Social Distancing Violation Detection Using YOLOv3

Kiran Sai Gunisetty
Computer Science *1148912*
Lakehead University
Thunder Bay, Canada
kguniset@lakeheadu.ca

Venkata Naga Akshita Atmuri
Computer Science *1152600*
Lakehead University
Thunder Bay, Canada
vatmuri@lakeheadu.ca

Manusree Gurijala
Computer Science *1152980*
Lakehead University
Thunder Bay, Canada
mgurijal@lakeheadu.ca

***Abstract*—One of the most significant and effective strategies for containing the recent viral pandemic is maintaining social distance (SD). To cope with this limitation, governments are establishing rules on the minimum interpersonal distance between people. Given this condition, it's vital to track our adherence to physical constraints in our daily lives to determine what's causing probable distance boundaries to break and whether this constitutes a risk. There are many object detection techniques available such as Faster RCNNs, SSD, YOLO & so on.**

I. Introduction

With its lethal spread, the continuing COVID-19 coronavirus pandemic has produced a worldwide tragedy. Population vulnerability grows as a result of a lack of efficient restorative medications and ongoing different variants of this virus like alpha variant, beta variant, delta variant and many more [1]. Even after getting vaccinated, it is a better measure to maintain social distancing. This method reduces contact between person to person. Although a person is fully vaccinated, where required by federal, state, local, territorial laws, rules, and regulations, including local business and workplace guidance, each person should wear masks and still follow the guidelines of social distancing [2].

The virus is mostly spread by persons who are in close proximity to one another (within 6 feet) over an extended length of time. When an infected individual sneezes, coughs, or speaks, the virus spreads through the air as droplets from their nose or mouth scatter and infect adjacent persons. The droplets also travel via the respiratory system to the lungs, where they begin to damage lung cells. According to recent research, those who have no symptoms but are infected with the virus have a role in the infection's transmission. As a result, it is essential to keep a distance of at least 6 feet from others, even though you do not have any symptoms [3].

Fig. 1. Social distancing

The reason to choose a paper and a project on social distancing is because it is an important protocol to be followed by every single one of us due to the pandemic. Many researchers have already developed models to detect social distance violations using SOTA, CNN, R-CNN & so on.

But there are lot of challenges such as low light conditions or camera inclination angle or camera view angle. So to overcome these challenges, we need updated and more accurate models. We as a group have taken this as a challenge to develop a model, test its accuracy and overcome at least one of these challenges.

II. Related Works

The paper [3] uses the YOLOv3 object recognition paradigm. This is used to identify humans in video sequences. This experiment uses a pre-trained algorithm using a human dataset. The detection model uses bounding box information to identify persons. To determine the bounding box distance, Euclidean distance is used. In video sequences, a tracking algorithm is also used to detect if an individual is violating the minimum distance. The results show that the framework successfully identifies persons who walk too close together and cross social distances; moreover, the transfer learning technique improves the model's overall efficiency. The model's tracking accuracy is 95%.

In reference to the paper [4], they look at how photos from social media may be utilised to predict crowd numbers at city events. They accomplish this by compiling a social media dataset, comparing the efficacy of face recognition and object detection, and estimating crowd size using cascaded methodologies. Their findings demonstrate that object recognition-based approaches are the most accurate in predicting crowd size in city events using social media pictures. Face recognition and object recognition algorithms are also shown to be more suited for estimating crowd size for social media pictures taken in parallel view, with selfies covering people's entire faces and individuals in the background being at the same distance from the camera. Cascaded techniques, on the other hand, are better for photos shot from above.

The authors of the paper [6] suggested a framework that uses the YOLOv3 model to detect individuals and the Deepsort technique was used to track them using bounding boxes and issued IDs. They used a frontal view data set from an open image data set (OID) source. The authors also compared the results with SSD and faster-RCNN. The authors of [7] created an autonomous drone-based social distance monitoring model. They used the custom data set to train the YOLOv3 model. The data set consists of frontal and side views of a small number of persons. The work is also being extended to include facial mask monitoring. YOLOv3 algorithm and the drone camera assist in determining social distance and monitoring people if they are wearing masks or not.

III. Proposed Method

YOLO, a Convolutional Neural Network (CNN) is a real-time object identification system. CNNs are classifier-based systems that can interpret incoming images as organized arrays of data and recognise patterns[9]. YOLO has the benefit of being significantly faster than other networks while yet maintaining accuracy. It allows the model to examine the entire image at test time, allowing it to make predictions based on the image's overall context. YOLO and other convolutional neural network algorithms assign a score to regions based on their resemblance to predefined categories.

The YOLOv3 algorithm divides an image into a grid first. Each grid cell predicts the placement of a certain number of boundary boxes (or anchor boxes) around items that score well in the predefined classes[9]. Each boundary box has a confidence score that indicates how accurate it believes the prediction should be, and each bounding box identifies only one object. To determine the most common forms and sizes, the boundary boxes are created by clustering the dimensions of the ground truth boxes from the original dataset. There are significant differences between YOLOv3 and previous versions in terms of class speed, precision, and specificity as seen in Fig 2 [9].

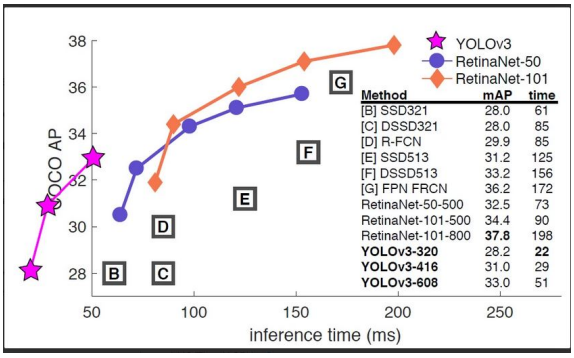

Fig. 2. YOLOv3 comparison chart

IV. Experimental Setup

We collected a couple of pedestrian footage from kaggle and shutterstock which have three different types of camera view angles - Top, Parallel and Between top and parallel. That can be seen in Fig 3. The approach was tested on 3 videos. These videos have different camera viewpoints. The pre-trained YOLOv3 for detecting people in the frames and output of one of the frames can be seen in Fig 4. The main reason behind choosing YOLOv3 for our work is that it is used for human detection as it improves predictive accuracy, particularly for small-scale objects. The main advantage is that it has adjusted network structure for multi-scale object detection. Furthermore, for object classification, it uses various independent logistic rather than softmax [3].

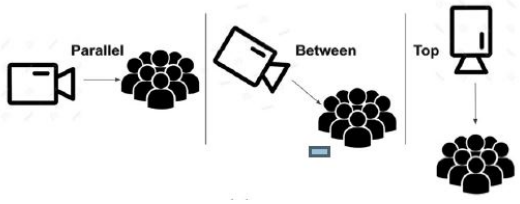

Fig. 3. Different Camera View Angles

We used three yolo weights (Pre-trained speed optimised weight file) to train the model- 320, 416 and 608 COCO datasets [10]. When the model is trained with different weights, there will be a trade off between speed and accuracy. Fig 5 shows the architecture of our method. The first step is frame collection from the video. Once we get the frame, we perform basic pre-processing where we resize the frame to (416,416). After the frame is processed, the blob of the image is collected and passed as an input to our model. Blob means Binary Large Object and refers to a group of collected pixels [5]. Once the bounding boxes are detected, we apply a technique called Non-max suppression. It is a technique used in numerous computer vision tasks and has a collection of algorithms to select one bounding box out of many overlapping bounding boxes. NMS is used by most object detection algorithms to reduce a large number of detected bounding boxes to only a few. Windowing is used by most object detectors at the most basic level. Hundreds of thousands of windows of varying sizes and shapes are created. These windows are said to contain only one object, and each class is assigned a confidence score by the classifier. After the detector generates a huge number of bounding boxes, the best ones must be filtered away. The most often used algorithm for this task is NMS [8][14].

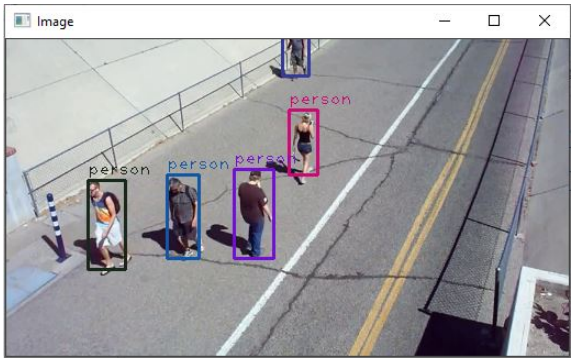

Fig. 4. People detected in one of the frames

As we can see in Fig 6, once an object is detected in a frame, we will get a lot of bounding boxes with different confidence values. So to get the best bounding box out of them, we applied this technique. In the next step, we have written the code to draw a bounding box around people in the frame. We have used classId to draw bounding

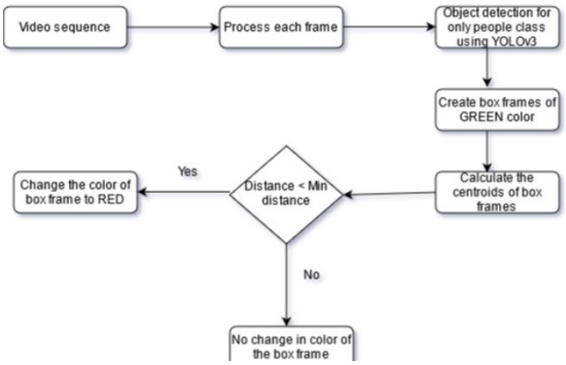

Fig. 5. Architecture

boxes only for people and exclude remaining objects in the frame. Once we get bounding boxes for all people, centroids are calculated for all of them. Then distance is calculated between all centroids. If the distance between the centroids is less than MIN_THRESHOLD, then the bounding box color is changed to Red. Similarly each frame is processed and people are tracked with centroids and bounding boxes. There's a counter at the top of the window which keeps track of High risk and Low risk people and it can be seen in Fig 7.

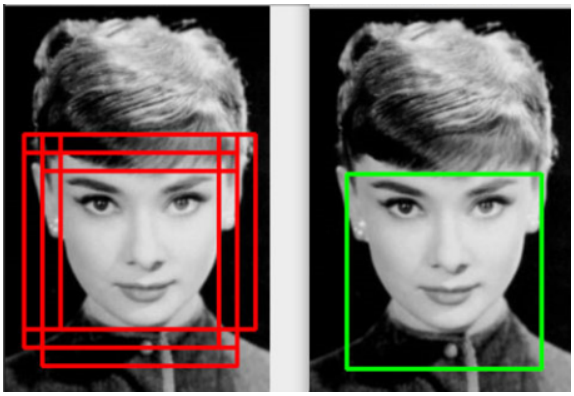

Fig. 6. Bounding Boxes

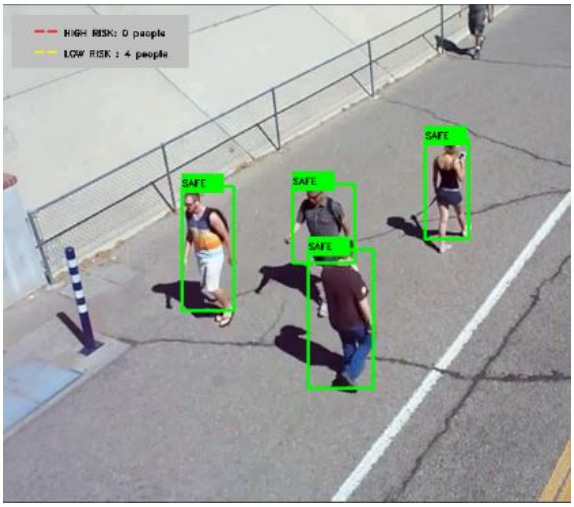

Fig. 7. Count of High Risk and Low Risk people

Fig. 8. Count of High Risk and Low Risk people

V. RESULTS

We have calculated the time taken by all 3 YOLO weights. The output can be seen in Fig 9. As mentioned in above sections, YOLO 320 takes less time compared to the other 2 models and YOLO 608 takes more time. As we can see in Fig 7, people are marked safe and in green rectangles as they maintain minimum distance without violating social distance. In Fig 8, bounding box color changed to red and count of high risk people was increased.

From the images we can see that that people are effectively detected by the model at several scene

locations. But, in some cases, as the appearance is changing, model misses few detections. It may be because, as we used pre-defined speed optimized weight files to train the model, individual's appearance changes from overhead view. These issues have been discussed in the next section with few examples.

```
High speed, less accuracy
Time taken by model trained with YOLO320: 116.609448 seconds
moderate speed, moderate accuracy
Time taken by model trained with YOLO416: 118.202419 seconds
less speed, High accuracy
Time taken by model trained with YOLO608: 119.283648 seconds
```

Fig. 9. Result

VI. Limitations and Challenges

Every machine learning or deep learning model has a drawback. Even we faced some problems when running the model. If the camera angle is in between top and parallel view, when people cross each other, the person is not detected. Also in some instances, the model detects shadow or pole as a person as seen in Fig 10. In the top angle view, the model does not detect all people in the frame. It detects multiple people in a single frame in a few instances as seen in Fig 11.

To overcome these issues, we tried implementing the top-down transformation (also known as Bird's eye view) once an object is detected in the frame. But we couldn't complete due to time constraints. For calculating the minimum distance we need to implement metric to pixel conversion which can be done using camera calibration. These can be considered as future works. Also other models can also be implemented and compared to YOLOv3.

VII. Conclusion

To conclude, we performed object detection using YOLOv3 and processed the videos to check if social distancing has been violated. We used 3 different speed optimized weight files to train our model and calculated the time taken by all three of them. The examples which we

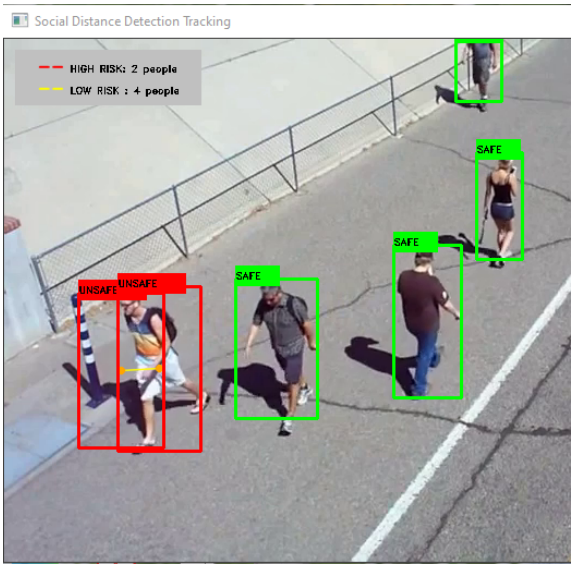

Fig. 10. Miss detection

Fig. 11. Miss detection from Top view

referred online to understand the logic behind the implementations have been listed in the references section [11][12][13]. The detection model provides bounding box information, which includes centroid coordinates. The pairwise centroid distances between identified bounding boxes are calculated using the Euclidean distance. An approximation of physical distance to the pixel is utilised to check for social distance violations between persons, and a threshold is set. A violation threshold is utilized to determine whether or not the distance value breaches the set minimum social distance. In addition, for monitoring persons in the scene, a

centroid tracking technique is used. The framework effectively identifies persons moving too close together and breaches social distancing.

REFERENCES

[1] https://nccid.ca/covid-19-variants/

[2] https://www.cdc.gov/coronavirus/2019-ncov/vaccines/fully-vaccinated-guidance.html

[3] Imran Ahmed, Misbah Ahmad, Joel J.P.C. Rodrigues, Gwanggil Jeon, Sadia Din. **"A deep learning-based social distance monitoring framework for COVID-19".** Accepted 19 October 2020. Available: https://doi.org/10.1016/j.scs.2020.102571

[4] V. X. Gong, W. Daamen, A. Bozzon, S. P. Hoogendoorn, **"Counting people in the crowd using social media images for crowd management in city events."** Available: https://doi.org/10.1007/s11116-020-10159-z(0123456789().,-vol(V0)12345.

[5] https://learnopencv.com/blob-detection-using-opencv-python-c/

[6] Punn, N. S., Sonbhadra, S. K., & Agarwal, S. (2020b). **"Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and Deepsort techniques."** arXiv:2005.01385.

[7] Ramadass, L., Arunachalam, S., & Sagayasree, Z. (2020). **"International Journal of Pervasive Computing and Communications"**

[8] https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/

[9] https://viso.ai/deep-learning/yolov3-overview/

[10] https://github.com/pjreddie/darknet

[11] https://gist.github.com/deepak112/7c95270d8e8baa414ef698b49f80c709

[12] https://github.com/ParthPathak27/Social-Distancing-Detector

[13] https://github.com/nandinib1999/object-detection-yolo-opencv/blob/master/yolo.py

[14] https://developer.ibm.com/recipes/tutorials/deep-learning/non-max-suppression/