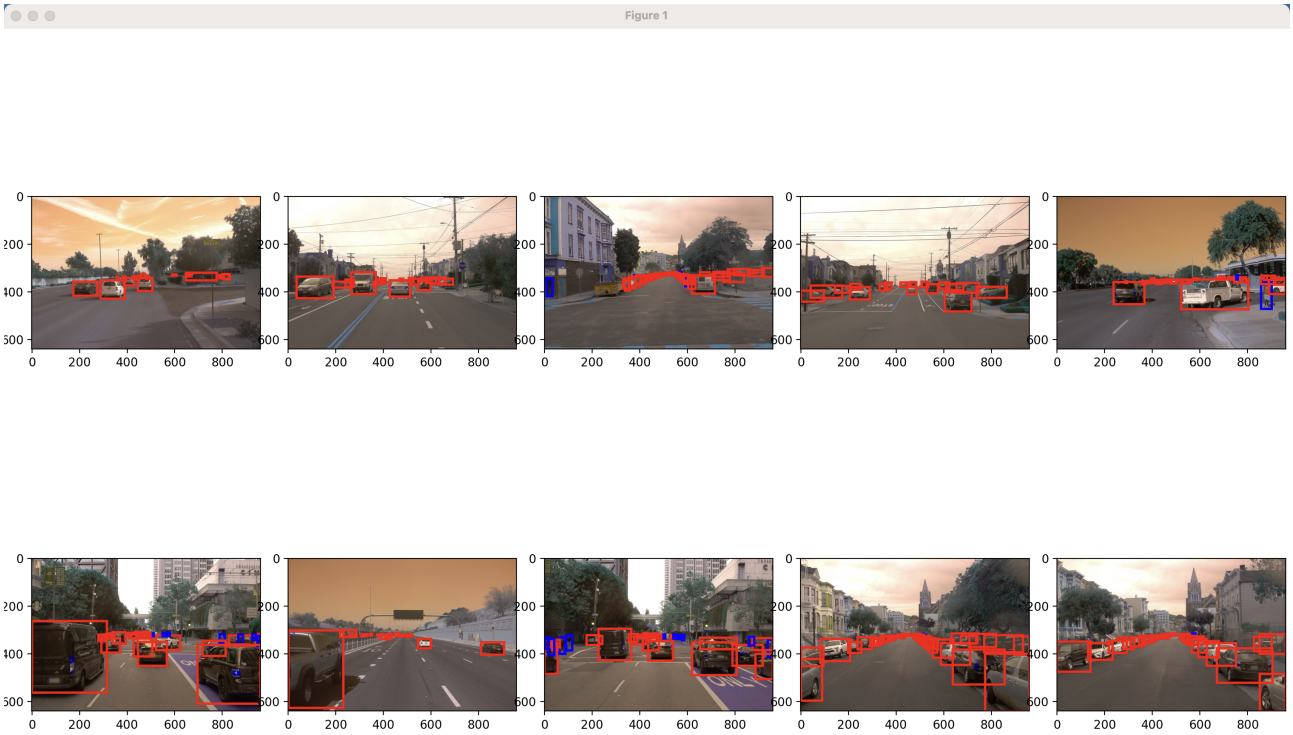
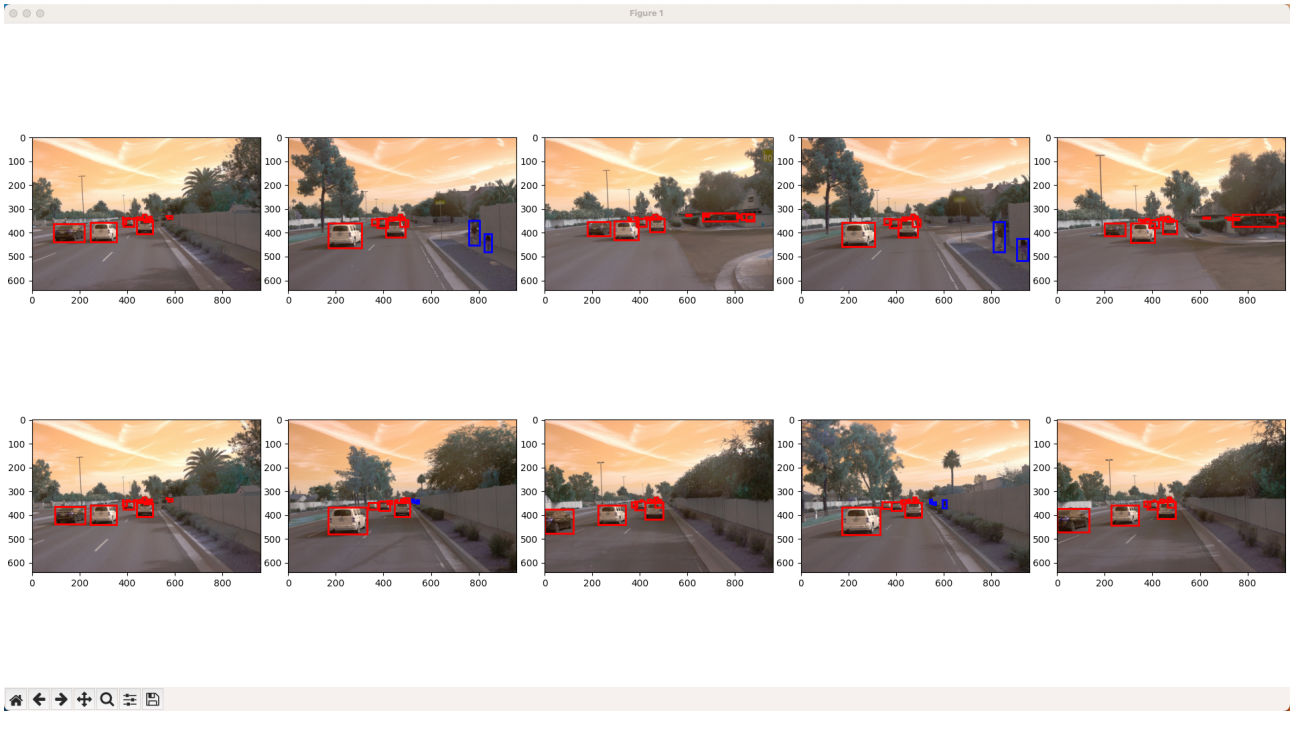
Rquired visualizations

Below is displaying 10 image and label from waymo data set.



Upper Shuffle, below not shuffled



This is my tensorboard capture without augmentation, here I can find that the model changes learning rate, after “total loss” started to decrease, the model decreases learning rate too, to find optimum value preventing being trapped in local minima, Also I can find that loss decreases as training goes on.

It’s strange that when training in Udacity Workspace loss don’t goes up around 2k steps, maybe In local, weights of pertained model are not loaded.

```
I0320 08:44:18.527478 139932546647808 model_lib_v2.py:682] Step 3400 per-step time 0.801s loss=2.332
INFO:tensorflow:Step 3500 per-step time 0.778s loss=1.789
I0320 08:45:37.092653 139932546647808 model_lib_v2.py:682] Step 3500 per-step time 0.778s loss=1.789
INFO:tensorflow:Step 3600 per-step time 0.764s loss=1.822
I0320 08:46:55.621256 139932546647808 model_lib_v2.py:682] Step 3600 per-step time 0.764s loss=1.822
```

Upper image is from Udacity workspace and it shows that model reached loss 1.822 in just 3.6k steps, whereas laptop shows bigger loss(~22)around that step.

Please refer that I couldn’t use tensor board in Udacity workspace because

Firefox, Chrome keep crashing, this is running on my laptop, so I have to capture tensorboard while training is on going on my laptop, it take so much time to complete training in laptop.
Below is capture around 20k steps

Data Augmentation

```
#aug      You, 26 seconds ago • Uncommitted changes
from object_detection.protos import preprocessor_pb2
# Construct a new PreprocessingStep object
my_new_data_augmentation = preprocessor_pb2.PreprocessingStep()
# I would like to randomly change some color images to gray with %20 probability
my_new_data_augmentation.random_rotation90.probability = 0.8
my_new_data_augmentation.random_adjust_hue.max_delta = 0.02
my_new_data_augmentation.random_downscale_to_target_pixels.random_coef = 0.8
my_new_data_augmentation.random_downscale_to_target_pixels.min_target_pixels = 2
my_new_data_augmentation.random_downscale_to_target_pixels.max_target_pixels = 3
print(my_new_data_augmentation)
pipeline_config.train_config.data_augmentation_options.append(my_new_data_augmentation)
```

I configured pipeline.config file to use “random_rotation90”, “random_adjust_hue”, “random_downscale_to_target_pixels” augmentation methods.
reason for choose those options:
rotation90 : it will help In case car is on the roads that has severe declination or inclination.
random_hue : it will help for lens flare and hue deviation of objects under colored transparent matters.
Downscale : it will help to retrofit to inferior camera sensor.
Below is generated config file.

```
train_config {
  batch_size: 2
  data_augmentation_options {
    random_horizontal_flip {
    }
  }
  data_augmentation_options {
    random_crop_image {
      min_object_covered: 0.0
      min_aspect_ratio: 0.75
      max_aspect_ratio: 3.0
      min_area: 0.75
      max_area: 1.0
      overlap_thresh: 0.0
    }
  }
  data_augmentation_options {
    random_downscale_to_target_pixels {
      random_coef: 0.8
      min_target_pixels: 2
      max_target_pixels: 3
    }
  }
```