Implementing YOLOv3 and Retinanet50 on the Berkeley Deep Drive Dataset

Jacob Ieyoub

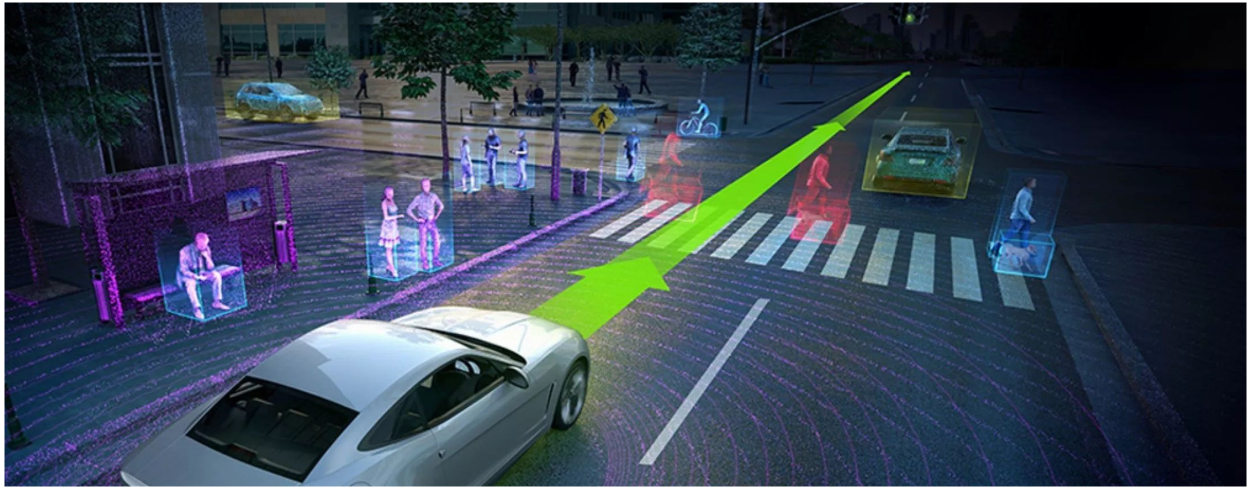
Springboard

Self-Driving Cars

Inspiration for this project was the book *Autonomy* by Lawrence Burns.

The Goal: Train existing object detection models (YOLOv3 and Retinanet50) on the Berkeley Deep Drive (BDD) dataset for object detection on street-level videos. 10 classes: bike, bus, car, motorcycle, person, rider, traffic light, traffic sign, train and truck.

Potential Clients: Companies developing self-driving car software



Approach

Train YOLOv3 and Keras Retinanet50 object detection models on BDD classes.

Google Colab, OpenCV, Keras and TensorFlow were all used for the project.

Resources:

Berkeley Deep Drive

YOLOv3 in the Cloud

YOLOv3 : An Incremental Improvement

YOLO : Real-Time Object Detection

Darknet Repository

Real-time object detection for autonomous vehicles using deep learning

Berkeley Deep Drive

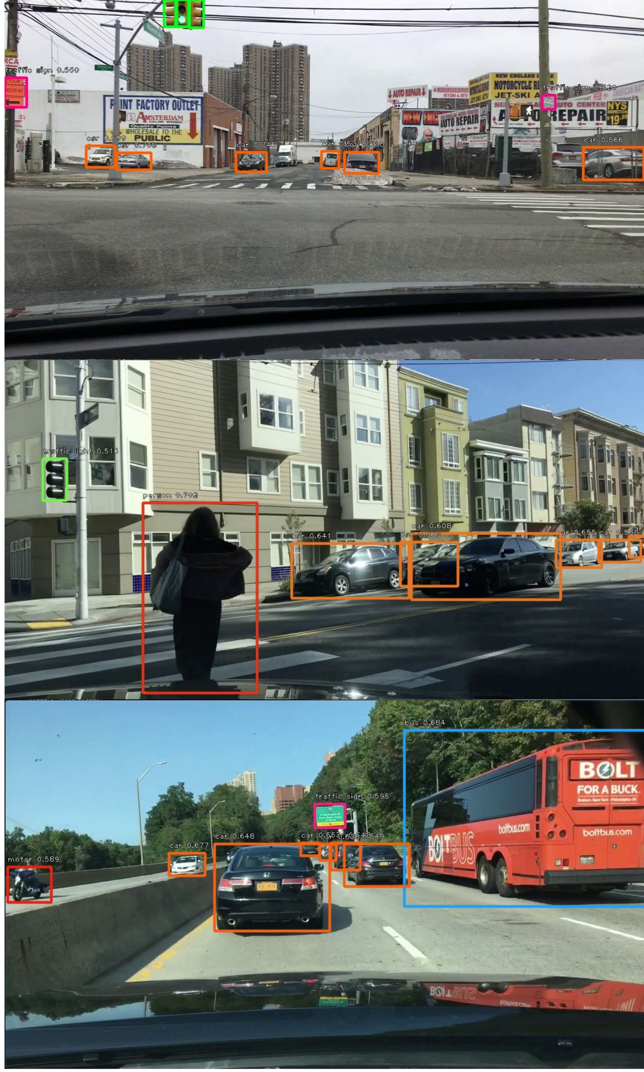
Dataset created by UC Berkeley in several different driving environments. Challenging for object detectors.

Locations include San Francisco, Berkeley and New York City

Weather includes clear, overcast, snowy, rainy, partly cloudy

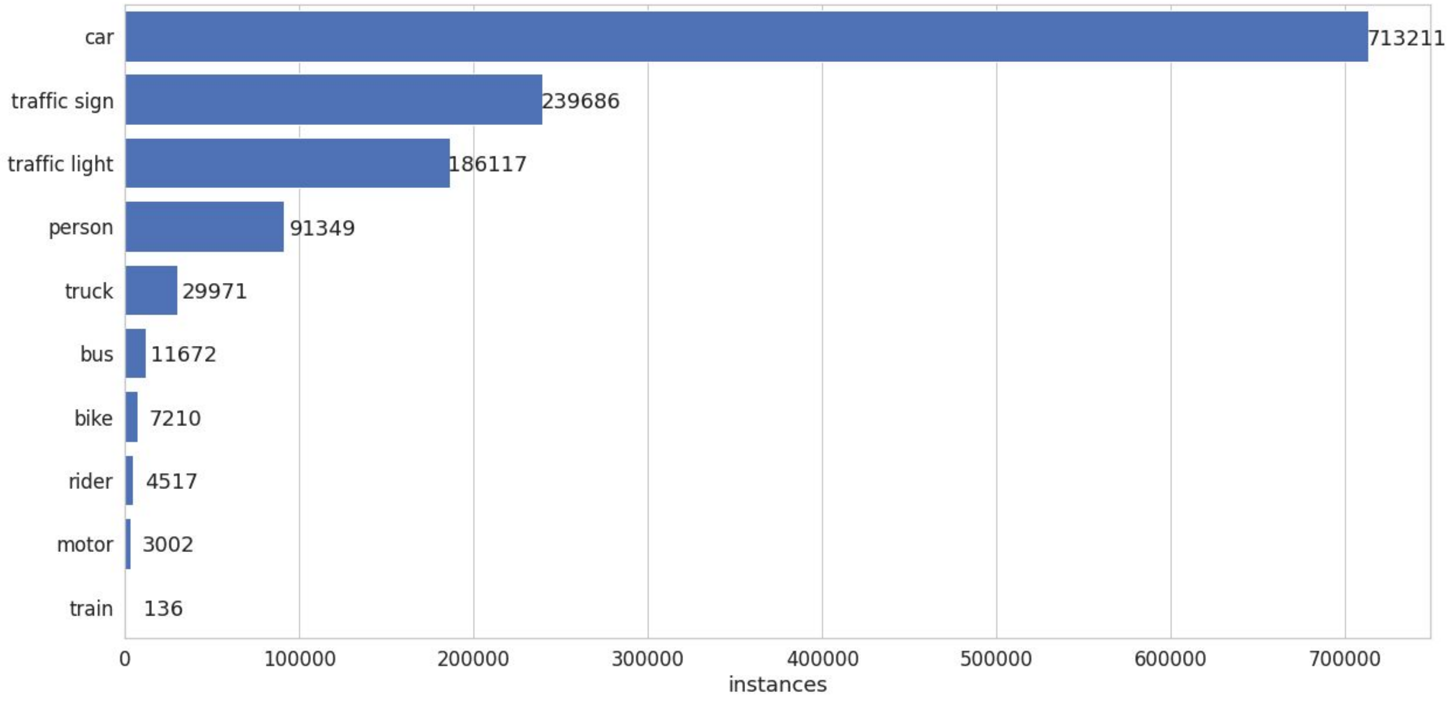
Driving scene includes city street, highway, residential, parking lot, tunnel, gas station

Also includes day, night and dawn/dusk times of day.



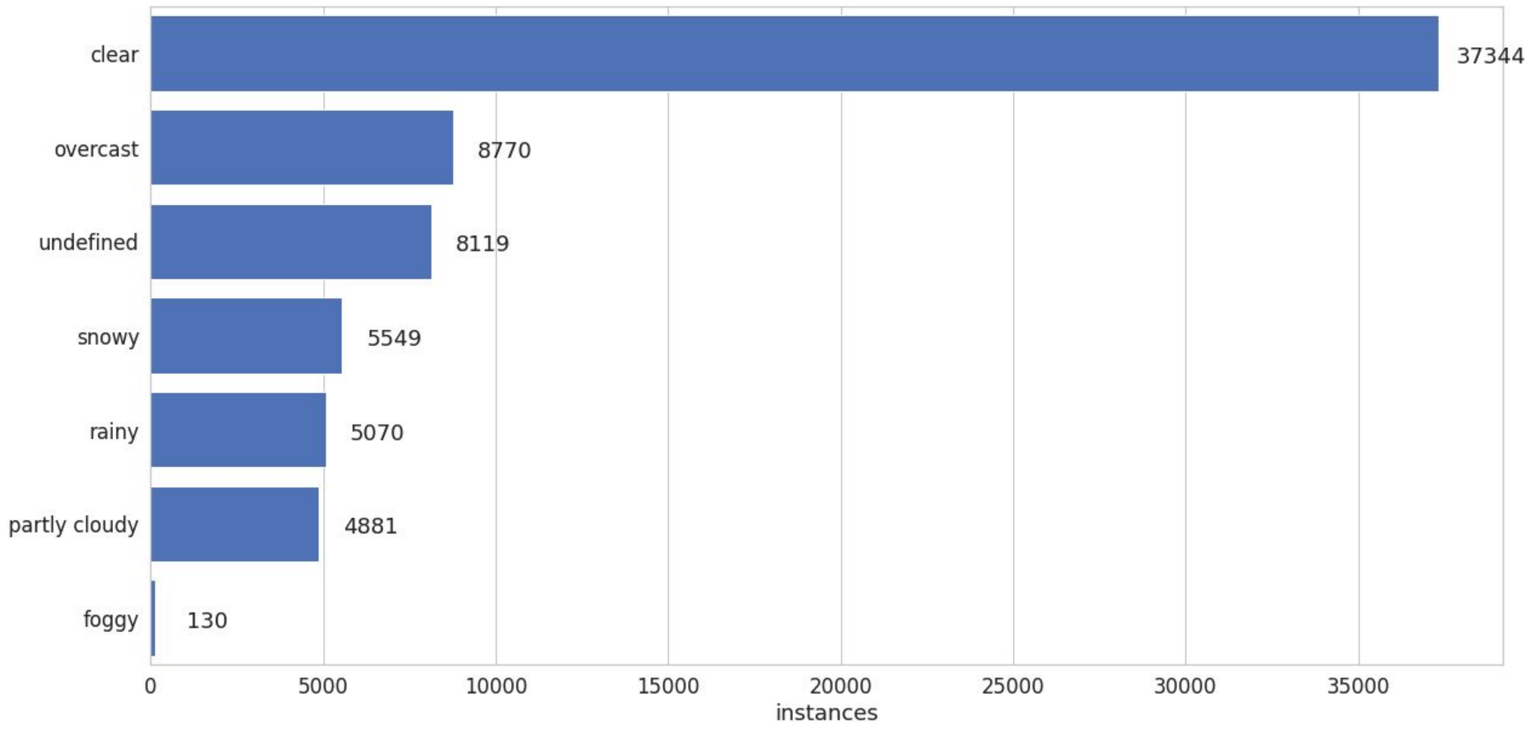
Exploring BDD

Class Distribution

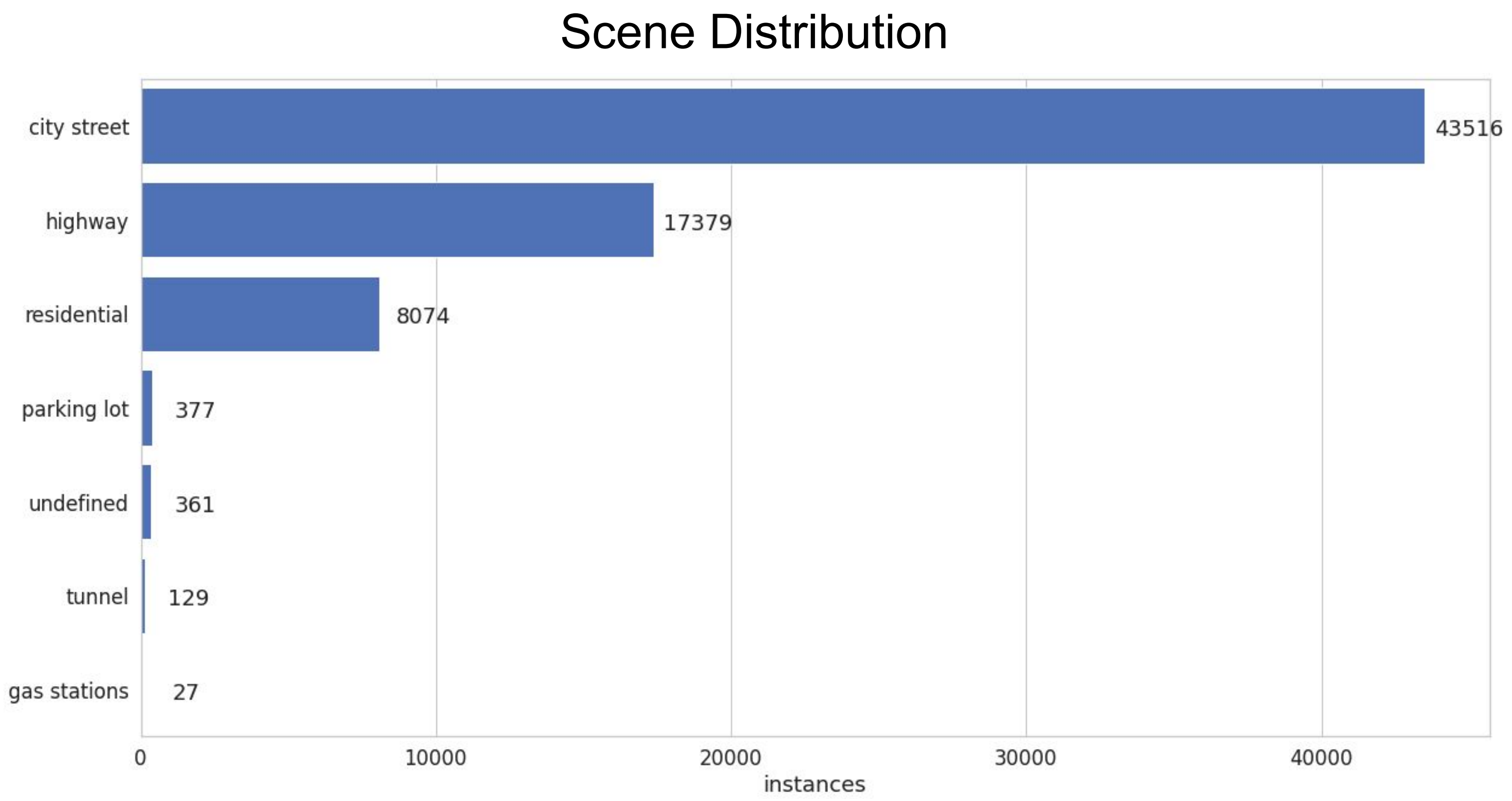


Exploring BDD

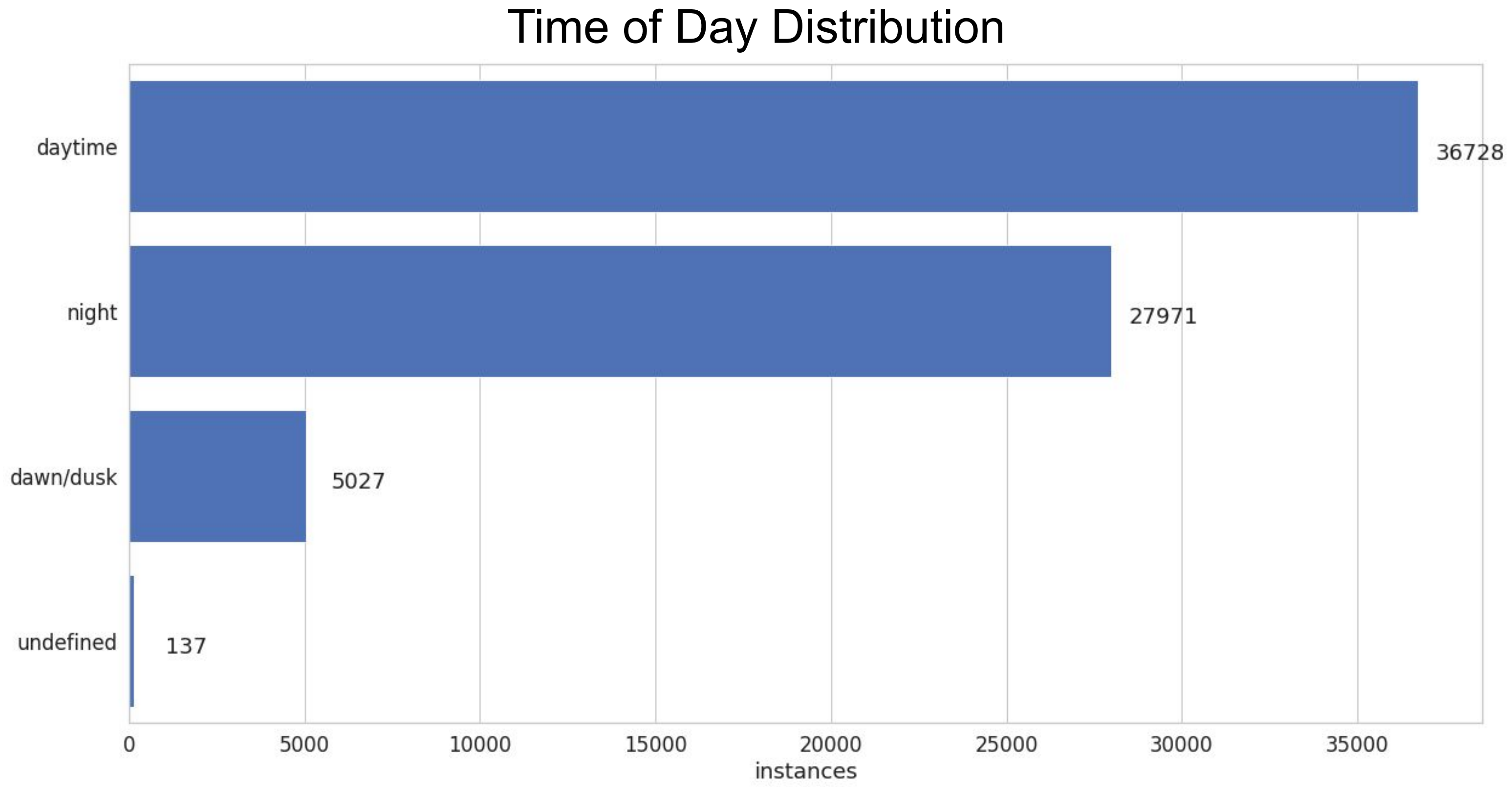
Weather Distribution



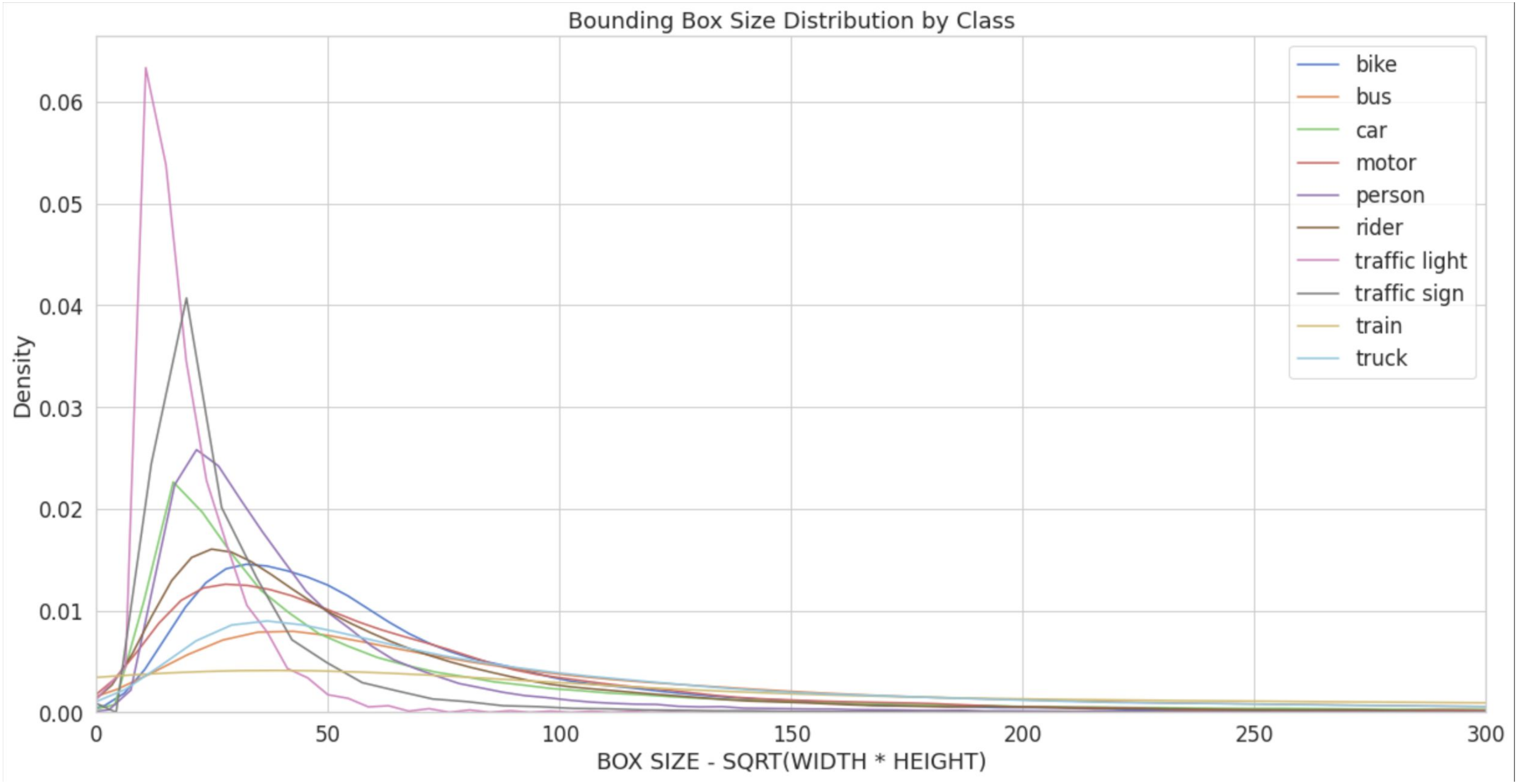
Exploring BDD



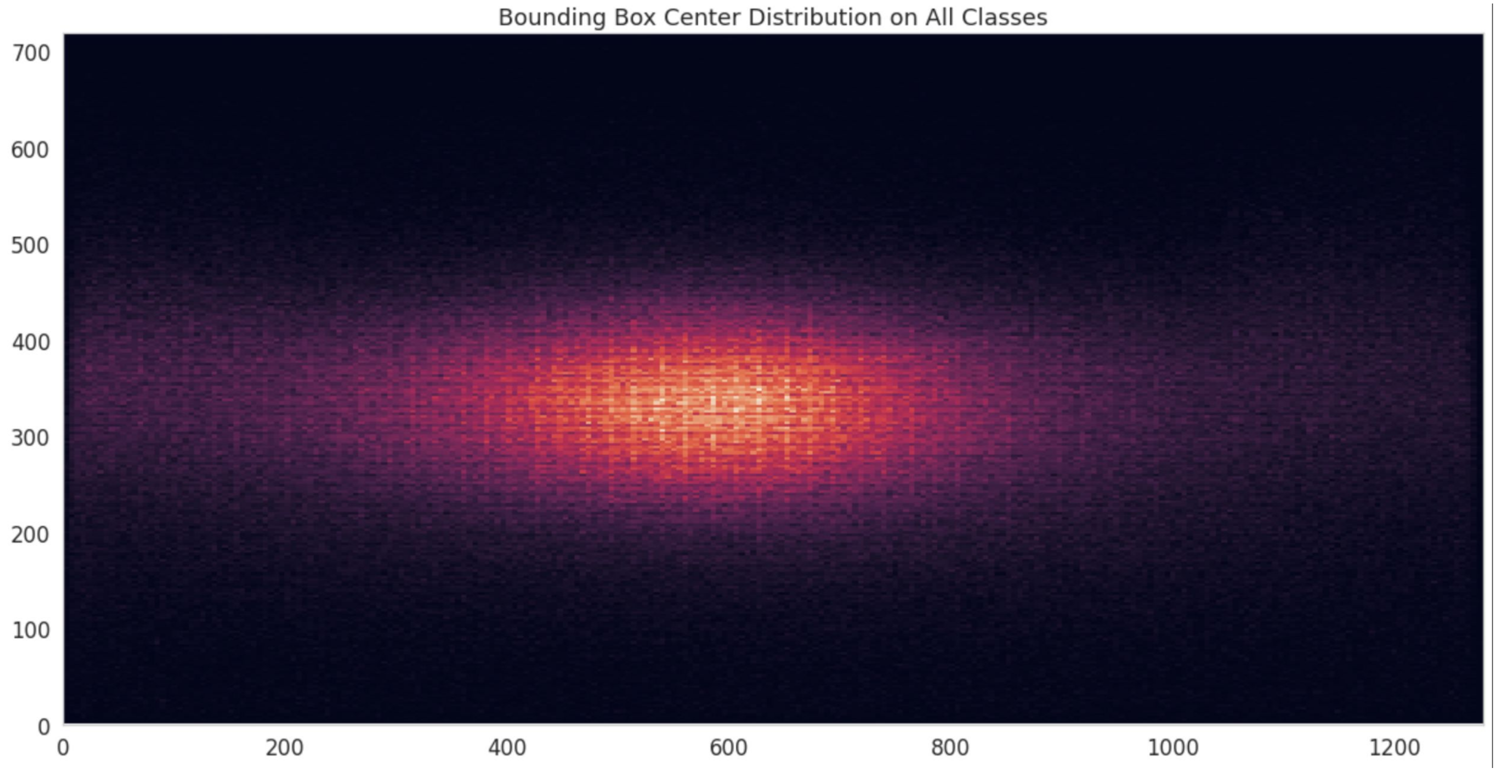
Exploring BDD



Exploring BDD



Exploring BDD



YOLOv3

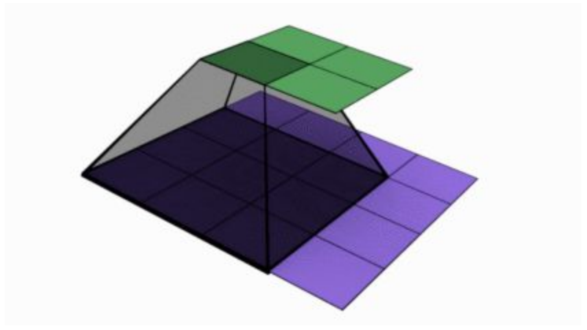
Convolutional Neural Network

One stage detector

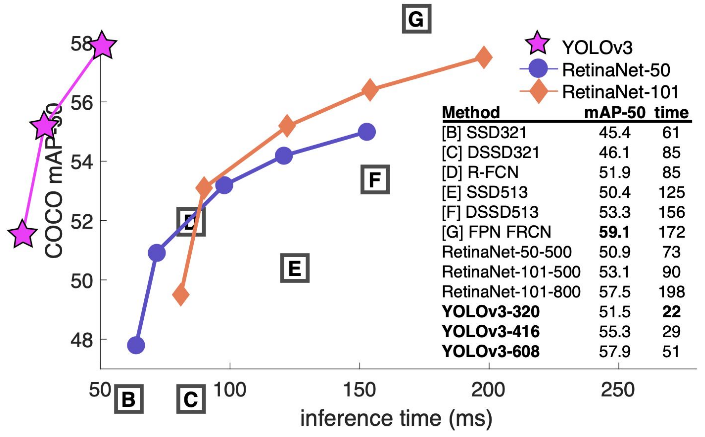
Known for speed without sacrificing too much mAP

Ideal for real-time object detection.

As seen in the graph, has just as good mAP as Retinanet systems at lower inference times.



Convolutional Neural Network. Here taking 4x4 matrix to 2x2 matrix.



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

Performance of YOLOv3 vs. other popular object detection models on the COCO dataset.

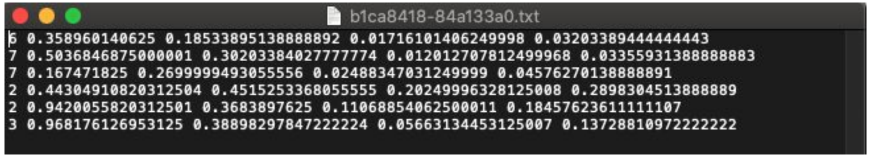
Training - YOLOv3

Implemented YOLOv3 training using Google Colab with the darknet framework.

Training requires:

- Images
- annotation .txt files - (<object class>,<x-center>,<y-center>, <box_width>, <box_height>)

  all float values relative to width and height of image



```
6 0.358960140625 0.18533895138888892 0.01716101406249998 0.03203389444444443
7 0.5036846875000001 0.30203384027777774 0.012012707812499968 0.03355931388888883
7 0.167471825 0.2699999493055556 0.02488347031249999 0.04576270138888891
2 0.44304910820312504 0.4515253368055555 0.20249996328125008 0.2898304513888889
2 0.9420055820312501 0.3683897625 0.11068854062500011 0.18457623611111107
3 0.968176126953125 0.38898297847222224 0.05663134453125007 0.13728810972222222
```

Example of .txt annotation file, name is same as image file

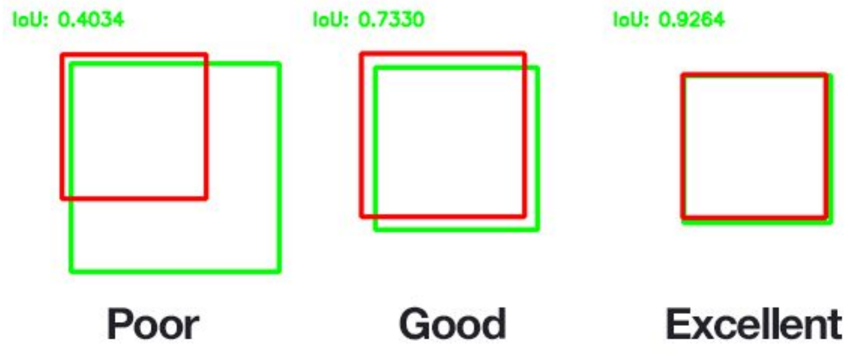
- image path .txt files (train.txt and test.txt)
- .data file that contains the following: number of classes, path to train.txt file, path to test.txt file, path to classes.names file and path to save weights during training
- network configuration .cfg file
- pretrained .weights file

Evaluation

Two common metrics for Object Detection:

Intersection of Union (IoU)

-fraction of intersection between predicted box and ground truth box



Intersection over Union

mean Average Precision (mAP)

-a way to combine average precision(AP) and IoU into one metric. Specifically as calculating the AP across all classes at a specified IoU Threshold. Traditionally @ IoU = 0.5

$$\mathrm{mAP} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{AP}_i$$

mean Average Precision

Evaluation

YOLOv3

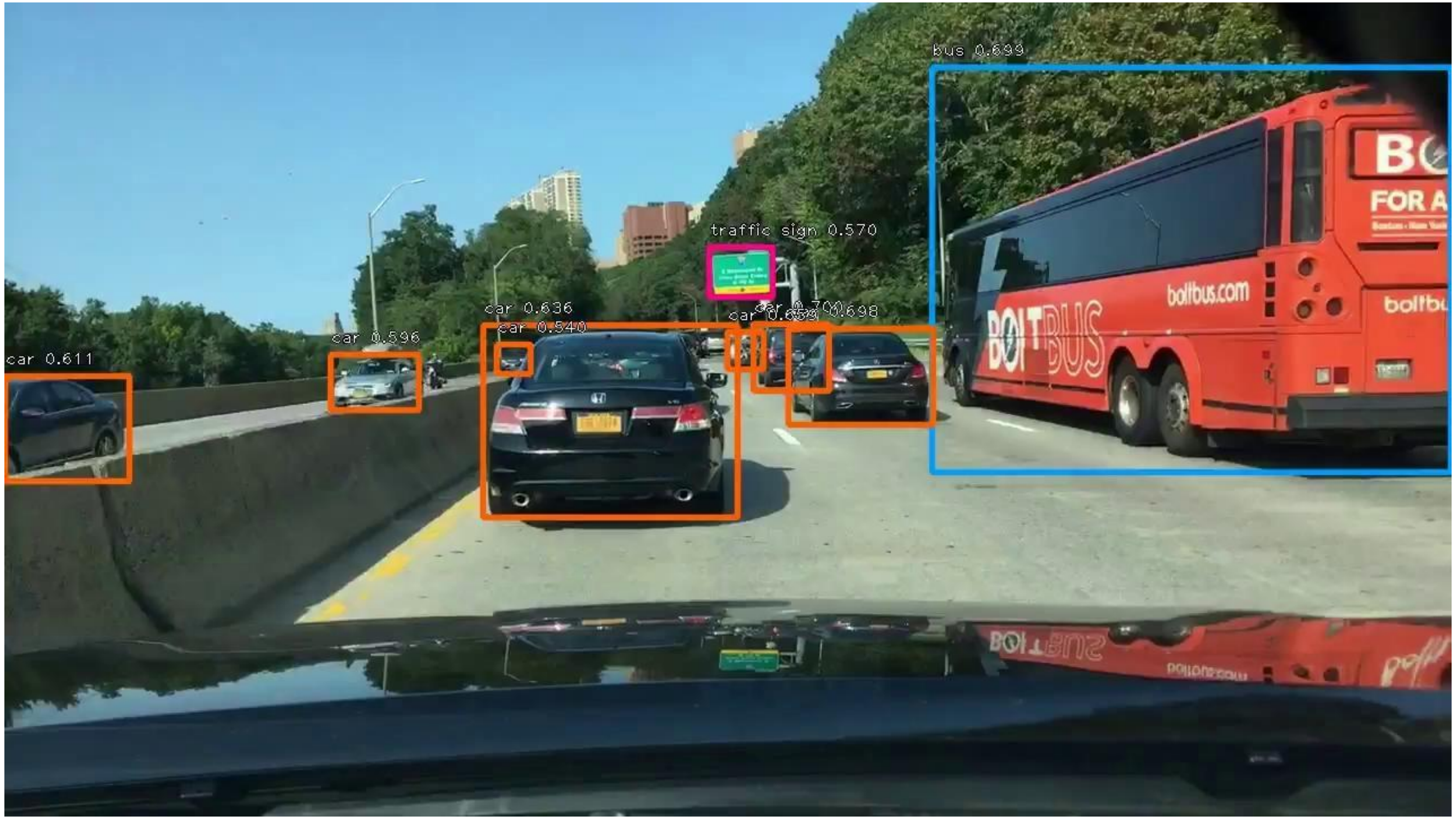
mAP @ IoU 0.5 = 33.19

As expected, train class had AP = 0.00 because of extremely low sample size.

Motor, rider and bike had smallest sample size after train. This reflects in in their lower APs relative to other classes.

class	AP
car	58.96
traffic sign	50.01
truck	47.27
bus	43.60
traffic light	36.12
person	31.73
bike	25.27
rider	23.94
motor	15.04
train	0.00

Example Video - [Google Drive Link]



Conclusion / Future Work

Out of the box, YOLOv3 performs pretty well on Berkeley Deep Drive.

To improve performance, parameter optimization should be explored. Changing the image input size, image augmentation during training, batch size, etc.

Increase number of samples of the classes with a small sample size by using other datasets.

Incorporate object tracking so that objects maintain their detection frame by frame.

Experiment with newer versions of YOLO.