Object Detection in an Urban Environment Writeup

1) Project overview

Object detection techniques are an important part of self-driving car applications. Detecting and localizing the object around cars help to reduce the number of accidents in Self-driving cars. This project is the fifth project in the Data Scientist nano degree sponsored by BOSCH. In this project, we will use TensorFlow API to detect objects in urban environments. This API simplifies the training and development of object detection models in TensorFlow. We will use the Waymo dataset which is used by many practitioners to provide the best models for self-driving car applications. You can find this data at (https://waymo.com/open/).

2) Dataset

As mentioned above, we will use the Waymo dataset. In brief, we will analyze the dataset and then split it into training, validation, and test sets based on the cross-validation method.

We were provided with a dataset of images of urban environments containing annotated cyclists, pedestrians, and vehicles.

We performed an extensive data analysis including the computation of label distributions, display of sample images, and checking for object occlusions to get some helpful information about the dataset.

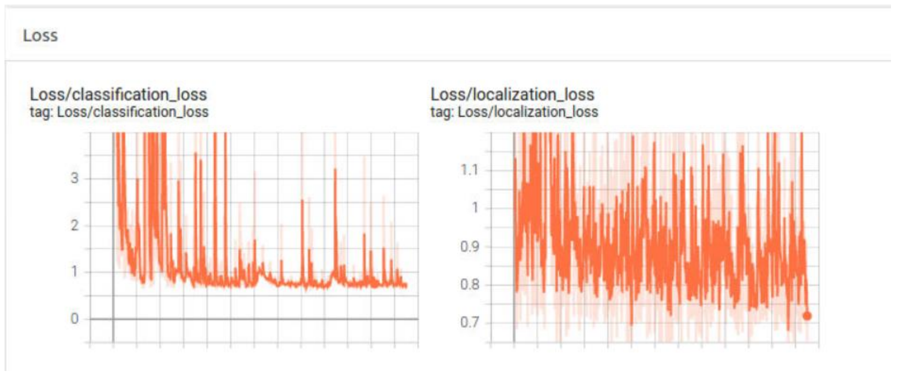
3) Cross-validation

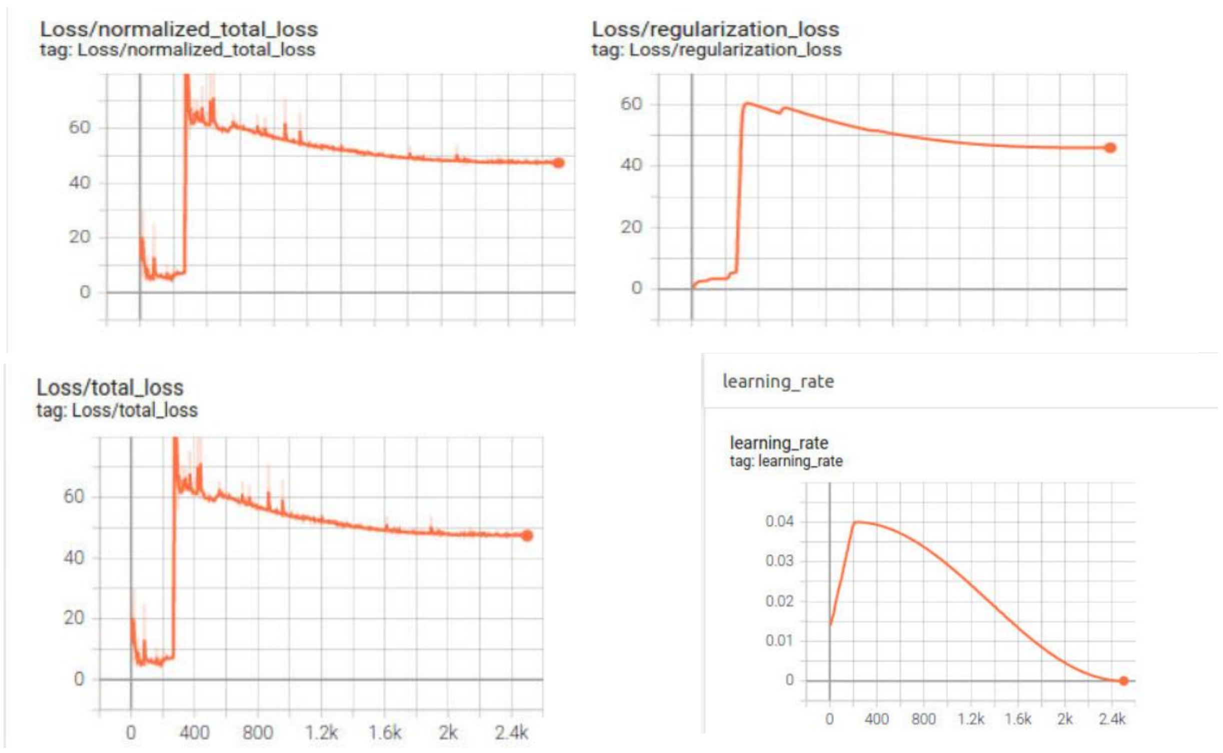
The data were split into three sets, Training, Validation, and test sets. Each set has its own folder containing its corresponding images split from the processed Waymo open data. The Training set is used to train the model, while the validation set is used to validate the model on the specified hyperparameters, and when we decide to work on a model, we test it using the test set.

4) Training

The first experiment (Experiment 1)

In the first experiment, we performed an object detection task using the specified configuration in the `pipeline_new.config` file. Hers is the result from the tensor board





From the previous visualization, we could conclude that the model has little ability to generalize well,

The training loss and the evaluation loss were high, the training Loss was (47.367) and the evaluation loss was (47.898678).

The training model needs to be trained on more images

One of the most common ways to get more images is data Augmentation,

Data Augmentation is used to obtain a different image from the same image by either changing the orientation, color, and brightness or even cropping the image and adding noises. This will give the model the chance to learn about more diverse data needed to be more accurate when predicting objects in unseen data.

- Data Augmentation

There are many options for data augmentation provided at https://github.com/tensorflow/models/blob/master/research/object_detection/protos/preprocessor.proto

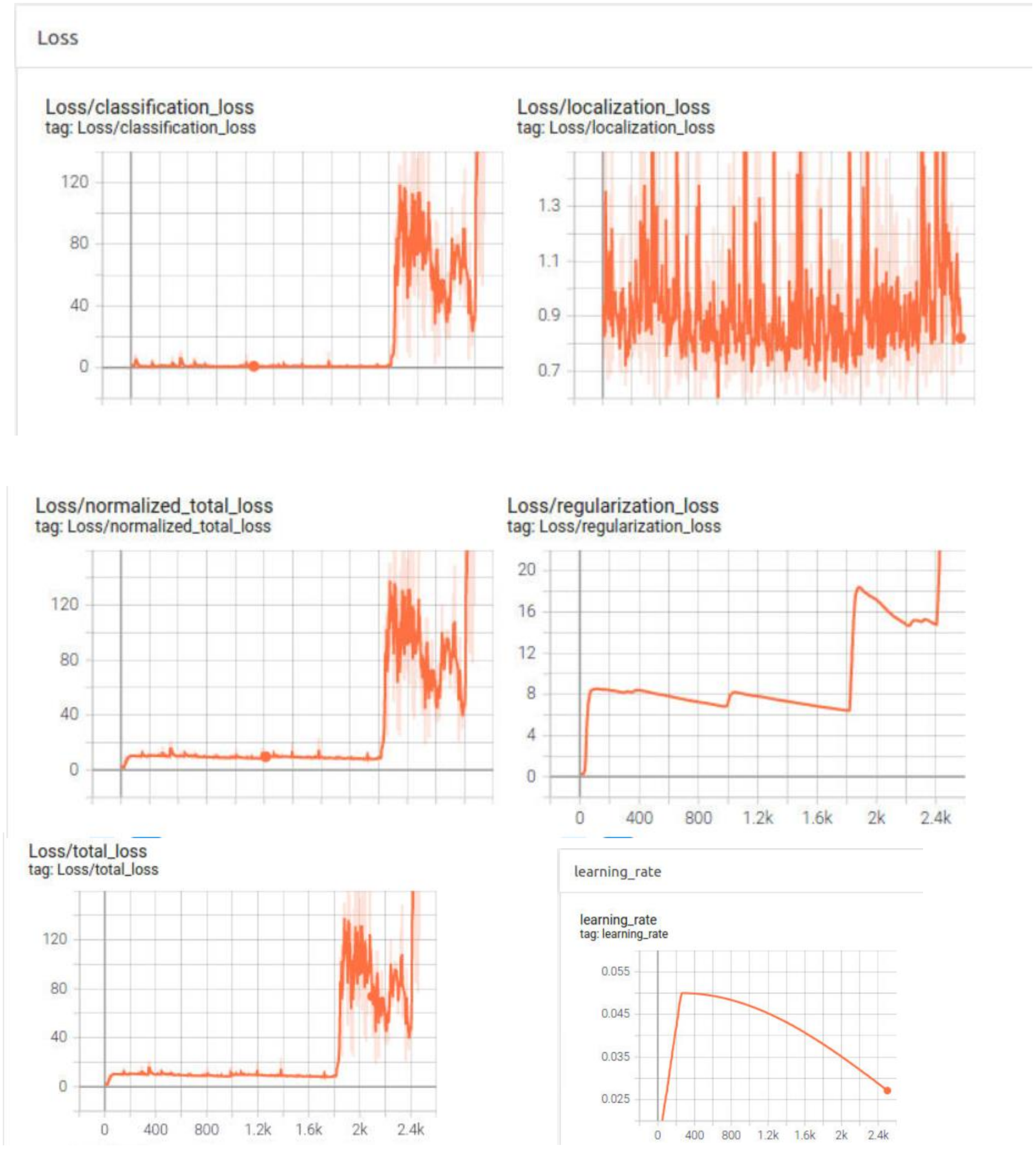
a) Experiment 1

In the first data augmentation experiments, we will work on some techniques to augment the data like:

1) random adjust saturation.

2) Randomly convert the RGB image to the gray image

Here is the Tensor board for this Experiment.



From the previous visualization, we could conclude that the model Starts to overfit the training data, and this will lead to little ability to generalize well,

The training loss started low at (10.105) then it fluctuates till it reaches (391.011) at the final step

The training loss and the evaluation loss were very high, the training Loss was (391.011) and the evaluation loss is (415.855347).

After augmenting the data, the losses increase after decreasing gradually at the beginning, which means that the model overfits the data. A lot of other options exist to use like changing the used model, but it requires changing the config file and there are a lot of other augmentation techniques as mentioned on the website above.