**Write-up Document**

Project Overview:

This project is aimed at detecting different objects in an image of real world traffic. Such a model can be deployed in self driving cars to understand the environment around the vehicle and information of upcoming traffic/objects. We are using tensorflow object detection API’s pre trained model to build our model which learns to detect objects in urban environment

Set up:

Downloading a pre trained model:

* cd /home/workspace/experiments/pretrained\_model/
* wget http://download.tensorflow.org/models/object\_detection/tf2/20200711/ssd\_resnet50\_v1\_fpn\_640x640\_coco17\_tpu-8.tar.gz
* tar -xvzf ssd\_resnet50\_v1\_fpn\_640x640\_coco17\_tpu-8.tar.gz
* rm -rf ssd\_resnet50\_v1\_fpn\_640x640\_coco17\_tpu-8.tar.gz

PS : No need to create a config, we already have it

Model Training:

cd /home/workspace

python experiments/model\_main\_tf2.py --model\_dir=experiments/reference/ --pipeline\_config\_path=experiments/reference/pipeline\_new.config

Tensorboard visualization:

python -m tensorboard.main --logdir experiments/reference/

Evaluation:

python experiments/model\_main\_tf2.py --model\_dir=experiments/reference/ --pipeline\_config\_path=experiments/reference/pipeline\_new.config --checkpoint\_dir=experiments/reference/

Export Trained Model:

python experiments/exporter\_main\_v2.py --input\_type image\_tensor --pipeline\_config\_path experiments/reference/pipeline\_new.config --trained\_checkpoint\_dir experiments/reference/ --output\_directory experiments/reference/exported/

Extract animation:

python inference\_video.py --labelmap\_path label\_map.pbtxt --model\_path experiments/reference/exported/saved\_model --tf\_record\_path data/test/segment-12200383401366682847\_2552\_140\_2572\_140\_with\_camera\_labels.tfrecord --config\_path experiments/reference/pipeline\_new.config --output\_path animation.gif

**Dataset**

* Exploratory Data Analysis:
  + Displaying 10 images with color coding for each class(more details in Exploratory Data Analysis.ipynb )
  + Training data
  + **A picture containing text, road

    Description automatically generated**
  + Similar images are displayed for validation and test sets too
  + Additional analysis of class distribution:
    - Training:

Text

Description automatically generated

* + - Validation data

Graphical user interface, text, application

Description automatically generated

* + - Testing data

Graphical user interface, text, application

Description automatically generated

* Explore Image augmentation
* First run
  + Training
    - Reference experiment

**Graphical user interface, chart

Description automatically generated**

* + Evaluation

**Text

Description automatically generated**

**Improving on reference:**

* Augmentation applied
  + random\_adjust\_hue : This is done so that vehicles are detected not based on their colors anad conversely all cars should be deteced as cars irrespective of its color

**A picture containing text, road, outdoor, sign

Description automatically generated**

* + adjust\_gamma :  Gamma value changes difference between light and dark areas(icreasing gamma would make darker area darker and light areas lighter)

**Graphical user interface

Description automatically generated**

* + random\_jitter\_boxes : this changes box bounds by a value and would help in generalization

**A picture containing text, outdoor, street, way

Description automatically generated**

* Final Run
  + Training
    - Loss is smaller with all augmentations mentioned above
    - Algorithm has started to pick up objects in image as seen in testing
    - Low classification and localization loss
    - Slow decreasing of loss towards optimal(not yet there)

**Chart

Description automatically generated**

* + Validation

**A screenshot of a computer

Description automatically generated with medium confidence**

Other Modifications of tensorflow obejct detection API explored include:

1. Changing optimizer from momentum optimizer to RMS optimizer
2. Changing learning rate schedule from cosine decay to exponential decay
3. increasing batch size is no longer possible with memory usage overhead
4. changing achor\_generator values
5. IOU threshold for NMS
6. using drop out and augmentation to avoid overfitting