Object Detection in an Urban Environment Project

Project overview

The objective of this project was to utilize a convolutional neural network (CNN) on data from Waymo to detect and classify objects, namely cyclists, pedestrians, and vehicles. This task is important within the context of self-driving car systems because said systems need to be able to construct a model of the surrounding environment (i.e., perceive) so as to be able to move about safely and effectively in the world.

Set up

The procedure taken in this project were those given in the *Project Instructions (Workspace)* sub-section of the final ***Object Detection in an Urban Environment*** section.

Dataset

Dataset analysis

The *Exploratory Data Analysis* notebook was used to display the images. These images showed objects during the day and at night, of different sizes, sometimes close together, and occasionally blurred/hazy. These observations were used to inform the data augmentation strategy.

Cross validation

The hold-out cross-validation approach was employed splitting the 100 records into training, validation and testing sets in the ratio of 87:10:3. This means that at no point does the training model have access to the validation and test data. Moreover, given the time constraints of this exercise, hold-out validation has the advantage, in general, of only needing to train once. Other cross validation methods exist, such as leave one out (LOO) or k-fold cross validation. However, they are generally more expensive and time-consuming to train. Given the already heavy computational load of Deep Learning algorithms they are less suited.

Training

Reference experiment

Following the instructions outlined, a reference model was trained. As shown in Figures 1a-c, the loss was around 4.5, with mean average precision and recall values very close to 0.

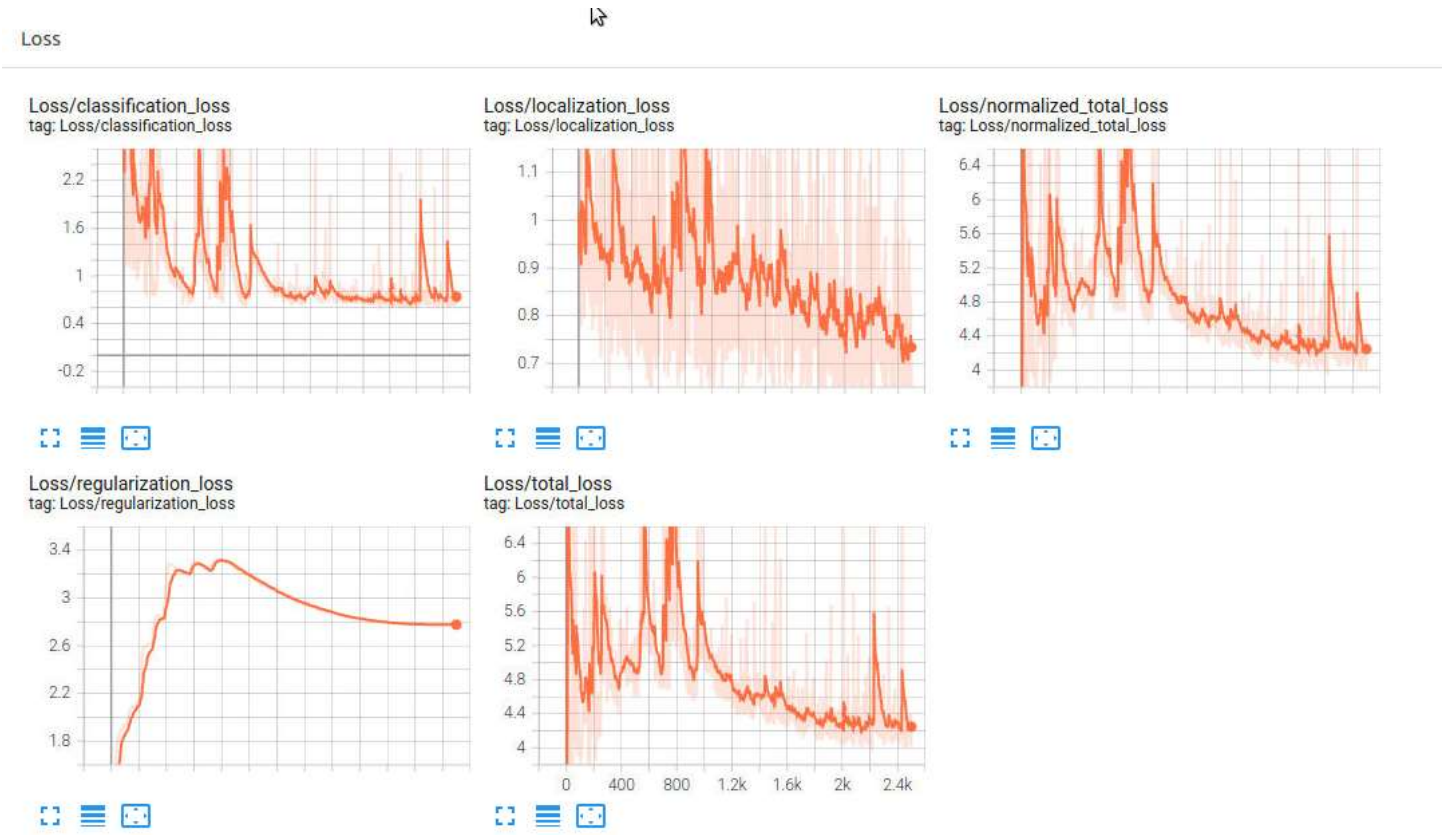

Figure 1a: Training loss of reference model

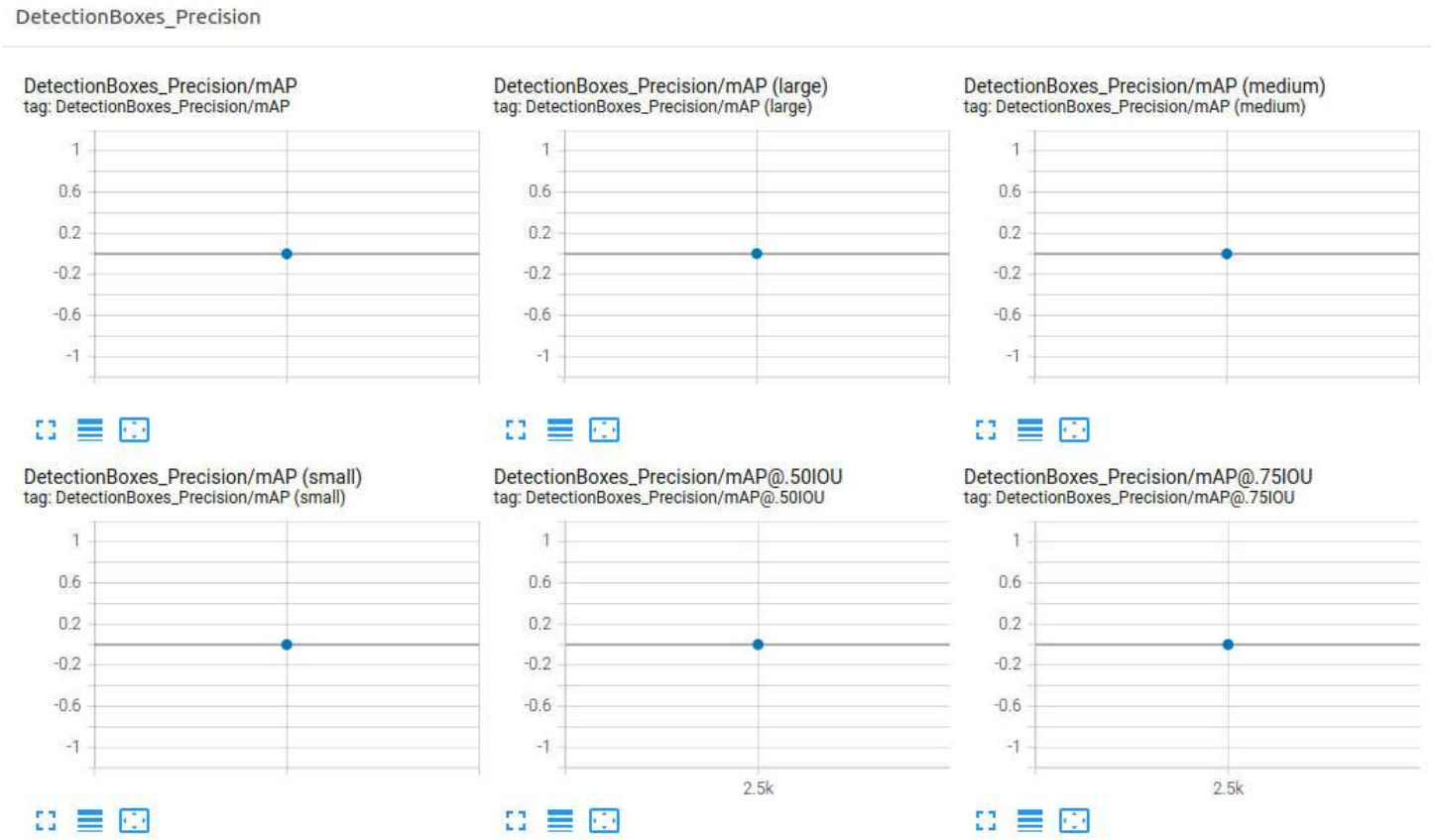

Figure 1b: Mean Average Precision of reference model

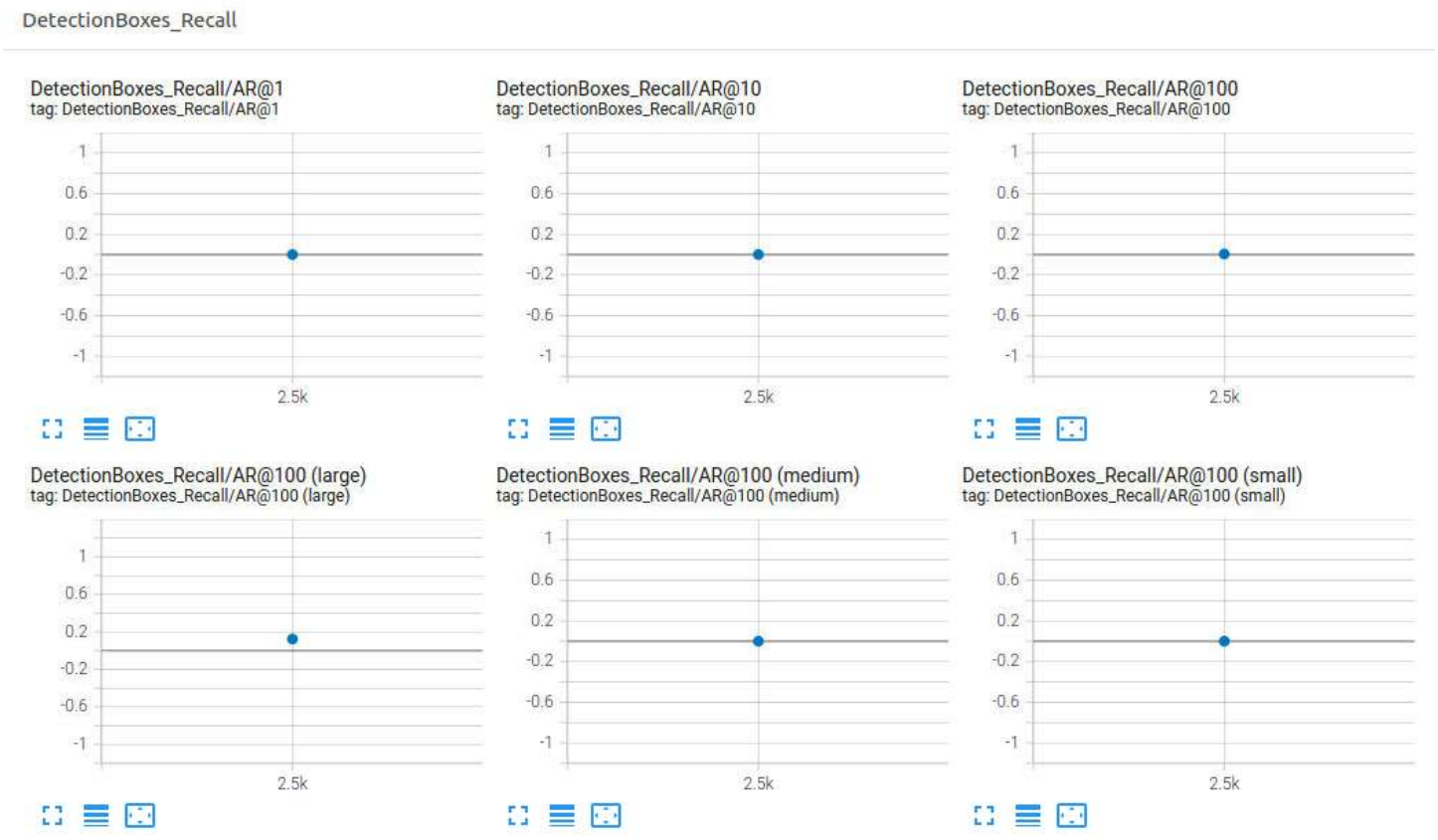

Figure 1c: Recall of reference model

Improve on the reference

The first attempt at improving the performance of the model was to employ augmentations. The following were utilized:

(i) random_horizontal_flip, given that an object could appear on either side of the frame and facing in opposite directions.
(ii) random_adjust_brightness, given that objects could be viewed at different times of the day or night.
(iii) random_crop_image, as this has been reported to improve the performance when detecting small objects.

As shown in Figures 2a-c, the loss was only slightly lower and the precision and recall about the same or slightly higher.

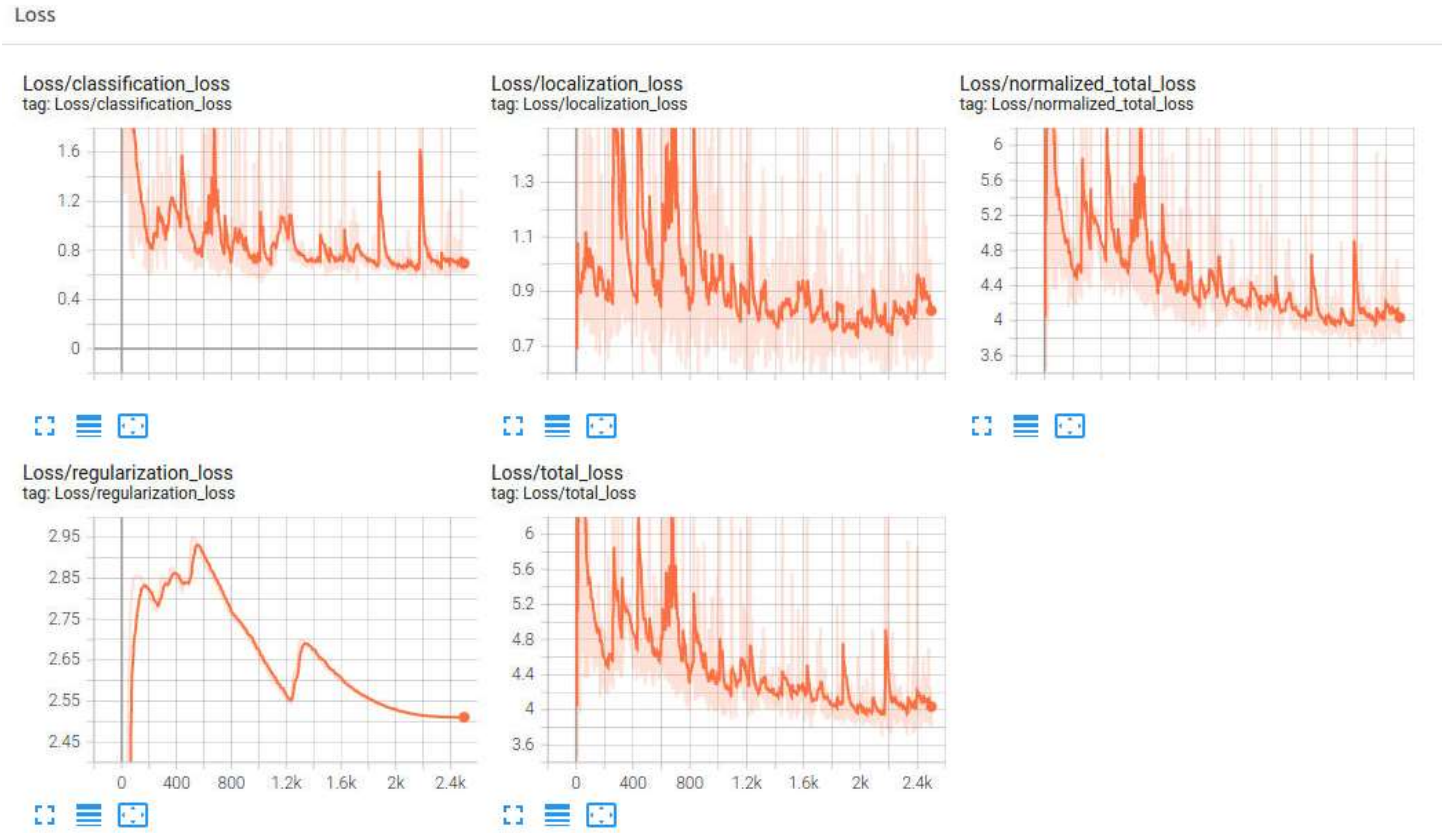

Figure 2a: Training loss of augmented model

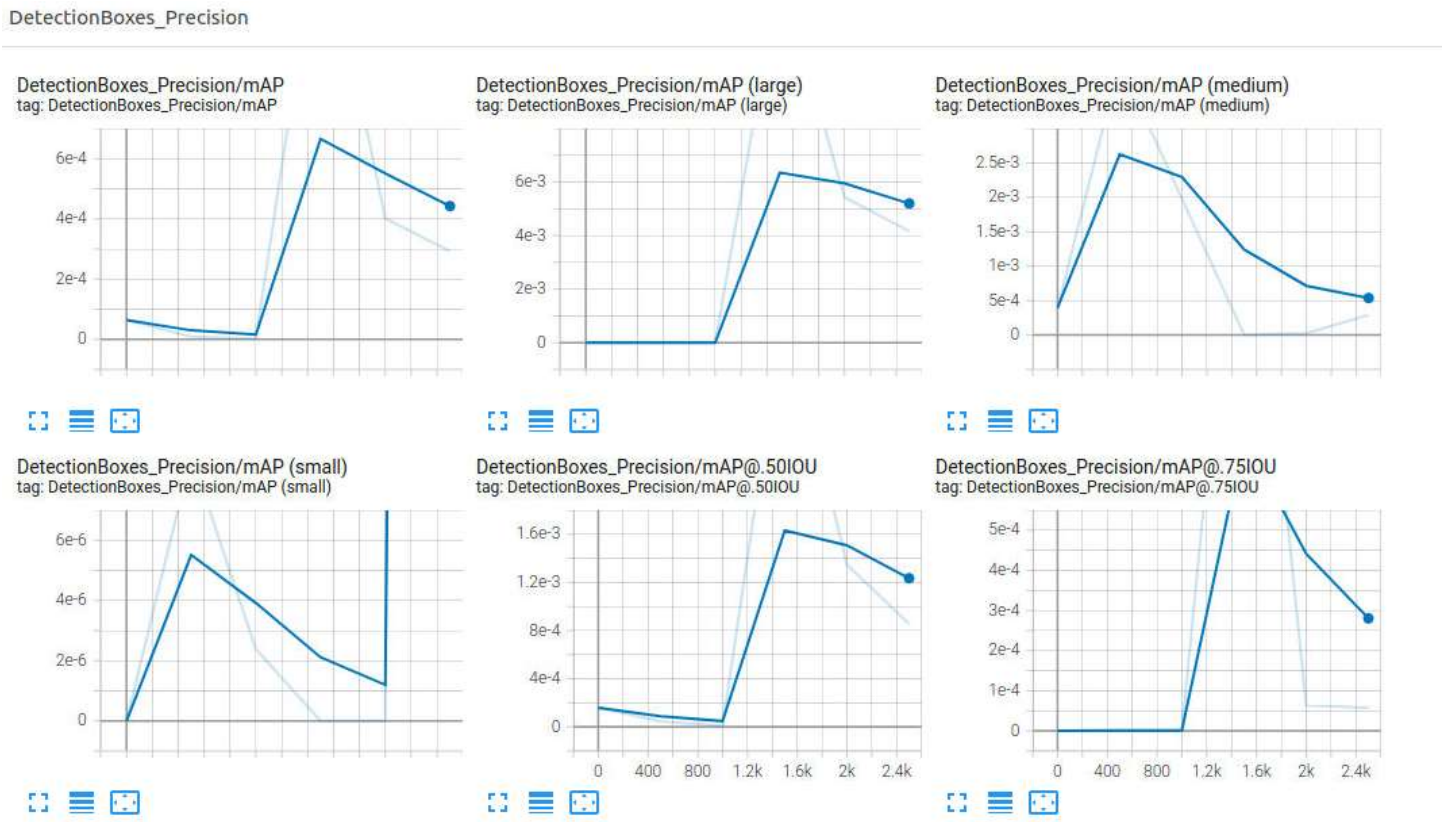

Figure 2b: Mean Average Precision of augmented model

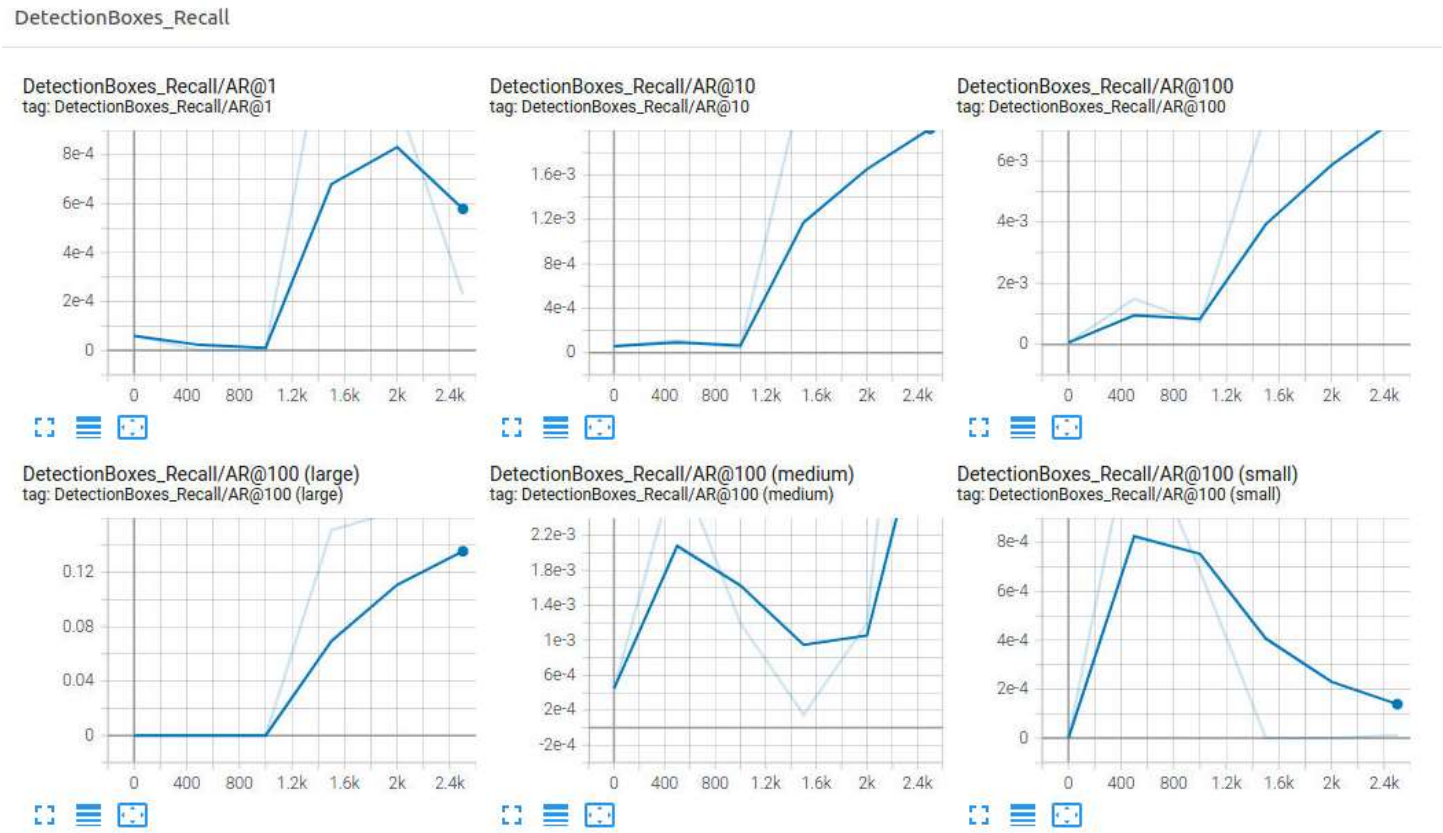

Figure 2c: Recall of reference model

The second change sought to address the high variability in the losses during training. This indicated a lot of jumping around during gradient descent. To mitigate this, the batch size was increased from 2 to 4. In this case, there was a marked improvement, with almost a 1 point drop in the loss, and an increase in some of the precision and recall metrics.

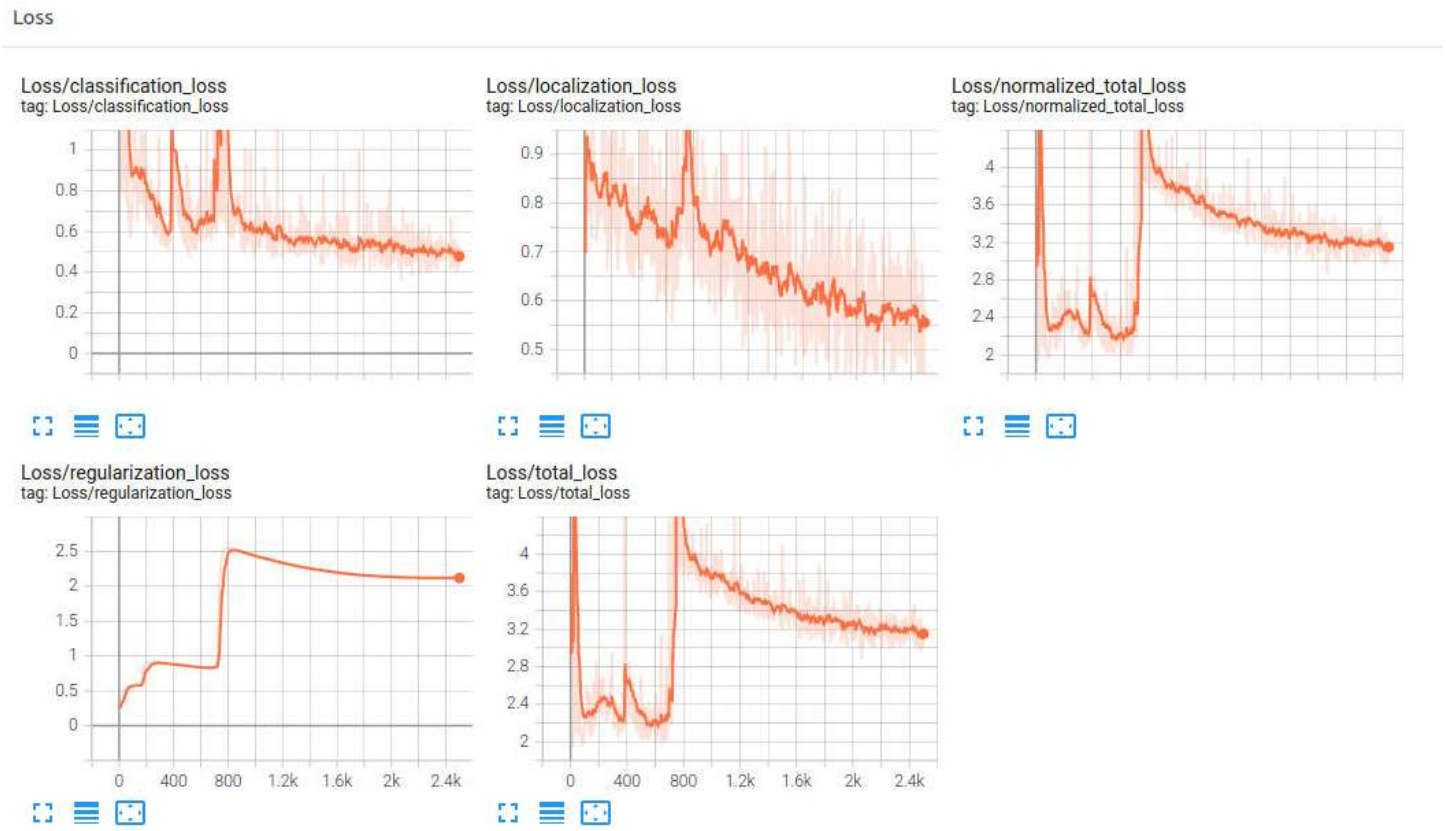

Figure 3a: Training loss of augmented model with increased batching

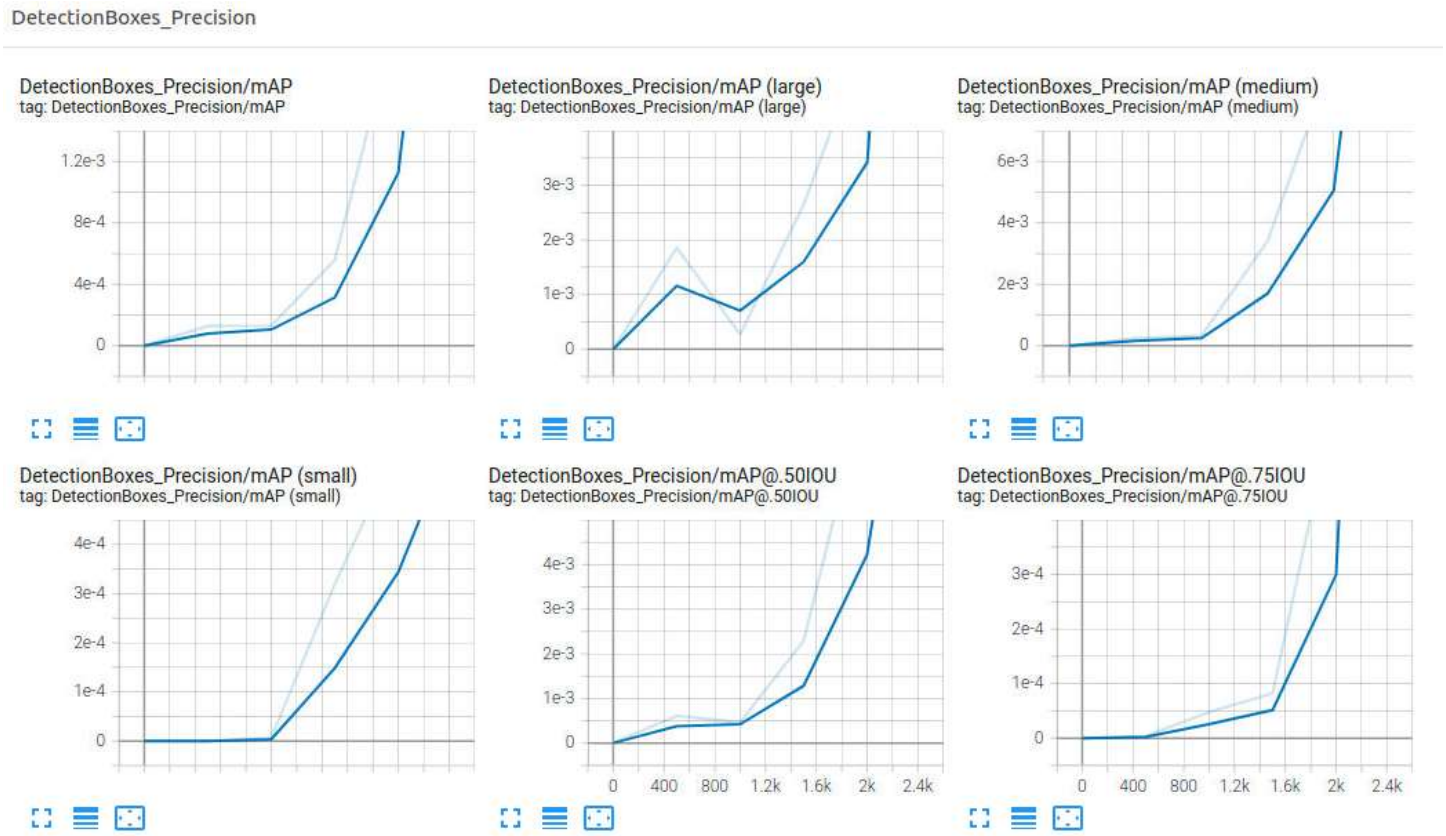

Figure 3b: Mean Average Precision of augmented model with increased batching

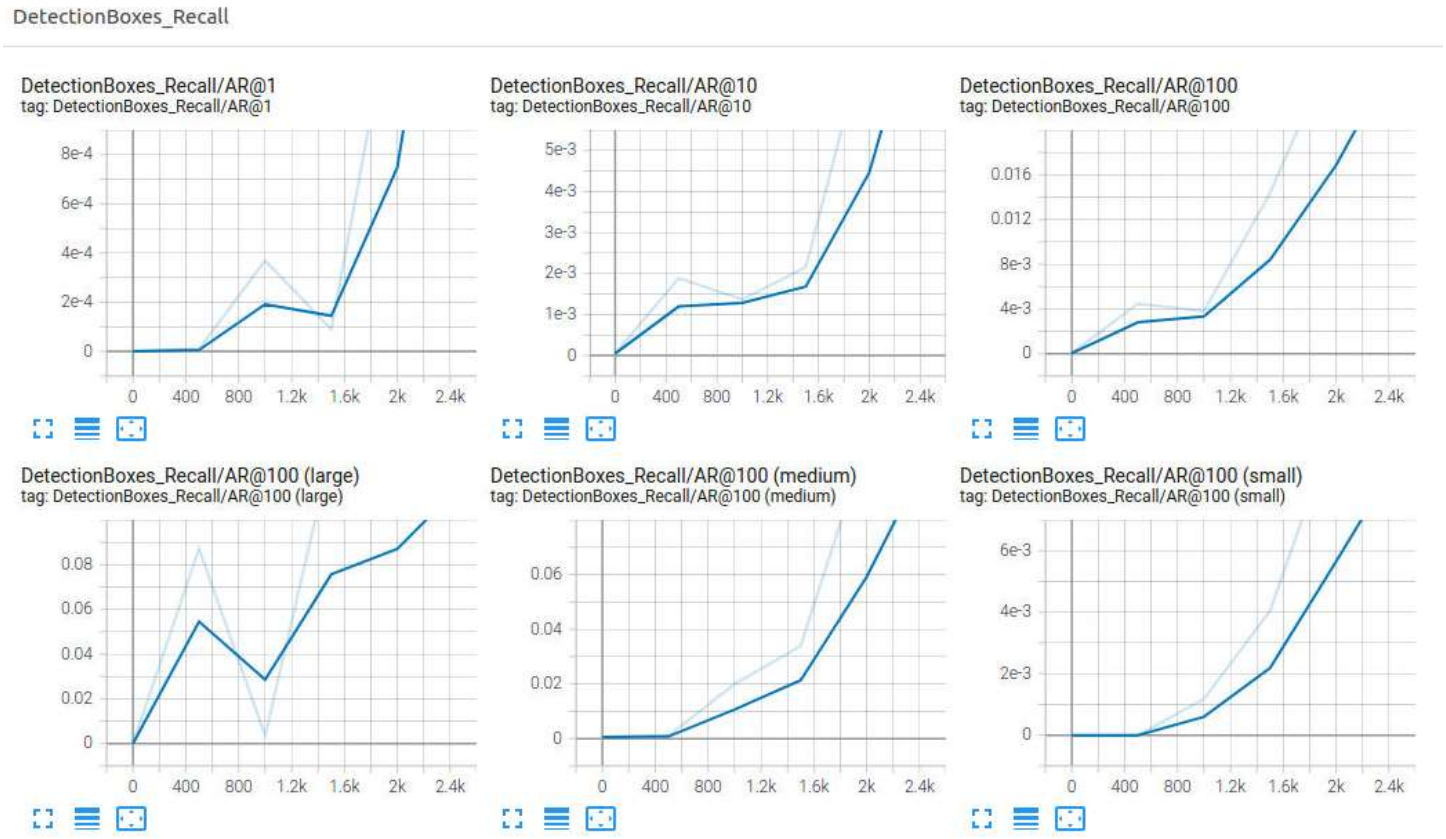

Figure 3c: Recall of augmented model with increased batching

The third change was to determine if using the popular (owing to its excellent performance on a wide range of tasks) ADAM optimizer rather than that of the momentum would yield better results. Figure 4 showed that this was not the case as the loss was reaching a plateau well above the previous model at above 5.

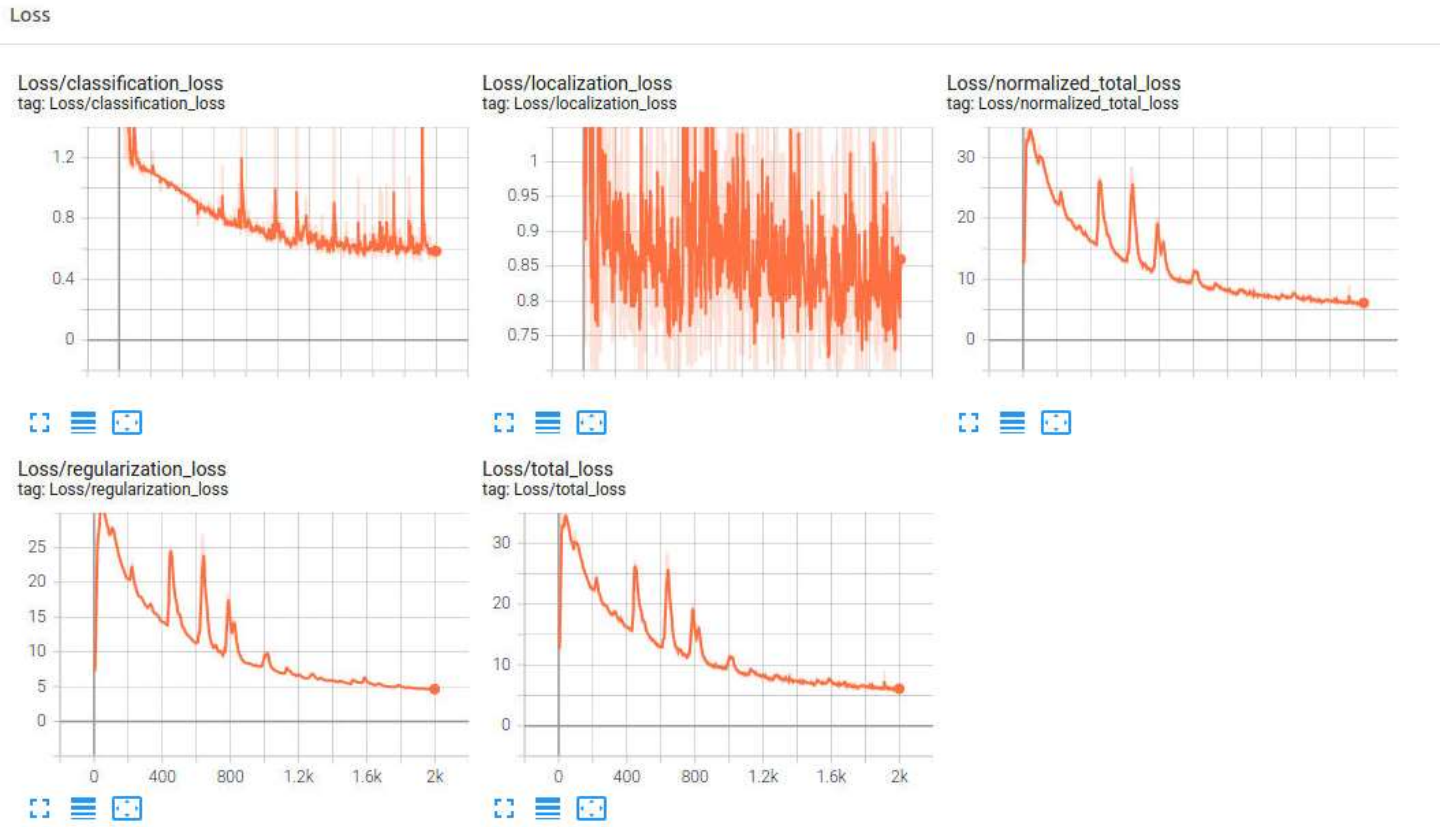

Figure 4: Training loss of an Adam-optimised augmented model with increased batching

Judging that the poorer performance of ADAM might have been due to the high initial loss and a slow drop-off, a change was made to the learning parameters in an attempt to drive the loss down more quickly. Specifically, the base learning rate was increased from 0.04 to 0.1 and the number of warmup steps was increased to 500. The idea was that the model would drive the loss function down faster over a longer initial period, hopefully enough to allow the loss to approach that of the current best performing momentum with augmentation and increased batching. This was achieved, as demonstrated in Figure 5a. However, this did not carry over to the evaluation as performance deteriorated across all measures (Figure 5b and c).

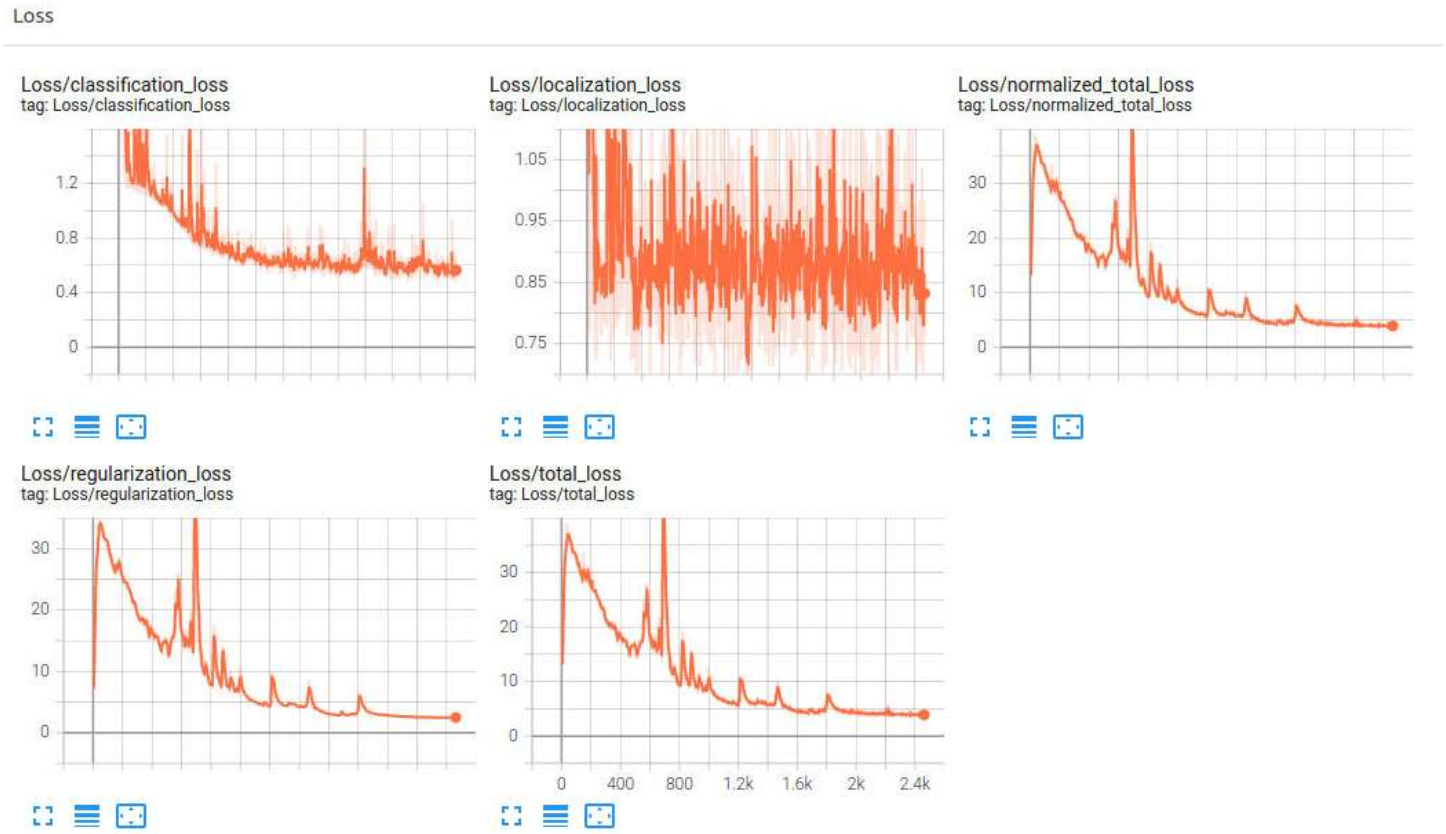

Figure 5a: Training loss of an Adam-optimised augmented model with modified learning parameters and increased batching.

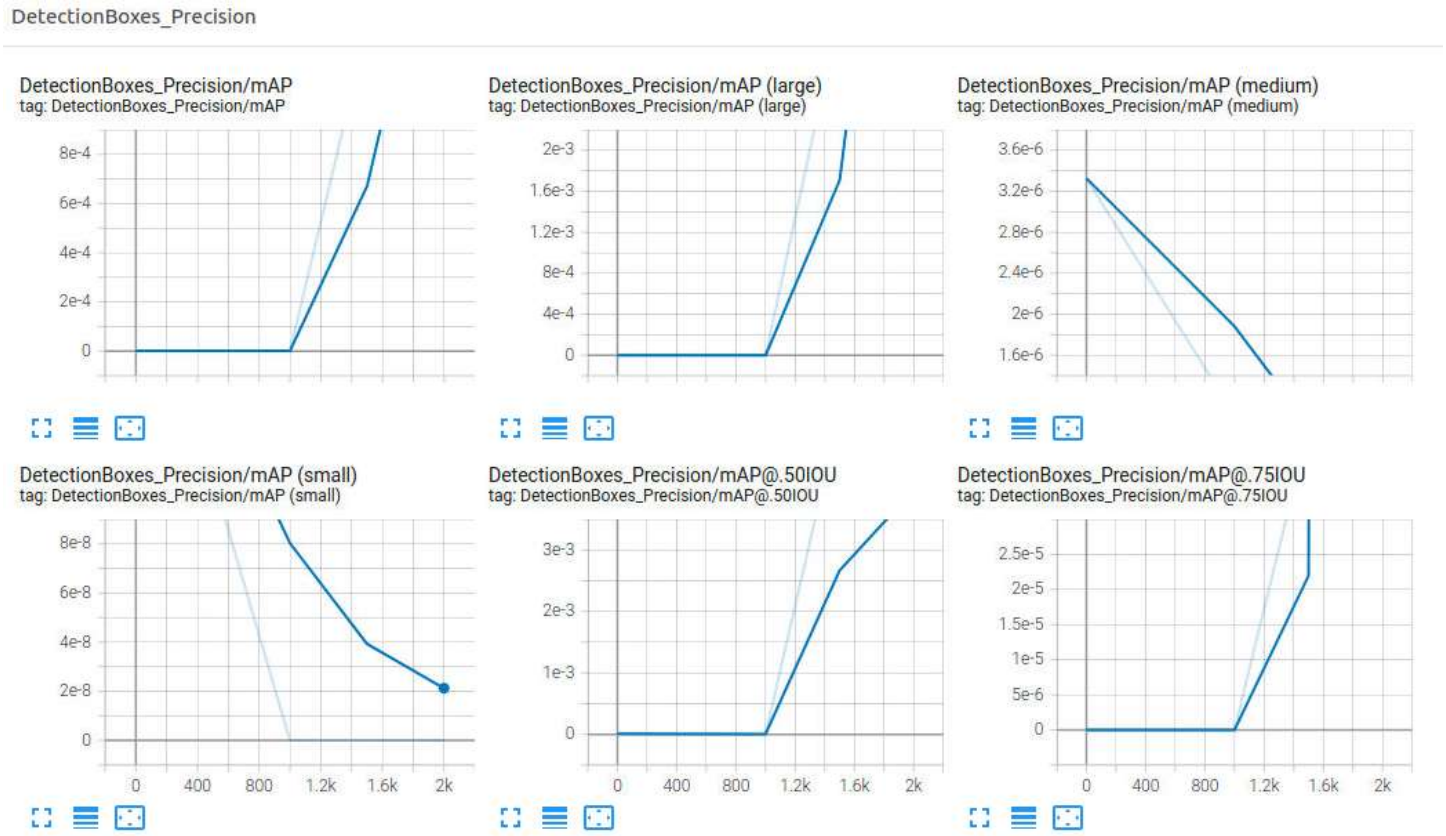

Figure 5b: Mean Average Precision of an Adam-optimised augmented model with modified learning parameters and increased batching.

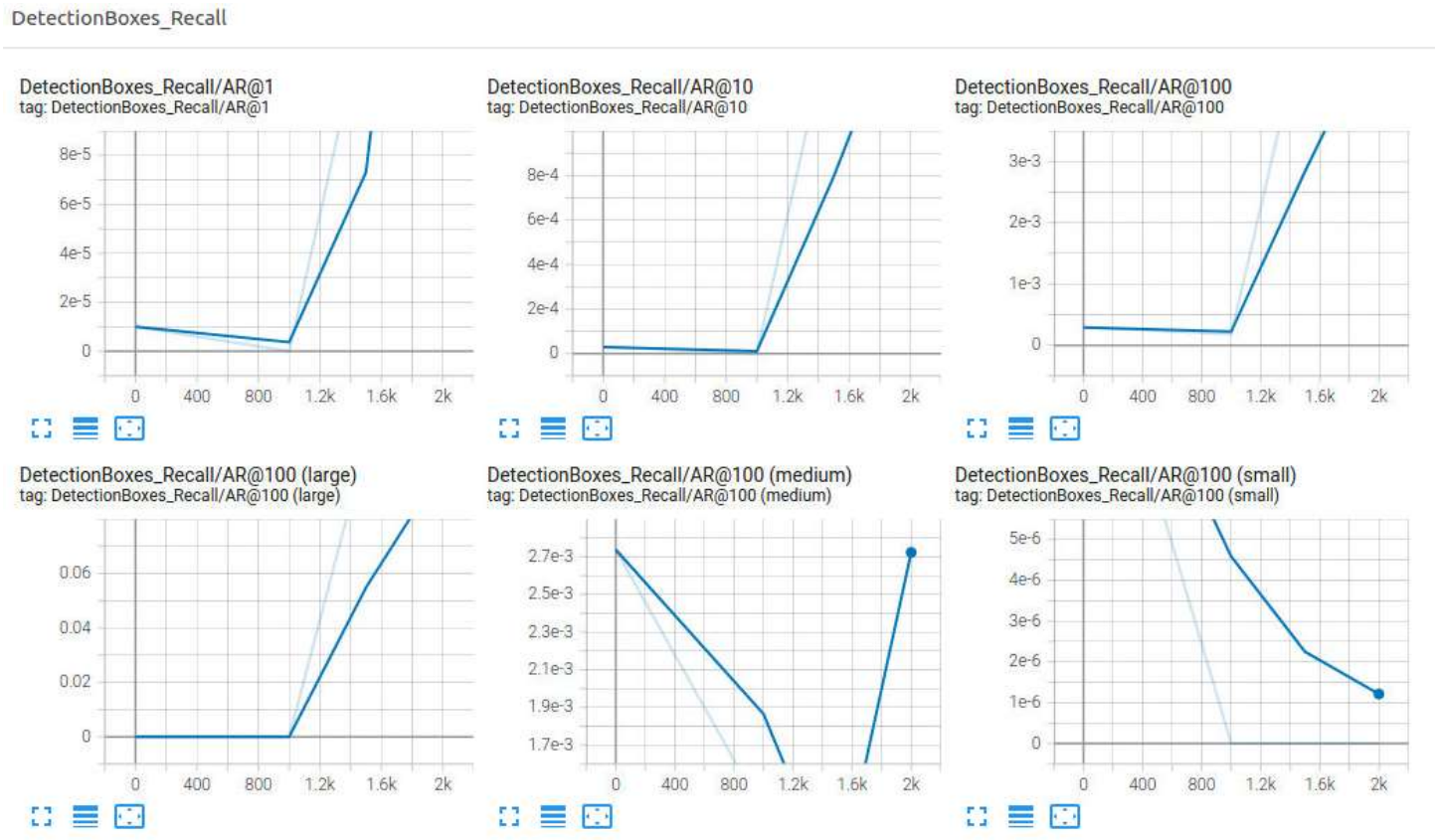

Figure c: Recall of an Adam-optimised augmented model with modified learning parameters and increased batching.

Since Single-shot and other YOLO algorithms struggle with high concentrations of small objects, our current model might have issues with the objects that are close together. Thus, I sought to use a Fast RCNN but encountered issues downloading the tarball as shown below:

```
(sdc-c1-gpu-augment) root@eb85a4985661:/home/workspace/experiments/pretrained_model# wget http://download.tensorflow.org/models/object_detection/tf2/20200711/faster_rcnn_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
--2022-06-30 00:11:50--  http://download.tensorflow.org/models/object_detection/tf2/20200711/faster_rcnn_resnet50_v1_fpn_640x640_coco17_tpu-8.tar.gz
Resolving download.tensorflow.org (download.tensorflow.org)... 74.125.124.128, 2607:f8b0:4001:c14::80
Connecting to download.tensorflow.org (download.tensorflow.org)|74.125.124.128|:80... connected.
HTTP request sent, awaiting response... 403 Forbidden
2022-06-30 00:11:50 ERROR 403: Forbidden.
```

A final consideration was the type of learning rate scheduler. Although the scheduler parameters were modified during testing, the scheduler itself, a cosine decay, was not changed as it provides a smooth decay function rather than the jumps that would be experienced by a scheduler such as the step decay.

Testing

The best performing model was that with augmentation and increased batching and so it was used to create the movies of the test data.

The movies as well as the exploratory notebook are to be found within the same repository as this document.