Report on challenge of Finding Phone

**Step 1) Data Preprocessing**

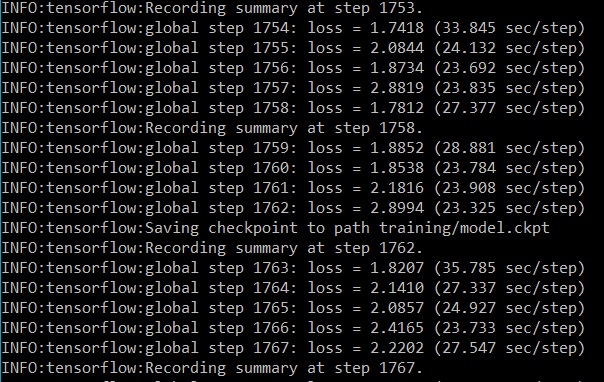
The dataset consists of 134 images and text file with labels for each image. Initial step was to create a tfrecord file for the data and splitting the data into train and test. Record file is generated using the tfrecord.py file in the Extras folder.

The record file generated is then used in the model config file as input along with \*.pbtxt file which contains list of all the objects to be detected. You can find the config file in training folder.

**Step 2) Training**

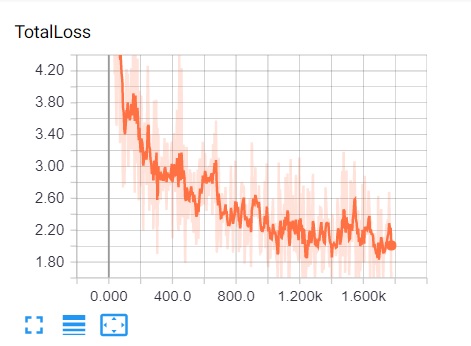
I have used Google Tensorflow Object Detection API to detect the phone. I am using SSD mobilenet pre-trained model, and then using transfer learning to train the model to detect another object, in this case, phone. The benefit of transfer learning is training can be quicker and few sample images are required. SSD mobilenet is used for Real-time Detection of objects in Video due to its high FPS the speed of detection is faster as compared to other models. But there is a trade off in terms of accuracy and speed. For this case, it works well.

After Training for 1700 time-steps



Tensor Board

Graph of loss



**Step 3) Testing the accuracy**

After training the model, it is saved and modelname.ckpt, model.index and model.meta file are used to generate the frozen\_inference.pb file which is used for testing. The inference file is generated using export\_inference\_graph.py file. (It is in models🡪Research🡪Object\_detection folder).

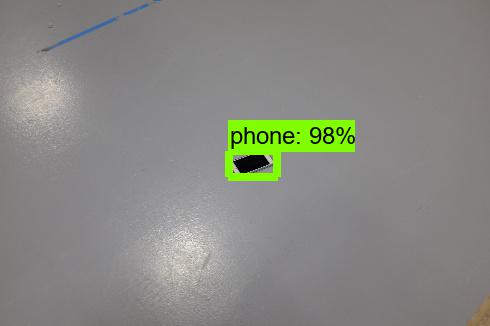
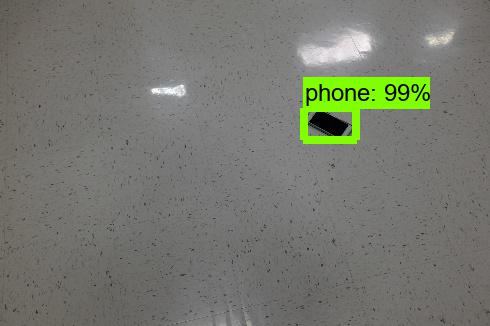
Some of the results

0.235,0.474 0.470,0.479

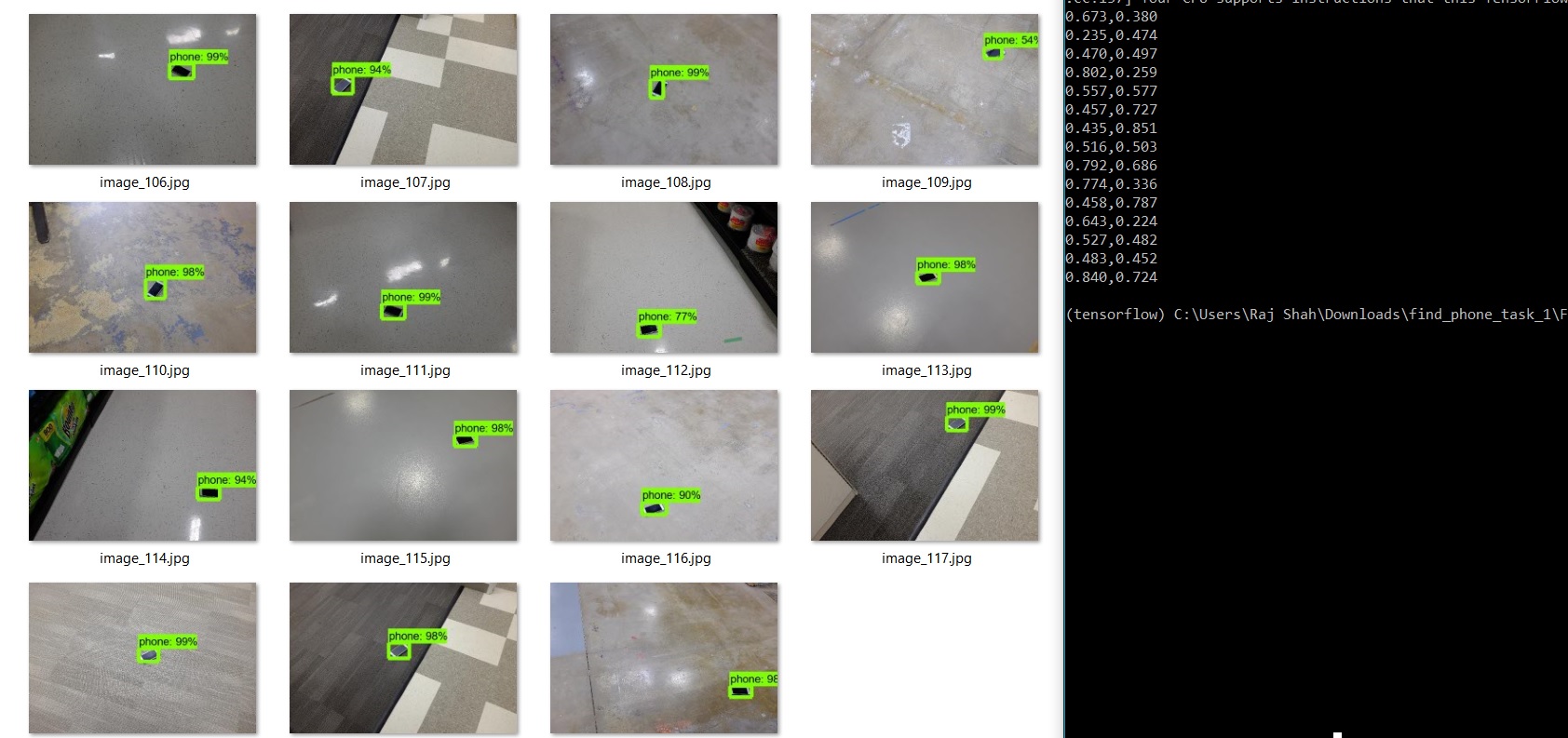
 

0.457,0.727 0.435,0.851

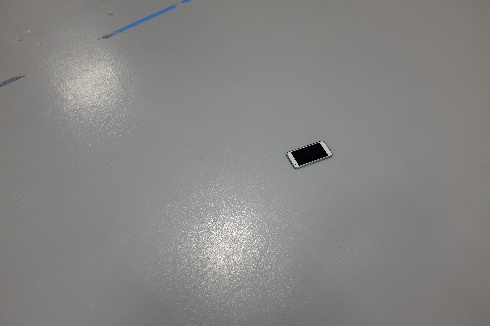
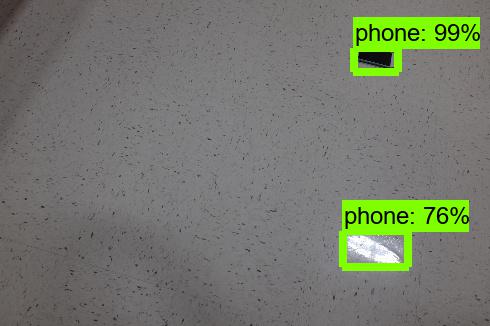
 

0.516,0.503 0.792,0.686

Results



**Some Missed Cases which fail to detect and False Positives**

I have also included a paper in extras folder on SSD: Single Shot Multibox detector for more information. It performs well on COCO, PASCAL VOC and ILSVRC datasets.

Models comparison:

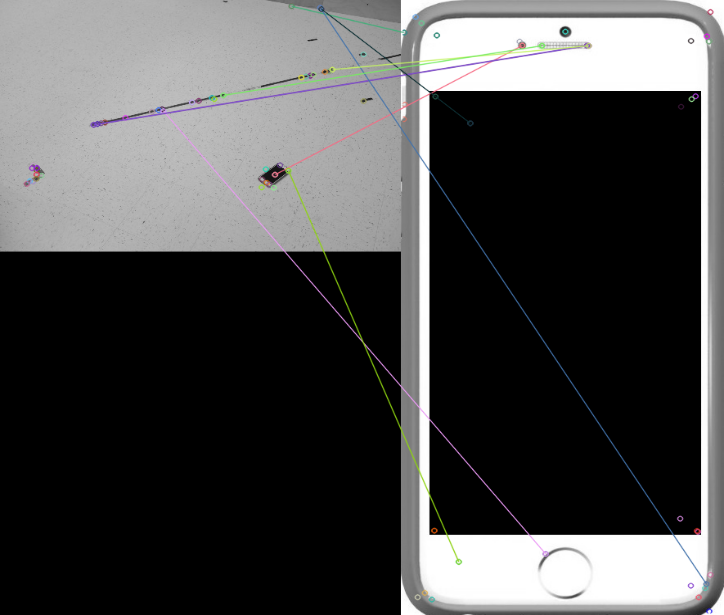
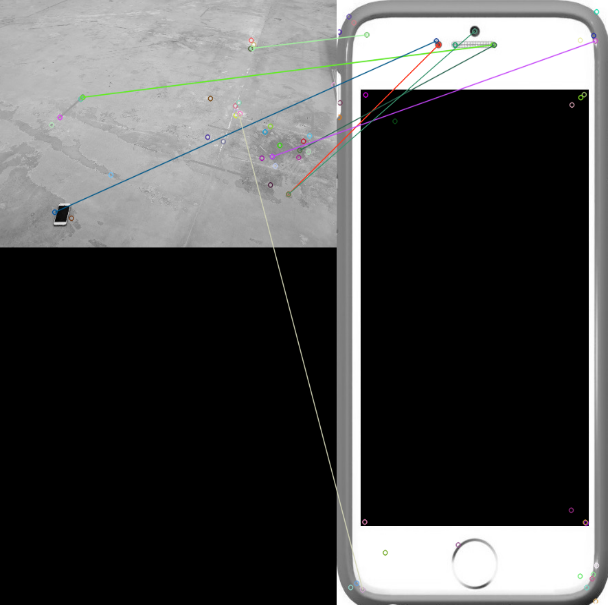
<https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md>

**How to improve Accuracy?**

1. Increase number of training samples, this can be generated by using Generative Adversarial Networks (GANs) if it is not possible to generate manually. It makes use of Bayes theorem to give new samples from the probability distribution. It has two networks which compete against each other. (Paper included in extras folder to read more).
2. Make use of resnet model for training, I was training on CPU so was not able to train on resnet model as it requires high computation power. So, if GPU access is available train on that which would improve accuracy.

**Alternate Attempt**

I also tried using another method by just using OpenCV SIFT (Scale Invariant Feature Transform) feature detector. But results were not that great. I took an image of phone and tried to map the features of phone on the test image. There is a separate folder “Sift\_phone” which has the files

The SIFT feature detector returns Keypoints and descriptors from the phone on right and tries to map them on the query image. I thought of trying this as its scale invariant, so as to avoid the problem of orientation of phone in all query images.