Report

Design

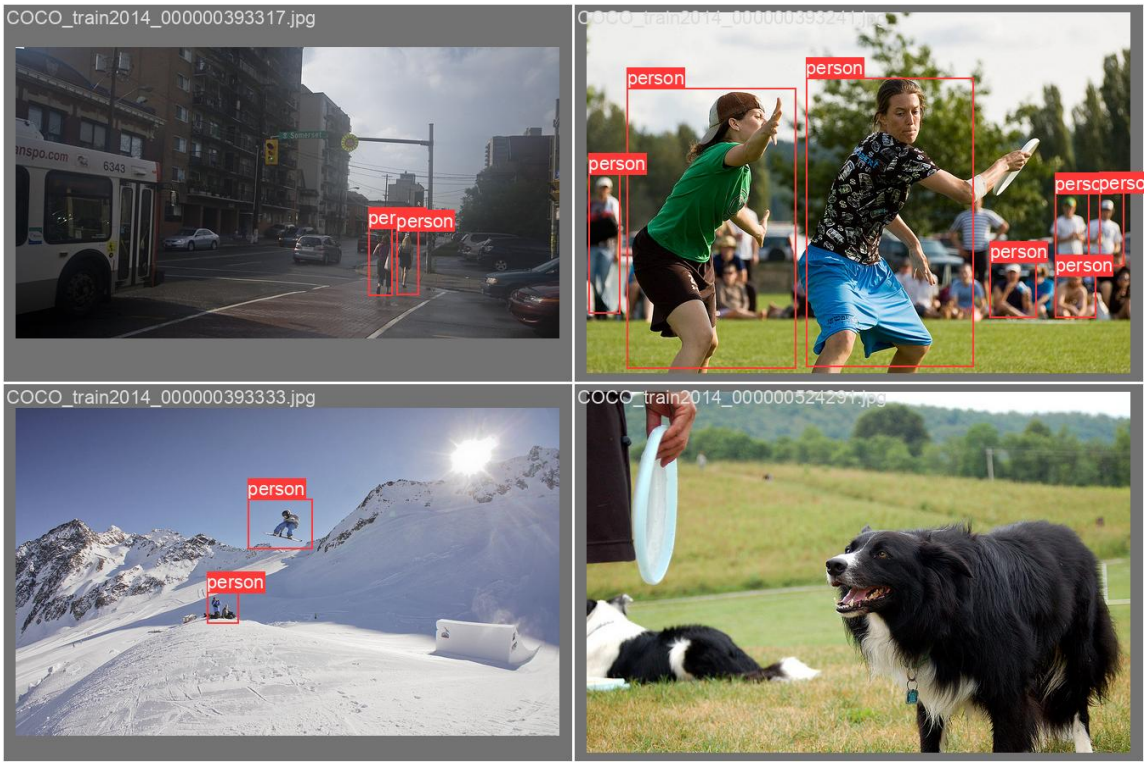
- **The project is designed in a way to detect humans and also track them in a live stream**
- **The models used for this are Yolov5s (small) and Yolov4 - tiny**
- **If we prioritize accuracy and are willing to allocate slightly more computational resources, Yolov5s may be the better choice.**
- **It's likely to provide better detection performance while still being relatively lightweight compared to larger models.**
- **If real-time performance is of utmost importance and we can accept some trade-offs in detection accuracy, Yolov4 Tiny might be suitable for your use case.**
- **Hence both models are designed to be lightweight and efficient, but they have different trade-offs in terms of accuracy and performance.**

For Yolov5s

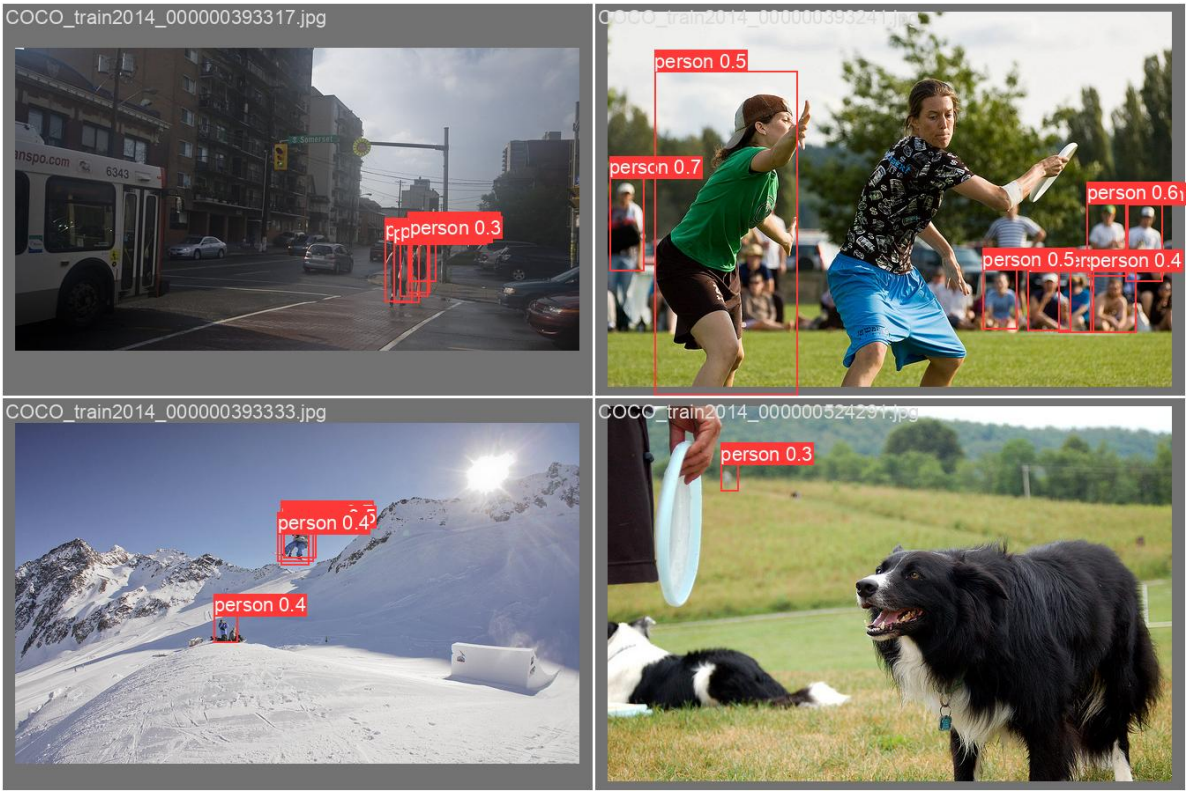
Implementation

1. **I have cloned Github and installed all the required dependencies**
2. **Loading of the training dataset**
3. **This dataset is loaded from official website of COCO**
4. **Labelling of the different images was done before loading it and it was saved in form of .txt file so as to be used with Yolo.**

Detection :



Actual labelling

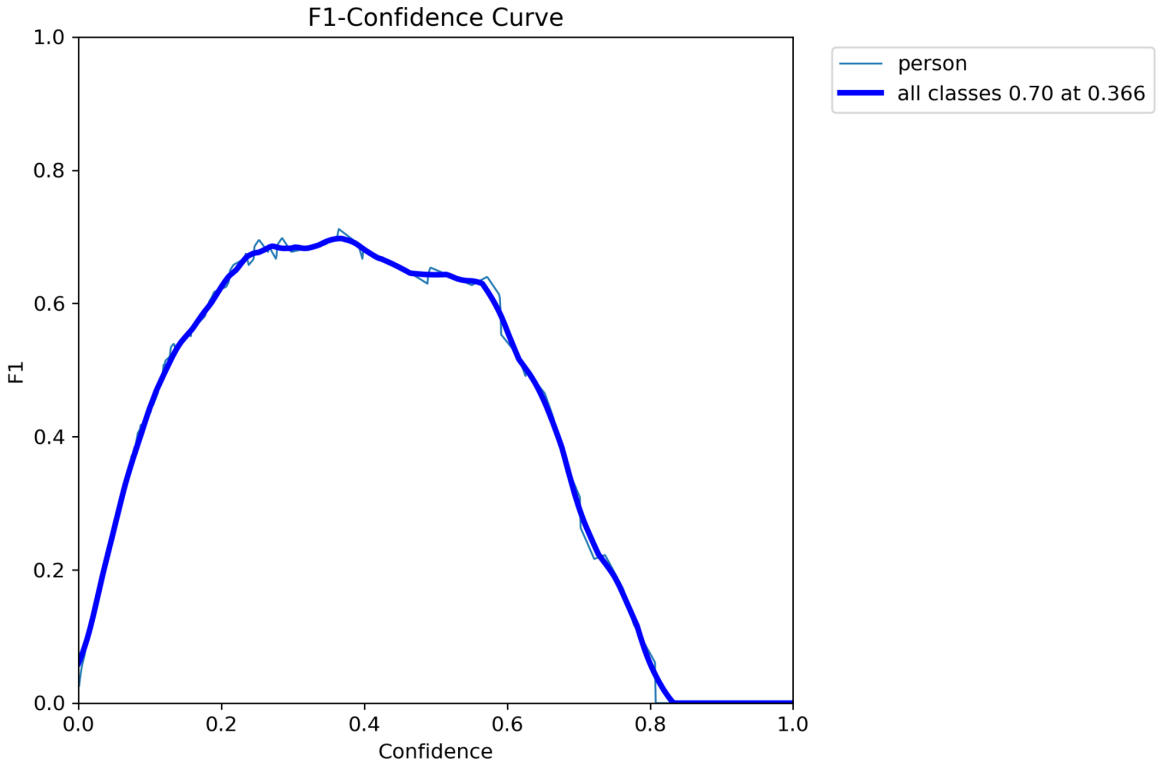


Prediction

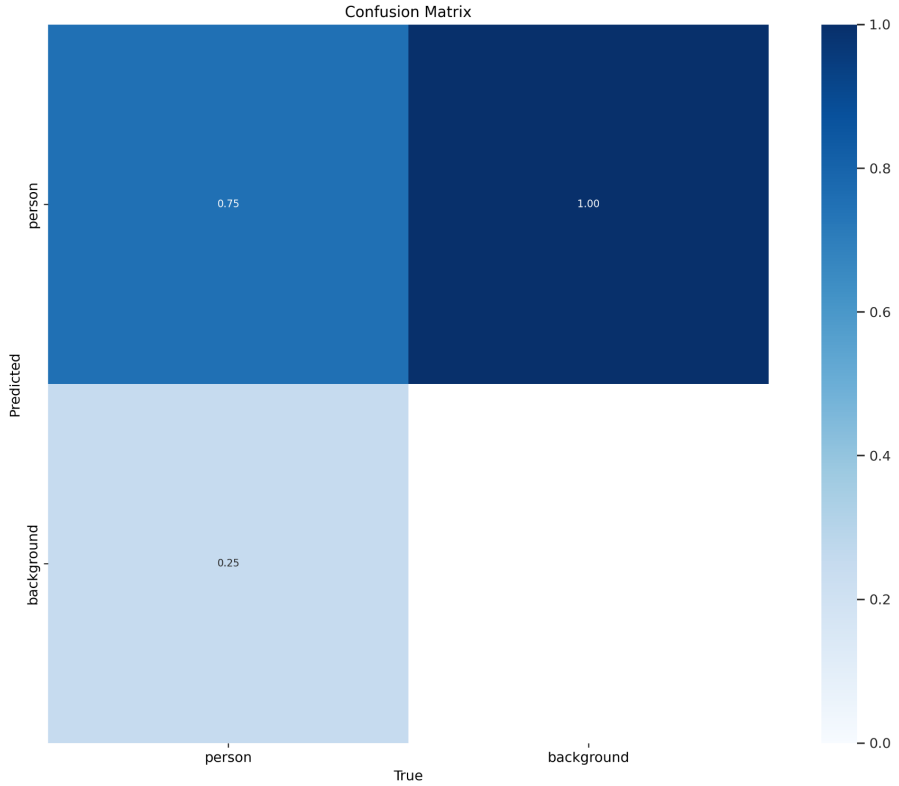
Evaluation

Results for prediction :

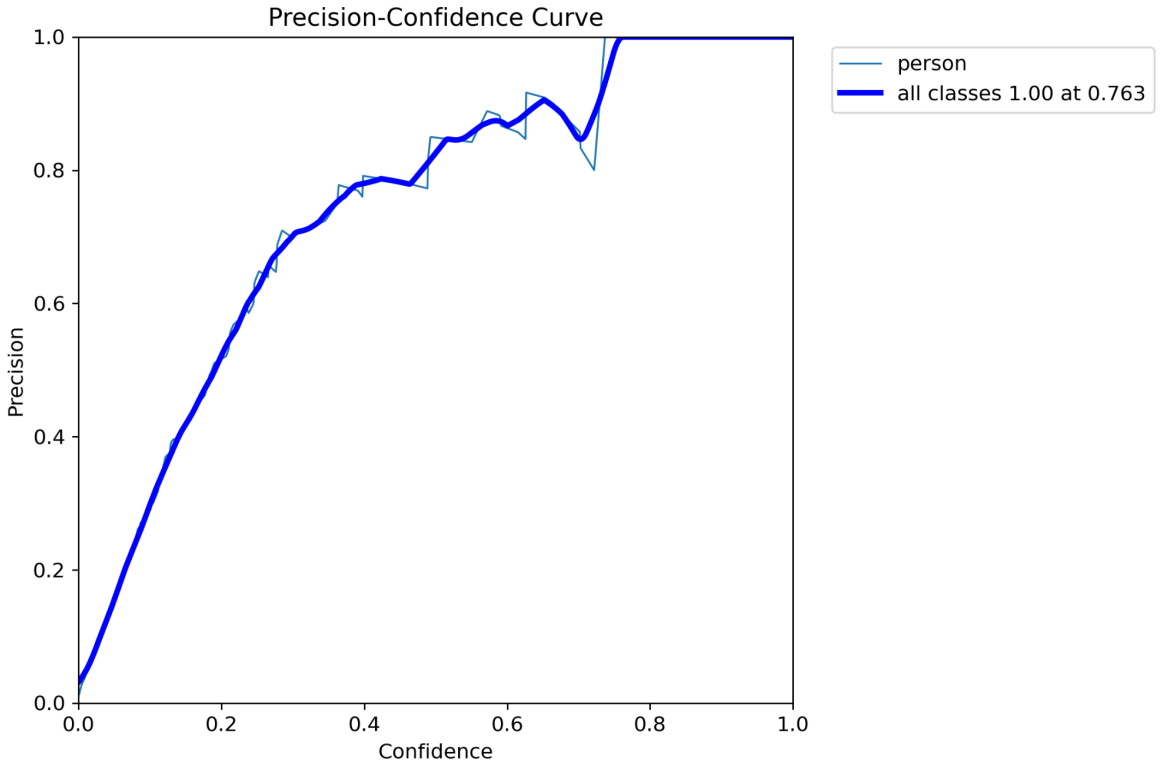
F1 curve



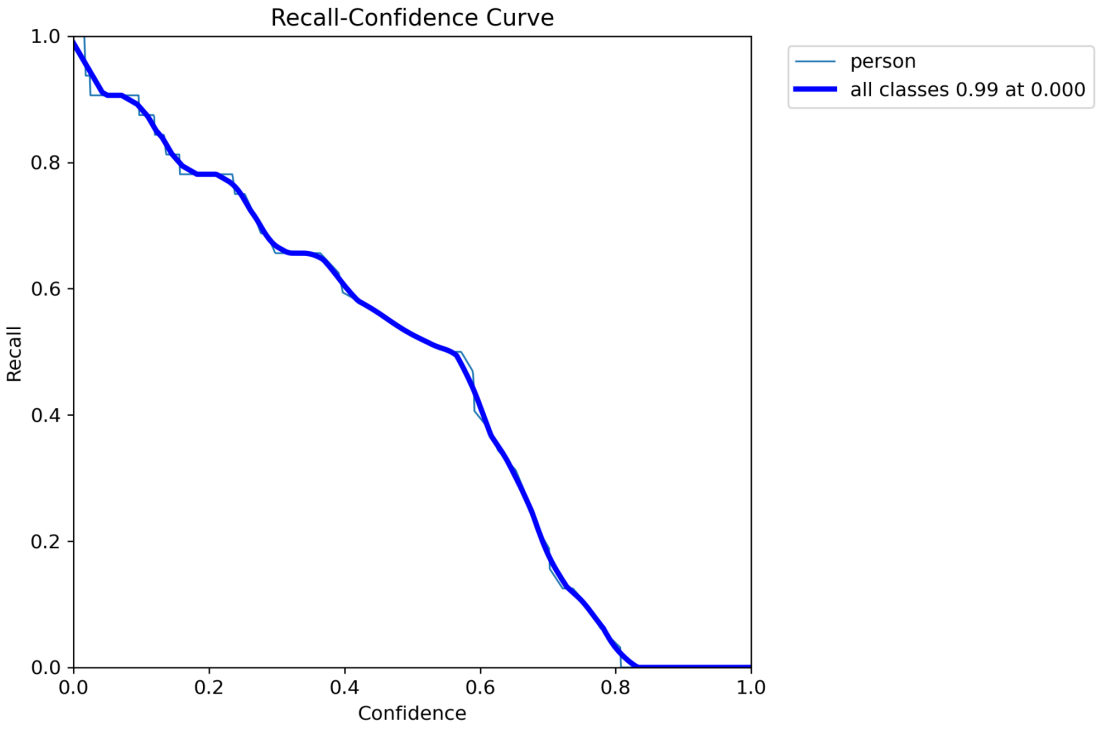
Confusion Matrix:



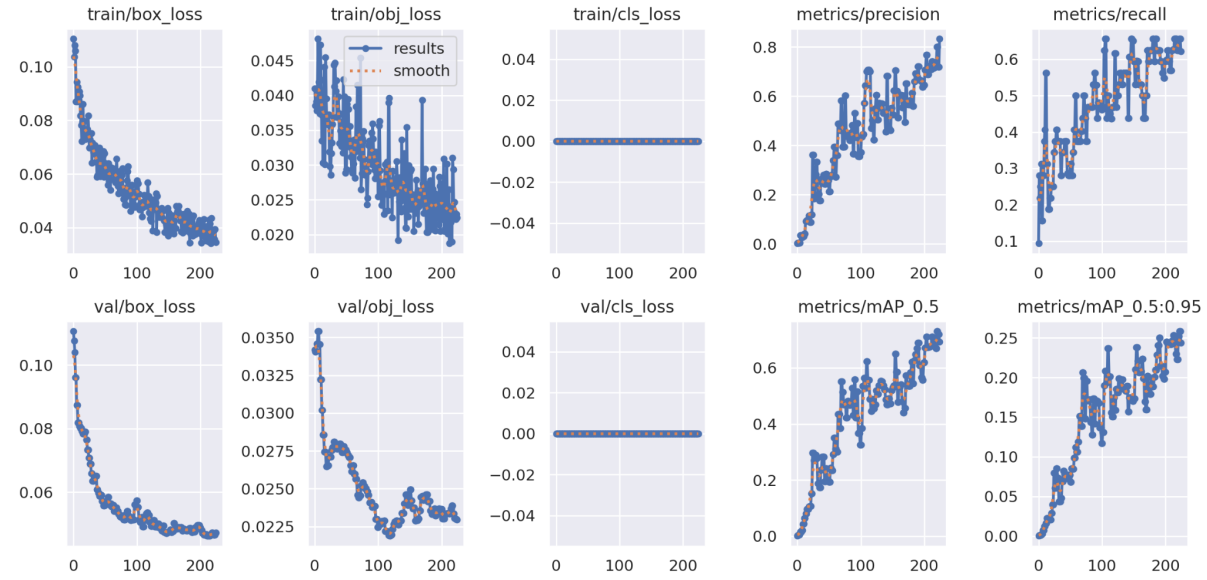
Precision Curve :



Re-call Curve:



Overall Results:




```
225 epochs completed in 2.676 hours.
Optimizer stripped from runs/train/exp2/weights/last.pt, 14.4MB
Optimizer stripped from runs/train/exp2/weights/best.pt, 14.4MB

Validating runs/train/exp2/weights/best.pt...
Fusing layers...
Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs
               Class     Images  Instances          P          R      mAP50   mAP50-95: 100%|██████████| 3/3 [00:03<00:00,  1.32s/it]
                 all         12         32      0.718      0.656      0.723      0.259
```

The overall accuracy of the model is 72.3%

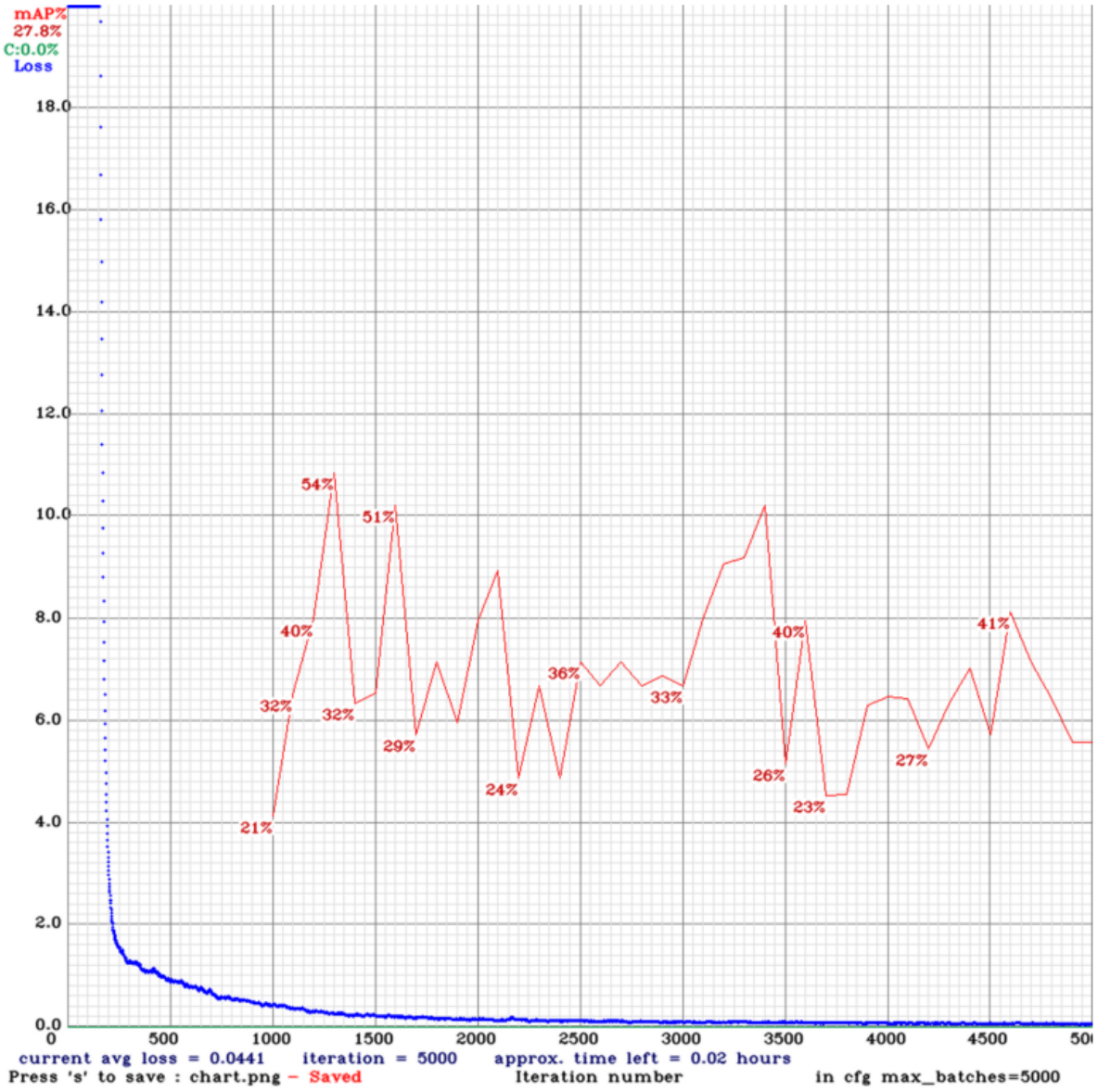
Human detection and Tracking - https://youtu.be/L4IOtyuSDXw

Yolov4 - tiny

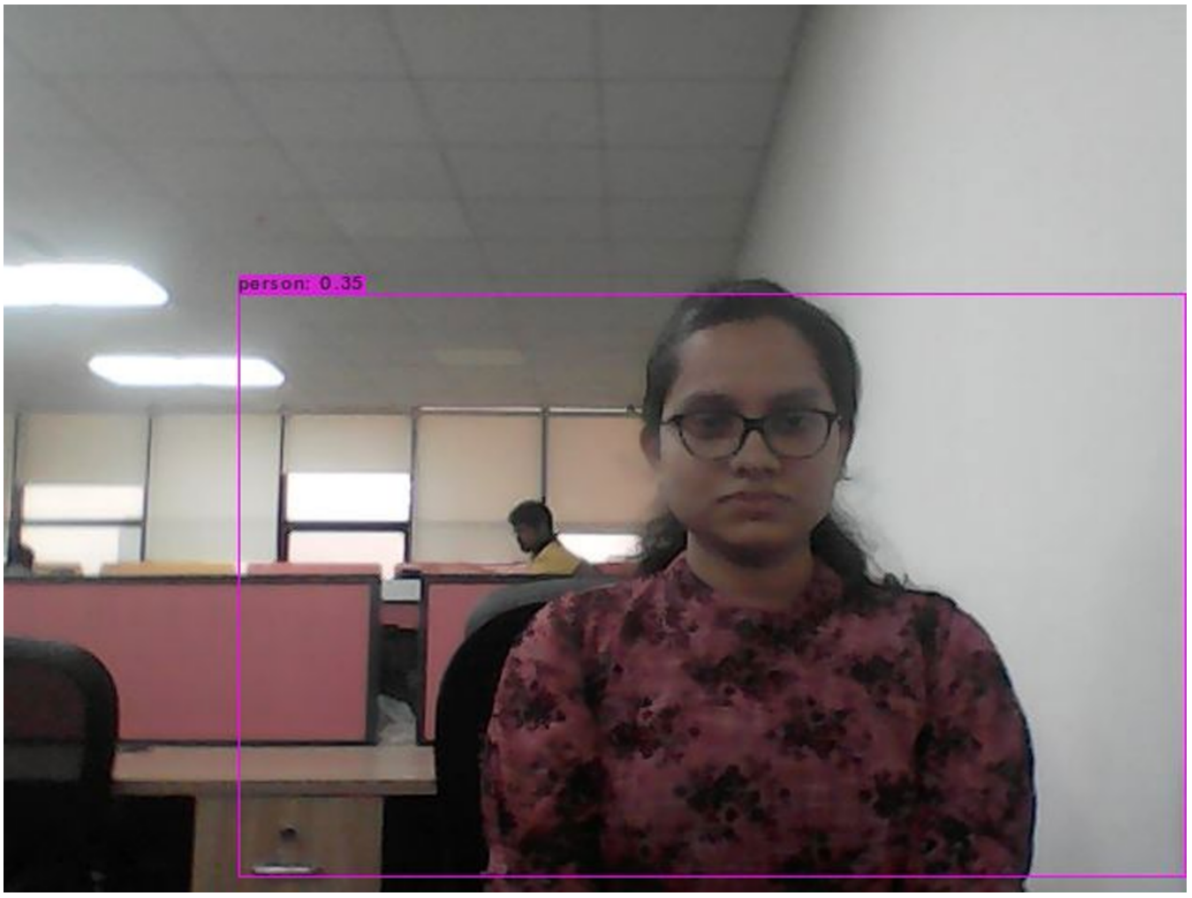
Implementation

1. **Creating the required files containing the training and testing data and other files having configuration of the required data.**
2. **Installing the required dependencies to create darknet.**
3. **Copying the files in the darknet and downloading the weights**
4. **Training the model with the required number of iterations**
5. **Number of iterations is decided based on the number of classes**

Evaluation



Overall summary of the model



Detection through web-cam

```
 (next mAP calculation at 5000 iterations)

 Tensor Cores are used.
 Last accuracy mAP@0.50 = 27.78 %, best = 54.17 % H5000/5000: loss=0.1 map=0.28 best=0.54 hours left=0.0
 5000: 0.050108, 0.044081 avg loss, 0.000261 rate, 0.908194 seconds, 320000 images, 0.022146 hours left

 calculation mAP (mean average precision)...
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
3
 detections_count = 10, unique_truth_count = 7
class_id = 0, name = person, ap = 27.78%         (TP = 3, FP = 5)
```

The last accuracy of this model is 27.78% (quite low) and best is 54.17%

Detection through live feed of camera - https://youtu.be/jrDGounPRnY

Human detection and tracking - https://youtu.be/7g8P8DGo1kQ