



# Deep Learning Project

Face Mask Detection



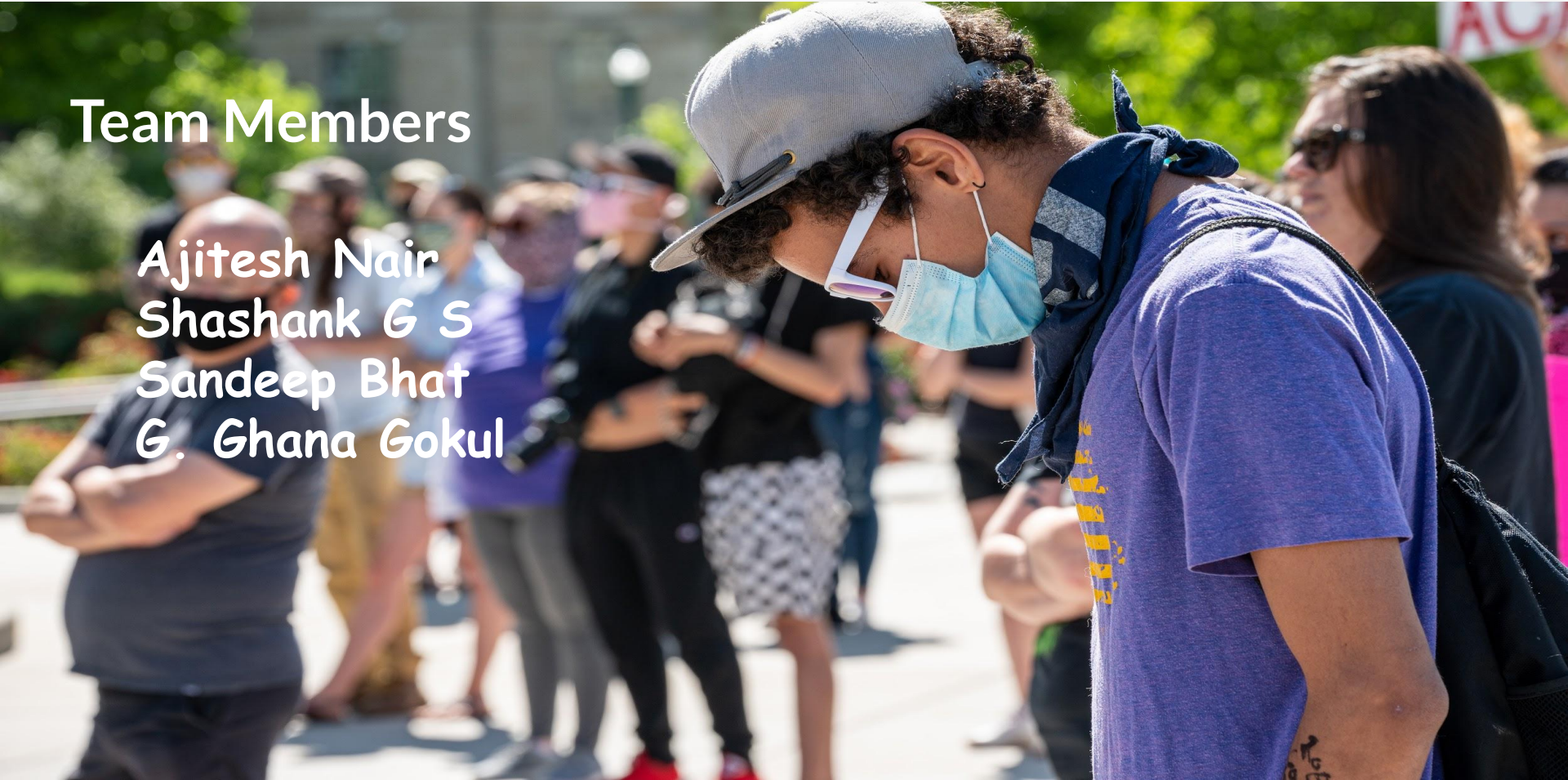
# OUTLINE

1. Title and Team Members
2. Problem Statement
3. Literature Survey
4. Architecture/Flow Diagram
5. Data Preprocessing
6. Data Modelling
7. Evaluation and Detecting Face mask
5. Software and Hardware Requirements
6. Final Deliverables
7. Team Info
8. References

# FACE MASK DETECTION

## Team Members

Ajitesh Nair  
Shashank G S  
Sandeep Bhat  
G. Ghana Gokul





# Problem statement

The use of facial masks in public spaces has become a social obligation since the wake of the COVID-19 global pandemic and the identification of facial masks can be imperative to ensure public safety. Detection of facial masks in video footages is a challenging task primarily due to the fact that the masks themselves behave as occlusions to face detection algorithms due to the absence of facial landmarks in the masked regions. In this work, we propose an approach for detecting facial masks in videos using deep learning.

# An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network (2020 IEEE)

In this paper, They proposed 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 (CCTV) cameras. While a person without a mask is detected, the corresponding authority is informed through the city network.

From the video footage facial images are extracted and CNN is used for feature extraction from the images.<sup>4</sup>

## Methodology

- Dataset collection
- Deep Learning Architecture
- Screening and Informing the Authority

## Results

The model is trained for 100 epochs. This system detects a face mask with an accuracy of 98.7%. and has AUC of 0.985

## Deep Learning Framework to Detect Face Masks from Video Footage (IEEE 2020)

In this paper, They proposed a system that proposed framework capitalizes on the MTCNN face detection model to identify the faces and their corresponding facial landmarks present in the video frame. These facial images and cues are then processed by a neoteric classifier that utilises the MobileNetV2 architecture as an object detector for identifying masked regions.

### A. Face Detection

For the task of face detection, they utilized the Multi-Task Cascaded Convolutional Neural Network (MTCNN) as the baseline model. The model is a cascaded structure comprising three stages of deep convolutional networks that predict the facial landmarks.

**Stage 1** consists of a Fully Convolutional Network (FCN) called Proposal Network (P-Net) [14], which is used to obtain the potential candidate windows in the input image pyramid and their bounding box regression vectors.

**Stage 2** consists of a CNN called Refine Network (RNet)[14] to which all the candidate windows obtained from the previous stage are fed. R-Net mainly works to filter these candidate windows.

**Stage 3** comprises of a CNN called O-Net [14], which is responsible for proposing facial landmarks from the candidate facial regions obtained from the previous stage.



# Deep Learning Framework to Detect Face Masks from Video Footage (IEEE 2020)



## **B. Facial Mask Prediction :**

For the task of identifying faces which are covered by a facial mask, they utilised the MobileNetV2 architecture , which is an effective feature extractor for object detection and segmentation.

There are a total of 3 convolutional layers in a block, where the latter two are: a depthwise convolution that filters the input and a 11 point-wise convolution.

We utilise this base model of the MobileNetV2 architecture as a feature extractor for facial mask detection.

After passing through a ReLU activation function, they use a softmax function to get the probability distribution over the predicted classifications. This is how the facial mask classifier is able to predict whether a subject in a given frame is wearing a facial mask or not.

# Deep Learning Framework to Detect Face Masks from Video Footage (IEEE 2020)

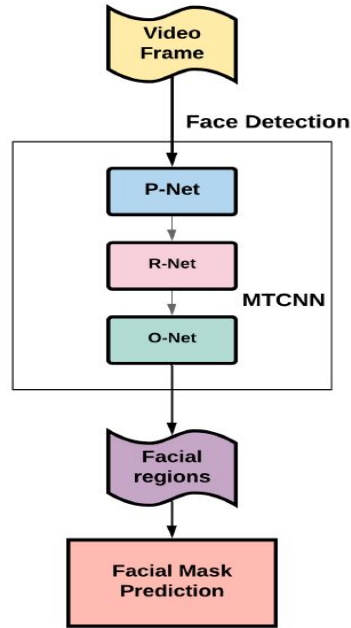


Fig. 1: Workflow of proposed framework

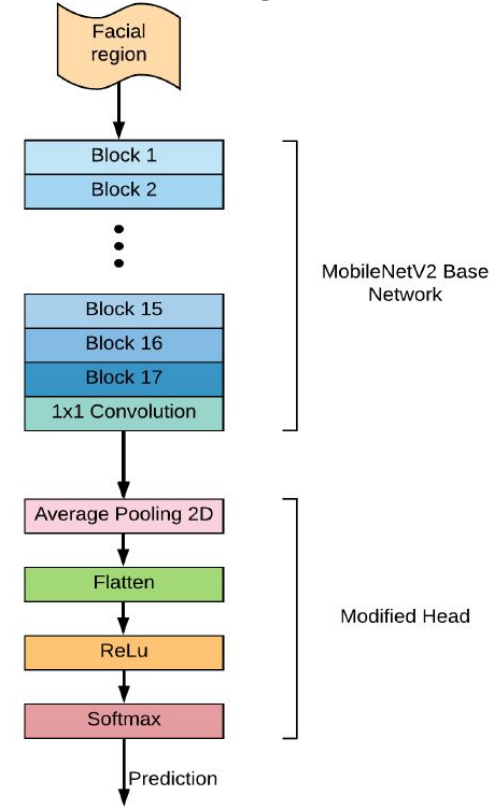



Fig. 3: Facial mask classifier constructed using MobileNetV2 architecture



# MASKED FACE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK (IEEE 2019)



In the real-world, when a person is uncooperative with the video surveillance systems, then wearing a mask is common. In the presence of these masks, current face recognition performance degrades. An abundant number of researches work has been performed for recognizing faces under different conditions like changing pose or illumination, degraded images, etc. Still, difficulties created by masks are usually disregarded. The primary concern to this work is about facial masks, and especially to enhance the recognition accuracy of different masked faces.

## Methodology

**Input Dataset:** IIIT-Delhi Disguise Version 1 Face Database, Masked Face Dataset (MFD)

## Masked Face Detection Using MTCNN:

Multi-task Cascaded Convolutional Neural Network The model first rescaled the image to a certain extent of sizes. It is called an image pyramid. Then the candidate facial regions are introduced by the first network, P-Net or Proposal Network. The second network known as R-Net or Refine Network refines the bounding boxes. And finally, the third network, O-Net or Output Network determines facial landmarks from the image. Those networks P-Net, R-Net, and O-Net can perform classification of face, regression of bounding box and localization of facial landmarks. That's why this model is known as a multi-task network.

# MASKED FACE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK (IEEE 2019)



## **Feature Extraction using FaceNet:**

Developed by Google. This pre-trained FaceNet model is used here as a baseline for a deep network. The FaceNet is combined with a batch layer and a very deep CNN network.

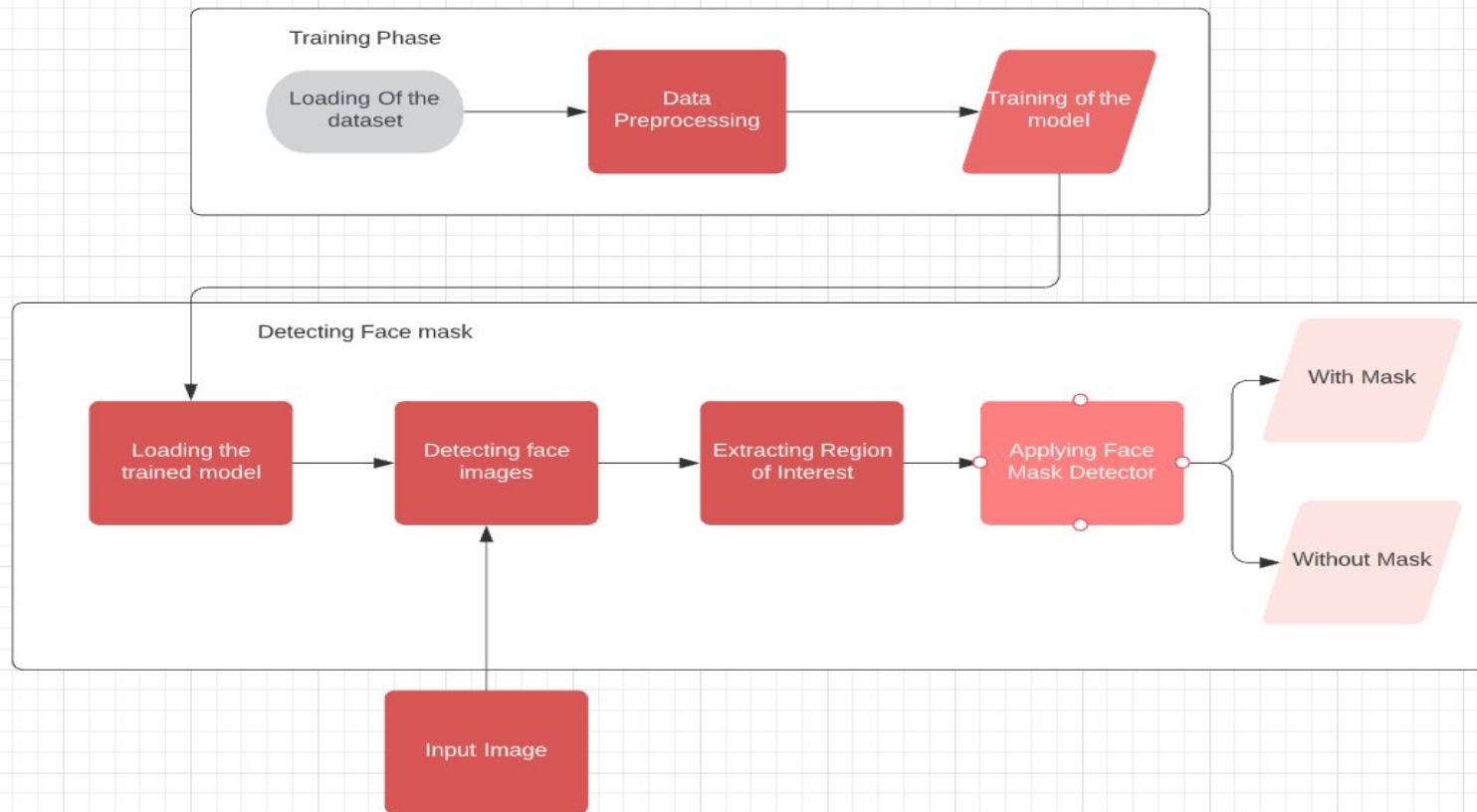
## **Face Verification Using SVM:**

In this segment, we examine a test face to other train faces using SVM. The classification result is considered as correct if the distance among the test image and the train image of the identical person is minimum. A masked face similarity is measured upon the masked and unmasked face by estimating an L2 normalization within the features key points collected from the net structure.

## **Result**

They were able to achieve an accuracy of 82%

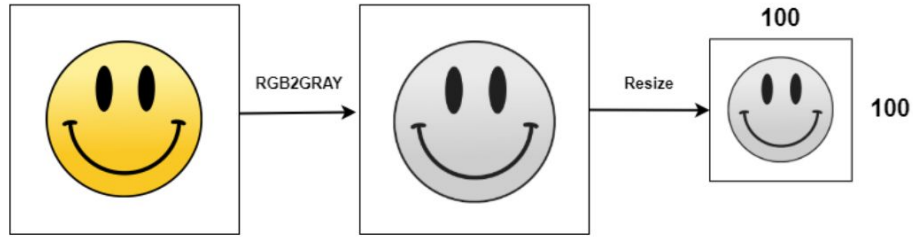
# FLOW DIAGRAM



# Project Details Breakdown



## Data Preprocessing



# DATA PREPROCESSING



```
▶ img_size=100
data=[]
target=[]

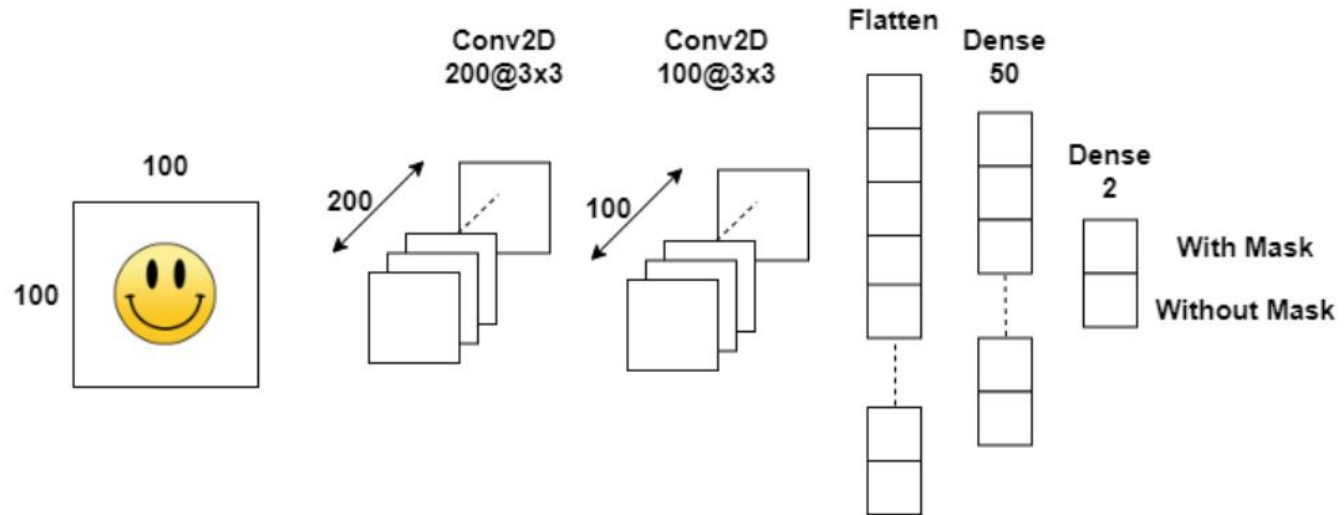
for category in categories:
    folder_path=os.path.join(data_path,category)
    img_names=os.listdir(folder_path)

    for img_name in img_names:
        img_path=os.path.join(folder_path,img_name)
        img=cv2.imread(img_path)

        try:
            gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
            #Coverting the image into gray scale
            resized=cv2.resize(gray,(img_size,img_size))
            #resizing the gray scale into 100x100, since we need a fixed common size for all the images in the dataset
            data.append(resized)
            target.append(label_dict[category])
            #appending the image and the label(categorized) into the list (dataset)

        except Exception as e:
            print('Exception:',e)
            #if any exception rasied, the exception will be printed here. And pass to the next image
```

## Convolutional Neural Network Architecture



```
▶ from keras.models import Sequential
from keras.layers import Dense,Activation,Flatten,Dropout
from keras.layers import Conv2D,MaxPooling2D
from keras.callbacks import ModelCheckpoint

model=Sequential()

model.add(Conv2D(200,(3,3),input_shape=data.shape[1:]))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#The first CNN layer followed by Relu and MaxPooling layers

model.add(Conv2D(100,(3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#The second convolution layer followed by Relu and MaxPooling layers

model.add(Flatten())
model.add(Dropout(0.5))
#Flatten layer to stack the output convolutions from second convolution layer
model.add(Dense(50,activation='relu'))
#Dense layer of 64 neurons
model.add(Dense(2,activation='softmax'))
#The Final layer with two outputs for two categories

model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

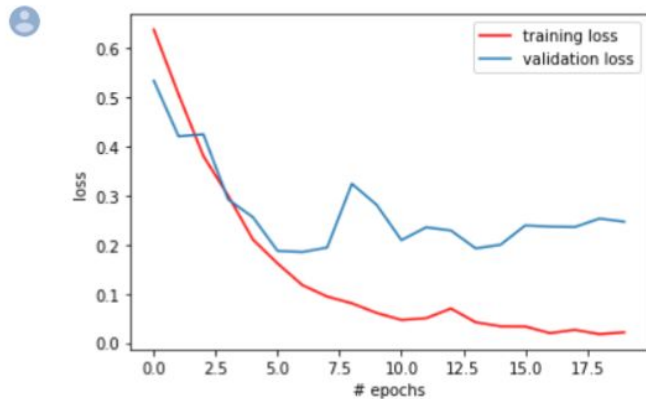


# Evaluation

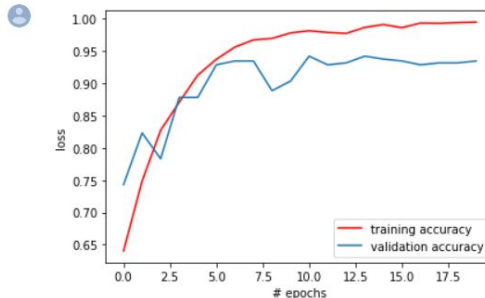


```
from matplotlib import pyplot as plt
```

```
plt.plot(history.history['loss'],'r',label='training loss')
plt.plot(history.history['val_loss'],label='validation loss')
plt.xlabel('# epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```



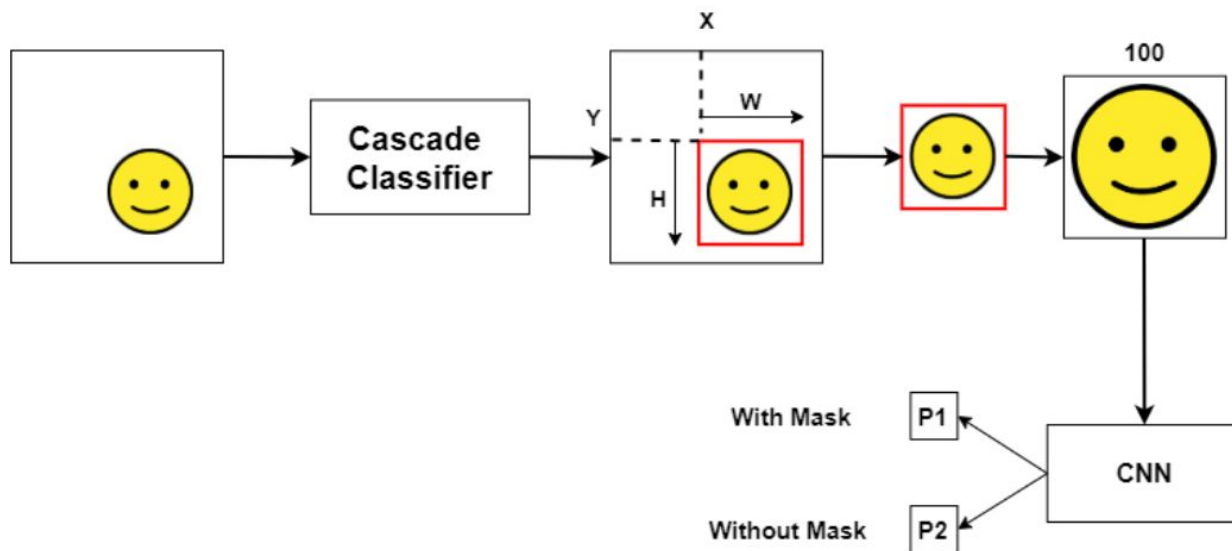
```
plt.plot(history.history['accuracy'],'r',label='training accuracy')
plt.plot(history.history['val_accuracy'],label='validation accuracy')
plt.xlabel('# epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```



```
[ ] print(model.evaluate(test_data,test_target))
```

```
27/27 [=====] - 7s 271ms/step - loss: 0.4259 - accuracy: 0.9061
[0.4258634150028229, 0.9060642123222351]
```

## Detecting Faces with and without masks



```
[ ] from keras.models import load_model
import cv2
import numpy as np
```

```
▶ model = load_model('model-006.model')

face_clsfr=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

source=cv2.VideoCapture(0)

labels_dict={0:'MASK',1:'NO MASK'}
color_dict={0:(0,255,0),1:(0,0,255)}
```

```
▶ while(True):

    ret,img=source.read()
    gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces=face_clsfr.detectMultiScale(gray,1.3,5)

    for x,y,w,h in faces:

        face_img=gray[y:y+w,x:x+w]
        resized=cv2.resize(face_img,(100,100))
        normalized=resized/255.0
        reshaped=np.reshape(normalized,(1,100,100,1))
        result=model.predict(reshaped)

        label=np.argmax(result,axis=1)[0]

        cv2.rectangle(img,(x,y),(x+w,y+h),color_dict[label],2)
        cv2.rectangle(img,(x,y-40),(x+w,y),color_dict[label],-1)
        cv2.putText(img, labels_dict[label], (x, y-10),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)

    cv2.imshow('LIVE',img)
    key=cv2.waitKey(1)

    if(key==27):
        break

cv2.destroyAllWindows()
source.release()
```

# REQUIREMENTS



## Software Requirements:

**Language** :Python 3.5 -3.7

**Tools** :Tensorflow and Keras

**Libraries** :Open Cv

,numpy,Scipy,Sklearn etc

Python 3.5–3.7.

**Os**:Windows 7 or later,macOS 10.12. 6  
Ubuntu 16.04 or later.

## Hardware Requirements:

a. Monitor screen - This will display the information to the user through screen

Camera/webcam : To capture images(Normally present in most of the computer/laptops.)



Processor requirement :Intel i5 or above /amd ryzen 3000 series or above

Mouse -The user can see various options available and do operation with help of this the software can interact with the movement of the mouse and its buttons

Keyboard-The software will interact with the keystrokes of the keyboard.

# Deliverables

1. Selecting appropriate dataset
2. Feasibility study
3. Trained model which is capable of detecting mask, classifying a person as wearing mask or not
4. Project implementation
5. Testing and evaluation
6. Outcomes /results

## References

1. M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud and J. -H. Kim, "An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), 2020, pp. 1-5, doi: 10.1109/IEMTRONICS51293.2020.9216386.
2. Joshi, Aniruddha & Joshi, Shreyas & Kanahasabai, Goutham & Kapil, Rudraksh & Gupta, Savyasachi. (2020). Deep Learning Framework to Detect Face Masks from Video Footage.
3. A. Das, M. Wasif Ansari and R. Basak, "Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV," 2020 IEEE 17th India Council International Conference (INDICON), 2020, pp. 1-5, doi: 10.1109/INDICON49873.2020.9342585.

# Team Info



Name : Sandeep Bhat

Ajitesh Nair

Shashank G S

G. Ghana Gokul.

SRN: PES2201800632

PES2201800681

PES2201800706

PES220180077

Section : 'A'.

'B'

'B'

'B'





# Thank you.

