VR Mini Project

Pranav Kumar
IMT2018020
IIIT Banglore
Pranav.Kumar@iiitb.org

Udith Sai
IMT2018081
IIIT Banglore
Udith.Sai@iiitb.org

Shashank Reddy
IMT2018069
IIIT Banglore
Shashank.Reddy@iiitb.org

Pavan Sudeesh
IMT2018517
IIIT Banglore
Pavan.Sudeesh@iiitb.org

I. AIM

To build a vision based automatic door system where the door will open for the humans only when they were a mask.

II. Introduction

To build our AIM we will divide project that into 3 sub tasks.
1.) Human Detection.
2.) Face Detection.
3.) Mask Detection.
So, first if the door need to open first the entering object has to be a human then we will detect the face in that object and at last we will check for the mask.

III. Human Detection.

We will classify the objects entering into classes Human and non human(classes all other than humans) for our project. For detection we used YOLO algorithm.
YOLO (You Only Look Once) is a method or a way to do object detection. It is the algorithm /strategy behind how the code is going to detect objects in the image.

Why YOLO over other algos?
Earlier frameworks , looked at different parts multiple times hence took long time for Detection and they are slow and inefficent YOLO takes a differnt approach it looks at the image only once(unlike others) It is very fast hence we use this.
So total we have 4 things
i) input image – the one we want to classify.
ii) YOLO config file – configuration file for YOLO
iii) Pre-trained YOLO weights: As YOLO is a DNN we have weights for each network and stored them.
iv) Text file Containing Class Names: As i mentioned we are dividing the objects into classes one Human and Non-human the objects detected by YOLO will be stored in a text file with its class names so non- human objects include all classes other than humans.

Identify applicable funding agency here. If none, delete this.

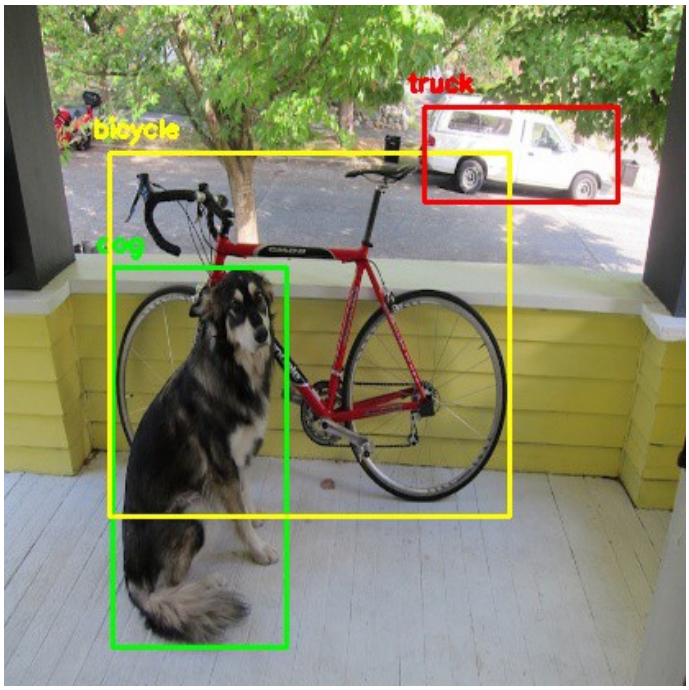

Figure 1. This is how the YOLO detects objects and name it to classes.

IV. Face Detection.

We already classified the entering objects into classes Human and other not Human(classes other than humans) now we will detect the face of the human. We detected face using VIOLA JONES. Let the input frame in the below diagram be the list of the humans which has classified by the Human Detector and the remaining process of mask detection goes as shown in the diagram.

A. Viola Jones Algorithm.

Given an image(this algorithm works on grayscale image), the algorithm looks at many smaller subregions and tries to find a face by looking for specific features in each subregion. It needs to check many different positions and scales because an image can contain many faces of various sizes. Viola and Jones used Haar-like features to detect faces in this algorithm. The Viola Jones algorithm has four main steps, which we shall discuss in the sections to follow:

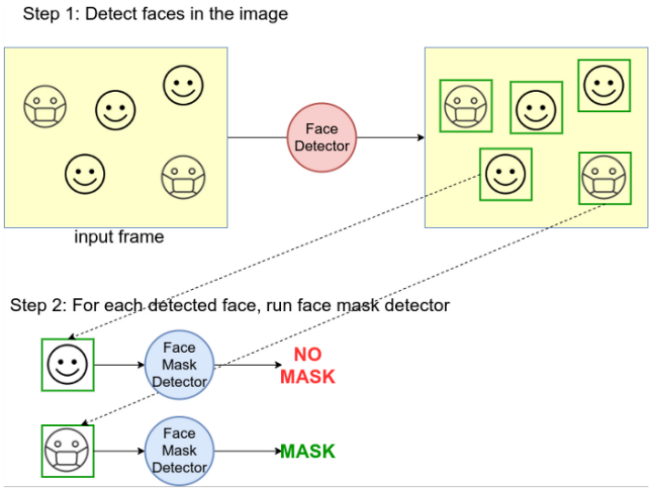

Figure 2. Remaining Work Flow

1) Selecting Haar-like features: Common properties of every humans like the eyes region is darker than its neighbour pixels, and the nose region is brighter than the eye region. A simple way to find out which region is lighter or darker is to sum up the pixel values of both regions and compare them. The sum of pixel values in the darker region will be smaller than the sum of pixels in the lighter region. It has again 3 sub features namely 3 sub features Edge features, line features and four sided features.

2) Creating an integral image: In the previous section, we have seen that to calculate a value for each feature, we need to perform computations on all the pixels inside that particular feature. In reality, these calculations can be very intensive since the number of pixels would be much greater when we are dealing with a large feature. An integral image is an quick way to calculate the sum of pixel values in an image or rectangular part of an image.

3) Running AdaBoost training: As the image we have can have many lakhs of features we need AdaBoost to identify the best features. To find the best feature Adaboost checks the performance of the classifier. To calculate the performance of a classifier, you evaluate it on all subregions of all the images used for training. Some subregions will produce a strong response in the classifier. Those will be classified as positives, meaning the classifier thinks it contains a human face. The classifiers that performed well are given higher importance or weight. The final result is a strong classifier.

4) Creating classifier cascades: let the Adaboost have gave us some thousands of features but still detecting face in them is a time taking factor so we use cascading its main use is to quickly discard non-faces and avoid wasting time. We set up a cascaded system in which we divide the process of identifying a face into multiple stages. In the first stage, we have a classifier which is made up of our best features, in other words, in the first stage, the subregion passes through the best features such as the feature which identifies the nose bridge or the one that identifies the eyes. In the next stages, we have all the remaining features.

We passed a image an image from internet for testing and the results are pretty accurate.

Figure 3. Input image.

Figure 4. This is how it rounds the face.

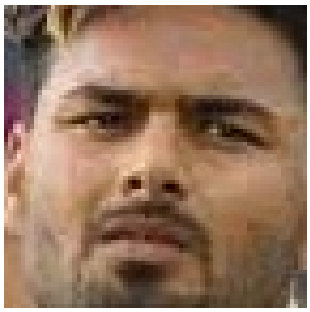

Figure 5. just the face image will be cropped and will look as above

V. Mask Detection

We are going to build this Mask Detection in two parts. In the first part, we will write a python script using Keras to train face mask detector model. In the second part, we test the results in a real-time webcam using OpenCV.
1.) We will train the model with Neural Networks in the first

part.
2.) In second part we will use the trained model and take a live feed and the model trained will detect whether the object is wearing mask or not.

VI. Conclusion

The results are pretty accurate and and time taken to detect was very low in individual detection processes Human, face and mask respectively. The overall accuracy was 91 percentage on the test data.

Figure 6. Guy here is wearing mask so it is showing mask

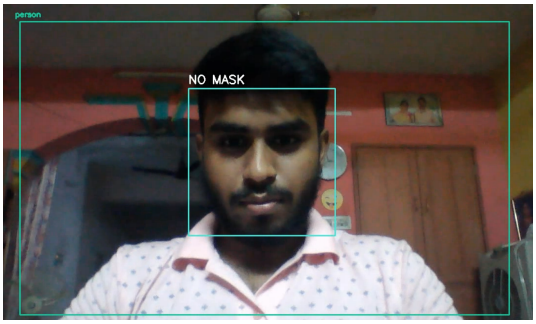

Figure 7. Guy here is not wearing mask so it is showing no mask

Acknowledgment

We would like to thank our professor Dinesh sir and Ta's Tejas Kotha and Swathi s mishra for giving us this interesting project, helping us through out our project and for solving our doubts.

References

https://towardsdatascience.com/yolo-object-detection-with-opencv-and-python-21e50ac599e9
https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html
https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/
https://www.mygreatlearning.com/blog/viola-jones-algorithm/
https://data-flair.training/blogs/face-mask-detection-with-python/