FULL ASS GUIDE FOR RUBX CUBE ALARM

**Python script using imageSnap to collect training data:**

import os

import subprocess

import random

while (True):

#check input of user

value = input("Type U for unsolved, S for solved, T for tests, E for escape\n")

rand = random.randint(1,10000)

if value == "U" or value == "u":

print("U!")

name = "imagesnap unsolved" + str(rand) + ".jpg -d \"USB Camera #3\""

os.system(name)

print("done!")

elif value == "S" or value == "s":

print("S!")

name = "imagesnap solved" + str(rand) +".jpg -d \"USB Camera #3\""

os.system(name)

print("done!")

elif value == "T" or value == "t":

print("T!")

name = "imagesnap test" + str(rand) + ".jpg -d \"USB Camera #3\""

os.system(name)

print("done!")

elif value == "B" or value == "b":

print("B!")

name = "imagesnap background" + rand + ".jpg -d \"USB Camera #3\""

os.system(name)

print("done!")

elif value == "E" or value == "e":

break

print("bye!")

else:

print("nope")

Next steps:

Instead of creating a network from scratch, I’m going to be re-working the one created and used in Tensorflow-for-poets-2

https://github.com/tensorflow/hub/raw/master/examples/image\_retraining/retrain.py

We’re going to run this code:

python -m scripts.retrain \ --bottleneck\_dir=tf\_files/bottlenecks \ --model\_dir=tf\_files/models/"${ARCHITECTURE}" \ --summaries\_dir=tf\_files/training\_summaries/"${ARCHITECTURE}" \ --output\_graph=tf\_files/retrained\_graph.pb \ --output\_labels=tf\_files/retrained\_labels.txt \ --image\_dir=tf\_files/flower\_photos

|  |  |
| --- | --- |
| python -m scripts.retrain | Calling command -m – module name  Executes as a script |
| --bottleneck\_dir=tf\_files/bottlenecks | Stores final bottleneck values in directory |
| --summaries\_dir=tf\_files/training\_summaries/"${ARCHITECTURE}" \ --output\_graph=tf\_files/retrained\_graph.pb \ --output\_labels=tf\_files/retrained\_labels.txt \ | direct to the different directories of the scripts |
| --image\_dir=tf\_files/flower\_photos | Image directory (which folder to train from) |

Python retrain.py --image\_dir ~/Documents/RubixCubeAlarm/images –image\_dir --summaries\_dir=tf\_files/training\_summaries/"${ARCHITECTURE}" \ --output\_graph=tf\_files/retrained\_graph.pb \ --output\_labels=tf\_files/retrained\_labels.txt \ -- --saved\_model\_dir = tf\_files

imageSnap -l to see what the device is listed as

tensorboard --logdir /tmp/retrain\_logs

tensorboard --logdir /tmp/test

So in order to visualize, save, etc the trainig model, we have to output the summaries to a specific location which we will then direct TensorBoard to

Commands: --summaries\_dir and –logdir

This is the line in the code that uses that:

*# Merge all the summaries and write them out to the summaries\_dir*

*merged = tf.summary.merge\_all()*

*train\_writer = tf.summary.FileWriter(FLAGS.summaries\_dir + '/train',*

*sess.graph)*

*validation\_writer = tf.summary.FileWriter(*

*FLAGS.summaries\_dir + '/validation')*

|  |  |
| --- | --- |
| Training data | Python retrain.py --image\_dir ~/Documents/RubixCubeAlarm/images --summaries\_dir=/tmp/test/6 \ --output\_graph=/tmp/test/6 \ --saved\_model\_dir=/tmp/test/6 --how\_many\_training\_steps=100 -- |
| Visualizing data with tensorboard | tensorboard --logdir /tmp/test/1 |

Python retrain.py --image\_dir ~/Documents/RubixCubeAlarm/images --summaries\_dir= tmp/training\_summaries/"${ARCHITECTURE}" \ --output\_graph=tf\_files/retrained\_graph.pb \ --output\_labels=tf\_files/retrained\_labels.txt \ -- --saved\_model\_dir = tf\_files

tensorboard --logdir /tf\_files3/training\_summaries

tensorboard --logdir /tmp/test

Python retrain.py --image\_dir ~/Documents/RubixCubeAlarm/images --summaries\_dir=/tmp/test/2 \ --output\_graph=/tmp/test/2 \ --saved\_model\_dir=/tmp/test/2 --how\_many\_training\_steps=75

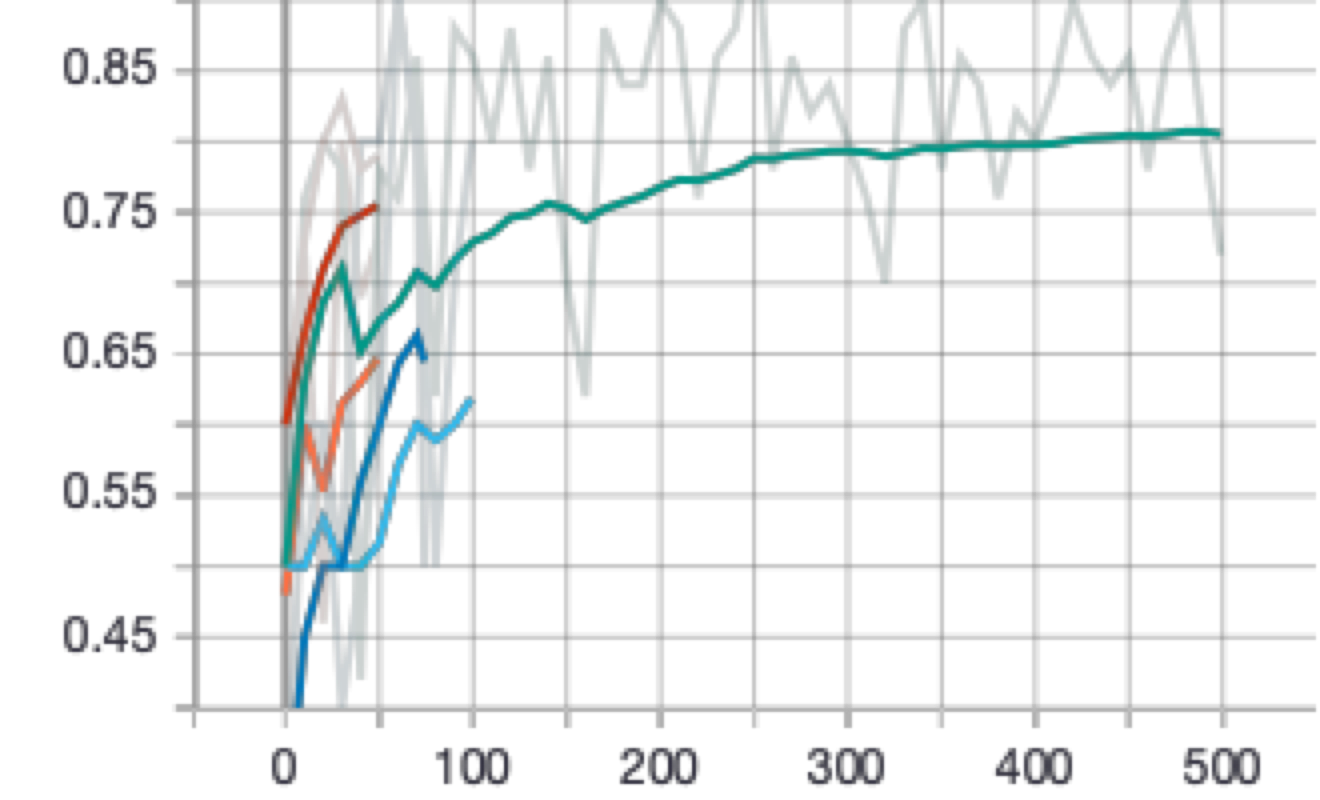
Python retrain.py --image\_dir ~/Documents/RubixCubeAlarm/images --summaries\_dir=/tmp/test/3 \ --output\_graph=/tmp/test/3 --how\_many\_training\_steps=50

|  |
| --- |
| python label\_image.py --graph=/tmp/output\_graph.pb --labels=/tmp/output\_labels.txt --input\_layer=Placeholder --output\_layer=final\_result --image=$HOME/Documents/RubixCubeAlarm/images/test.jpg |

\*\*UPGRADED CODE TO retrain\_upgraded.py and label\_image\_upgraded.py

Test 6: 2000 steps

Python retrain.py --image\_dir ~/Documents/RubixCubeAlarm/images --summaries\_dir=/tmp/test/6 \ --output\_graph=/tmp/test/6/output\_graph.pb \ --how\_many\_training\_steps=2000 --output\_labels=/tmp/test/6/output\_labels.txt

 Green: Validation maxes out at 0.8

python label\_image\_upgraded.py --graph=/tmp/test/6/output\_graph.pb --image=$HOME/Documents/RubixCubeAlarm/images/test.jpg --labels=/tmp/test/6/output\_labels.txt

Test 7: trying to save graph to tmp/test/6

Python retrain\_upgraded.py --image\_dir ~/Documents/RubixCubeAlarm/images --summaries\_dir=/tmp/test/6 --output\_graph=/tmp/test/6/output\_graph.pb --how\_many\_training\_steps=100 --output\_labels=/tmp/test/6/output\_labels.txt

TODO: Add brightness adjustments

--random\_brightness=20

--how\_many\_training\_steps=1000

SAVING TO TF\_FILES TEST: **WORKING**

Python retrain\_upgraded.py --image\_dir ~/Documents/RubixCubeAlarm/images --how\_many\_training\_steps=100 --model\_dir=tf\_files --output\_labels=tf\_files/output\_labels.txt --summaries\_dir=tf\_files/retrain\_logs --output\_graph=tf\_files/output\_graph.pb

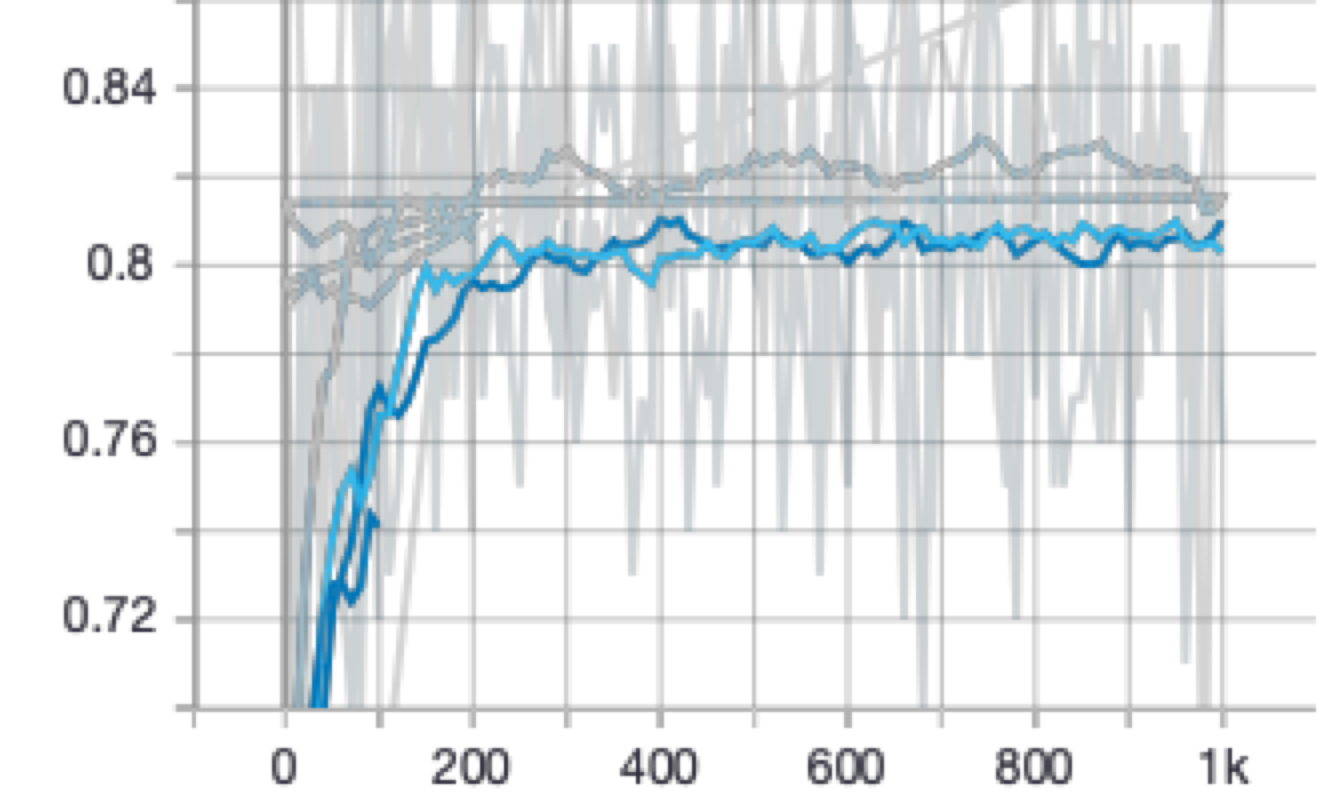
python label\_image\_upgraded.py --graph=tf\_files/output\_graph.pb --labels=/tmp/output\_labels.txt --image=$HOME/Documents/RubixCubeAlarm/images/test.jpg

This included just 100 steps, so I am interested if running longer will give better prediction

Also will be adding on --random\_brightness=20.

I’ll be saving this to tf\_files/2

Python retrain\_upgraded.py --image\_dir ~/Documents/RubixCubeAlarm/images --how\_many\_training\_steps=100 --model\_dir=tf\_files/7 --output\_labels=tf\_files/7/output\_labels.txt --summaries\_dir=tf\_files/7/retrain\_logs --output\_graph=tf\_files/7/output\_graph.pb --print\_misclassified\_test\_images True --random\_brightness=10

Currently the validation is getting stuck at 0.8 ish.

Tried increasing the depth, but no effect

Currently just trying to get Tensorboard to display the images…

Okay t

This code to the left at least gets the image summary included. Hot dang

Switching architecture to mobile net:

Testing on flowers!!!

Python retrain\_upgraded.py --image\_dir ~/Documents/RubixCubeAlarm/flower\_photos --how\_many\_training\_steps=100 --model\_dir=tf\_files/8 --output\_labels=tf\_files/8/output\_labels.txt --summaries\_dir=tf\_files/8/retrain\_logs --output\_graph=tf\_files/8/output\_graph.pb --print\_misclassified\_test\_images True --architecture mobilenet\_1.0\_224

Success!!

Working on runner.py:

python label\_image\_upgraded.py --image=$HOME/Documents/RubixCubeAlarm/tensorflow-for-poets-2/scripts/test.jpg --graph=tf\_files/6/output\_graph.pb --labels=tf\_files/6/output\_labels.txt