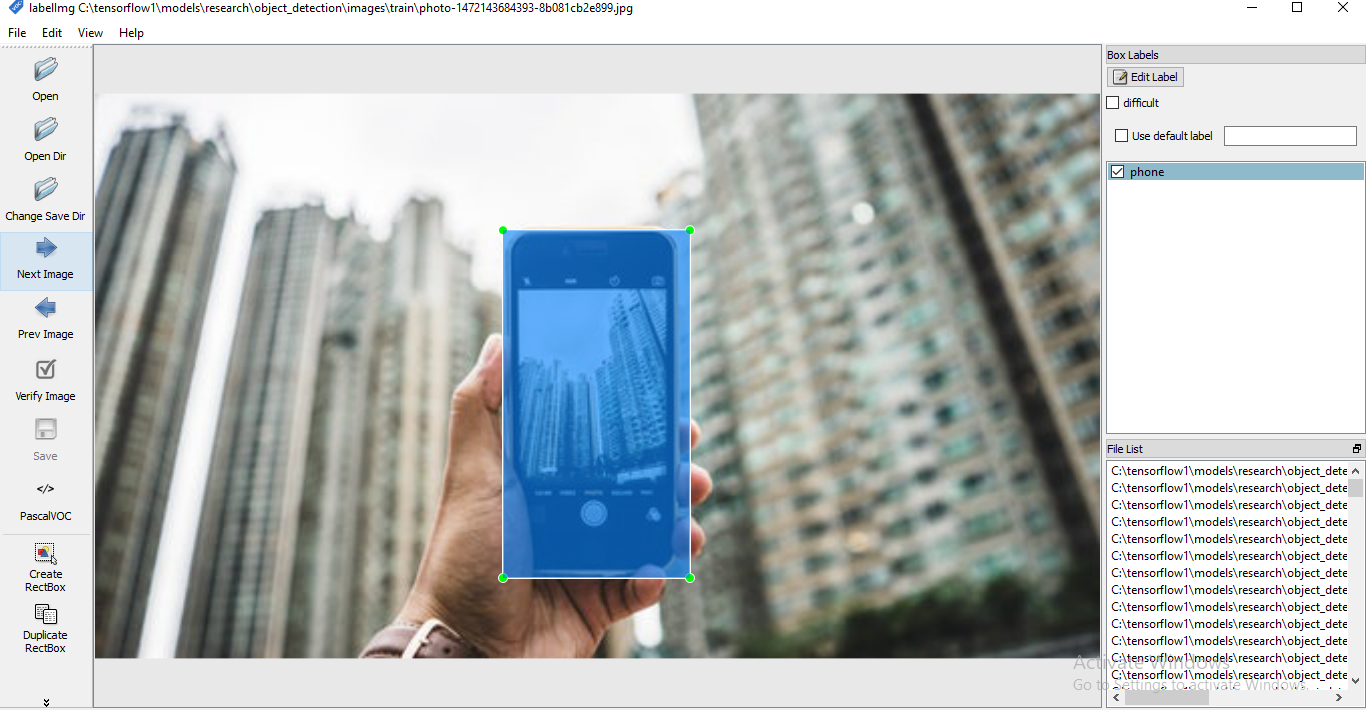
**Object Detection Training**

I have used Tensorflow to train my own object detection classifier and have programmed using the python programming language.

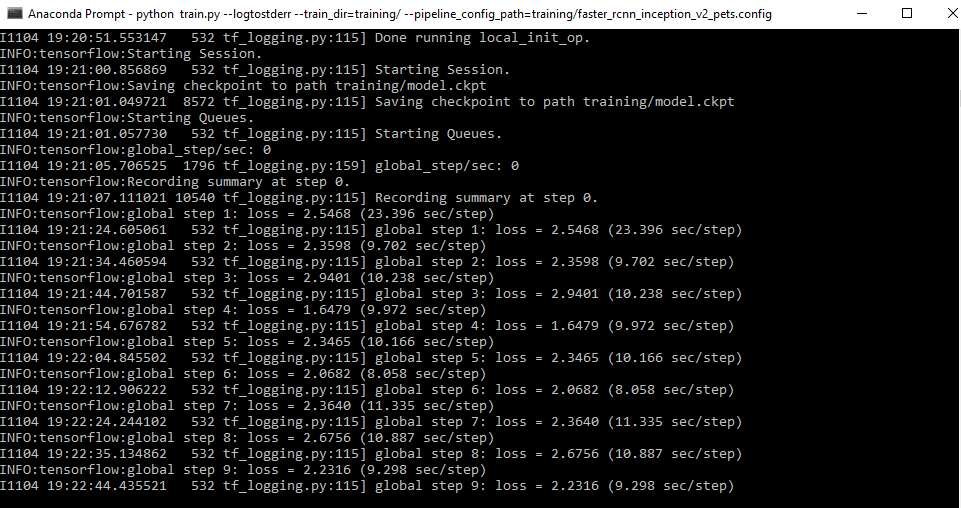
I have selected **Smart Phones** as my objects for detection. I had to create an environment of tensorflow using anaconda prompt. After creating the environment, I have installed all the necessary packages for training like numpy, opencv, matlib, pillow, tensorflow-gpu etc.

After installing the necessary packages, I collected images of smart phone with different orientations, sizes, lightings, rotation as these influence the training accuracy highly. I collected around 250 images for my training and kept around 20 images for testing my classifier. After collecting the positive images, I have labelled all the images i.e. creating bounding boxes with the item as **‘phone’**. When labelling the images, we generate xml files which contain the coordinates of the bounding boxes. After we have obtained xml files for every image, we create a single CSV file which contains the records of the all the image’s attributes.

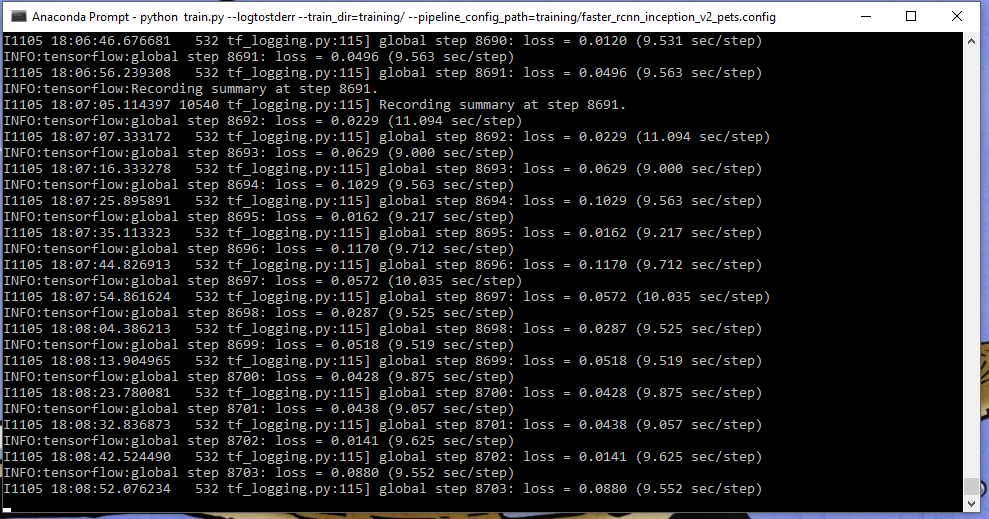


**Fig.1. Labelling Images**

Given the amount of training images being low, I have decided to take a model that gives higher accuracy (albeit more training time is required.) I have taken the **faster\_rcnn\_inception\_v2\_coco** as my configuration model which is used to train my images. Due to the low processing power of my system, the training time was around **24 hours** with each step taking around **12.3 sec**. At the starting of the training the loss was around **2.5** and as the training progressed, the loss was consistently around **0.06**. This took around 9600 steps. For every 5 minutes or 10 steps, a checkpoint was created along with a frozen inference graph to keep track of progress in case of abrupt failure or termination. After the loss was consistently around **0.6,** I decided to stop the training my classifier and test the images. I put the threshold of the accuracy greater than **0.8.** This means that my classifier will visualise the detected bounding box for an object only if the accuracy is greater than **80%.**



**Fig.2. Beginning of Training**



**Fig.3. Loss is consistently <0.06**

I have created three python programs to test my object detector:

1. **Object Detection by Image.**
2. **Object Detection by Video.**
3. **Object Detection by Webcam.**

In all 3 programs, I have used Tkinter Filedialog to let the user select the desired Image or Video to detect the objects in them.

In the first program (Object Detection by Image), the user has to select an image for testing and then that image is compared by the object detector and the detection is performed if it satisfies the requirements. The **frozen\_inference\_graph.pb** is the classifier with which the object is detected.

When I executed my program, it took frames per second was 1 which is too low to run. Unfortunately, due to the low processing power of my system, I was not able to detect the objects in the video as it requires huge processing power. However the object was detected despite of very low fps.

When the Object detection by webcam was executed, although very slow, I was able to accurately detect the trained object.