**Social Distancing Tool**

The main goal of this project is to help determine and try to maintain an essential balance between work-life and pandemic control at the critical times of a pandemic outbreak. By using social distancing tools, many organizations can analyze and build a strategic approach to practice social distancing in order to stay safe. This project will help determine how distant people are in a place at a particular time that helps in taking effective measures in terms of social distancing.

**Datasets** : In this project, we have 2 videos. Both the videos will be converted into image frames. These image frames will be stored in data sets respectively and the images in the datasets will be used for training and object detection.

**Methodology:**

* Video Selection
* Creating datasets by converting video into image frames
* Training using faster R-CNN inception models
* Object Detection--Human Detection
* Creating bounding boxes
* Computing the distance between humans
* Converting the frames back into video

### Literature Survey:

Social distancing is one of the most effective measure that is being followed throughout the world during the critical times of a pandemic outbreak. By practicing social distancing, the spread of the virus can be constricted. It is by practicing social distancing that many countries over the world are able to reduce the impact of the corona virus on their people. This is very useful in crowded environments and can play major role in keeping people and communities safe. The social distancing tool can be merged with security cameras to analyze and monitor the progress. People are instructed to follow social distancing and using this project we can detect any violations on social distancing policy. Few organizations like Amazon are using social distancing tools in their warehouses to enhance social distancing policy.

Initially, I started researching about how to do my project and about how to build an object detection model for my project. I have known about multiple methods that I can work with to perform object detection on my data.

### Using Pre-trained Model:

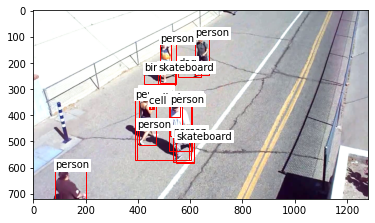
Initially, I thought of applying pre-trained models using which I can perform object detection. Here, I used resnet50 FPN model and performed object detection where initially, my sample video is converted into image frames and then these image frames are trained for object detection. Since, I have used a model that has been pre-trained, and the model has directly identified the objects in the image by drawing bounding boxes around them. Here, I used labels from COCO data set and train our images.

### Challenges and Solutions:

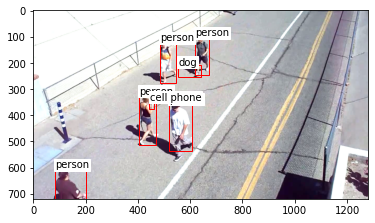
Since this model is predicting different objects other than humans, I tried to remove other objects like cars, mobile phones that are irrelevant and consider only human objects in that image. This is done by taking only objects with label value 1 which happens to be humans. But, I observed that shadows of humans are also being predicted as humans. Also, in some images, there were people over-lapping on others, which in-turn made the bounding boxes over-lap or coincide with other boxes.

Therefore, in order to overcome this problem, I adjusted Intersection Over Union(IOU) and used min\_score metrics and to avoid shadows and over-lapping.

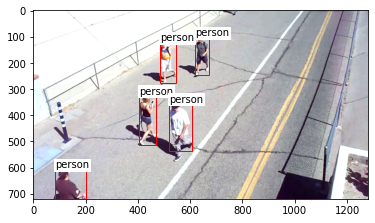
Similar to the previous method, I took an image and bounding boxes are drawn around all the objects. In the below image, we can observe that even the overlapping objects are detected. That is, IOU value is taken as 1. It can be observed that there are other bounding boxes inside one bounding box.



Here, I used Non-Max Suppression and, reduced the IOU index value to 0.1 because we do not need other objects. Here, objects with 10% over-lapping or less is considered and over-lapping boxes with more than 10% is ignored. By doing so, most of the other over-lapping object boxes are avoided. But, here boxes with humans and more than 10% of over-lapping are also being avoided.



Also, while trying to analyze the image output, shadows are still being detected. While trying to analyze this problem, I observed a trend in shadows, that is, Shadows has less predicted score. Therefore, min\_score is used as one of the filters to solve this problem. Here, we are changing the min\_score value accordingly to ignore the shadows. We take min\_score to be 0.9.



To get better predictions regarding overlapping, its index is taken as 0.5 and min\_score is taken as 0.9. We can observe the result on another image as well below.



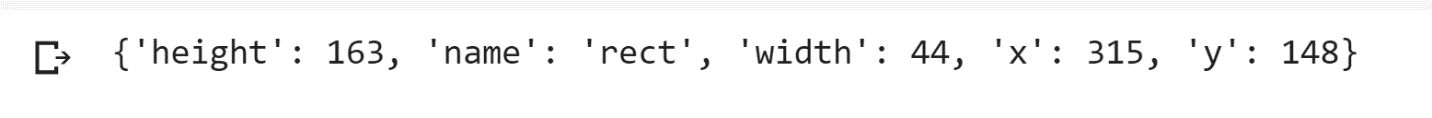
### Model Building :

I tried building a backbone layer for my network. Initially, I took a sample video and converted it into image frames. I used this approach mainly to train my own neural net with image data I have rather than using which are already pre-trained. And to do so, I need bounding boxes and their labels for each box. I used online tools and manually created bounding boxes and labels. The output for the bounding boxes is in the form of centroid, height and width. After that I Transformed that to a dictionary of boxes and labels which can be fed as an input to Backbone Network.

### Pre-processing:

Initially, a data frame is created with obtained bounding boxes values and labels by taking columns with filename, label and bounding boxes. Here, I resized the image frames into 600\*600 dimesions so that the training time will be less and more accurate results can be delivered.

 The result of bounding box stored as dictionary in a string is shown below.



Then, a new data frame is created by splitting Bounding Box dictionary with key as column names and values as rows.

In general, bounding box values are given to an image model in form of Top Left and Bottom Right and are called as xmin,ymin and xmax,ymax for processing.

Hence, centroid, width and height is converted to Top Left and Bottom Right of a bounding box. After dropping all the useless columns, we use groupby to group all the lists and labels of objects in one image as a dictionary by using filename. And then all the values and image arrays are converted into tensor format.  Since faster RCNN only accepts image array as a list of tensors and Bounding boxes and labels as list of dictionary, we give the input as required.

Here, I have built a backbone network similar to convolutional classifier network but here I did not use linear layers or flatten the output because my goal here is to learn features by training my own input data and not to classify images. After that, I passed my data on to Faster RCNN predefined model to do the prediction.

Problems:

The Model which I trained using my input data is not predicting any output on test images, though it's running fine. I have tried different possibilities, and as per my knowledge, I think the issue is because of input labels or because of size of training set.

### Results :

After object detection step, I had to calculate the distance between people to check if they are socially distant or not. To do that, I used a centroid point which is the top-center point of all bounding boxes. Then, I calculated the distance between them. Here, to understand the video better, people who are socially distant are represented with green boxes where as people who are not socially distant are represented with red boxes. This can be installed in cameras at multiple public places to analyze whether people are following social distancing especially during the times of virus outbreaks.

[Street\_video.avi](https://drive.google.com/open?id=1Cr9nKFGZWEhdq1pzWP_07JHjiz63zKMA&authuser=1)

[bridge\_video.avi](https://drive.google.com/open?id=1Ct8CFhjWiFvGWFCxJxUo84yf-VCIfduC&authuser=1)

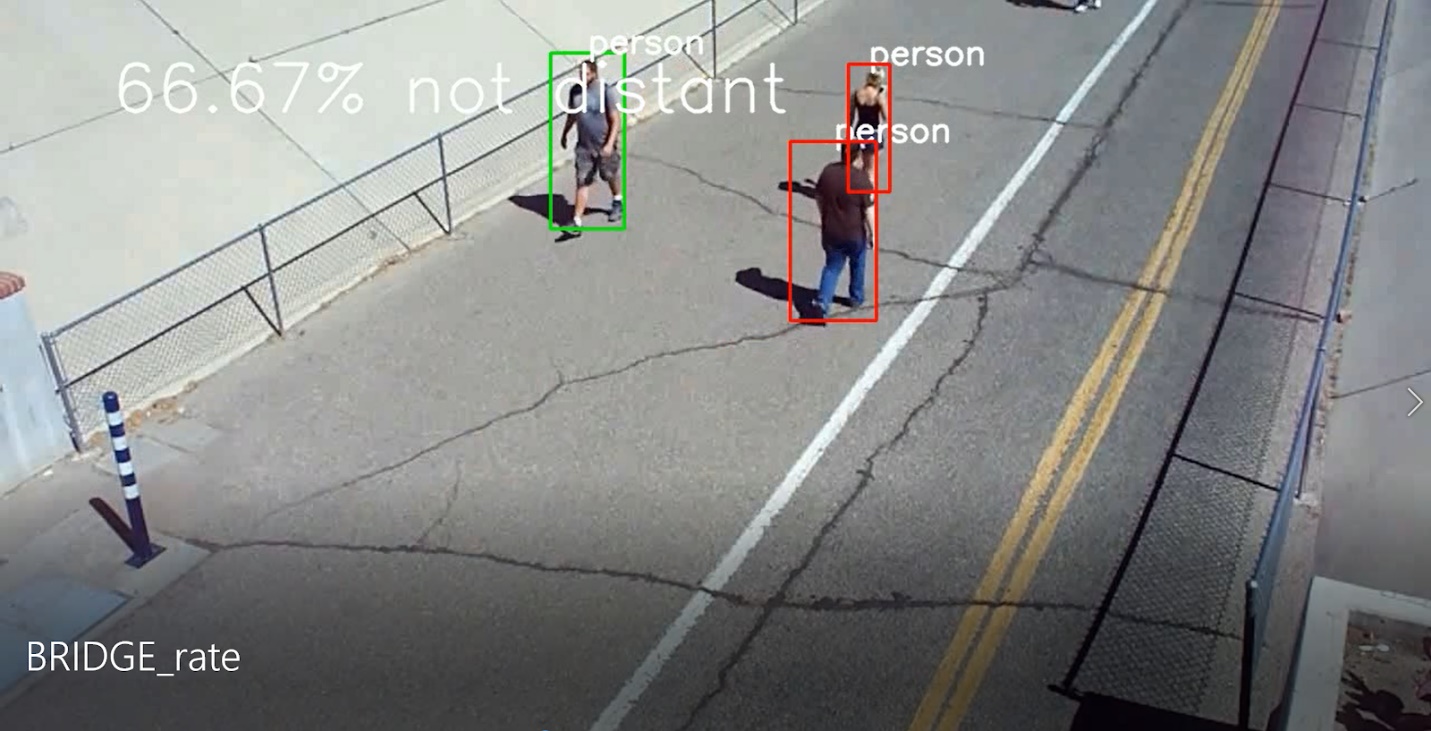
Bird's Eye View:

To get the bird’s eye view of the video , I used four point transformation method where, the 4 points are in the order of—top left, top right, bottom right, bottom left of the bounding box. I used getperspectivetransform and warpperspective methods from cv2 to transform the image frame into bird's eye view.

Percentage Calculation:

Here, the percentage of people who are close to each other is calculated based on the people in red boxes. Percentage is calculated for all image frames and is shown in the video as '% not distant'.

One of the issues I faced is sorting the images to convert them back into video in the correct order. I solved it using sort function and re.sub() using which the image frames are sorted before being converted into video.





### Backbone Model:

For the backbone model, since previously there was no output being displayed and assuming the problem to be with region\_ids of the bounding boxes, I labelled each person with unique name like p1, p2, p3 and so on, in the online tool. When these images along with the box coordinates were given to the model, there were no accurate predictions. Here, the accuracy was very less and due to less accuracy, more irrelevant boxes were being predicted even where there are no objects. The predicted score of the boxes was around 0.4-0.5 for the backbone model.

To get better predictions, I increased the input data by taking the images and bounding box coordinates of two different videos from the pre-trained model, combined them, converted them into tensor and passed them onto the model. But, even then, the predictions were not accurate.

### Conclusions and Future Work:

With respect to the industrial application, we can install this in cameras at any public or work-places to know whether people are following social distancing or not which is the main aim of the project. In the future, this will be more appropriate if we can find ways to alert people when they are not being socially distant at a work-place.

By working with more layers in the model, model performance might be enhanced and the accuracy can be improved. Also, by working with unsupervised learning techniques, accuracy might be improved.

### References:

1. <https://arxiv.org/pdf/1506.01497.pdf>
2. <https://arxiv.org/pdf/1504.08083.pdf>
3. <http://calvin-vision.net/wp-content/uploads/Publications/alexe12pami.pdf>
4. <https://www.pyimagesearch.com/2020/06/29/opencv-selective-search-for-object-detection/?__s=ykidezmtyvpcvobs1rc5>
5. <https://www.pyimagesearch.com/2020/07/06/region-proposal-object-detection-with-opencv-keras-and-tensorflow/>
6. <https://www.telesens.co/2018/03/11/object-detection-and-classification-using-r-cnns/#Anchor_Generation_Layer>
7. <https://arxiv.org/pdf/1904.00641.pdf>
8. <https://www.analyticsvidhya.com/blog/2018/06/understanding-building-object-detection-model-python/>
9. <https://arxiv.org/pdf/1902.09968.pdf>
10. <https://hackernoon.com/the-best-image-annotation-platforms-for-computer-vision-an-honest-review-of-each-dac7f565fea>
11. <http://www.robots.ox.ac.uk/~vgg/software/via/via-1.0.6.html>