

# **Object Detection in an Urban Environment**

## **1. Project overview**

The main objective is to implement detection of object in an urban environment. The dataset used here is the Waymo Open Dataset. Tensor board is used for displaying the training and evaluation results. Parameters tuning has also been done. The main steps along with implementation details are mentioned below.

## **2. Setup**

### **Step 1 - Exploratory Data Analysis (EDA)**

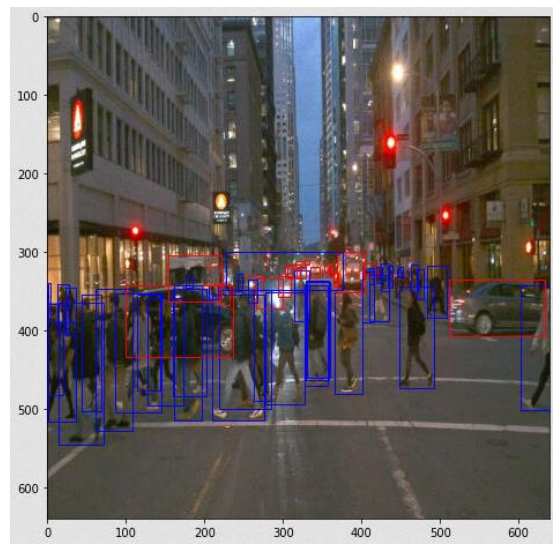
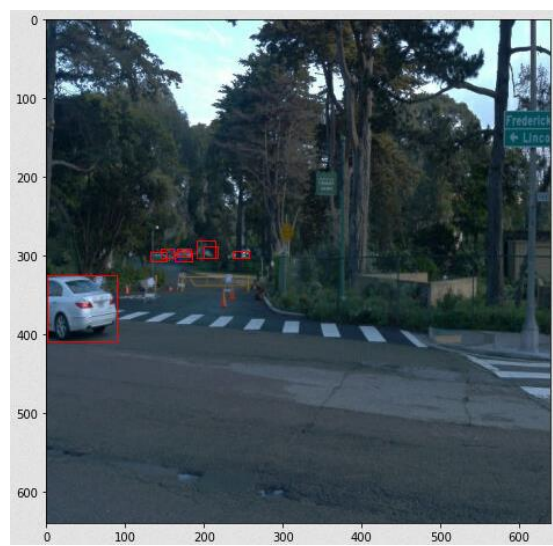
File: Explore data analysis.ipynb

Several steps are performed in EDA on the processes Waymo dataset.

Major steps include

- a. Importing libraries for plotting
- b. Writing the directory of the source dataset
- c. Writing a function to display image and the bounding boxes. For reference, following two images are mentioned here. (Fig 1 and Fig 2)  
(Red bounding box: Vehicles; Blue bounding box: Pedestrian; Green bounding box: Green)
- d. Addition EDA is performed to show Object count with respect to Class ID (Fig. 3)  
(Class 1: Vehicles; Class 2: Pedestrian; Class 4: Bicyclist)  
It is observed that the dataset is skewed, i.e. there is an imbalance of classes.

Dataset analysis: The data is divided in three folders namely Train, Val and test. The data is present in TF record format. Different types of images with objects, different weather conditions, dark, bright, different sizes are present.





*Fig. 1 Images to show the data set (environment, weather and classes distribution)*

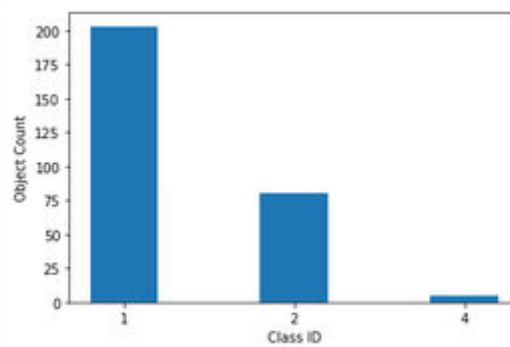


Fig. 2

## Step 2 - Edit the config file

File: new\_pipeline.config

Before the training, we need to change the config file to change the location of the training and validation files, as well as the location of the label\_map file and pretrained weights. A new config file called pipeline\_new.config is created and following modifications are done

- Addition of data augmentation options (brightness, contrast, hue, saturation and color)
- Changing learning rate base (Old: 0.04, New: 0.0005)
- Changing Total steps (Old: 2500, New: 4500)
- Changing warm up learning rate (Old: 0.01, New: 0.0005)
- Changing warm up steps (Old: 200, New: 300)

## Step 3 - Model Training and Evaluation

The training process was done and Tensor board was used to visualize the training and evaluation graphs. The training and evaluation graphs are mentioned below

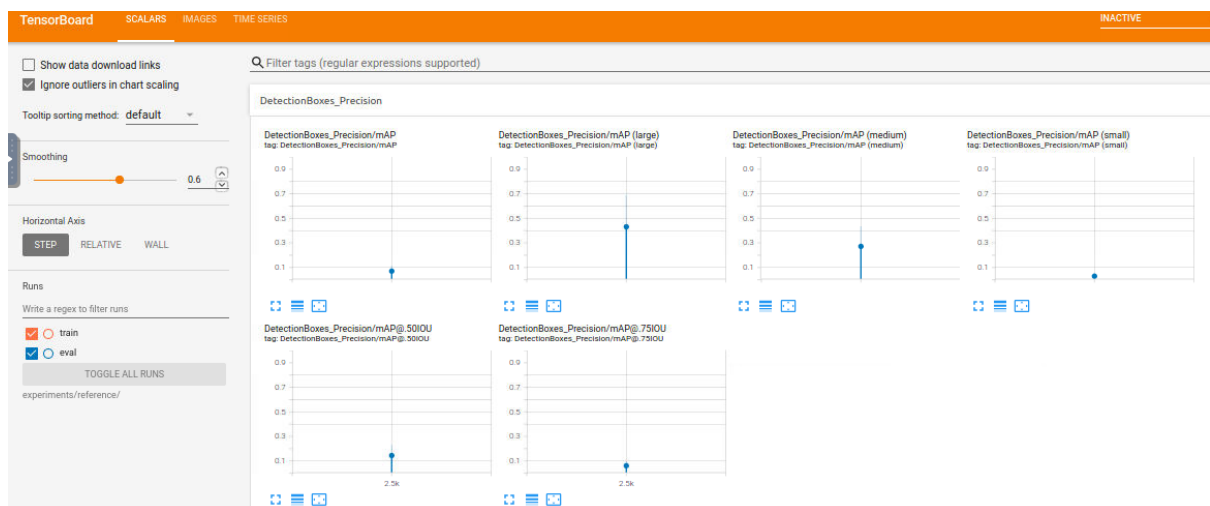


Fig. 4

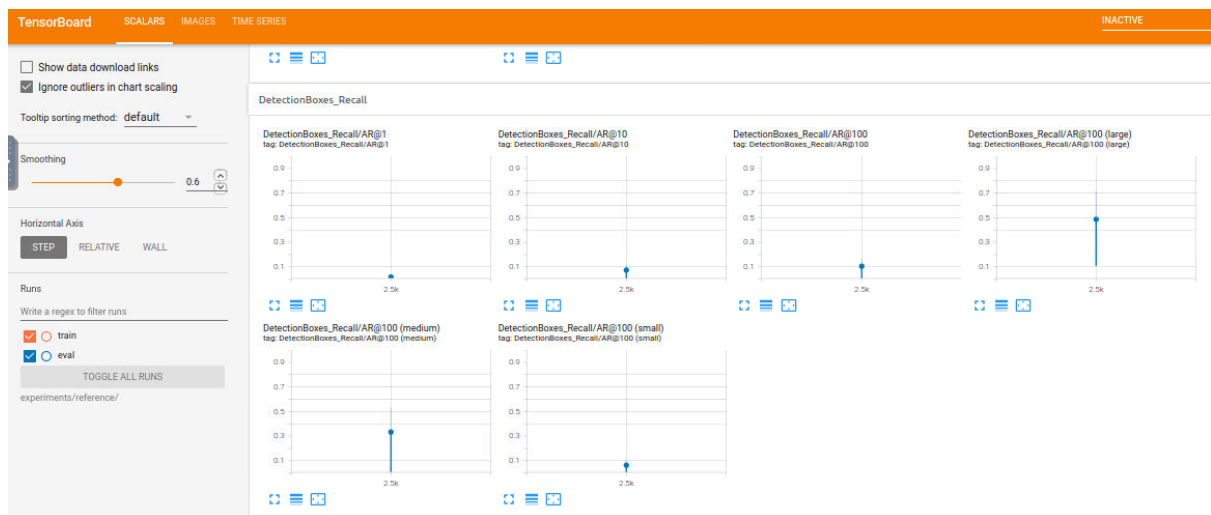


Fig. 5

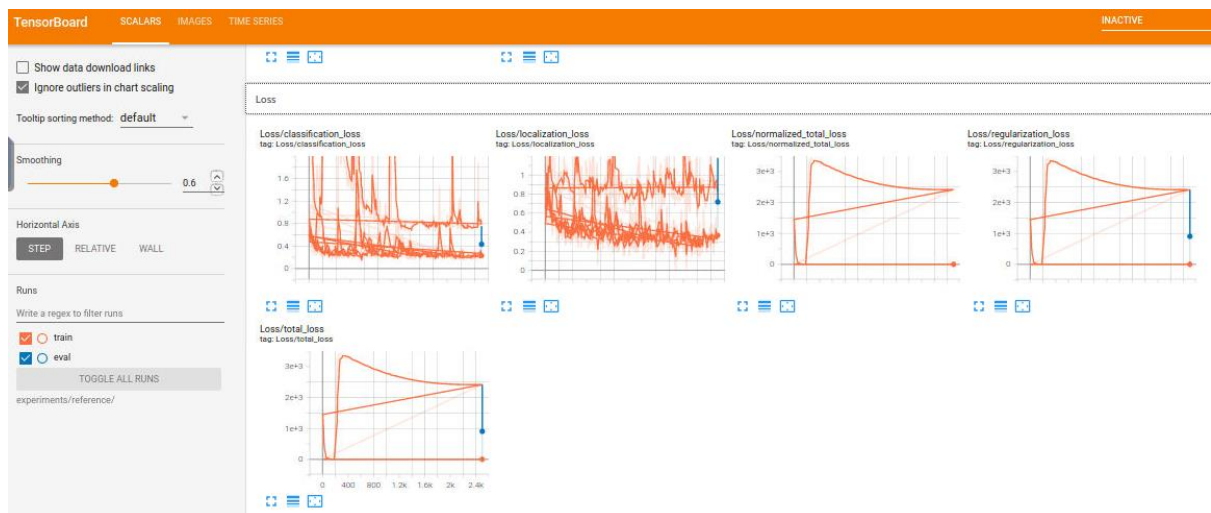
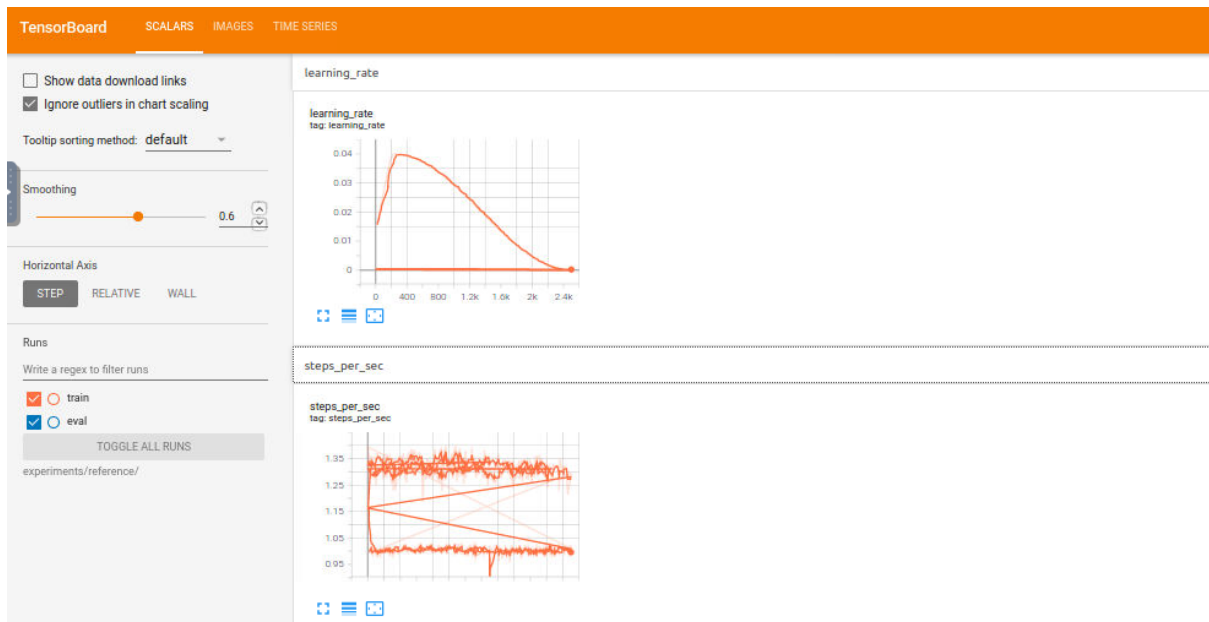


Fig. 6





```

root@a440aeb19bd8: /home/workspace 149x32
INFO:tensorflow:Step 1400 per-step time 0.991s loss=0.705
[0311 16:36:14.762388 140512734979904 model_lib_v2.py:682] Step 1400 per-
step time 0.991s loss=0.705
INFO:tensorflow:Step 1500 per-step time 1.006s loss=0.816
[0311 16:37:54.603789 140512734979904 model_lib_v2.py:682] Step 1500 per-
step time 1.006s loss=0.816
INFO:tensorflow:Step 1600 per-step time 0.996s loss=0.799
[0311 16:39:35.479957 140512734979904 model_lib_v2.py:682] Step 1600 per-
step time 0.996s loss=0.799
INFO:tensorflow:Step 1700 per-step time 1.013s loss=0.760
[0311 16:41:15.535774 140512734979904 model_lib_v2.py:682] Step 1700 per-
step time 1.013s loss=0.760
INFO:tensorflow:Step 1800 per-step time 1.004s loss=0.793
[0311 16:42:55.361418 140512734979904 model_lib_v2.py:682] Step 1800 per-
step time 1.004s loss=0.793
INFO:tensorflow:Step 1900 per-step time 1.005s loss=0.592
[0311 16:44:35.352305 140512734979904 model_lib_v2.py:682] Step 1900 per-
step time 1.005s loss=0.592
INFO:tensorflow:Step 2000 per-step time 1.004s loss=0.849
[0311 16:46:14.891058 140512734979904 model_lib_v2.py:682] Step 2000 per-step time 1.004s loss=0.849
INFO:tensorflow:Step 2100 per-step time 1.012s loss=0.831
[0311 16:47:55.642828 140512734979904 model_lib_v2.py:682] Step 2100 per-step time 1.012s loss=0.831
INFO:tensorflow:Step 2200 per-step time 0.989s loss=0.731
[0311 16:49:35.743353 140512734979904 model_lib_v2.py:682] Step 2200 per-step time 0.989s loss=0.731
INFO:tensorflow:Step 2300 per-step time 1.007s loss=0.703
[0311 16:51:15.419174 140512734979904 model_lib_v2.py:682] Step 2300 per-step time 1.007s loss=0.703
INFO:tensorflow:Step 2400 per-step time 0.997s loss=0.764
[0311 16:52:55.053409 140512734979904 model_lib_v2.py:682] Step 2400 per-step time 0.997s loss=0.764
INFO:tensorflow:Step 2500 per-step time 1.020s loss=0.840
[0311 16:54:34.851697 140512734979904 model_lib_v2.py:682] Step 2500 per-step time 1.020s loss=0.840

```

Fig. 7 Training

```

root@a440aeb19bd8: /home/workspace 149x32
I0311 17:03:04.882646 140559725688640 model_lib v2.py:991] + DetectionBoxes_Precision/mAP@.50IOU: 0.226475
INFO:tensorflow: + DetectionBoxes_Precision/mAP@.75IOU: 0.088630
I0311 17:03:04.884148 140559725688640 model_lib v2.py:991] + DetectionBoxes_Precision/mAP@.75IOU: 0.088630
INFO:tensorflow: + DetectionBoxes_Precision/mAP (small): 0.040616
I0311 17:03:04.885674 140559725688640 model_lib v2.py:991] + DetectionBoxes_Precision/mAP (small): 0.040616
INFO:tensorflow: + DetectionBoxes_Precision/mAP (medium): 0.401533
I0311 17:03:04.887250 140559725688640 model_lib v2.py:991] + DetectionBoxes_Precision/mAP (medium): 0.401533
INFO:tensorflow: + DetectionBoxes_Precision/mAP (large): 0.516608
I0311 17:03:04.888913 140559725688640 model_lib v2.py:991] + DetectionBoxes_Precision/mAP (large): 0.516608
INFO:tensorflow: + DetectionBoxes_Recall/AR@1: 0.025433
I0311 17:03:04.890609 140559725688640 model_lib v2.py:991] + DetectionBoxes_Recall/AR@1: 0.025433
INFO:tensorflow: + DetectionBoxes_Recall/AR@10: 0.106718
I0311 17:03:04.892165 140559725688640 model_lib v2.py:991] + DetectionBoxes_Recall/AR@10: 0.106718
INFO:tensorflow: + DetectionBoxes_Recall/AR@100: 0.153480
I0311 17:03:04.893676 140559725688640 model_lib v2.py:991] + DetectionBoxes_Recall/AR@100: 0.153480
INFO:tensorflow: + DetectionBoxes_Recall/AR@100 (small): 0.096377
I0311 17:03:04.895183 140559725688640 model_lib v2.py:991] + DetectionBoxes_Recall/AR@100 (small): 0.096377
INFO:tensorflow: + DetectionBoxes_Recall/AR@100 (medium): 0.471537
I0311 17:03:04.896646 140559725688640 model_lib v2.py:991] + DetectionBoxes_Recall/AR@100 (medium): 0.471537
INFO:tensorflow: + DetectionBoxes_Recall/AR@100 (large): 0.591200
I0311 17:03:04.898236 140559725688640 model_lib v2.py:991] + DetectionBoxes_Recall/AR@100 (large): 0.591200
INFO:tensorflow: + Loss/localization_loss: 0.429932
I0311 17:03:04.899660 140559725688640 model_lib v2.py:991] + Loss/localization_loss: 0.429932
INFO:tensorflow: + Loss/classification_loss: 0.241567
I0311 17:03:04.901180 140559725688640 model_lib v2.py:991] + Loss/classification_loss: 0.241567
INFO:tensorflow: + Loss/regularization_loss: 0.244137
I0311 17:03:04.902604 140559725688640 model_lib v2.py:991] + Loss/regularization_loss: 0.244137
INFO:tensorflow: + Loss/total_loss: 0.915635
I0311 17:03:04.904036 140559725688640 model_lib v2.py:991] + Loss/total_loss: 0.915635

```

Fig. 8 Evaluation

#### Step 4 - Improve the performances

File: Augmentations.ipynb

- To improve the performances data augmentation is implemented as mentioned in Step 2
- Also, parameters in Explore augmentations.ipynb has been modified.
- Following augmentation techniques have been used
  - a. Random crop image
  - b. Random adjust contrast
  - c. Random adjust saturation
  - d. Random adjust hue
  - e. Random adjust brightness
  - f. Random distort color

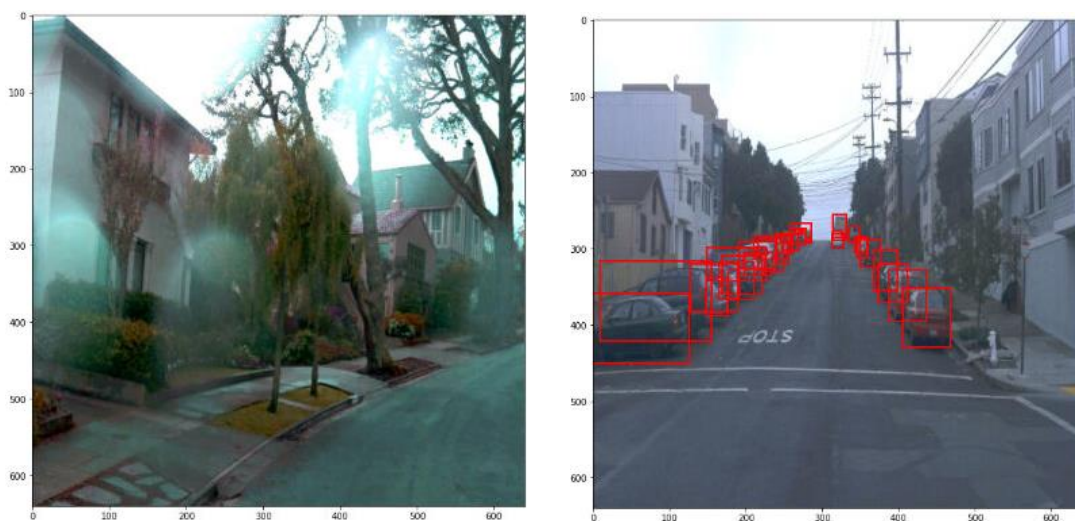


Fig. 9 After augmentation images