18CSE388T - ARTIFICIAL NEURAL NETWORKS

PROJECT REPORT: MASK DETECTOR USING CNN AND OPENCV

Submitted By:

Jack Praveen Raj Ilango

RA1811026010004

1. Objective Of Project:

Due to the on-going COVID-19 pandemic, masks have become mandatory in many public places as a safety measure to stop the spread of the virus. Thus, there comes a need for a system that can automatically detect pedestrians who are not wearing masks in public places such as streets, shopping malls, government offices etc. This project seeks to solve that problem by providing a reliable method of building a ***Face Mask Detector*** using *Convolutional Neural Networks (CNN) with Python*. With further improvements these types of models could be integrated with CCTV or other types cameras to detect and identify people without masks. With the prevailing worldwide situation due to COVID-19 pandemic, these types of systems would be very useful for many kinds of institutions around the world.

2. Pre-Processing Performed To Raw Data:

For training our face mask detector, we have split the project into two parts. Firstly, we focused on loading our face mask detection dataset from disk, training a model (using Keras/TensorFlow) on this dataset, and then serializing the face mask detector to disk. After that, once the face mask detector is trained, we can then move on to loading the mask detector, performing facial landmark detection. And then classifying each face as with_mask or without_mask.

This dataset consists of 3835 images belonging to two classes:

- with_mask: 1916 images
- without_mask: 1919 images

This is the sample dataset used for the project:
https://drive.google.com/drive/folders/1088NleP3VhRlS2df2eyHaWmarJdBlSxV?usp=sharing

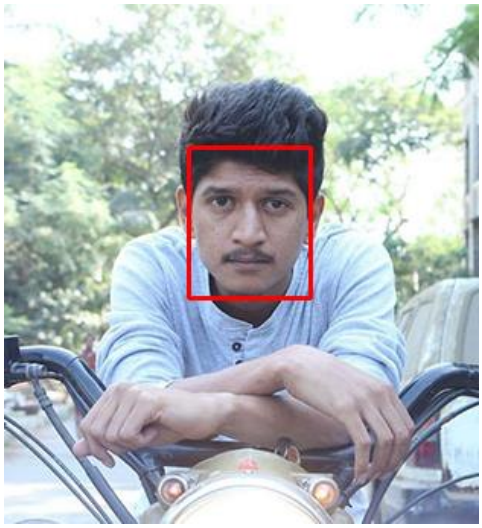
3. Features Extracted:

We used dlib and OpenCV to detect facial landmarks in an image. Facial landmarks are used to localize and represent salient regions of the face, such as:

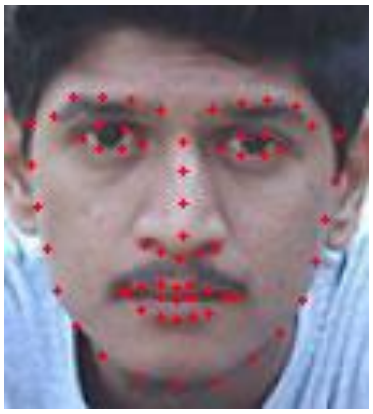
- Eyes
- Eyebrows
- Nose
- Mouth
- Jawline

Facial landmarks have been successfully applied to face alignment, head pose estimation, face swapping, blink detection and much more. Detecting facial landmarks is a subset of the shape prediction problem. Given an input image (and normally an ROI that specifies the object of interest), a shape predictor attempts to localize key points of interest along the shape.

The pre-trained facial landmark detector inside the dlib library is used to estimate the location of 68 (x, y)-coordinates that map to facial structures on the face.



Region of Interest (ROI) detected from original image.



Region of Interest (ROI) separated and facial landmarks detected.

4. Model Used:

This project seeks to solve that problem by providing a reliable method of building a Face Mask Detector using *Convolutional Neural Networks (CNN)* using Python. In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analysing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics.

5. Details Of The Experiment:

The imports for our training script includes Tensorflow, Keras, MobileNet and OpenCV. The face mask detector didn't use any morphed masked images dataset. The model is accurate, and since the MobileNetV2 architecture is used, it's also computationally efficient and thus making it easier to deploy the model to embedded systems.

Our set of tensorflow.keras imports allow for: Data augmentation, Loading the MobilNetV2 classifier, Building a new fully-connected (FC) head, Pre-processing, Loading image data etc.

Now Training of the CNN Model is done through the following steps:

Firstly, we start by grabbing all of the *imagePaths* in the dataset and initializing data and labels lists. Then we loop over the *imagePaths* and we load pre processing images. The pre-processing steps include resizing to 224×224 pixels, conversion to array format, and scaling the pixel intensities in the input image to the range [-1, 1]. Finally, we append the pre-processed image and associated label to the data and labels lists, respectively.

Once training is complete, we'll evaluate the resulting model on the test set. We need to make predictions on the test set and get all the class label indices. Now, we grab the highest probability class label indices and then serializes our face mask classification model to disk.

Now, we can train our face mask detector using Keras, TensorFlow, and Deep Learning through the terminal. The Validation set and accuracies are given below:

```
$ python train_mask_detector.py --dataset dataset
[INFO] loading images...
[INFO] compiling model...
[INFO] training head...
Train for 34 steps, validate on 276 samples
Epoch 1/20
34/34 [==============================] - 30s 885ms/step - loss: 0.6431 - accuracy: 0.6676 - val_loss: 0.3696 - val_accuracy: 0.8242
Epoch 2/20
34/34 [==============================] - 29s 853ms/step - loss: 0.3507 - accuracy: 0.8567 - val_loss: 0.1964 - val_accuracy: 0.9375
Epoch 3/20
34/34 [==============================] - 27s 800ms/step - loss: 0.2792 - accuracy: 0.8820 - val_loss: 0.1383 - val_accuracy: 0.9531
Epoch 4/20
34/34 [==============================] - 28s 814ms/step - loss: 0.2196 - accuracy: 0.9148 - val_loss: 0.1306 - val_accuracy: 0.9492
Epoch 5/20
34/34 [==============================] - 27s 792ms/step - loss: 0.2006 - accuracy: 0.9213 - val_loss: 0.0863 - val_accuracy: 0.9688
...
Epoch 16/20
34/34 [==============================] - 27s 801ms/step - loss: 0.0767 - accuracy: 0.9766 - val_loss: 0.0291 - val_accuracy: 0.9922
Epoch 17/20
34/34 [==============================] - 27s 795ms/step - loss: 0.1042 - accuracy: 0.9616 - val_loss: 0.0243 - val_accuracy: 1.0000
Epoch 18/20
34/34 [==============================] - 27s 796ms/step - loss: 0.0804 - accuracy: 0.9672 - val_loss: 0.0244 - val_accuracy: 0.9961
Epoch 19/20
34/34 [==============================] - 27s 793ms/step - loss: 0.0836 - accuracy: 0.9710 - val_loss: 0.0440 - val_accuracy: 0.9883
Epoch 20/20
34/34 [==============================] - 28s 838ms/step - loss: 0.0717 - accuracy: 0.9710 - val_loss: 0.0270 - val_accuracy: 0.9922
[INFO] evaluating network...
              precision    recall  f1-score   support

   with_mask       0.99      1.00      0.99       138
without_mask       1.00      0.99      0.99       138

    accuracy                           0.99       276
   macro avg       0.99      0.99      0.99       276
weighted avg       0.99      0.99      0.99       276
```

Finally the Output can be found using dlib and OpenCV to detect facial landmarks in the frame and check if the person is wearing a mask or not.

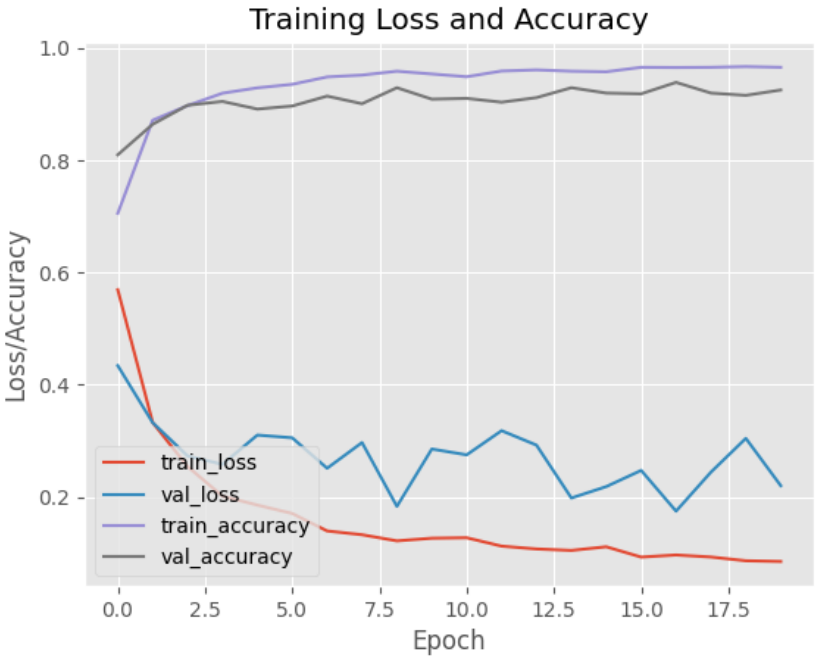
6. Results:

The proposed face mask detector has been successfully trained by using CNN deep learning methods on the sample datasets and the face mask detection process has been successfully performed by the trained detector being tested on the test data set. The only drawback of this model is that face masks can not be detected if the image has been obscured or altered due to weather conditions and motion.

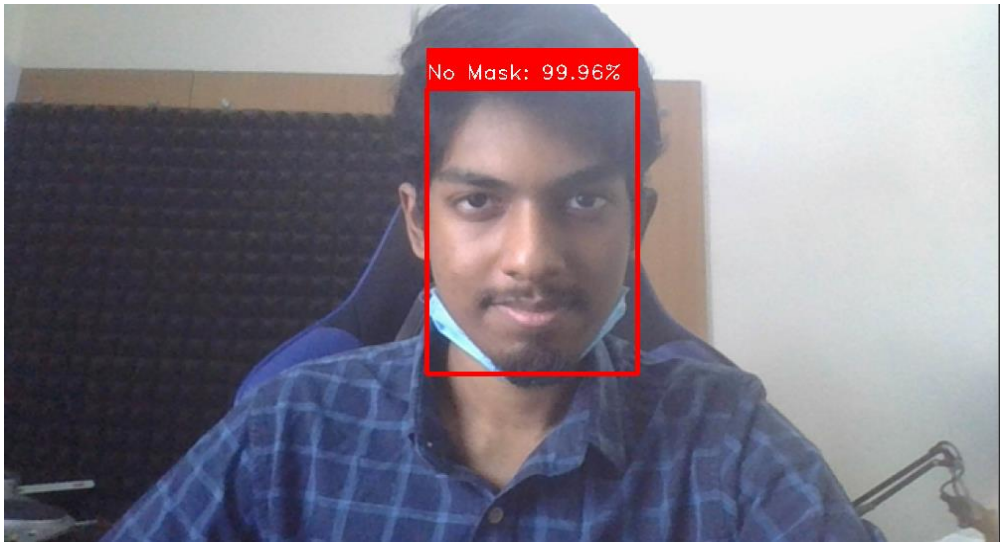
Our model gave 93% accuracy for Face Mask Detection after training via ***tensorflow-gpu==2.0.0:***

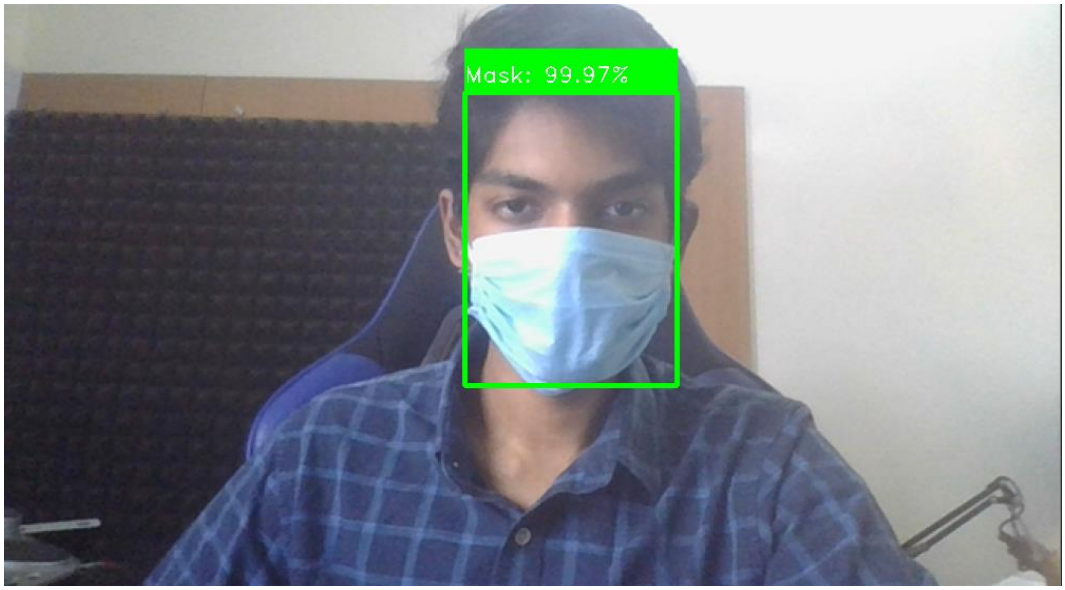
```
[INFO] evaluating network...
              precision    recall  f1-score   support

   with_mask       0.99      0.86      0.92       383
without_mask       0.88      0.99      0.93       384

    accuracy                           0.93       767
   macro avg       0.93      0.93      0.93       767
weighted avg       0.93      0.93      0.93       767

[INFO] saving mask detector model...
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

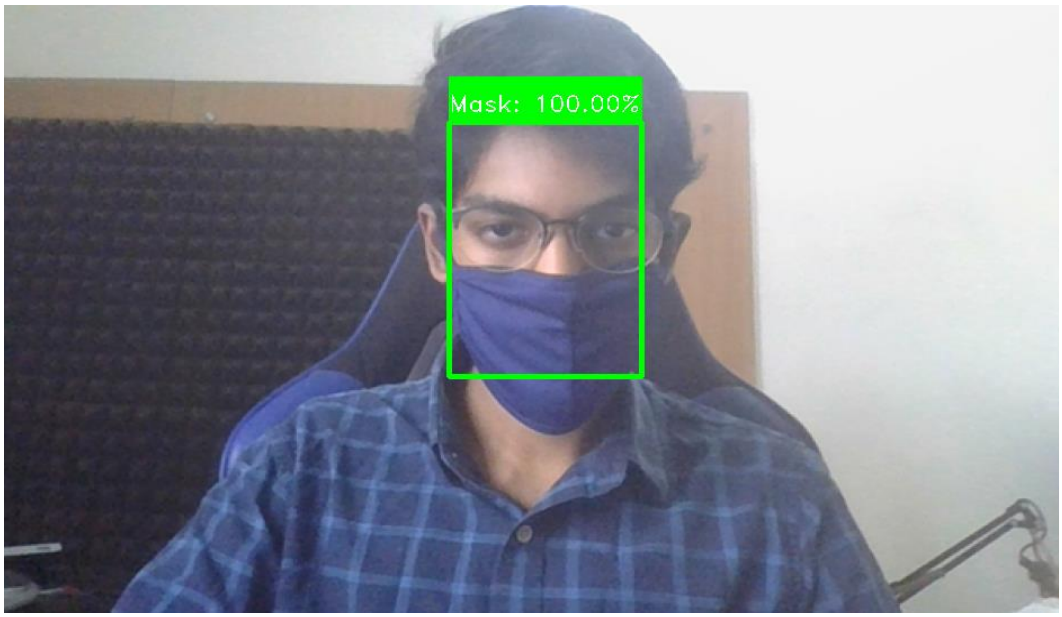
We got the following accuracy/loss training curve plot:



Output:







Submitted By:

1. Jack Praveen Raj Ilango

   RA1811026010004

   3rd Year – Batch 2

2. Karan Sanal

   RA1811026010013

   3rd Year – Batch 1