Traffic Light Classifier: Training and Testing the object detection model(Faster-RCNN) using tensorflow:

Nidhi Arora - Author

Object detection using deep learning framework tensorflow and faster r-cnn model

## Steps involved to train a object detection classifier using a pre-trained model (Faster R-CNN inception classifier) on AWS

setting up the environment:

Hw: NVIDIA GPU(CUDA supported GPU)

CUDA 8 which is compatible with tensorflow 1.4

CUDA 9 which is compatible with tensorflow 1.5 and higher version

Sw: Install tensorflow GPU 1.4

Install object detection API from tensorflow github models directory

Set PYTHONPATH and PATH environment variables

Run object\_detection\_tutorial.ipynb to check installation.

collecting datasets:

METHOD 1: collect your own datasets and annotate the images using LabelImg tool.

use xml\_to\_csv.py script to create csv file.

use create\_tf\_record.py script to generate tf Records i.e Train.record file.

METHOD 2: use ROSBAGS for collecting Data & generating Record files from it.

company name

Downloading and configuring pre-trained model:

Download the Pre trained models “Faster\_RCNN\_V2\_inception\_coco model” to train the classifier.

Configure the “Faster\_RCNN\_inception\_coco\_sim.config” file for following parameters:

A.) num\_classes (no. of labels used in images)

B.) input\_path (Path to train.record) file

C.) inference\_graph (Path to pre-trained chk point I.e .ckpt file from ssd model folder)

D.) num\_examples (number of images to be used while evaluating)

E.) Output\_path (Path to .Pb file I.e the frozen inference graph which will be generated)

Training the classifier:

Start training the classifier using train.py file in the object\_detection folder and the above .config file.

Process followed:

**Training**

The first thing you need to do is to select the pre-trained model you would like to use. You could check and download a pre-trained model from [Tensorflow detection model zoo Github page](https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md).

Once downloaded, extract all file to the folder you had created for saving the pre-trained model files.

Next you need to copy *models/research/sample/configs/<your\_model\_name.config>* and paste it in the project repo. You need to configure 5 paths in this file. Just open this file and search for PATH\_TO\_BE\_CONFIGURED and replace it with the required path. I used pre-trained faster RCNN trained on COCO dataset and I have added modified config file (along with PATH\_TO\_BE\_CONFIGURED as comment above lines which has been modified) for same in this repo. You could also play with other hyper parameters if you want.

Now you are all set to train your model, just run the following command with models/research as present working directory.

**cmd:**

python object\_detection/legacy/train.py --train\_dir=<path\_to\_the folder\_for\_saving\_checkpoints>--pipeline\_config\_path=<path\_to\_config\_file>

company name

**Example:**

python object\_detection/legacy/train.py --train\_dir=/Users/vijendra1125/Documents/tensorflow/object\_detection/speaker\_detection/CP --pipeline\_config\_path=/Users/vijendra1125/Documents/tensorflow/object\_detection/speaker\_detection/faster\_rcnn\_resnet101\_coco.config

Let it train till loss will be below 0.1 or even lesser. once you see that loss is as low as you want then give keyboard interrupt. Checkpoints will be saved in CP folder. Now its time to generate inference graph from saved checkpoints.

**Cmd:**

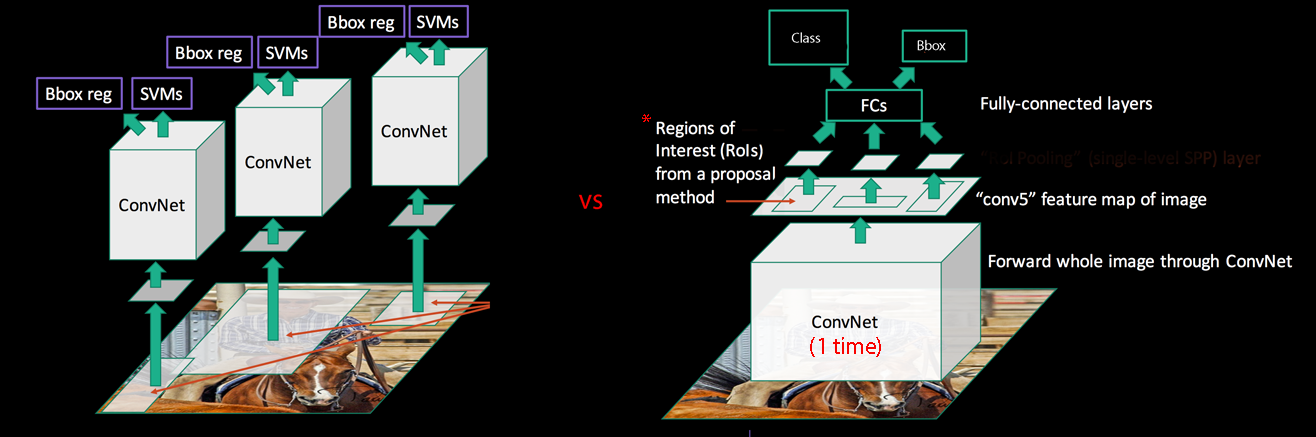
python object\_detection/export\_inference\_graph.py --input\_type=image\_tensor --pipeline\_config\_path=<path\_to\_config\_file> --trained\_checkpoint\_prefix=<path to saved checkpoint> --output\_directory=<path\_to\_the\_folder\_for\_saving\_inference\_graph>

**Example:**

python object\_detection/export\_inference\_graph.py --input\_type=image\_tensor --pipeline\_config\_path=/Users/vijendra1125/Documents/tensorflow/object\_detection/speaker\_detection/faster\_rcnn\_resnet101\_coco.config --trained\_checkpoint\_prefix=/Users/vijendra1125/Documents/tensorflow/object\_detection/speaker\_detection/CP/model.ckpt-1691 --output\_directory=/Users/vijendra1125/Documents/tensorflow/object\_detection/speaker\_detection/IG

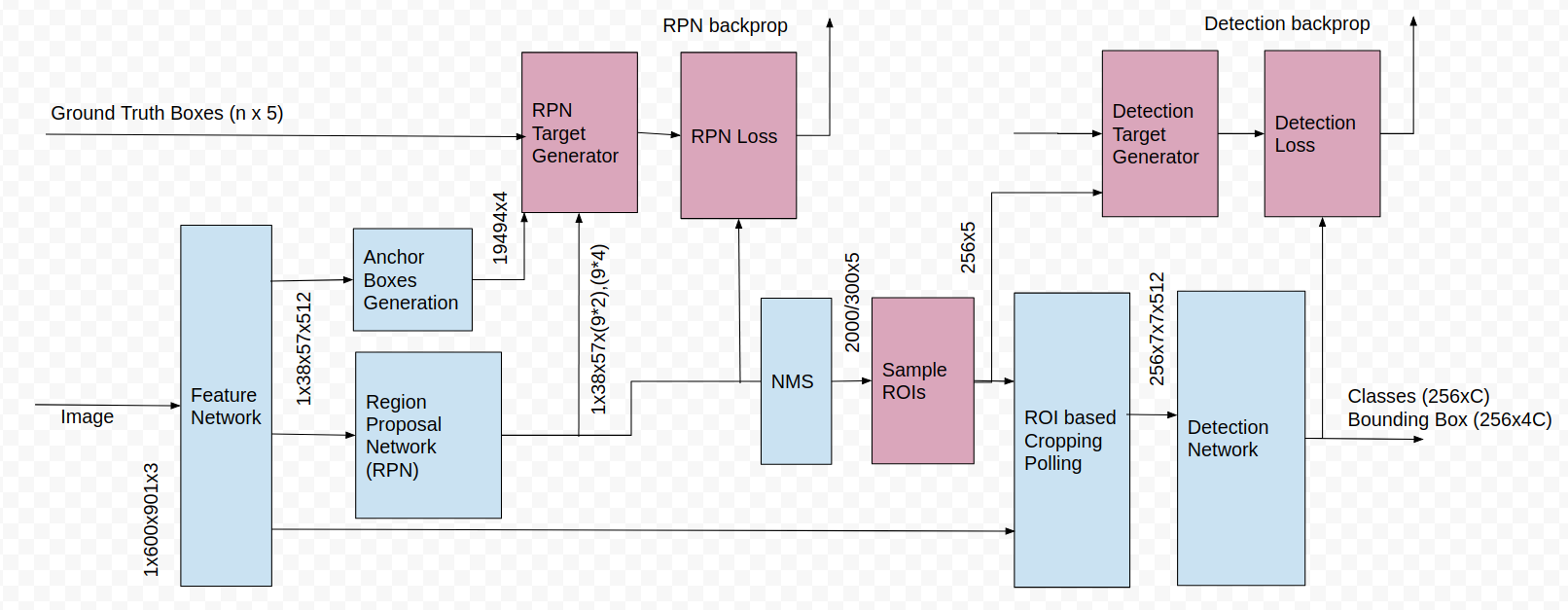
Evaluating the classifier:

Evaluate the trained model using object\_detection\_tutorial.ipynb file and the generated .Pb file for seeing the classification results on the test images with your inference graph.

Difference between RCNN and Faster-RCNN

Performance and diagram showing difference

Faster RCNN working:



**References**:

Training the model: <https://github.com/alex-lechner/Traffic-Light-Classification>

Training & Testing process followed(Blog): <https://medium.com/@vijendra1125/tensorflow-api-custom-object-detection-55444de6562d>

ipython script (eval.ipynb) used for evaluating the model and modified accordingly : https://github.com/vijendra1125/Tensorflow\_Object\_detection\_API-Custom\_Faster\_RCNN

Ipynb used to evaluate the trained model is located @:

[ndharora2008@gmail.com](mailto:ndharora2008@gmail.com) —> Google Colab —> Python2\_gpu\_Object\_detection.ipynb(Process to upload the Training and Testing Data to Google Drive using Colab) + tl\_classification.ipynb (Evaluation File for running inference graph)

Model was trained on an AWS machine.