Comparative Deep Learning Approaches for Drone-Based Object Detection

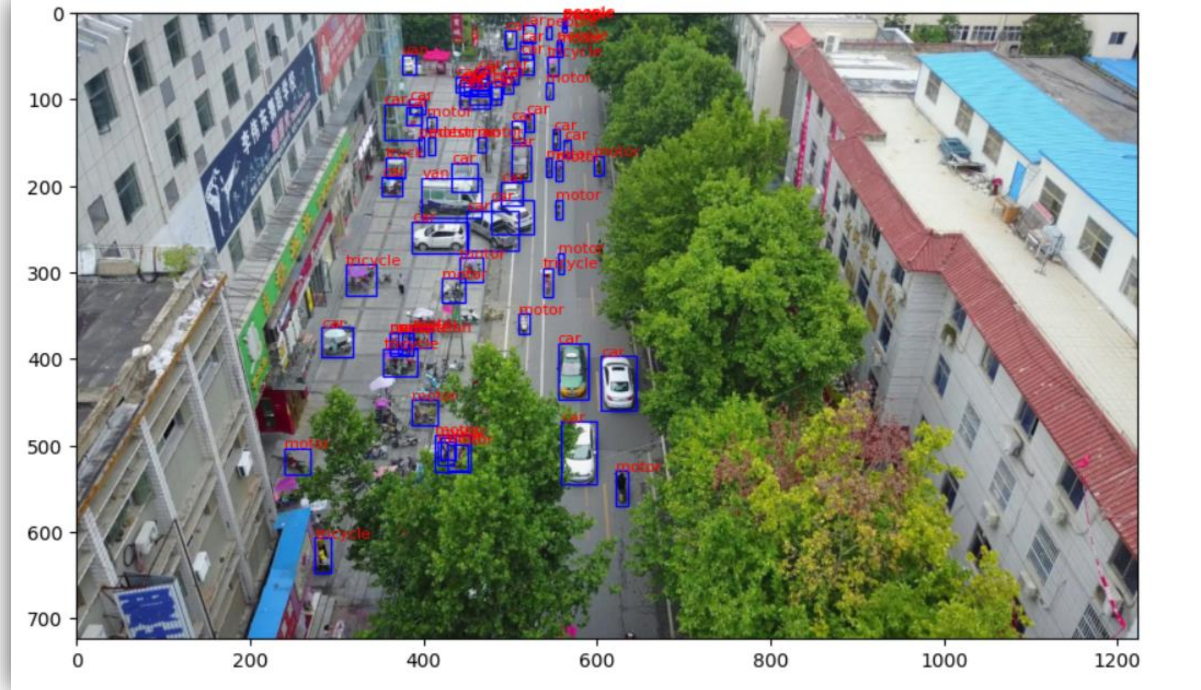
Nimesh Arora (Team Lead).
Ashish Gurung.
Dikshant Sagar.
Harika Yarramalli.
Ismael Valenzuela.
Kavya Reddy.
Safal Rijal.
Shreyas Teli.

Dataset named VisDrone

- •261,908 frames and 10,209 static images
- •14 cities in China, urban and country environments
- •2.6M manually annotated bounding boxes of pedestrians, cars, bicycles, and tricycles
- •Scene visibility, object class, and occlusion attributes
- •Collected using various drone platforms in diverse conditions.

What can we detect?

- Targets of interest: Pedestrians, cars, bicycles, tricycles, buses, trucks, motorcycles, and people on electric bicycles
- Levels of occlusion and scene visibility vary, making detection challenging
- Annotations provided for both static images and video clips
- Includes object attributes such as the number of visible sides and type of occlusion



Application/Motivation

- Train and evaluate object detection algorithms for drone-based surveillance.
- Improve the safety and efficiency of drone-based transportation systems.
- Develop autonomous drone systems for obstacle detection and avoidance.
- Provide insights into pedestrian and vehicle traffic patterns for urban planning.
- Monitor and predict the spread of infectious diseases in crowded areas.

Features and Labels

Features

- High resolution images captured by high-altitude drones.
- Large-scale dataset contains covering a wide range of scenes and environments.
- Detailed annotations for each image, including bounding boxes for objects of interest, occlusion levels, and truncation levels.
- Images captured in challenging scenarios, such as crowded scenes, occlusions, and small objects
- Benchmark for evaluating the performance of object detection algorithms. High-resolution

Labels

- Pedestrian
- Car
- Truck
- Bus
- Train
- Motorcycle
- Bicycle
- Tricycle

Environments, Tools & Frameworks

Project Challenges

- Data Format Inconsistencies

➢ The algorithm implementations used different Image file Formats, Annotation Formats, and Naming Conventions.

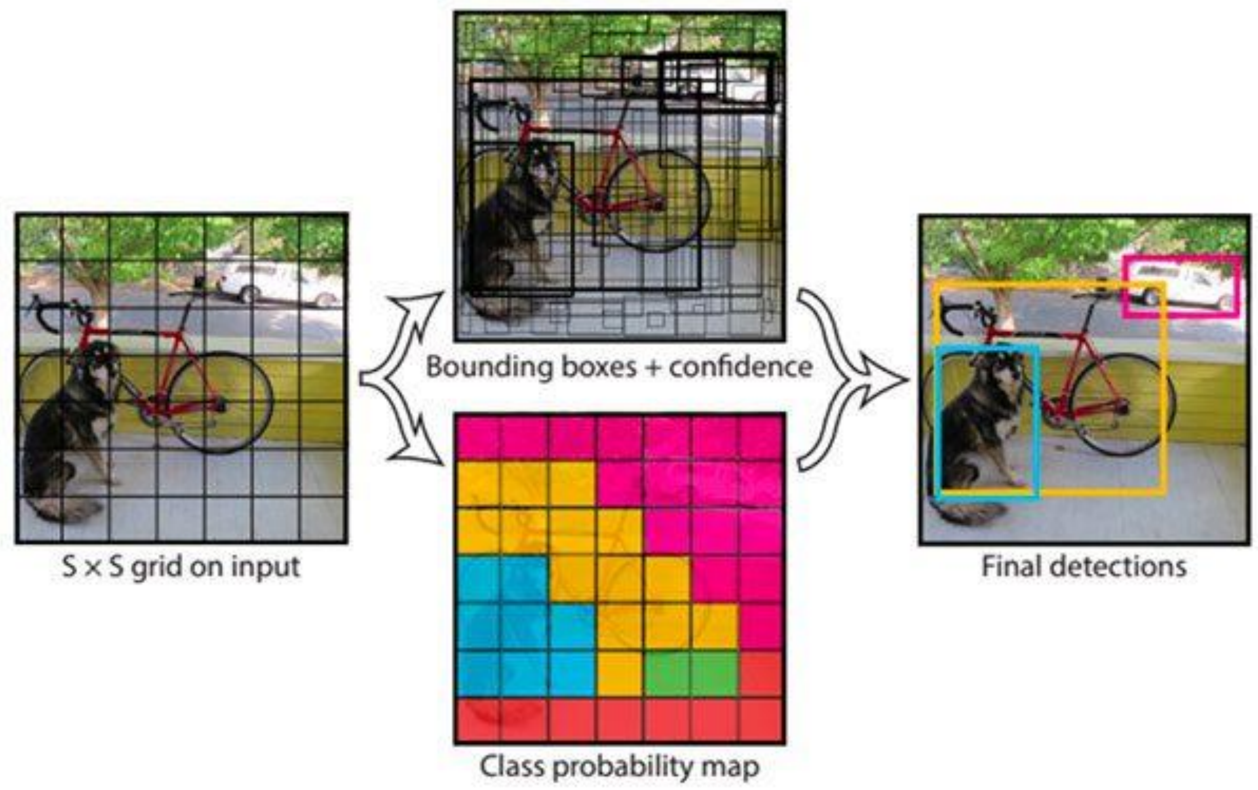
➢ Required time and effort to manually resolve the inconsistencies and preprocess the data to fit the model's input format.

➢ Lack of Computational Resources

Limited to Google Collab & Kaggle (Free GPU)

Models Implemented

- Single Shot Detector
- Faster RCNN with Resnet50
- Faster RCNN with Resnet50_fpn
- Mask RCNN
- Yolo v3 (Extra-Large)
- Yolo v5 (Small and Extra-Large)
- Yolo v8 (Small and Extra-Large)



Evaluation Metrices

- Quantitative Metrics

- Precision
- The fraction of correctly predicted positive samples (true positives) out of all positive predictions made by the model (true positives and false positives).
- Precision = $\frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}}$

- Recall
- The fraction of correctly predicted positive samples (true positives) out of all actual positive samples present in the dataset (true positives and false negatives).
- Recall = $\frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}}$

- PR Curve
- A graphical representation of the precision-recall trade-off for different thresholds used in the model.

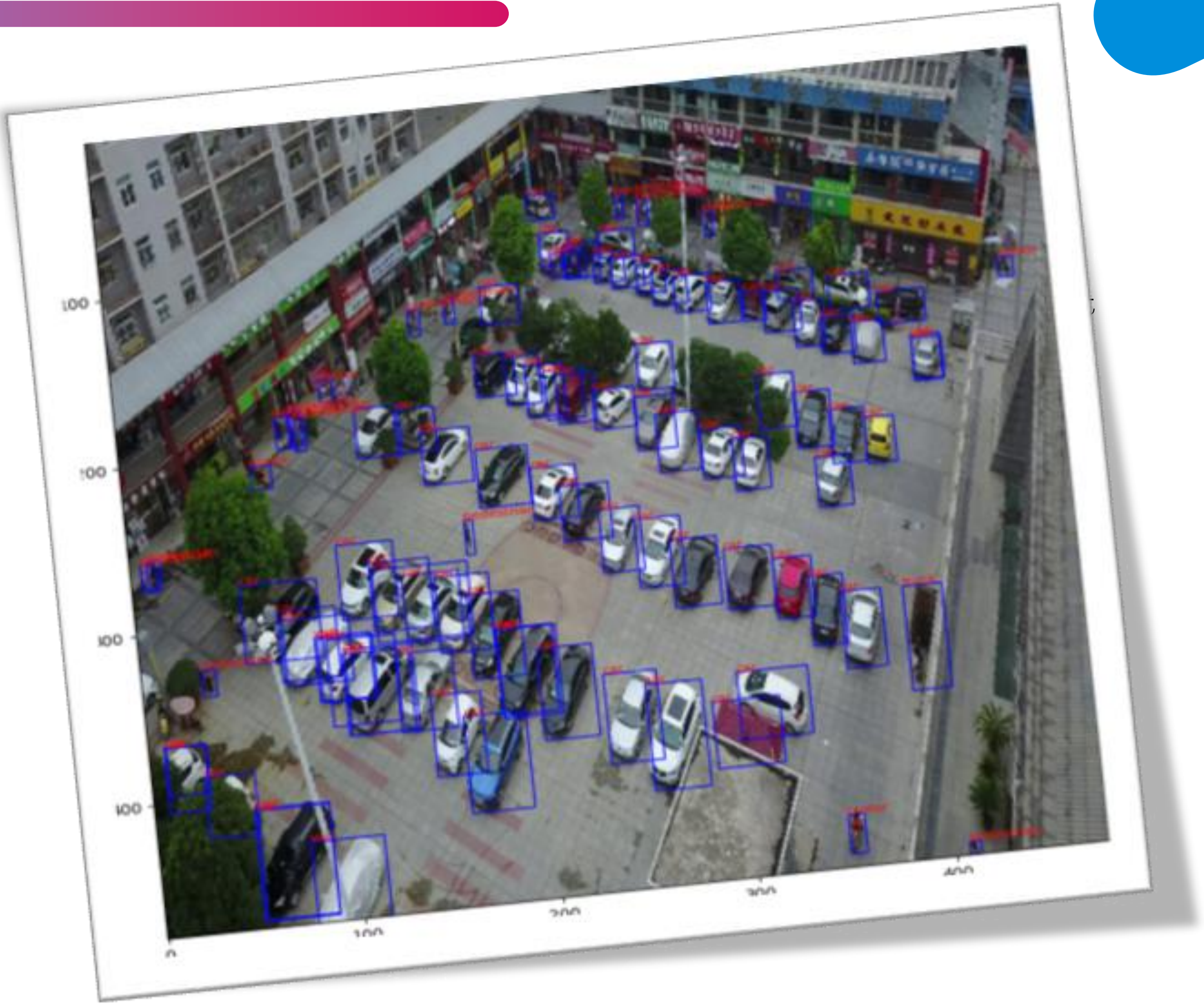
- mAP(mean Average Precision)
- It calculates the average precision across all recall levels for a given set of classes.

$$mAP = \frac{1}{N} \Sigma AP_i$$

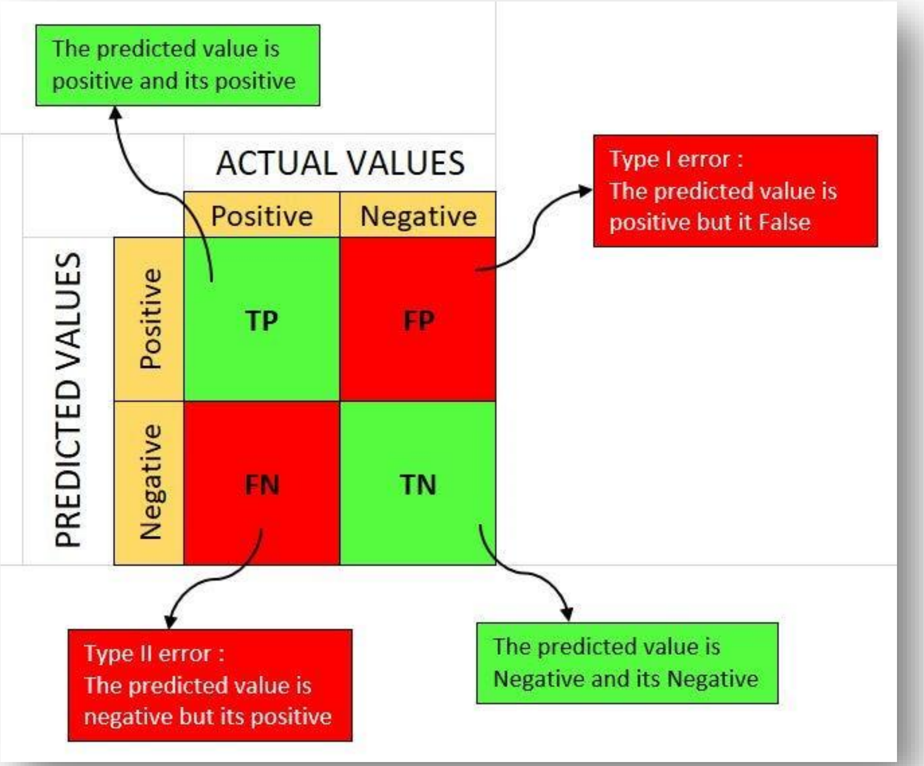
Evaluation Metrices

- Qualitative Metrics

  - Visual Plotting of prediction of Image



Evaluation Metrics



- Composite Metrics

- F1 Score/Curve:

  The harmonic mean of precision and recall. It provides a balanced measure of model performance across precision and recall.

- Confusion Matrix:

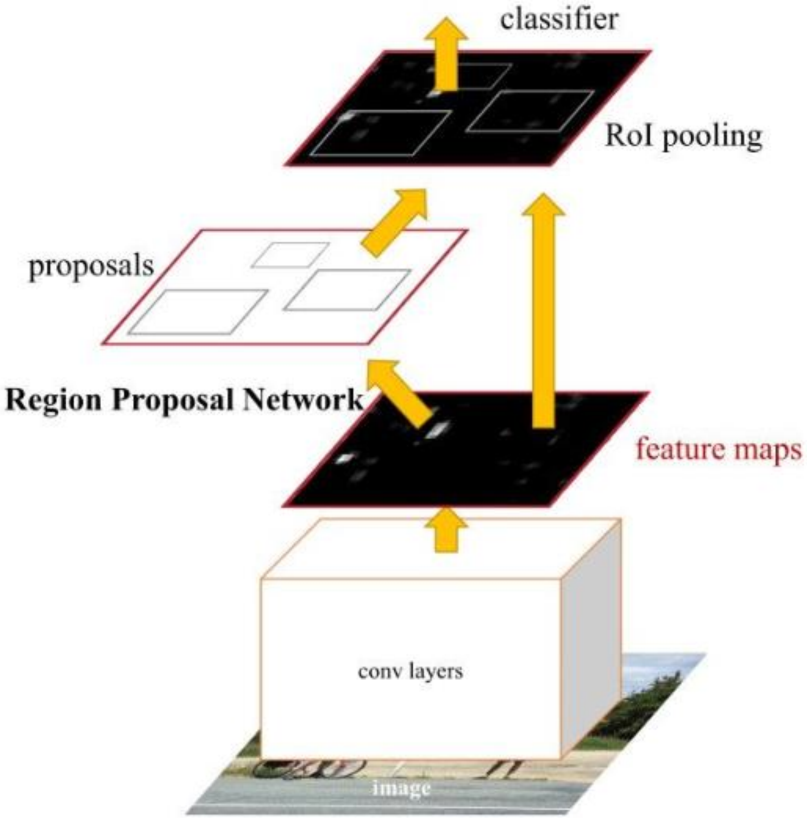
  Statistical classification, a confusion matrix, also known as an error matrix[11] is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one; in unsupervised learning, it is usually called a matching matrix.

Preprocessing

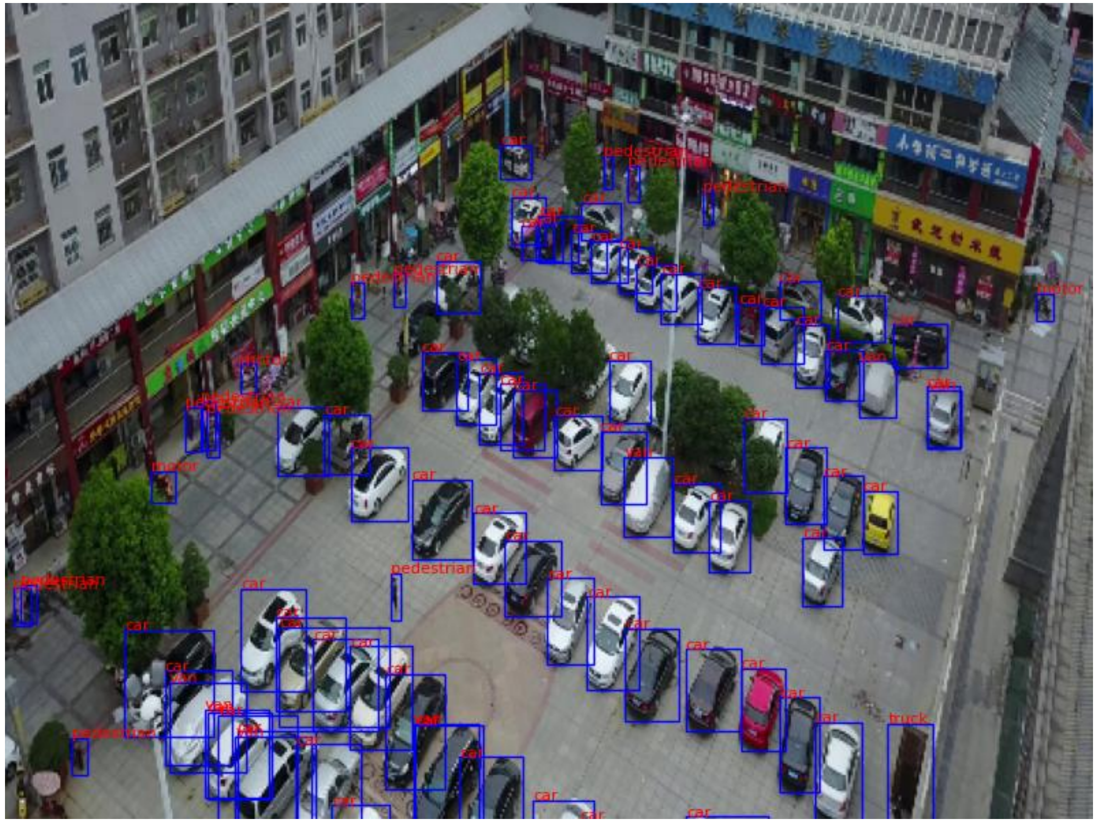
These steps ensures that the images are properly formatted and normalized before being used as input to the neural network, which can help improve the accuracy of the object detection model.

- Loading in the image file using PIL or Open-CV.
- Resizing image to a fixed size
- Translating the bounding box coordinates accordingly.
- Converting the image and coordinates to tensor.
- Normalizing the Image and coordinates.

Faster RCNN

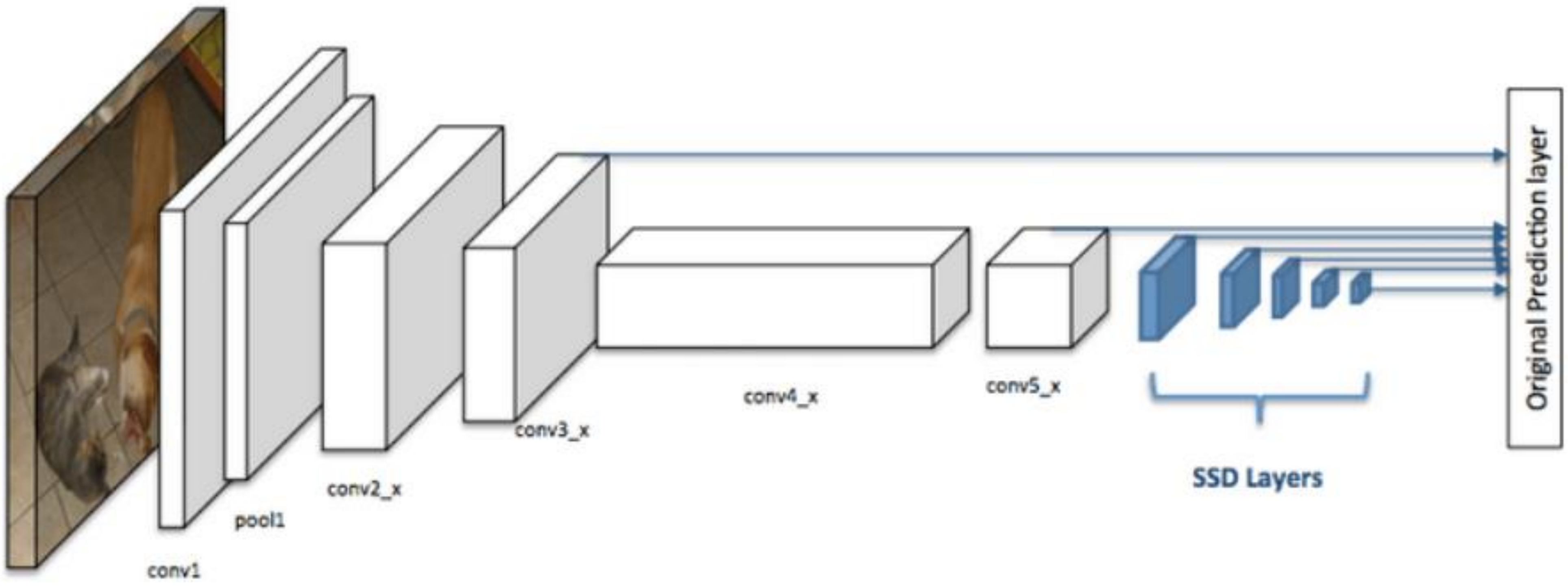

classifier
RoI pooling
proposals
Region Proposal Network
feature maps
conv layers
image

Faster RCNN

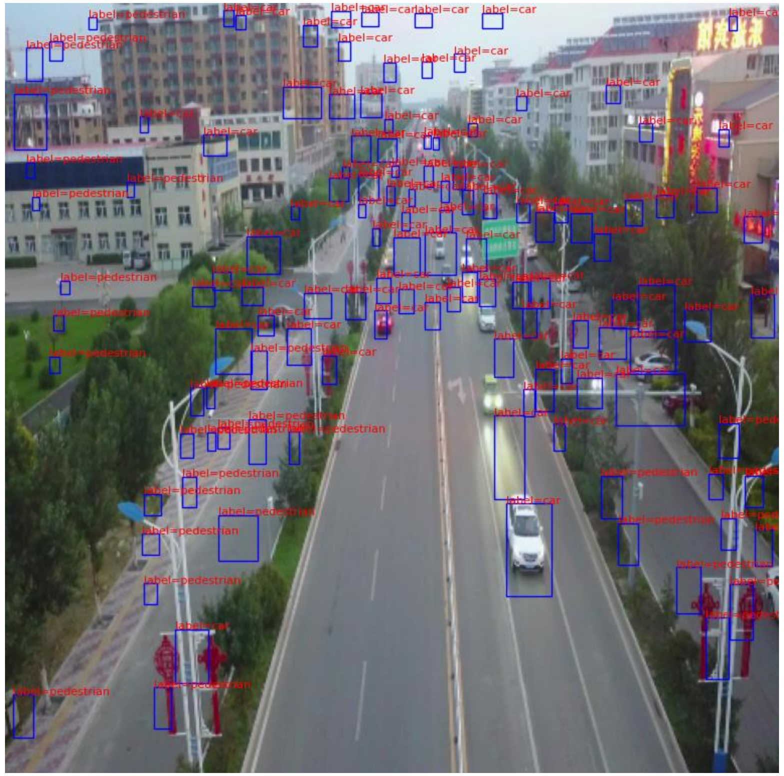


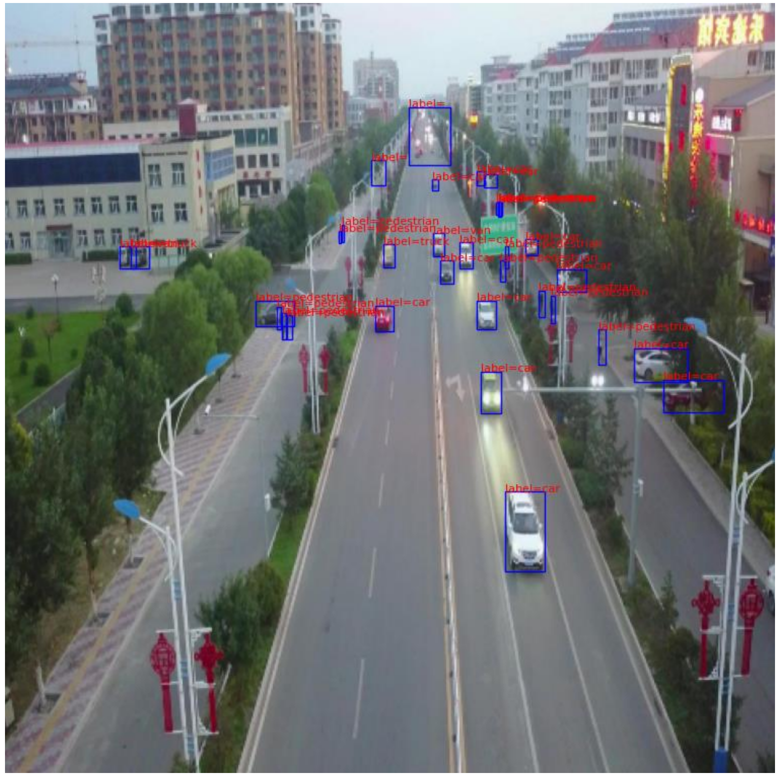
- Resnet50 & Resnet50-FPN
- Applied Non-Max suppression
- Trained in 10 epochs
- maP score of 0.23 & 0.27

Single Shot Detection



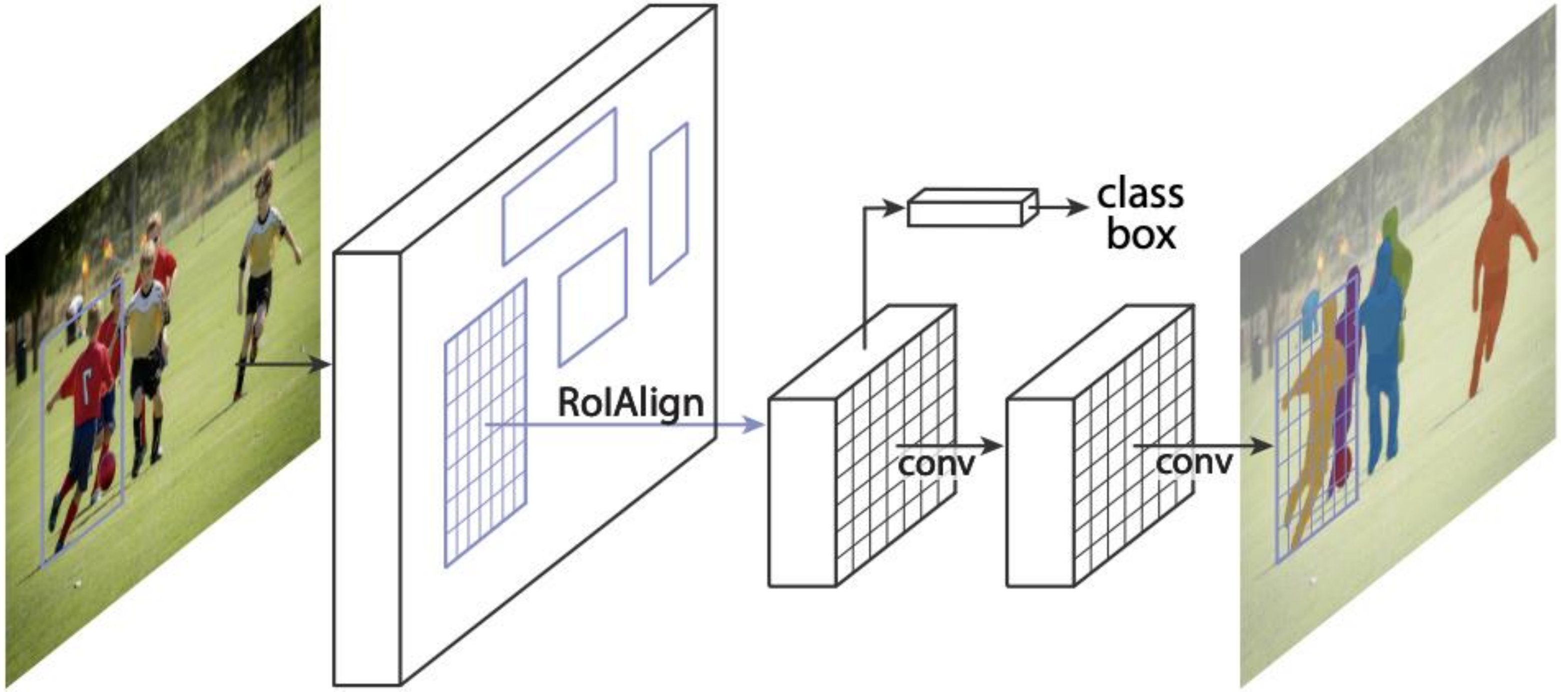
Single Shot Detection



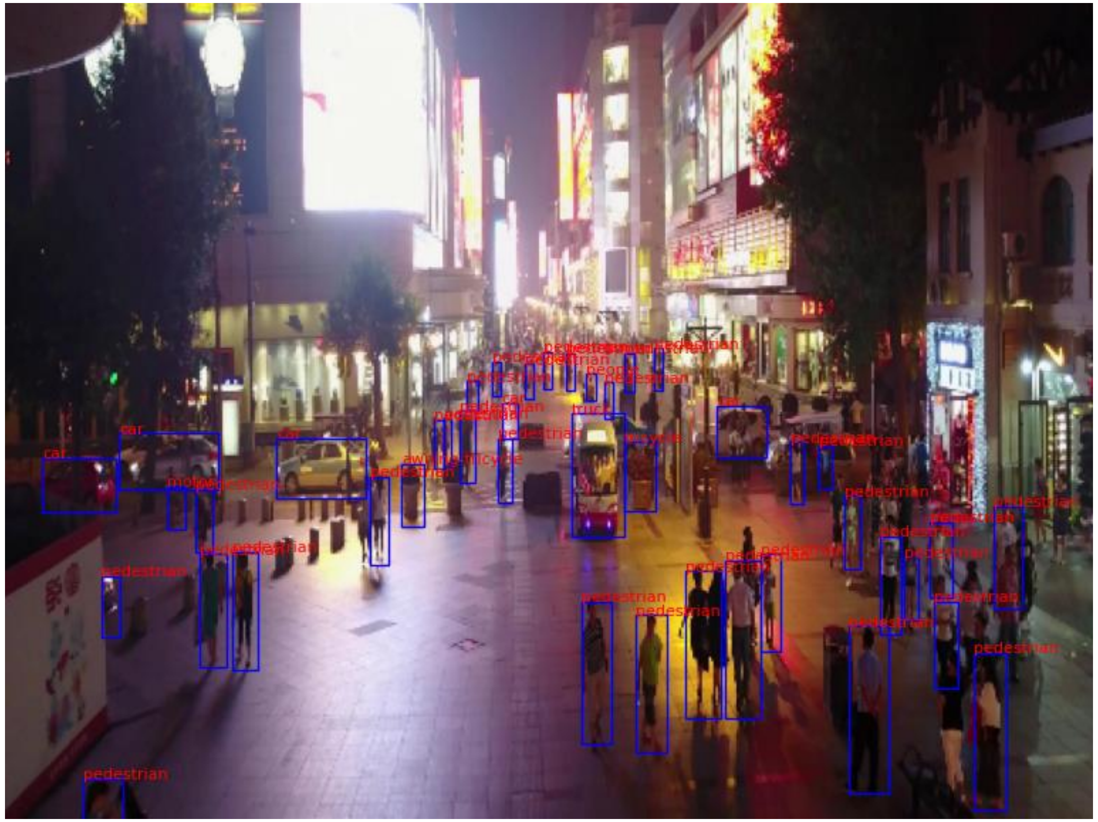


- SSD320Lite-mobilenet-v3-large
- Applied Non-Max suppression
- Trained on 2 epochs
- MAP : 0.12

Mask RCNN



Mask RCNN

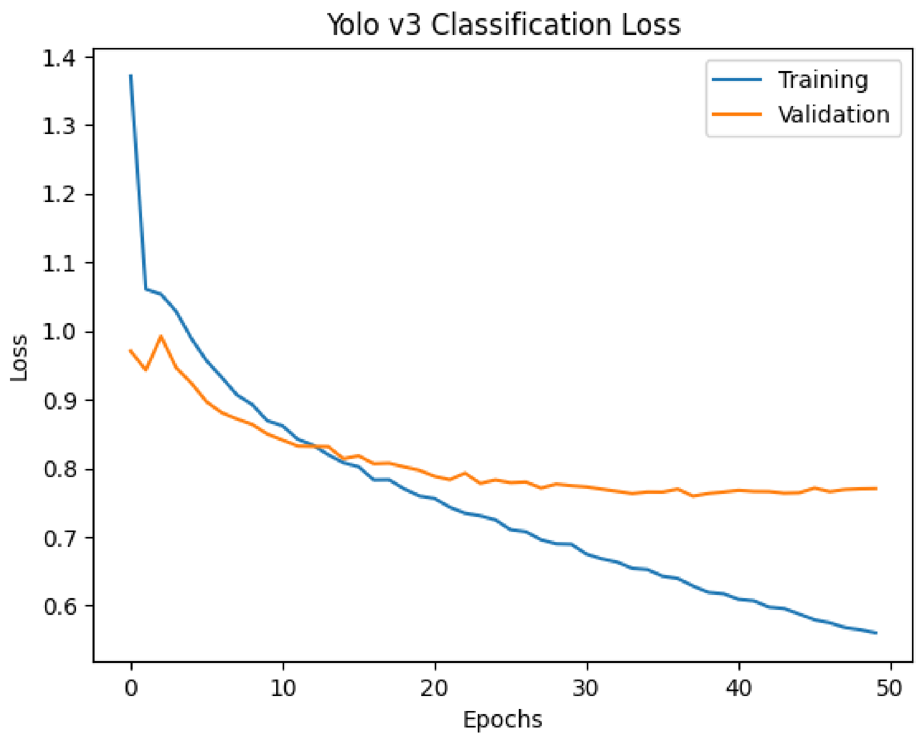


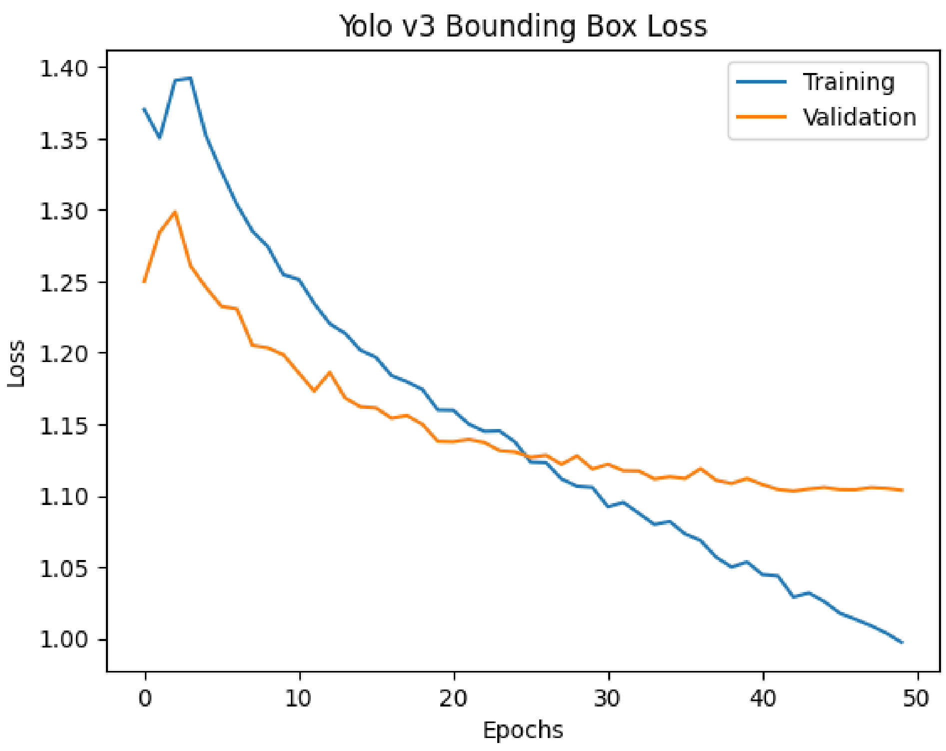
- Restnet50-FPN
- Applied Non-Max suppression
- Trained on 10 epochs, early termination at 5
- MAP score of 0.29

Yolo V3

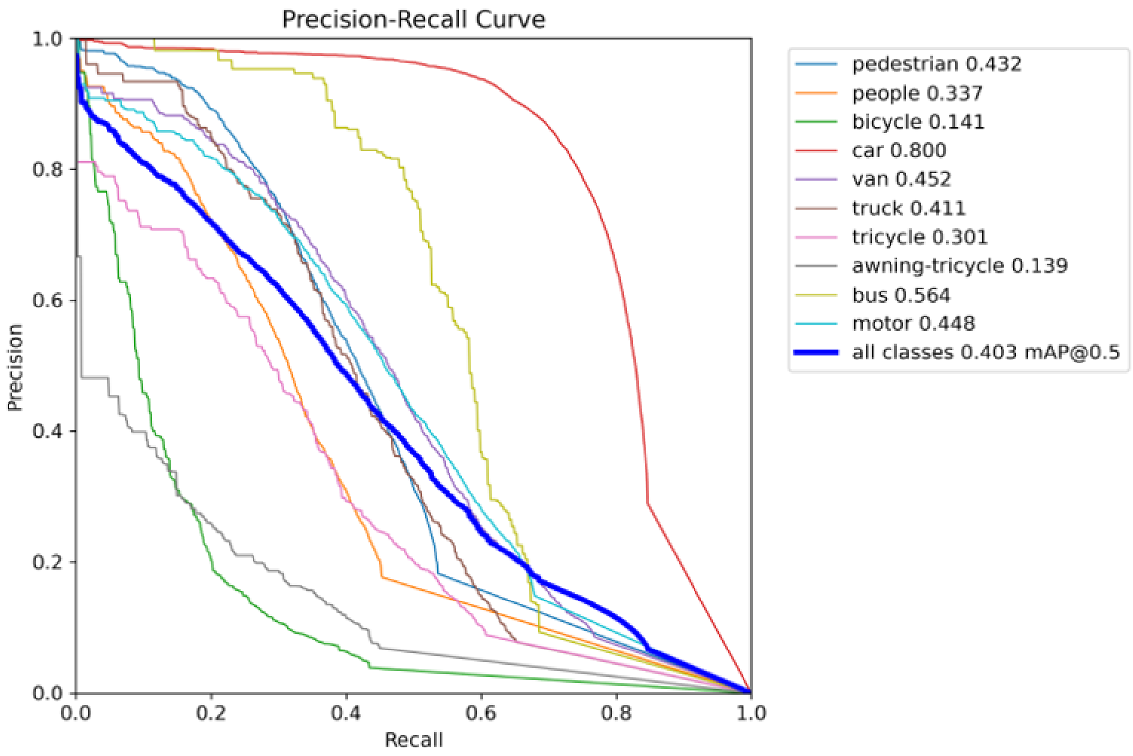
- The YOLOv3 model was used from the Ultralytics implementation and trained with the Visdrone dataset.
- The YOLOv3 model was not originally designed to identify the specific objects of interest or the perspective from which a drone captures objects, so it had to be trained specifically for this purpose.
- The training process involved adjusting the weights and biases of the model through multiple iterations until it could accurately recognize the desired objects in the drone footage.
- Despite the challenges of training the model for this specific use case, the team were confident that YOLOv3 would be effective given its reputation as a state-of-the-art, advanced model that is both efficient and accurate.

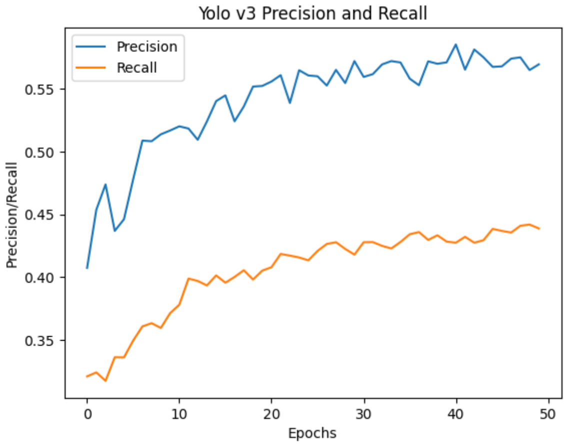
Classification Loss Yolo V3



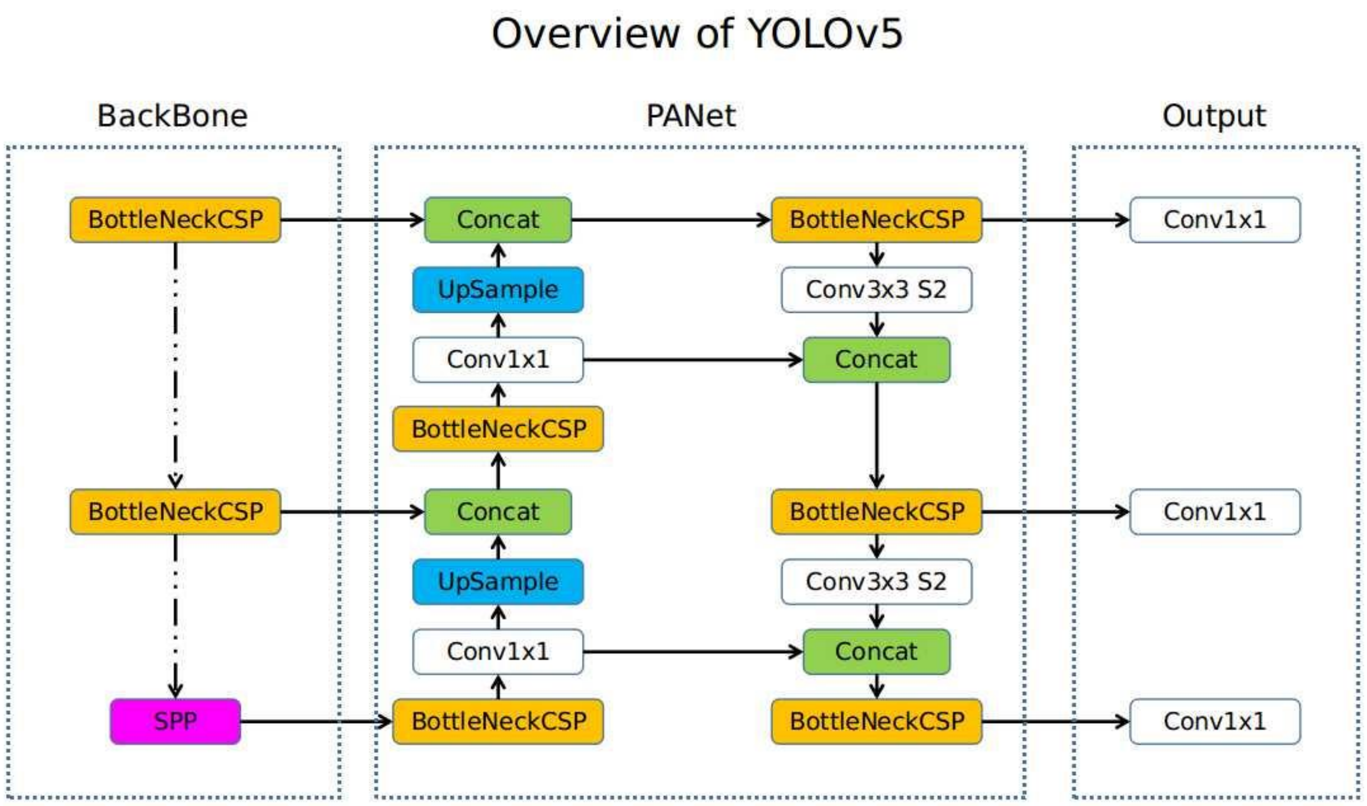


Precision-Recall Curve Yolo V3



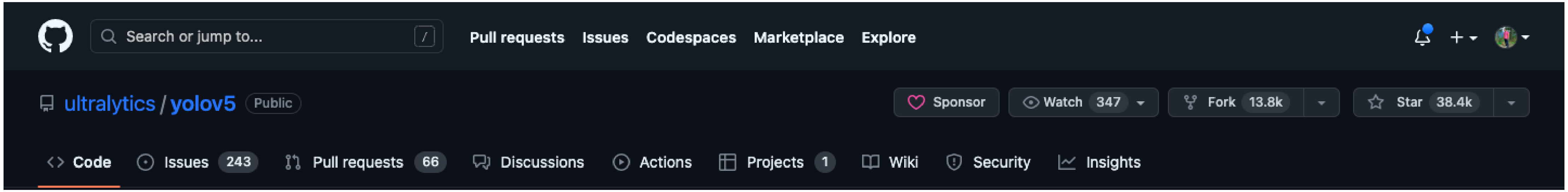


Yolo V5



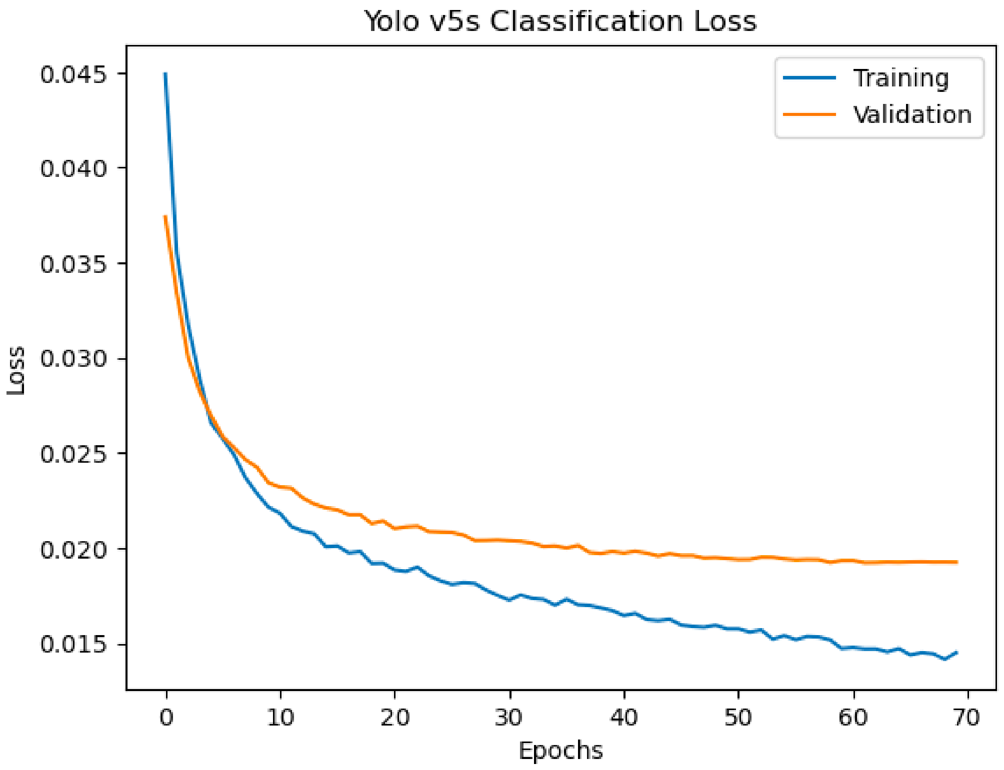
Overview of YOLOv5
BackBone
PANet
Output
BottleNeckCSP
BottleNeckCSP
SPP
Concat
UpSample
Conv1x1
BottleNeckCSP
Concat
UpSample
Conv1x1
BottleNeckCSP
BottleNeckCSP
Conv3x3 S2
Concat
BottleNeckCSP
Conv3x3 S2
Concat
BottleNeckCSP
Conv1x1
Conv1x1
Conv1x1

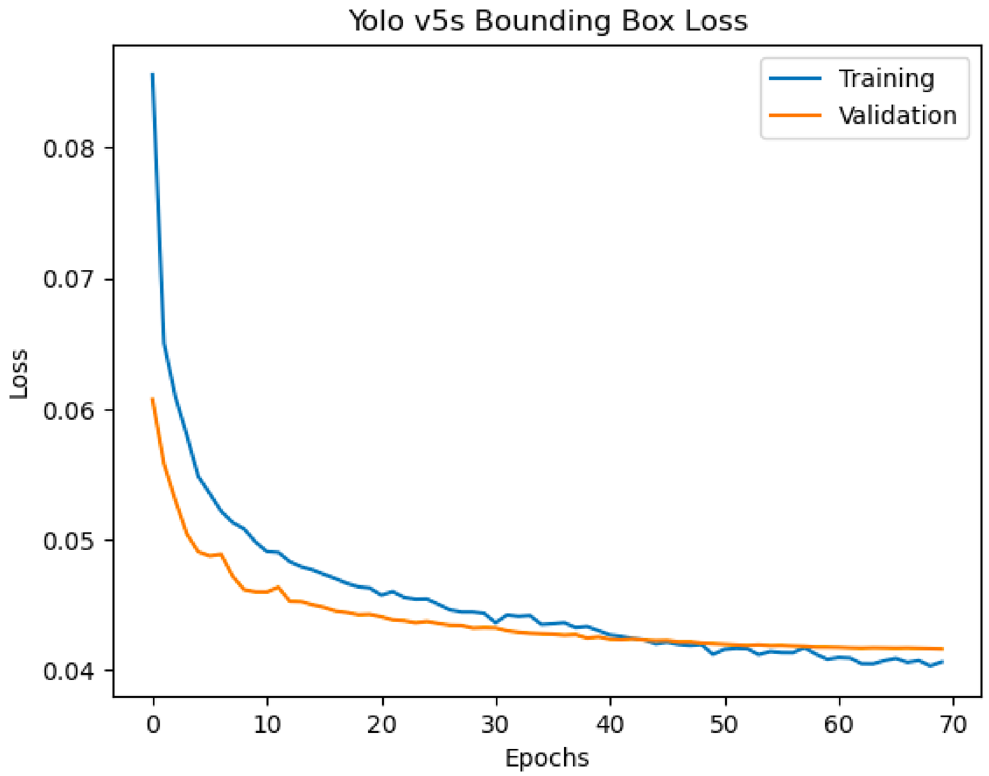
Yolo V5 .contd



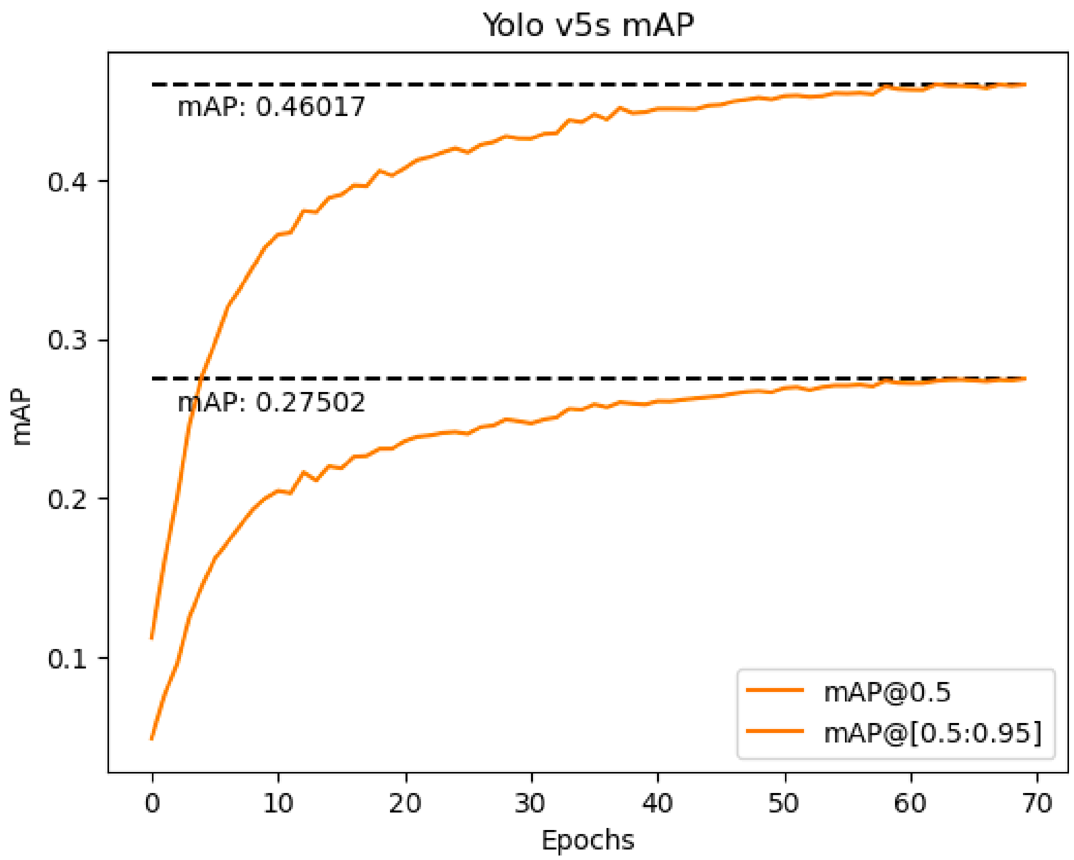

```bash
#!/bin/bash

#SBATCH -J yolotrain            # job name
#SBATCH -p compute              # queue name
#SBATCH -N 1                    # total no. of nodes requested
#SBATCH -n 4                    # total no. of mpi tasks requested
#SBATCH --gres=gpu:4            # number of GPUs per node
#SBATCH -t 24:00:00             # run time (hh:mm:ss)


module load gcc/9.4.0 openmpi/4.0.6/gcc/9.4.0 cuda11.4/toolkit/11.4.2 ucx/1.9.0/gcc/9.4.0

python -m torch.distributed.run --nproc_per_node 4 train.py --img 1024 --batch 8 --epochs 70 \
    --data data/VisDrone.yaml --cfg ./models/yolov5x.yaml --weights yolov5x.pt  --name yolov5x_visdrone  --cache --device 0,1,2,3
```

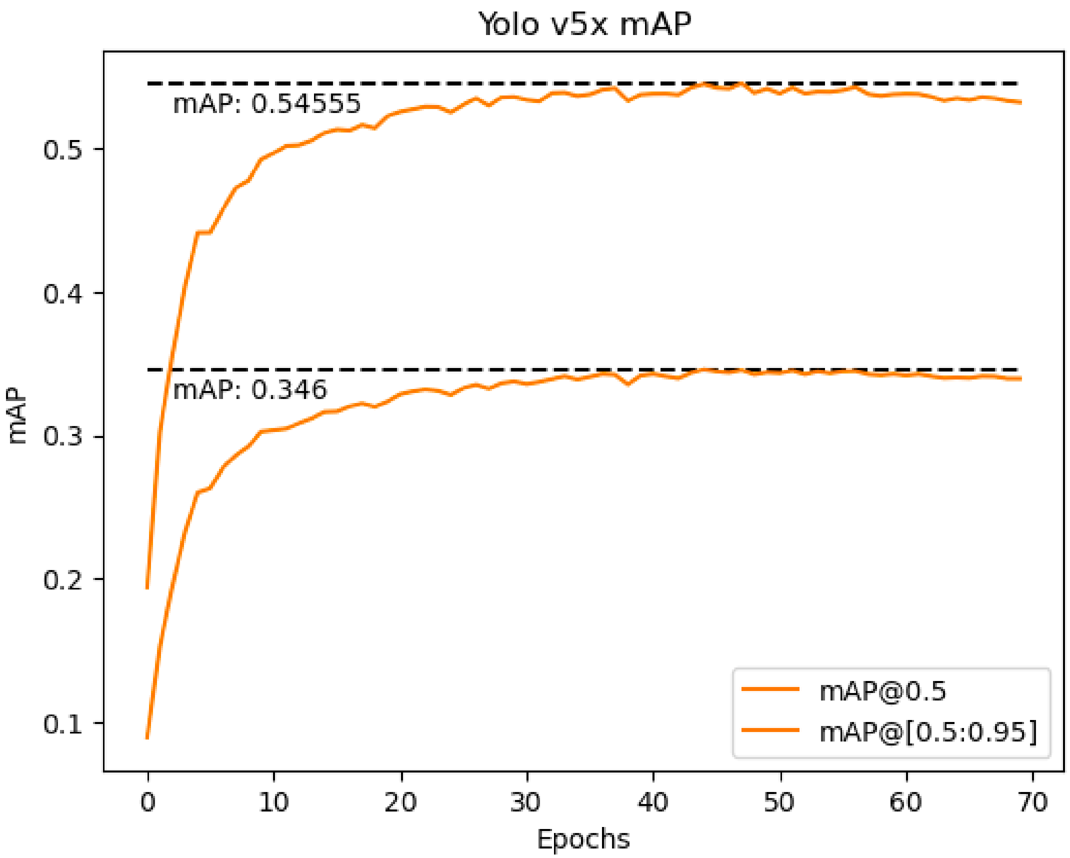
Training Curves



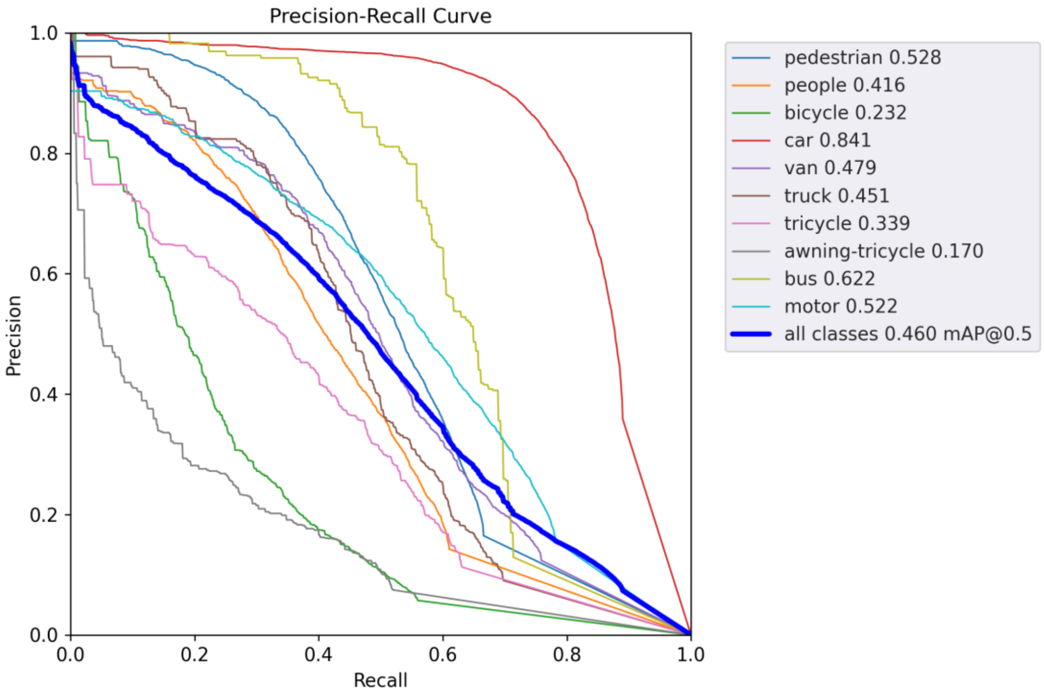


Training mAP Curves

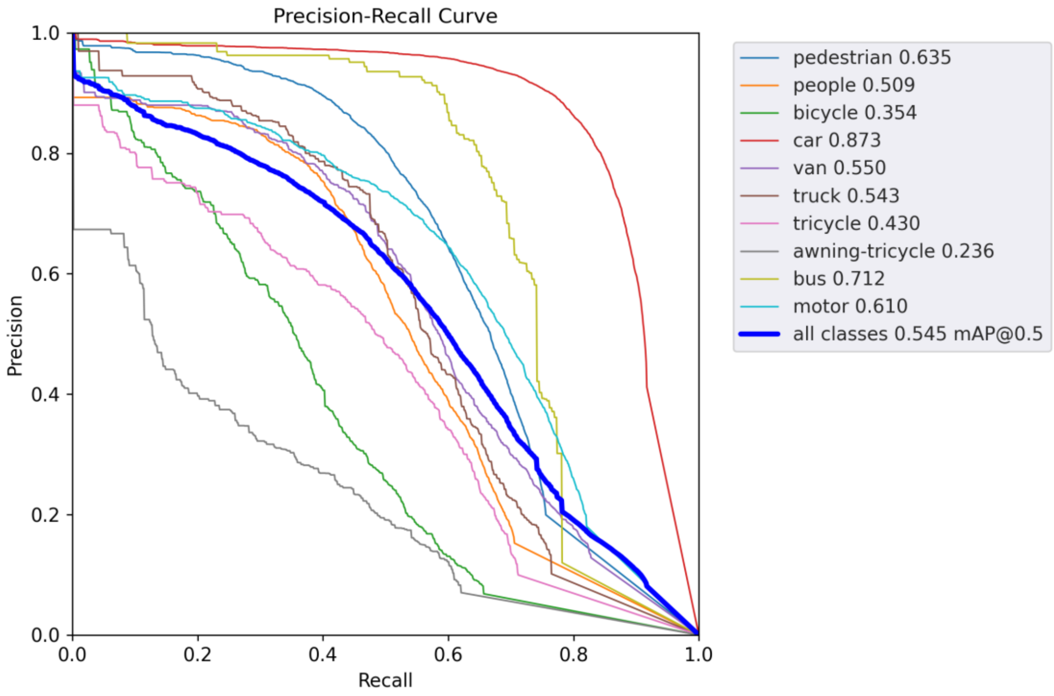




Precision-Recall Curve Yolo V5

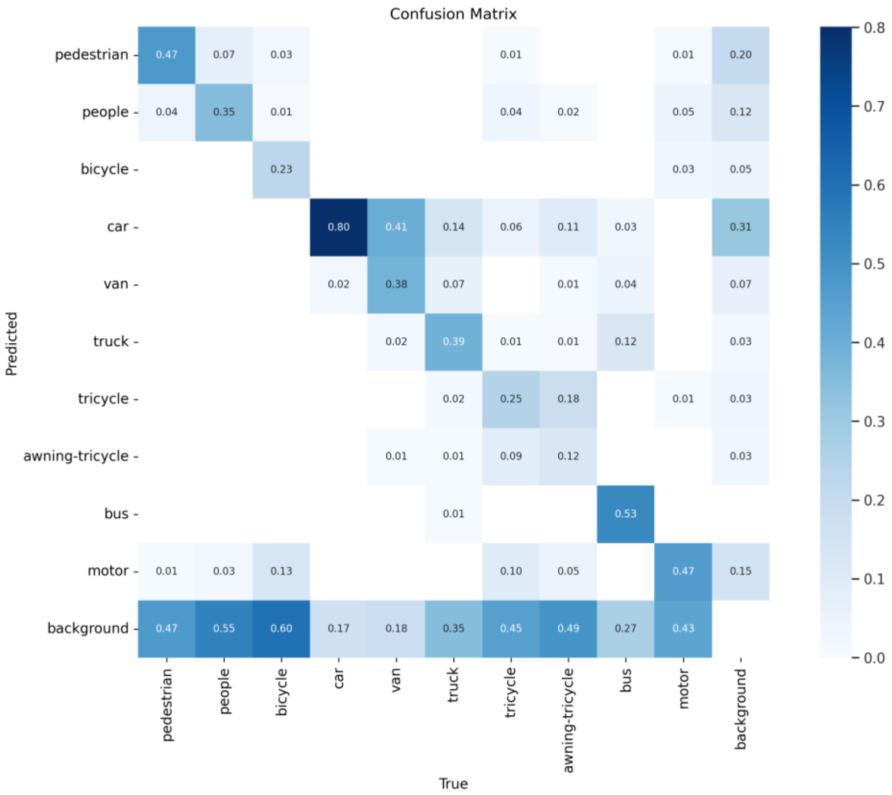


Small Version

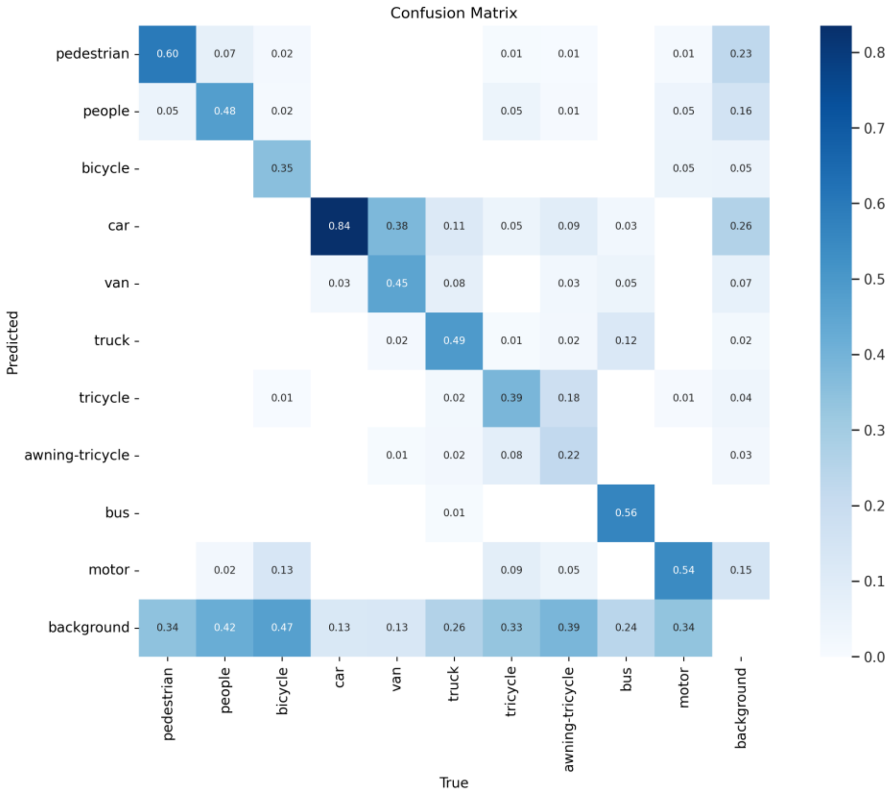


Extra Large Version

Confusion Matrix



Small Version

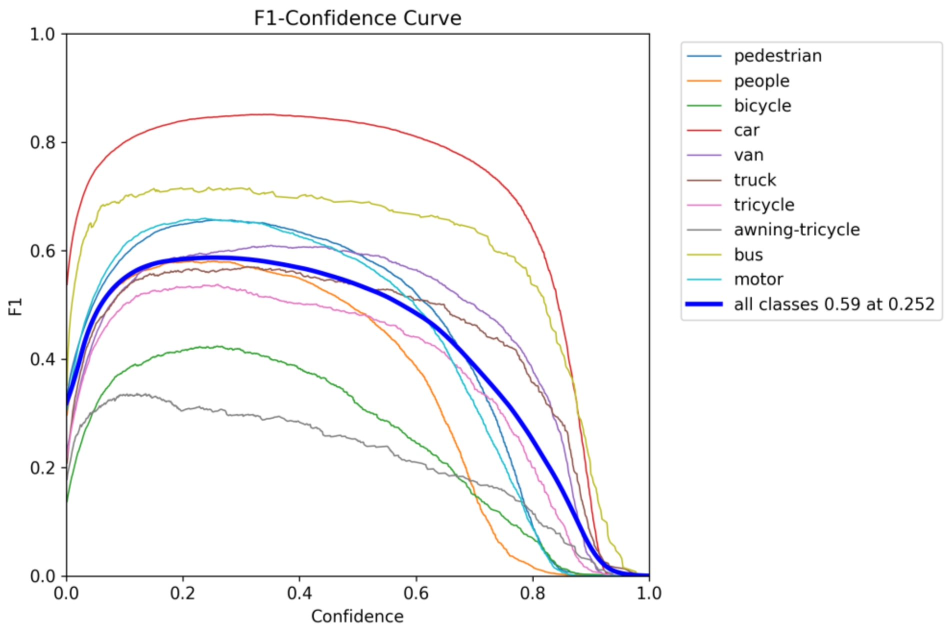


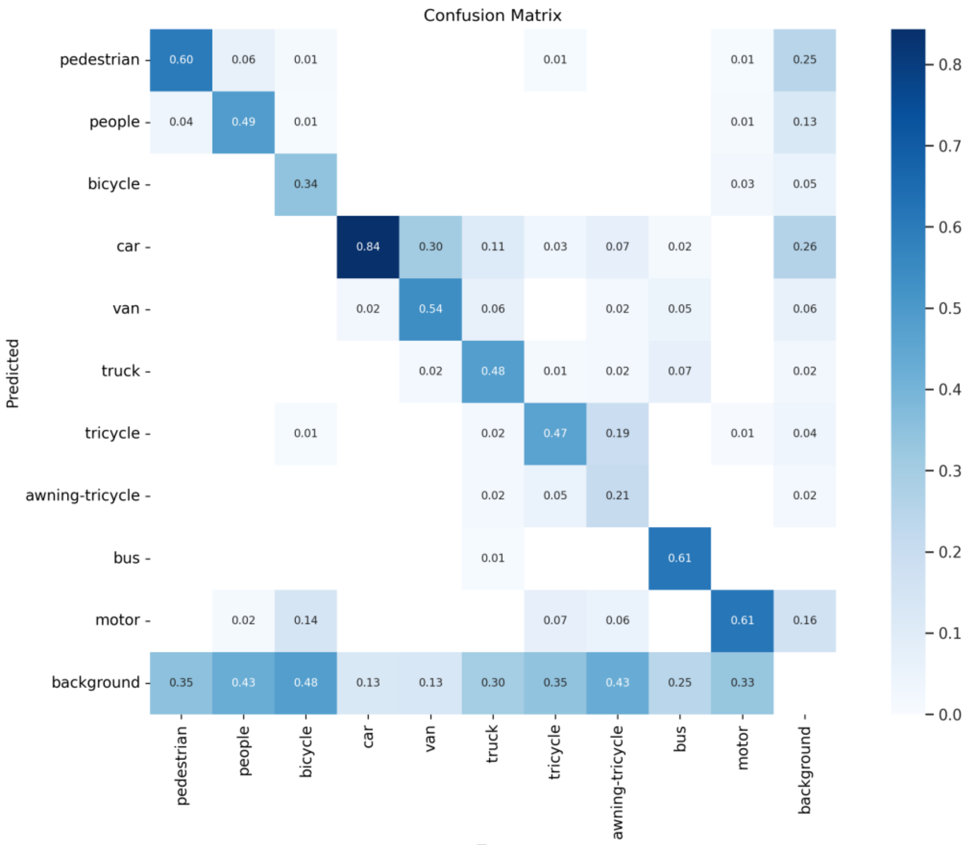
Extra Large Version

Yolo V8

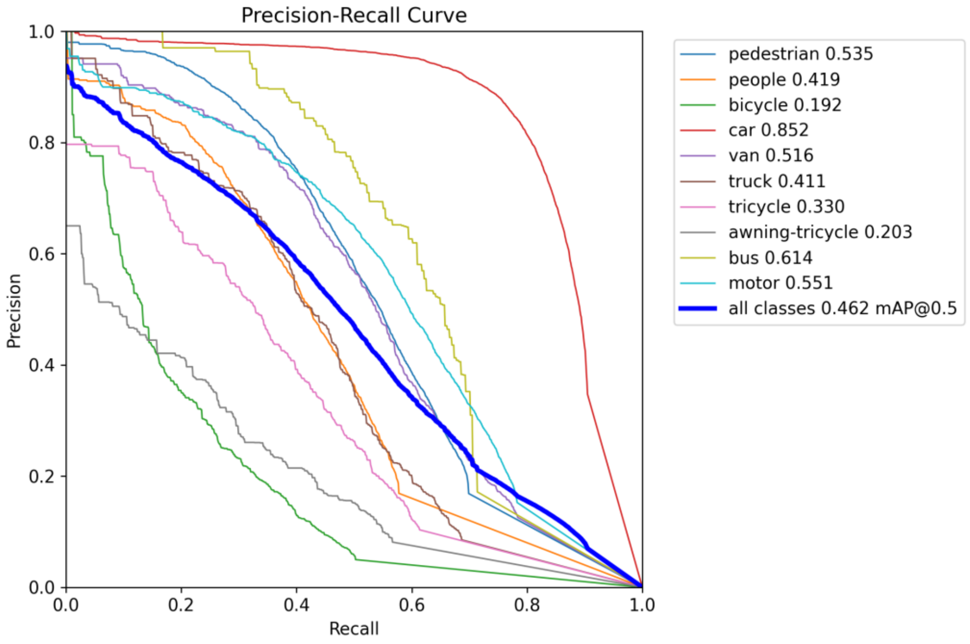
- YOLOv8 was modified to detect 12 classes.
- Limited VRAM capacity (16GB) caused hindrance during training.
- Poor accuracy due to white car, white lines on the road, and light post overlapping with the car.
- Further training could potentially improve the model.
- Extensive experimentation was required to optimize hyperparameters.
- Early stopping was employed to prevent overfitting or diverged training.
- Two versions of the YOLOv8 model were trained: yolov8s (small) and yolov8x (extra-large), which differed in the number of hidden layers, image resolution, and parameters.

Yolo V8 .contd

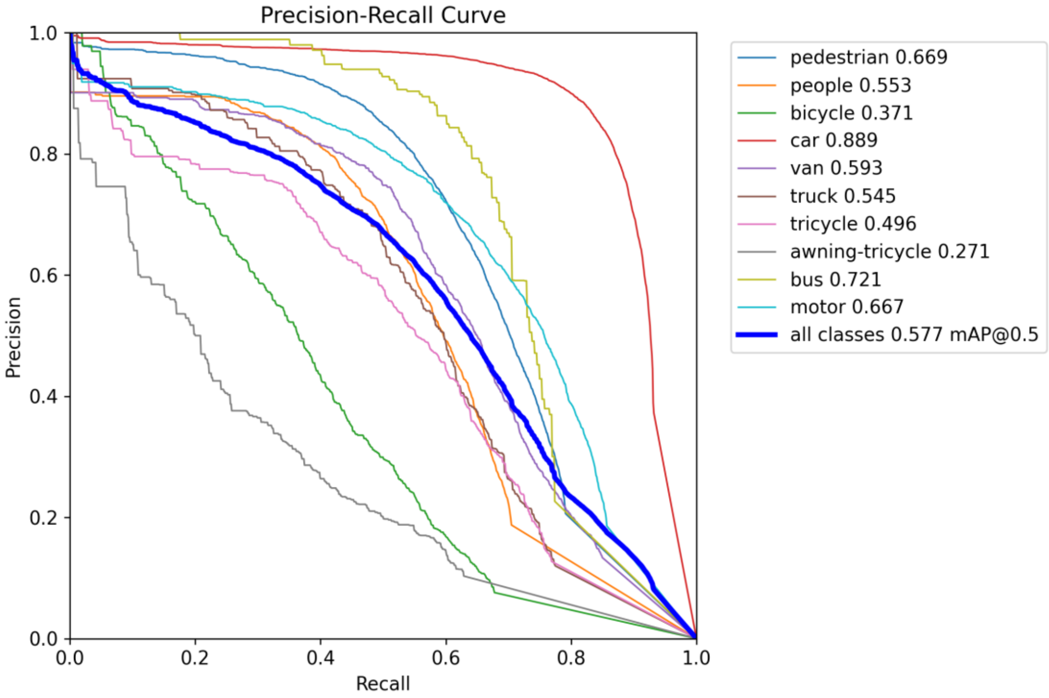




Precision-Recall Curve Yolo V8

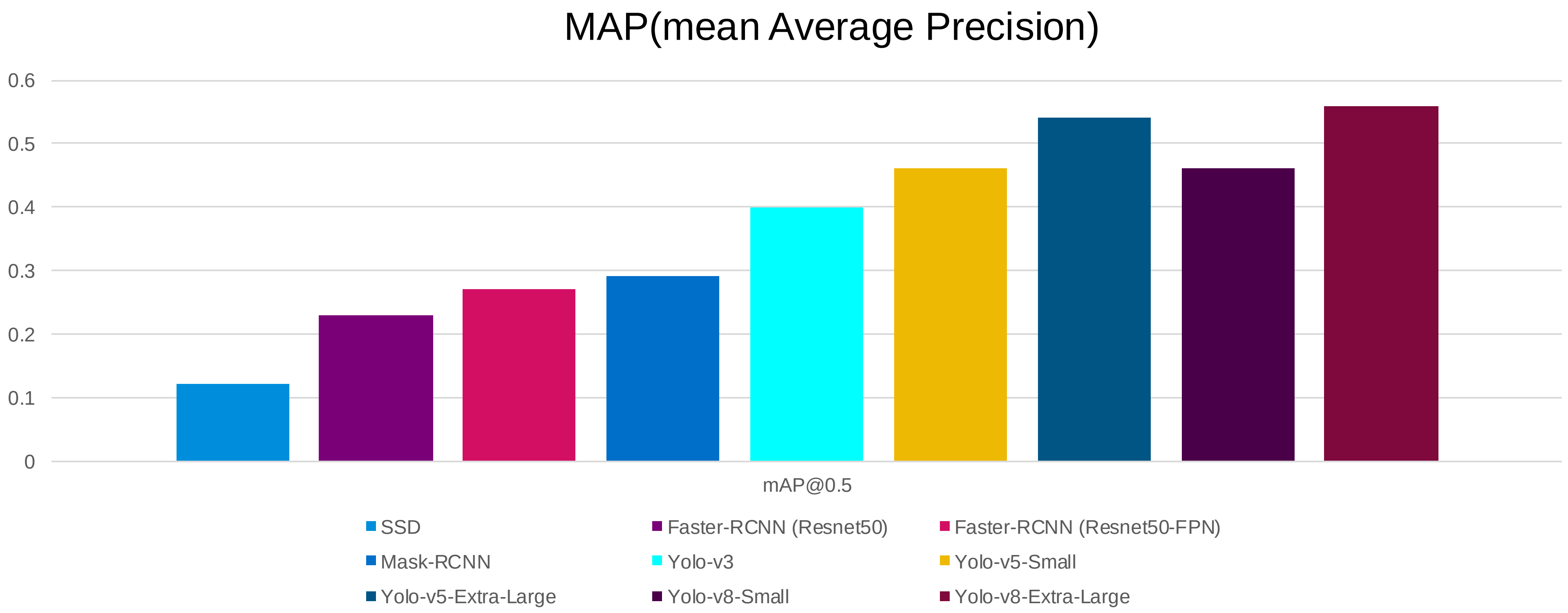


Extra Large Version

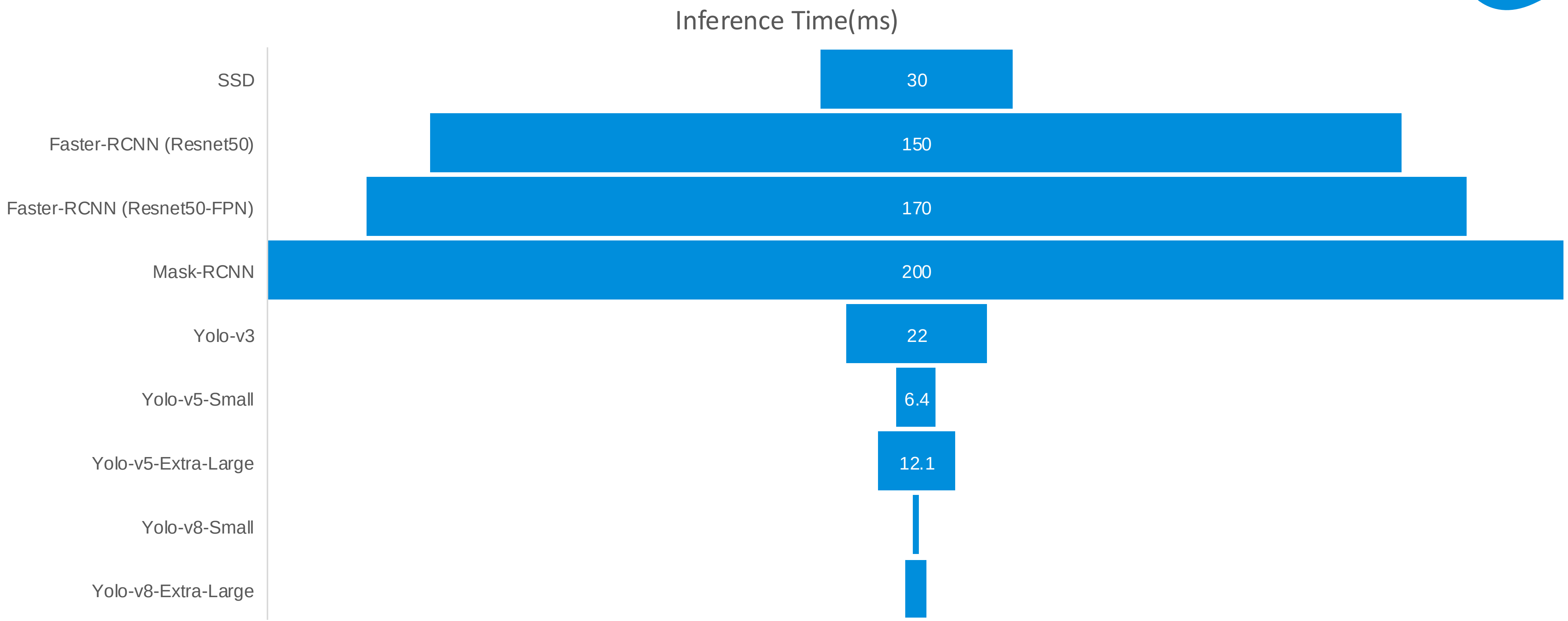


Small Version

mAP


MAP(mean Average Precision)
0.6
0.5
0.4
0.3
0.2
0.1
0
mAP@0.5
SSD
Faster-RCNN (Resnet50)
Faster-RCNN (Resnet50-FPN)
Mask-RCNN
Yolo-v3
Yolo-v5-Small
Yolo-v5-Extra-Large
Yolo-v8-Small
Yolo-v8-Extra-Large

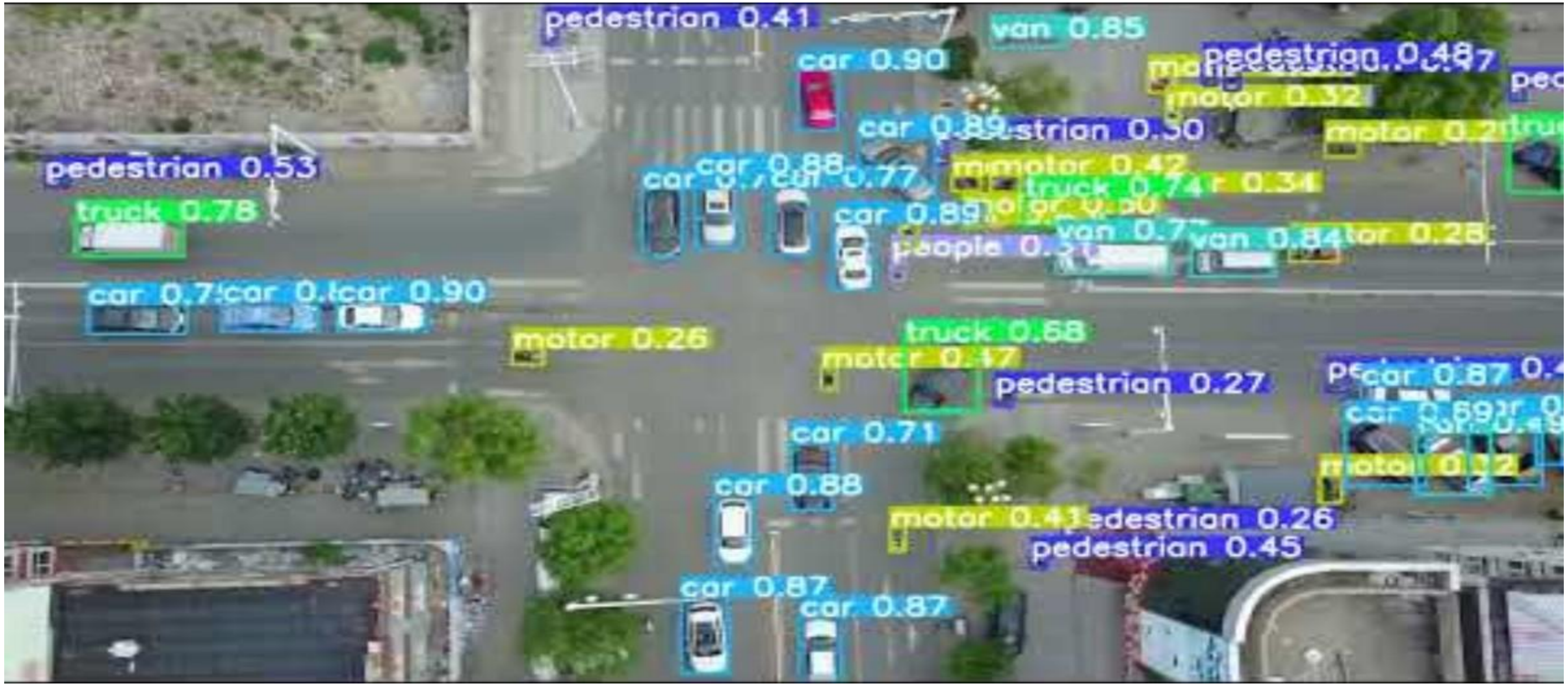
Inference Time



Demo

Moving Object Observation

Moving Object Detection



Future Work

1. Currently the models are been trained with limited computation and experiments (epochs = 10 for Non-Yolos and epochs = 50 for Yolos), targeting to use epochs = 100 for all model and let early Stoppping stop training at global maxima of best.pt.

2. Currently, the dataset is images for training (6000), with a lower resolution of 480, so we target to train it with multifold times 6000 (higher the better), with hd-resolution to achieve better mAP, PR curve and improved loss curve.

3. Currently, we only targeted images as a data set and aspire to use video as the next step for training not just YoloV5 and YoloV8 but also variants of RCNNs.

4. As next step, using ensemble learning could also improve metrics better.

5. Making custom model (via adding more dense layers) (via adding dropout layers) (multi-level transfer learning) on top of base-models, will help us with better predictions.

Conclusion

In conclusion, the project on Object Detection via Drone Surveillance has demonstrated the significant impact and potential of combining drone technology with advanced object detection algorithms.

THANK YOU!

Questions?

Dataset: paperswithcode.com/dataset/visdrone

Github: github.com/nia194/Object-Detection