Project Overview

Object detection is a vital part of the self-driving car perception system, it allows for the identification of objects using camera images. This can allow for identification of various participants in a scene and especially useful for the identification of colours which other sensor types would have issues with, also it can be used to reinforce data from other senor modalities to derive a more robust scene representation.

The objective of this project is to re-train the Tensorflow Resnet50 SSD object detection model using the Waymo open dataset to allow for the classification of vehicles, pedestrians and cyclist.

Setup

This project was run on the Udacity workspace VM using the files provided.

All datasets are stored in the data folder, which contains the Train, Val and Test folders, the pre trained model along with its checkpoints are stored on the experiments/pretrained_model folder.

After the necessary modifications are done to the pipeline.config file, training is executed using the model_main_tf2.py file, once completed validation can be run using the generated checkpoints and the model evaluated.

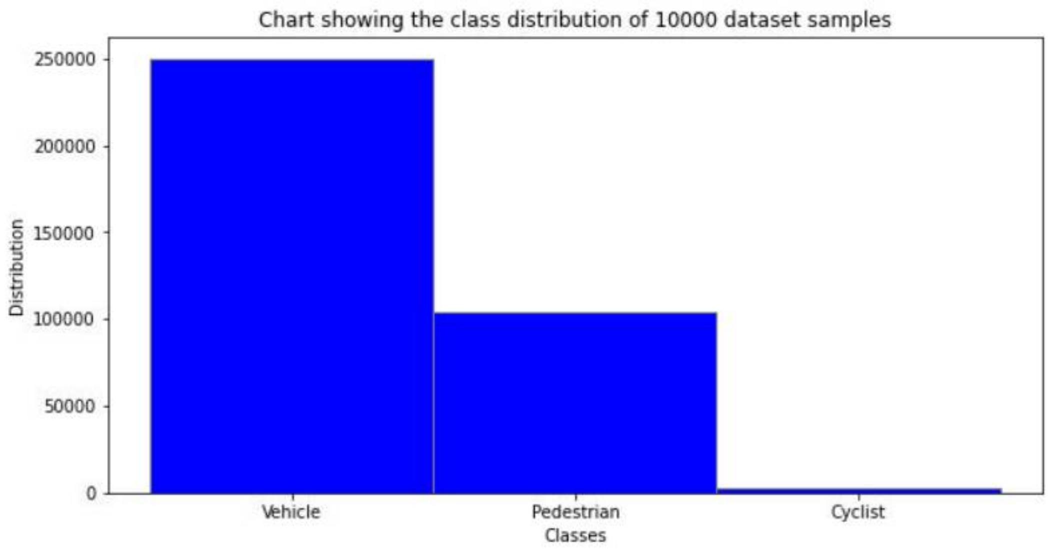
Dataset

The data has been provided as individual tfrecord files which contains annotated frames of various recorded driving scenes.

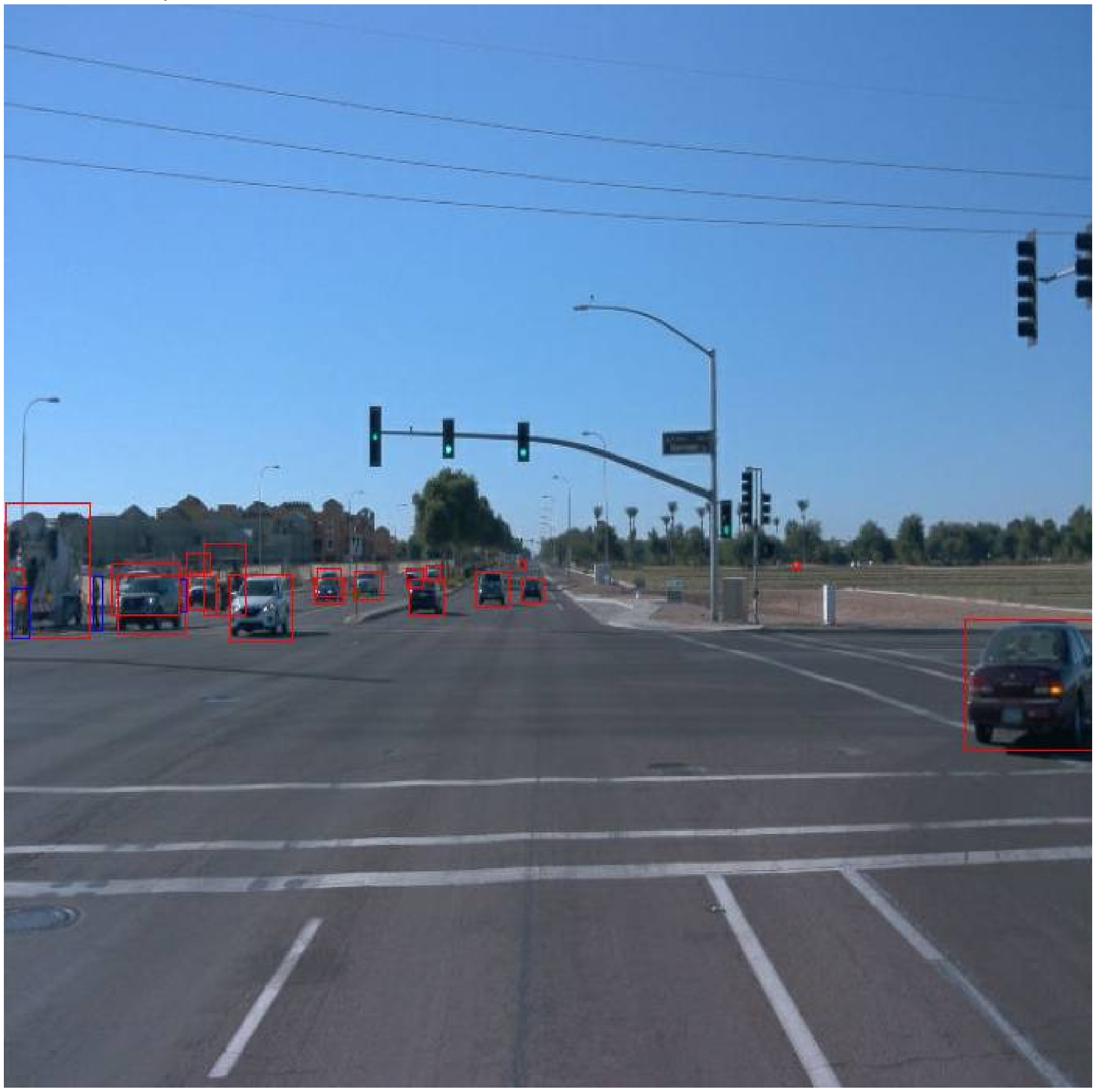
The datasets contain a wide variety of road conditions with examples with bright sunlight, rain, low light and night scenes. There is also a wide variety of scene participants, examples can be found with cars, trucks, busses and other vehicle types at varying angles, occlusions and distance from the camera.

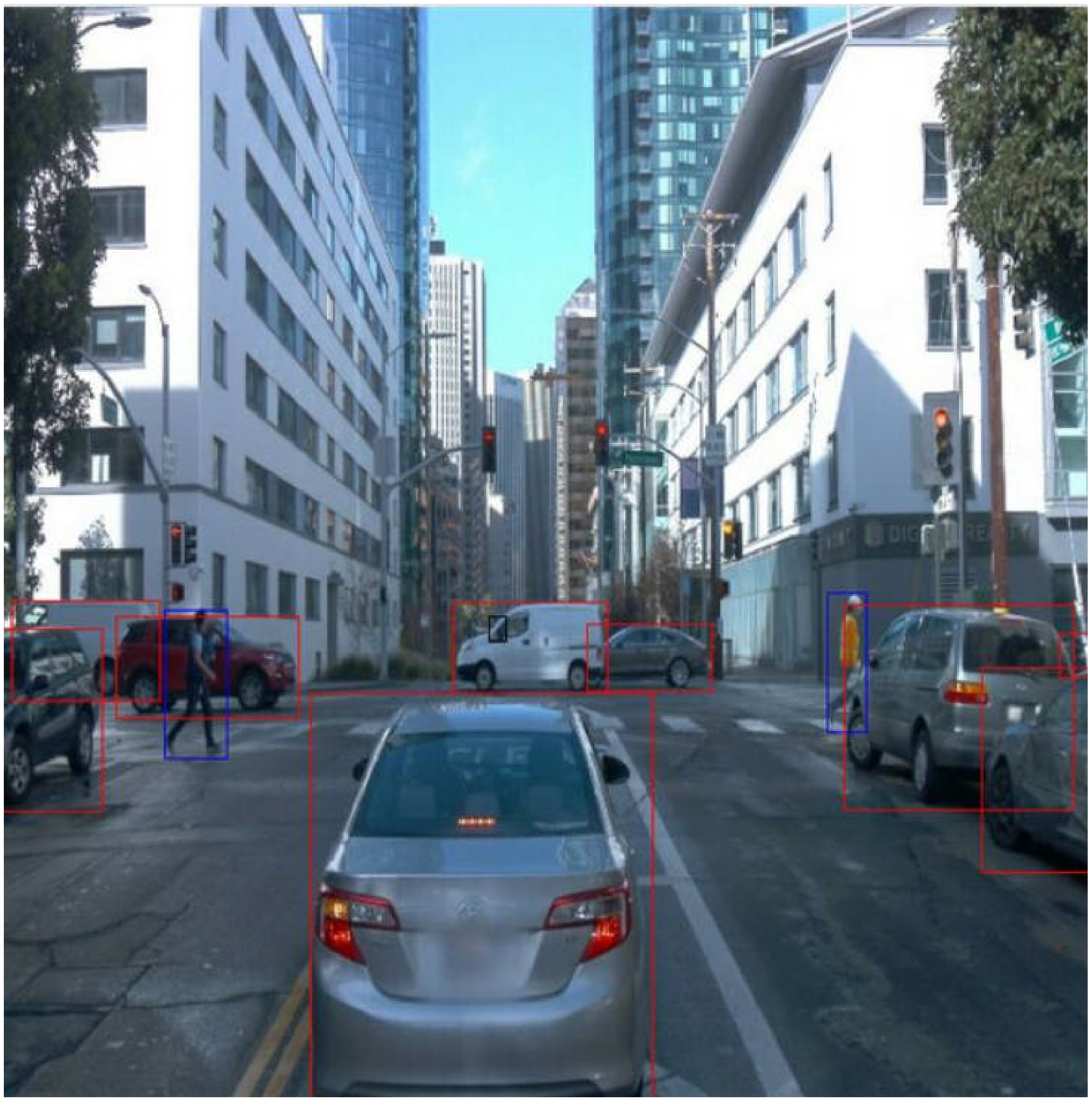
It was found after looping through the classes in 10000 random samples, it can clearly be seen that there is a major class imbalance, with vehicles being most common, pedestrians at less than half this frequency and cyclist with minimal occurrences.

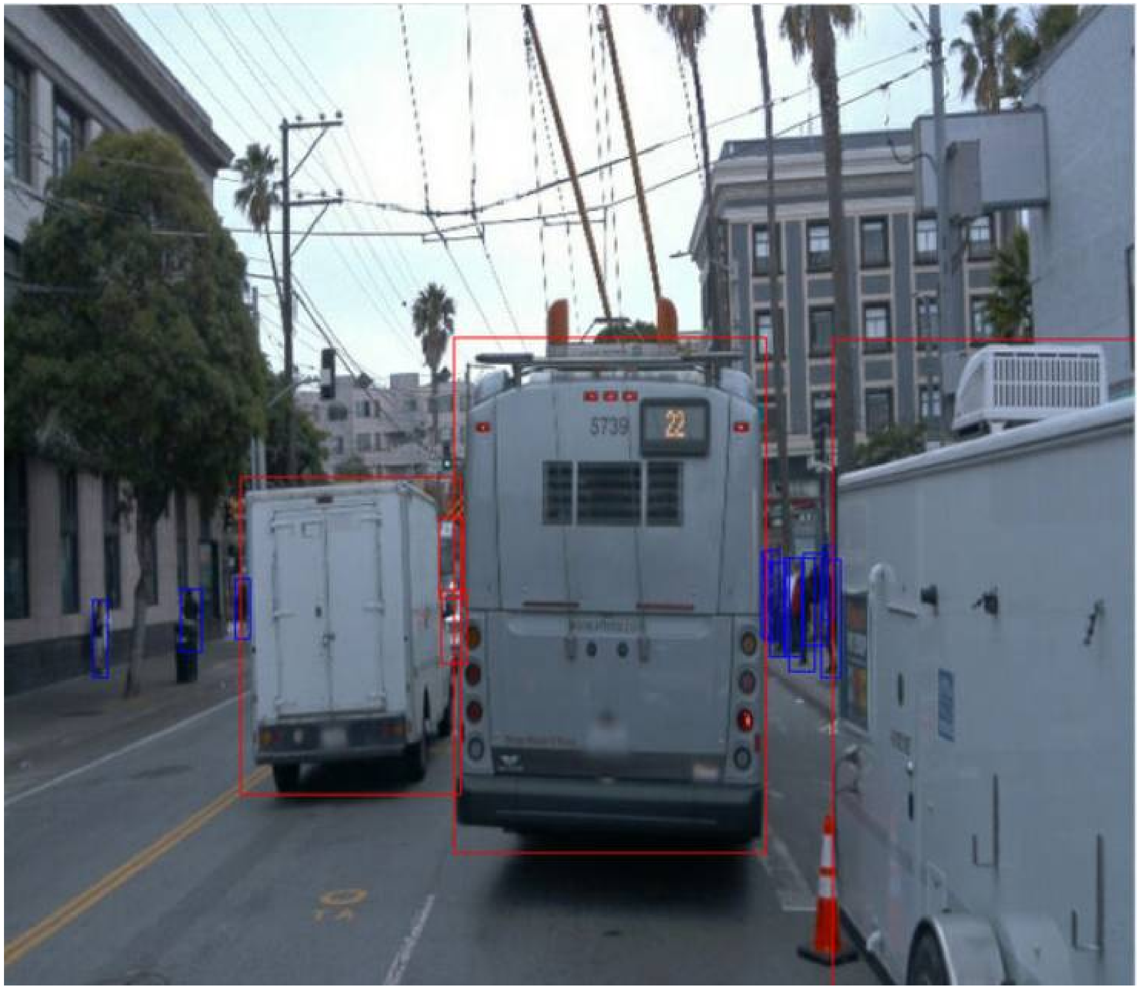
The below chart shows this class imbalance:

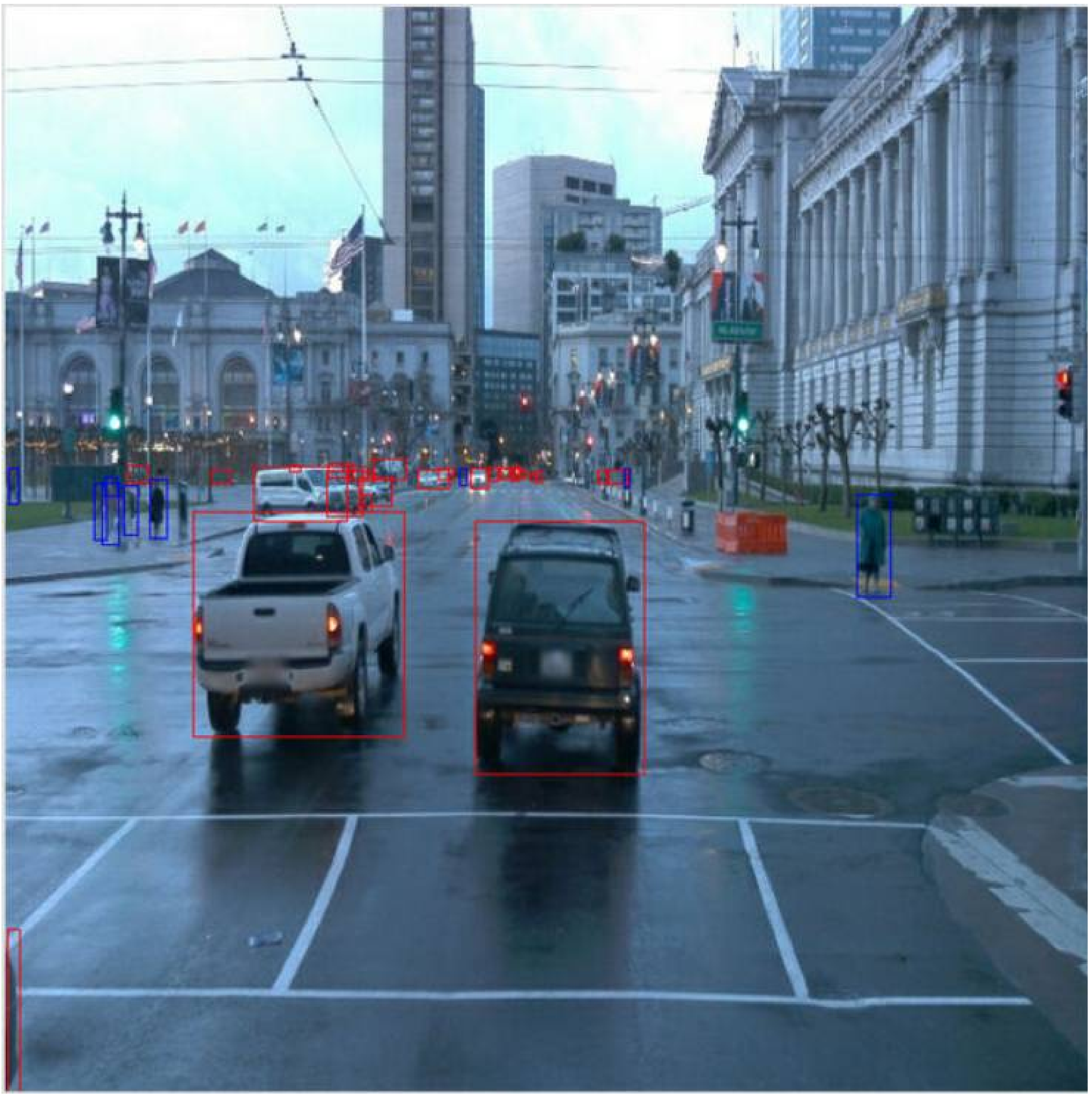


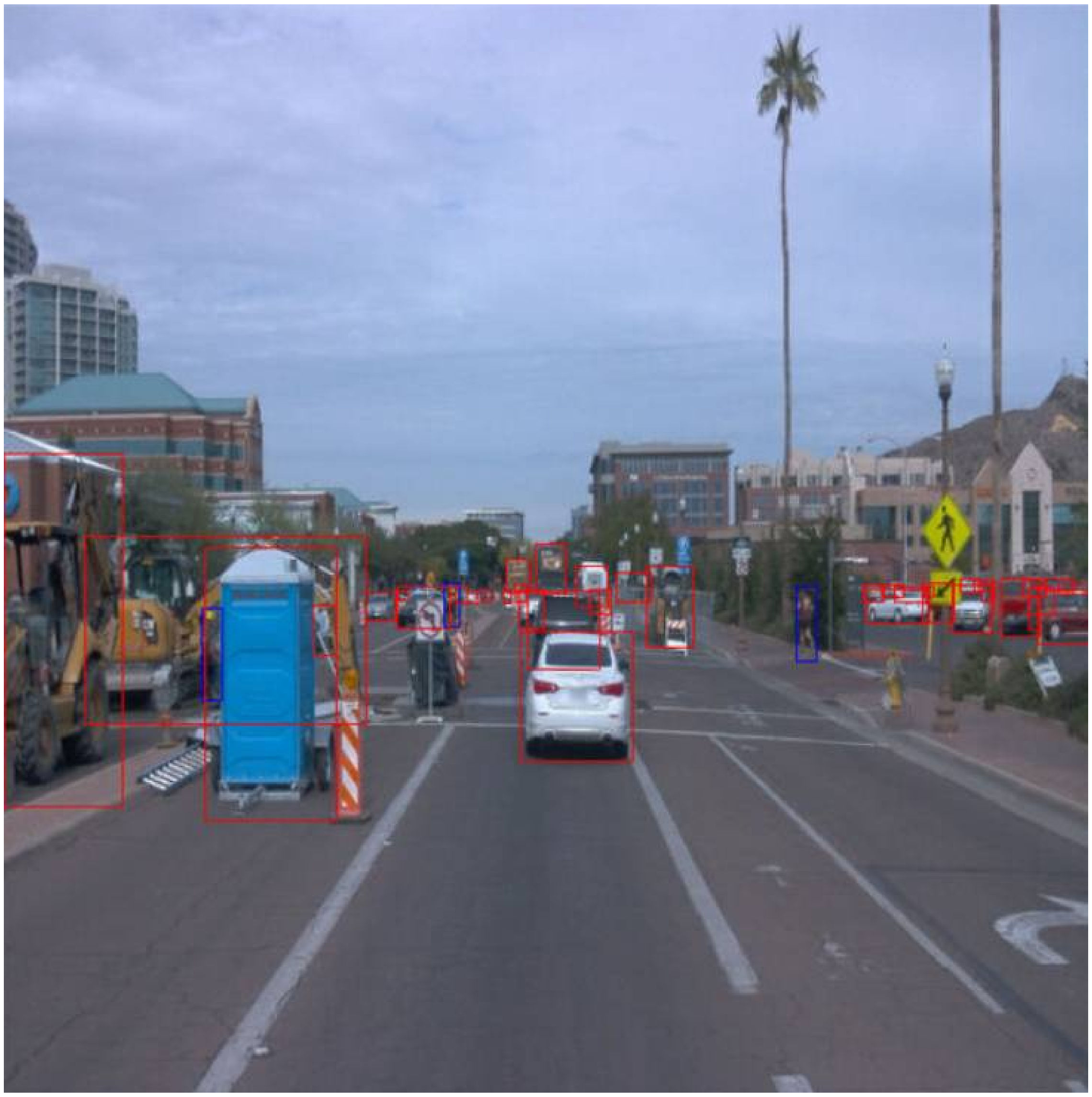
Below are sample frames from the dataset.






5739
22



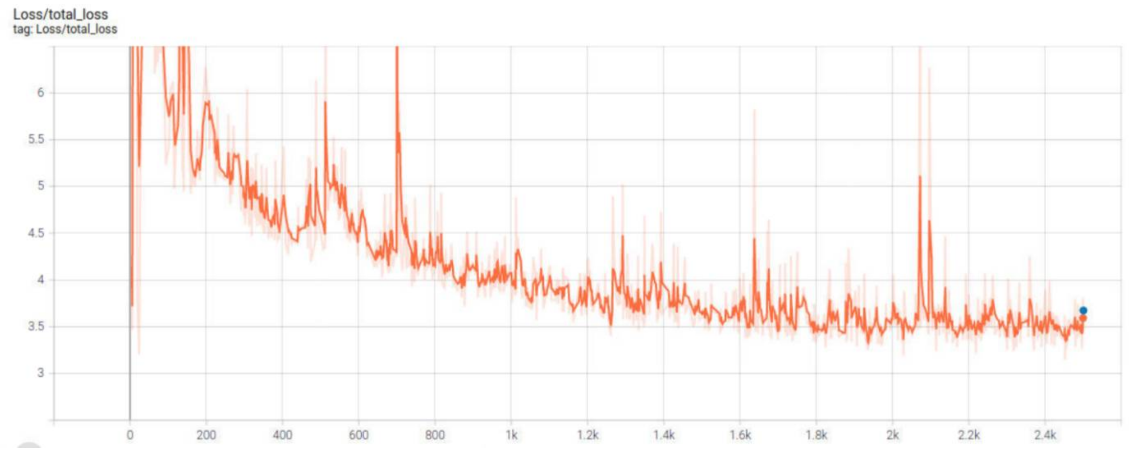


Training

Reference Experiment

The reference experiment was run for 2500 epochs using the default configuration of the config file. This yielded a total loss of approximately 3.5 with the curve starting to remain flat.

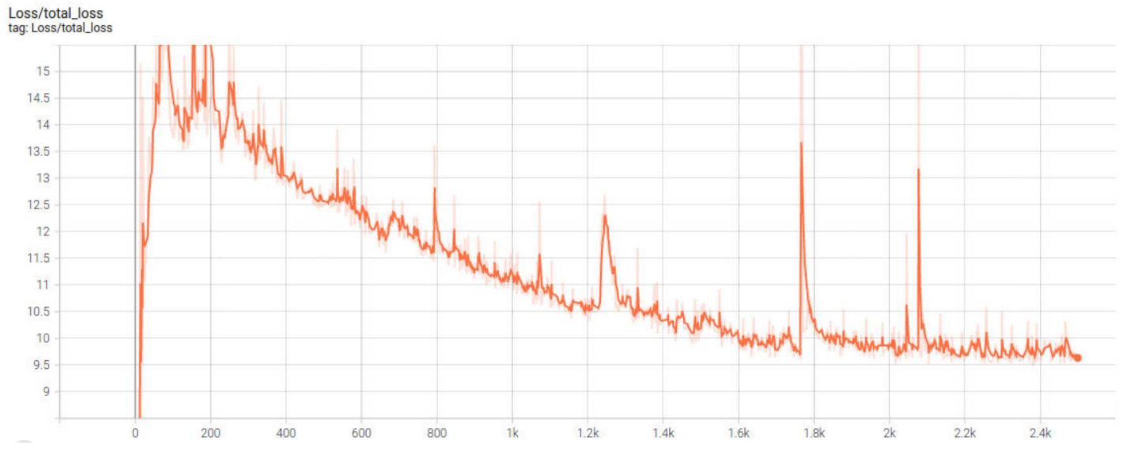
See below total loss curve for the reference experiment:



Experiment 0

For this experiment I added in some data augmentation (random adjust brightness, random adjust saturation, random black patches) in an effort to give the model a more diverse dataset to be trained on, but this had negative results with a total loss of approximately 10.25 after 2500 epochs.

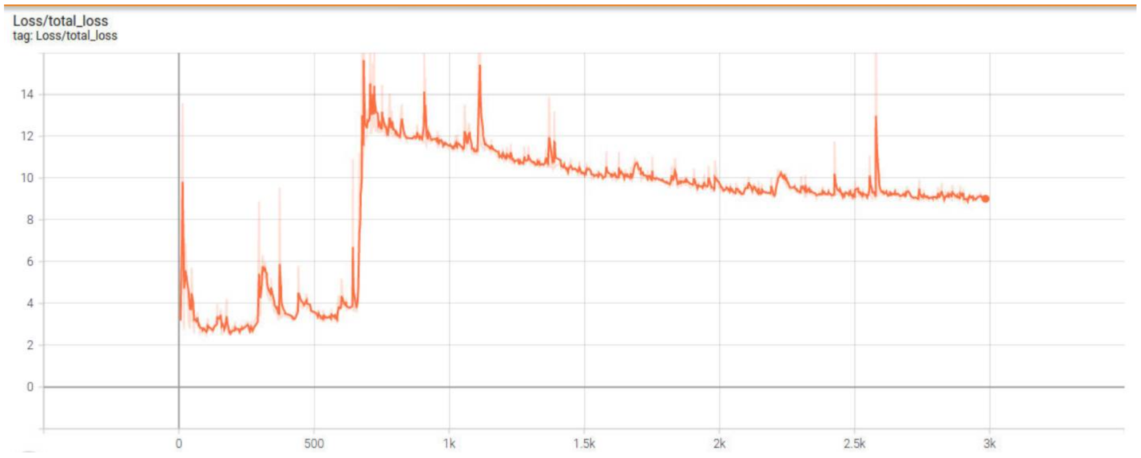
See below total loss curve for the experiment zero:



Experiment 1

For this experiment it tried changing the learning rate to 0.04 and increasing the number of epochs to 3500 but this provided even worse results with a total loss of approximately 9.

See below total loss curve for the experiment one:

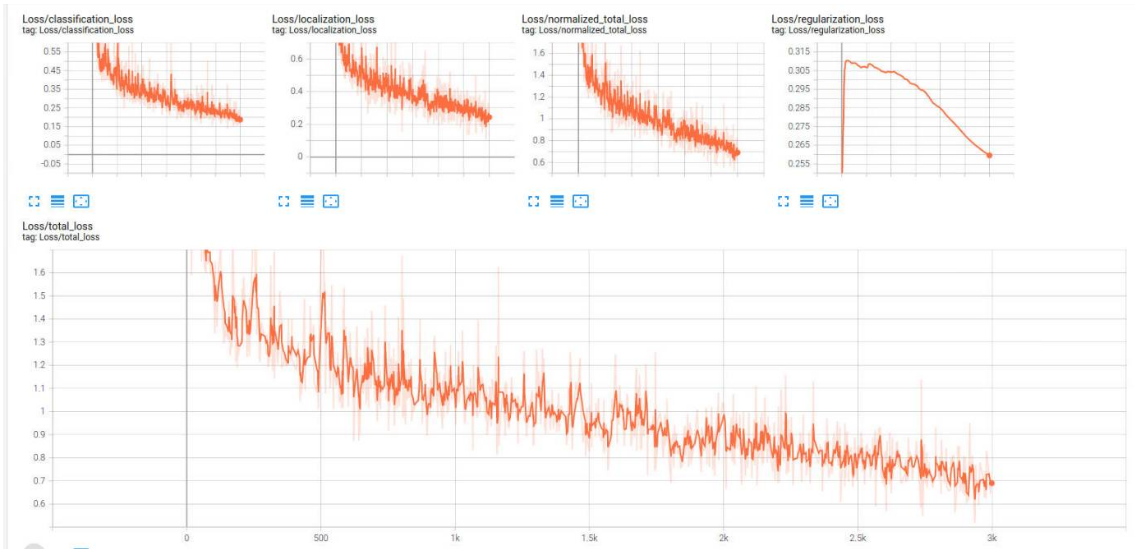


Experiment 2

For this experiment I removed the random_black_patches augmentation as I felt that this would not be indicative of the types of data seen in the real world, I also changed the batch size to 4 and set the warm up steps to 1000.

This experiment provided favourable results with a total loss 0.7 at 3000 epochs, this could have been even further improved but I kept getting an out of memory error at this point.

See below total loss curve for the experiment zero:


Loss/classification_loss
tag: Loss/classification_loss
Loss/localization_loss
tag: Loss/localization_loss
Loss/normalized_total_loss
tag: Loss/normalized_total_loss
Loss/regularization_loss
tag: Loss/regularization_loss
Loss/total_loss
tag: Loss/total_loss
1k
1.5k
2k
2.5k
3k