Traffic Light Recognition

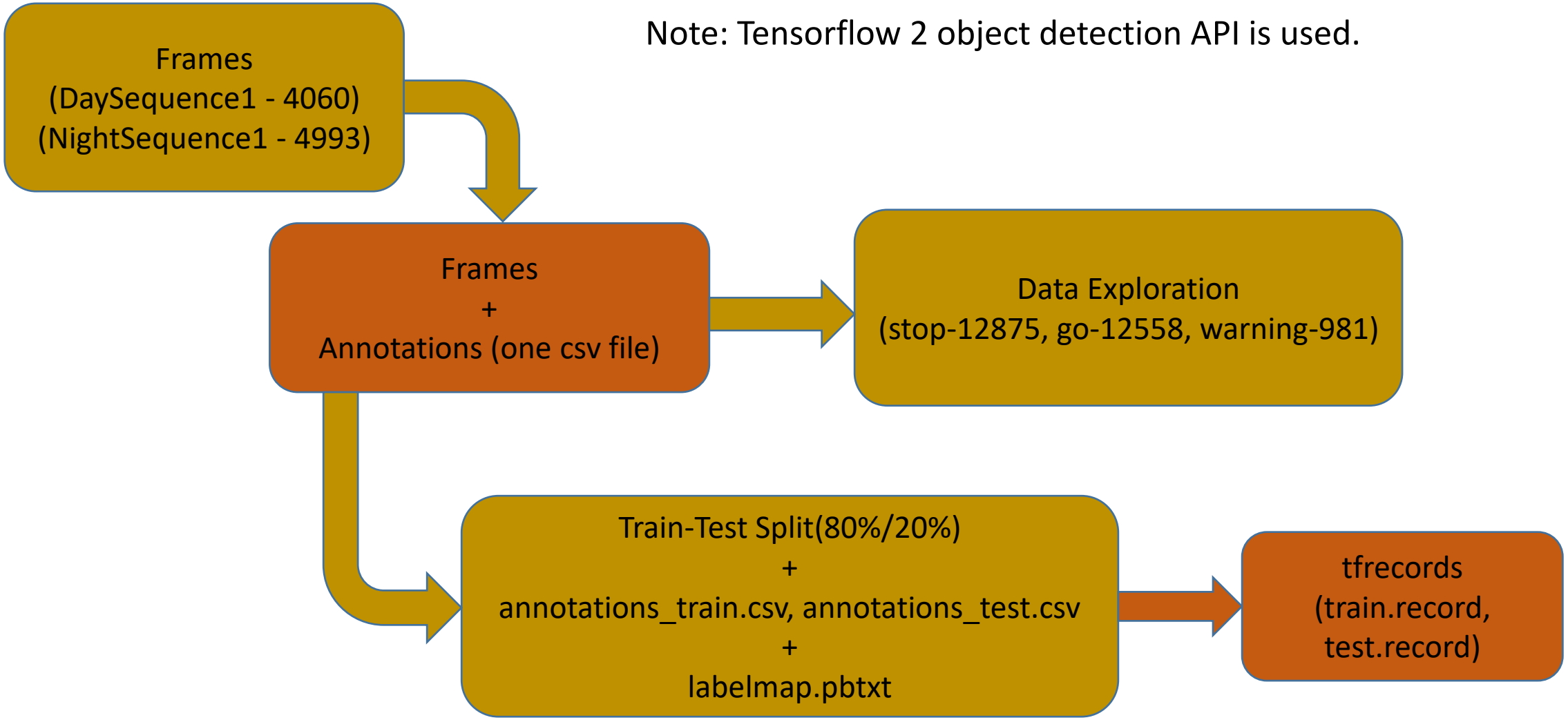
Mahesh Raut

Points Covered

- Section 1: LISA Traffic Light Dataset
- Section 2: Software Setup
- Section 3: Deep Learning for Traffic Light Recognition
- Section 4: Selecting the right model architecture
- Section 5: Training model – Faster RCNN, SSD
- Section 6: Evaluation of Faster RCNN and SSD
- Section 7: Comparison of models

LISA Traffic Light Dataset

Note: Tensorflow 2 object detection API is used.



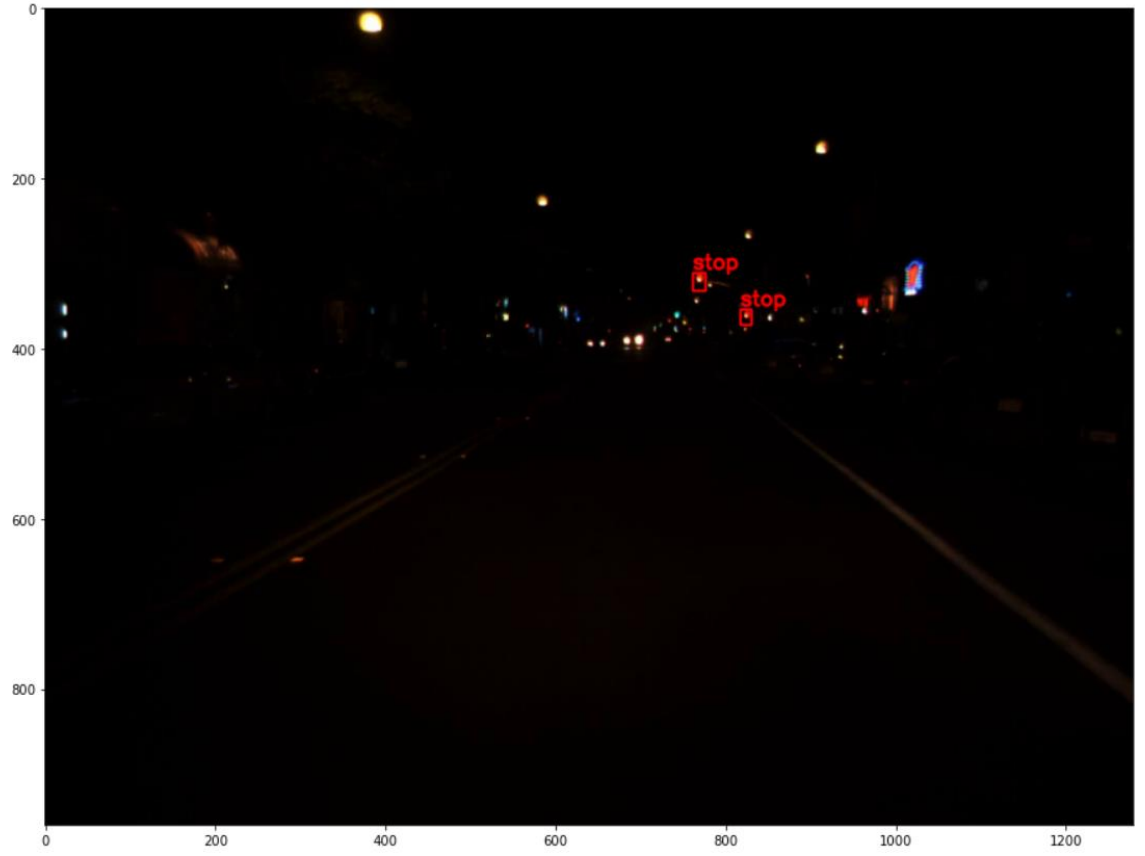
Dataset Source: https://www.kaggle.com/datasets/mbornoe/lisa-traffic-light-dataset

LISA Traffic Light Dataset – Data Exploration

Day Sequence

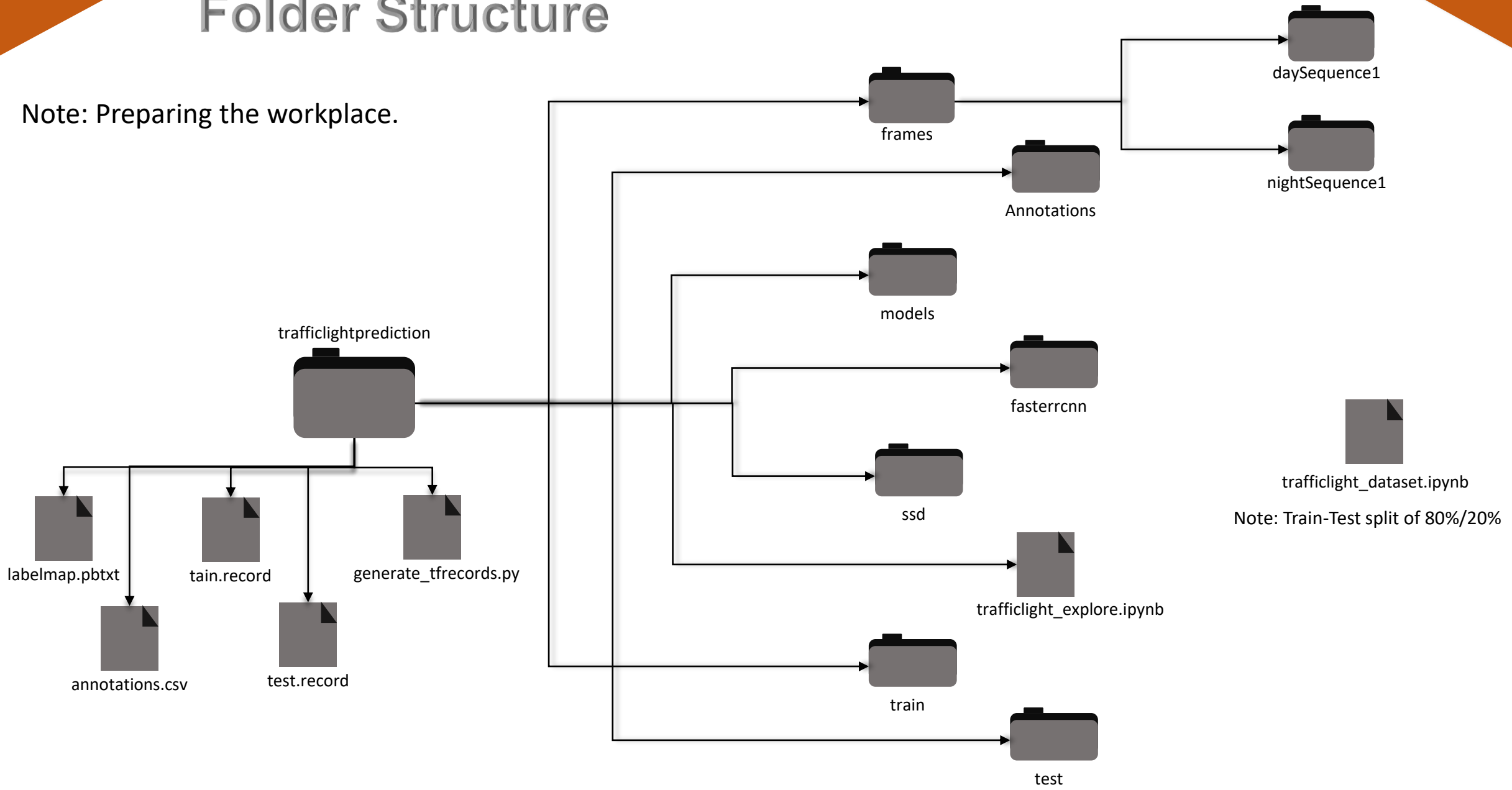
LISA Traffic Light Dataset – Data Exploration

Night Sequence

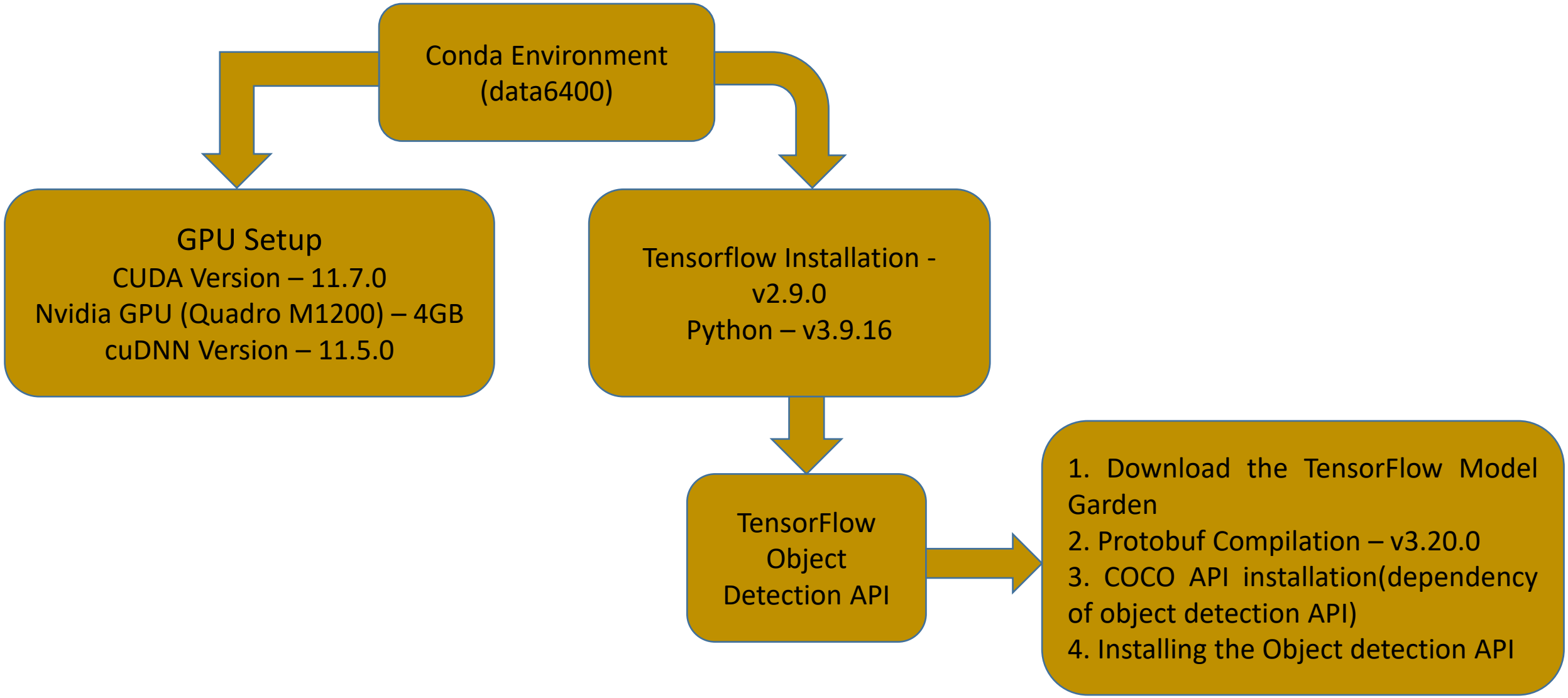


Folder Structure

Note: Preparing the workplace.



Software Setup



Why Deep Learning for Traffic Light Recognition?

Deep Learning directly tackles these issues by:

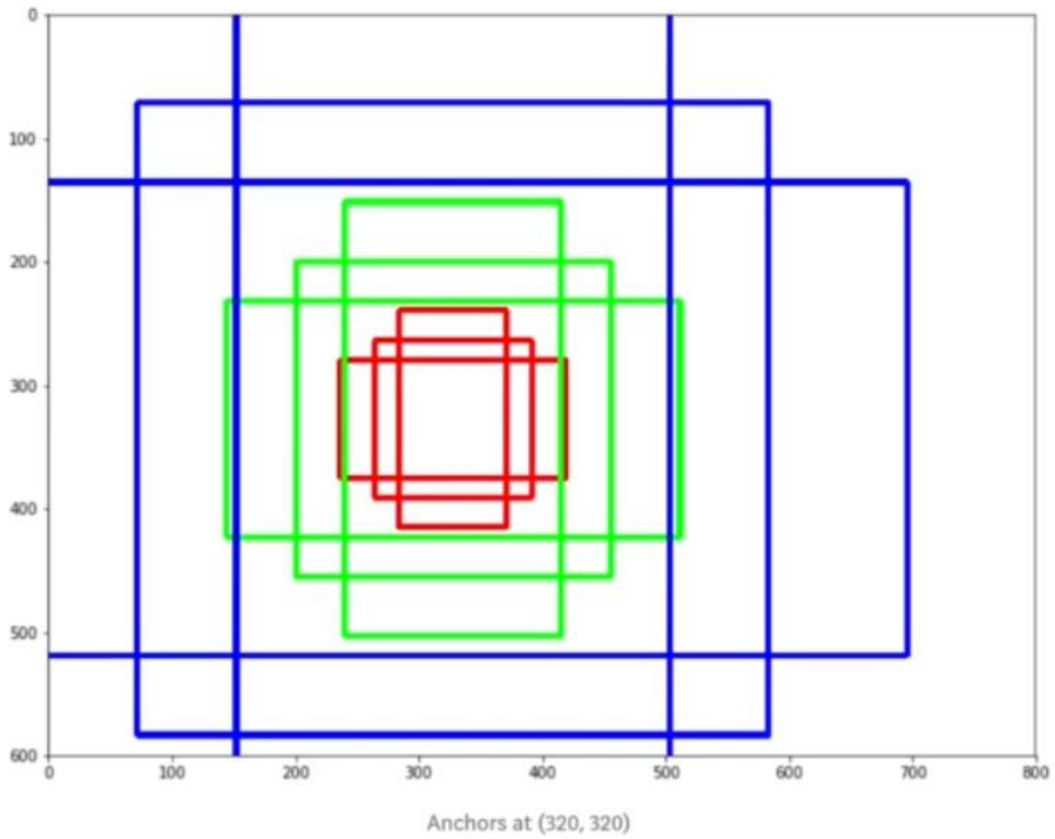
1. “Learning” very distinguishing features which improves accuracy.
2. Sharing computation when computing these features, which improves speed.
3. Learning enough features distributions to be able to generalize well on new unseen images.

Categories of Neural Networks Used for Traffic Light Recognition

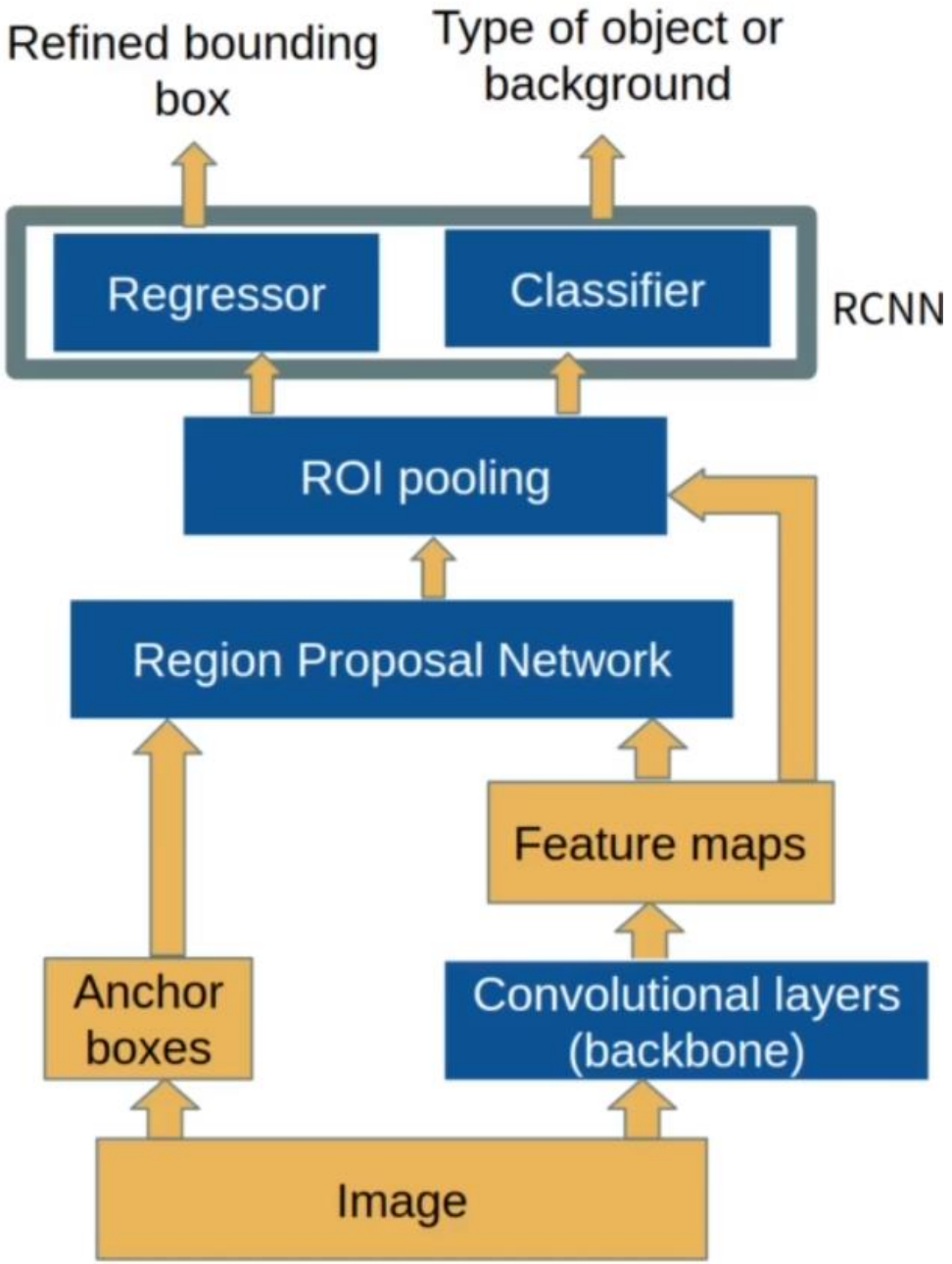
There are two types of Deep Neural Networks used for Traffic Light Recognition:

1. “One stage” object detectors (SSD).
2. “Two stages” object detectors (Faster RCNN).

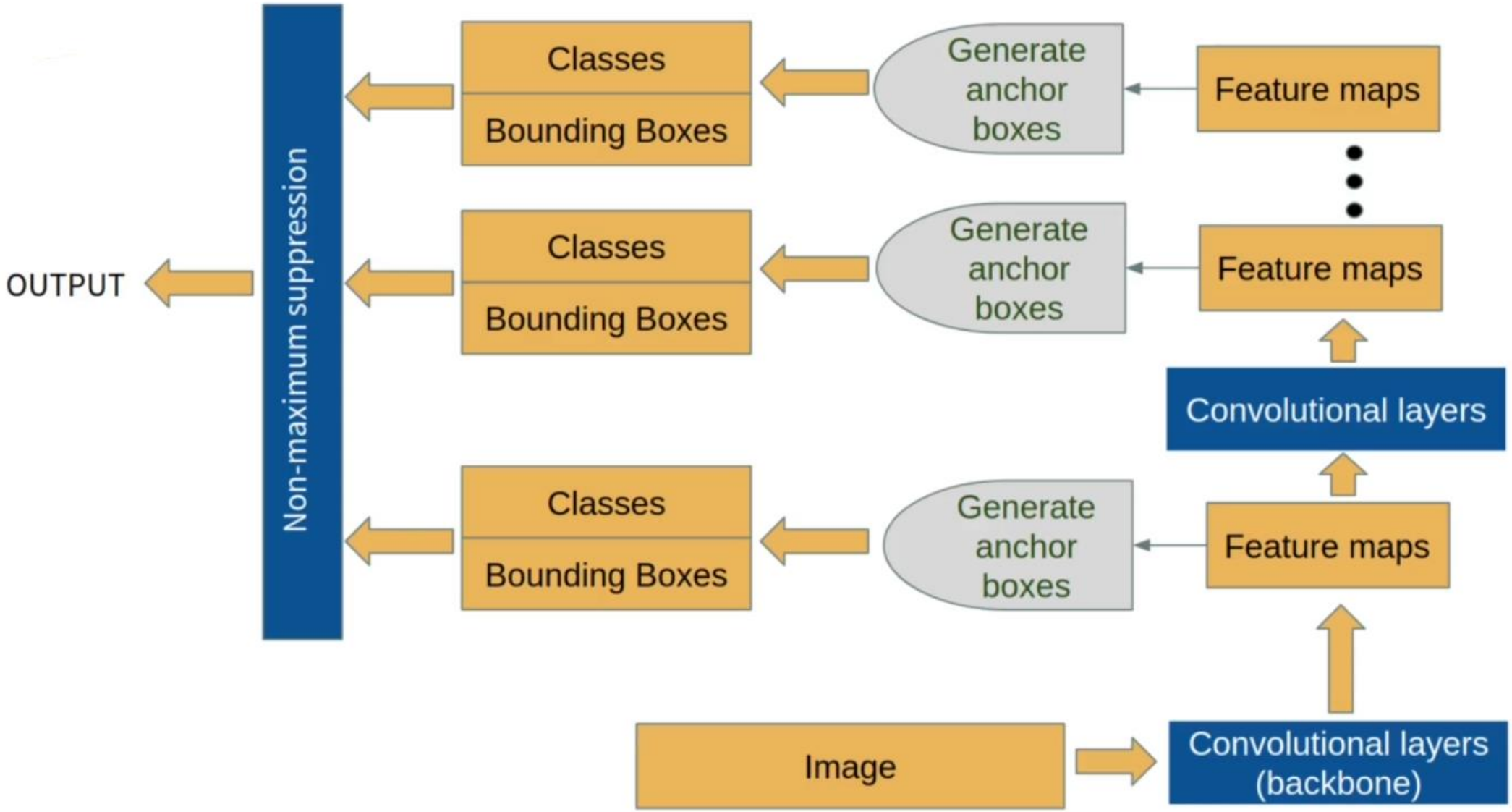
Faster RCNN



Anchors at (320, 320)

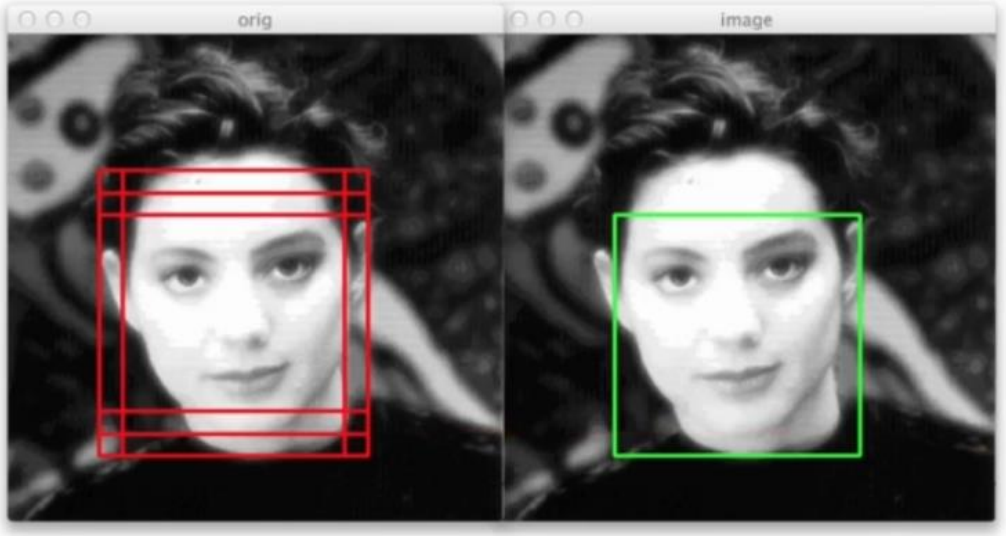


SSD (Single Shot multibox Detector)



SSD (Single Shot multibox Detector)

Non-maximum Suppression:



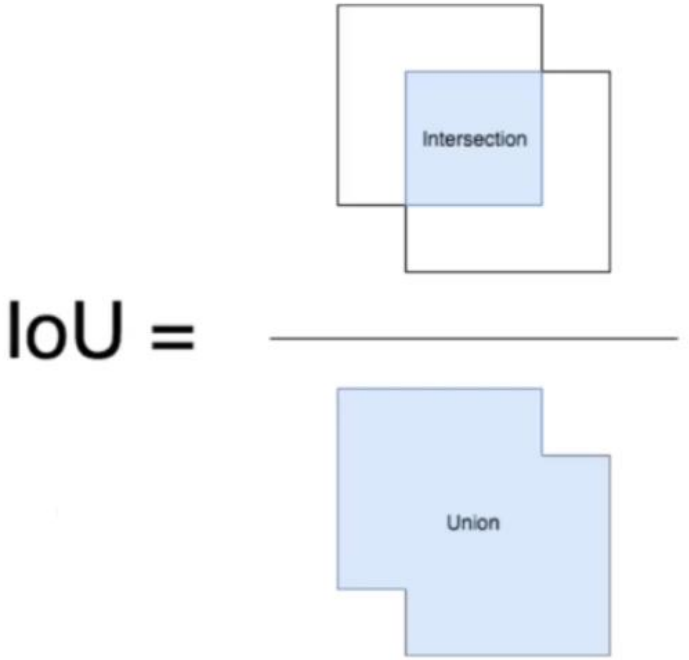
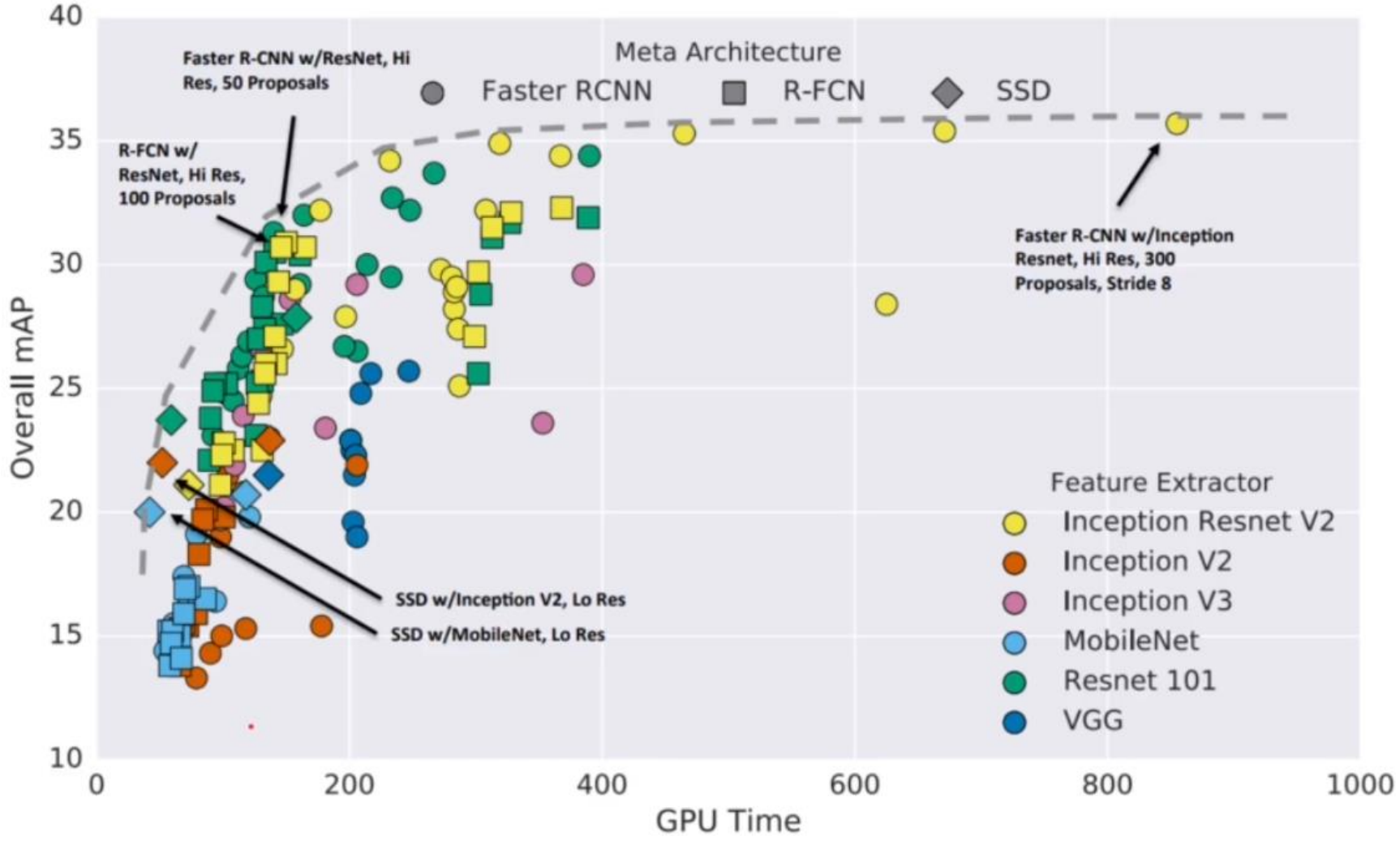


Image Source: pyimagesearch, towardsdatascience

Selecting the right model architecture

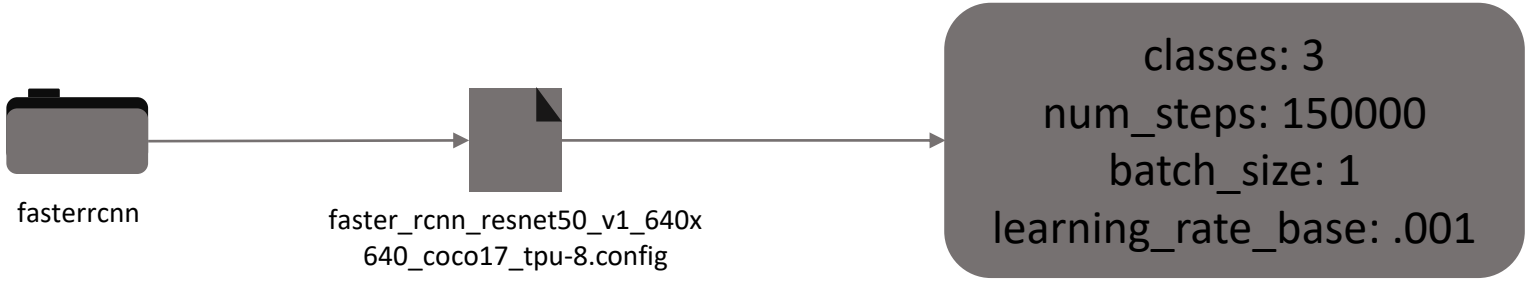
Accuracy Vs Time:



Steps for building object detector using Tensorflow 2 object detection API

1. Downloaded a pretrained Faster RCNN Resnet50 Neural Network and SSD Resnet50.
2. Downloaded configuration file corresponding to pretrained neural network.
3. Configure training using the configuration file (path to training data, number of classes, path to testing data).
4. Run training and evaluation using Tensorflow 2 API scripts.

Training Faster RCNN on local machine



- To train on own system:

From the tensorflow/models/research/ directory
python object_detection/model_main_tf2.py --pipeline_config_path trafficlightprediction/fasterrcnn/faster_rcnn_resnet50_v1_640x640_coco17_tpu-8.config --model_dir trafficlightprediction/fasterrcnn/training_process/ --alsologtostderr

- To eval on own system:

From the tensorflow/models/research/ directory
python object_detection/model_main_tf2.py --pipeline_config_path trafficlightprediction/fasterrcnn/faster_rcnn_resnet50_v1_640x640_coco17_tpu-8.config --model_dir trafficlightprediction/fasterrcnn/training_process/ --checkpoint_dir trafficlightprediction/fasterrcnn/training_process/ --alsologtostderr

- To run tensorboard test:

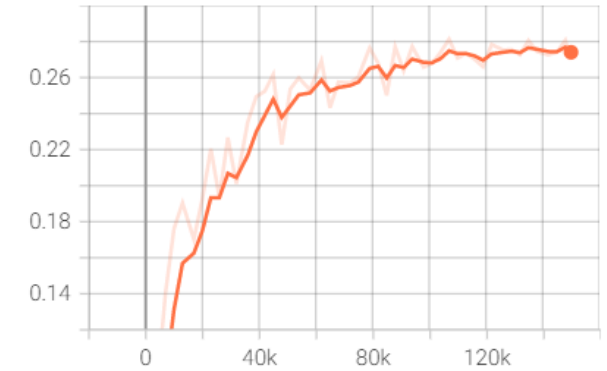
tensorboard --logdir trafficlightprediction/fasterrcnn/training_process/eval/

Evaluation of Faster RCNN

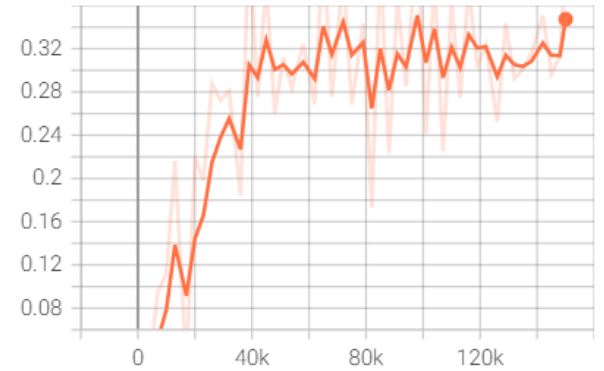
Note:
mAP at 150K step: 0.274
total time to train: 1d 10h 48m 21s

DecisionBoxes_Precision:

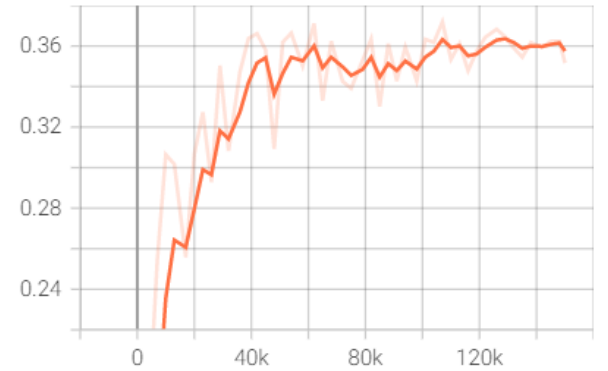
DetectionBoxes_Precision/mAP
tag: DetectionBoxes_Precision/mAP



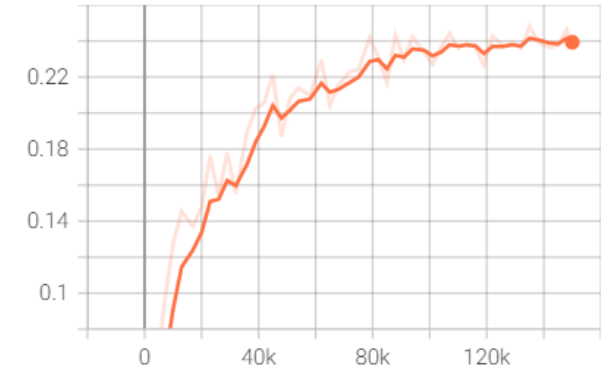
DetectionBoxes_Precision/mAP (large)
tag: DetectionBoxes_Precision/mAP (large)



DetectionBoxes_Precision/mAP (medium)
tag: DetectionBoxes_Precision/mAP (medium)

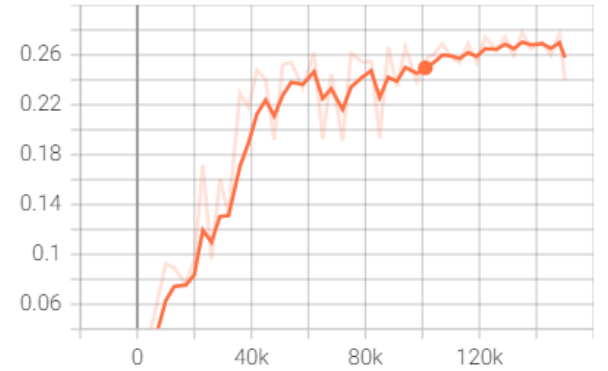


DetectionBoxes_Precision/mAP (small)
tag: DetectionBoxes_Precision/mAP (small)



DetectionBoxes_Precision/mAP@.50IOU
tag: DetectionBoxes_Precision/mAP@.50IOU

DetectionBoxes_Precision/mAP@.75IOU
tag: DetectionBoxes_Precision/mAP@.75IOU

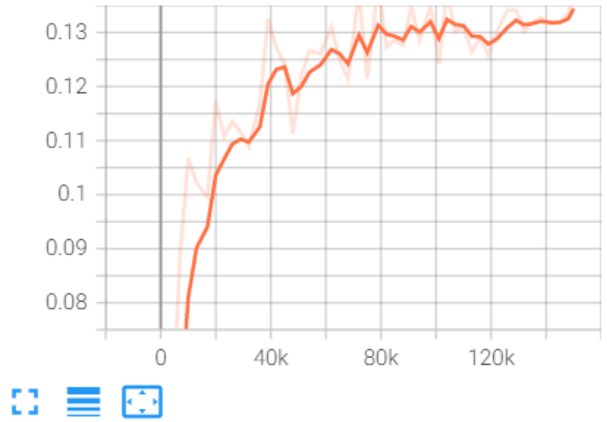


Evaluation of Faster RCNN

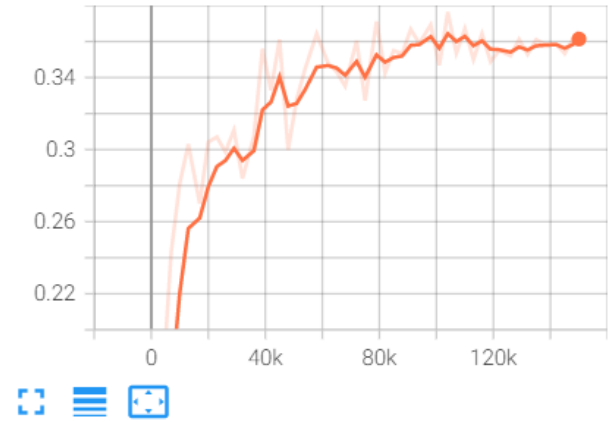
Note:
recall at 150K step: 0.1344

DecisionBoxes_Recall:

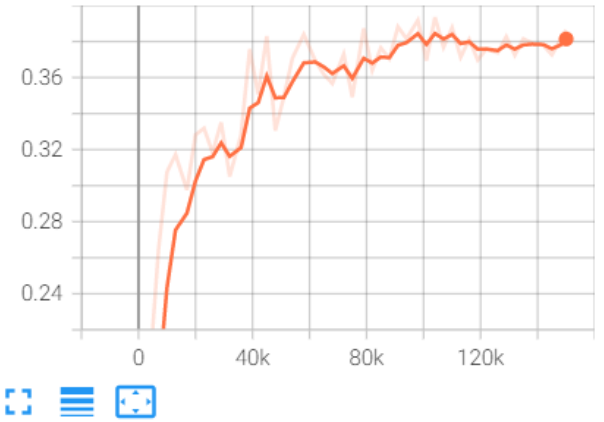
DetectionBoxes_Recall/AR@1
tag: DetectionBoxes_Recall/AR@1



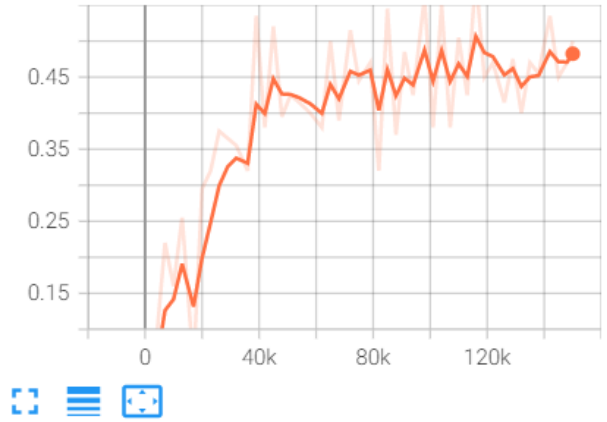
DetectionBoxes_Recall/AR@10
tag: DetectionBoxes_Recall/AR@10



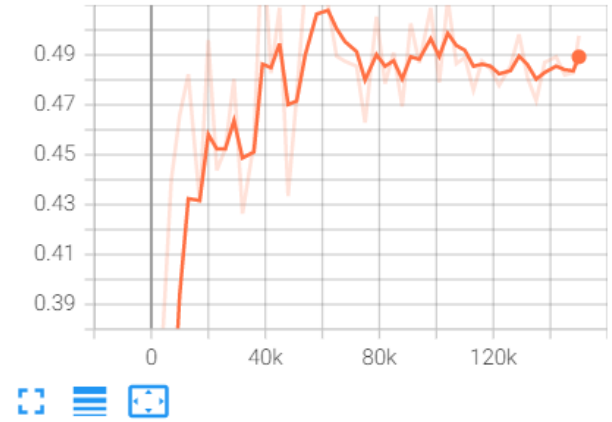
DetectionBoxes_Recall/AR@100
tag: DetectionBoxes_Recall/AR@100



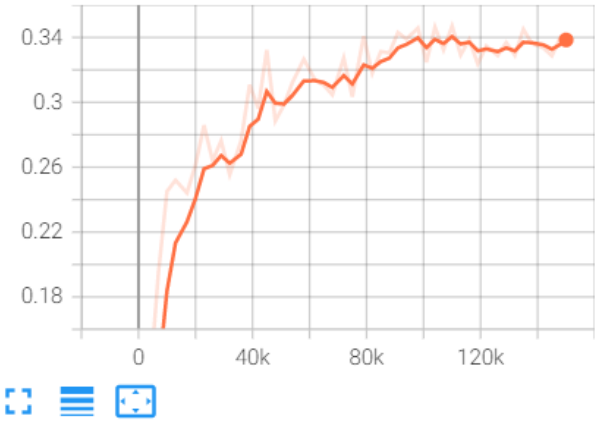
DetectionBoxes_Recall/AR@100 (large)
tag: DetectionBoxes_Recall/AR@100 (large)



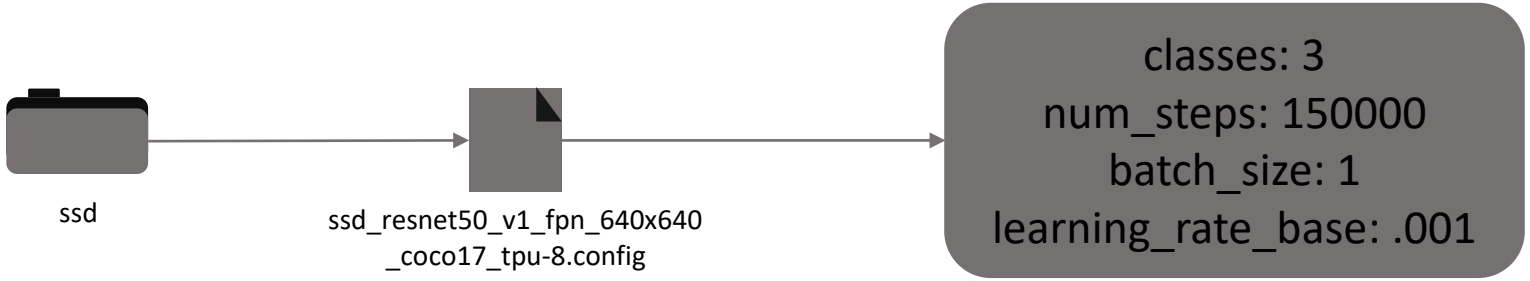
DetectionBoxes_Recall/AR@100 (medium)
tag: DetectionBoxes_Recall/AR@100 (medium)



DetectionBoxes_Recall/AR@100 (small)
tag: DetectionBoxes_Recall/AR@100 (small)



Training SSD on local machine



- To train SSD on own system:

```
# From the tensorflow/models/research/ directory
python object_detection/model_main_tf2.py --pipeline_config_path
trafficlightprediction/ssd/ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config --model_dir
trafficlightprediction/ssd/training_process/ --alsologtostderr
```

- To eval SSD on own system:

```
# From the tensorflow/models/research/ directory
python object_detection/model_main_tf2.py --pipeline_config_path
trafficlightprediction/ssd/ssd_resnet50_v1_fpn_640x640_coco17_tpu-8.config --model_dir
trafficlightprediction/ssd/training_process/ --checkpoint_dir trafficlightprediction/ssd/training_process/
--alsologtostderr
```

- To run tensorboard SSD test:

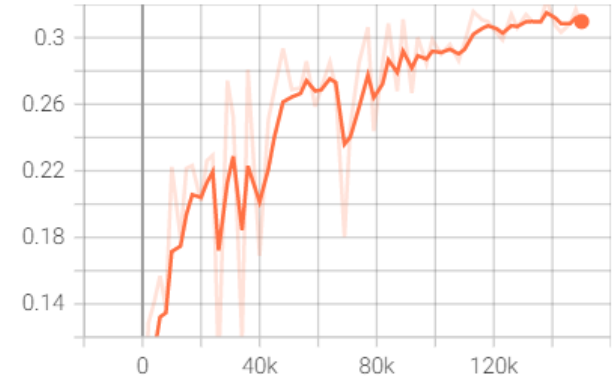
```
tensorboard --logdir trafficlightprediction/ssd/training_process/eval/
```

Evaluation of SSD

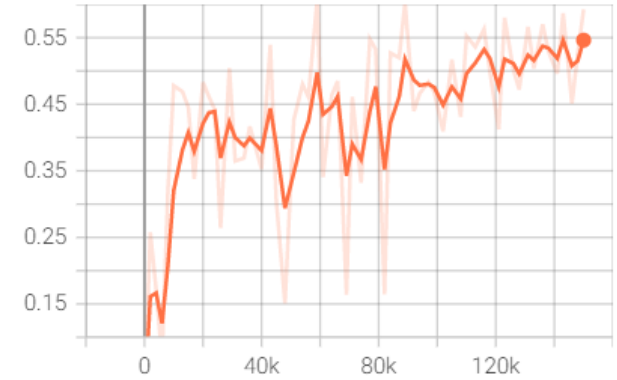
Note:
mAP at 150K step: 0.3097
total time to train: 1d 15h 44m 19s

DecisionBoxes_Precision:

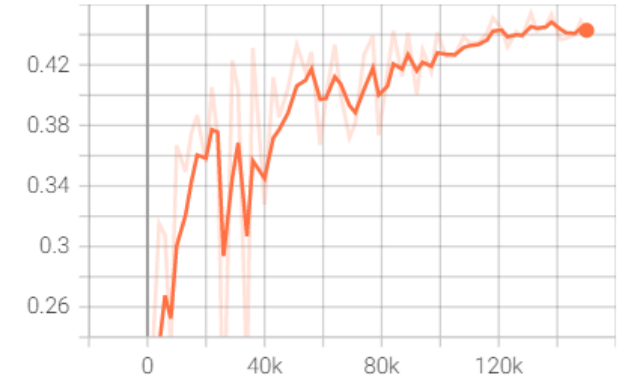
DetectionBoxes_Precision/mAP
tag: DetectionBoxes_Precision/mAP



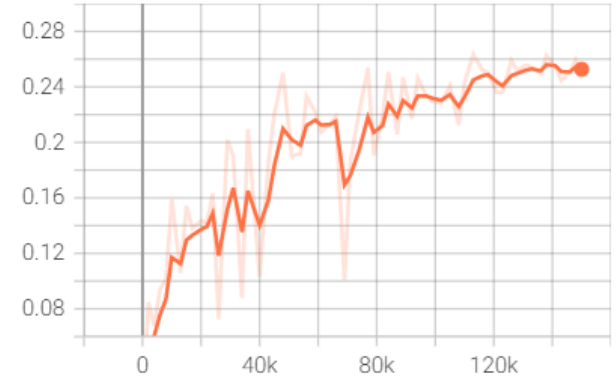
DetectionBoxes_Precision/mAP (large)
tag: DetectionBoxes_Precision/mAP (large)



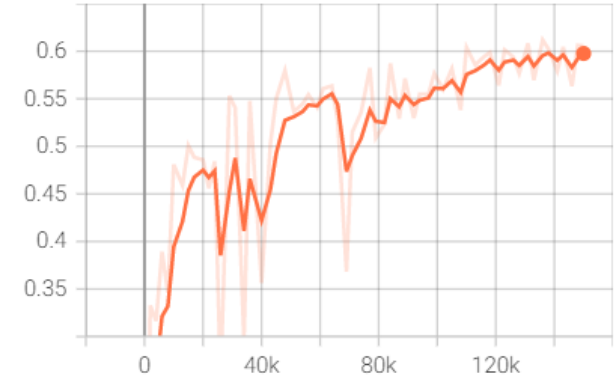
DetectionBoxes_Precision/mAP (medium)
tag: DetectionBoxes_Precision/mAP (medium)



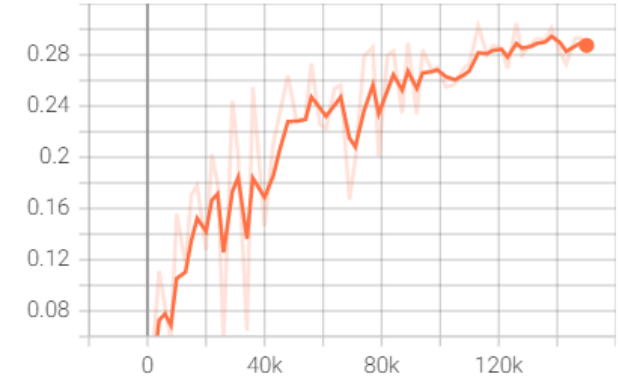
DetectionBoxes_Precision/mAP (small)
tag: DetectionBoxes_Precision/mAP (small)



DetectionBoxes_Precision/mAP@.50IOU
tag: DetectionBoxes_Precision/mAP@.50IOU



DetectionBoxes_Precision/mAP@.75IOU
tag: DetectionBoxes_Precision/mAP@.75IOU

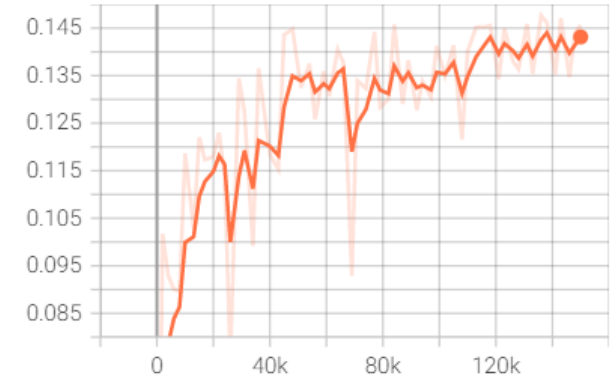


Evaluation of SSD

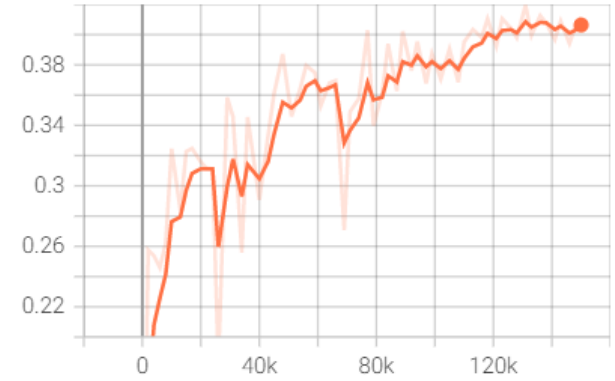
Note:
recall at 150K step: 0.1431

DecisionBoxes_Recall:

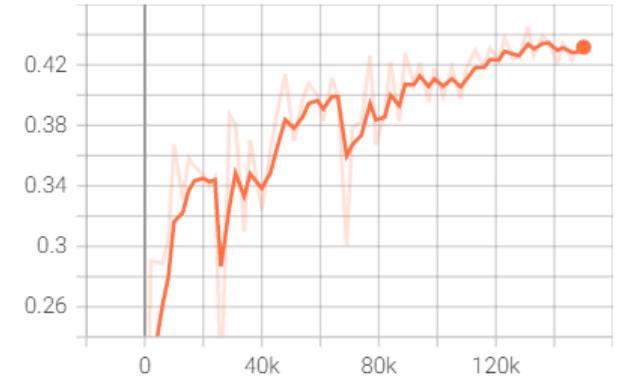
DetectionBoxes_Recall/AR@1
tag: DetectionBoxes_Recall/AR@1



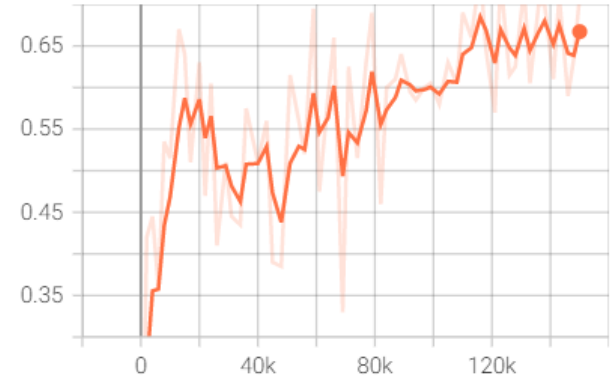
DetectionBoxes_Recall/AR@10
tag: DetectionBoxes_Recall/AR@10



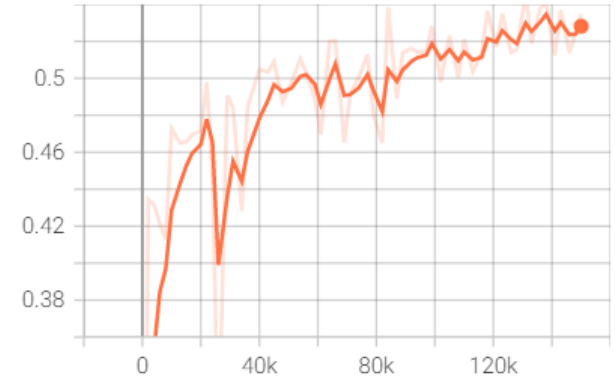
DetectionBoxes_Recall/AR@100
tag: DetectionBoxes_Recall/AR@100



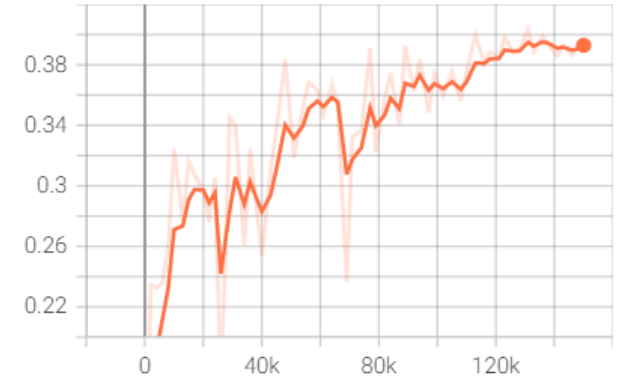
DetectionBoxes_Recall/AR@100 (large)
tag: DetectionBoxes_Recall/AR@100 (large)



DetectionBoxes_Recall/AR@100 (medium)
tag: DetectionBoxes_Recall/AR@100 (medium)



DetectionBoxes_Recall/AR@100 (small)
tag: DetectionBoxes_Recall/AR@100 (small)



Comparison of Faster RCNN and SSD

Trade-offs	Faster RCNN	SSD
Pretrained Model	faster_rcnn_resnet50_v1_640x640_coco17_tpu-8	ssd_resnet50_v1_640x640_coco17_tpu-8
Steps Trained	150k steps	150k steps
Batch size	1	1
Time Taken for training	1d 10h 48m 21s	1d 15h 44m 19s
Trained On	Nvidia GPU (Quadro M1200) – 4GB	Nvidia GPU (Quadro M1200) – 4GB
Loss/classification_loss	0.04605	0.3593
Loss/localization_loss	0.05035	0.2461
Loss/total_loss	0.1223	0.7817
mAP(0.55:0.95)	0.274	0.3091
Recall	0.1344	0.1431

Conlcusion

- mAP for SSD has been good compared to Faster RCNN for LISA Traffic Light Dataset, but training time has been longer.
- The losses while training are less in Faster RCNN as compared to SSD.
- While Faster R-CNN may require more fine-tuning, its two-stage architecture with separate region proposal and object detection stages can lead to better accuracy, especially for larger objects.
- In contrast, SSD's direct feature map-based approach can reduce computation and improve speed but may result in lower accuracy for larger objects.
- Other factors to consider when choosing between Faster R-CNN and SSD may include the hardware requirements for training and inference.

References

- https://www.kaggle.com/datasets/mbornoe/lisa-traffic-light-dataset
- Jensen MB, Philipsen MP, Møgelmose A, Moeslund TB, Trivedi MM. Vision for Looking at Traffic Lights: Issues, Survey, and Perspectives. I E E E Transactions on Intelligent Transportation Systems. 2016 Feb 3;17(7):1800-1815. Available from, DOI: 10.1109/TITS.2015.2509509
- Philipsen, M. P., Jensen, M. B., Møgelmose, A., Moeslund, T. B., & Trivedi, M. M. (2015, September). Traffic light detection: A learning algorithm and evaluations on challenging dataset. In intelligent transportation systems (ITSC), 2015 IEEE 18th international conference on (pp. 2341-2345). IEEE.
- Zhang Lizhe Cheng Fei Li Wei, Liu Kai. 2020. Object detection based on an adaptive attention mechanism.
- https://towardsdatascience.com/a-simple-step-by-step-installation-guide-for-tensorflow-tensorflow-object-detection-api-c1660d4ae533
- https://en.wikipedia.org/wiki/Object_detection
- https://machinelearningmastery.com/object-recognition-with-deep-learning/
- https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/training.html