COVID-19 FACE MASK DETECTION WITH DEEP LEARNING AND COMPUTER VISION

Saksham Agarwal / Aravind Veluri / Harshvardhan Kumar / Anvit Patil
Visual Recognition (Imtech CSE)
IIITB
Banglore , Karnataka

***Abstract*—COVID-19 pandemic is causing a global health crisis. According to the World Health Organization (WHO), one of the most effective protection methods is to wear a face mask in public areas. This is especially crucial in mass public areas like shopping malls, etc. An economic and societal approach of using AI is to create a safe environment for everyone. Hence, in this report we build an effective, real-time automatic door control system for allowing a human wearing a face mask, detected from a live feed through a webcam. We use both deep and classical machine learning approaches to help determine if the person in front of the door is wearing a mask, using a 3 stage pipeline of human detection using YOLO, face detection using Viola Jones, mask detection using a deep convolutional neural network. In addition, we use this system to identify one person at a time to implement effective social distancing.**

***Index Terms*—Deep Learning, Computer Vision, OpenCV, Tensorflow, Keras**

I. Introduction

Before the pandemic, only a few people wore masks and that too if they were concerned with the quality of air. However, ever since the COVID-19 pandemic, masks have become a requisite if one has to go outdoors. A number of research studies [1] have shown that face mask is quite effective in stopping the transmission of the SARS-CoV-2 virus.
Over 122 million people and 2 and half a million deaths were caused by the COVID-19 in a year, thus making it one of the biggest pandemics in the history of mankind.
In light of such statistics, there was a significant rise of emerging technologies in healthcare field like Artificial Intelligence, Internet of Things etc. Artificial Intelligence, in particular, has a number of applications in tackling the pandemic. It can be used evaluate large amount of data and then use to forecast distribution of COVID-19. This can act as an early warning and could be used to classify vulnerable populations in other parts of the world.
As a preventive measure, many countries made it mandatory for the public to wear face masks if they are outdoors. Keeping this in mind we introduce a face mask detection model based on computer vision and deep learning. The following system is to be integrated with automated door systems which open up only if the person is wearing a mask. The model utilises classical machine learning and deep learning algorithms.

II. Approach

A. *Reference model*

We had a previously implemented model which used Viola Jones face detector with a convolutional neural network that detects masks. Even though this model performed well on the given dataset, it didn't exactly perform well in our real-time testing, with the reasons being listed below:

- If the mask colour was slightly dark, then the mask was not being detected.

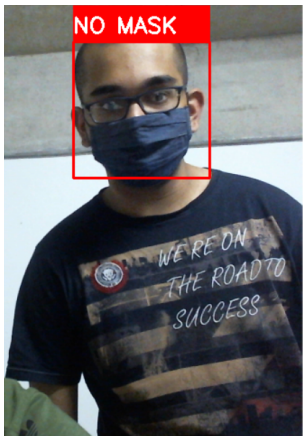

Fig. 1. Dark coloured masks ignored

Fig. 2. Light coloured masks being detected

- Dark skinned people's masks were not being detected even if the mask was light coloured.

After looking at the data-set, we felt that the bugs can be solved by a more extensive data-set. Other than this, the model did not implement a human detector, which is also a requisite in our given system.

B. *Dataset*

In dataset [1] that we used, there were images of white surgical masks photoshopped onto other people. This dataset was not suitable for detecting masks of other colors. Dataset [2] was of size 23Gb, which would be too much for the model to train. Dataset [3] had 12k images which was perfect for our model as it was not too large and it had images of people with a variety of masks.

C. *Pipeline*

We propose a three stage pipeline solution for effectively building the door entry system.

1) ***Human Detection****:* Our first stage is human detection. There can be false positives if human detection is avoided, for example, someone showing a picture of a face with a mask on their phone to the webcam. We use the YOLO algorithm for human detection.

YOLO(You Only Look Once) is the faster, real-time solution for object detection. It is faster than RCNN in orders of magnitude. Instead of using parts of the image like RCNN and Faster RCNN does, YOLO performs bounding box object detections directly on the image, and also gives confidence values for those bounding boxes.

YOLO distributes the picture into $S \times S$ grid cells. Each cell is responsible for predicting 5 bounding boxes, supporting maximum of 5 different object predictions, which result in too many boxes, so instead YOLO predicts probability of classes, to help remove bounding boxes which have low probability, resulting in lesser, filtered bounding boxes. YOLO is implemented in Darknet. YOLOv3 adds few layers of improvements by increasing layers to overcome the problem of detecting smaller objects, implementing in a better Darknet(Darknet-53), using logistic regression over softmax for multilabel classification.

One limitation of YOLO though, is that it doesn't perform as well as RCNN and Faster-RCNN on small objects in the image. This wouldn't be a hindrance for us since the person won't be standing very far away from the camera.

In this report, we use 2 YOLO models for comparison, YOLOv3 and YOLOv3-tiny, which was proposed to improve the running time of YOLOv3. YOLOv3 architecture has 106 layers, where as YOLOv3-tiny has only 24, as shown in figure 3. Comparison can be seen in the results section.

2) ***Face Detection****:* Face detection is performed by Viola Jones method. Viola Jones detects frontal face better than faces looking sideways, upwards or downwards. Image is converted into gray scale and then detects it before finding the location on the colored image. Viola-Jones detects face within the box

Layer	Type	Filters	Size/Stride	Input	Output
0	Convolutional	16	3 × 3/1	416 × 416 × 3	416 × 416 × 16
1	Maxpool		2 × 2/2	416 × 416 × 16	208 × 208 × 16
2	Convolutional	32	3 × 3/1	208 × 208 × 16	208 × 208 × 32
3	Maxpool		2 × 2/2	208 × 208 × 32	104 × 104 × 32
4	Convolutional	64	3 × 3/1	104 × 104 × 32	104 × 104 × 64
5	Maxpool		2 × 2/2	104 × 104 × 64	52 × 52 × 64
6	Convolutional	128	3 × 3/1	52 × 52 × 64	52 × 52 × 128
7	Maxpool		2 × 2/2	52 × 52 × 128	26 × 26 × 128
8	Convolutional	256	3 × 3/1	26 × 26 × 128	26 × 26 × 256
9	Maxpool		2 × 2/2	26 × 26 × 256	13 × 13 × 256
10	Convolutional	512	3 × 3/1	13 × 13 × 256	13 × 13 × 512
11	Maxpool		2 × 2/1	13 × 13 × 512	13 × 13 × 512
12	Convolutional	1024	3 × 3/1	13 × 13 × 512	13 × 13 × 1024
13	Convolutional	256	1 × 1/1	13 × 13 × 1024	13 × 13 × 256
14	Convolutional	512	3 × 3/1	13 × 13 × 256	13 × 13 × 512
15	Convolutional	255	1 × 1/1	13 × 13 × 512	13 × 13 × 255
16	YOLO				
17	**Route 13**				
18	Convolutional	128	1 × 1/1	13 × 13 × 256	13 × 13 × 128
19	Up-sampling		2 × 2/1	13 × 13 × 128	26 × 26 × 128
20	**Route 19 8**				
21	Convolutional	256	3 × 3/1	13 × 13 × 384	13 × 13 × 256
22	Convolutional	255	1 × 1/1	13 × 13 × 256	13 × 13 × 256
23	YOLO				

Fig. 3. YOLOv3-tiny architecture

by using haar-like features. A number of boxes detect these haar-like features and then all the boxes put together helps determining where the face is. One drawback of this method is that it is not rotation invariant.

Haar-like features

- Edge features
- Line-features
- Four-sided features

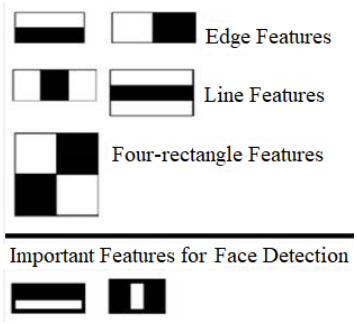

Fig. 4. Haar cascades in Viola Jones

The horizontal and the vertical features describe what eyebrows and the nose look like to the machine.

3) ***Mask Detection****:* Taking further the face detection, we built a mask detection model. We used a deep convolutional neural network, implemented in Keras. We load the preprocessed dataset into the model, then follow the architecture shown in Fig. 5.

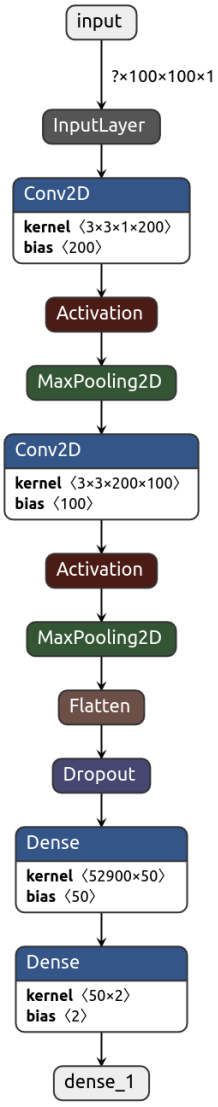

Fig. 5. Mask detector architecture

*4) **System building**:* We convert the above model into a system where when a person enters into the frame of a camera we give them entry based on whether they have a mask. We first linked our webcam for image extraction and then send that image into the human detection pre-trained model(YOLO), discussed above, and get the bounding boxes of people in the image. We send the cropped, resized image of a single person, who has the maximum probability among all the people detected, to the face classifier which returns the detected face if any on the person.

Note: Taking only one person at a time ensures social distancing between people while entering since, if there are multiple people trying to enter, the YOLO model would just get confused and keep choosing one of them randomly. One of the minor improvements would be to add a logical system where if YOLO detects multiple people, the door would just not open.

Extracted face region would be sent to the mask detection model(described above) to predict if the person is wearing a mask. To ensure no false positives, we add a consistent detection mechanism, where the door would only open if the person is identified to have been continuously wearing mask for 3-5 seconds, without a no-mask reading in between that lasts for more than 1 second.

Once the system detects that there is a consistent reading, it greets the person with a full green bordered screen along with a screenshot of the scene, virtually indicating that the door is now open.

III. Results

A. *YOLOv3 vs YOLOv3-tiny*

The size of YOLOv3 is <50MB which is suitable to be put into sensors, etc, for easy compatibility with the automatic doors.
Along with this we decided to go with the YOLOv3-tiny version since it provided us with more FPS as compared to the original one, and we didn't lose that much accuracy since the person is not that far from the camera.

Type of FPS	YOLOv3-tiny	YOLOv3-608
Average	**12**	**1**
Max	**22**	**5**

B. *CNN performance*

As you can see from the graphs below we have our rate of decay of our loss function and the subsequent increase in the accuracy of our model. In general, our final accuracy turned out to be around **98%** which we thought of as accurate enough for our mask detection model.

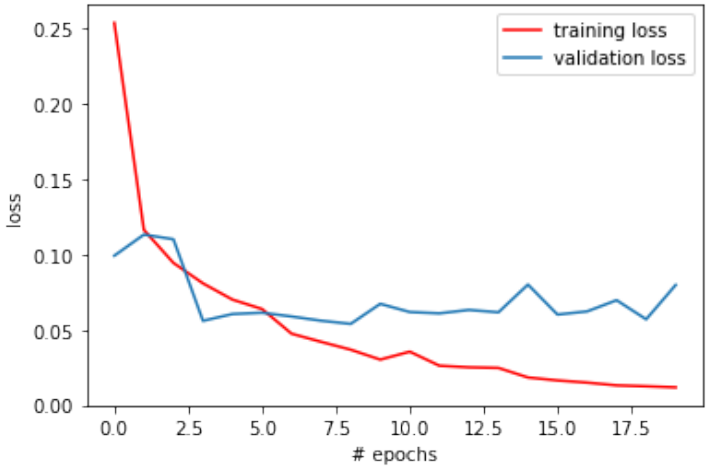

Fig. 6. Loss function of our model

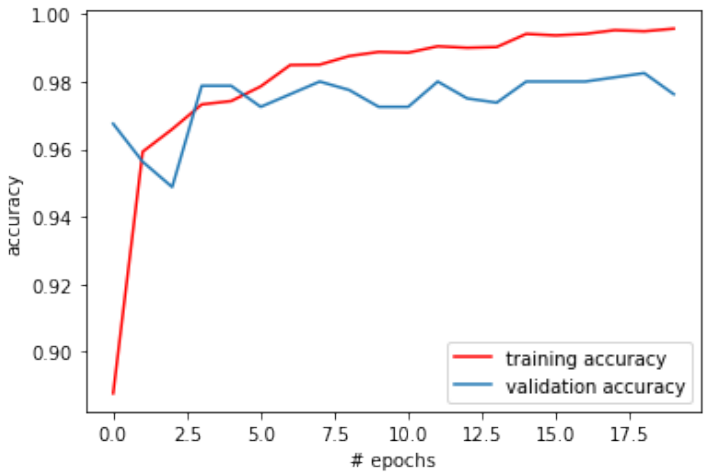

Fig. 7. Accuracy of our model

IV. Improvements:

1) As of now our **Face Detection** is rotation variant, but in later stages we can try to make it rotation invariant by transforming the image's highest gradient vector to the horizontal direction via Histogram of Gradients(HOG).
2) There are some lighting and shadow issues due to which a person is not detected in the Face Detection algorithm, can use Histogram Equalization or some other methods to improve upon this.
3) Our system is still a bit lax in terms of detecting hands on face as a mask , we can use some data set to negatively train it against that.

Acknowledgments

We would like to thank Prof. Dinesh Babu Jayagopi and all the Teaching assistants of this course for providing us with the resources and help needed to complete this Mini Project. We thoroughly enjoyed this project and got to learn a lot from it as well.

References

[1] https://drive.google.com/drive/folders/1WCxe1EuxLo6qyGVpupcEMTgN83xpgHM_?usp=sharing
[2] https://www.kaggle.com/prasoonkottarathil/face-mask-lite-dataset
[3] https://www.kaggle.com/ashishjangra27/face-mask-12k-images-dataset
[4] https://www.thelancet.com/journals/landig/article/PIIS2589-7500(21)00003-0/fulltext
[5] https://www.kaggle.com/ashishjangra27/face-mask-12k-images-dataset/activity
[6] https://www.researchgate.net/figure/You-Only-Look-One-v3-tiny-YOLOv3-tiny-network-structure_tbl1_335043703
[7] https://ieeexplore.ieee.org/document/9074315
[8] https://appsilon.com/object-detection-yolo-algorithm/
[9] https://pjreddie.com/darknet/yolo/
[10] https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b
[11] https://netron.app/
[12] https://www.pyimagesearch.com/2017/11/06/deep-learning-opencvs-blobfromimage-works/
[13] https://iopscience.iop.org/article/10.1088/1757-899X/732/1/012038