FaceMask and Social Distancing Detection

2021

Introduction

Background:

Given the current crisis the world is facing, facemask and social distancing (minimum of 6ft apart) has become part of the norm in most places. However, many people are still not obeying the guidelines presented by the government and this has caused the spread of the virus to increase at a near uncontrollable rate. Therefore, there is a need to create a detection system that could be implemented to a simple device like a smart mirror placed strategically at the entrance of popular venues. This could help remind the people or deter people from forgetting or ignoring the current health safety regulations and guidelines.

Goal / Objective

- To create a simple facemask detection system using machine learning models such as:
    - ConvNet
    - InceptionV3
    - MobileNet
    - DenseNet
    - VGG19
- Next, add in a social distancing rule using euclidean method on the image to ensure a minimum of 6 feet rule is followed
- Create a GUI that can track images live
- Pick a suitable model to be implemented in the smart mirror in my house for a test-run.

Data set

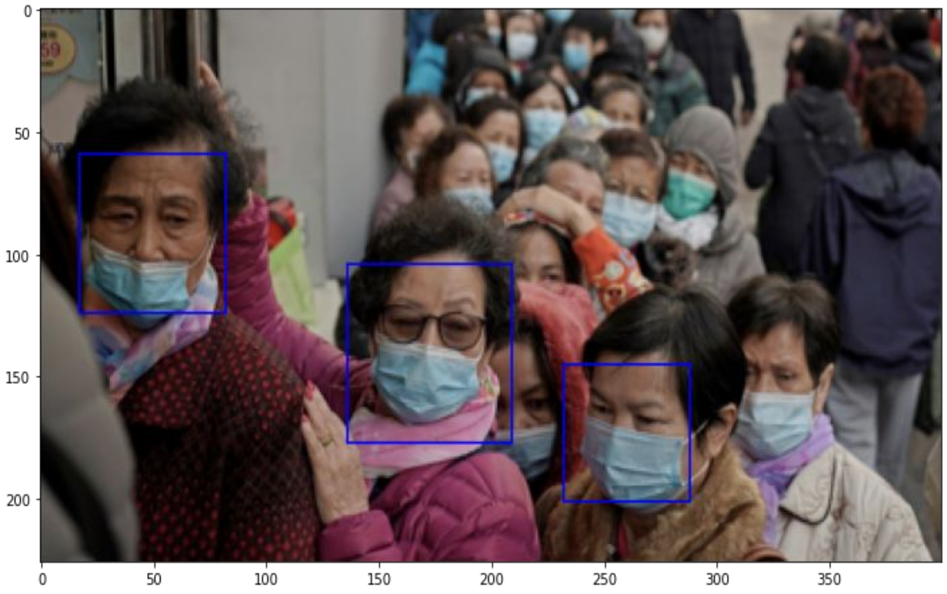
The image data set used consists of photos scraped from Unsplash by author Engin Akyurt, containing images split into categories such as with and without mask.

- Dataset directories:
    - Train
        - With Mask
        - Without Mask
    - Test
        - With Mask
        - Without Mask

Haarcascades: This OpenCV resource is able to detect face, profile, eyes, smile, and upper body. However, for our purposes, I have only used the face detection system which could be found on:

https://github.com/opencv/opencv/tree/master/data/haarcascades

Face Detector



This is a sample image showing how many faces the haarcascade data is able to detect. The blue box was created by resizing image using tuples.

- ScaleFactor = 1.1
- minNeighbors = 4

Although the faces detected are only the first few (front), we can manipulate the images by augmenting to improve the face detector.

Social Distancing Detector

- The red box indicates that these people have violated the social distance minimum requirement.
- The minimum social distance was set by using the euclidean method. The distance set was 150 for this image.
- For reference, if the euclidean distance was set to 120 the lady on the left will have a green box around her face indicating that the minimum distance has been exceeded.

Machine Learning Models Comparison

ConvNet (CNN)

```
loss: 0.5626 - accuracy: 0.9047 - val_loss: 0.1653 - val_accuracy: 0.9375

loss: 0.2312 - accuracy: 0.9493 - val_loss: 0.4176 - val_accuracy: 0.8875

loss: 0.2430 - accuracy: 0.9676 - val_loss: 0.1446 - val_accuracy: 0.9688

loss: 0.2117 - accuracy: 0.9638 - val_loss: 0.5848 - val_accuracy: 0.9531

loss: 0.2506 - accuracy: 0.9637 - val_loss: 0.4514 - val_accuracy: 0.9625

loss: 0.1663 - accuracy: 0.9742 - val_loss: 0.8821 - val_accuracy: 0.9062

loss: 0.0944 - accuracy: 0.9811 - val_loss: 2.2687 - val_accuracy: 0.7563

loss: 0.1266 - accuracy: 0.9797 - val_loss: 0.2329 - val_accuracy: 0.9719

loss: 0.0996 - accuracy: 0.9782 - val_loss: 0.2773 - val_accuracy: 0.9656

loss: 0.0902 - accuracy: 0.9835 - val_loss: 0.1397 - val_accuracy: 0.9812
```

Model layers:

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_94 (Conv2D)           (None, 126, 126, 32)      896
_________________________________________________________________
batch_normalization_94 (Batc (None, 126, 126, 32)      128
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 63, 63, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 63, 63, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 127008)            0
_________________________________________________________________
dense (Dense)                (None, 2)                 254018
=================================================================
```

Consist of neural networks connected by neurons that works in its own receptive field to process data in a grid-like topology

InceptionV3

```
loss: 0.1622 - accuracy: 0.9484 - val_loss: 0.0446 - val_accuracy: 0.9844
loss: 0.0567 - accuracy: 0.9863 - val_loss: 0.0550 - val_accuracy: 0.9906
loss: 0.0654 - accuracy: 0.9857 - val_loss: 0.0186 - val_accuracy: 0.9937
loss: 0.0440 - accuracy: 0.9921 - val_loss: 0.1058 - val_accuracy: 0.9906
loss: 0.0550 - accuracy: 0.9904 - val_loss: 0.0276 - val_accuracy: 0.9906
loss: 0.0192 - accuracy: 0.9961 - val_loss: 0.0212 - val_accuracy: 0.9937
loss: 0.0980 - accuracy: 0.9878 - val_loss: 0.0562 - val_accuracy: 0.9937
loss: 0.0259 - accuracy: 0.9922 - val_loss: 0.0951 - val_accuracy: 0.9781
loss: 0.0532 - accuracy: 0.9907 - val_loss: 0.0083 - val_accuracy: 0.9969
loss: 0.0513 - accuracy: 0.9929 - val_loss: 0.0709 - val_accuracy: 0.9937
```

Model layers:

Layer (type)	Output Shape	Param #
inception_v3 (Functional)	(None, 2, 2, 2048)	21802784
flatten_1 (Flatten)	(None, 8192)	0
dense_1 (Dense)	(None, 2)	16386

A CNN network architecture that uses label smoothing, factorized 7x7 convolutions, and auxiliary classifier to transfer label information along the network

MobileNet

```
loss: 0.0950 - accuracy: 0.9689 - val_loss: 0.0368 - val_accuracy: 0.9937

loss: 0.0314 - accuracy: 0.9943 - val_loss: 0.0357 - val_accuracy: 0.9969

loss: 0.0130 - accuracy: 0.9989 - val_loss: 7.6572e-04 - val_accuracy: 1.0

loss: 0.0321 - accuracy: 0.9968 - val_loss: 0.0732 - val_accuracy: 0.9969

loss: 0.0124 - accuracy: 0.9991 - val_loss: 0.0604 - val_accuracy: 0.9875

loss: 0.0275 - accuracy: 0.9962 - val_loss: 0.0939 - val_accuracy: 0.9969

loss: 0.0113 - accuracy: 0.9979 - val_loss: 0.0328 - val_accuracy: 0.9969

loss: 0.0181 - accuracy: 0.9986 - val_loss: 0.0753 - val_accuracy: 0.9969

loss: 0.0246 - accuracy: 0.9988 - val_loss: 0.0556 - val_accuracy: 0.9969

loss: 0.0036 - accuracy: 0.9992 - val_loss: 0.0555 - val_accuracy: 0.9969
```

Model layers:

```
Layer (type)                 Output Shape              Param #
=================================================================
mobilenet_1.00_128 (Function (None, 4, 4, 1024)        3228864
_________________________________________________________________
flatten_2 (Flatten)          (None, 16384)             0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 32770
=================================================================
```

Designed for mobile and embedded vision applications by using depthwise separable convolutions to build deep neural network while having low-latency

DenseNet

```
loss: 0.1503 - accuracy: 0.9576 - val_loss: 0.1068 - val_accuracy: 0.9844
loss: 0.0362 - accuracy: 0.9937 - val_loss: 0.0137 - val_accuracy: 0.9969
loss: 0.0242 - accuracy: 0.9960 - val_loss: 0.1098 - val_accuracy: 0.9875
loss: 0.0130 - accuracy: 0.9982 - val_loss: 0.1197 - val_accuracy: 0.9937
loss: 0.0842 - accuracy: 0.9915 - val_loss: 0.2181 - val_accuracy: 0.9844
loss: 0.0708 - accuracy: 0.9941 - val_loss: 0.1656 - val_accuracy: 0.9844
loss: 0.0365 - accuracy: 0.9974 - val_loss: 0.0740 - val_accuracy: 0.9969
loss: 0.0178 - accuracy: 0.9984 - val_loss: 0.0708 - val_accuracy: 0.9969
loss: 0.0132 - accuracy: 0.9992 - val_loss: 0.0473 - val_accuracy: 0.9969
loss: 0.0249 - accuracy: 0.9985 - val_loss: 0.0807 - val_accuracy: 0.9969
```

Model layers:

Layer (type)	Output Shape	Param #
densenet201 (Functional)	(None, 4, 4, 1920)	18321984
flatten_3 (Flatten)	(None, 30720)	0
dense_3 (Dense)	(None, 2)	61442

As the name suggest it uses dense blocks, where dense connections between layers are made to connect all layers directly with each other

VGG19

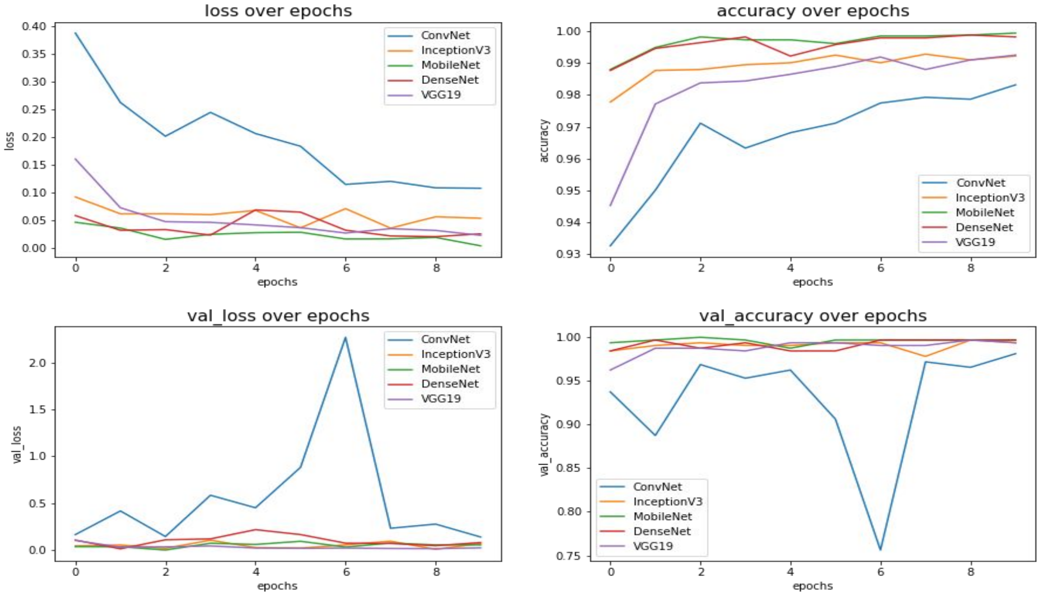
```
loss: 0.2778 - accuracy: 0.8863 - val_loss: 0.1028 - val_accuracy: 0.9625
loss: 0.0855 - accuracy: 0.9730 - val_loss: 0.0315 - val_accuracy: 0.9875
loss: 0.0429 - accuracy: 0.9869 - val_loss: 0.0348 - val_accuracy: 0.9875
loss: 0.0464 - accuracy: 0.9845 - val_loss: 0.0443 - val_accuracy: 0.9844
loss: 0.0439 - accuracy: 0.9878 - val_loss: 0.0221 - val_accuracy: 0.9937
loss: 0.0385 - accuracy: 0.9873 - val_loss: 0.0191 - val_accuracy: 0.9937
loss: 0.0291 - accuracy: 0.9900 - val_loss: 0.0216 - val_accuracy: 0.9906
loss: 0.0315 - accuracy: 0.9891 - val_loss: 0.0176 - val_accuracy: 0.9906
loss: 0.0237 - accuracy: 0.9929 - val_loss: 0.0157 - val_accuracy: 0.9969
loss: 0.0181 - accuracy: 0.9948 - val_loss: 0.0242 - val_accuracy: 0.9937
```

Model layers:

Layer (type)	Output Shape	Param #
vgg19 (Functional)	(None, 4, 4, 512)	20024384
flatten_4 (Flatten)	(None, 8192)	0
dense_4 (Dense)	(None, 2)	16386

Consists of 19 layers: 16 concolution layers, 3 fully connected layers, 5 maxpooling layers, and 1 softmax layer

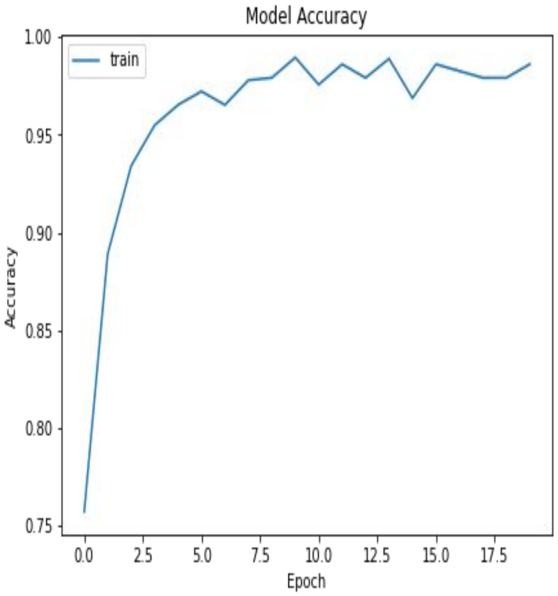
Model Comparison

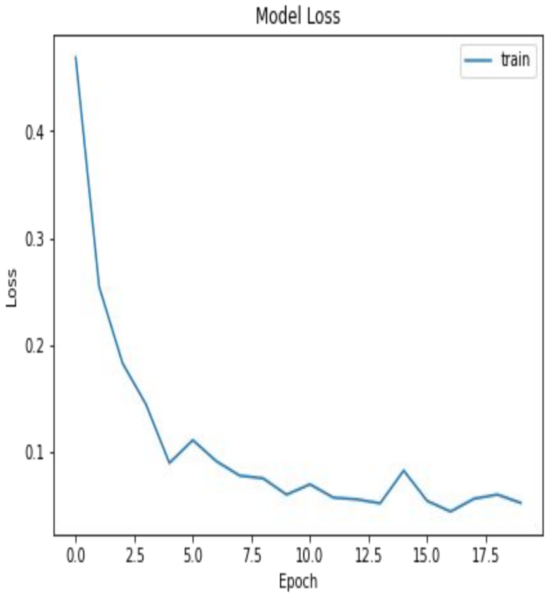


Observation:

- ConvNet seemed to be very underfitting and behaved poorly in learning the dataset
- In terms of losses, all other models seemed to be relatively close
- Looking at the accuracy, from best to worst, MobileNet and DenseNet tops the chart
- InceptionV3 and VGG19 ended with accuracy that is relatively similar towards the end of the epochs
- VGG 19 was chosen as the model due to its steady learning rate unlike the other models that can be considered either overfitting too much or underfitting

Chosen Model - VGG19



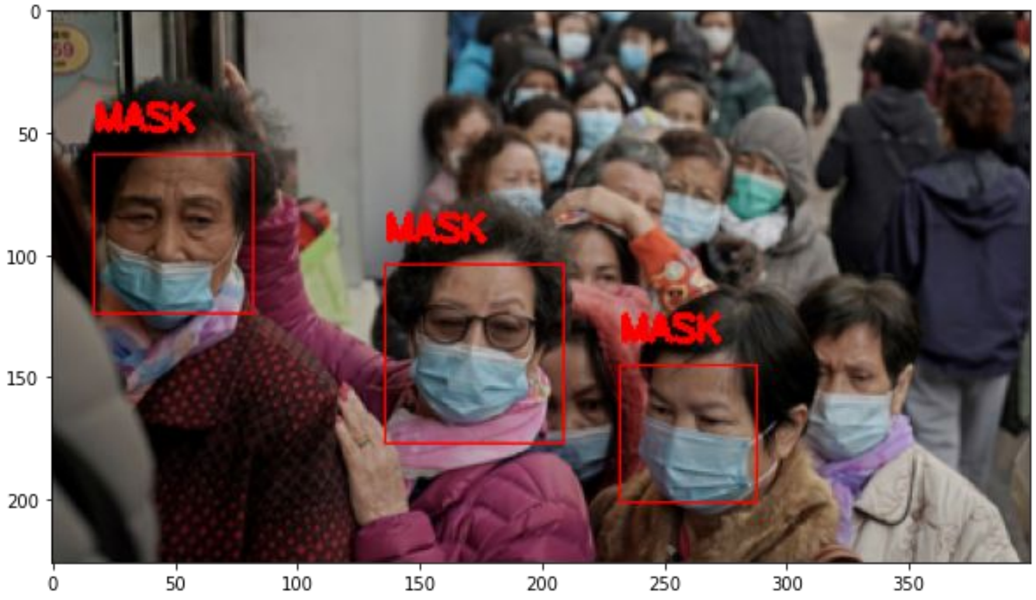


Model Evaluation:

- Loss = 0.0587
- Accuracy = 0.9758

The model seemed to be stabilizing at an accuracy of above 95% and a loss of below 0.1% after the 8th epoch with a minor hiccup on epoch 13 that could have been caused by an abnormal image.

Results



Identified Faces:

- All wore mask as indicated by the detector
- All violates the social distance of minimum 6 feet apart as indicated by the red boxes around the faces

Unidentified faces

- Image not recognized by the Haarcascade face recognition system

Improvements / Suggestions

- Improve the face detector model creating our own model for face detection using more neural network layers. This will allows us to detect smaller or blurrer faces in the images.
- Integrate model into a live system feed that will show mask and social distance violation at the moment. This could be used in many places especially entrances to commercial, industrial, and even walkways. This concept is similar in idea to a speed trap whereby speed detectors with clear visual of your current speed is portrayed to remind drivers not to exceed the speed limit.
- Start by implementing into my house's smart mirror which will be connected to a front door camera. This will allow me to safeguard myself from people not wearing mask.