Submission Template

Project overview

This section should contain a brief description of the project and what we are trying to achieve. Why is object detection such an important component of self driving car systems?

In this project data provided by Waymo must be detected and classified using SSD Resnet 50 640x640 model available in TensorFlow 2 Detection Model Zoo. The dataset consists of images in the urban environment with annotated vehicles, pedestrians and cyclists. Initially dataset is explored and studied to see the distribution of classes, quality of images based on weather, daylight, clarity, etc. In the next step augmentations are applied to balance the dataset. Finally the dataset is trained to detect and classify the images.

Set up

This section should contain a brief description of the steps to follow to run the code for this repository.

This code was run on the workspace following the steps mentioned in project instructions (workspace).

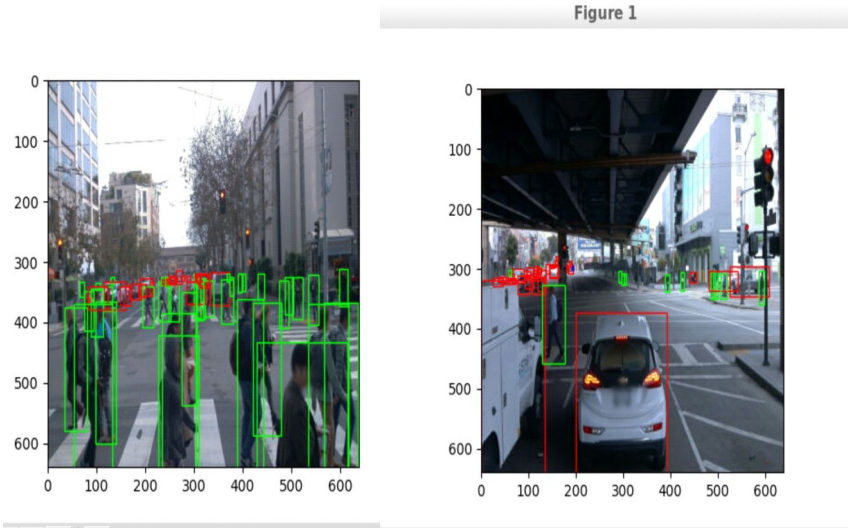
Dataset

Dataset analysis

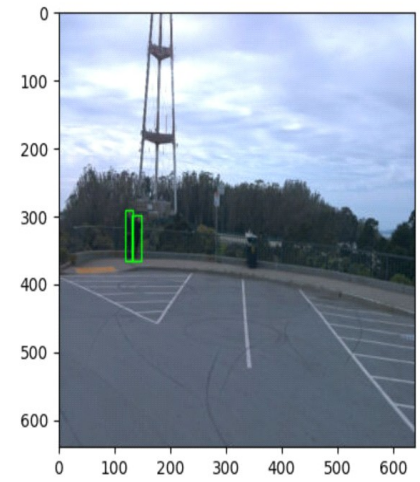
This section should contain a quantitative and qualitative description of the dataset. It should include images, charts and other visualizations.

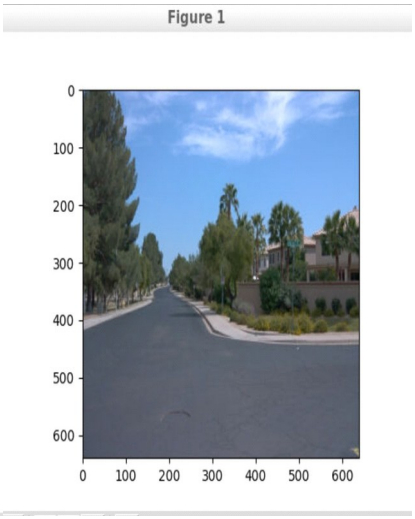
The images in the dataset have been shuffled and randomly picked to study the variations in quality, weather, density and class density. A few images featuring sparse and dense distribution of classes, low brightness, bad weather and blurry image quality have been displayed.

High Density of pedestrians, vehicles

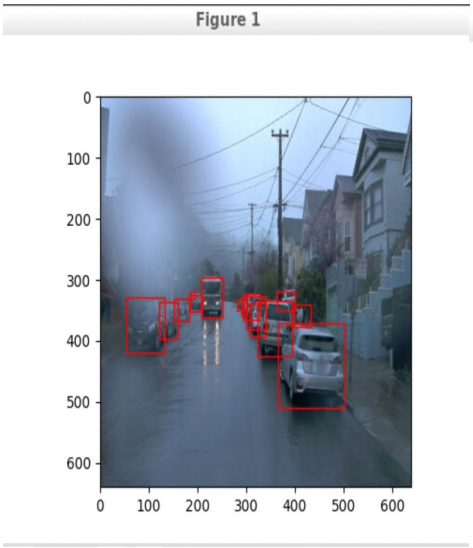


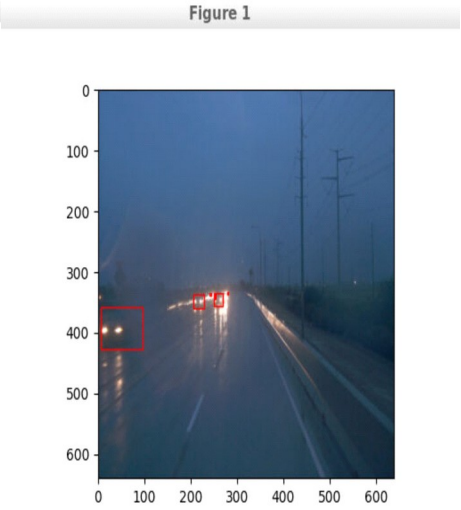
Low density of classes



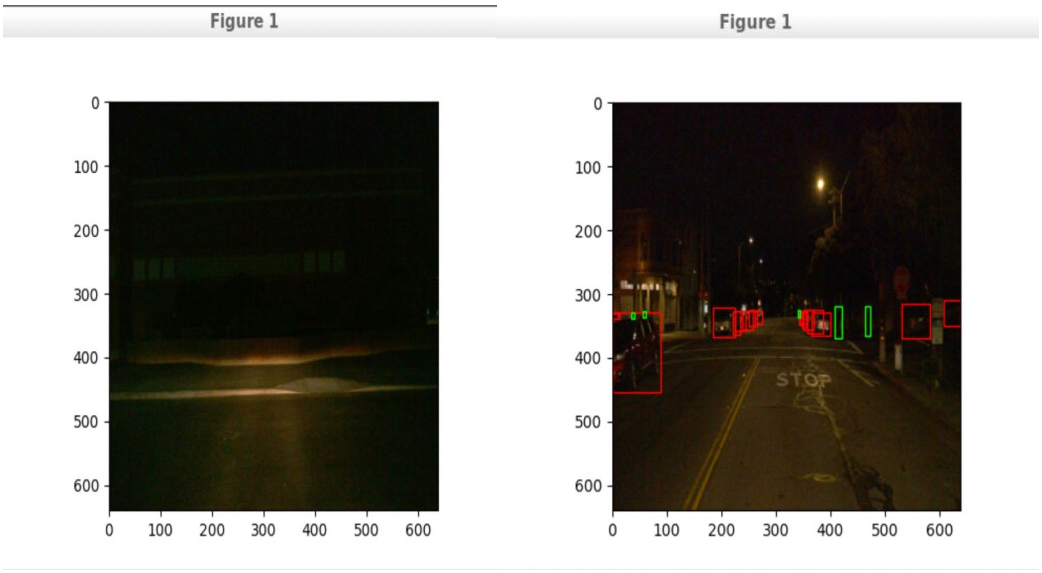


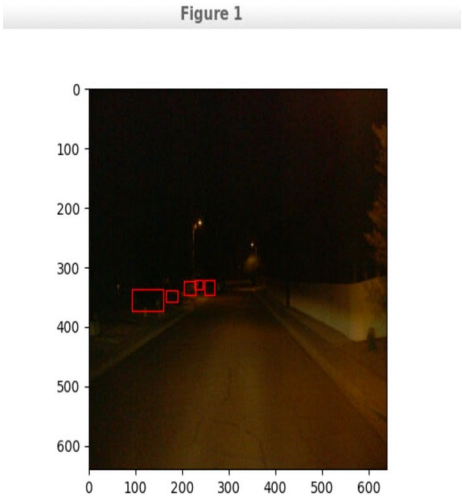
Blurry quality / Bad weather





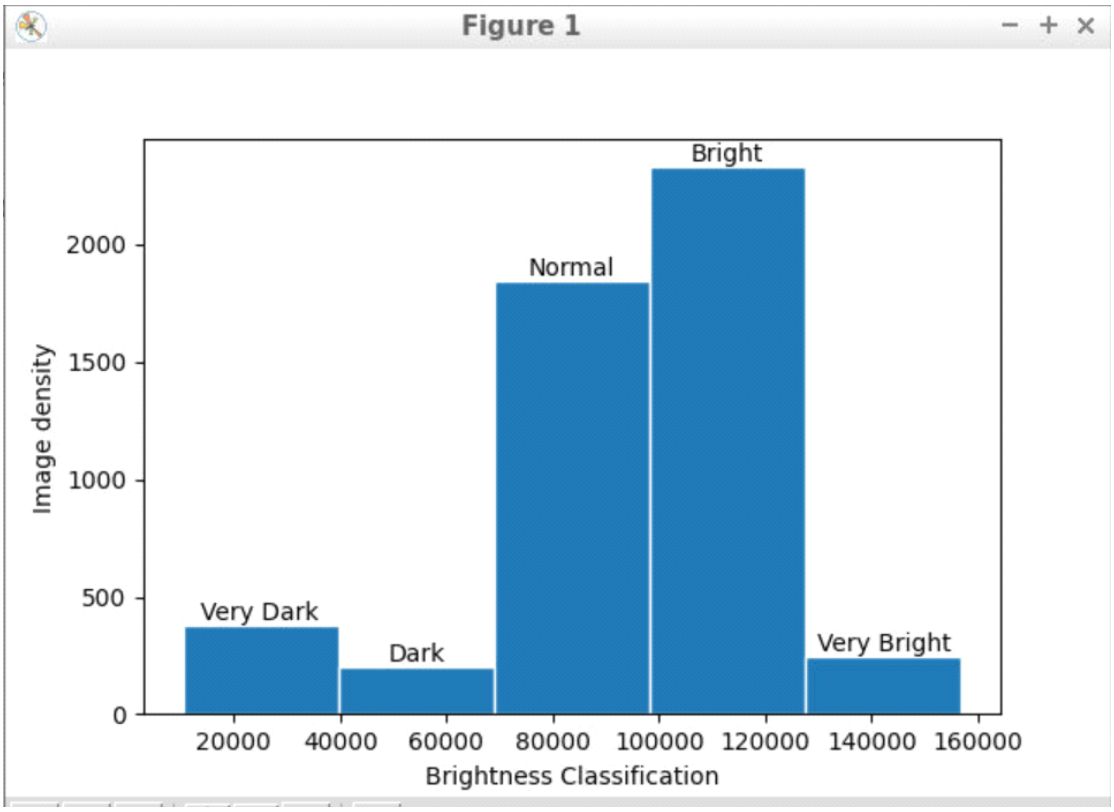
Low brightness images





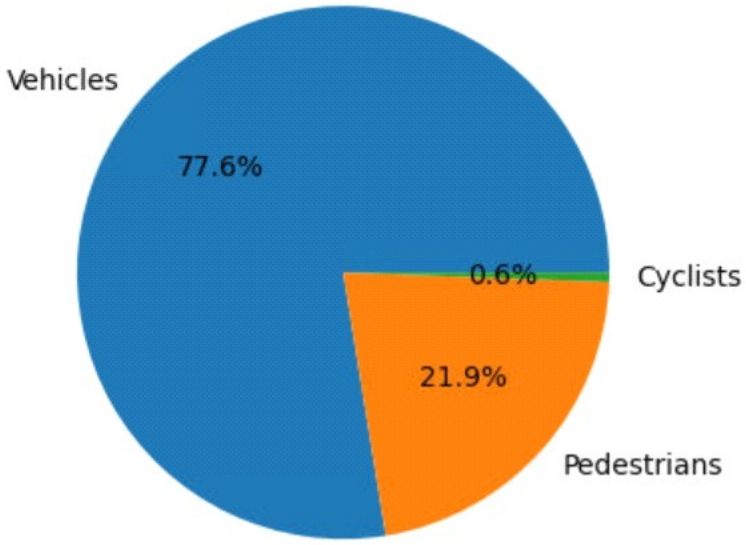
After implementing and running code to study the classification of classes and brightness of images the following was inferred.

i) The average brightness of all pixels for every image in a set of 5000shuffled and randomly picked images was calculated and plotted on a histogram. It was seen that majority of the images had good brightness. This can be seen in the plot below.

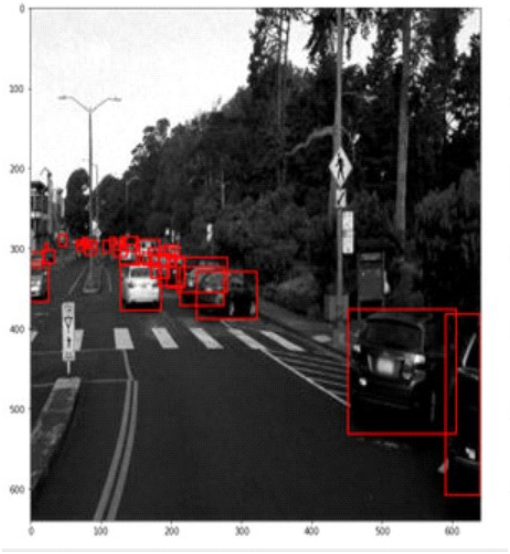


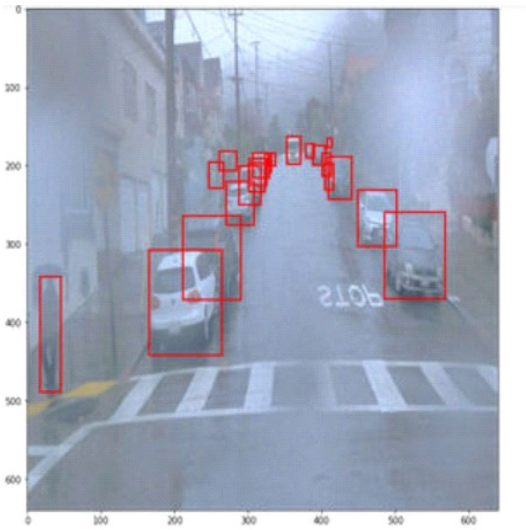
II) A second study was done on the distribution of classes. The images were shuffled and the distribution of classes on a set of 10000 images was evaluated. The vehicles clearly dominated at 77.6%, the pedestrians at 21.9% and cyclists at a mere 0.6%.

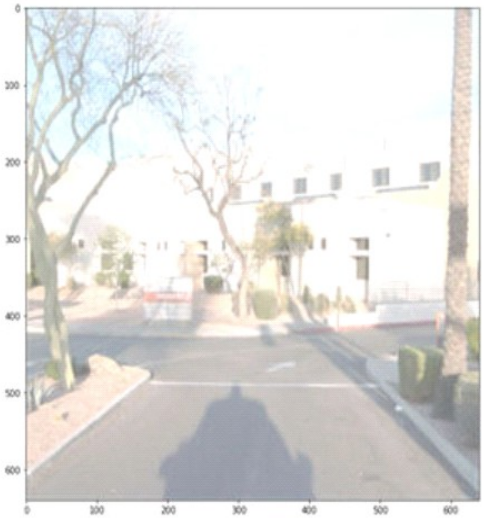
Figure 1

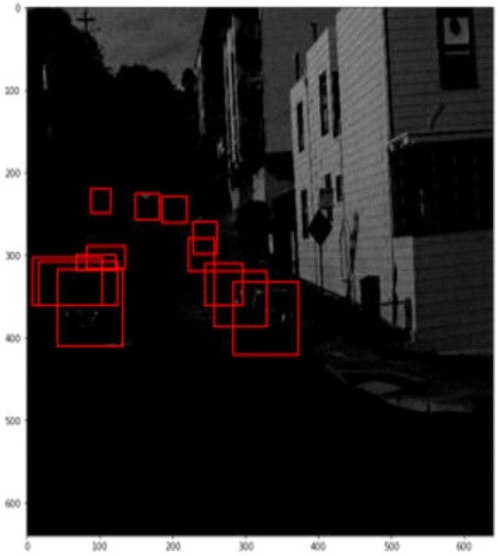


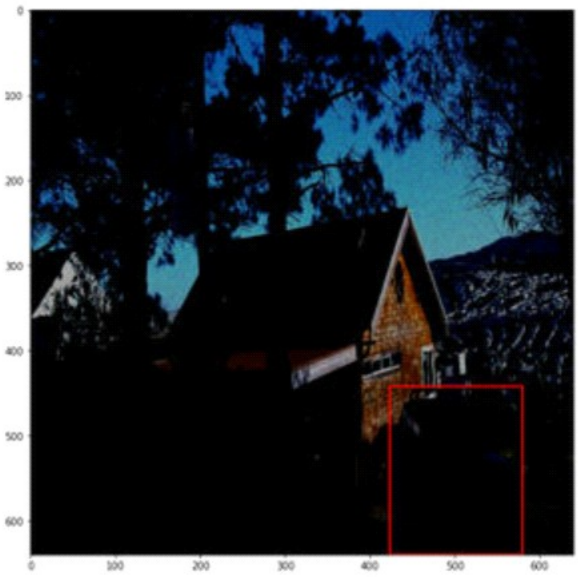
With the following understanding, augmentations were applied to further diversify and balance the data. Random horizontal flip was applied to avoid any feature dependence. Random adjust brightness and random adjust contrast were applied based on the study regarding brightness as in the histogram. Random adjust hue, random adjust saturation, random pixel value scales were added to increase diversification needed for image quality, weather and blurry conditons. Random rgb to gry was added to remove dependency on color and emphasize special features of vehicles, pedestrians and cyclists. A few augmented images have been attached below.











Cross validation

This section should detail the cross validation strategy and justify your approach.

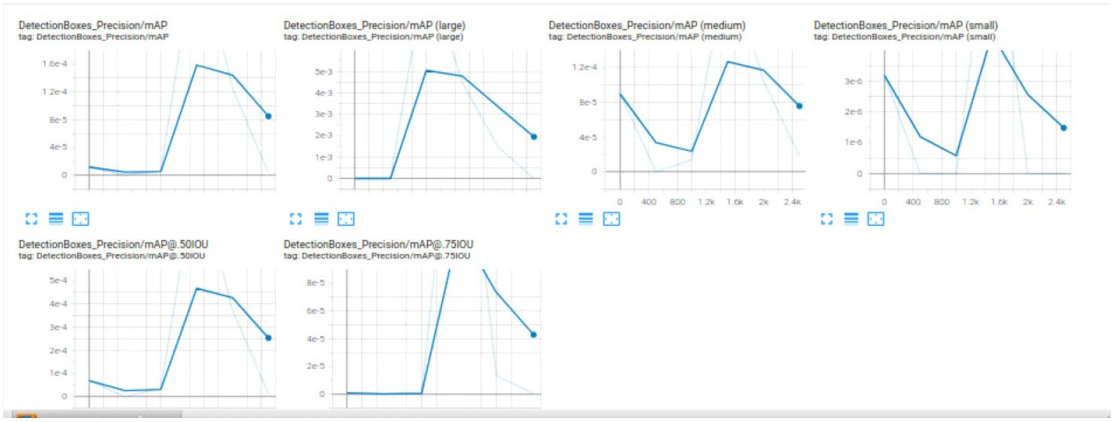
The dataset has already been split and is available in folders train, test and val.

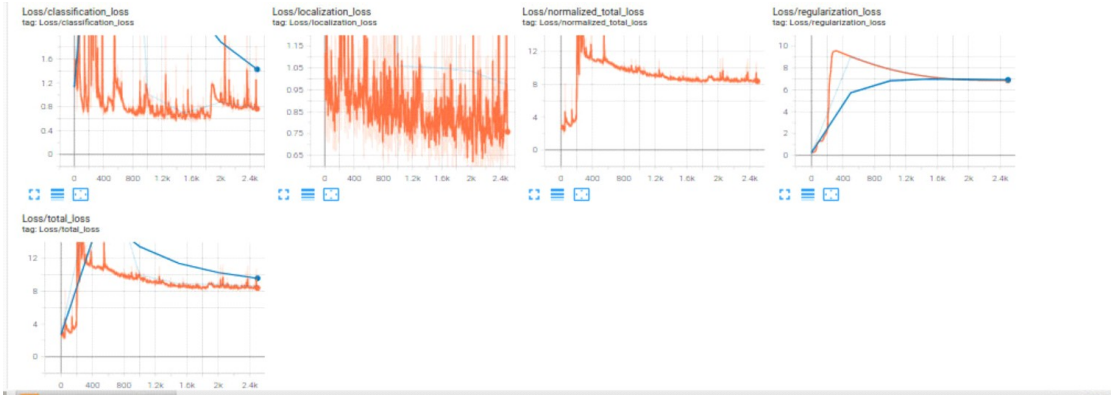
Training

Reference experiment

This section should detail the results of the reference experiment. It should includes training metrics and a detailed explanation of the algorithm's performances.

Tf Object Detection API relies on Config files. The config file used here is 'pipeline.config', which is the config for a SSD Resnet 50 640x640 model. The config file must be changed to include the location of the training and validation files, as well as the location of the label_map file, pretrained weights. The batch size needs to be adjusted. This is done by running a command as mentioned in the instructions. The training is then done using the newly edited config file.This is monitored by running a tensorboard instance. Finally the evaluation process is launched, initially with checkpoint 1 and updated till the last available checkpoint to get a continuous curve. The results are attached below.





The results were not good as expected. The mAP precision is around 8e-5. The training loss is high at around 8 and seems to be saturating.

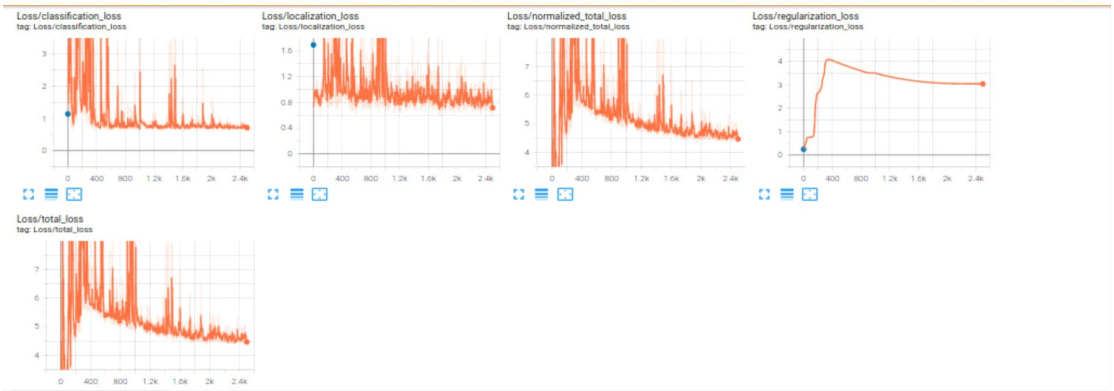
Improve on the reference

This section should highlight the different strategies you adopted to improve your model. It should contain relevant figures and details of your findings.

All the edited pipeline_new config files are stored in respective experiment* folders (E.g. \home\ experiments\experiment1\) along with the Tensorboard screenshots

1)Experiment1

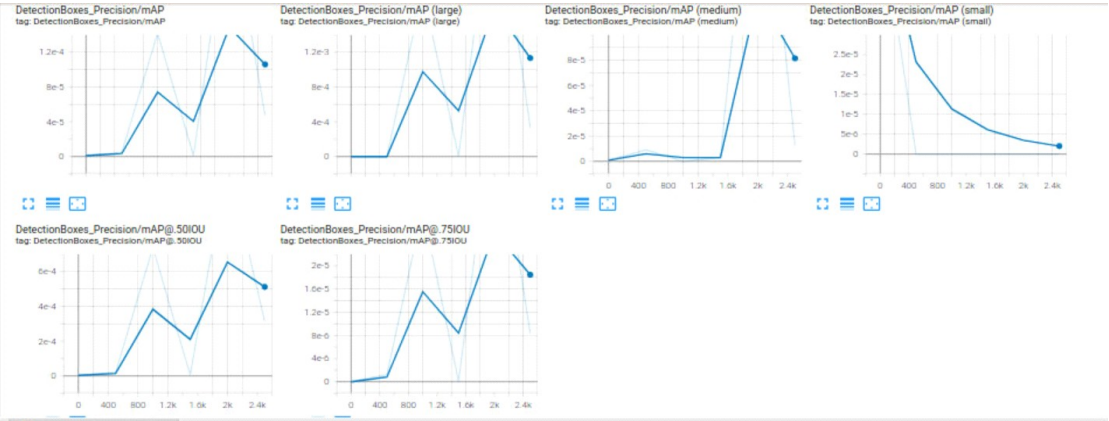
The augmentations for rgb to gry, pixel scale, brightness, contrast, hue, saturation and horizontal flip were added with no other additional changes. The new config file was stored in folder experiment1 and training and evaluation was performed. The results on tensorboard have been attached.

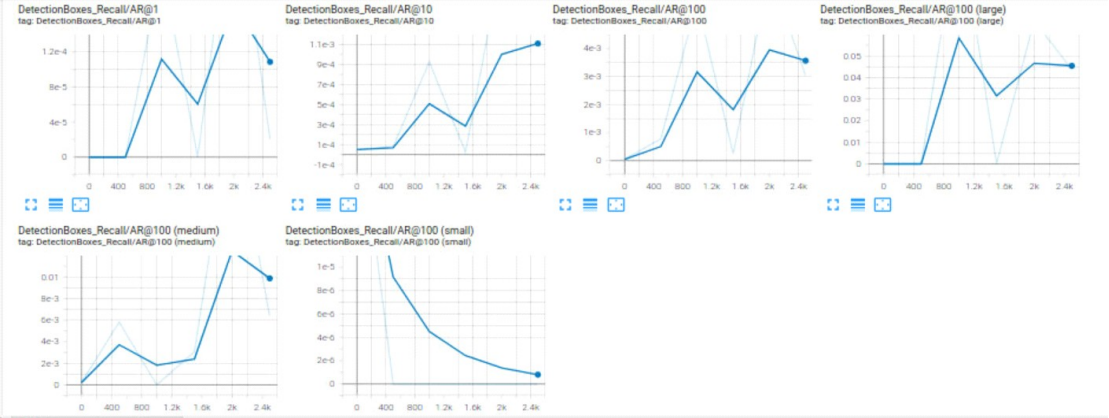


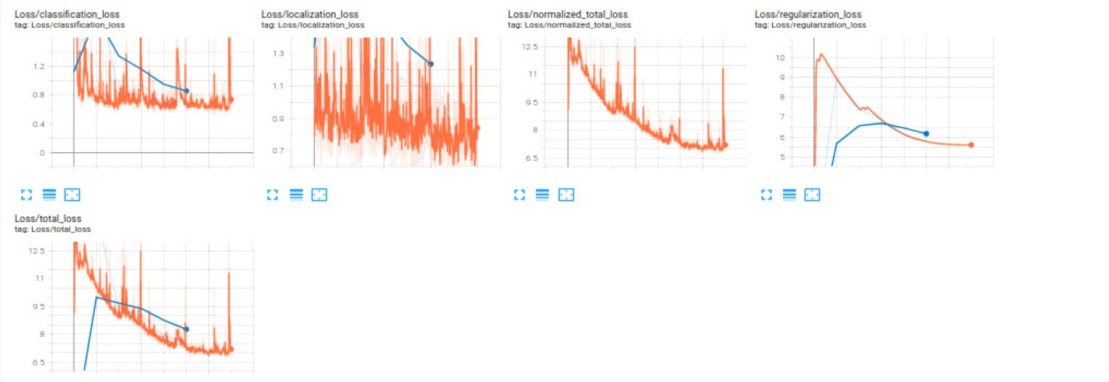
The checkpoints were unfortunately deleted to make space to run next trial before the evaluation and hence the evaluation process did not yield any results. However it can be noticed that there is considerable decrease in losses to around 4.4 and it has still not saturated. Perhaps an increase in the number of steps would further decrease the loss.

2) Experiment2

The parameters in the momentum optimizer field were tweaked and all previously mentioned augmentations included and a new config file was generated. The steps were increased to 3500, learning rate base and warm up learning rate rounded to 0.05 and 0.02 respectively. The tensorboard results have been attached.



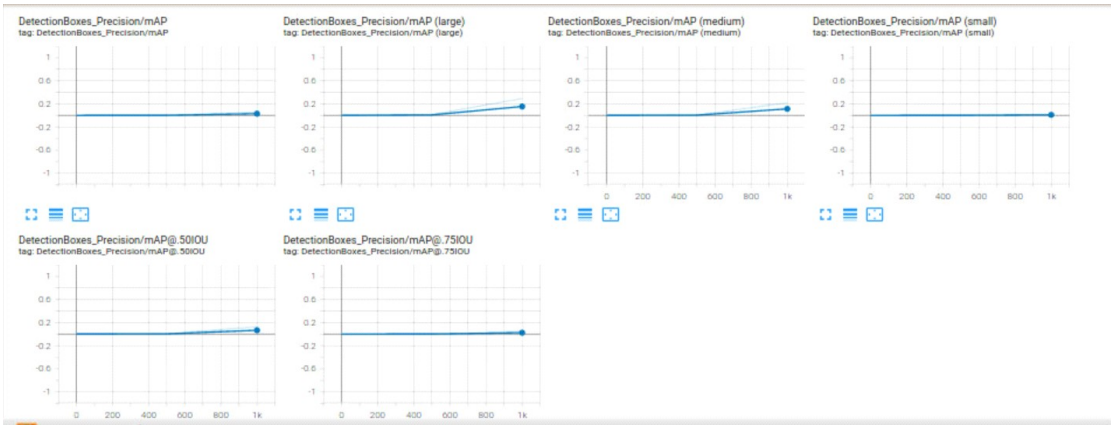


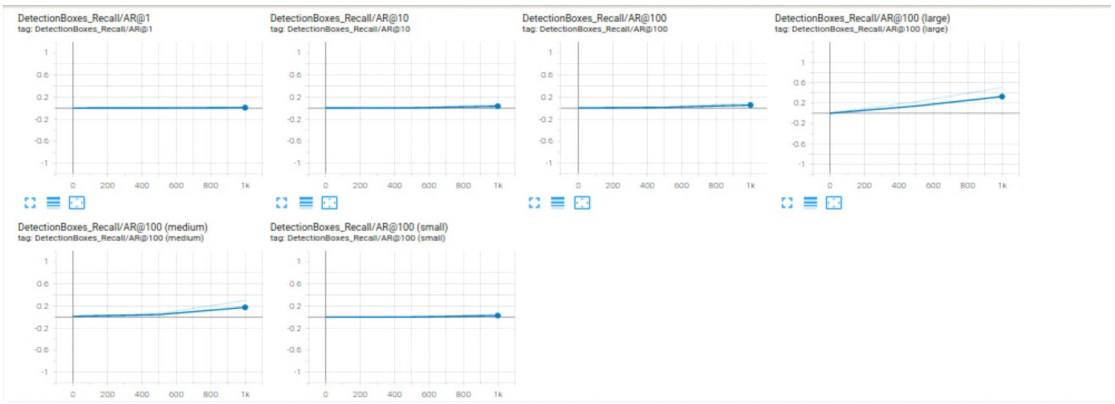


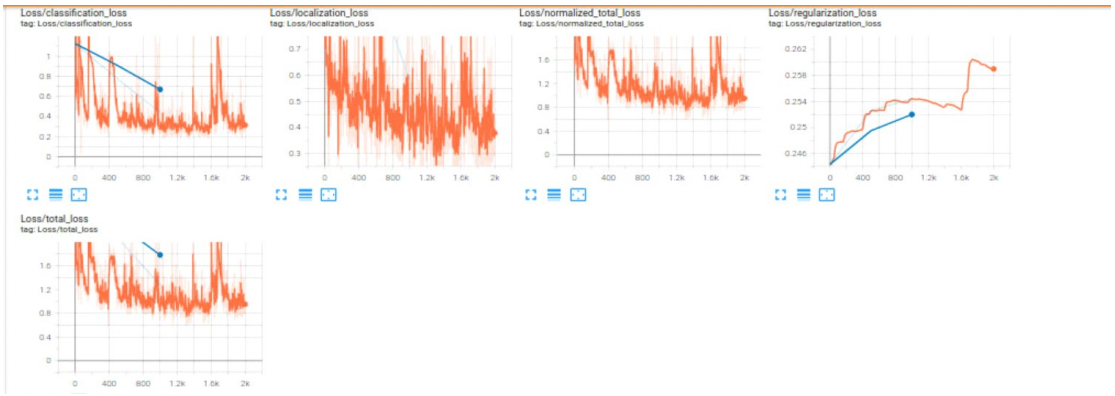
There was an error message saying 'No space available'. Therefore the last checkpoint was deleted before running the evaluation process. It can be also seen that only 2500 steps have been executed due to lack of space. The loss and mAP precision are worser than before and this can be attributed to larger learning rate.

3) Experiment3

The Adam optimizer has been used instead of the momentum optimizer with a learning rate of 0.0001 and number of steps decreased to 3000.

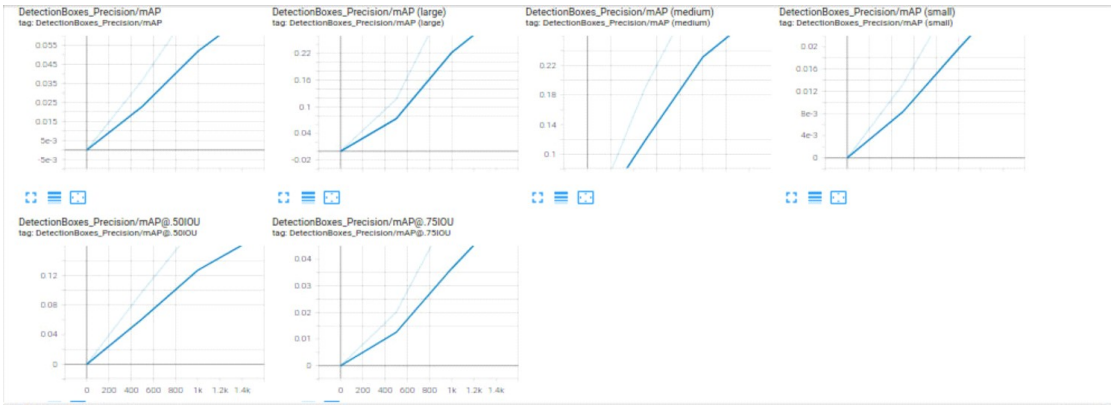


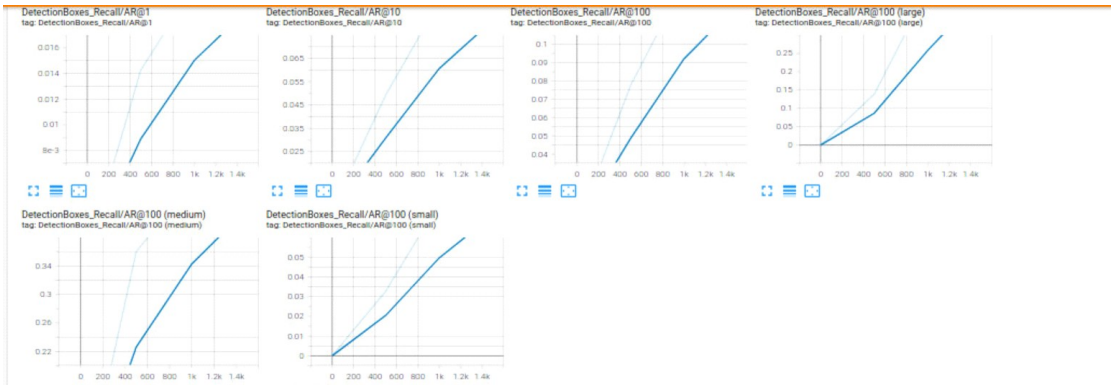


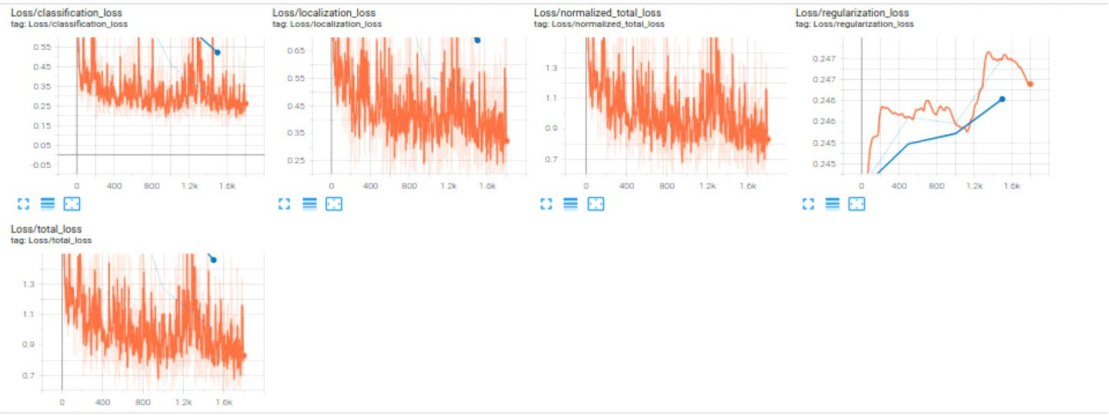


The mAP precision seems to be slightly improved. There is a considerable decrease in loss to values around 0.9. However only 2000 steps have been run due to lack of space. A run with higher number of steps seemd to be benefecial in further decreasing the losses. In this case inorder to run the evaluation the first checkpoint was deleted to make space, but it can be seen that despite this evaluation has not run successfully till the end.

4)Experiment4



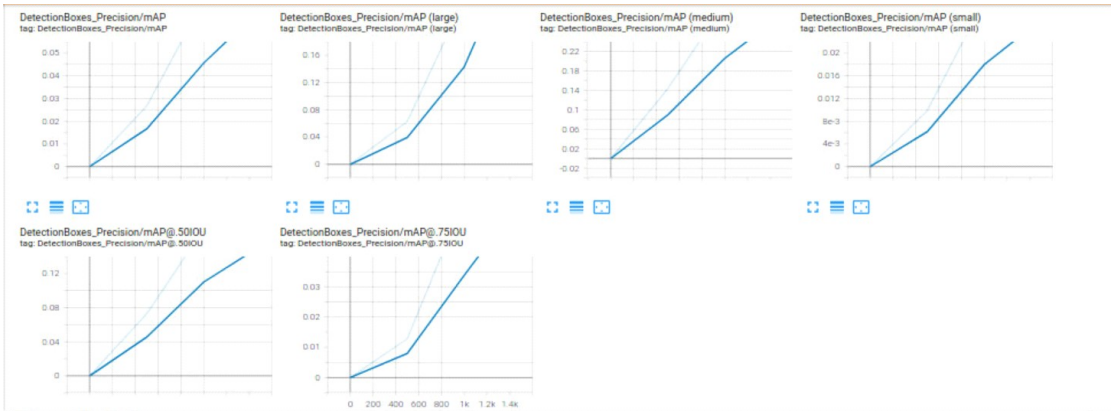


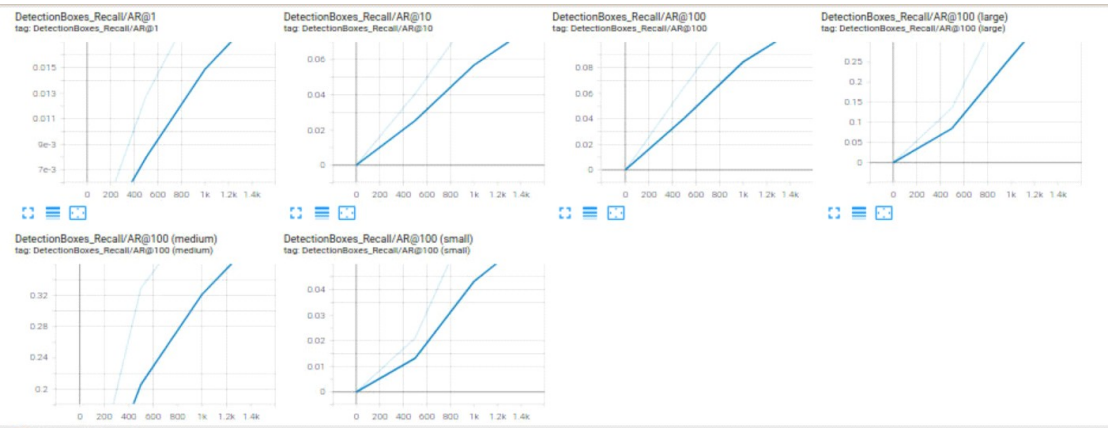


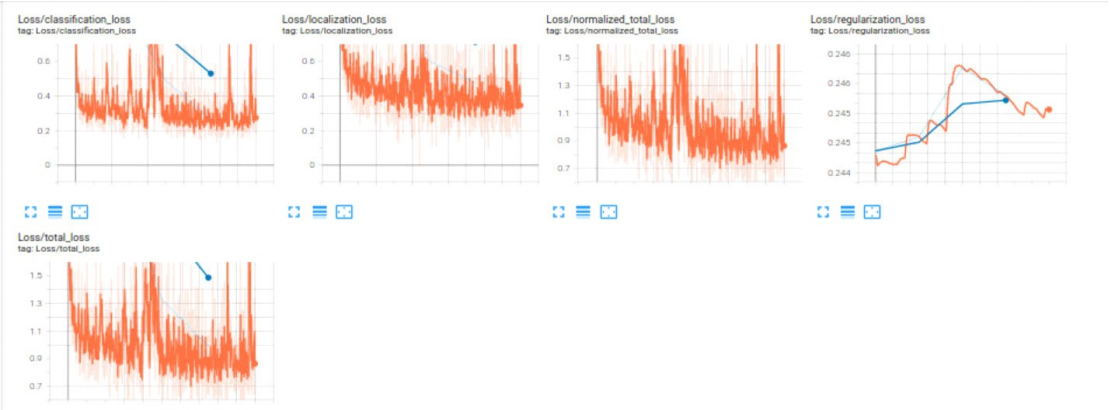
The adam optimizer has been utilized again with a decreased learning rate of 0.00005 and the steps have been decreased to 1800. The number of epochs have been increased to 2. The new config file has been stored in the folder Experiment4. It can be seen that the loss is now around 0.8 and increasing the number of steps can further improve the results. There is also consistent increase in mAP precision and mAP recall.

5)Experiment5

The learning rate in the adam optimizer has been reduced to 0.00003 as there seems to be oscillations in the losses in the earlier results. The steps have been increased to 3000 but the number of epochs have been decreased to counter the memory loss. Furthermore all earlier experimental results were deleted to improve performance and memory.







There does not seem to be much change in the result with it even being slightly worse than the previous results . The lack of memory poses a problem this time as well. However due to constraints in GPU usage time, further experiments were not possible.

Inference:

There are many features that can be tweaked in the pipeline_new.config file. Other augmentations available in preprocessor.proto can be used. There are plenty of models available in the TensorFlow 2 Detection Model Zoo. Of particular interest would be the YOLO algorithm, however this has not been mentioned in the model zoo. A study needs to be done to know which model would yield optimal results in this scenario. The training can also be done with different combinations of activation function, optimizers and loss functions. The batch size, epoch number, regularization, learning rate can also be tuned to get optimal results.