Implementation

5Training Model

Dataset

The dataset of 1376 images was taken [1]. There are 690 images with mask and 686 images without mask.

Fig 1. with_mask

Fig 2. without_mask

Data Prep-processing

The dataset we are utilizing comprises of pictures with various tones, various sizes, and various directions. Consequently, we need to change over all the pictures into grayscale in light of the fact that we require to be certain that tone ought not be a basic point for identifying cover. From that point onward, we need to have all the pictures in a similar size (100x100) prior to applying it to the neural organization.

```
In [7]:
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)

            resized=cv2.resize(gray,(img_size,img_size))

            data.append(resized)
            target.append(label_dict[category])

        except Exception as e:
            print('Exception:',e)
```

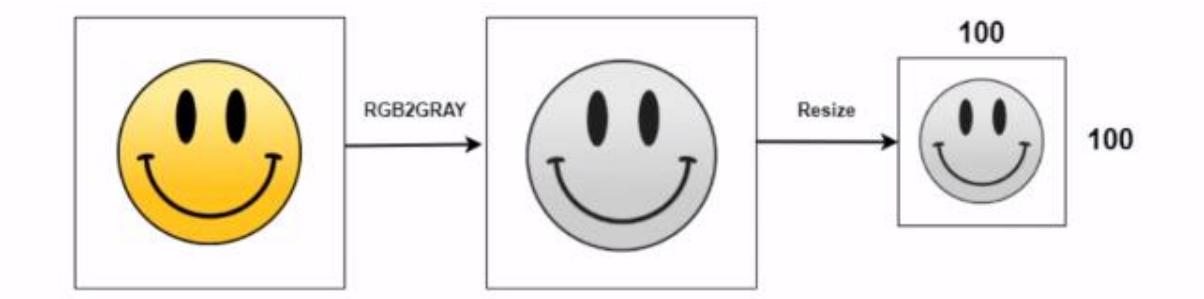


Fig 3. Data Preprocessing

Loading the saved numpy arrays

```
In [8]: import numpy as np
        data=np.array(data)/255.0
        data=np.reshape(data,(data.shape[0],img_size,img_size,1))
        target=np.array(target)
        from keras.utils import np_utils
        new_target=np_utils.to_categorical(target)

In [10]: new_target.shape

Out[10]: (1376, 2)

In [11]: np.save('images.npy',data)
         np.save('lables.npy',new_target)

In [12]: import numpy as np

In [17]: data=np.load('images.npy')

In [18]: new_target=np.load('E:\Project\lables.npy')

In [19]: data.shape

Out[19]: (1376, 100, 100, 1)
```

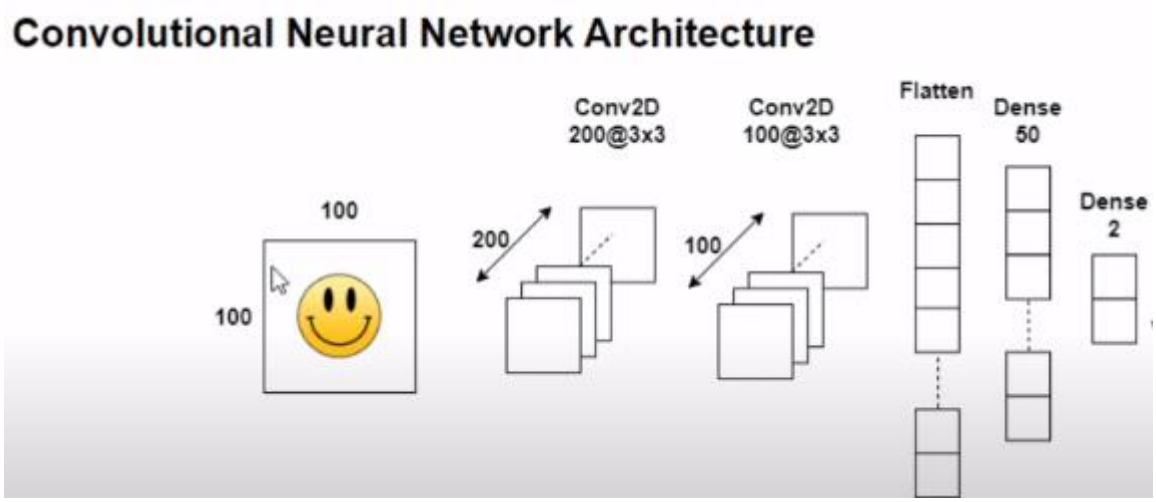
Building Sequential Model

Now, we construct our Sequential CNN model with different layers, for example, Conv2D, MaxPooling2D, Flatten, Dropout and Dense. In the last Dense layer, we utilize the 'softmax' capacity to yield a vector that gives the likelihood of every one of the two classes. Since we have two categories (with cover and without veil) we can utilize categorical_crossentropy**.** First, we will load the data from the files that we created in the previous step. Then we are making a Neural network using Convolutional and MaxPooling layers. Then, the output is flattened and fed into a fully connected Dense layer with 50 neurons and finally into layers with 2 neurons as it will output the probabilities for a person wearing a mask or not, respectively**.**

```
In [22]: model=Sequential()

         model.add(Conv2D(200,(3,3),input_shape=data.shape[1:]))
         model.add(Activation('relu'))
         model.add(MaxPooling2D(pool_size=(2,2)))


         model.add(Conv2D(100,(3,3)))
         model.add(Activation('relu'))
         model.add(MaxPooling2D(pool_size=(2,2)))

         model.add(Flatten())
         model.add(Dropout(0.5))
         model.add(Dense(50,activation='relu'))
         model.add(Dense(2,activation='softmax'))

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

Training the CNN Model

In this progression we fit our pictures in the preparation set and the test set to our Sequential model we made-up utilizing keras library. I have prepared the model for 100 epochs. In any case, we can prepare for more number of epoch to achieve higher accuracy in case there happens over-fitting.




```
In [24]: from sklearn.model_selection import train_test_split
         train_data,test_data,train_target,test_target=train_test_split(data,new_target,test_size=0.1)

In [25]: train_data.shape

Out[25]: (1238, 100, 100, 1)

In [26]: train_target.shape

Out[26]: (1238, 2)

In [27]: checkpoint = ModelCheckpoint('model-{epoch:03d}.model',monitor='val_loss',verbose=0,save_best_only=True,mode='auto')
         history=model.fit(train_data,train_target,epochs=100,callbacks=[checkpoint],validation_split=0.2)
```

```
val_loss: 0.1027

val_accuracy: 0.9758
```

We received above result after the 100 epochs iterations.

Plot our accuracy and loss curve

In [31]:
```
N = 100
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), history.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), history.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), history.history["accuracy"], label="train_acc")
plt.plot(np.arange(0, N), history.history["val_accuracy"], label="val_acc")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="center right")
plt.savefig("CNN_Model")
```

S

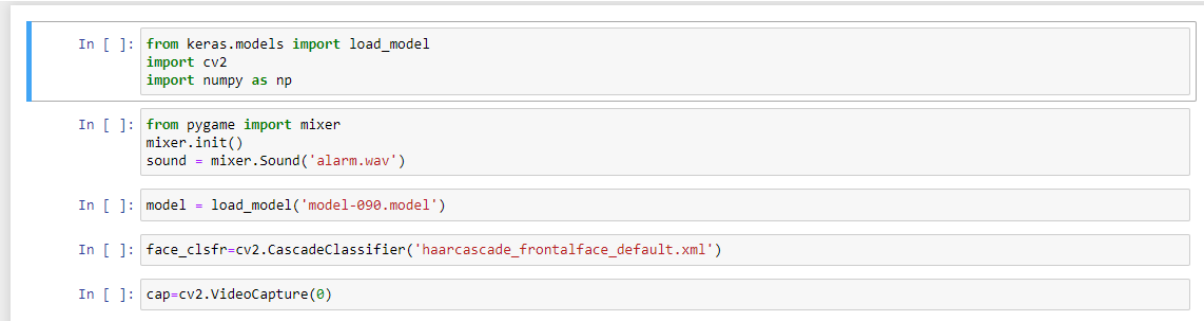
Testing Model

Loading the model

The Keras deep learning framework supports saving trained model and loading them for later use. It will predict the possibility of the two classes ([without_mask, with_mask]). Based on whose probability is higher, the label will be chosen and displayed around faces. Also by importing the mixer from pygame we added the beep alert for the person without mask.

The model -090.model is the lastly trained CNN model.

We can also add local ip survivalance camera using Droid cam application.

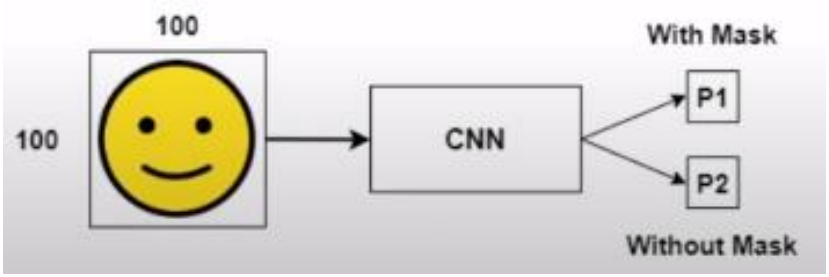


```
In [ ]: from keras.models import load_model
        import cv2
        import numpy as np

In [ ]: from pygame import mixer
        mixer.init()
        sound = mixer.Sound('alarm.wav')

In [ ]: model = load_model('model-090.model')

In [ ]: face_clsfr=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

In [ ]: cap=cv2.VideoCapture(0)
```

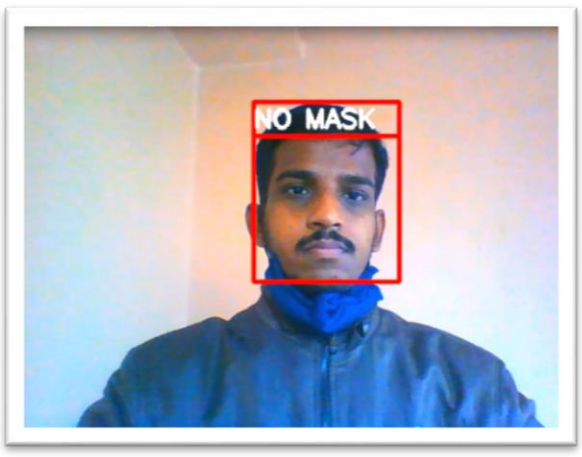
Detecting Faces with and without Masks

We need to label the two probabilities (0 for with_mask and 1 for without_mask). After that, we need to set the bounding rectangle color using RGB values. I've given RED and GREEN as two colors. In the last step, we use the OpenCV library to run an infinite loop to use our web camera in which we detect the face using the Cascade Classifier. The code webcam = cv2.VideoCapture(0) denotes the usage of webcam. The model will predict the possibility of each of the two classes ([without_mask, with_mask]). Based on which probability is higher, the label will be chosen and displayed around our faces. I had here removed the frame for with mask video because it will be easy to detect the person without mask.

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


while(True):

    ret,frame=cap.read()
    gray=cv2.cvtColor(frame,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(frame,(x,y),(x+w,y+h),color_dict[label],4)
        cv2.rectangle(frame,(x,y-40),(x+w,y),color_dict[label],4)
        cv2.putText(frame, labels_dict[label], (x, y-10),cv2.FONT_ITALIC, 1,(255,255,255),4)

        if(labels_dict[label] =='MASK'):
           print("No Beep")
        elif(labels_dict[label] =='NO MASK'):
                sound.play()
                print("Beep")

    cv2.imshow('Mask Detection App GCEK',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```



Output :



Conclusion and Future Work

Conclusion

We get mostly the accurate detection of the facemask. The dataset is only of close and similar types of mask wearing images so that the final video frame detects only the solid colour mask. It will give warning when the person detects without mask in the video stream. As the era are blooming with rising tendencies the supply so we've got novel face masks detector that could probably make a contribution to public healthcare. We used OpenCV, tensor flow, keras and CNN to stumble on whether or not humans have been carrying face mask or not. The accuracy of the version is done and, the optimization of the version is a non-stop method and we are constructing a fantastically correct answer. This unique version can be used as a use case for facet analytics. Furthermore, the proposed approach achieves modern outcomes on a public face masks dataset. By the improvement of face masks detection, we are able to stumble on if the individual is carrying a face masks and permit their access might be of tremendous assist to the society.

Future Scope

The project can be utilized in the accompanying spots to recognize individuals with or without masks:

- Offices – Manufacturers, retail, other SMEs and corporate giants
- Hospitals/healthcare organizations
- Airports and railway stations
- Sports venues
- Entertainment and hospitality industry
- Densely populated areas

Analysing the current scenario, government and private organizations want to make sure that everyone working or visiting a public or private place is wearing masks throughout the day. The face mask detection platform can quickly identify the person with a mask, using cameras and analytics. Depending upon the requirements, the system is also adaptable to the latest technology and tools.