Face mask detection with track of social distancing

Prof Archana Chaudhari, Pranjali Sagar, Vivek Rugale, Nagesh Reure, Prashant Rathod, Abhijit Pokharkar

Department of Instrumentation and Control Engineering,
Vishwakarma Institute of Technology, Pune

Abstract— Today our entire world is suffering from covid-19 situation in which it is very essential to cover the face with mask and keep social distancing to control the spread of corona virus thus this paper provides a system which gives solution for face mask recognition and tracking of social distancing using platform of python opencv. Here we are building the model for face mask recognition using Mobil net and used tensorflow library for mask detection and used deep neural network model for face detection, and to get the distance between two peoples Euclidian distance formula is implemented

Keywords— tensorflow, covid-19, Mobil net, YOLO, object tracking, social distancing etc.

I. Introduction

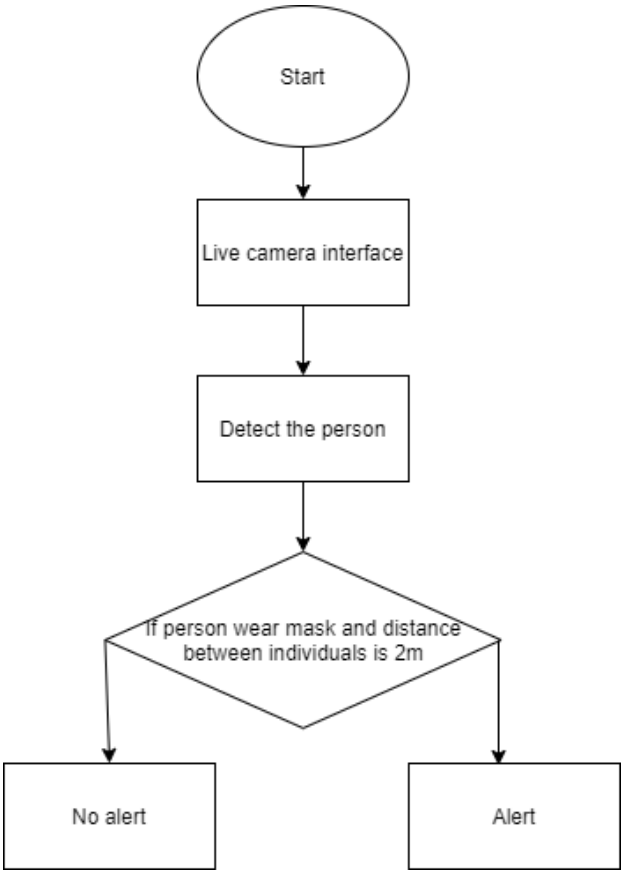
Now WHO recommends the people to wear face masks for avoiding the risk of corona virus transmission and also recommends that a social distance of at least 2m be maintained between individuals to prevent person-to-person spread of disease. Furthermore, many public service providers require customers to use the service only if they wear masks and follow safe social distancing. Therefore, face mask detection and safe social distance monitoring has become an important computer vision task to help the global society. This paper describes approach to prevent the spread of the virus by monitoring in real time if person is following safe social distancing and wearing face masks in public places.

This method is used on real-time video surveillance to monitor public places to detect if persons wearing face mask and maintaining safe social distancing. Our solution uses neural networking models to analyze Real-Time Streaming Protocol video streams using Opencv and MobileNet V2 and basically we are using pertained model for social distancing tracking

II. Methodology

The proposed system helps to ensure the safety of the people at public places by automatically monitoring them whether they maintain a safe social distance, and also by detecting whether or not and individual wears face mask. This section briefly describes the solution architecture and how the proposed system will automatically function in an automatic manner to prevent the coronavirus spread.

A. Flowchart



B. Algorithm for Face mask detection

In the proposed system, four steps are followed, such as:

1) Data collection and pre-processing

2) Model development and training

3) Model testing

4) Model implementation

1) Data collection and pre-processing

The proposed system uses a custom data set consisting of face images with different types of face masks which are labeled and used for the training of our models. The dataset used to train our proposed face mask detector consists of 1889 images. Before the custom face mask image dataset is labelled, the data set is divided into the training data set and the testing data set. The Training data set should consist of 80% images to train the algorithm effectively and for prediction accuracy and the Testing data set should consist of 20% images to test the prediction accuracy of the algorithm. The images in the training data collection are classified into two categories: mask and no mask

2) Model building and Training

Here we are using MobileNetV2 model, which is a highly efficient architecture... The custom data set is loaded into the project directory and the algorithm is trained on the basis of the labeled images. In pre-processing steps, the image is resized to 224×224 pixels, converted to numpy array format and the corresponding labels are

added to the images in the dataset before using our SSD model as input to build our custom model with MobileNetV2 as the backbone and train our model using the Tensorflow Object Detection API. Before model training begins, tensorflow helps in Data augmentation and download pre-trained ImageNet weights to make the algorithm's prediction efficiency accurate. After downloading the pre-trained weights and creating a new fully-connected head, the algorithm is trained with both the pre-trained ImageNet weights and the annotated images in the custom data set by tuning the head layer weights without updating weights of base layers. We trained our model for 1000 steps using the Adam optimizing algorithm, the learning decay rate for updating network weights, and the binary cross-entropy for mask type classification. Parameters were initialized for the initial learning rate of INIT_LR = 1e-4, number of epoch EPOCHS = 10 and batch size BS = 32.

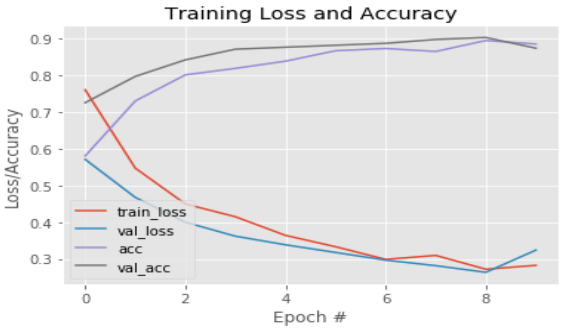


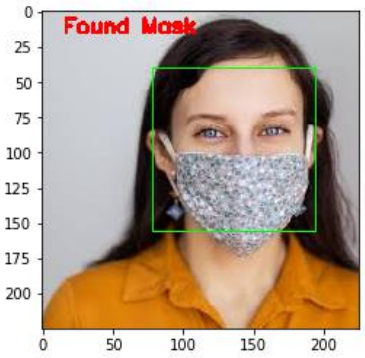
Fig. Model training accuracy/loss curves

3) Model Testing

. Once the model is trained with the custom data set and the pre-trained weights given, we check the accuracy of the model on the test dataset by showing the bounding box with the name of the tag and the confidence score at the top of the box.

```
Faces found:  1
[92.491394   7.5086026]
Found Mask

<matplotlib.image.AxesImage at 0x2987c383188>
```

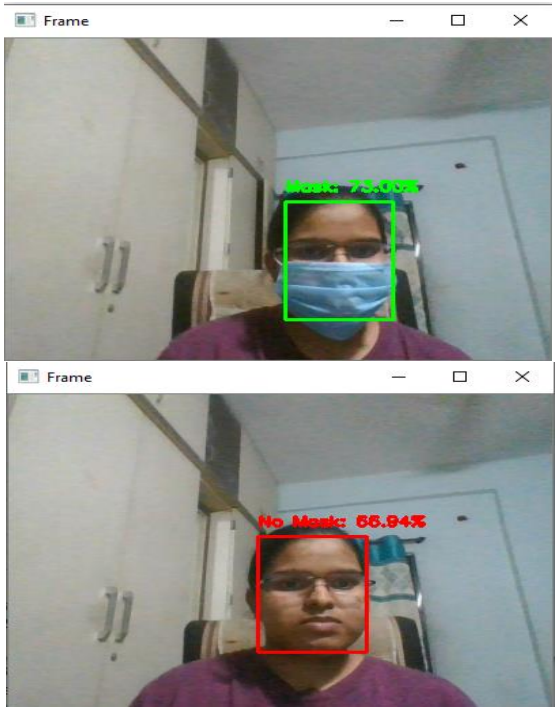


Test result of model

The system detects the social distancing and masks with a precision score of 87.30% with loss 0.2816

4) Model Implementation

The camera takes real-time video to the model and can continuously and automatically monitors public places and checks whether people wear a mask or not



C. Algorithm for tracking social distancing

Tracking of social distancing is based on three algorithms –

1) Object detection

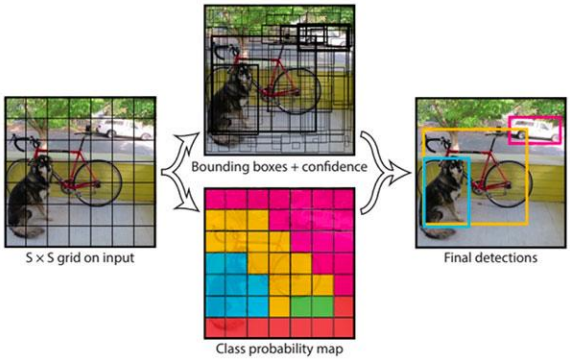
2) Object tracking

3) Distance measurement

All three steps are discussed below in detail –

1. Object detection

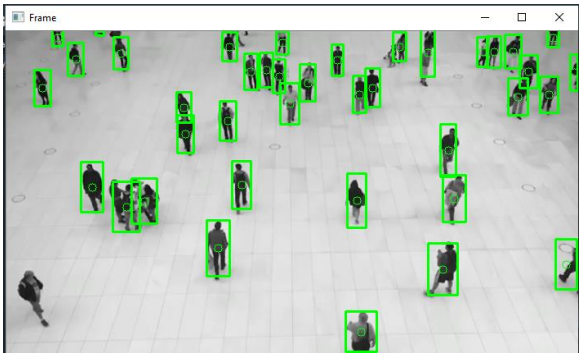
In this project, YOLO algorithm is used for object detection. YOLO (You Only Look Once) is a very fast and different object detection algorithm from others. In this algorithm, entire image is processed in a single instance with single CNN.



As shown in above figure, YOLO divides the input image into grids. Classification and localization is applied on each grid. YOLO then predicts the corresponding object probability and bounding boxes for each grid. In this project, only the person class is to be detected out of other 9000 classes YOLO can detect. So for each grid the output will be a six dimensional vector representing the bounding boxes and probability.

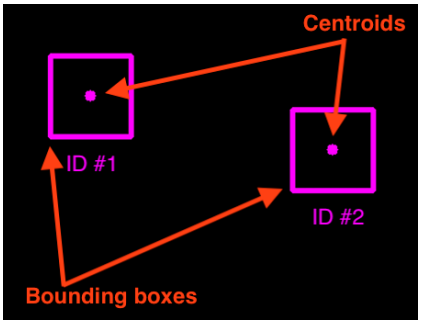
pc
bx
by
bh
bw
c1

Output of YOLO consists of the object probability (pc), bounding box dimensions (bx, by, bh, bw) and c1 is the class. So if the person class is detected then c1 = 1 otherwise 0. In this project, YOLOv3 object detector model is used which is pre-trained on the COCO dataset. COCO dataset consists of 80 classes out of which only person class is used. Following image shows the output of YOLO algorithm on a video frame-

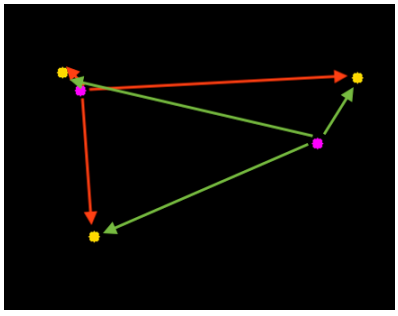


2. Object tracking

Object tracking consists of considering the initial object detections (bounding boxes), assigning a unique ID for each detection and tracking them as they move in a frame while assigning new IDs for new detections. Each detection has a centroid.



For tracking, Euclidian distance is calculated between the detected centroids. If the same object moves in frame then in the next-frame its distance from the original position will be less. So these closed pair of centroids will be assigned with the same ID in updated frame and the other detections with larger distances will have new ID.



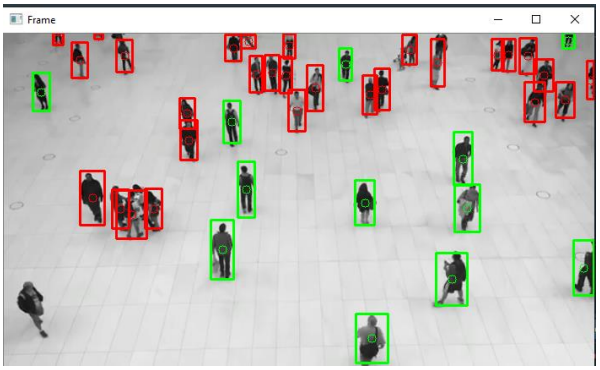
In the above image, the closed pair of yellow and purple centroids will have same ID.

3. Distance measurement

For the purpose of distance measurement, the focal length of the camera used for taking video is to be calculated –

$$F = (P \times D) / W$$

Where F is focal length, P is pixels the object covering, D is the distance of object from camera and W is the width of the object. Using this focal length, we can calculate the distance of moved objects also. The output we are getting from YOLO consists of the bounding box for a person and centroid of the boxes. Therefore, to keep the track of social distance between persons we simply calculate the Euclidean distance between the two centroids. We assign a threshold value for the minimum distance between the two persons. If any person violets this threshold and does not maintain social distance then, that person is assigned with a red colored bounding box otherwise the color of the bounding box is green. Following image shows the final result of the algorithm-



If social distancing is violated, then we can see the red bounding boxes in the output. Therefore, using object detection, object tracking and distance measurement a successful social distance tracking system is created.

Acknowledgment *(Heading 5)*

We would like to thank our guide Prof. Archana Chaudhari for her help and guidance throughout the project.

Conclusion

In this paper, we proposed an approach that uses computer vision and MobileNet V2 architecture to help maintain a secure environment and ensure individuals protection by automatically monitoring public places to avoid the spread of the COVID-19 virus and assist police by minimizing their physical surveillance work in containment zones and public areas where surveillance is required by means of camera. We have given appropriate solution for the tracking of social distancing and the identification of face masks that would help to ensure human health.

References

[1] S. Wang Chen, Horby Peter W, Hayden Frederick G, Gao George F. A novel coronavirus epidemic of global concern for health. It's the Lancet. 2020;395(10223):470-473. 10.1016 / S0140-6736(20)30185-9.

[2] Matrajt L, Leung T. Evaluating the efficacy of social distancing strategies to postpone or flatten the curve of coronavirus disease. Emerg Infect Dis, man. 2020:

[3] Chen, S., Zhang, C., Dong, M., Le, J., R., M., 2017b. Using rankingcnn for age estimates, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

[4] Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen, "Mobilenetv2: Inverted residues and linear bottlenecks," IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510-4520. 2018.

[5] C.Fu, W.Liu, A.Ranga, A. Tyagi, A. Berg, "DSSD: deconvolutional single shot detector model," arXiv preprint arXiv:1701.06659, (2017)

[6] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi: “You Only Look Once: Unified, Real-Time Object Detection” University of Washington, Allen Institute for AI, Facebook AI Research

[7] Omkar Masurekar, Omkar Jadhav, Prateek Kulkarni, Shubham Patil, “Real Time Object Detection Using YOLOv3” Student, Department of Computer Engineering, TEC, University of Mumbai, Mumbai, India