|  |
| --- |
| PUNJAB ENGINEERING COLLEGE |
| FACE MASK DETECTION |
| SUMMARY OF PROJECT |

|  |
| --- |
| Nancy Pareta  6-11-2020 |

MTECH (COMPUTER SCIENCE ENGINEERING)

PUNJAB ENGINEERING COLLEGE

**FACE MASK DETECTION**

Face Mask Detection dataset:

A face mask detection dataset consists of “with mask” and “without mask” images. I have used the dataset to build a face mask detector with computer vision and deep learning using Python, OpenCV, and TensorFlow/Keras.

The dataset we’ll be using here today was created by PyImageSearch reader [**Prajna Bhandary.**](https://www.linkedin.com/feed/update/urn%3Ali%3Aactivity%3A6655711815361761280/)

This dataset consists of **1,376 images** belonging to two classes:

* with\_mask

: 690 images

* without\_mask

: 686 images

Our goal is to train a custom deep learning model to detect whether a person *is* or *is not* wearing a mask.

**PROJECT CONTENTS:**

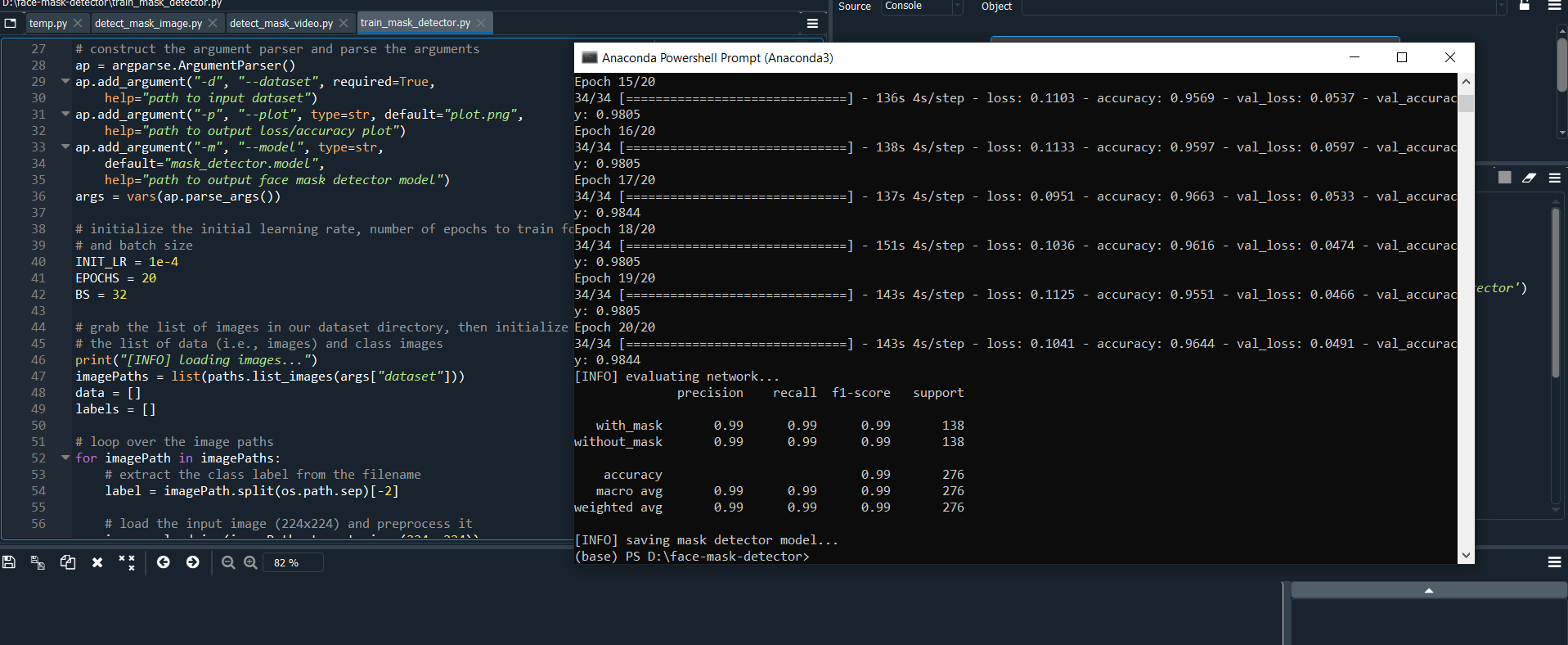
Dataset - directory contains the data of images of with\_mask and without\_mask.

Examples – directory contains demo images and the images given in the assignment.

I’ll be reviewing three Python scripts in this assignment:

* train\_mask\_detector.py: Accepts our input dataset and fine-tunes MobileNetV2 upon it to create our mask\_detector.model
* A training history: plot.png containing accuracy/loss curves is also produced
* detect\_mask\_image.py: Performs face mask detection in static images
* detect\_mask\_video.py: Using your webcam, this script applies face mask detection to every frame in the stream

Firstly, trained the model for the dataset and got following result:



To accomplish this task, we’ll be fine-tuning the [MobileNet V2 architecture](https://arxiv.org/abs/1801.04381)**,** a highly efficient architecture that can be applied to embedded devices with limited computational capacity (ex., Raspberry Pi, Google Coral, NVIDIA Jetson Nano, etc.)

As you can see, we are obtaining **~99% accuracy** on our test set.

**We can see there are little signs of overfitting, with the validation loss *lower* than the training loss because the dataset used is of less images. It can be improved by using large data set and training it for more epochs.**

**Implementing our face mask detector for images with OpenCV**

Now that our face mask detector is trained, we have done the following:

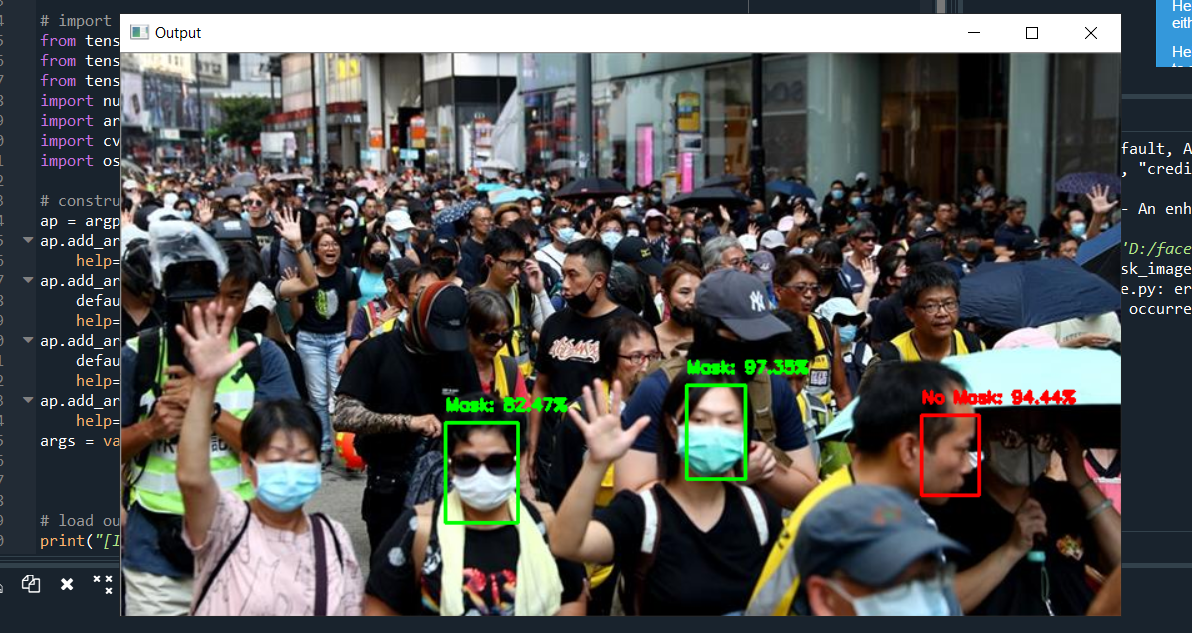
1. Load an input image from disk
2. Detect faces in the image
3. Apply our face mask detector to classify the face as either

with\_mask

 or

without\_mask

**Results:**



**As, we can see my model is recognizing only frontline people, not all the people in the image this is limitation of our model. It still needs improvement to detect all the objects in the picture to determine whether they are wearing mask or not.**

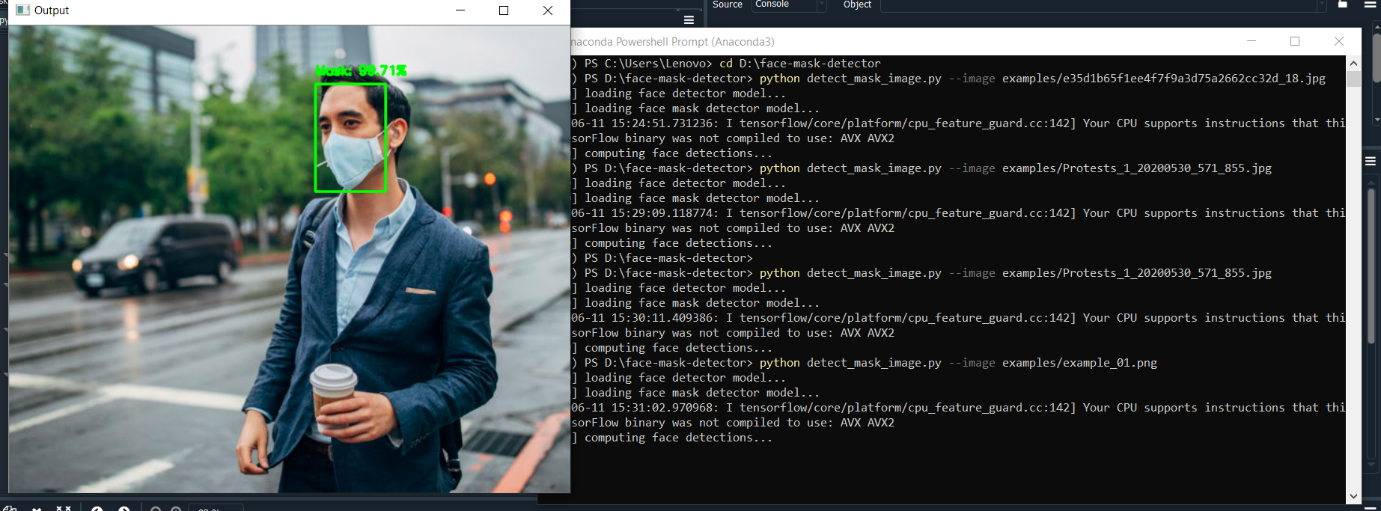
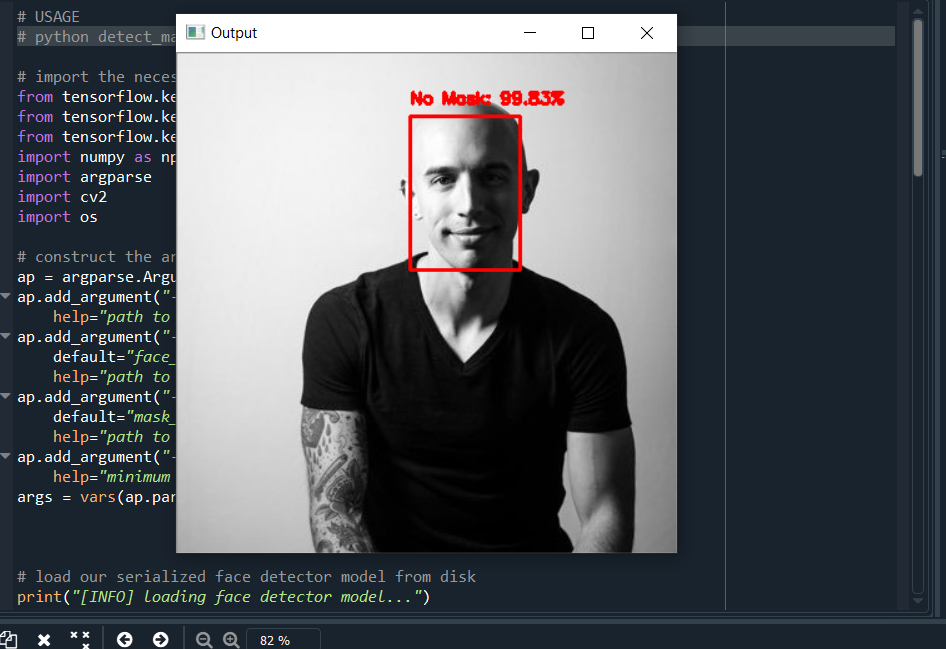
In order to classify whether or not a person is wearing in mask, we first need to perform face detection — **if a face is not found (which is what happened in this image), *then the mask detector cannot be applied!***

The reason we cannot detect the face in the foreground is because:

1. It’s too obscured by the mask
2. The dataset used to train the *face detector* did not contain example images of people wearing face masks

Therefore, if a large portion of the face is occluded, my face detector will likely fail to detect the face.

For normal single object static images our model is working absolutely fine:



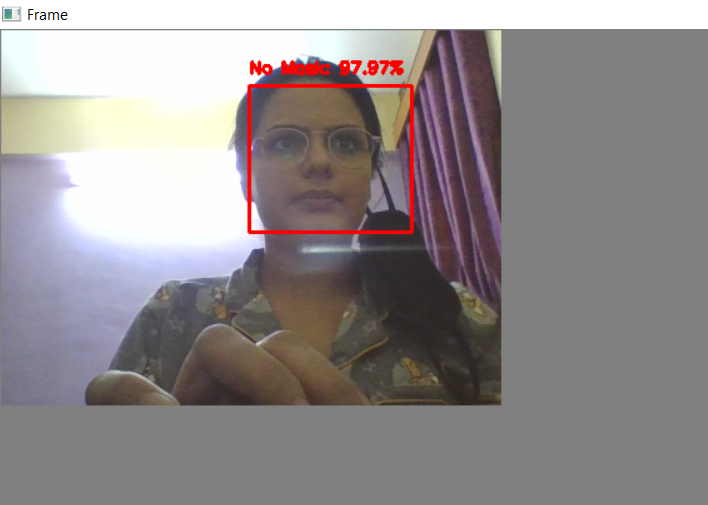
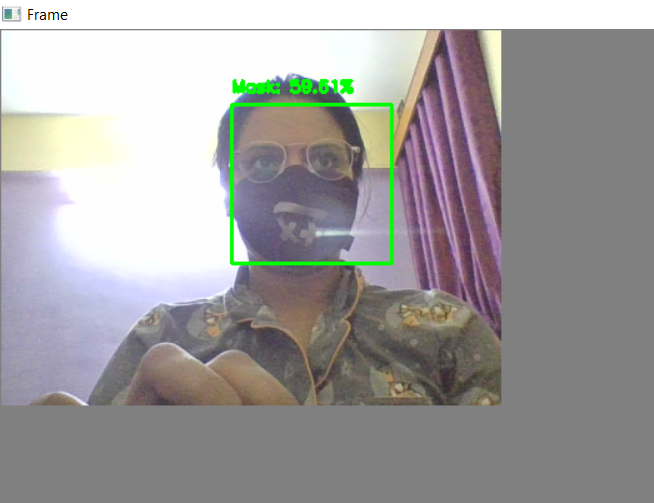
**Implementing our face mask detector for Real time video stream with OpenCV**

In python code, detect\_and\_predict\_mask.py

 Function accepts three parameters:

* frame: A frame from our stream
* faceNet: The model used to detect where in the image faces are
* maskNet: face mask classifier model

**RESULTS:**



As you can see from the results sections above, my face mask detector is working quite well despite:

**Having limited training data**

The with\_mask class being artificially generated. To improve our face mask detection model further, you should gather actual images (rather than artificially generated images) of people wearing masks. While our artificial dataset worked well in this case, there’s no substitute for the real thing.

Secondly, one should also gather images of faces that may “confuse” our classifier into thinking the person is wearing a mask when in fact they are not — potential examples include shirts wrapped around faces, bandana over the mouth, etc. All of these are examples of something that could be confused as a face mask by our face mask detector. Finally, one should consider training a dedicated two-class object detector rather than a simple image classifier.

My current method of detecting whether a person is wearing a mask or not is a two-step process:

Step #1: Perform face detection

Step #2: Apply our face mask detector to each face

The problem with this approach is that a face mask, by definition, obscures part of the face. If enough of the face is obscured, the face cannot be detected, and therefore, the face mask detector will not be applied. To circumvent that issue, you should train a two-class object detector that consists of a with\_mask class and without\_mask class. Combining an object detector with a dedicated with\_mask class will allow improvement of the model in two respects.

First, the object detector will be able to naturally detect people wearing masks that otherwise would have been impossible for the face detector to detect due to too much of the face being obscured.

Secondly, this approach reduces our computer vision pipeline to a single step — rather than applying face detection and then our face mask detector model, all we need to do is apply the object detector to give us bounding boxes for people both with\_mask and without\_mask in a single forward pass of the network. Not only is such a method more computationally efficient, it’s also more “elegant” and end-to-end.