SRM UNIVERSITY AP
Amaravati

School of Computer Science and Engineering

REAL TIME FACE-MASK DETECTION

A project submitted
in partial fulfillment of the requirements for the degree
of Bachelor of Technology (CSE)

By:
1)Veda Gayathri Ravi(AP18110010381)
2)Purna Praveen.B(AP18110010379)
3)Aparna Gurram(AP18110010662)
4)Asritha.P(AP18110010378)

UNDER THE GUIDANCE OF :
Dr. RAVI KANT KUMAR
(Assistant Professor)

ABSTRACT:

COVID-19 is one of the most contagious illnesses known to man. Coronavirus is a virus that originated in China and soon spread over the world. The coronavirus has infected more than 258,044,984 individuals. USA, India and Brazil are the most affected countries in the world. Approximately 52 lakhs people died due to various reasons which are caused by corona.We know that the second wave changed lives of many people.All governments are taking a strict care to protect their people.A fraction of Covid's insurance policies are hand cleaning, veiling (mask), and 6 feet social distance. We have seen the Corona Virus second wave which is very irresistible.So to prevent the virus,the Government needs to monitor the people whether they are wearing their masks or not to stop further spread of covid. Not only wearing the mask but the mask should be in the proper position.So,to monitor this we are using a strong technology that uses MobileNet .This paper demonstrates the face mask recognization using MobileNet and Python.

1. **INTRODUCTION**:

The coronavirus affected 223 countries, according to the World Health Organization (WHO).We are on the verge of seeing the third influx of Covid. As we have seen that the number of cases in this second wave of coronavirus increased twice the rate of the first wave. The Coronavirus epidemic is affecting millions of people worldwide. The main cause of the rising number of coronavirus cases is human behaviour. In public places, the government is attempting to impose the wearing of a mask and physical separation.

However, when the administration tries to restore offices, theatres, cafés, and businesses, many individuals are disregarding the orders given by the Government. Because people infected with the coronavirus do not display any symptoms for a long time, there is a greater risk of the virus spreading if the government's safeguards are not followed. The government and other public sectors should employ a face mask recognition model that can constantly scan people to see if they are wearing a face mask or not to avoid the spread of coronavirus. Checking someone whether they are wearing a mask or not manually is a tough job and time-consuming operation that requires a lot of labour.

Not only Corona virus,we have a serious problem in India that is Pollution.Recently,in Delhi schools and colleges are shut due to worsening levels of air pollution.The PM levels between zero to 50 is considered as good and between 51-100 is satisfactory levels of pollution according to air quality index(AQI).But, in Delhi figures are greater than 400 which is very severe to the people living in Delhi.So,in that case it is better to wear a mask to avoid pollution a little bit on our end. We don’t know how it will affect us so to avoid such situations wearing a mask helps to live a better life without any diseases.

This is why we should do this activity with the help of a face mask detection model. There are numerous face mask identification algorithms on the market, however many of them have serious flaws because they were constructed using outdated deep learning and AI calculations. In this project, we'll show a face mask identification model using MobileNet it says whether we are wearing a mask or not and with its probability.We are using MobileNet because it is faster in process than CNN and uses lesser parameters than other algorithms.

2. <u>LITERATURE SURVEY:</u>

Before reaching the model proposed here, some spot checks were performed on the project. Similar and target areas of interest. First, they worked second Color-based mask recognition. Don't do it It works in real time. The position of the face mask under consideration has not been calculated accurately. Can mitigate it The problem of the huge variety of pixel sizes [1]. Second one, they have worked upon VFR. It can detect Precise boundaries with high exactness. By improving the goal of organization contribution for the best tradeoff among speed and precision. A strong correlation between target sizes and positions for face mask scenes. Friendly to the base of the image Reduced execution and recognition failure. A constant recognition speed is not enough. RoI pooling Not exactly the same as cropping from the original image [2]. Thirdly, we worked on semantic segmentation. Operators like erosion and expansion I took over again. End-to-end trainable means to evenly combine mask detectors with mask detectors Communication network. Generalization of problems with finding object bounding boxes. Poor standardization of face mask Not supported. Changing the color representation is not enough for a consistently robust design identification.The presentation isn`t comparable to the proposed strategy [3].In the fourth, they worked on YOLO (you only look once). it takes A huge amount of face mask recording. Complex face masks cannot be recognized. Do not eliminate Redundancy test box. Consistency isn't improved in outer slope. Don't do your best to act Abnormally bent information from the face cover [4]. In fifth, they worked on a deep learning classifier. Detection time is heading in real time sequence. Face mask detection procedures can reduce complex backgrounds and text clutter It's not a face mask. It uses the latest architecture for object recognition. B. Highlight the pyramid network. Reference colours aren't converted to polar co-ordinates . The image will fail because the text alignment angle is so large. This is a limitation. Of the content detector. Small geometric changes throughout the frame can lead to significant imaging errors [5].

3. <u>REQUIREMENTS:</u>

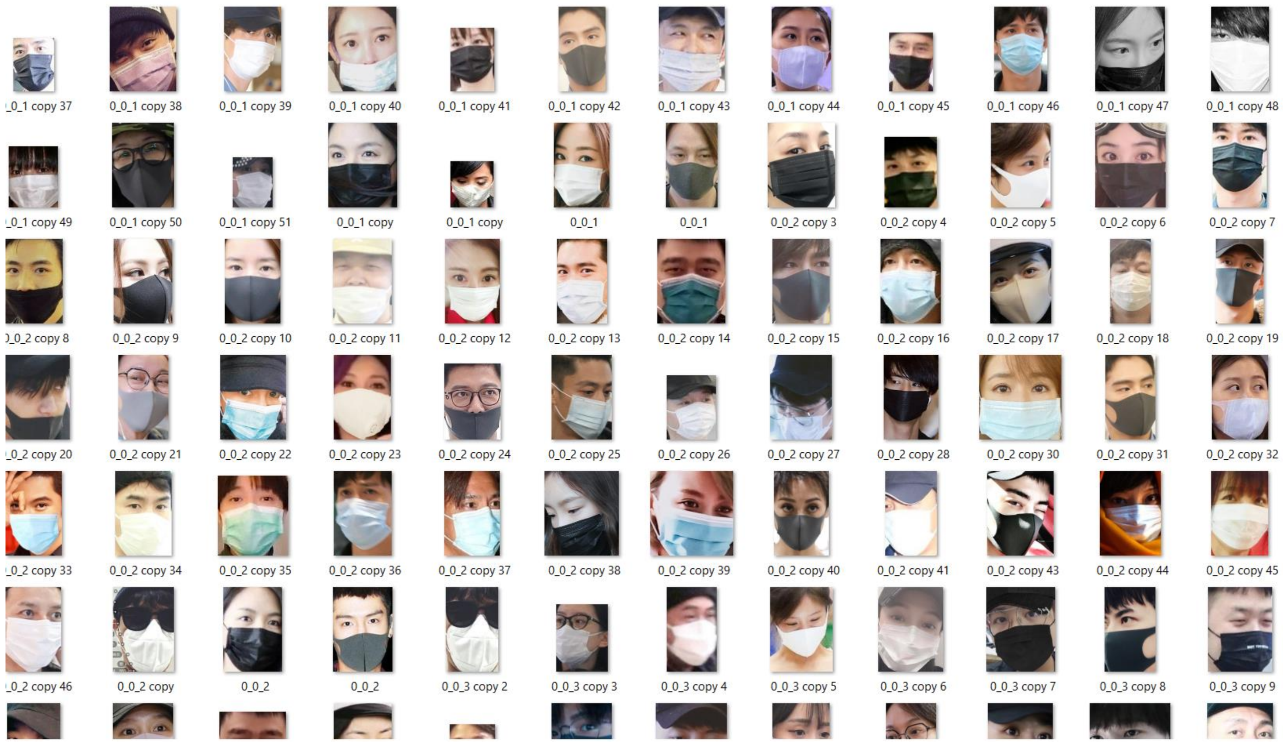
1.We took the dataset from keggal website(with mask:1915,without mask:1918)

2. Install tensorflow
3. Install keras
4. Install imutils
5. Install numpy
6. Install opencv-python
7. Install matplotlib
8. Install scipy
9. Jupyter Notebook(To execute the code)

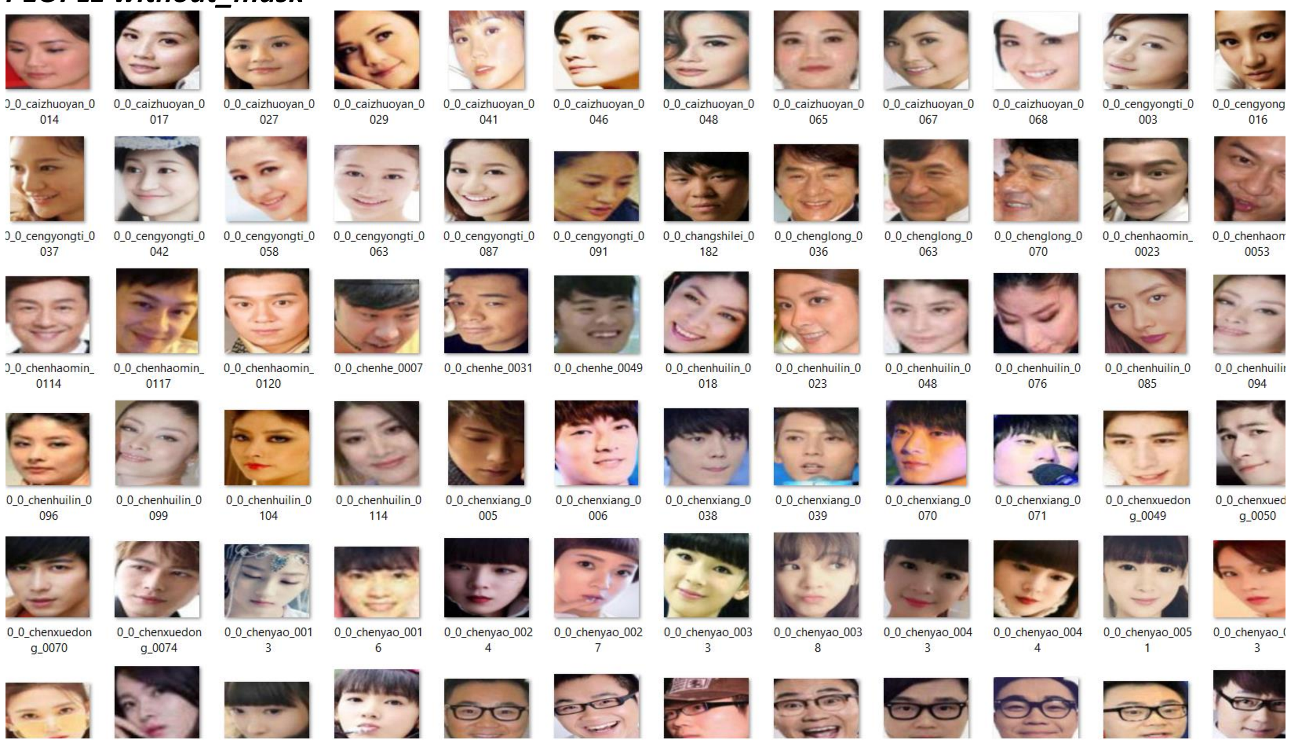
4. **<u>DATA SET</u>:** In this project, the database used to train the model is taken from the keggal website and it is a facial mask recognition dataset developed in keggal.In this dataset,we have with mask dataset and without mask dataset.In this with mask dataset,we will be having 1915 pictures of people having masks on them and in without mask dataset we are having 1918 pictures of people not having masks.So,these pictures are used to train and test the model.Using sklearn convenience method,our data is segmented into 20%~0.20 of testing and 80%~0.80 of training.

5. **<u>SAMPLE IMAGES:</u>**

PEOPLE with_mask

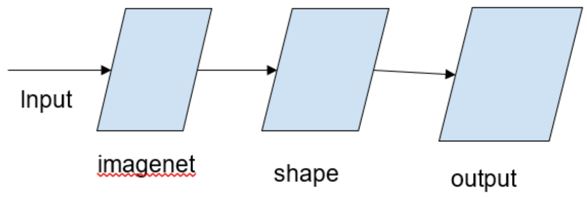


PEOPLE without_mask

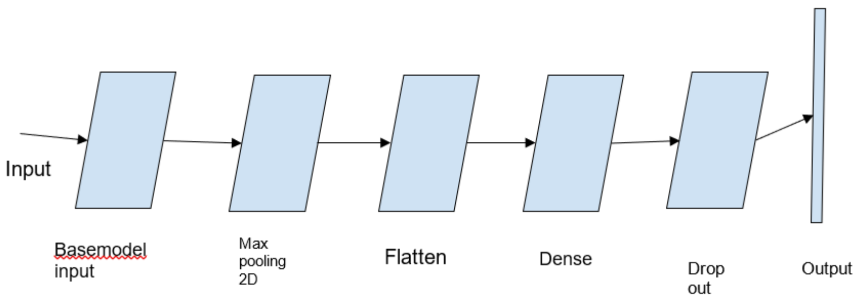


6. **PROPOSED MODEL ARCHITECTURE:**

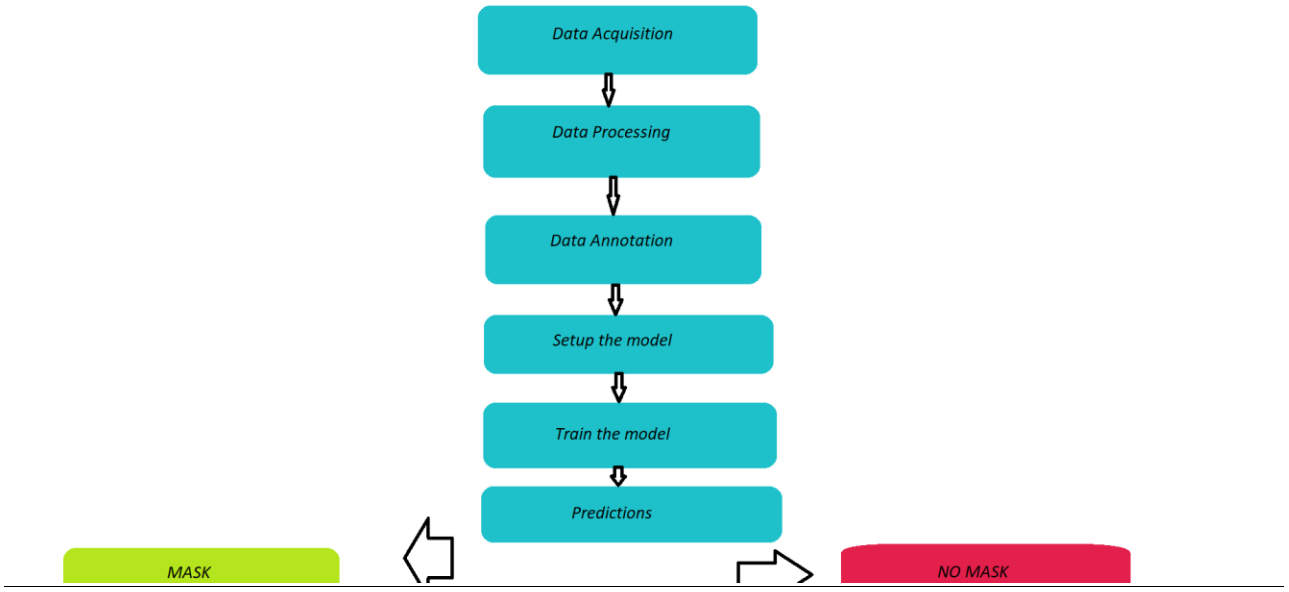
BASE MODEL



HEAD MODEL



7. WORKFLOW STRUCTURE:



8. METHODOLOGY:

We are using MobileNet algorithm as it uses deep convolutional layer as the basic main layer which controls the delay while considering the size of the model.

We will collect a dataset from the keggal website and do code in the Jupyter Notebook.

8.1 Install the dependencies:

TENSORFLOW- for fast computing

KERAS-reduces loads, simple and provides consistancy

IMUTILS-Helps in making basic image processing functions like translation,rotation,resizing easier.

NUMPY-Fast and consumes less memory

OPENCV -PYTHON-Helps in processing image to identify the faces of humans

MATPLOTLIB –Helps in plotting GUI application

ARGPARSE-Provides flexibility by parsing the arguements

SCIPY-Manipulate and Visualize the data

8.2 DATASET: We need to arrange a folder with the pictures of having mask and name the folder as with_mask and then create another folder with people not having the masks and name it as without_mask.

8.3 DATA PRE-PROCESSING:In this Data Preproocessing,we need to convert all the images that is the pictures of with_mask and without_mask into arrays.So with all those arrays we are creating a Deep-Learning Model.In this stage,we are writing our code into train_mask_directory.py where we are mentioning all the directories, dataset to convert the images into arrays.Two lists are created in this data processing,they are data[] and labels[].data[] contains all the images and labels[] consists of images after labeling them into with_mask and without_mask.Target size of our model is 224 x 224.

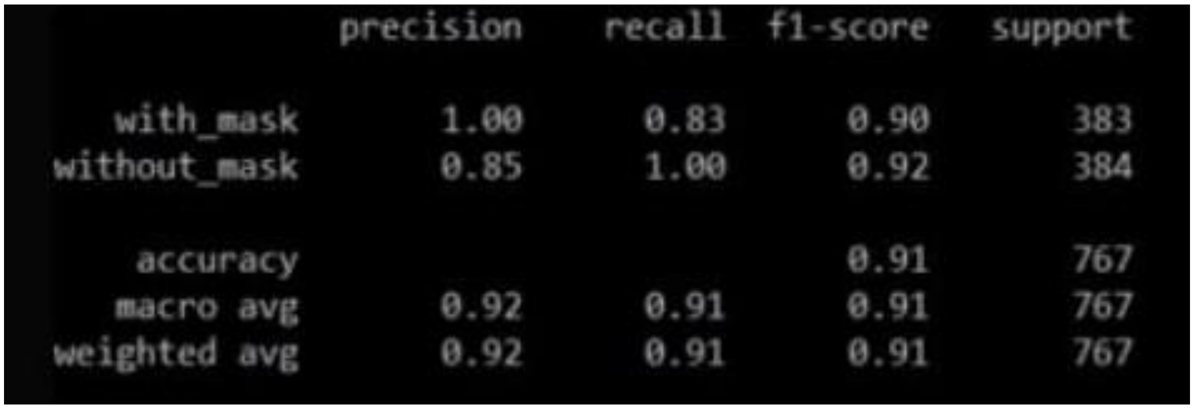
8.4 TRAINING THE MODEL:In this step,we are training the model with given dataset.In order to train the model we need to

- Initialize the learning rate i.e..,1e-4~0.00001(if learning rate is less then loss will be calculated properly with good accuracy),number of epochs to train(20) and the batch size(32).

- Insert the list into dataset directory and then initialize the list of data.
- Perform one-hot encoding on the labels.
- Construct the training image generator for data augmentation.
- Construct the base model -AveragePooling2D with the size (7,7),Flatten,Dense with relu activation layer,Dropout and dense with softmax activation.
- Plot the training loss and Accuracy and image will be saved as plot.png
- And this model is saved as mask_detector.model

8.5 RUN AND VIEW THE ACCURACY:

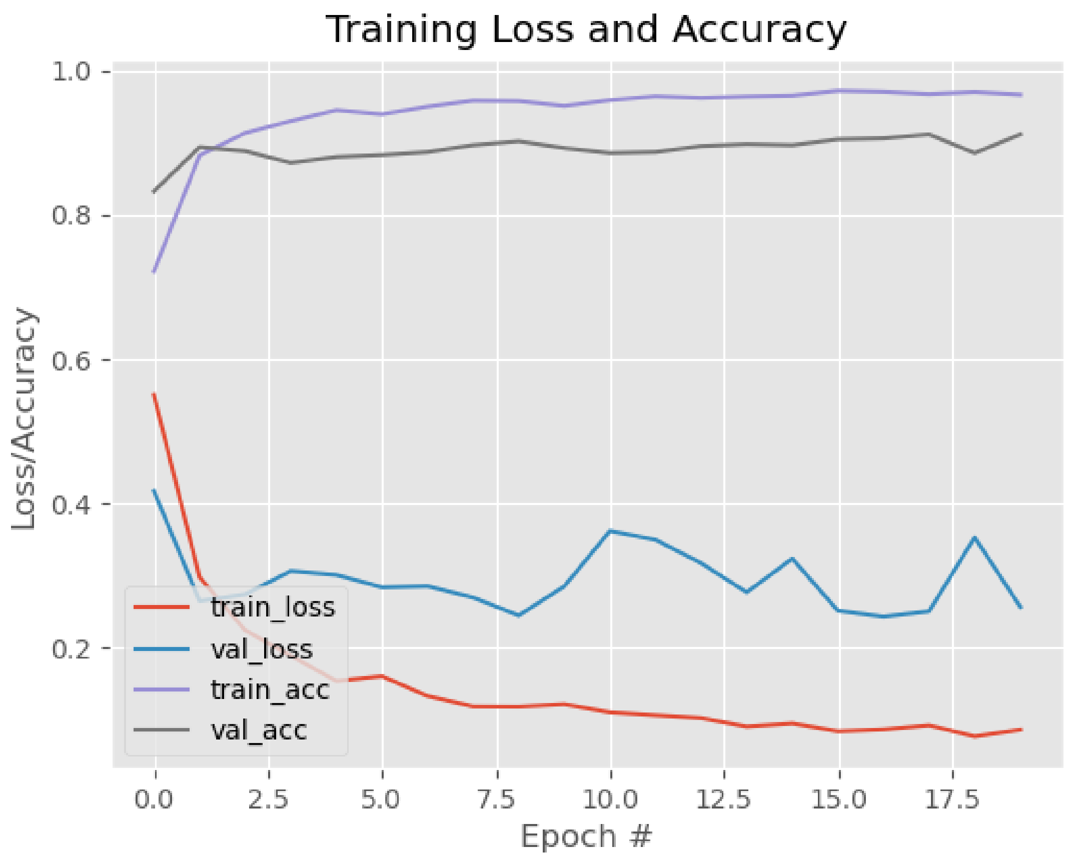
In this training loss and accuracy, on X axis we will be having Loss/Accuaracy and on Y axis we will be having Epoch#.For plotting the graph,it uses ggplot2.And the picture of the accuracy is saved in the computer as “plot.png”.As we can see that,in this graph accuracy is good and loss is also being reduced.So,that we can conclude that we got a good model.



```
              precision    recall  f1-score   support

   with_mask       1.00      0.83      0.90       383
without_mask       0.85      1.00      0.92       384

    accuracy                           0.91       767
   macro avg       0.92      0.91      0.91       767
weighted avg       0.92      0.91      0.91       767
```

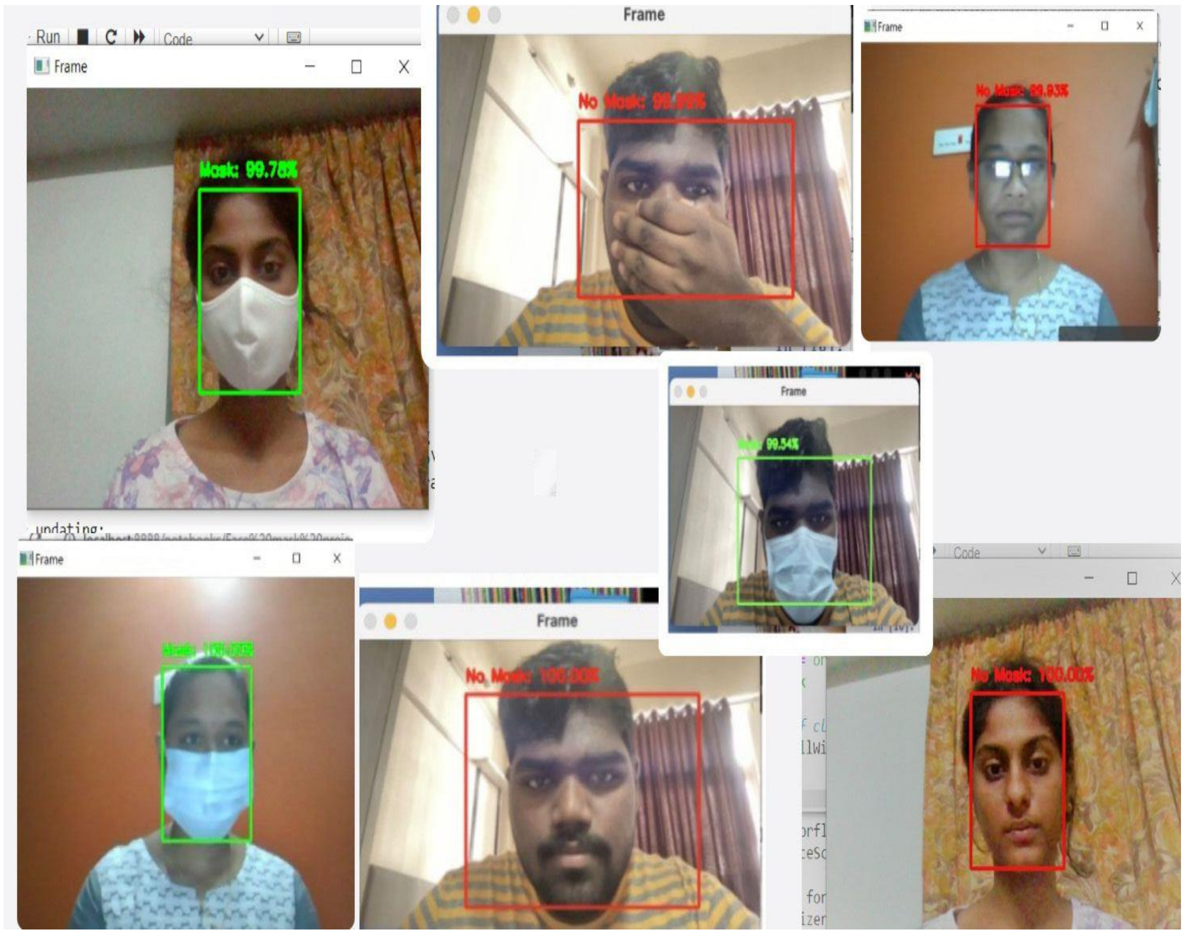


8.6 DETECT THE FACE: We created a folder called face_detector where it has two sub-folders called deploy.prototxt and res10_300x300_ssd_iter_140000.cafemodel to detect the face of the person first.

8.7 DETECT THE MASK: For detecting whether we are wearing a mask or not.We need to write the code into detect_mask_video.py.

- We need to import the libraries required.
- Mentioning the dimensions of the frame and then constructed a blob out of it.
- Passing the blob through the network to detect faces.
- Initialize the list of faces with their corresponding locations and predictions from mask
- Filtering the weak detections
- Need to ensure whether the dimensions are in frame or not.
- Need to add rectangle box for faces.
- Need to load the face mask detector model from disk.
- Initializing the video stream
- Including the probability in the label
- Displaying the output frame
- If the mask is detected then it will display in green rectangle box with probability otherwise it will show in red rectangle box with the probability of not wearing.

9 OUTPUT:


Frame
Mask: 99.78%
No Mask: 99.99%
No Mask: 99.93%
Mask: 99.54%
Mask: 100.00%
No Mask: 100.00%
No Mask: 100.00%

10 CODE:

train_mask_detector.py

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.utils import to_categorical
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
import os
INIT_LR = 1e-4
EPOCHS = 20
BS = 32
DIRECTORY = r"C:\Users\HP\Desktop\Realtime face mask detection\Face-Mask-Detection-
master\dataset"
CATEGORIES = ["with_mask", "without_mask"]
print("[INFO] loading images...")
data = []
labels = []
for category in CATEGORIES:
   path = os.path.join(DIRECTORY, category)
   for img in os.listdir(path):
         img_path = os.path.join(path, img)
         image = load_img(img_path, target_size=(224, 224))
         image = img_to_array(image)
         image = preprocess_input(image)

         data.append(image)
         labels.append(category)
lb = LabelBinarizer()
labels = lb.fit_transform(labels)
labels = to_categorical(labels)
```

```
data = np.array(data, dtype="float32")
labels = np.array(labels)
(trainX, testX, trainY, testY) = train_test_split(data, labels,
        test_size=0.20, stratify=labels, random_state=42)
aug = ImageDataGenerator(
        rotation_range=20,
        zoom_range=0.15,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.15,
        horizontal_flip=True,
        fill_mode="nearest")
baseModel = MobileNetV2(weights="imagenet", include_top=False,
        input_tensor=Input(shape=(224, 224, 3)))
headModel = baseModel.output
headModel = AveragePooling2D(pool_size=(7, 7))(headModel)
headModel = Flatten(name="flatten")(headModel)
headModel = Dense(128, activation="relu")(headModel)
headModel = Dropout(0.5)(headModel)
headModel = Dense(2, activation="softmax")(headModel)
model = Model(inputs=baseModel.input, outputs=headModel)
for layer in baseModel.layers:
        layer.trainable = False
print("[INFO] compiling model...")
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
model.compile(loss="binary_crossentropy", optimizer=opt,
        metrics=["accuracy"])
print("[INFO] training head...")
H = model.fit(
        aug.flow(trainX, trainY, batch_size=BS),
        steps_per_epoch=len(trainX) // BS,
        validation_data=(testX, testY),
        validation_steps=len(testX) // BS,
        epochs=EPOCHS)
print("[INFO] evaluating network...")
predIdxs = model.predict(testX, batch_size=BS)
predIdxs = np.argmax(predIdxs, axis=1)
print(classification_report(testY.argmax(axis=1), predIdxs,
        target_names=lb.classes_))
print("[INFO] saving mask detector model...")
model.save("mask_detector.model", save_format="h5")
N = EPOCHS
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), H.history["accuracy"], label="train_acc")
plt.plot(np.arange(0, N), H.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.savefig("plot.png")
```

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 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, (224, 224),
                (104.0, 177.0, 123.0))


        faceNet.setInput(blob)
        detections = faceNet.forward()
        print(detections.shape)


        faces = []
        locs = []
        preds = []


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

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


                if confidence > 0.5:

                        box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
                        (startX, startY, endX, endY) = box.astype("int")


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


                        face = frame[startY:endY, startX:endX]
                        face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
                        face = cv2.resize(face, (224, 224))
                        face = img_to_array(face)
                        face = preprocess_input(face)

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

        if len(faces) > 0:

                faces = np.array(faces, dtype="float32")
                preds = maskNet.predict(faces, batch_size=32)

        return (locs, preds)


prototxtPath = r"face_detector\deploy.prototxt"
weightsPath = r"face_detector\res10_300x300_ssd_iter_140000.caffemodel"
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)


maskNet = load_model("mask_detector.model")

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


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()
```

11 <u>ACCURACY</u>:

Keras, Numpy,Tensorflow are used to train the model in the text editor. After completing all 20 epoch tests and training, the accuracy was 0.9661~96 percent, and the loss was decreased (by epochs).

12 <u>CONCLUSION:</u>

In this project, we Learn how to build a face mask detector using Keras, Tensorflow, MobileNet and OpenCV. We can apply this to live cam recorders. Further improvements allow these types of models to be incorporated into CCTV cameras to recognize and identify people without masks. The face mask detector did not use a dataset containing a morphed masked image. The model is accurate and uses the MobileNetV2 architecture.Therefore, this system can be used in real-time applications that require face mask detection for security reasons due to the outbreak of Covid19. This project can be used in various public places like metros, marts, malls, functions, etcetra.

<u>13. FUTURE IMPROVEMENTS:</u>

- In future,we can develop our model by detecting whether a person is having a mask or not when he/she kept an obstacle near mask area.
- Detect whether the mask is in correct position or not.
- By including additional pictures(like phone used as mask,hand used as a mask,etcetra) into our dataset.

14. <u>REFERENCES</u>:

[1] I. Buciu, "Color quotient-based mask detection," 2020 International Symposium on

ISETC, 2020. [2] Trunk-branch ensemble cnn for video-based face recognisation. C Ding, 2017 [3] T. Meenpal, A. Balakrishnan and A. Verma, "Facial Mask Detection using Semantic Segmentation," 2019 4th International Conference on Computing, Communications and Security (ICCCS), 2019[4]M. R. Bhuiyan, S. A. Khushbu and M. S. Islam, "A Deep Learning-Based Assistive System to Classify COVID-19 Face Mask for Human Safety with YOLOv3, "ICCCNT 2020, Kharagpur, India, 2020, pp. 1-5, October 2020. [5] C. Jagadeeswari, M. Uday Theja, "Performance Evaluation of Intelligent Face Mask Detection System with various Deep Learning Classifiers", IJAST, vol. 29, no. 11s, pp. 3083 - 3087, May 2020.