PROJECT REPORT

On

Real-Time Face Mask Detector Using Python, OpenCV, Keras

(CST SPL 1 IIIrd Semester Mini Project)

Submitted to:

Mr. Hemant Singh Pokhariya

(CC-CST SPL 1- III Sem)

Guided By:

Dr.Vishen Gupta

(Resource Person)

Submitted By:

Mr. Piyush Rana

Roll No.:2017523

CST SPL 1-III Sem

Session – 2021-2022

ABSTRACT

After the breakout of the worldwide pandemic COVID-19, there arises a severe need of protection mechanisms, face mask being the primary one. According to the World Health Organization, the corona virusCOVID-19 pandemic is causing a global health epidemic, and the most successful safety measure is wearing a face mask in public places. Convolutional Neural Networks (CNNs) have developed themselves as a dominant class of image recognition models. The aim of this research is to examine and test machine learning capabilities for detecting and recognize face masks worn by people in any given video or picture or in real time. This project develops a real-time, GUI-based automatic Face detection and recognition system. It can be used as an entry management device by registering an organization's employees or students with their faces, and then recognizing individuals when they approach or leave the premises by recording their photographs with faces. The proposed methodology makes uses of Principal Component Analysis (PCA) and HAAR Cascade Algorithm. Based on the performance and accuracy of our model, the result of the binary classifier will be indicated showing a green rectangle superimposed around the section of the face indicating that the person at the camera is wearing a mask, or a red rectangle indicating that the person on camera is not wearing a mask along with face identification of the person.

PROPOSED METHODOLOGY

We use Convolutional Neural Network and Deep Learning for Real Time Detection and Recognition of Human Faces, which is simple face detection and recognition system is proposed in this paper which has the capability to recognize human faces in single as well as multiple face images in a database in real time with masks on or off the face. Pre-processing of the proposed frame work includes noise removal and hole filling in colour images. After pre-processing, face detection is performed by using CNNs architecture. Architecture layers of CNN are created using Keras Library in Python. Detected faces are augmented to make computation fast. By using Principal Analysis Component (PCA) features are extracted from the augmented image. For feature selection, we use Sobel Edge Detector.

PROGRAMMING LANGUAGE USED:

In this project the programming language used is python. Python is an interpreter, interactive, object- oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. Python combines remarkable power with very clear syntax. It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface. Finally, Python is portable: it runs on many Unix variants, on the Mac, and on Windows 2000 and later.

PRE-REQUISITE FOR THE PROJECT:

1. Python
2. Machine learning
3. Deep learning
4. Keras API
5. OpenCV API

METHODOLY FOLLOWED

1. INSTALLATION OF THE DEPENDANCIES:

Some dependencies must be installed in order to successfully compilation of this project. They are:

- Tensorflow
- Keras
- OpenCV
- Numpy
- Matplolib
- Scipy

2. DATASET:

The dataset was divided into 2 categories that is with mask and without mask so that the model can have better accuracy compared to the model which are just trained with just the face mask data and it is mainly used for the input dataset for training to detect for a real time video.

3. DATA PREPROCESSING:

Most of the images have been augmented using open- source computer vision library (OPEN CV) which was then applied to all the input raw images to that those can be converted to clean versions which could easily be used as input for the neural network.

4. EXTRACTING THE IMAGES FROM VIDEO SAMPLES:

We cannot train the neural network in the video data; we need different and a huge number of images to train the model in order to train the machine. The video specifications are at 30 frames per second and all these frames get converted directly into images and then hereby result to a total of 60,000 images .

5. DATA ACCRETION:

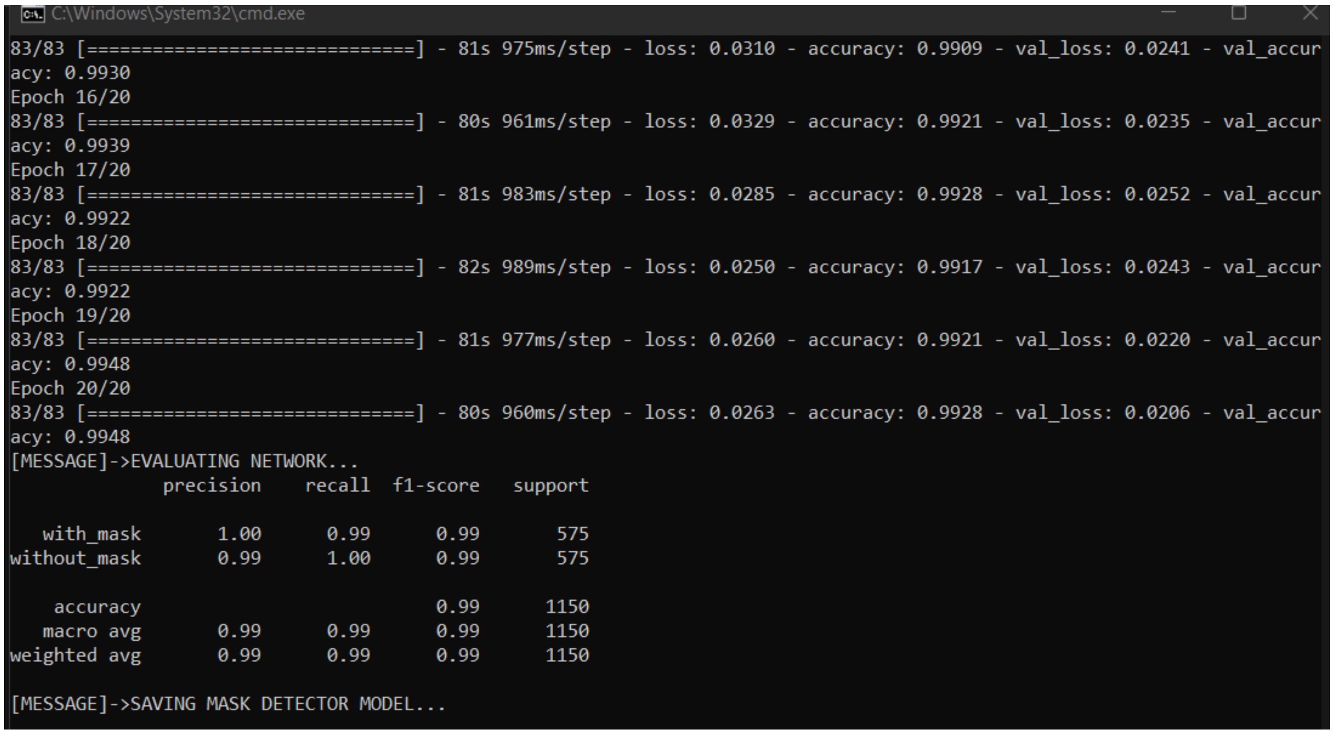
In the first phase of Deep Learning training, it is very important to have a lot of data so that the model can learn all the variations in the images. A common method to increase the number of training sets is to use data accretion. It was also used to generate new images by performing a set of accretion operations on the images which were extracted from video frames.

6. EXTRACTION COORDINATED FROM IMAGES

To locate and find main parts of the face such as mouth, nose, jawline, eyes facial coordinates are used. These coordinates are essential for head pose estimation, blink detection, yawning detection, etc. An open-source C++ library has a pre-written function that can be used to obtain facial coordinates. This library is programmed to find the y & x coordinates of more than 67 facial coordinates to map the facial structure.

7. TRAINING THE MODEL:

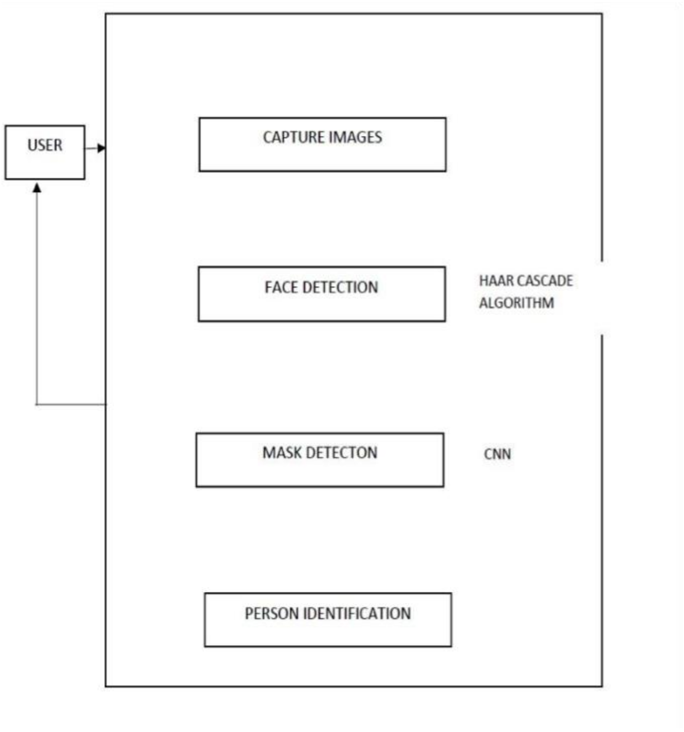
While training we need to made a virtual python environment and name it 'test' and then activate it and then install all the requirements in it and then we can start the training by having an optimal number of epochs for the CNN and the training of the neural network is very necessary as the model learns patters that are specific to sample data to a great extent. The total number of epochs used in this project is 20.



```
83/83 [==============================] - 81s 975ms/step - loss: 0.0310 - accuracy: 0.9909 - val_loss: 0.0241 - val_accur
acy: 0.9930
Epoch 16/20
83/83 [==============================] - 80s 961ms/step - loss: 0.0329 - accuracy: 0.9921 - val_loss: 0.0235 - val_accur
acy: 0.9939
Epoch 17/20
83/83 [==============================] - 81s 983ms/step - loss: 0.0285 - accuracy: 0.9928 - val_loss: 0.0252 - val_accur
acy: 0.9922
Epoch 18/20
83/83 [==============================] - 82s 989ms/step - loss: 0.0250 - accuracy: 0.9917 - val_loss: 0.0243 - val_accur
acy: 0.9922
Epoch 19/20
83/83 [==============================] - 81s 977ms/step - loss: 0.0260 - accuracy: 0.9921 - val_loss: 0.0220 - val_accur
acy: 0.9948
Epoch 20/20
83/83 [==============================] - 80s 960ms/step - loss: 0.0263 - accuracy: 0.9928 - val_loss: 0.0206 - val_accur
acy: 0.9948
[MESSAGE]->EVALUATING NETWORK...
              precision    recall  f1-score   support

   with_mask       1.00      0.99      0.99       575
without_mask       0.99      1.00      0.99       575

    accuracy                           0.99      1150
   macro avg       0.99      0.99      0.99      1150
weighted avg       0.99      0.99      0.99      1150

[MESSAGE]->SAVING MASK DETECTOR MODEL...
```

8. <u>FACE DETECTION MODEL:</u>

With the help of face detector file(prototxt and caffemodel)and deep learning mask detector model we will detect the face and camera operation with OpenCV.We have created an object facenet in which we are using cv2.dnn.readnet method where we are giving the path of these two face detection files. And with the help of load_model we are opening our model created. And with VideoStream we are opening the camera. We are creating a frame(an image) for the video stream and giving it a width size 300.
Now we are defining a function detect_predict_mask where we are passing these parameters(facenet,masknet,frame)which return the location of the dimensions of face and prediction which predict the perentage of person wearing mask or not.At last we gave the label and color to the frame whether the person is wearing mask or not. As cv2 follow BGR color coding if mask(0,255,0)and if not(0,0,255).



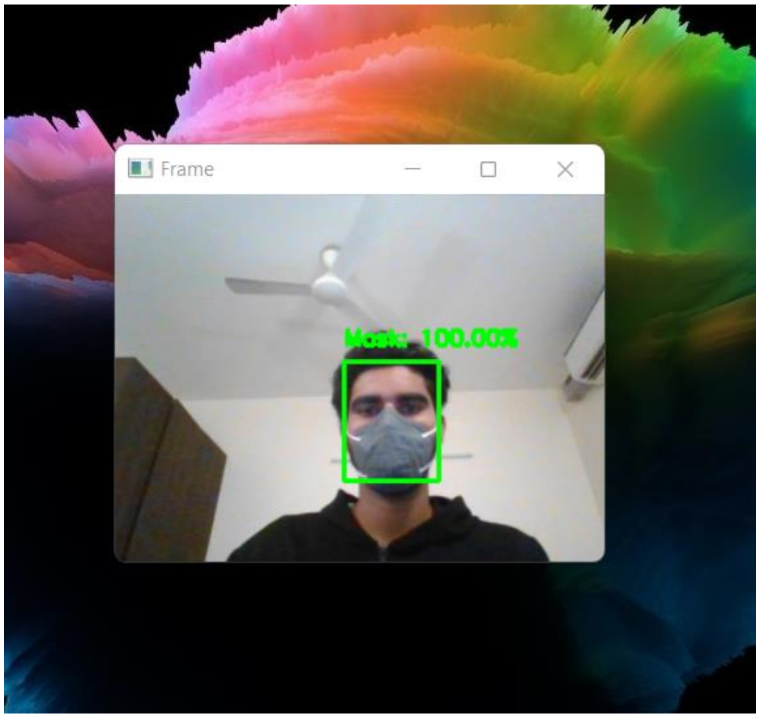
<u>System Architecture</u>

CONCLUSION

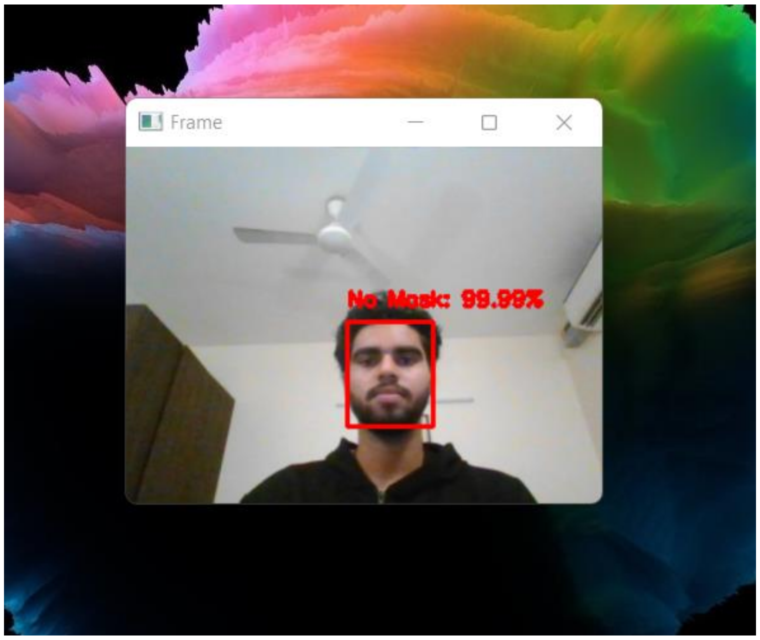
Our proposed system can detect and recognize human face(s) in real-time world. Compared to the traditional face detection and recognition system, the face detection and recognition based on CNN model along with the use of Python libraries has shorter detection and recognition time and stronger robustness, which can reduce the miss rate and error rate. It can still guarantee a high test rate in a sophisticated atmosphere, and the speed of detection can meet the real time requirement, and achieve good effect. The proposed CNN model shows greater accuracy and prediction for detecting and recognising human faces. The results show us that the current technology for face detection and recognition is compromised and can be replaced with this proposed work. Therefore, the proposed method works very well in the applications of biometrics and surveillance.

ACTION SHOT

1. WITH MASK



2. WITHOUT MASK



APPENDIX

Code: https://drive.google.com/drive/folders/1-lhJuRzkef2eBu6Z2rT9f87zWW7yRaco?usp=sharing

REFERENCE

1. https://www.ijert.org/research/real-time-face-mask-detection-and-recognition-using-python-IJERTCONV9IS07014.pdf
2. https://pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/
3. https://www.mdpi.com/2071-1050/13/12/6900/htm
4. https://keras.io/api/preprocessing/image/#imagedatagenerator-class
5. https://www.geeksforgeeks.org/opencv-python-tutorial/