

Deep Learning Project

Object Detection for Drones

By Smeet Dedhia 200968236

PHASE 1

Problem Statement

Need for the project

Modern drones are equipped with cameras and are very prospective for a variety of commercial uses such as aerial photography, surveillance, etc. In order to massively deploy drones and further reduce their costs, it's necessary to power drones with smart computer vision and autopilot. In the application of aerial photography, object detection and tracking are essential to capturing key objects in a scene. There are significant challenges with drones due to top-down view angles and real-time constraints. Additionally, the strong weight and area constraint of embedded hardware that limits the drones to run computation intensive algorithms, such as deep learning, with limited hardware resource

Existing Models

Generally, we use Object Detection Models like YOLO, SSD Mobilenet & RetinaNet to detect objects.

When we Directly apply previous models to tackle object detection task on drone-captured scenarios we majorly face three problems.

1. First, the object scale varies violently because the flight altitude of drones change greatly.
 2. Second, drone-captured images contain objects with high density, which brings in blockage between objects.
 3. Third, drone-captured images always contain confusing geographic elements because of covering large area.
- These problems make the object detection of drone-captured images very challenging.

Our Solution

We will train the existing Models using object-detection datasets (visDrone Dataset) that are specific to drones so that we can improve the accuracy of these models.

Steps & Workflow:

1. Load existing weights for SSD MobileNet, YOLO and RetinaNet.
2. Train these 3 models on the visDrone dataset.
3. Evaluate performance of all models on actual drone footage
4. Compare the models in terms of accuracy, speed and memory.

About the Dataset

VisDrone is a large-scale benchmark with carefully annotated ground-truth for various important computer vision tasks, to make vision meet drones. The VisDrone2019 dataset is collected by the AISKYEYE team at Lab of Machine Learning and Data Mining, Tianjin University, China. The benchmark dataset consists of 288 video clips formed by 261,908 frames and 10,209 static images, captured by various drone-mounted cameras, covering a wide range of aspects including location (taken from 14 different cities separated by thousands of kilometers in China), environment (urban and country), objects (pedestrian, vehicles, bicycles, etc.), and density (sparse and crowded scenes). Note that, the dataset was collected using various drone platforms (i.e., drones with different models), in different scenarios, and under various weather and lighting conditions. These frames are manually annotated with more than 2.6 million bounding boxes of targets of frequent interests, such as pedestrians, cars, bicycles, and

tricycles. Some important attributes including scene visibility, object class and occlusion, are also provided for better data utilization.

Annotation Format

bbox_left, bbox_top, bbox_width, bbox_height, score, object_category, truncation, occlusion

bbox_left

The x coordinate of the top-left corner of the predicted bounding box

bbox_top

The y coordinate of the top-left corner of the predicted object bounding box

bbox_width

The width in pixels of the predicted object bounding box

bbox_height

The height in pixels of the predicted object bounding box

Score

the score in the DETECTION file indicates the confidence of the predicted bounding box enclosing an object instance. The score in GROUNDTRUTH file is set to 1 or 0.

- 1 indicates the bounding box is considered in evaluation.
- 0 indicates the bounding box will be ignored.

Object Category

The object category indicates the type of annotated object:

- ignored regions(0)
- pedestrian(1)
- people(2)
- bicycle(3)
- car(4)
- van(5)
- truck(6)
- tricycle(7)
- awning-tricycle(8)
- bus(9)
- motor(10)
- others(11)

Truncation

The score in the DETECTION result file should be set to the constant -1.

The score in the GROUNDTRUTH file indicates the degree of object parts appears outside a frame:

- no truncation = 0 (truncation ratio 0%)
- partial truncation = 1 (truncation ratio 1% ~ 50%)

Occlusion

The score in the DETECTION file should be set to the constant -1.

The score in the GROUNDTRUTH file indicates the fraction of objects being occluded

- no occlusion = 0 (occlusion ratio 0%)
- partial occlusion = 1 (occlusion ratio 1% ~ 50%)
- occlusion = 2 (occlusion ratio 50% ~ 100%)

Exploratory Analysis

- Ten object categories: pedestrian, person, car, van, bus, truck, motor, bicycle, awning tricycle, and tricycle
- Images are manually annotated with more than 2.6 million bounding boxes of targets of frequent interest.
- 288 video clips formed by 261,908 frames and 10,209 static images, captured by various drone-mounted cameras

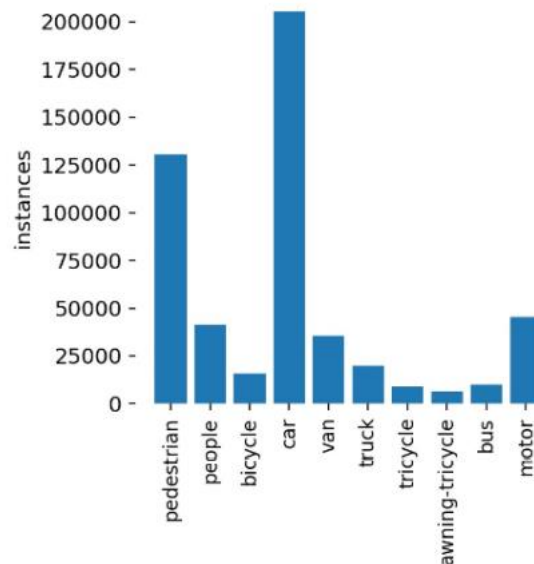


Figure 8. The number of labels of each category.

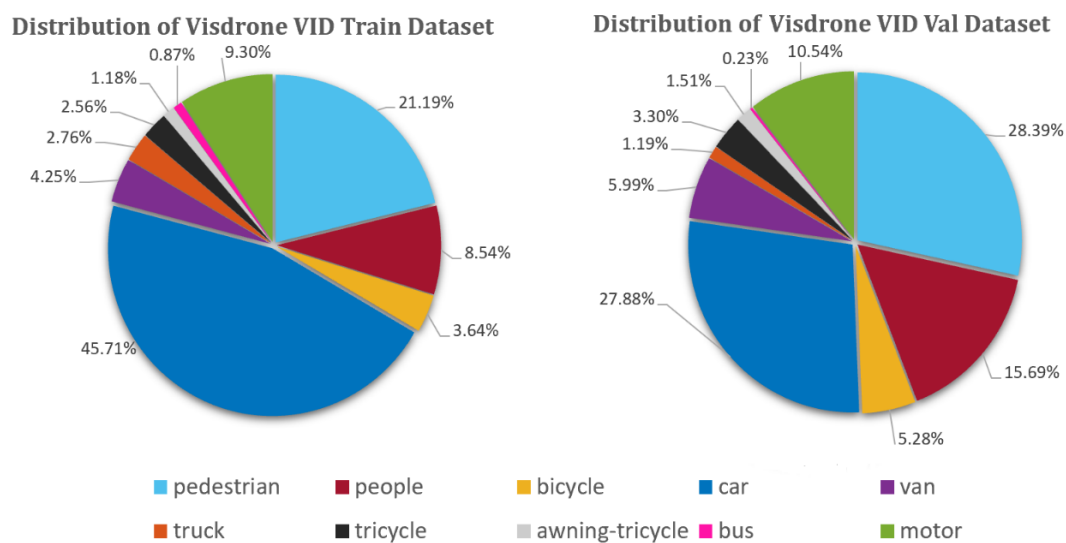


Figure 4. The class distribution in Visdrone VID-Train, Val Dataset.

Sample 1st Image:

Annotations for 1st image:

684,8,273,116,0,0,0,0
 406,119,265,70,0,0,0,0
 255,22,119,128,0,0,0,0
 1,3,209,78,0,0,0,0
 708,471,74,33,1,4,0,1

1st Image:



1st Image with some annotations plotted



Pre-processing

1. Firstly, we need to convert the annotations from the custom format to the standard PASCAL VOC format.
2. For YOLO model, we convert PASCAL VOC to YOLO annotation format.
3. For SSD MobileNet, we convert PASCAL VOC to Tensorflow Record format.

This prepares our dataset for training the three models.

Refer to preprocessing python notebook for detailed code,

Objectives

For the three models- SSD Mobilenet, YOLO and RetinaNet:

1. Improve the accuracy of all models by training on drone specific dataset.
2. Compare accuracy of all the models.
3. Compare speed of the models.
4. Compare memory footprint of all the models.

Evaluation Metrics

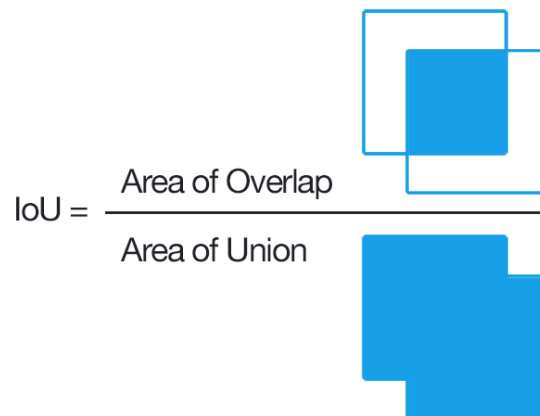
The performance of algorithms is evaluated by the average precision (AP) across different object categories and intersection over union (IoU) thresholds. Specifically, AP is computed by averaging over all 10 IoU thresholds, i.e., in the range [0.50 : 0.95] with uniform step size 0.05 of all categories.

IoU

Intersection over Union (IoU) is used when calculating mAP. It is a number from 0 to 1 that specifies the amount of overlap between the predicted and ground truth bounding box.

- an IoU of 0 means that there is no overlap between the boxes
- an IoU of 1 means that the union of the boxes is the same as their overlap indicating that they are completely overlapping

IoU is an important accuracy measure to track when gathering human annotations. The industry best practice is to include a minimum IoU requirement for their human annotation tasks, to ensure that the annotations that are delivered have an IoU $\geq X$ (where $X = 0.95$ is typical) with respect to the “perfect” annotation of that object, as determined by the annotation schema for the project (i.e. box vehicles as tight as possible, including all visible parts of them, including the wheels). State of the art detectors, on the other hand, typically do not perform at a 0.95 IoU


$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

mAP in short

Mean average precision (mAP) is calculated by first gathering a set of predicted object detections and a set of ground truth object annotations.

- For each prediction, IoU is computed with respect to each ground truth box in the image.
- These IoUs are then thresholded to some value (generally between 0.5 and 0.95) and predictions are matched with ground truth boxes using a greedy strategy (i.e. highest IoUs are matched first).
- A precision-recall (PR) curve is then generated for each object class and the average precision (AP) is computed. A PR curve takes into account the performance of a model with respect to true positives, false positives, and false negatives over a range of confidence values.
- The mean of the AP of all object classes is the mAP.
- When evaluating the COCO dataset, this is repeated with 10 IoU thresholds from 0.5 to 0.95 and averaged.

PHASE 2.1

Literature Review

S.No.	Author	Type of Paper	Title & Link	Models	Pros & Cons
1	Subrahmanyam Vaddi	Thesis	Efficient Object Detection	RCNN	Pros: First detector which proposed the two step detection approach. Cons: has a complex multi-step training pipeline which requires careful tweaking of parameters quite inefficient as it has overhead of running every region proposal through the convolutional base.
				Fast RCNN	Pros: ROI algorithm reduces overhead by running the convolutional base just once over the entire image to generate a feature map. It crops the regions from this generated feature map instead of the entire input image thus reducing both the time and space. Fast RCNN is more speed and memory efficient compared to RCNN introduces a simpler single step training pipeline and a new loss function Cons: it still suffers with an overhead of region proposal step done by selective search algorithm, taking lot of time to generate proposals
				Faster RCNN	Pros: Faster RCNN introduced the Region Proposal Network (RPN) to replace Selective Search, which have the capability of predicting regions of multiple scales and aspect ratios across the image using concept called anchors Cons: Scale invariance was one of the problems not dealt by Faster RCNN. The model should be able to detect the object up close and also from far way

				FPN	Pros: FPN model solves the problem Of scale invariance by creating multiple feature maps that are designed to represent the image at different scales. This is done using lateral scales.
				RetinaNet	Pros: RetinaNet employs a Feature Pyramid Network (FPN) built on top of ResNet as backbone. RetinaNet solved the problem of class imbalance with their novel loss function and boasts the state of the art results in accuracy and speed compared to all other detectors. Cons: Even though RetinaNet performed well on ImageNet dataset, it had major drawbacks in performance on aerial image dataset because of presence of very small scaled objects in aerial images.
				DFPN	Pros: To avoid class imbalance, it used a dynamic loss function, which down weights the loss contributed by easily classified examples. This loss function can automatically down weight the contribution of easy examples during training and rapidly focus the model on hard examples.
2	Song Han, William Shen, Zuozen Liu		Deep Drone	Modified Faster RCNN	Pros: model has slightly worse accuracy than the baseline, but has 12x faster speed. Cons: Only trained to detect people.
3	Meeth Shetty		YOLO vs SSD	YOLO	Pros: YOLO detection model is more targeted for real time performance. YOLO trains on entire images instead of regions and directly optimizes detection performance. Bounding boxes as well as class probabilities of these bounding boxes are predicted by a single convolution network Training of YOLO is based on a loss function which corresponds directly to detection performance and it

					<p>trains the entire model jointly</p> <p>Cons: Although YOLO is close to real time, it has spatial limitations on bounding box predictions as each grid cell predicts only two bounding boxes and can have only one category.</p> <p>it does not perform well on small objects and those appear in groups</p> <p>YOLO while being fast was less accurate</p>
				SSD	<p>Pros: SSD adds several feature layers to the end of the network, which is responsible for predicting the offsets to default boxes with different scales (i.e., sizes) and aspect ratios and their associated confidences</p> <p>SSD512 model significantly outperforms the state-of-the-art Faster R-CNN in terms of accuracy on PASCAL VOC and COCO, while being 3× faster</p> <p>Real-time SSD300 model runs at 59 FPS, which is faster than the current realtime YOLO alternative while producing markedly superior detection accuracy</p> <p>Cons: Not suitable for small objects.</p>
4	P Zhu, L Wen, D Du, X Bian, H Ling		VisDrone DET 2021		

Shortlisted Models & Comparison

Considering all the models and their pros-cons, we select **YOLO**, **SSD** & **RetinaNet** as candidate models for this project.

These models are ‘Single Step Detectors’ and hence are optimized for speed and memory.

We will conduct a comparative study between all three models and evaluate their performance on aerial images dataset – VisDrone.

Research papers show that RetinaNet is the best of the three, while SSD closely edges out YOLO.

Parameters	YOLO	SSD	RetinaNet
Backbone	GoogleNet	VGG16	ResNet
Learning Method	SGD	SGD	SGD
Input Resolution	448*448	300*300	224*224
Language	C	C++	C++

Platform	DarkNet	Caffe	Caffe
----------	---------	-------	-------

I will be training RetinaNet.

Nikhith (200968044) from DSE-A will train SSD and Siddharth (200968016) from DSE-A will training YOLO.
At the end of the project, we will compare performance of all three models.

Baseline Model

The baseline model for all YOLO, SSD & RetinaNet is Convolution Network.
All the three models use state of the art image feature extractors like VGG-16 and Inception.

PHASE 2.2

Submitted as 4 Jupyter Notebook files:

1. YOLO_training
2. RetinaNet training
3. SSD_training
4. Model_Comparisons

PHASE 3

Summary of Models used:

Model 1: SSD MobileNet V2

- Trained on: MS COCO dataset.
- Object Classes: 90 (Which includes our 12 target classes)

This model is used for detection as is, without any additional training or transfer learning.
Hence, this will give us an idea of how models trained on generic object detection datasets like MS COCO perform on Aerial Images.

Model 2: RetinaNet

- Trained on: VisDrone Dataset of Aerial Images
- Object Classes: 12

This model was trained on Google Colab and Local System under time and performance constraints.
Due to these constraints, the model was trained only for 25 epochs, which is not sufficient.
Hence, this will give us an idea of how a self trained model would perform on Aerial Images

Model 3: TPH YOLOv5

- Trained on: VisDrone Dataset of Aerial Images
- Object Classes: 12

This model was trained professionally on the VisDrone dataset with no constraints.
This was model was ranked 5th in the VisDroneDET 2022 challenge.
Hence, this will give us an idea of how a State of the Art model would perform on Aerial Images.

Comparison of mean Average Precision (mAP)

Mean average precision (mAP) is calculated by first gathering a set of predicted object detections and a set of ground truth object annotations.

- For each prediction, IoU is computed with respect to each ground truth box in the image.
- These IoUs are then thresholded to some value (generally between 0.5 and 0.95) and predictions are matched with ground truth boxes using a greedy strategy
- The mean of the AP of all object classes is the mAP.

Overall mAP:

Model 1- SSD MobileNet V2: 0.202

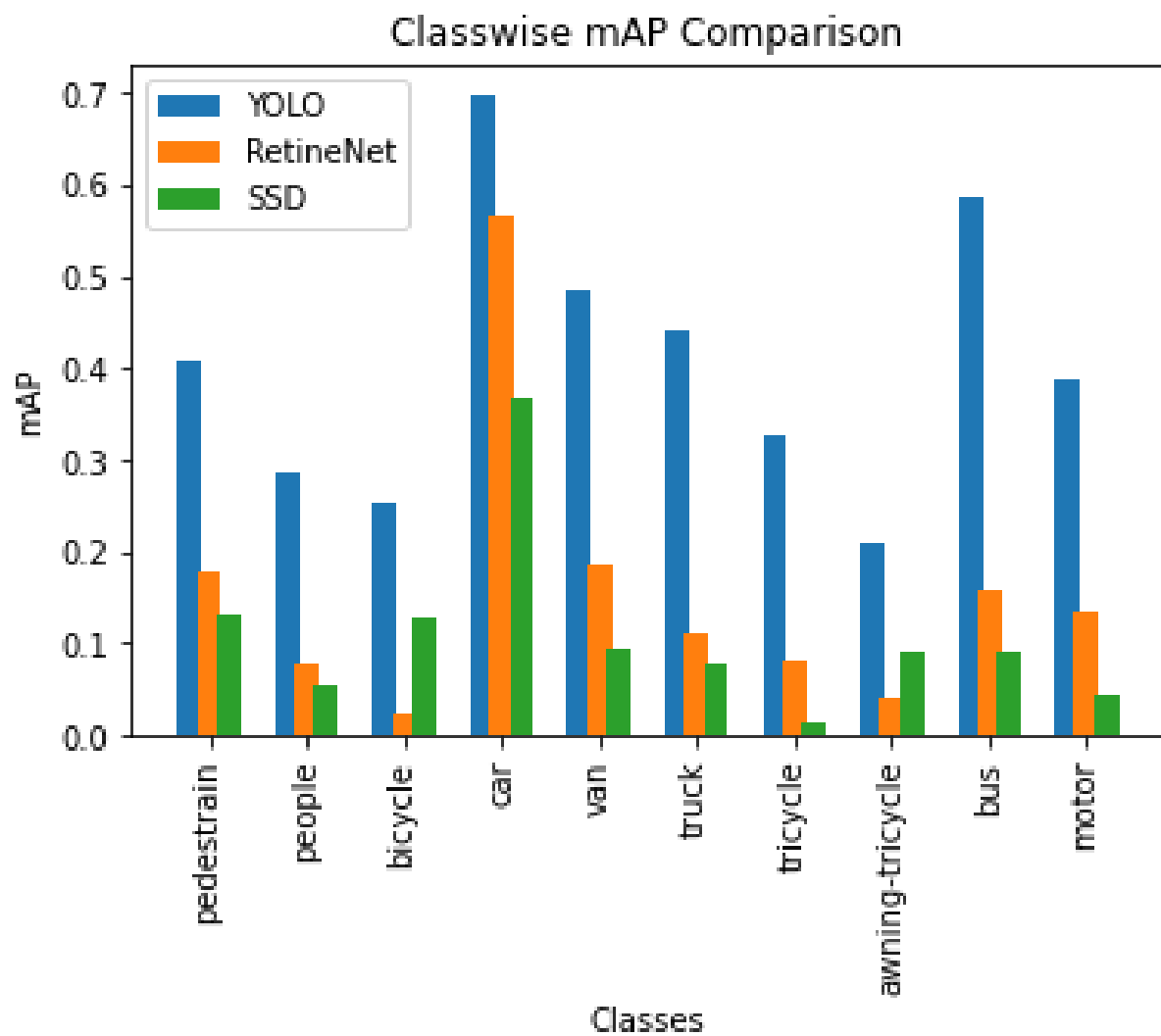
Model 2- RetinaNet: 0.2797

Model 3- TPH YOLOv5: 0.408

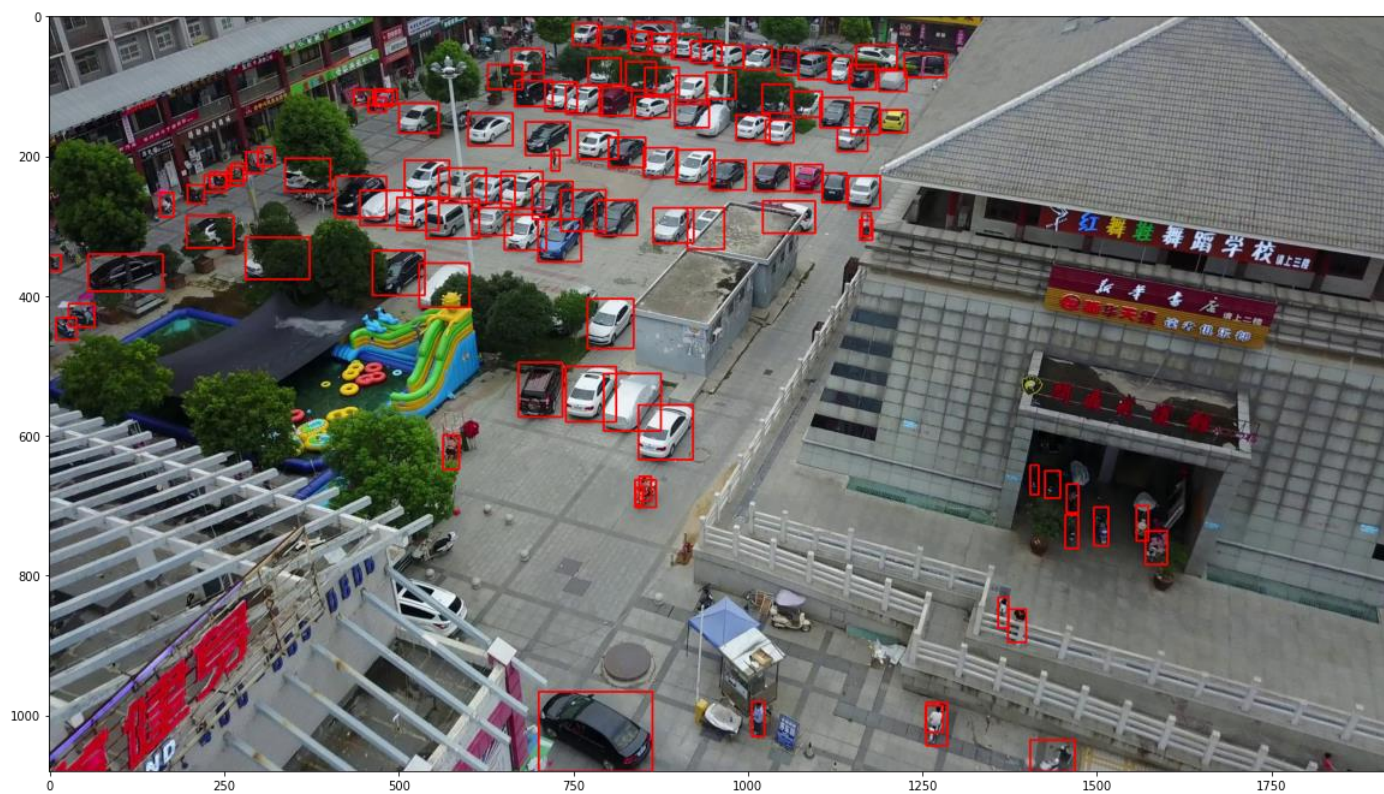
Class-wise mAP:



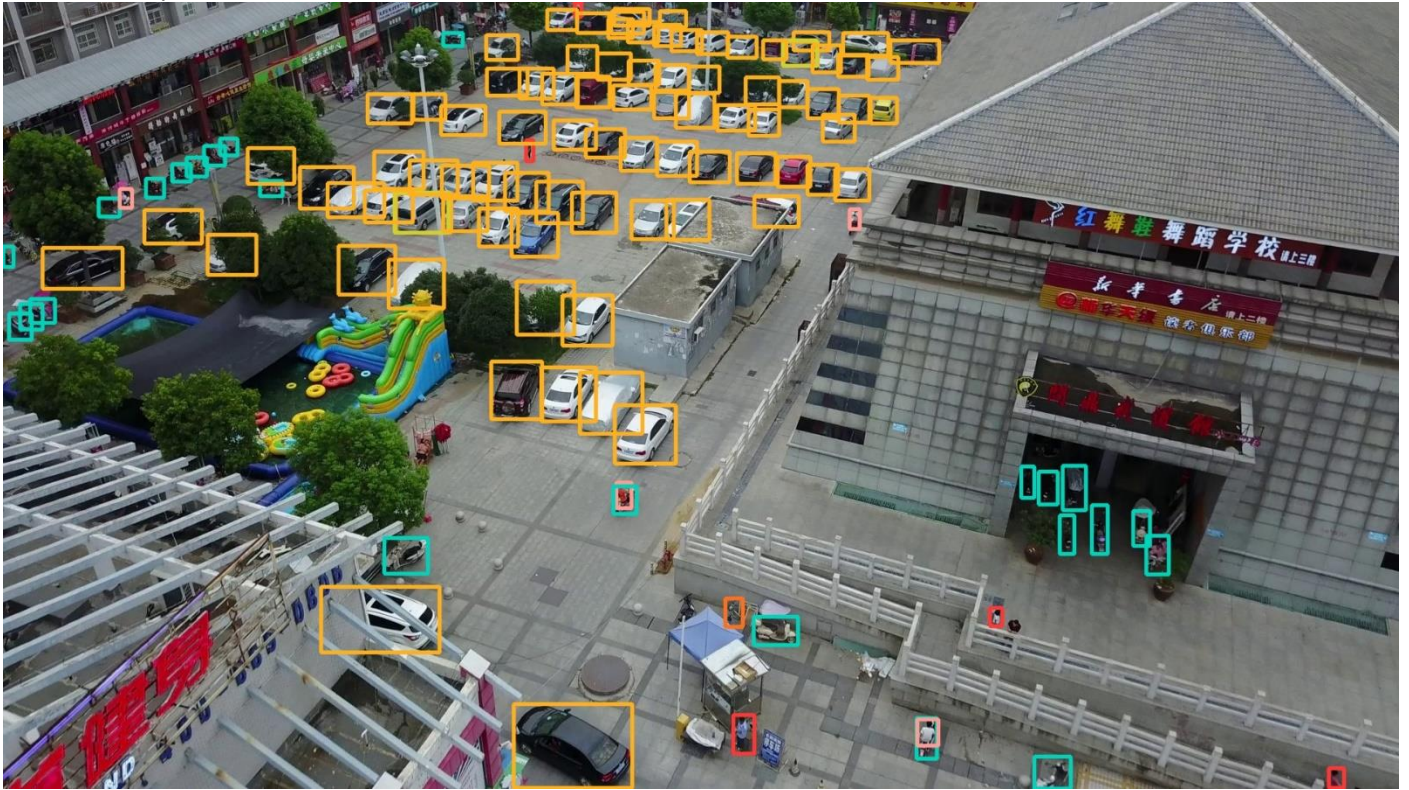
Overall Comparison of Models:



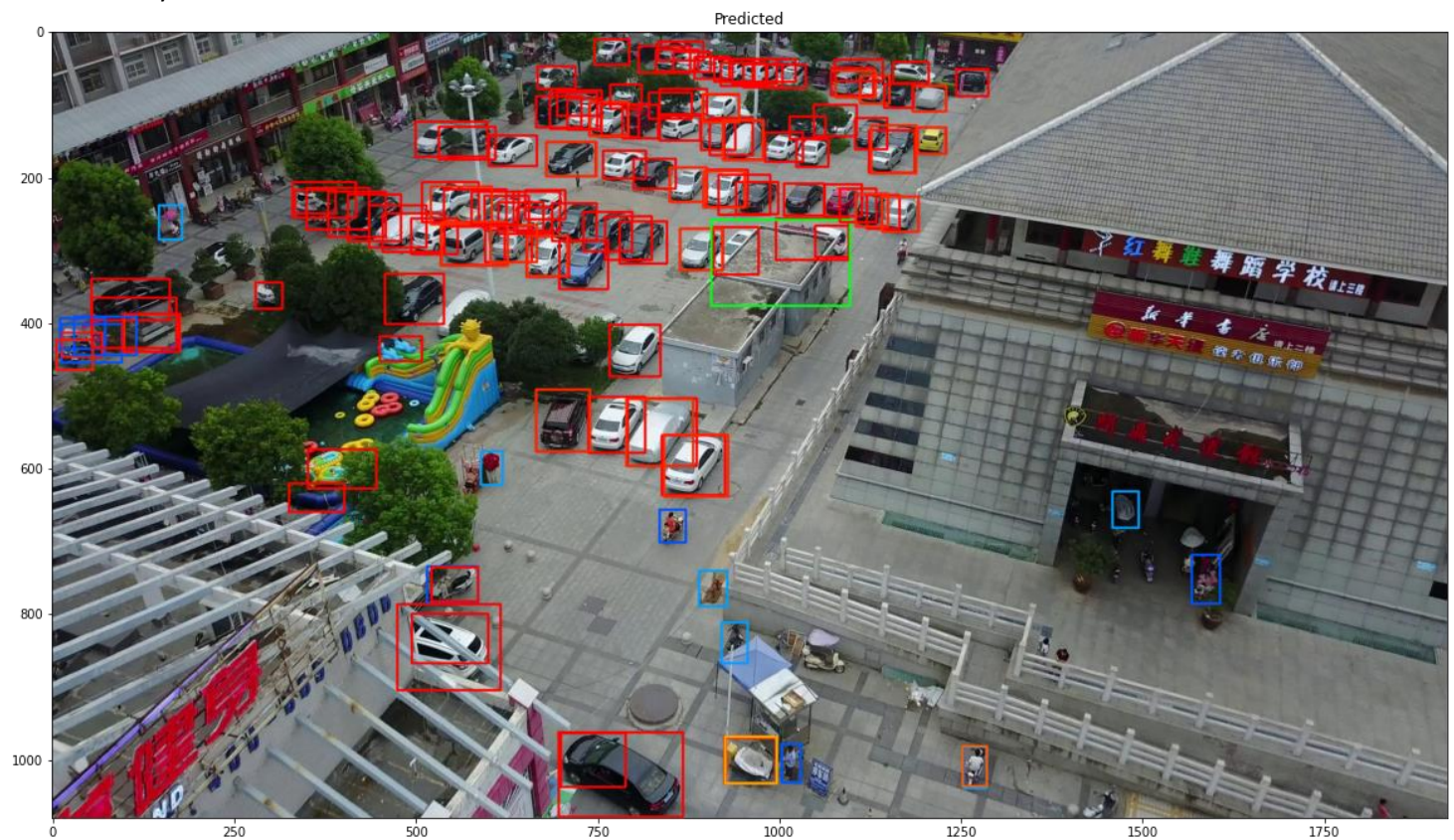
Actual Annotations:



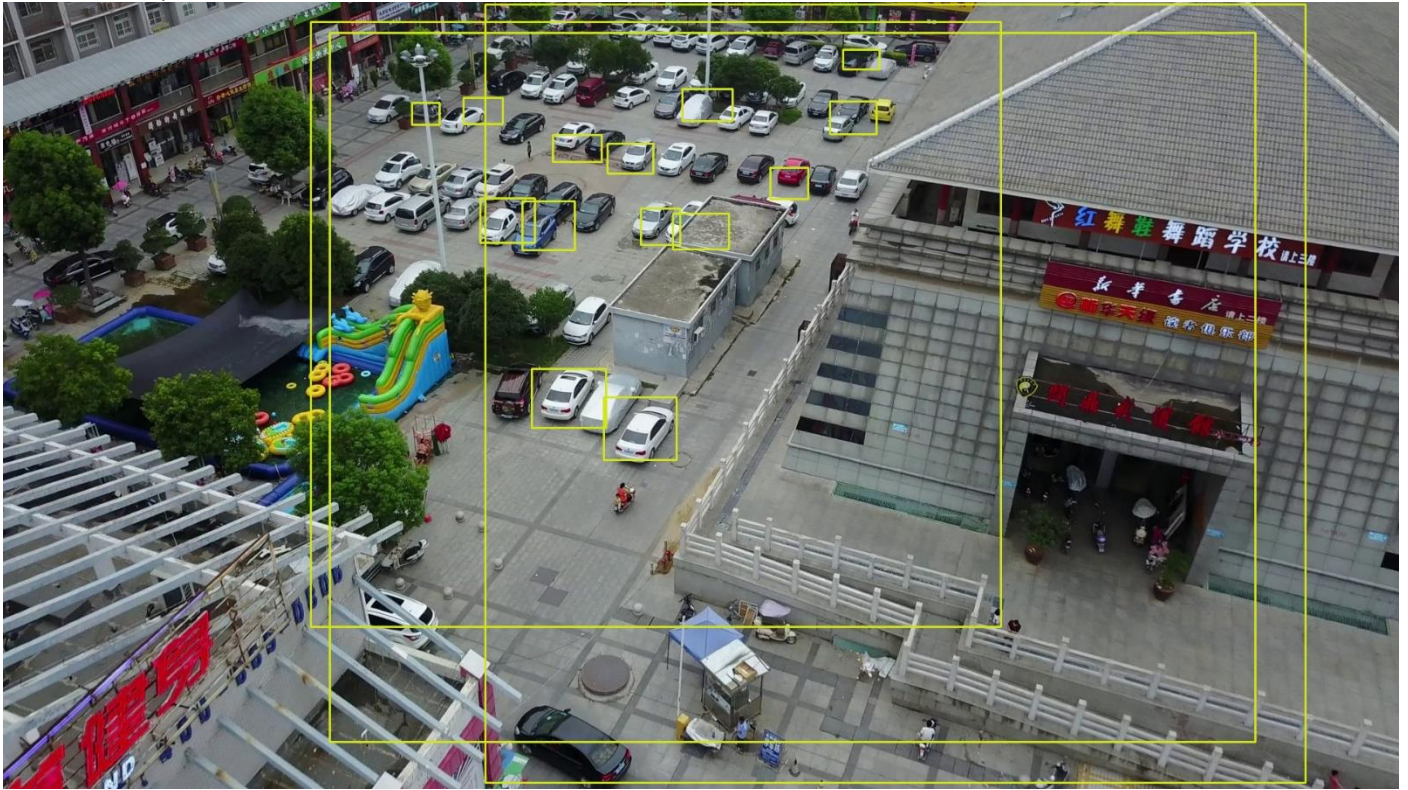
Predictions by YOLO:



Predictions by RetinaNet:



Predictions by SSD:



Conclusion

While all models fail to detect all the objects, we can see that:

1. YOLO model has the maximum mAP score and hence has the best performance
2. RetinaNet model has middling mAP. It needs to be trained on much more epochs which is not possible on colab.
3. SSD model has the least mAP. This is apparently because proper training of SSD requires 200+ hrs.