**TENSORFLOW CPU OBJECT DETECTION**

**FASTER RCNN**

**STEPS:**

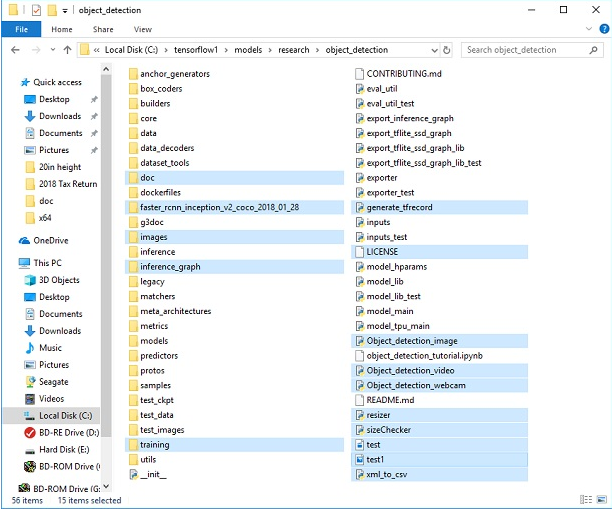
1)Install Anaconda in the local machine

2)Download the Github repository and extract anywhere. For eg: I extracted it in “C:\tensorflow”. Link: <https://github.com/tensorflow/models>

3)The link below gives different models to be used for detecting the objects. For eg: I used faster rcnn. So download [faster\_rcnn\_inception\_v2\_coco](http://download.tensorflow.org/models/object_detection/faster_rcnn_inception_v2_coco_2018_01_28.tar.gz) tar file and extract it.

Link:<https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md>

4)Download the github repository given below and extract the files in the place where the model is extracted “C:\tensorflow\**models\research\object\_detection”** and move the extracted model(faster rcnn) also inside the object\_detection folder. Link: <https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10>



5)Remove the files as stated below:

* All files in \object\_detection\images\train and \object\_detection\images\test
* The “test\_labels.csv” and “train\_labels.csv” files in \object\_detection\images
* All files in \object\_detection\training
* All files in \object\_detection\inference\_graph

6)Set up an anaconda environment and activate it: Go to the base directory of tensorflow and then type the command in cmd

conda create -n rakhesh pip python=3.6

activate rakhesh

7)Install tensorflow version 1.2 using the following command after activating virtual environment

pip install --ignore-installed --upgrade tensorflow==1.12

8)Install the necessary packages :

conda install -c anaconda protobuf

pip install pillow

pip install lxml

pip install Cython

pip install contextlib2

pip install jupyter

pip install matplotlib

pip install pandas

pip install opencv-python

9)Set the pythonpath to models and research and research\slim folders;:

set PYTHONPATH=C:\tensorflow1\models;C:\tensorflow1\models\research;C:\tensorflow1\models\research\slim

echo %PYTHONPATH%

set PATH=%PATH%;PYTHONPATH

echo %PATH%

10)now protobufs need to be compiled. Follow the commands below to do it:

cd C:\tensorflow1\models\research

**PROTOS:** protoc --python\_out=. .\object\_detection\protos\anchor\_generator.proto .\object\_detection\protos\argmax\_matcher.proto .\object\_detection\protos\bipartite\_matcher.proto .\object\_detection\protos\box\_coder.proto .\object\_detection\protos\box\_predictor.proto .\object\_detection\protos\eval.proto .\object\_detection\protos\faster\_rcnn.proto .\object\_detection\protos\faster\_rcnn\_box\_coder.proto .\object\_detection\protos\grid\_anchor\_generator.proto .\object\_detection\protos\hyperparams.proto .\object\_detection\protos\image\_resizer.proto .\object\_detection\protos\input\_reader.proto .\object\_detection\protos\losses.proto .\object\_detection\protos\matcher.proto .\object\_detection\protos\mean\_stddev\_box\_coder.proto .\object\_detection\protos\model.proto .\object\_detection\protos\optimizer.proto .\object\_detection\protos\pipeline.proto .\object\_detection\protos\post\_processing.proto .\object\_detection\protos\preprocessor.proto .\object\_detection\protos\region\_similarity\_calculator.proto .\object\_detection\protos\square\_box\_coder.proto .\object\_detection\protos\ssd.proto .\object\_detection\protos\ssd\_anchor\_generator.proto .\object\_detection\protos\string\_int\_label\_map.proto .\object\_detection\protos\train.proto .\object\_detection\protos\keypoint\_box\_coder.proto .\object\_detection\protos\multiscale\_anchor\_generator.proto .\object\_detection\protos\graph\_rewriter.proto .\object\_detection\protos\calibration.proto .\object\_detection\protos\flexible\_grid\_anchor\_generator.proto

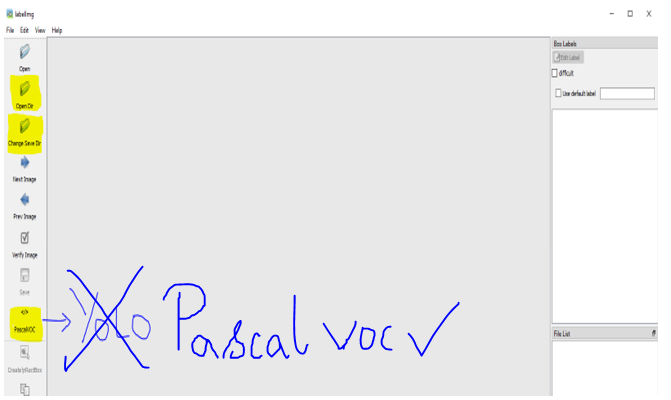
For simplicity copy the above command completely and paste it virtual environment and run it and to build it use the commands below.

python setup.py build

python setup.py install

11) To run the object detection algorithm we need data and labels. Data can be collected from google by using **BULK IMAGE DOWNLOADER** that is available as an extension in google chrome. Once the data is collected we need to label it. This can be done by using **labelImg** software. Follow the steps in github to download it for the respective OS:<https://github.com/tzutalin/labelImg> (or) Downloads list for windows: <https://tzutalin.github.io/labelImg/>

12) **)** Once the labelImg is downloaded and installed start it either from terminal or by manually. To start through terminal type the command **labelImg** in terminal



Click on the **open dir** to load the downloaded images folder. Now click on the **change** **save dir** to save the xml in **\images\train and  \images\test folder.** Keep both the images and xml files in the same respective train and test folder.Both the folders should be available in the same images directory. Dont change the **pascalVOC format** . And then press **W** to start the bounding boxes. After each image is done with the bounding boxes save the image with **ctrl+s** to save the annotations in the **labels** folder. Repeat this until every image is labelled. Every image is found in the **File list** in **right-bottom.**

13)Go to the object detection folder and execute the command to convert xml to csv file.

python xml\_to\_csv.py

CSV files will be created in the images folder.

**14)now tf records must be generated:** to generate it run the command:

Before that open the tf\_generate.py and make the changes to adhere the classes which we use:

For eg:

REPLACE THIS

# TO-DO replace this with label map

def class\_text\_to\_int(row\_label):

if row\_label == 'nine':

return 1

elif row\_label == 'ten':

return 2

elif row\_label == 'jack':

return 3

elif row\_label == 'queen':

return 4

elif row\_label == 'king':

return 5

elif row\_label == 'ace':

return 6

else:

None

WITH THIS

# TO-DO replace this with label map

def class\_text\_to\_int(row\_label):

if row\_label == 'softdrinks':

return 1

else:

None

Now execute the tf\_generate.py in object\_detection folder:

python generate\_tfrecord.py --csv\_input=images\train\_labels.csv --image\_dir=images\train --output\_path=train.record

python generate\_tfrecord.py --csv\_input=images\test\_labels.csv --image\_dir=images\test --output\_path=test.record

14)now save a new file named labelmap.pbtxt in C:\tensorflow1\models\research\object\_detection\training with the following texts inside. The text in this file denotes the class names like :

item {

id: 1

name: 'softdrinks'

}

**15)Now training must be configured:**

Navigate to C:\tensorflow1\models\research\object\_detection\samples\configs and copy the faster\_rcnn\_inception\_v2\_pets.config file into the \object\_detection\training directory. Then, open the file with a text editor. There are several changes to make to the .config file, mainly changing the number of classes and examples, and adding the file paths to the training data.

Make the following changes to the faster\_rcnn\_inception\_v2\_pets.config file. Note: The paths must be entered with single forward slashes (NOT backslashes), or TensorFlow will give a file path error when trying to train the model! Also, the paths must be in double quotation marks ( " ), not single quotation marks ( ' ).

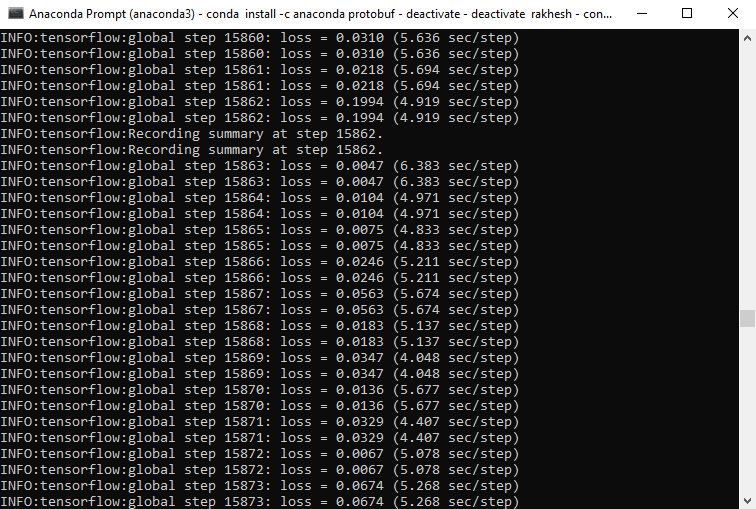
* Line 9. Change num\_classes to the number of different objects you want the classifier to detect. For the softdrinks detector, it would be num\_classes: 1.
* Line 106. Change fine\_tune\_checkpoint to:
  + fine\_tune\_checkpoint: "C:/tensorflow1/models/research/object\_detection/faster\_rcnn\_inception\_v2\_coco\_2018\_01\_28/model.ckpt"
* Lines 123 and 125. In the train\_input\_reader section, change input\_path and label\_map\_path to:
  + input\_path: "C:/tensorflow1/models/research/object\_detection/train.record"
  + label\_map\_path: "C:/tensorflow1/models/research/object\_detection/training/labelmap.pbtxt"
* Line 130. Change num\_examples to the number of images you have in the \images\test directory (not necessary, if the testing image exceedes 1101 then we need to change orelse we can leave it)
* Lines 135 and 137. In the eval\_input\_reader section, change input\_path and label\_map\_path to:
  + input\_path : "C:/tensorflow1/models/research/object\_detection/test.record"
  + label\_map\_path: "C:/tensorflow1/models/research/object\_detection/training/labelmap.pbtxt"

Save the file after the changes have been made. That’s it! The training job is all configured and ready to go!

**16)To train the model:**

Simply move train.py from /object\_detection/legacy into the /object\_detection folder and then continue following the steps below.

python train.py --logtostderr --train\_dir=training/ --pipeline\_config\_path=training/faster\_rcnn\_inception\_v2\_pets.config

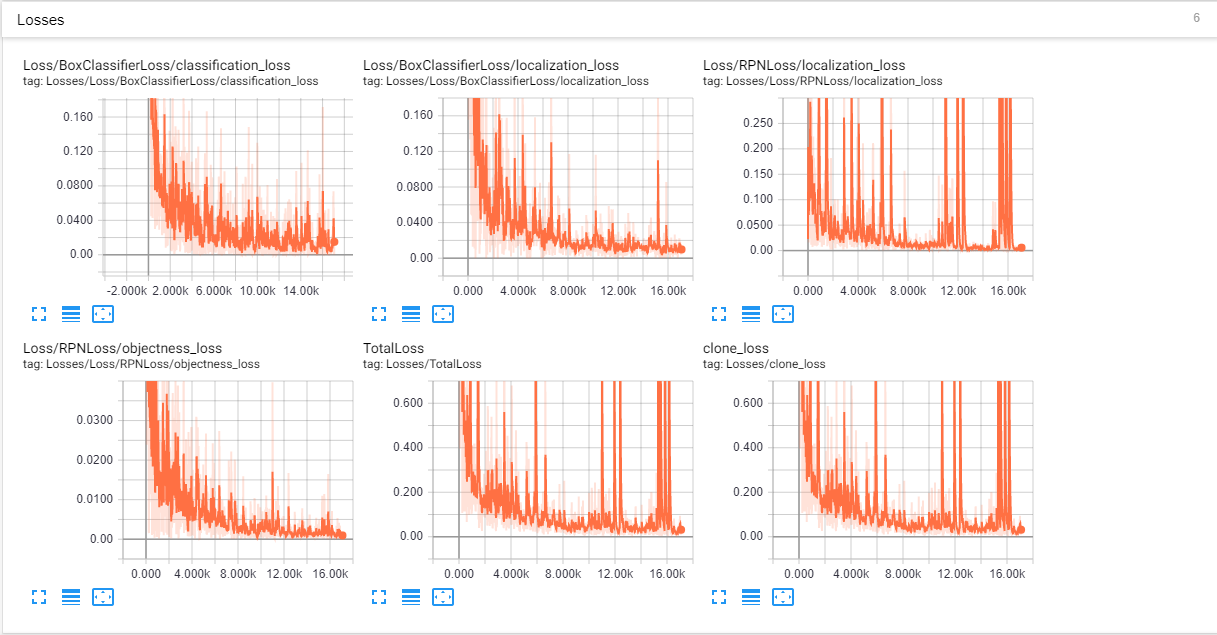


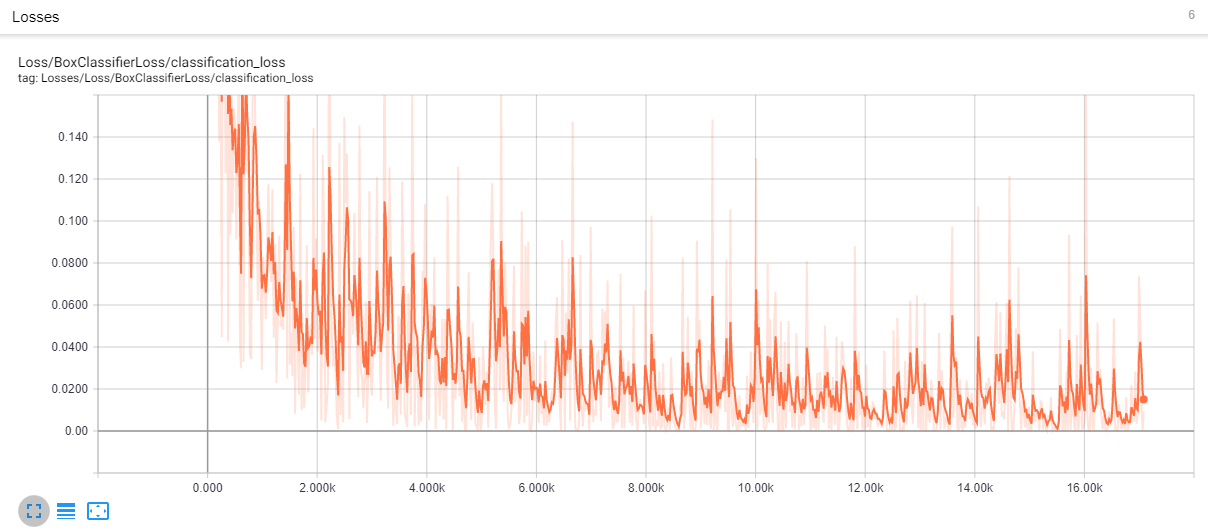
NOTE:If the error appears as maximum box coordinate is greater than 1.000 or some value. Go to  ./core/box\_list\_ops.py and change the value of box co-ordinate to properly anchor the images. Change in this line: max\_assert = tf.Assert(tf.greater\_equal(1.1, box\_maximum), change the value of 1.1 according to the value in error orelse change the following line as false in the py script: check\_range=False

The average loss should be less than 0.05 for better training. This can be viewed with the help of tensorboard. Command is: in object\_detection type the command:

tensorboard --logdir=training

If the tensorboard site is not reachable. Type the following command to execute in local host: tensorboard --logdir=training --host localhost





After certain iterations and epochs, the loss will get always less than 0.05 . now make and keyboard interrupt in anaconda virtual environment by pressing ctrl+c

**17)To export inference graph:**

Once the training is over copy the highest number from \object\_detection\training folder and replace it in ‘XXXX’ the command and execute the command from terminal In object detection folder.

python export\_inference\_graph.py --input\_type image\_tensor --pipeline\_config\_path training/faster\_rcnn\_inception\_v2\_pets.config --trained\_checkpoint\_prefix training/model.ckpt-XXXX --output\_directory inference\_graph

This creates a frozen\_inference\_graph.pb file in the \object\_detection\inference\_graph folder. The .pb file contains the object detection classifier.

**18)Detection of objects:**

Python scripts to test it out on an image, video, or webcam feed are available in object detection folder. For an image to be tested place the image in the object detection folder and specify the num\_classes(number of classes) in the py script and then put the image path in the script. Similarly for video also follow the same steps. Enter “idle” in the prompt. Idle will open and run the script to see the detected object.

