FACE MASK DETECTION

Project Report submitted in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

Submitted by

MAYANK TOMAR (20SCSE1010871)

AKASH ATTRI (20SCSE1010867)

IN

SCHOOL OF COMPUTING SCIENCE & ENGINEERING

Under the Supervision of

:MS.ARCHANA
(ASSISTANT PROFESSOR)

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Statement of Project Report Preparation

1. Thesis title: “FACE MASK DETECTION”

2. Degree for which the report is submitted: “BACHELOR OF TECHNOLOGY 3

Project Supervisor was referred to for preparing the report.

4. Specifications regarding thesis format have been closely followed.

5. The contents of the thesis have been organized based on the guidelines.

6. The report has been prepared without resorting to plagiarism.

7. All sources used have been cited appropriately.

8 The report has not been submitted elsewhere for a degree.

TABLE OF CONTENT

ABSTRACT

COVID-19 is one of the most dangerous forms of diseases which is caused by corona virus. It is a highly transmissible disease. WHO has already declared it as a pandemic .Therefore at the current scenario, due to outbreak of this pandemic (COVID-19), face masks has become the necessary tool of everyone to avoid spread of disease to some extent. There has been a great demand for the development of software which can easily recognize person who is wearing a mask. Therefore we are going to develop a system which will fulfill the need using Deep Learning . We will use convolutional neural network to train our model. Here two categories of dataset will be used. The one which contains set of images of faces with mask and the other without face masks We will train the program with this data set to learn to decide whether a person's face is masked or not. OpenCV, tensorflow and keras will be used for the real time face detection with live stream through web camera. After the successful deployment of the product, we will be able to design a software which can be installed at various places such as at the entry gate of colleges, railway stations ,air ports ,temples, hotels and shops etc. This will easily detect the persons who are entering without face masks by using cameras of the systems. Hence this product is the need of the hour for us to develop so as to work for the safety of humans.

LIST OF FIGURES

CHAPTER -1

INTRODUCTION

There has been a great rise in wearing of face masks by people across the world. The reason behind this is the the very dangerous COVID-19 disease which has created havoc in the lives of people across the world . This is caused by corona virus which attacks on the lungs of human and causes severe respiratory problems In 2020, rapid transmission of this disease across the world made WHO to declare it as a pandemic. A large number of deaths has been occured in almost each country due to this pandemic. Keeping it in mind we have utilized the power of artificial intelligence to prepare a face mask detector which will help to distinguish people wearing face masks and those who are without face masks. As this is a highly transmissible disease so we need to wear mask while moving through public places .OpenCV has been used to do the real time face detection by live streams using web camera. Laws has been made for people to wear masks while roaming at public places but it is very difficult task to regulate persons who are not wearing masks. So we need to do this by using machines and hence Our Face masks detector will contribute a lot in controlling the entry of unmasked people from various places like railway stations, airports, colleges, temples ,hotels, shops, etc . Although vaccines are being provided in various parts of the countries but yet the disease has not gone and we need to be alert of ourselves that corona strain can infect us .So best way to avoid transmission is to always wear mask while going out of the house for any work .The main idea behind this research paper is to present a report on Face Mask Detector where we took advantage of those available technological advancements to develop classification models for Detection of masked and unmasked faces. The Problem Statement in this is we have to develop a classification model for given dataset of with mask and without mask.

CHAPTER 2:

REQUIREMENTS , FEASIBILITY AND LITERATURE REVIEW

Required tools

1. Tensorflow
2. Keras
3. Imutils
4. Numpy
5. Opencv-python
6. Matplotlib
7. Scipy
8. Jupyter Notebook

Feasibility Analysis :

Feasibility studies reflect a project's unique goals and needs, so each is different. However, the other tips below can apply broadly to undertaking a feasibility study There has been a great demand for the development of softwares which can easily recognize person who is wearing a mask.Therefore we are going to develop a system which will fullfil the need using Deep Learning.Here we will use a set of images (data set) of two categories .The one which contains persons wearing face masks and the other without face masks.

We will train the program with this data set to learn to decide whether a person is wearing a face mask or not by simply providing the image or by using camera of the system to detect the necessary.

Literature Reviews/Comparative study:

In past days facial detection models were deployed using edge, line, and centre near features and patterns were detected from those features, these methods are very effective in getting the desired result and the effort of computation is also low[1].

AdaBoost is based on regression classifier which fits with the regression function on the original set of dataset and here few false classified objects also adjust during the process of back propagation to optimize the desired results[2].

Viola Jones Detector had proposed an object model for real time which was used to detect various classes of objects. This uses 24x24 base window size to evaluate any image with line , edge and four rectangular features[3].

UNet and SNet were used for doing segmentation of heart ventricular . That model adjusted the weights in such a fashion that more weights were provided to useful features and less weights were transferred to less important features.[4]

In this paper MTCNN has been used for the identification of faces which are masked. Facial recognition is a an interesting subject in the field of CV.

CNN has the tendency to learn important features on its own. MTCNN does a great work which leaves behind several other face detection systems. It performs in three stages. At first, it makes several copies of the images of various scales which is known as image pyramid. This first stage is known as P-Net. The next stage is R-Net. This help in refining the bounding boxes. The last stage is O-Net. This gives the final touches (landmarks) on images.[5]

Satapathy, Sandeep Kumar ,et al[6]., put forward a model to identify number plate which could help cops to chase several criminal cases. They used OCR based method to identify characters in the number plate which are processed to client server based model for obtaining the detailed knowledge about the owner.

Pathaket.al.[7] put forward a multidimensional biometric authentication system which would perform efficiently in dim lightening situations .

Patel, Ashok Kumar et. al[8] put forward a model to determine the quality of iron ore by obtaining the features from sample in the factory. Assessing the quality of the ore is very important.SVR support vector regressor is used for the online determination of the quality of ore.

Problem Formulation:

The main idea behind this research paper is to present a report on Face Mask Detector where we took advantage of those available technological advancements to develop classification models for Detection of masked and unmasked faces. The Problem Statement in this is we have to develop a classification model for given dataset of with mask and without mask.

CHAPTER - 3

CONCEPTS/ALGORITHM USED

Tool Used : (I) cascade classifier:-

Cascading is a particular case of ensemble learning which is based on the concatenation of several classifiers, by using all the datas that are obtained from the output from a given classifier as additional information for the next classifier in the cascade. Unlike voting or stacking ensembles, which are multi expert systems, cascading is a multistage one. Cascading classifiers are trained with several "positive" sample views of a particular object and arbitrary "negative" images of the same size. After the classifier is trained it can be applied to an area of a photo and detect the object in query. To search for the object in the entire frame, the search window can be moved across the image and check every location for the classifier. This process is mostly and frequently used in image processing for object detection and tracking, primarily facial detection and recognition.

(ii)Convolutional Neural Network:-

A convolutional neural network also known as CNN is an artificial neural network at the so far been most popularly used for analysing images. Although image analysis has been the most widespread used to CNN ,they can also be used for other data analysis and classification problems . Most generally we can think of CNN as an artificial neural network that has some type of specialisation for being able to pick out or detect patterns and makes sense of them. This pattern detection is what makes CNN so useful for image analysis. A convolutional neural network contains an input layer, hidden layers and an output layer. In a convolutional neural network, the hidden layers include layers that perform convolutions.Typically this includes a layer that performs a dot product of the convolution kernel with the layer's input matrix.This product is generally the Frobenius inner product, and its activation function is commonly known as ReLU.

CHAPTER 4: DESIGN AND IMPLEMENTATION

Design Structure:

Layout of the model:

Below is the layout of the model which will be prepared in two phases.

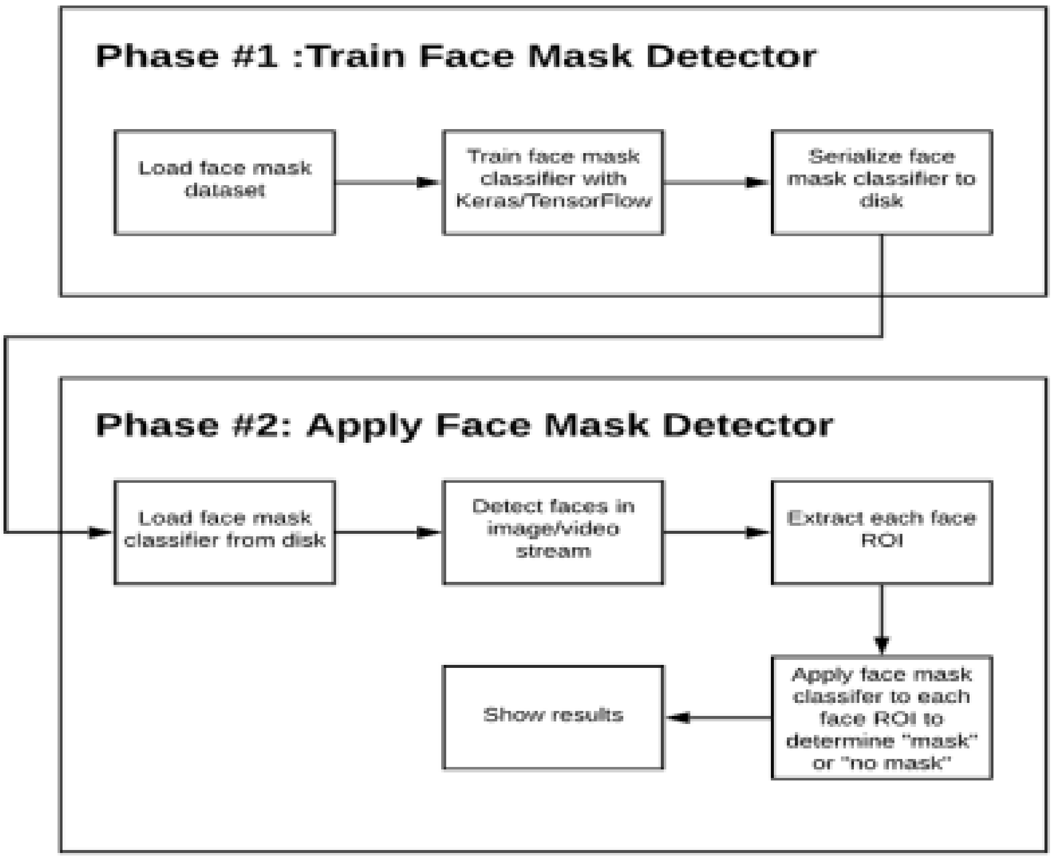

Phase #1 :Train Face Mask Detector
Load face mask dataset
Train face mask classifier with Keras/TensorFlow
Serialize face mask classifier to disk
Phase #2: Apply Face Mask Detector
Load face mask classifier from disk
Detect faces in image/video stream
Extract each face ROI
Apply face mask classifer to each face ROI to determine "mask" or "no mask"
Show results

Figure 1. Model layout

1. Training: In this phase we will train a model by passing the dataset into neural network after preprocessing of dataset images. In this way face mask detector will be serialized in the disk.

2. Deployment: After training the model we can then step forward towards loading the face mask detector, doing facial detection and then classify faces into categories of masked and without masked.

Implementation:

Data preprocessing: In this phase these dataset images of different colour ,different sizes, different orientations well be converted first to gray scale image assuming that the colour is not that much critical feature for detecting mask. Therefore we are converting these into gray and then we resize those images into 100x100 size since we need a common size of images before applying them to neural network.

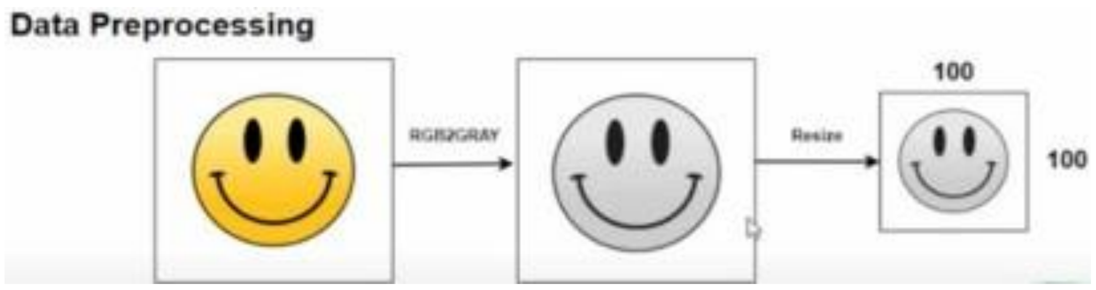


Figure 2: Data Preprocessing sample

For the implementation of the model, we import the opencv library and the os library. We give the path of the dataset. Then we load the folder names . Thereafter we create label(array) as 0 and 1 for folders inside the dataset . Then dictionary will be created for label and there we give keys as the folder names (with_mask and without_mask) and the labels as 0 and 1. we then load all the image names in a newly created list and then in a loop we need to go through each and every image inside the folder and then all the images (preprocessed images) will be loaded . Numpy will be imported and we will normalise the images and we will divide the images by 255.0 which will convert the pixel range into 0 and 1. Then we will reshape into 4 dimensional array as the neural networks need 4-dimensional array . We need to convert those to categorical representation since the last layer of the neural network has neurons with_mask and without_mask in the categorical representation .Thereafter we save the data and the target so data will be containing the images and the target will be containing whether its a face with mask or without mask. Convolutional neural network will be used for this application. It has two convolutional layer and we input the 100x100 image for those and two convolutional networks are having 200 kernels Convo2D 3x3 ,the first ,and the second convolutional layer has 100 Convo2D3x3 kenels and the we flatten those convolutions that we are going to get after that convolutional layer then finally it will be connected to dense layer of 50 neurons and the last layer will be the output layer and the output layer will have two neurons for with_mask and without_mask. Then saved dataset will be loaded. And passed through relu layer and pooling layer then a flatten layer and dropout layer to get rid of overfitting and then we have the 50 neuron dense layer and the last layer. Then we will split the dataset into training and testing with the help of scikit learn tool and train the model. After evaluation of the model it gives the testing accuracy of 96.

1. **Data Preprocessing:**

```
import cv2,os

data_path='dataset'
categories=os.listdir(data_path)
labels=[i for i in range(len(categories))]

label_dict=dict(zip(categories,labels)) #empty dictionary

print(label_dict)
print(categories)
print(labels)
```

```
 {'with mask': 0, 'without mask': 1}
['with mask', 'without mask']
[0, 1]
```

```
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 50x50, since we need a fixed common size for all the images in the data set
 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
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)
np.save('data',data)
np.save('target',new_target)
```

2. <u>Training the CNN:</u>

```
import numpy as np


data=np.load('data.npy')

target=np.load('target.npy')
#loading the save numpy arrays in the previous code


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'])

from sklearn.model_selection import train_test_split

train_data,test_data,train_target,test_target=train_test_split(data,target,test_size=0.1)

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=20,callbacks=[checkpoint],validation_split=0.2)

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()

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

Graphs:

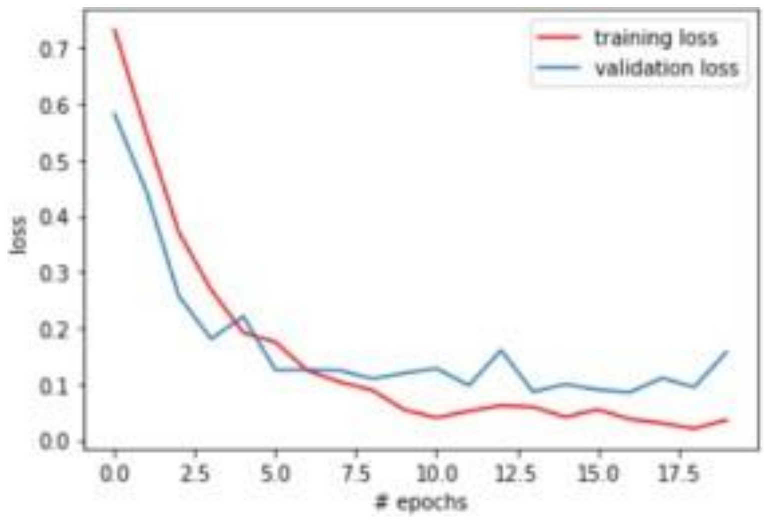


Figure 3: It shows a graph of training loss and validation loss

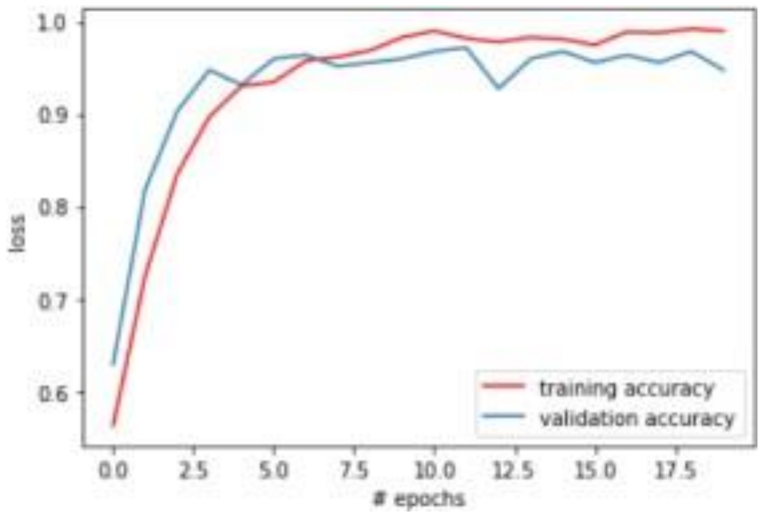


Figure 4: It shows graph of training accuracy and validation accuracy

3. **Detecting Masks:**

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

model = load_model('model-017.model')

face_clsfr=cv2.CascadeClassifier('haarcascade_frontalface_default.xml'

) source=cv2.VideoCapture(2)


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()
```

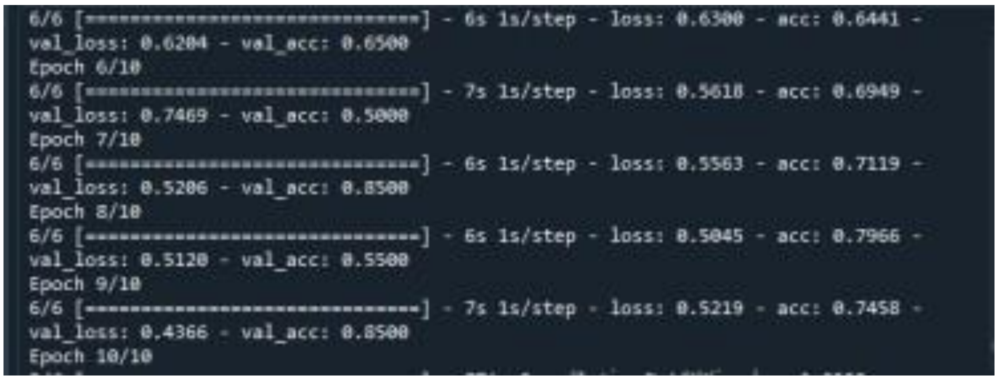
CHAPTER-5 : RESULT

We have used around 1500 images they are divided into training set and testing set these

images consists of with mask and without mask. Training testing split is 80%,20%

respectively. Then these images are trained to the M-CNN model. We have used different convolution, max pool and flatten , dropout, fully connected layers. We have used adam as optimization function and cross entropy as loss function network was trained in 100 epochs and achieved 96.0 accuracy . This model can able to detect single person in an image or multiple persons in a single image into two classes with mask and with out mask effectively around 96.5 accurately.

We have used cap = cv2.VideoCapture(0) to capture videos from web cam and this and cascade classifier for object detection from the live web cam.



```
6/6 [==============================] - 6s 1s/step - loss: 0.6300 - acc: 0.6441 -
val_loss: 0.6204 - val_acc: 0.6500
Epoch 6/10
6/6 [==============================] - 7s 1s/step - loss: 0.5618 - acc: 0.6949 -
val_loss: 0.7469 - val_acc: 0.5000
Epoch 7/10
6/6 [==============================] - 6s 1s/step - loss: 0.5563 - acc: 0.7119 -
val_loss: 0.5206 - val_acc: 0.8500
Epoch 8/10
6/6 [==============================] - 6s 1s/step - loss: 0.5045 - acc: 0.7966 -
val_loss: 0.5120 - val_acc: 0.5500
Epoch 9/10
6/6 [==============================] - 7s 1s/step - loss: 0.5219 - acc: 0.7458 -
val_loss: 0.4366 - val_acc: 0.8500
Epoch 10/10
```

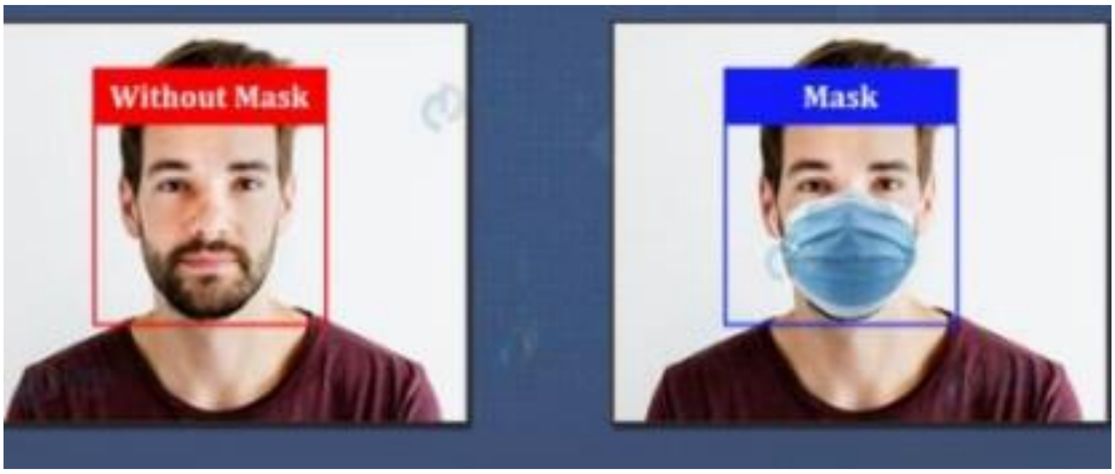


Figure5: Sample Output of the product

CHAPTER-6: DESCRIPTION OF USED TOOLS

OpenCV:

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms,which includes a comprehensive set of both classic and state-ofthe art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.Along withwell-established companies like Google, Yahoo, Microsoft,Intel,IBM, Sony,Honda,Toyota that employ the library, there are many startups such as Applied Minds, Video Surf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects atWillow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan

Tensorflow:

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. Itis used for both research and production at Google, TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). Tensor Flow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. The name Tensor Flow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors. During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from Google.Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems, not least RankBrain in Google search and the fun DeepDream project.It can run on single CPU systems, GPUs as well as mobile devices and large scale distributed systems of hundreds of machines.

1. Keras:

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides. Keras contains numerous implementations of commonly used neuralnetwork building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel. Keras is a minimalist Python library for deep learning that can run on top of Theano or Tensor Flow. It was developed to make implementing deep learning models as fast and easy as possible for research and development.It runs onPython 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks. It is released under the permissive MIT license.

CHAPTER - 7

LIMITATIONS AND FUTURE SCOPE OF THE PROJECT

Limitations:

1. The product is not able to detect objects from a distance greater than the 2

metres. 2. The product is unable to detect different objects in the same from

simultaneously. 3. The product is unable to detect transparent masks.

4. The product is not able to do recognitions related to light brightness frames.

<u>Involved Merits :</u>

➢ A way to control the spread of pandemic to some extent.
➢ A software to distinguish persons with and without masks.

➢ can be helpul at entry gates of schools and colleges.

can also be helpul at railway stations, airports ,temples and other crowded places.

➢ Best used with cameras installed in executable devices.

<u>Future Scope of the Project:</u>

1. The product is in demand as per the present scenario of the pandemic covid 19.

2. In future also this product can be very useful in case of scanning a masked face during entries at hospitals .

3. With the advancement of technology, the modified product can deal with the distant object too which will be great for the needs.

4. This product can be modified to detect various objects in a single frame simultaneously with use of some advanced algorithms.

5. The product can be very useful in making a thought of reopening schools and colleges with mandatory scanning of students and faculties and staff at the entry gates using this device.

CONCLUSION

Since the latest technology are stepping forward with trends in the availability therefore we have developed a real time face mask detector which can possibly contribute to the health of people. We have used convolutional neural network to train the model by image analysis . we have used cascading classifiers to get the images of dataset in desired size and orientation before passing them to neural network. Our models were tested with images and real-time streams. The model fulfills the required accuracy and hence is suitable for the deployment. With the development of the Face Mask Detector, it would be very easy for different working stations to detect the persons who are wearing masks and accordingly control their entrance at the entry gates.

REFERENCES:

For dataset

https://github.com/prajnasb/observations

[1]. Satapathy, Sandeep Kumar, et al. "Deep learning based image recognition for vehicle number information." *International Journal of Innovative Technology and Exploring Engineering* 8 (2019): 52-55.

[2]. Pathak, Mrunal, Vinayak Bairagi, and N. Srinivasu. "Multimodal Eye Biometric System Based on Contour Based E-CNN and Multi Algorithmic Feature Extraction Using SVBF Matching."*International Journal of Innovative Technology and Exploring Engineering*

[3]. Patel, Ashok Kumar, Snehamoy Chatterjee, and Amit Kumar Gorai. "Development of a machine vision system using the support vector machine regression (SVR) algorithm for the online prediction of iron ore grades." Earth Science Informatics 12.2 (2019): 197-210.

[4]. M. S. Ejaz and M. R. Islam, "Masked Face Recognition Using Convolutional Neural Network," 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), Dhaka, Bangladesh, 2019, pp. 1-6, doi: 10.1109/STI47673.2019.9068044.

[5]. P.Viola and M. Jones, "Rapid object detection using a boosted cascadeof simple features," in *Proceedings of the 2001 IEEE Computer SocietyConference on Computer Vision and Pattern Recognition. CVPR 2001*,vol. 1, Dec 2001, pp. I–I.