu**, for having permitted us to carry out the project.

We express gratitude to our guide, **Dr. Swathi J.N**, for guidance and suggestions that helped us to complete the project on time. Words are inadequate to express our gratitude to the faculty and staff members who encouraged and supported us during the project. Finally, we would like to thank our ever-loving parents for their blessings and our friends for their timely help and support.

Signature of Students:

Dahiya Ashish

(Siddharth Dahiya) (Ashish Poudel)

Executive Summary

Corona Virus Disease (COVID-19) has affected all of us in various ways. It has affected the health sector, business sector, education and many more fields. People have been confined inside their houses in isolation due to which their daily life has been affected. Corona Virus Disease is transmitted by air through droplets. It is recommended by the World Health Organization (WHO) to wear masks for preventing the rapid spread of Corona Virus. In order to run their daily lives, people need to go out to purchase basic needs and for their work. And a little bit of negligence in this scenario like no masks could spread the virus in an unprecedented manner. This negligence has been considered to formulate our project idea. Video surveillance for detecting masks in public places has been utmost necessary. In this project, we build a machine learning model using Mobile Netv2 and CNN algorithms. The model is able to detect if a person is wearing a mask or not. The project uses both image processing and video processing. The project hence can be implemented in public places for detecting unmasked faces and requesting them to wear masks. This will more or less help in prevention of Corona Virus.

CONTENTS **Page No.**

List of Figures

List of tables

Abbreviations

1.	WHO	-	World Health Organisation
2.	CNN	-	Convolutional Neural Network
3.	HTML	-	Hyper Text Markup Language
4.	CSS	-	Cascading stylesheet

1. INTRODUCTION

1.1 Objective

The main objective of the mask detection system is to make sure to detect people whether they are following covid norms properly or not by wearing a mask properly. Our project will work primarily on video surveillance to detect masked and unmasked faces. The project can be implemented easily in public places to ensure that there is no violation and everyone is wearing a mask.

In this project, we have used Mobile Netv2 and Convolutional Neural Network(CNN) algorithms to build a model. We have also used various libraries like OpenCV, NumPy, etc…

1.2 Motivation

Since the Corona Virus outbreak took place, it has affected most of the countries. To prevent the spread of coronavirus, many guidelines have been made by the World Health Organization(WHO). Also, various guidelines have been made by the government. Some of the common guidelines are to wear masks and maintain a social distance. Although it has been more than a year since the outbreak, still coronavirus cases are increasing. The curve of coronavirus cases was steep and high at the time of the outbreak. After a few months of outbreak, the curve has flattened. But still some countries are heavily affected and a second wave of virus is experienced. Various mutants of coronavirus are out there infecting more people per day. Although the vaccination process has started, it might take months for everyone to get vaccinated. So, strictly following the guidelines made by the government and WHO is an utmost necessity.

Although the guideless of wearing masks have been made by WHO and government, many people do not wear masks and it is becoming a sole cause for rapid spread of the coronavirus pandemic. Wearing a mask is a must and it should be monitored carefully for preventing the disease. So, a mask detection/ mask surveillance system is a need in this situation.

1.3 Background

Corona Virus has affected us in almost all sectors. Although the vaccination started, we have suffered a second wave of CoronaVirus and our condition worsened. Coronavirus affected us very badly in the first few months of the pandemic. The cases got reduced for a few months and then again, we got affected and we are still affected by the second wave of the pandemic. There are few points to consider and one of them is finding out where we went wrong.

As the spread of disease is through air, use of masks is the major preventive way that we can take. While we are suffering from the second wave, we can say that the cases increased rapidly because people stopped using masks or stopped caring as much as we should have. All this time, what we lacked was a mask detection system that could automatically detect faces with and without masks in mass gatherings and public places.

Although various object detection models have been developed, the detection of masks is a new model that needs to be developed. As the data for the training model is very limited, the accuracy of the model is being affected. So, the mask detection models out there have not been performing in good accuracy as we expect it to work. So, very few existing models and the problem of not enough accuracy is a problem that exists.

1.4 Literature Survey

S. no	Title	Authors	Year of Publication	Dataset	Methodology considered	Any Pros/Cons Mentioned	Future Work
1	Study of Masked Face Detection Approach in Video Analytics	Gayatri Deore, Ramakrishna Bodhula, Dr. Vishwas Udpikar, Prof. Vidya	2016	Not mentioned	Analog devices Inc.'s Cross core embedded studio and HOGSVM was used for detecting person and distance from camera. Face detection and face parts like eyes, nose and mouth is implemented by Viola Jones's algorithm. Viola Jones face detection procedure classifies images based on the value of simple features. There are three features, namely two rectangle, three rectangle and four rectangle. The value of a two-rectangle feature is computed by calculating the difference between the sum of the pixels within two rectangular regions.	Pros: Viola-Jones algorithm is robust with very high true detection rate and very low false positive rate, it is real time and at least 2 frames must be processed per second. Cons: Eye detection is reliable but prone to false eye detection	This proposed work may not have been able to detect the person when they are wearing a mask so to improve this accuracy of eye detection can be increased to help recognizing the person through his eye and eye line.
2	Deep	Md.	2020	Picasso,	To configure	Pros: Great	Fast

	Learning Based Assistive System to Classify COVID-19 Face Mask for Human Safety with YOLOv3	Rafiuzzaman Bhuiyan, Sharun Akter Khushbu, Md. Sanzidul Islam		People Art, KITTI MOD	YOLOv3 object names created to contain the name of the classes which model needs to detect, an input image is passed through the YOLOv3 model, the object detector finds the coordinates that are present in an image. For producing model output the neighbouring cells with high confidence rate of the features were added in the model output. 80 % of the data was used for training and rest is for validation.	detection speed and accuracy Cons: needs large number of backend network parameters and requires high hardware performance	RCNN object detection architecture can be used with YOLOv3 or the new version of YOLOv4 to increase the performance of the face detection system in real time video surveillance
3	Face mask detection using MobileNet and Global Pooling Block	Isunuri B Venkateswarlu, Jagadeesh Kakarla, Shree Prakash	2021	Simulated masked face dataset	Transfer learning has been used by using a pre-trained model MobileNet to use existing solutions to solve new problems. Global Pooling block transforms a multi-dimensional map into a 1D vector having 64 characteristics. Finally a softmax layer with 2 neurons takes the 1D vector and perform binary classification	Pros: Global pooling block has no parameter to optimize and hence overfitting is avoided in this layer	Face mask detection over multi-face images

4	Color quotient based mask detection	Ioan Buciu	2021	Masked Face Detection Dataset (MFDD), Realworld Masked Face Recognition Dataset (RMFRD) and Simulated Masked Face Recognition Dataset (SMFRD).	To detect faces a deep neural network (DNNN) learning network based on a single shot detection algorithm is used. Once the face is detected, the face region rectangle is split in 2 parts upper & lower. Now color quotient features are generated in a 2D vector by finding the lower part ratio and upper part ratio. To make the test statistically significant, they have used cross validation partition of the data. This was performed by repeating the creation of the training and test set for 100 times.	Pros: High detection accuracy, works for various resolution and also for different facial poses	To create a market ready product by merging alarm clock and notification system
5	Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV	Arjya Das, Mohammad Wasif Ansari, Rohini Basak	2021	Dataset 1: has 1376 images in which 690 images with people wearing face masks and the rest 686	The proposed method consists Of a cascade classifier and a pre-trained CNN. For image in the dataset Visualize the image in two categories: mask and no mask. Convert the RGB image to Grey-scale image and resize	Pros: The system can efficiently detect partially occluded faces whether they are with masks or hair or hands. Cons: Indistinct moving faces in the video	It can be extended to check whether a person is wearing a mask properly or not as the nose and

				images with people who do not wear face mask, Dataset 2 from Kaggle consists of 853 images and its countenances are clarified either with a mask or without a mask	this image into 100 X 100. Normalize the image and convert it into 4D array. To build the CNN model a convolution layer of 200 filters have been added and a second layer of 200 filters. A flatten layer to the network classifier has been added. In the end a final dense layer with 2 outputs for 2 categories has been inserted and the model is trained.	stream are difficult to detect.	mouth should be properly covered.
6	The Face Mask Detection For Preventing the Spread of COVID-19 at Politeknik Negeri Batam	Susanto Susanto; Febri Alwan Putra; Riska Analia; Ika Karlina Laila Nur Suciningtyas	2020	Not mentioned	Yolo V4 is implemented using two stage detectors. The first-stage detector consists of: Input- Resolution of input image is 1920*1080. Backbone- Darknet53 chosen as detector method, contains 29 Convolutional layers by 3*3 and each layer sent to the neck detector. Neck- PANet applied as the neck detector method. Dense prediction-	Pros: The algorithm is able to detect and distinguish a non-wearing and a wearing-mask precisely with any condition of the surrounding environment.	Adding Thermal Detection feature.

					YOLO v3 model used in this stage to generate the prediction. Second-stage detector: It has a sparse prediction which applies the faster R-CNN. Input to this layer is 3*3 layers got from the neck and the input prediction from the dense prediction.		
7	Face Mask Detection Using MobileNet V2 in The Era of COVID-19 Pandemic	Samuel Ady Sanjaya; Suryo Adi Rakhmawan	2020	1. Kaggle dataset and the Real-World Masked Face dataset (RMFD) 2. Dataset taken from CCTV, shop, and traffic lamp camera. (25 Cities of Indonesia)	Collect Data(Masked and Unmasked Data), Preprocessing (Resizing, converting to array, preprocessing using mobilenet v2, etc.), Split Data(75/25), Build Model, Testing and Implementation. MobileNet V2, a convolutional neural network architecture is used.	Pros: The model has 96.85 accuracy. The data collected from different cities can be used for statistical analysis of people wearing masks and appropriate action could be taken for preventing the spread of Covid.	Adding features like detecting social distancing violations in the same model.
8	Real-time Face Mask and Social Distancing Violation Detection System using YOLO	Krisha Bhambani; Tanmay Jain; Dr. Kavita A. Sultanpure	2020	WIDER-FACE(a face detection benchmark) and MAFA(Masked Faces) Datasets	Yolo v4 is implemented in the paper. The model has backbone, neck and dense/sparse prediction. CSPDarknet53 is used in the backbone. Spatial pyramid pooling(SPP) was	Pros: The paper considers implementation of mask detection and social distancing violation in the same model. Cons: The accuracy	Working more on improving accuracy of the model.

					used as the neck which contains blocks for increasing the receptive field and to aggregate parameters from different levels of the backbone. For detecting social distancing violations, focal length and sensor dimensions input was taken and concepts of optics were applied for calculating actual distance. e.g. sensor dimension / focal length = field dimension / distance to field	achieved is 94.75 % which is less than MobileNet v2 model.	
9	Face Mask Recognition using Machine Learning	Tejal Nerpagar; Sakshi Junnare; Janhavi Raut; Aarti Shah; Prof. P.C. Patil	2020	Not mentioned	The paper discusses on deep learning frameworks. TensorFlow, Keras, PyTorch, OpenCV, Caffe(a deep learning framework), MxNet(a library for deep learning), Microsoft Cognitive Toolkit are used in the paper for implementing mask detection in public places.	Pros: This model can be used for detecting masks and allowing entries in workplaces.	Adding Thermal Detection feature with the model.
10	Face Mask Detection System using Deep	Pinki; Prof. Sachin Garg	2020	Not mentioned	The paper proposes use of MobileNet v2 and Tensorflow classifiers for	Pros: The model can successfully identify the	The project can be integrate

	Learning				building the model. Also, for detecting COVID 19 face mask detector, OpenCV is used. The paper discusses on detecting masks in images as well as video.	person on image/video stream wearing face mask or not.	d with embedded systems for application in airports, railway stations, offices, schools, and public places to ensure that public safety guidelines are followed.

2. PROJECT DESCRIPTION AND GOALS

In this project, we aim to create a mask detection model that can detect faces with and without masks. Various technologies are considered while making this project. The project uses the Deep Learning model (MobileNet v2). Mobilenet is a convolutional neural network architecture. What distinguishes mobilenet models from other deep learning models is that mobilenet is lightweight as compared to other models. So, while creating this project, we not only want to just create a mask detection system but we want it to get implemented even in devices like mobile phones so that the project doesn't have much of a prerequisite in devices where the implementation is to be done.

With this project, we also want to improve the accuracy of our models so that the detection of masked and unmasked faces can be done properly. For that we are using the concepts of Transfer learning. Transfer learning takes what a model learns while solving one problem and applies it to a new application. As we have a very limited training dataset, we can use the concept of transfer learning and use the pre-existing mobilenet model for feature extraction. We then will add a few layers and start training the model. This will help to improve the model we'll develop in the project. Also, the concept of image data augmentation is used to artificially expand the size of a training dataset by creating modified versions of images in the dataset.

Some of the project goals are listed below:

- Create a mask surveillance system that detects faces with and without mask
- Use various concepts of Image Processing and Video Processing
- Create a light weight model so that it was wider scope of applicability
- Use the concepts of Transfer Learning
- Use image augmentation techniques for training the model as the dataset available has just few images

3. TECHNICAL SPECIFICATION

Hardware Specifications:

Processor – Intel(R) Core (TM) i7-8500U CPU @ 1.80GHz ($ CPU), ~ 1.99GHz

Graphic Card – AMD Radeon Graphics Processor

RAM – 8.0 GB

ROM – 1TB HDD

Software Specifications:

Operating System Used – Windows 10 64-bit

Technologies Used – Python, HTML, CSS, Flask

4. DESIGN APPROACH AND DETAILS

4.1 Design Approach

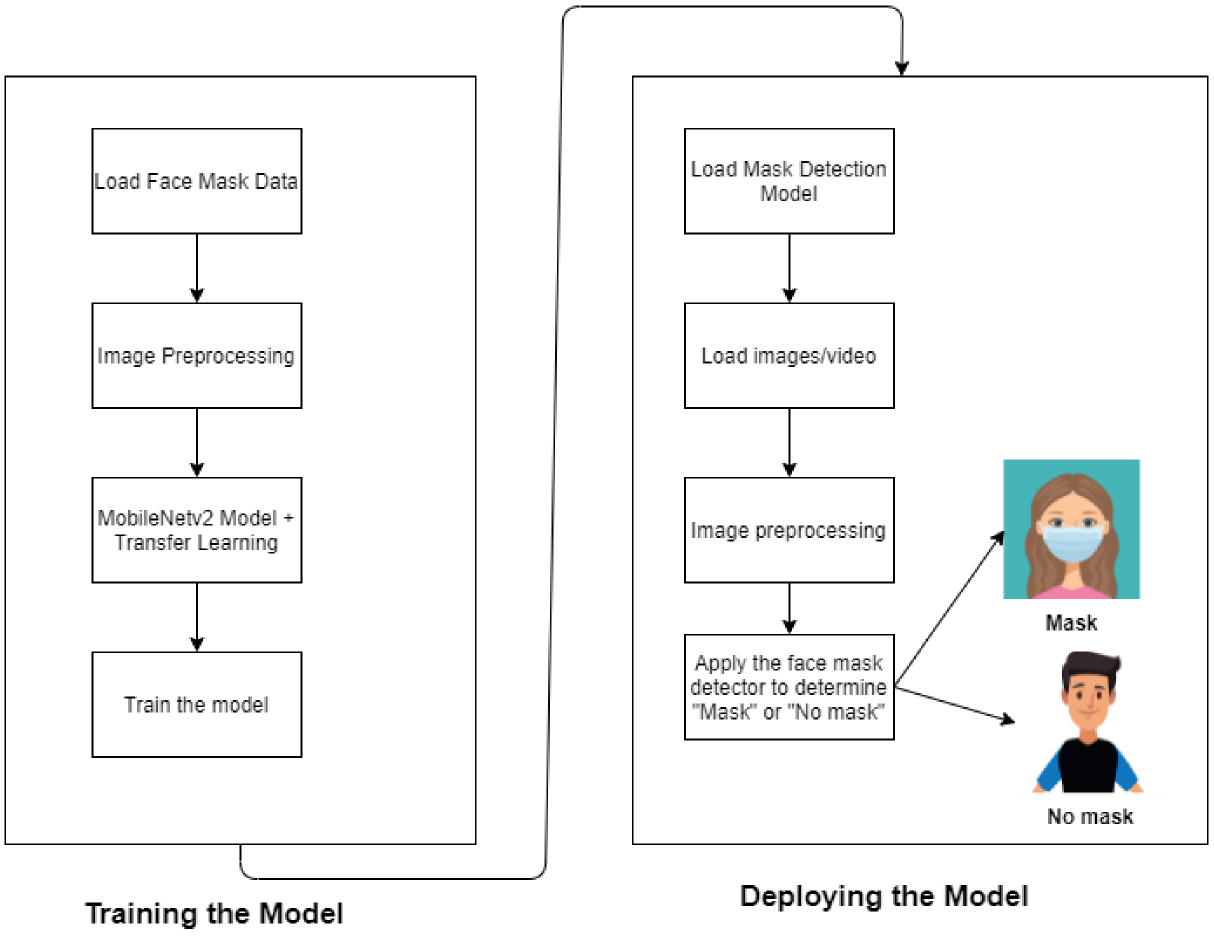


Figure-1: Workflow

The first step is loading the face mask data from the dataset. The dataset we are using has approximately 1800 images in both “with_mask” and “without_mask” folders. So, the first step will be to read these images. After reading the paths of images, we need to preprocess the image and prepare the image ready for feeding into the mobilenetv2 model. The preprocessing step prepares the image so that it is in the same format as the images that were used to train the network. Then the next step is to convert the images into numpy arrays so that we can carry out various operations in the image.

Also, as our dataset contains two folders (with_mask and without_mask), we need to use labelBinarizer and convert to categorical labels for changing the labels from string to some array. After applying label binarizer, we will get an array of the labels where [0.,1.] denotes without mask and [1.,0.] denotes with mask.

The next step will be splitting the data into training and testing. In this project, we used 20% data for testing and the rest 80% data for training the model. As we have just 1800 images in our dataset and allotting 20% to testing means that we’re very low in numbers for training the model. If we train the model with just a few images, we won't get a model with good accuracy. So, we in this project have used the concept of Image Data Generator i.e. image augmentation technique is used. Image Data Augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset. We have given a few parameters to consider in image augmentation. Some of them are rotation_range (rotates an image to a certain degree), zoom_range(zooms the image in the given scale), width_shift_range(shifts the width of an image), height_shift_range(Shifts the height of an image), etc…

The next step will be creating the model. We're using the MobileNetv2 model in this project. So, we need to import the model and as we are using the concept of transfer learning, the top layer of the mobilenet model is not considered. Transfer learning takes what a model learns while solving one problem and applies it to a new application. So, in this project we're using the mobilenetv2 model as a feature extractor and we're adding a few more layers into it to create a final model. We added AveragePooling2D, Flatten, Dense, Dropout and final Dense layers to the mobilenetv2 model. All the previous layers of the mobilenetv2 model are not trained and the final added layers are trained to create a mask detector model. After the model is created, the next step is to test the model. We tested the model first using the test data (20%) that we allotted.

Next step is to deploy the model in the real world scenario. For detecting faces, we used caffemodel and its prototxtPath and weightsPath. So, the next step is to read the image and load the model. The image once rest is preprocessed in the same way as mentioned above. Then, the caffemodel detects faces and our model detects mask/no mask. Whenever any face is detected, a rectangle is drawn and mask/no mask is labelled as per the prediction made by the Mask Detection Model that we created. Also, for video processing video stream is used. Using video stream, the video is captured frame by frame and the frame is processed and predicted in the same way as we predicted for images. User interface was created using Front end technologies HTML and CSS(Styling). The scripts were then integrated with the front end page using Flask.

4.2 Code

MASK TRAINING

```
import numpy as np#for array funcs

import os#to access dataset

import matplotlib.pyplot as plt#graphs

from imutils import paths#access dataset
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 ImageDataGenerator
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
dataset=r'C:\Python37\Projects\face-mask-detector\dataset'
imagePaths=list(paths.list_images(dataset))

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

    data.append(image)


data=np.array(data,dtype='float32')

labels=np.array(labels)

lb=LabelBinarizer()

labels=lb.fit_transform(labels)#read labels and analyse pattern

labels=to_categorical(labels)#label them in 0 & 1


train_X,test_X,train_Y,test_Y=train_test_split(data,labels,test_size=0.20,stratify=labe
ls,random_state=10)

#80%,20%

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,vertical_flip=True,
fill_mode='nearest')

#imp:- generate more number of images using current images,

#rotate first image by 20% and create second image, zoom

#that image and create one more image, shift the range,height_shift,

#shear the image,filp the image,fill nearly, one image will generate 6 more

#images now and hence dataset increases

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

#here weights starts downloading imagenet but remove last layer because we have

#only 2 class and this model now will have the knowledge of feature extraction

print(baseModel.summary())


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)
#To avoid overfitting
headModel=Dense(2,activation='softmax')(headModel)
model=Model(inputs=baseModel.input,outputs=headModel)

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

print(model.summary())
learning_rate=0.001
Epochs=20
BS=12
opt=Adam(lr=learning_rate,decay=learning_rate/Epochs)
model.compile(loss='binary_crossentropy',optimizer=opt,metrics=['accuracy'])
H=model.fit(
    aug.flow(train_X,train_Y,batch_size=BS),
    steps_per_epoch=len(train_X)//BS,
    validation_data=(test_X,test_Y),
    validation_steps=len(test_X)//BS,
    epochs=Epochs
)
model.save(r'C:\Python37\Projects\face-mask-detector\mobilenet_v2.model')
predict=model.predict(test_X,batch_size=BS)
predict=np.argmax(predict,axis=1)
print(classification_report(test_Y.argmax(axis=1),predict,target_names=lb.classes_))
```

```
# plot the training loss and accuracy

N = EPOCHS
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), H.history["acc"], label="train_acc")
plt.plot(np.arange(0, N), H.history["val_acc"], label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.savefig(r'C:\Python37\Projects\face-mask-detector\plot_v2.png')
```

MASK TESTING USING IMAGES

```
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
# In[ ]:
prototxtPath=os.path.sep.join([r'C:\Users\siddh\Mask-Detection-and-Recognition-using-Deep-Learning-Keras','deploy.prototxt'])
weightsPath=os.path.sep.join([r'C:\Users\siddh\Mask-Detection-and-Recognition-using-Deep-Learning-Keras','res10_300x300_ssd_iter_140000.caffemodel'])
```

```
#download models to detect the face from the image

# In[ ]:

prototxtPath

# In[ ]:

weightsPath

# In[ ]:

net=cv2.dnn.readNet(prototxtPath,weightsPath)

# In[ ]:

model=load_model(r'C:\Users\siddh\Mask-Detection-and-Recognition-using-Deep-
Learning-Keras\mobilenet_v2.model')

# In[ ]:

image=cv2.imread(r'C:\Users\siddh\Mask-Detection-and-Recognition-using-Deep-
Learning-Keras\example_02.png')

# In[ ]:

image

# In[ ]:

(h,w)=image.shape[:2]

# In[ ]:

(h,w)

# In[ ]:

blob=cv2.dnn.blobFromImage(image,1.0,(300,300),(104.0,177.0,123.0))

#for compressing the image

# In[ ]:

blob

# In[ ]:

blob.shape

# In[ ]:

net.setInput(blob)
```

```
detections=net.forward()

#to read image in dnn and store the face extraction values

# In[ ]:

detections

# In[ ]:

#loop over the detections

for i in range(0,detections.shape[2]):

    confidence=detections[0,0,i,2]




    if confidence>0.5:

        #we need the X,Y coordinates

        box=detections[0,0,i,3:7]*np.array([w,h,w,h])

        (startX,startY,endX,endY)=box.astype('int')


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

        (startX,startY)=(max(0,startX),max(0,startY))

        (endX,endY)=(min(w-1,endX), min(h-1,endY))




        #extract the face ROI, convert it from BGR to RGB channel, resize it to 224,224
and preprocess it

        face=image[startY:endY, startX:endX]

        face=cv2.cvtColor(face,cv2.COLOR_BGR2RGB)

        face=cv2.resize(face,(224,224))

        face=img_to_array(face)

        face=preprocess_input(face)
        face=np.expand_dims(face,axis=0)


        (mask,withoutMask)=model.predict(face)[0]


        #determine the class label and color we will use to draw the bounding box and
text

        label='Mask' if mask>withoutMask else 'No Mask'

        color=(0,255,0) if label=='Mask' else (0,0,255)


        #include the probability in the label

        label="{}: {:.2f}%".format(label,max(mask,withoutMask)*100)


        #display the label and bounding boxes

        cv2.putText(image,label,(startX,startY-
10),cv2.FONT_HERSHEY_SIMPLEX,0.45,color,2)

        cv2.rectangle(image,(startX,startY),(endX,endY),color,2)



cv2.imshow("OutPut",image)

cv2.waitKey(0)

cv2.destroyAllWindows()
```

<u>MASK TESTING USING VIDEO</u>

```
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
from imutils.video import VideoStream
import imutils
# In[2]:
def detect_and_predict_mask(frame,faceNet,maskNet):
    #grab the dimensions of the frame and then construct a blob
    (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()

    #initialize our list of faces, their corresponding locations and list of predictions

    faces=[]
    locs=[]
    preds=[]
    for i in range(0,detections.shape[2]):
        confidence=detections[0,0,i,2]

        if confidence>0.5:
        #we need the X,Y coordinates
            box=detections[0,0,i,3:7]*np.array([w,h,w,h])
            (startX,startY,endX,endY)=box.astype('int')
            #ensure the bounding boxes fall within the dimensions of the frame
            (startX,startY)=(max(0,startX),max(0,startY))
            (endX,endY)=(min(w-1,endX), min(h-1,endY))

            #extract the face ROI, convert it from BGR to RGB channel, resize it to
224,224 and preprocess it

            face=frame[startY:endY, startX:endX]

            face=cv2.cvtColor(face,cv2.COLOR_BGR2RGB)

            face=cv2.resize(face,(224,224))

            face=img_to_array(face)

            face=preprocess_input(face)

            faces.append(face)

            locs.append((startX,startY,endX,endY)

        #only make a predictions if atleast one face was detected

        if len(faces)>0:

            faces=np.array(faces,dtype='float32')

            preds=maskNet.predict(faces,batch_size=12)


        return (locs,preds)

# In[3]:

prototxtPath=os.path.sep.join([r'C:\Users\siddh\Mask-Detection-and-Recognition-
using-Deep-Learning-Keras','deploy.prototxt'])

weightsPath=os.path.sep.join([r'C:\Users\siddh\Mask-Detection-and-Recognition-
using-Deep-Learning-Keras','res10_300x300_ssd_iter_140000.caffemodel'])

# In[4]:

faceNet=cv2.dnn.readNet(prototxtPath,weightsPath)

# In[5]:

maskNet=load_model(r'C:\Users\siddh\Mask-Detection-and-Recognition-using-
Deep-Learning-Keras\mobilenet_v2.model')

# In[ ]:

vs=VideoStream(src=0).start()

while True:
     #grab the frame from the threaded video stream and resize it
     #to have a maximum width of 400 pixels
     frame=vs.read()
     frame=imutils.resize(frame,width=400)
     #detect faces in the frame and preict if they are waring masks or not
     (locs,preds)=detect_and_predict_mask(frame,faceNet,maskNet)
       #loop over the detected face locations and their corrosponding loaction
     for (box,pred) in zip(locs,preds):
        (startX,startY,endX,endY)=box
        (mask,withoutMask)=pred
        #determine the class label and color we will use to draw the bounding box and
text
        label='Mask' if mask>withoutMask else 'No Mask'
        color=(0,255,0) if label=='Mask' else (0,0,255
        #display the label and bounding boxes     cv2.putText(frame,label,(startX,startY-
10),cv2.FONT_HERSHEY_SIMPLEX,0.45,color,2
        cv2.rectangle(frame,(startX,startY),(endX,endY),color,2)
     #show the output frame
     cv2.imshow("Frame",frame)
     key=cv2.waitKey(1) & 0xFF


     if key==ord('q'):
        break
cv2.destroyAllWindows()
vs.stop()
```

5. **SCHEDULE, TASKS AND MILESTONES**

5.1 DATASET COLLECTION & TRAINING

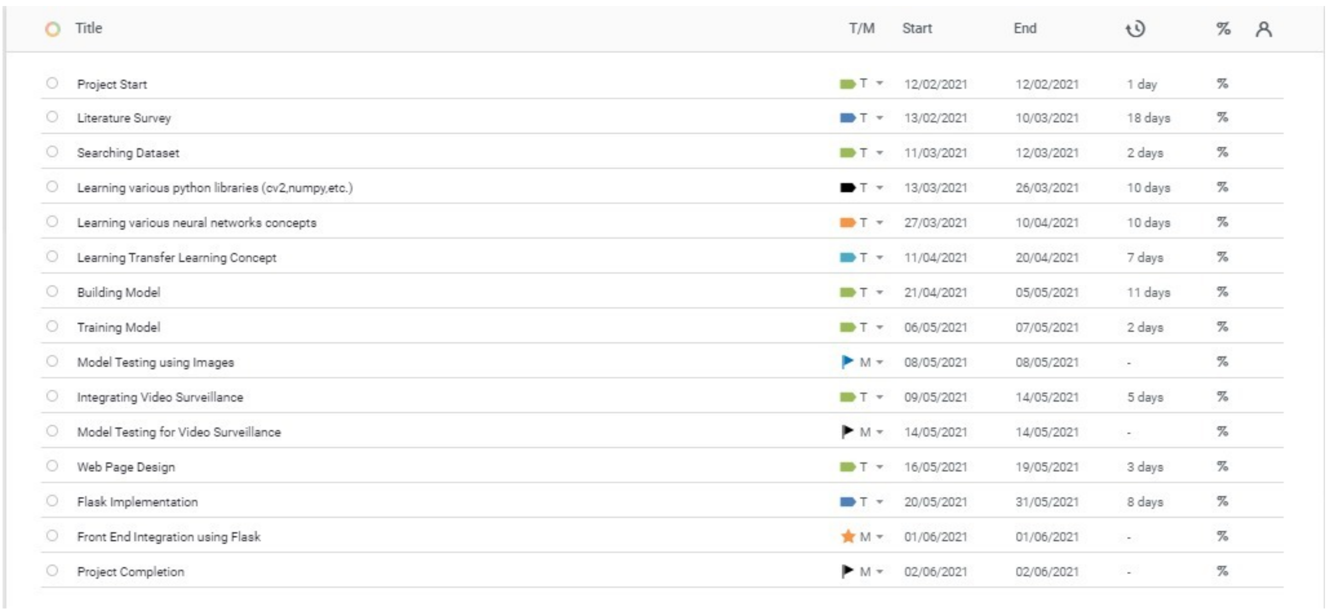
In the first task a dataset of images of two types are collected first, containing people wearing masks of different colour and shape and second contains images of people who are not wearing masks. After the dataset collection MobileNetV2 model is combined with imagenet for transfer learning by filtering only a few required attributes. When the baseModel was ready our dataset training began with 20 epochs. After the model is trained it was stored for later use and accuracy of the model was calculated.

5.2 USING TRAINED MODEL

After the model was trained 2 different functionalities of the project were created. First was to input an image to the model for mask detection and second was to integrate a video stream as input to take frames at a time and detect the mask on them. For proper results before using the trained model a caffe model was used to detect the rectangular coordinates of the face so that the trained model can be applied within a specific region of the image.

5.3 INTEGRATING WITH FRONTEND

After the script started showing correct results for the mask detection our trained model had to be connected with a user interface so that the user does not have to run the script manually. To achieve that our model is connected with the frontend using flask which automatically runs the python script to call the model and start the live video output stream to check whether a person is wearing a mask or not. This ensures that the user does need to have the prior knowledge of transfer learning and a fully functional ready to use project is created.



Title	T/M	Start	End		%	
Project Start	🟩 T	12/02/2021	12/02/2021	1 day	%	
Literature Survey	🟦 T	13/02/2021	10/03/2021	18 days	%	
Searching Dataset	🟩 T	11/03/2021	12/03/2021	2 days	%	
Learning various python libraries (cv2,numpy,etc.)	⬛ T	13/03/2021	26/03/2021	10 days	%	
Learning various neural networks concepts	🟧 T	27/03/2021	10/04/2021	10 days	%	
Learning Transfer Learning Concept	🟦 T	11/04/2021	20/04/2021	7 days	%	
Building Model	🟩 T	21/04/2021	05/05/2021	11 days	%	
Training Model	🟩 T	06/05/2021	07/05/2021	2 days	%	
Model Testing using Images	⚑ M	08/05/2021	08/05/2021	-	%	
Integrating Video Surveillance	🟩 T	09/05/2021	14/05/2021	5 days	%	
Model Testing for Video Surveillance	⚑ M	14/05/2021	14/05/2021	-	%	
Web Page Design	🟩 T	16/05/2021	19/05/2021	3 days	%	
Flask Implementation	🟦 T	20/05/2021	31/05/2021	8 days	%	
Front End Integration using Flask	★ M	01/06/2021	01/06/2021	-	%	
Project Completion	⚑ M	02/06/2021	02/06/2021	-	%	

Table-1: Tasks and milestones

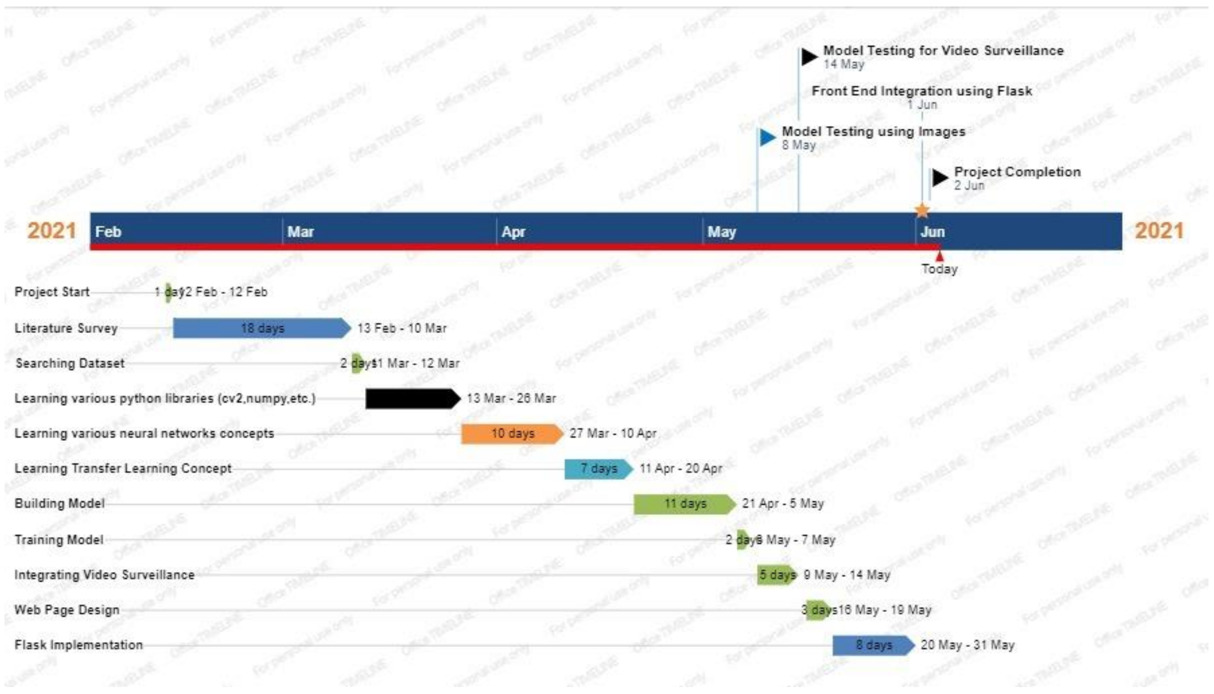


Figure-2: Gantt chart

6. PROJECT DEMONSTRATION

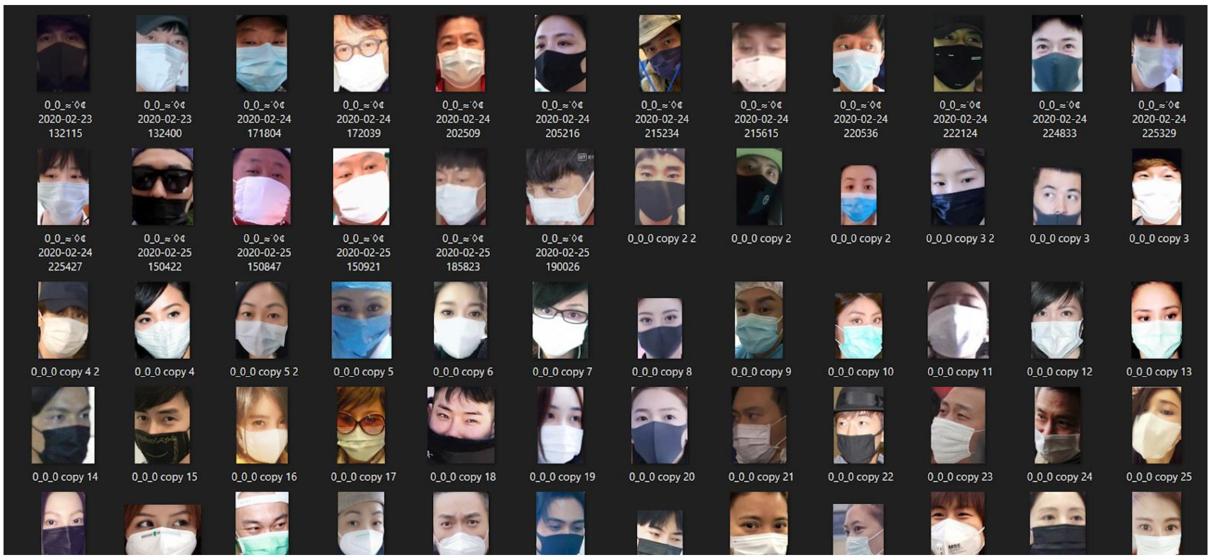


Figure-3: Dataset with mask

- Dataset-with mask

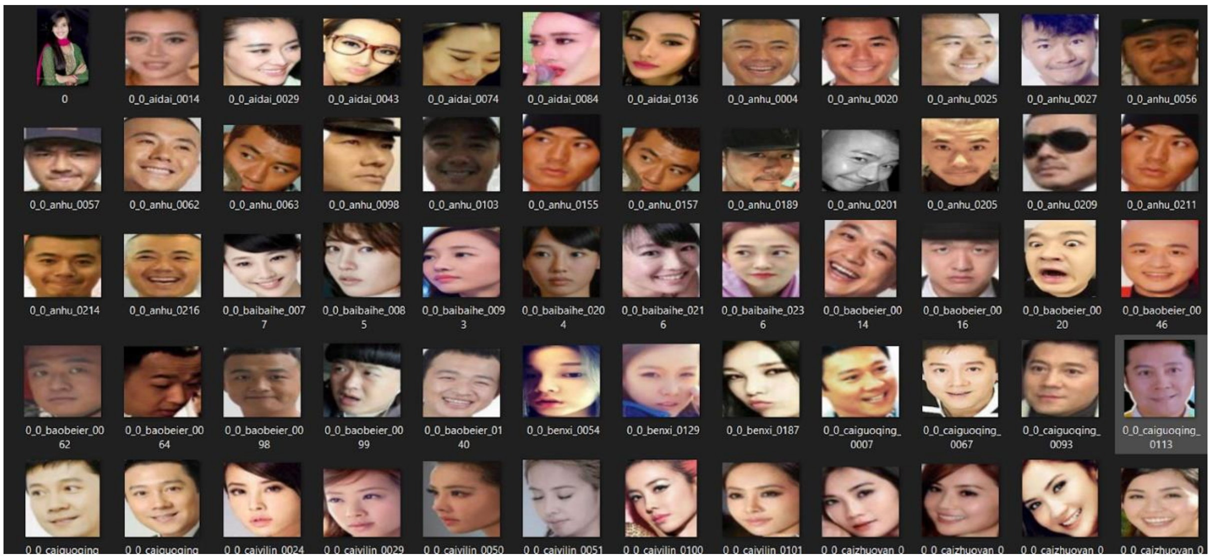
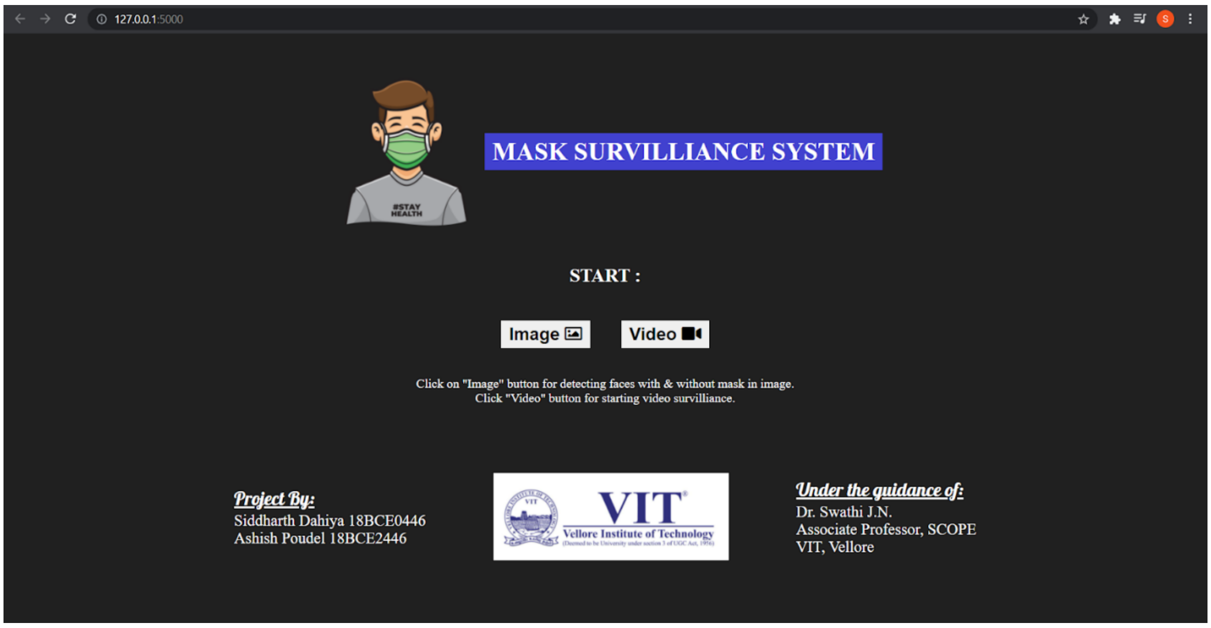


Figure-4: Dataset without mask

- Dataset-without mask


127.0.0.1:5000

MASK SURVILLIANCE SYSTEM

START :

Click on "Image" button for detecting faces with & without mask in image.
Click "Video" button for starting video survilliance.

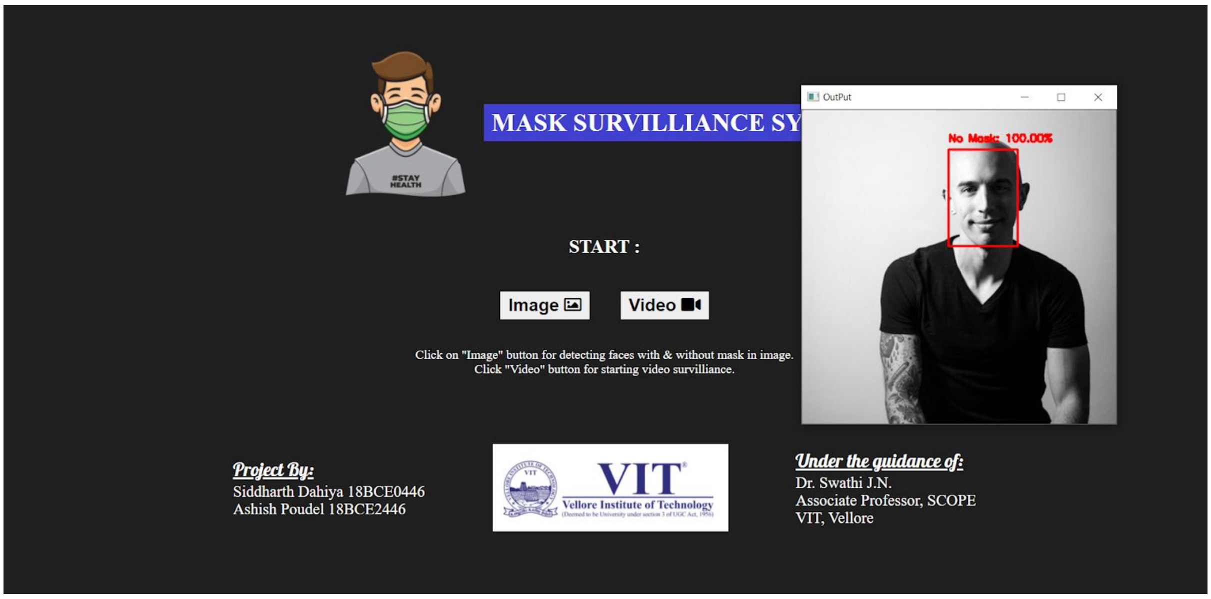
Project By:
Siddharth Dahiya 18BCE0446
Ashish Poudel 18BCE2446

VIT
Vellore Institute of Technology

Under the guidance of:
Dr. Swathi J.N.
Associate Professor, SCOPE
VIT, Vellore

Figure-5: Mask surveillance system

- Final deployed project


MASK SURVILLIANCE SY

START :

Click on "Image" button for detecting faces with & without mask in image.
Click "Video" button for starting video survilliance.

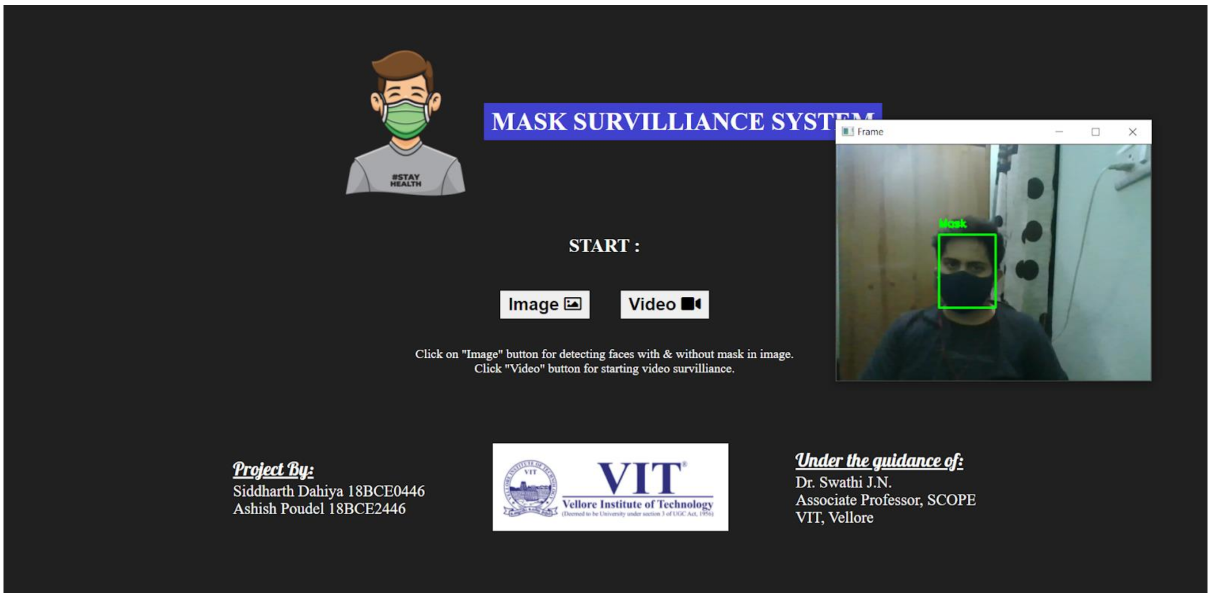
Project By:
Siddharth Dahiya 18BCE0446
Ashish Poudel 18BCE2446

VIT
Vellore Institute of Technology

Under the guidance of:
Dr. Swathi J.N.
Associate Professor, SCOPE
VIT, Vellore

Figure-6: Image script output

- After clicking the image button, the image script starts and detects the mask detection and shows whether there is a mask or not.



Click on "Image" button for detecting faces with & without mask in image.
Click "Video" button for starting video survilliance.

Project By:
Siddharth Dahiya 18BCE0446
Ashish Poudel 18BCE2446

Under the guidance of:
Dr. Swathi J.N.
Associate Professor, SCOPE
VIT, Vellore

Figure-7: Video script output

- After clicking the video button, the video script starts by taking frames and detects the mask detection and shows whether there is a mask or not.

7. **RESULT ANALYSIS**

The model is trained, approved and tried upon two datasets, covers and non-covers. The technique achieves accuracy up to 95.14% this enhanced precision mitigates the expense of errors. Dataset2(without cover) is more flexible than dataset 1 as it has numerous contents in the casing and various kinds of covers having various tones also. Consequently, the model accomplishes an accuracy of 95.14% on complete dataset in any case further improvement can likewise be achievable. One of the primary explanations for accomplishing this precision lies in MaxPooling. It provides rudimentary translation invariance to the internal representation along with the reduction in the number of parameters the model has to learn. The upgraded filter qualities and pool size help to filter out the fundamental segment (face) of the picture to recognize the detection of mask effectively without causing over-fitting.

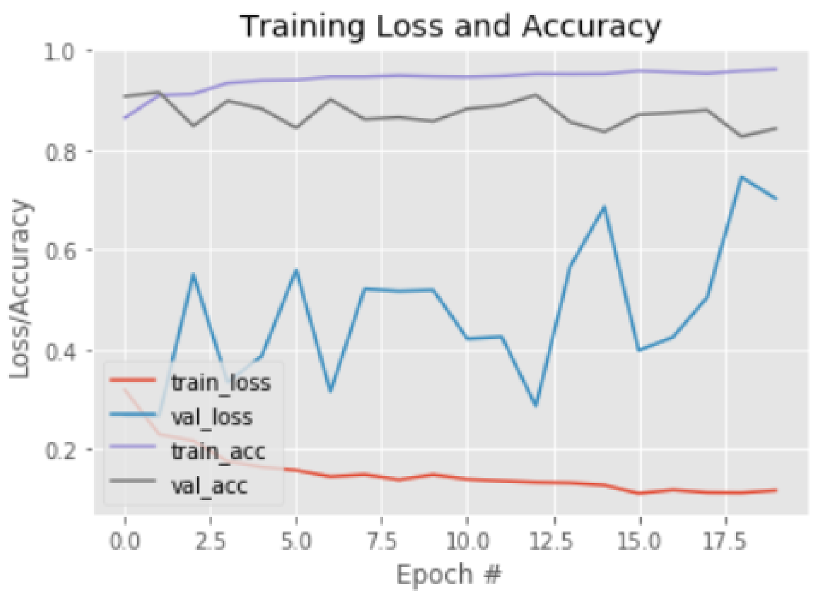


Figure-8: Epoch analysis

The main issues faced by the above method mainly consist of varying angles and lack of clarity of frame. Constantly moving faces in the video steam input makes it more difficult to detect masks. However, following the trajectories of several frames of the video helps to create a better decision.

Wearing a mask is obligatory nowadays in this covid pandemic. It is compulsory to wear a mask in public to avail any services. The deployed model will contribute immensely to the public health department, it can help them detect those people in a crowded place who are not wearing a mask and even if they are wearing it can also detect whether the mask worn by a person is properly aligned or not. In the froecoming future this can be further improved to detect whether the type of mask is surgical, N95 or not i.e. it is virus prone or not.

Google link for demo:-
https://drive.google.com/drive/folders/1ISIezEW_bMS4ck9mHDaHcmkaSHBr7Zx-?usp=sharing