Project Overview:

- The main objective of this project was to detect objects namely cars, pedestrians, and cyclists from a video source.
- As video is a combination of multiple images in a sequential manner we mainly focused on object detection on images.
- When we feed to video to the model to predict the objects we process individual frames and consider them as separate images

Set-Up:

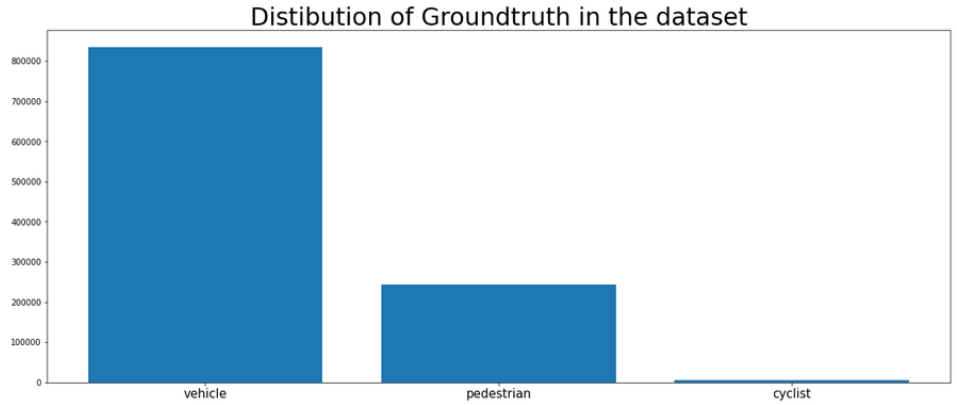
- To set up the project, we need to install some python libraries which will help us to load the data and perform various tasks including reading the data, creating Neural Network and other tasks used in training a model.
- The main library used for this project is the Tensorflow object detection library.
    - This library is built upon TensorFlow which is a library created by Google for Artificial Neural networks-related tasks.
    - To run this library we need the backend of Tensorflow too.
- The other libraries used in this project are mentioned in the requirements.txt file found in the main project directory.
- Also to run this project we will need a Graphical Processing unit as Neural Networks have proved to train faster on GPUs rather than CPUs.

Dataset:

- The Dataset used in this project is the official Waymo open dataset which is provided by Waymo - a self-driving car company based out of Palo Alto, California.
- Waymo open dataset is found on the google cloud services in storage.
    - There are about 700+ tf.record datasets in the storage from which this project has used 75.
    - Tf.record is a type of dataset which is made for TensorFlow model feeding as this makes the process faster and avoids CPU bottlenecking.
    - Tf.record dataset is basically a batch of datasets that is fed one at a time to the Tensorflow model as the model will be trained on the individual batches at any given point of time.

- Data Analysis:
    - The dataset has mainly 2 things,
        - Image
        - x, y coordinates of the bounding box

- From the further exploratory data analysis, it is found that the total number of vehicles, pedestrians, and Cyclists in the 75 tf.record instances we used are 835336, 244131, and 6054 respectively.
- The distribution in graphical form is as …



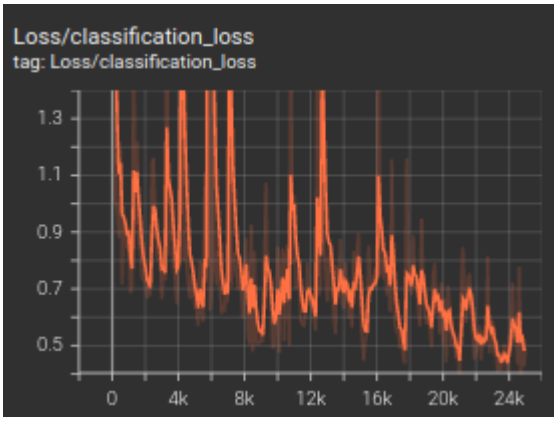
- Data Augmentation:
  - There are various data augmentations we've used to make the model more generalized.
  - The augmentations are as follows:
    - Random Horizontal Flip: This function flips the randomly selected frame horizontally.
    - Random Adjust Brightness: This function as the name suggests will change the brightness level in the randomly selected image frame
    - Random RGB to Gray: This function as the name suggests will convert the rgb style image to gray scaled image in the randomly selected image frame.
    - Random Adjust Contrast: This function as the name suggests will change the Contrast level in the randomly selected image frame.
    - Random Jitter Boxes: This function is used to add some noise in the bounding boxes in the randomly selected image frame.
  - Along with the above-mentioned augmentations we also use random_crop_image so as to crop some part of the image as not all the frames in real life are fully captured by the camera.

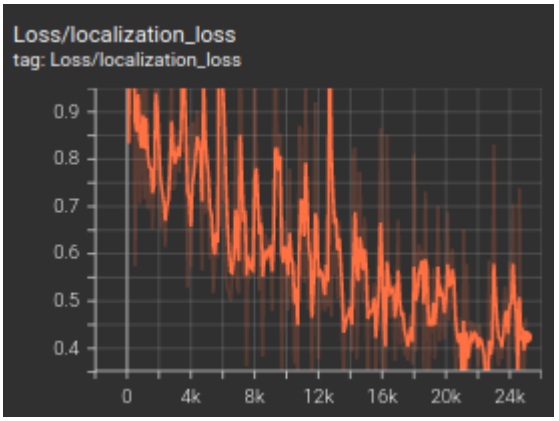
- Cross-Validation:
  - As we train the model on training data we need some validation data too to check how the model performs on unseen data.
  - For this we have split the data into 2 parts, one is training data and another is validation data.
    - In training data, we have 60 tf.records

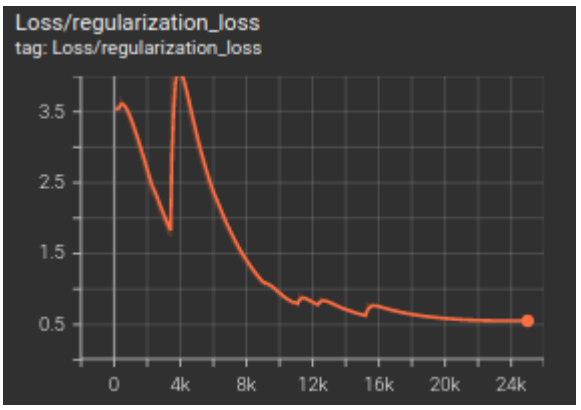
        - In validation data, we have 7 tf.records
    - When training the model we feed the validation data as validation_dataset so that the model can evaluate its performance on the validation data at every iteration.

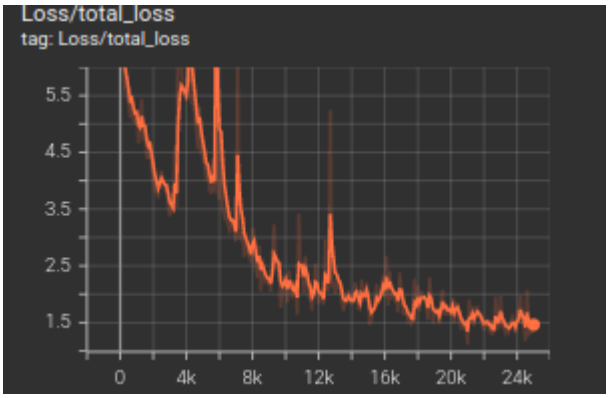
Training:

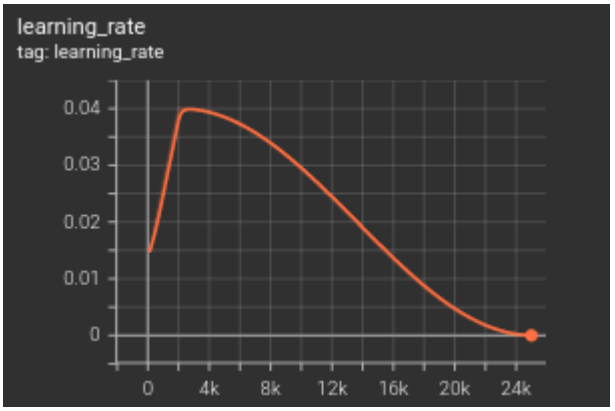
Note: The Image Size used in all the experiments is 640x640

- ## Reference Experiments:
    - For the First Reference experiment, the configurations were as follows…
        - Model architecture - ssd_resnet50_v1_fpn_keras
        - Data augmentation - random_crop_image, and random_horizontal_flip
        - Batch Size - 2
        - The Loss of the same experiment is as follows …
        - 

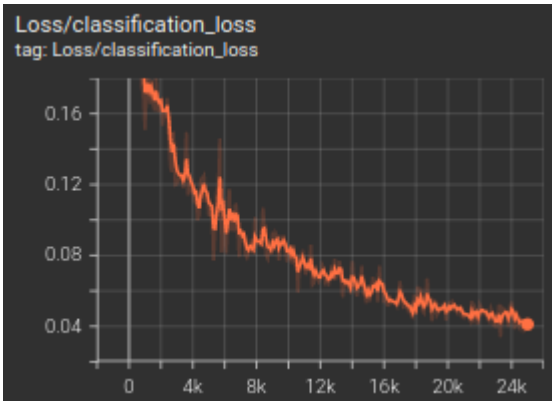
        - 

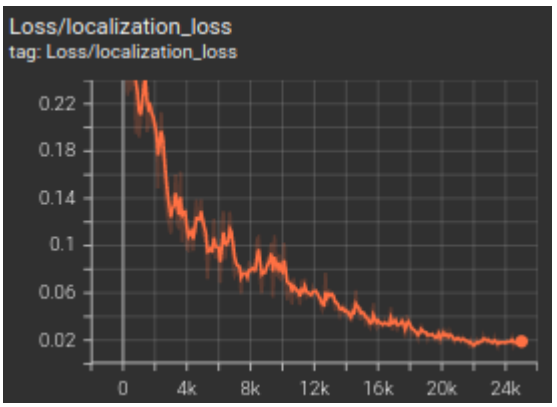


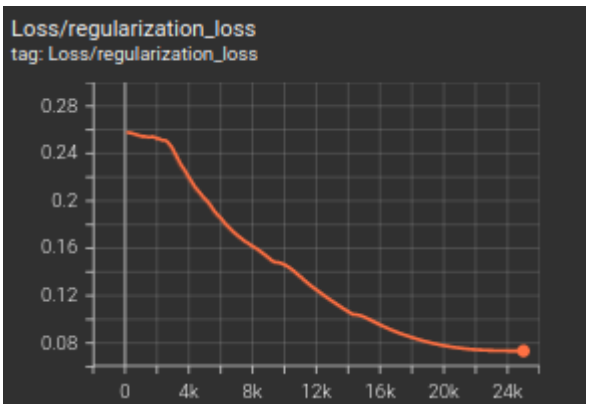


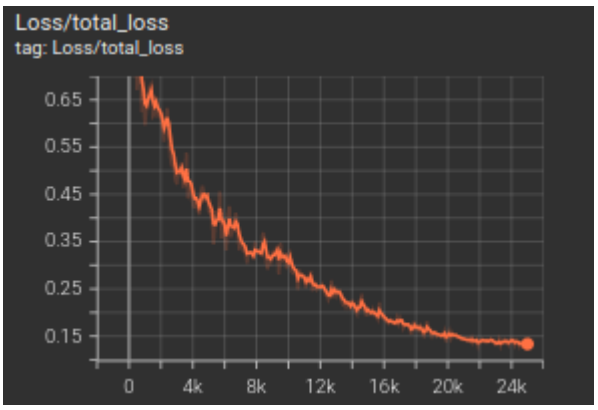


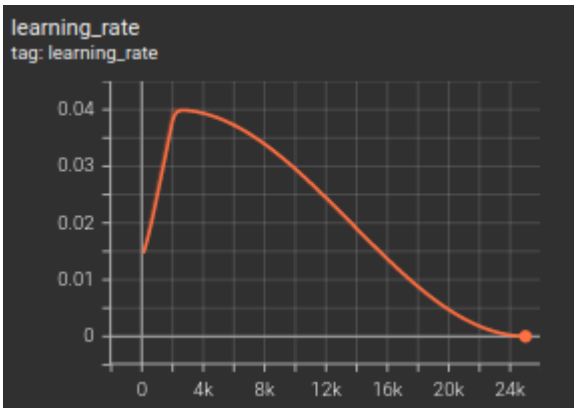
- The first graph is of Classification Loss
- The second graph is of Localisation Loss
- The Third Graph represents the Regularisation Loss
- The fourth graph displays the total loss of the entire experiment with respect to each epoch
- The fifth and the final graph reflects how the learning rate reduces with every passing epoch
- To check the configuration of this file you can refer to the new_pipeline.config file found in experiments/experiments0/references/
- The total number of steps for this experiment was 25000.

- Experiment 2:
  - For the Second experiment, the configurations were as follows…
    - Model architecture - ssd_resnet50_v1_fpn_keras
    - Data augmentation - random_crop_image, random_horizontal_flip, random_adjust_brightness, random_rgb_to_gray, random_adjust_contrast, random_jitter_boxes
    - Learning Rate was initially set to 0.04
    - Batch Size - 8
    - The Loss of the same experiment is as follows …
    - 

    - 

    - 

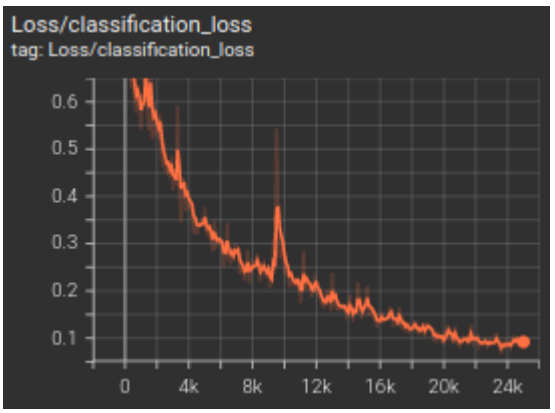
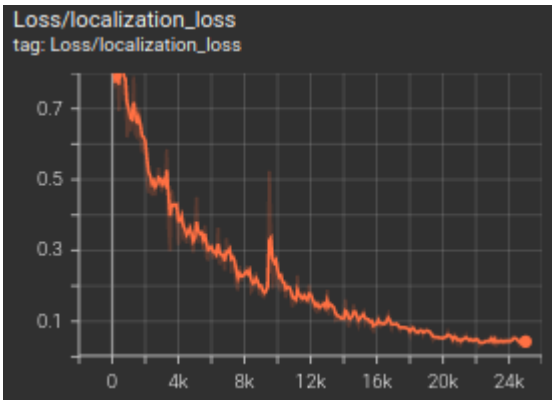
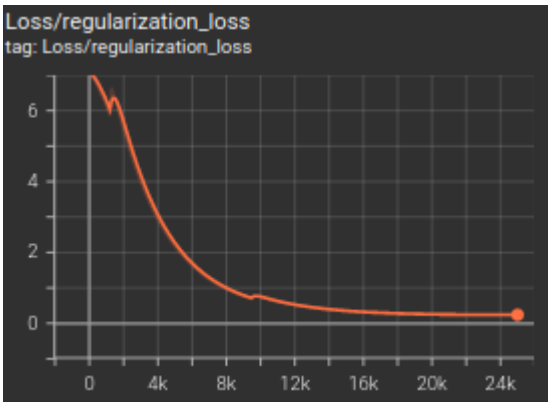


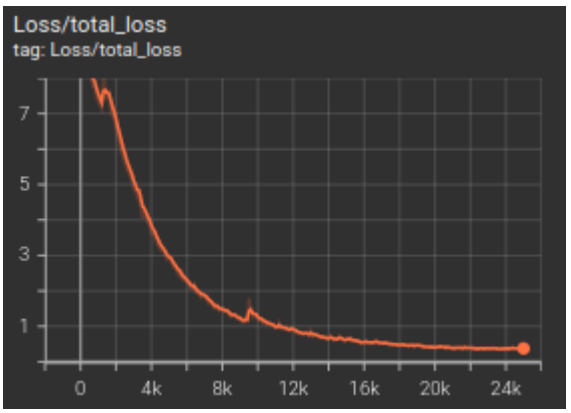


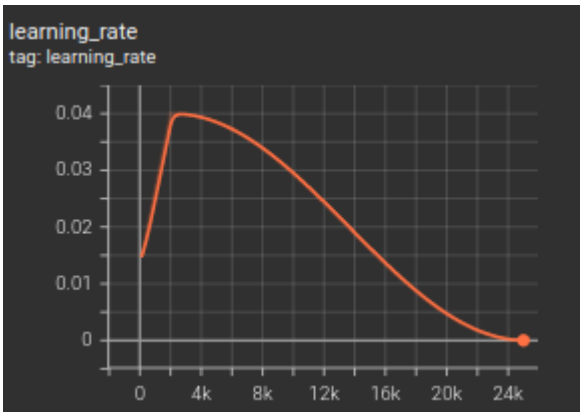
- The first graph is of Classification Loss
- The second graph is of Localisation Loss
- The Third Graph represents the Regularisation Loss
- The fourth graph displays the total loss of the entire experiment with respect to each epoch
- The fifth and the final graph reflects how the learning rate reduces with every passing epoch
- To check the configuration of this file you can refer to the new_pipeline.config file found in experiments/experiments1/references/
- The total number of steps for this experiment was 25000.

○ Experiment 3:

- For the Third experiment, the configurations were as follows…
  - Model architecture - ssd_resnet101_v1_fpn_keras
  - Data augmentation - random_crop_image, random_horizontal_flip, random_adjust_brightness, random_rgb_to_gray, random_adjust_contrast, random_jitter_boxes
  - Learning Rate was initially set to 0.04
  - Batch Size - 8
  - The Loss of the same experiment is as follows …

- 
Loss/classification_loss
tag: Loss/classification_loss
0.6
0.5
0.4
0.3
0.2
0.1
0
4k
8k
12k
16k
20k
24k

Loss/localization_loss
tag: Loss/localization_loss
0.7
0.5
0.3
0.1
0
4k
8k
12k
16k
20k
24k

Loss/regularization_loss
tag: Loss/regularization_loss
6
4
2
0
0
4k
8k
12k
16k
20k
24k



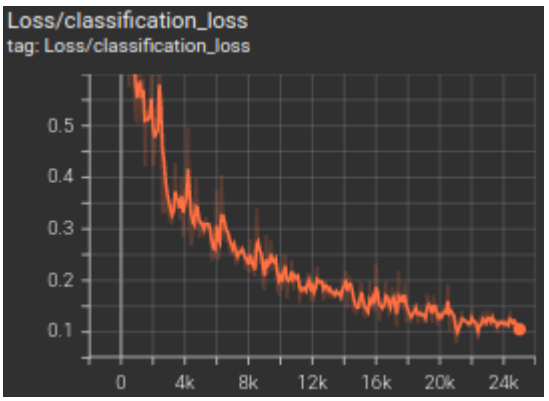
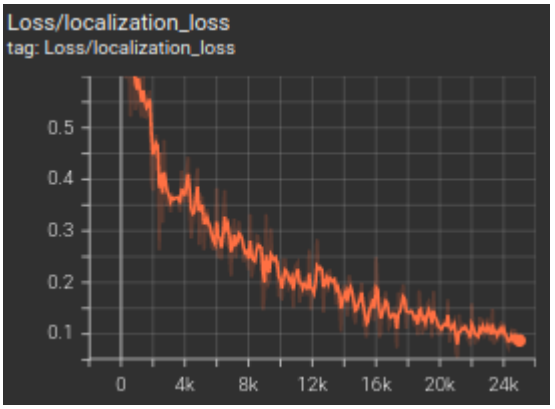
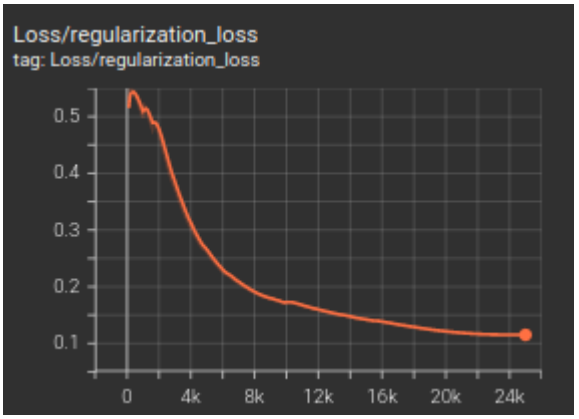


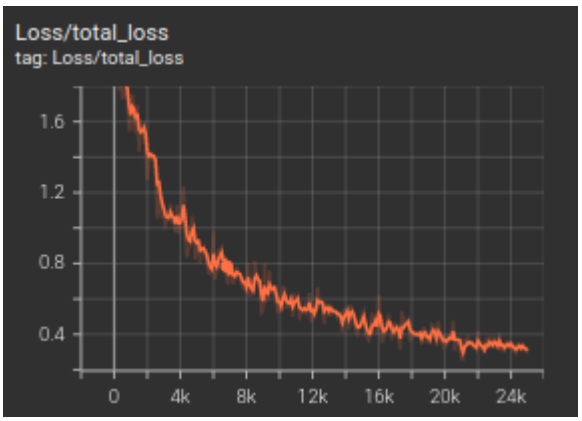
- The first graph is of Classification Loss
- The second graph is of Localisation Loss
- The Third Graph represents the Regularisation Loss
- The fourth graph displays the total loss of the entire experiment with respect to each epoch
- The fifth and the final graph reflects how the learning rate reduces with every passing epoch
- To check the configuration of this file you can refer to the new_pipeline.config file found in experiments/experiments2/references/
- The total number of steps for this experiment was 25000.

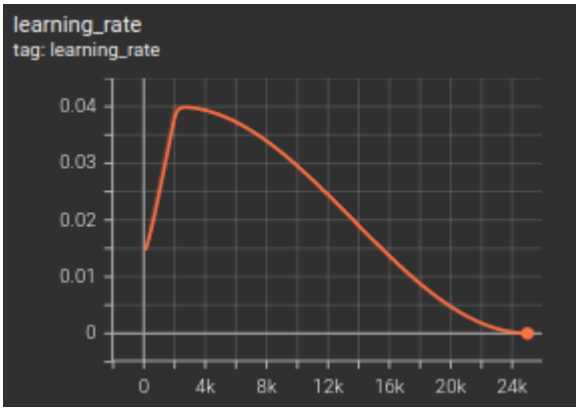
○ Experiment 4:

■ For the Third experiment, the configurations were as follows…

- Model architecture - ssd_resnet152_v1_fpn_keras
- The input size is 1024x1024x3
- Data augmentation - random_crop_image, random_horizontal_flip, random_adjust_brightness, random_rgb_to_gray, random_adjust_contrast, random_jitter_boxes
- Learning Rate was initially set to 0.04
- Batch Size - 4
- The Loss of the same experiment is as follows …

- 
Loss/classification_loss
tag: Loss/classification_loss
0.5
0.4
0.3
0.2
0.1
0
4k
8k
12k
16k
20k
24k

Loss/localization_loss
tag: Loss/localization_loss
0.5
0.4
0.3
0.2
0.1
0
4k
8k
12k
16k
20k
24k

Loss/regularization_loss
tag: Loss/regularization_loss
0.5
0.4
0.3
0.2
0.1
0
4k
8k
12k
16k
20k
24k





- The first graph is of Classification Loss
- The second graph is of Localisation Loss
- The Third Graph represents the Regularisation Loss
- The fourth graph displays the total loss of the entire experiment with respect to each epoch
- The fifth and the final graph reflects how the learning rate reduces with every passing epoch
- To check the configuration of this file you can refer to the new_pipeline.config file found in experiments/experiments3/references/
- The total number of steps for this experiment was 25000.