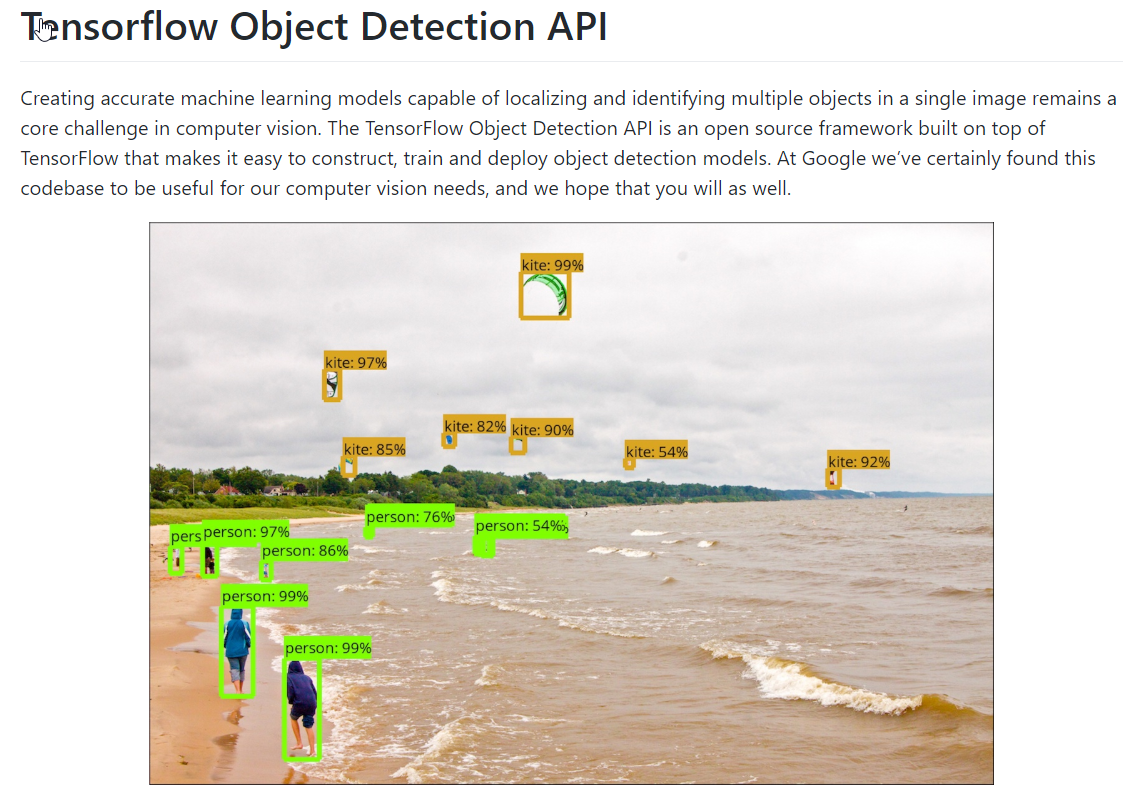
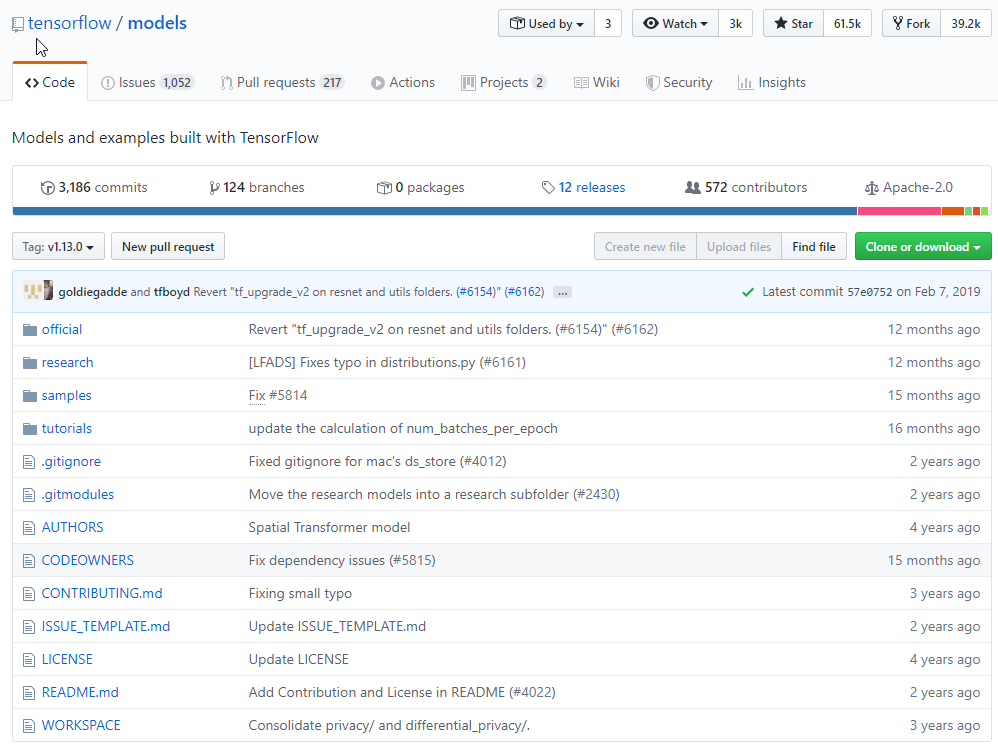
**MASK RCNN Using Tensorflow OBJECT DETECTION API**

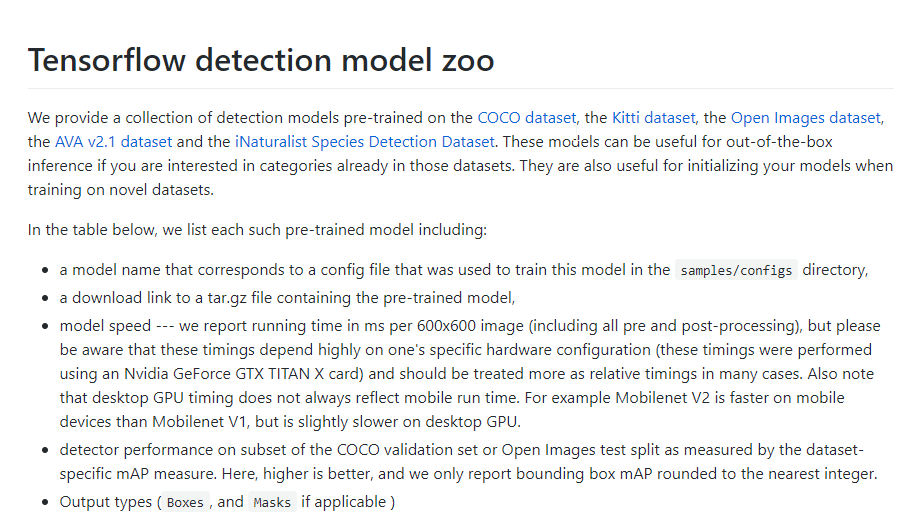
In June 2017, Google opened the TensorFlow Object Detection API. This project uses TensorFlow to implement most of the deep learning target detection frameworks, including MaskRCNN in reality.

<https://github.com/tensorflow/models/tree/master/research/object_detection>



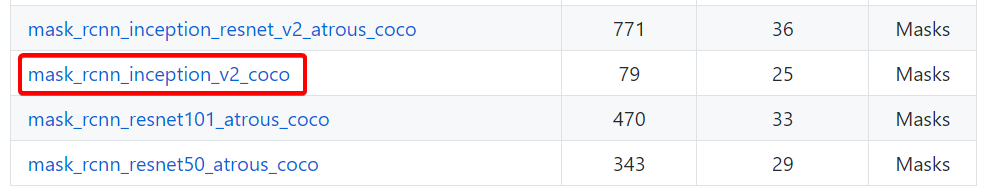
<https://github.com/tensorflow/models/tree/v1.13.0>





MASK RCNN Model link :-

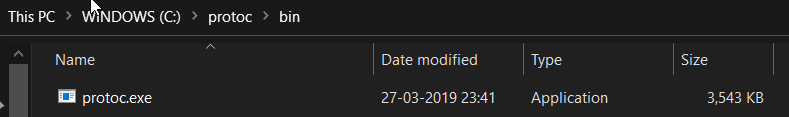
<http://download.tensorflow.org/models/object_detection/mask_rcnn_inception_v2_coco_2018_01_28.tar.gz>



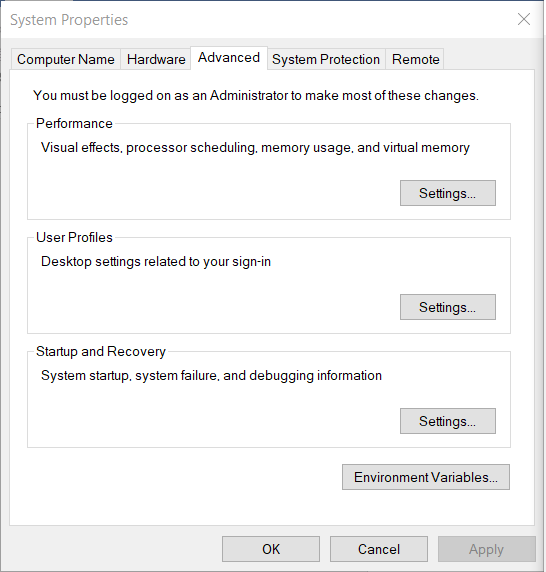
**For protobuff to py conversion**

<https://github.com/protocolbuffers/protobuf/releases/download/v3.11.0/protoc-3.11.0-win64.zip>

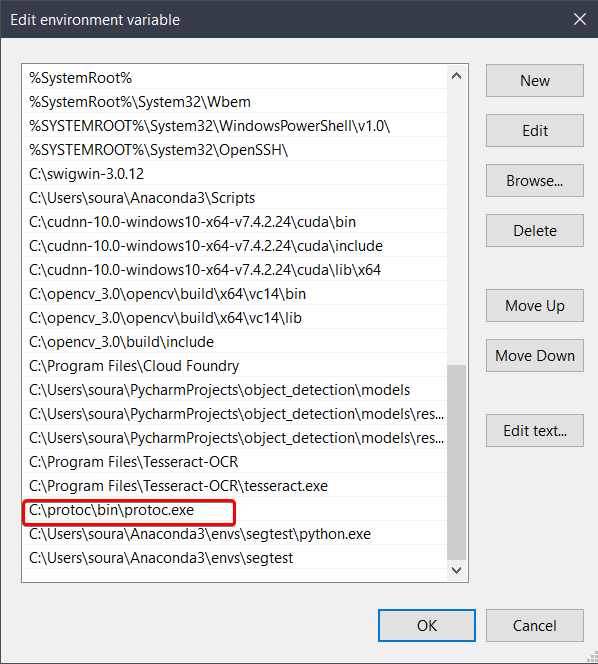
Unzip the folder and rename to protoc and move to C drive.



Add protoc.exe to system environment variables path.







Run this command from C:\tensorflow\_object\_detection\models\research

"C:/protoc/bin/protoc" object\_detection/protos/\*.proto --python\_out=.

**OR**

# linux mac

protoc object\_detection/protos/\*.proto --python\_out=.

#windows

protoc object\_detection/protos/\*.proto --python\_out=.

**Open your Anaconda Prompt**

**Creating virtual env using conda**

conda create -n your\_env\_name python=3.6

change your\_env\_name with your favourite name

Activate the

conda activate your\_env\_name

**Install the necessary packages**

pip install pillow lxml Cython contextlib2 jupyter matplotlib pandas opencv-python tensorflow==1.14.0

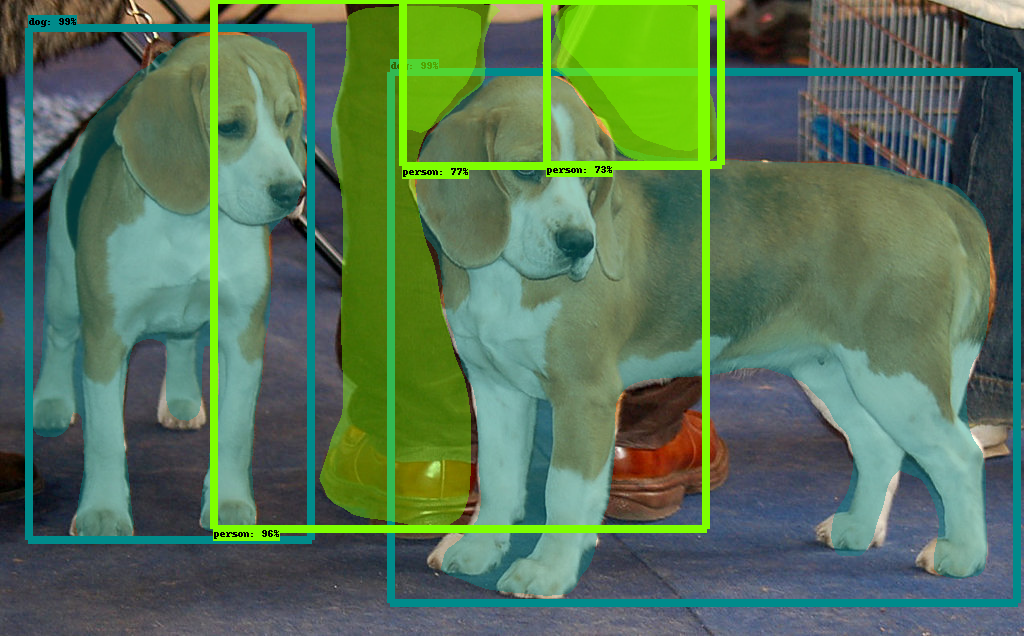
Now Open your Jupyter Notebook from the objection\_detection folder of the models repo

Path will be ---- models/research/object\_detection/

Open the **object\_detection\_tutorial.ipynb** notebook or you can run in Google Colab

**Execute all the Cells of the above notebook.**

**Final Output will look like**



**Data Annotation**

In data annotation, we will be using **Labelme Tool**

Download the Tool from the given Link:-

<https://github.com/wkentaro/labelme>

Installation of Labelme:

conda create --name=labelme python=3.6

source activate labelme

# conda install -c conda-forge pyside2

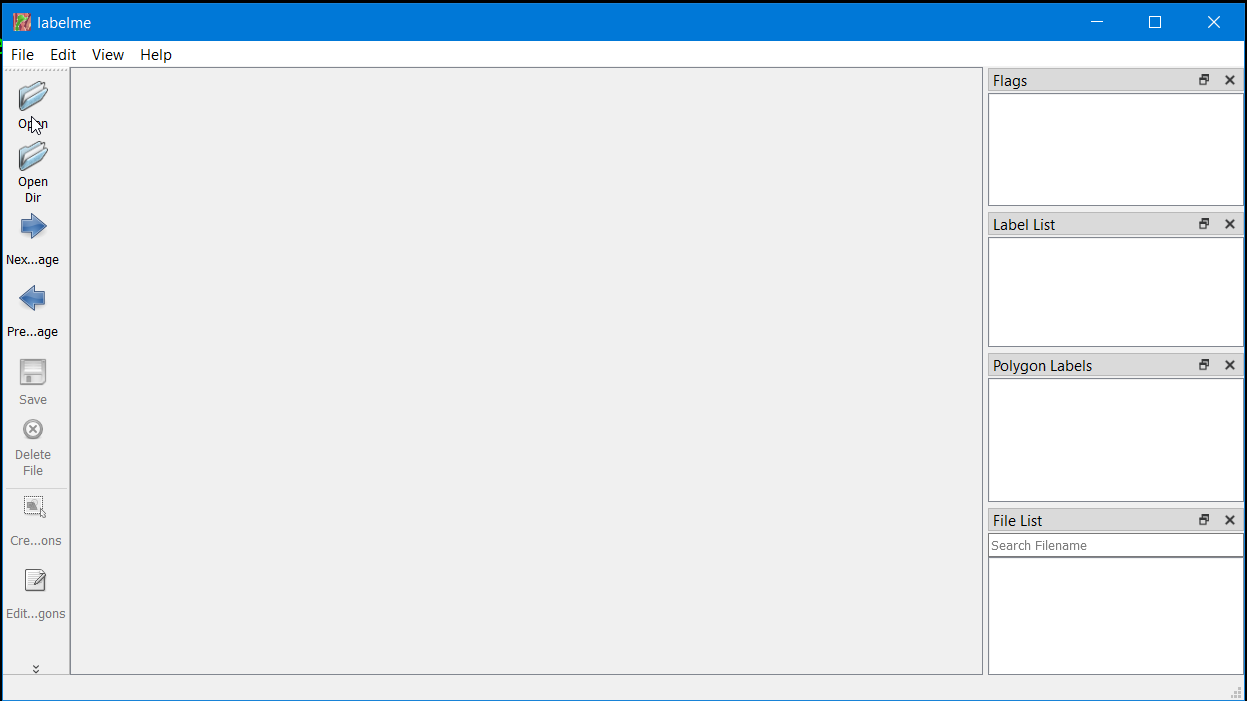
# conda install pyqt

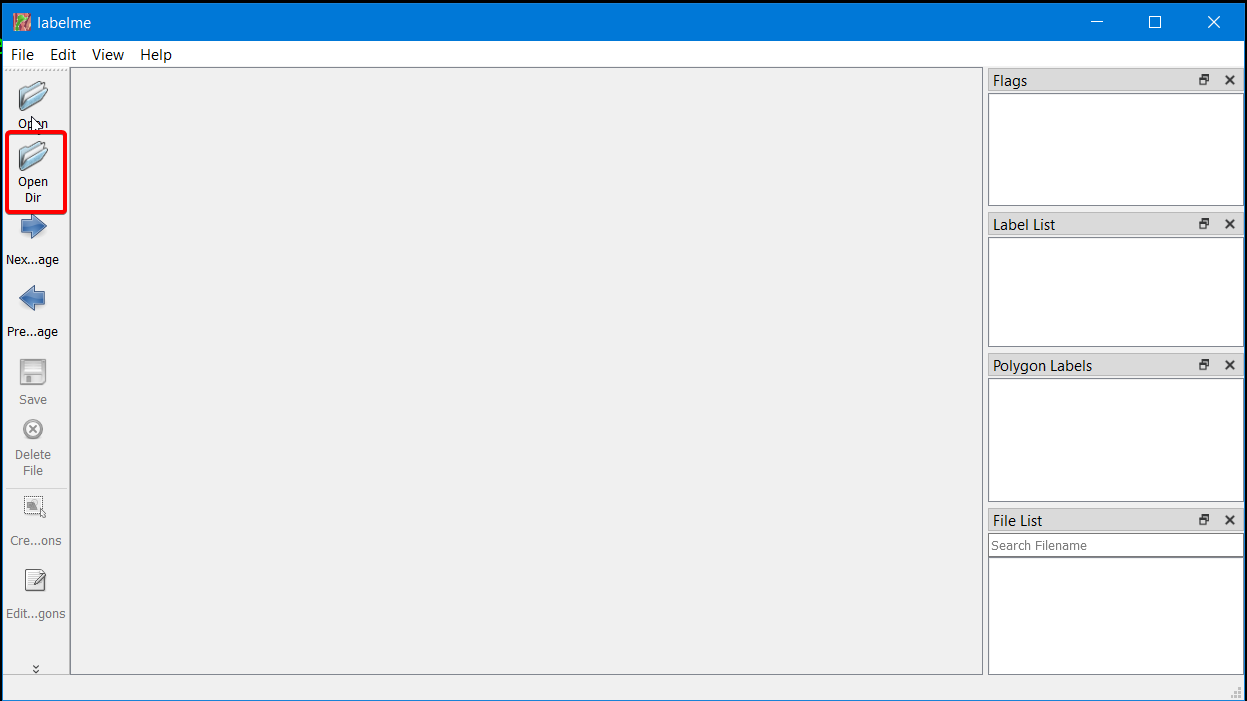
# pip install pyqt5 # pyqt5 can be installed via pip on python3

pip install labelme

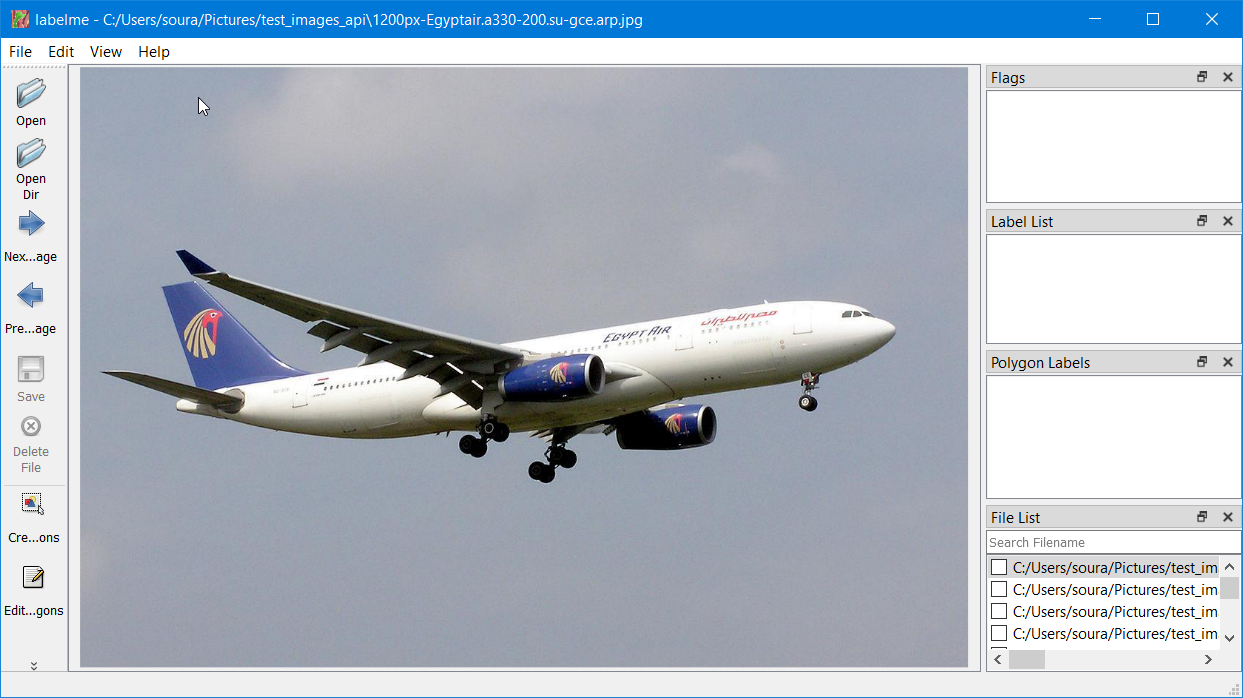
labelme

Then the interface of labelme will popup.

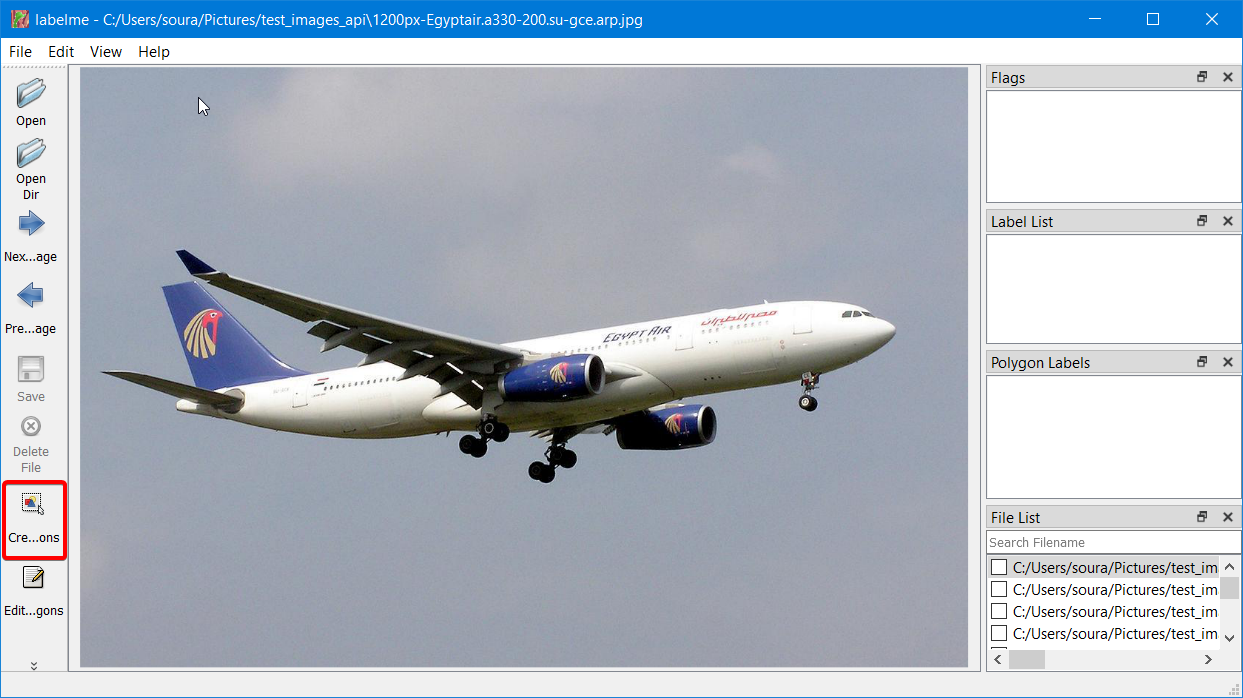


****

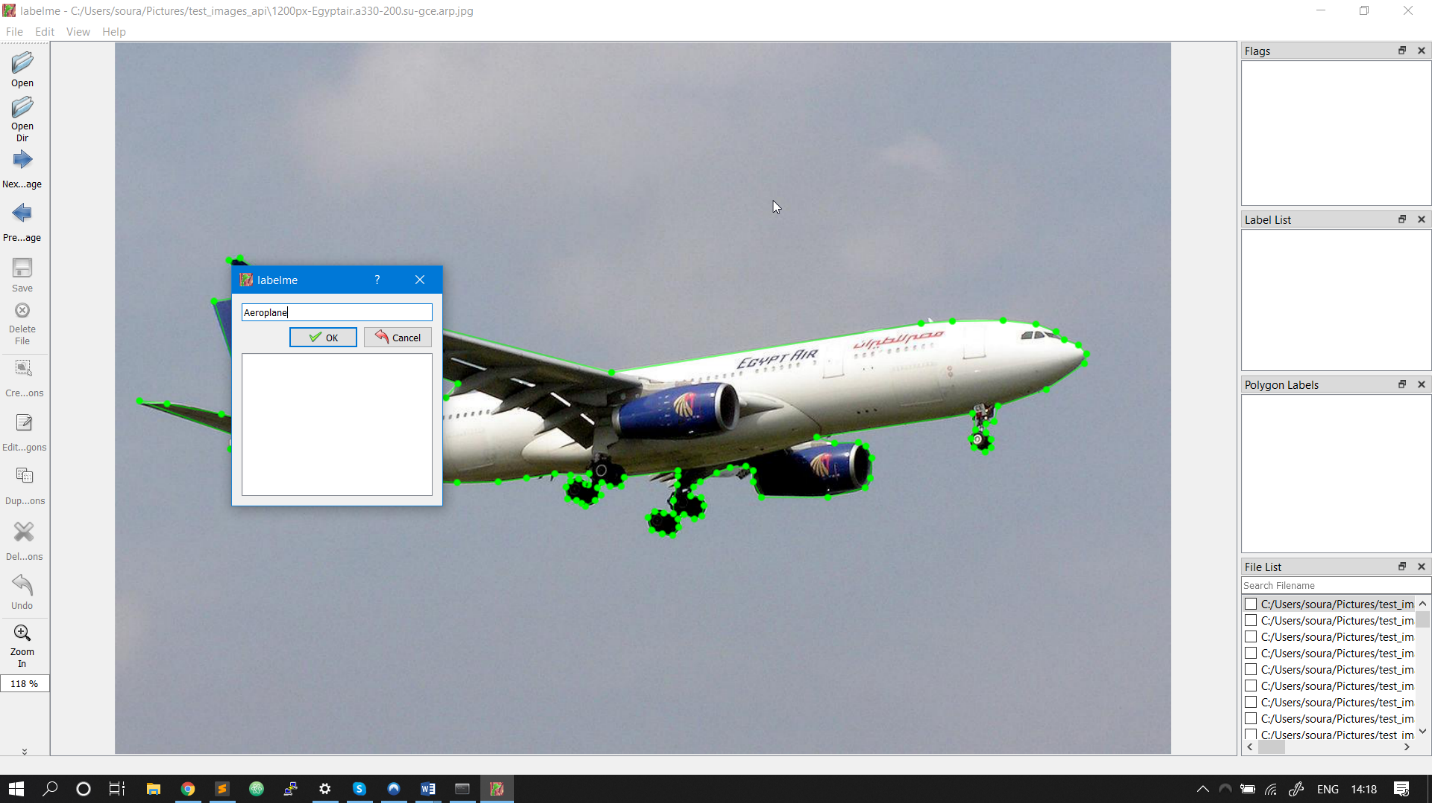
Click on **Open Dir** and select the folder where your test images or train images are. Then start labelling all images in the dataset.

****

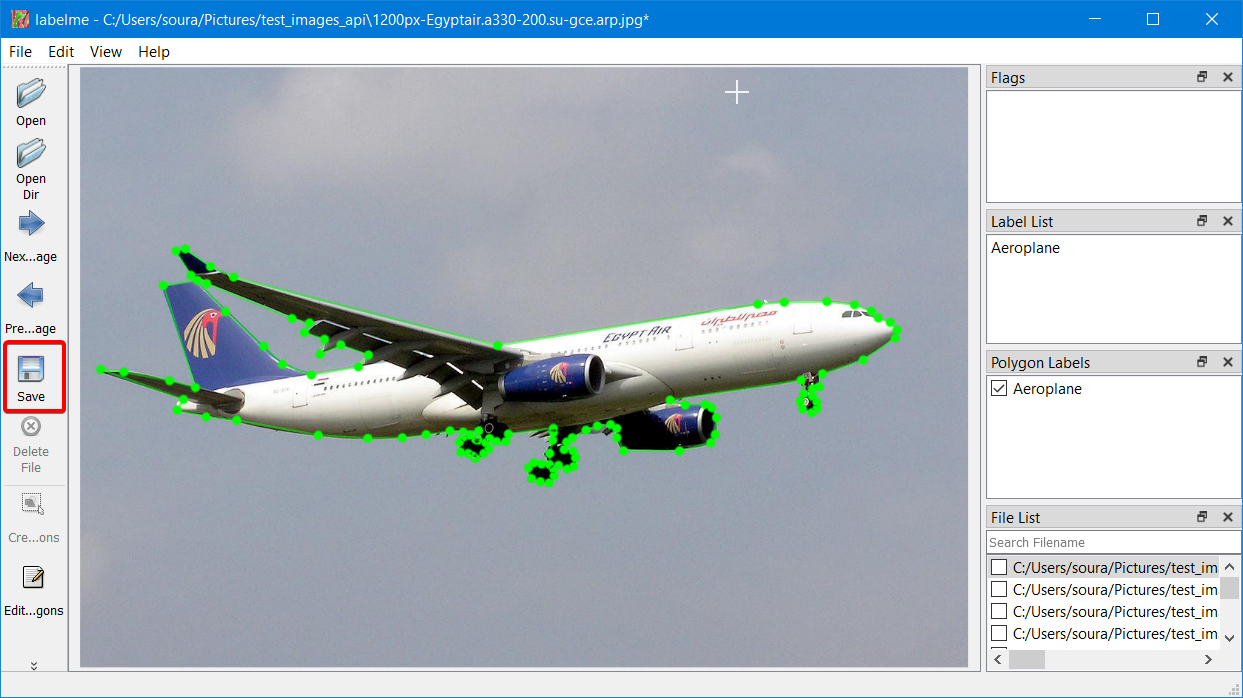
The selected is loaded in the Labelme Interface.

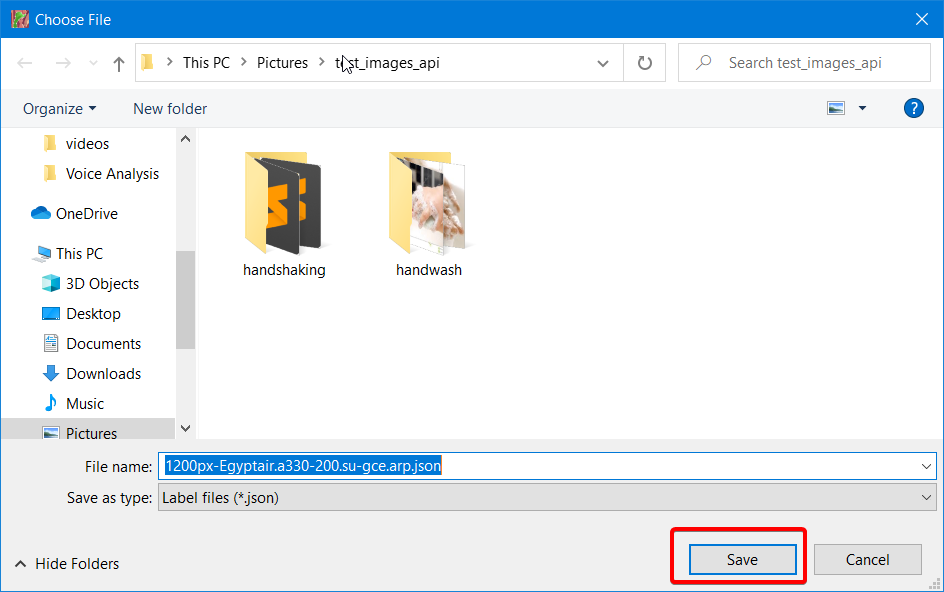
****

Click on **Start Drawing Polygons.**

****

Create the Polygon Box and write the Label name corresponding to the image

.****

Click on the Save button and a corresponding JSON file will be saved in the directory with the image.

**NEXT**

Navigate to the research folder.

Create a new folder called **images**. This is where we will be keeping our own train and validation data.

Create another new folder called **training** where we will be keeping our **model config file** and **labelmap.pbtxt.**

**Collect your dataset**

**Split the dataset into 80:20 ratio**

**Save all the images of the dataset in the images folder of research folder**

**Conversion of json to tfrecords**

python3 create\_tf\_record.py \  
 --images\_dir=your absolute path to read images and annotaion json files.  
 --label\_map\_path=your path to label\_map.pbtxt  
 --output\_path=your path to write .record

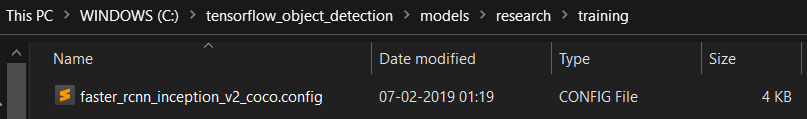
Now we can move to the training part : -

Go to C:\tensorflow\_object\_detection\models\research\object\_detection\samples\configs

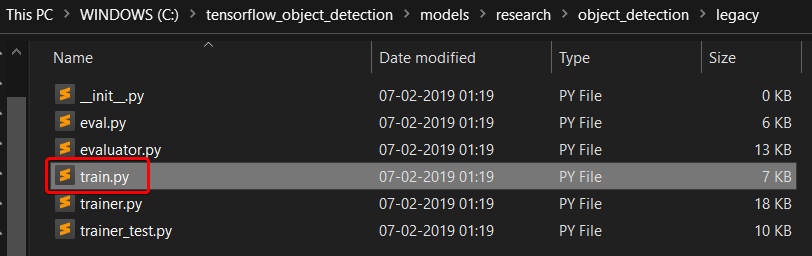


Copy mask\_rcnn\_inception\_v2\_coco.config and paste in

C:\tensorflow\_object\_detection\models\research\training



Go to C:\tensorflow\_object\_detection\models\research\object\_detection\legacy



Copy train.py and paste in C:\tensorflow\_object\_detection\models\research

Go to C:\tensorflow\_object\_detection\models\research\object\_detection

Copy export\_inference\_graph.py and paste in

C:\tensorflow\_object\_detection\models\research

**From the research folder execute**

Now we need to generate tfrecord from json files for training data and test data.

Open generate\_tfrecord.py

Replace class1 and others with your classnames.

**Create the labelmap.pbtxt**

The label map tells the trainer what each object is by defining a mapping of class names to class ID numbers. Use a text editor to create a new file and save it as labelmap.pbtxt

Replace the names of the classes with your class names.

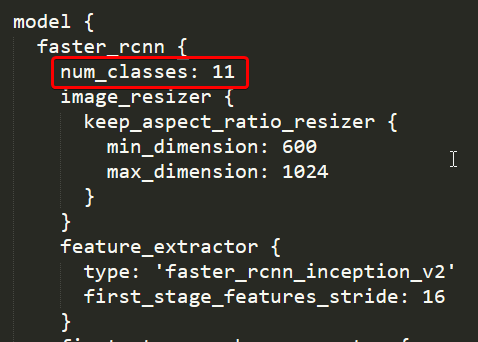
This is a sample one.



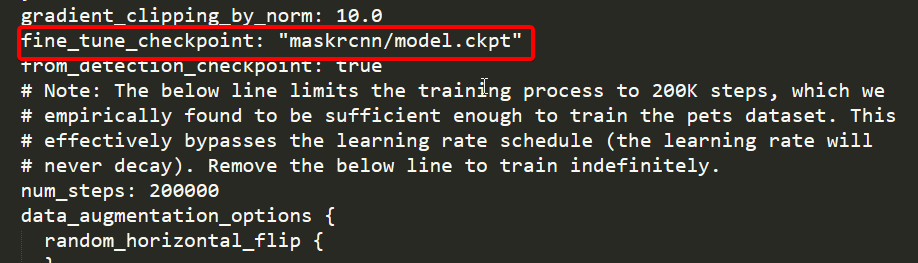
Save in the folder C:\tensorflow\_object\_detection\models\research\training

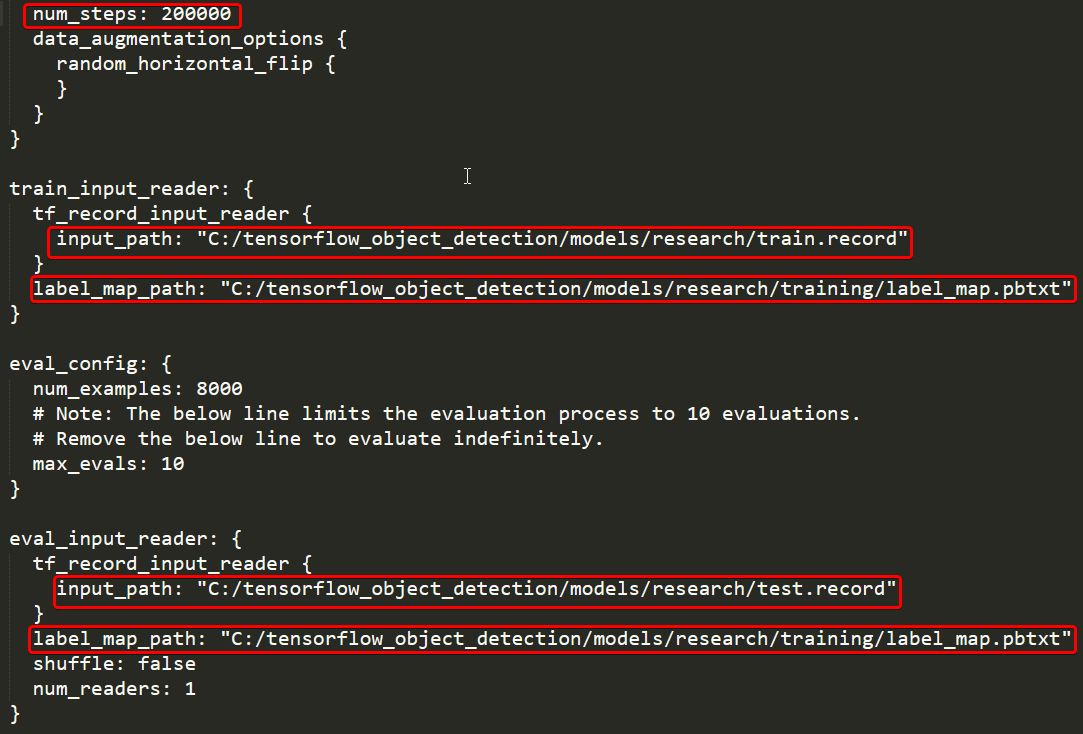
Go to C:\tensorflow\_object\_detection\models\research\training

Open **mask\_rcnn\_inception\_v2\_coco.config** with text editor, change the no of classes acc to your dataset



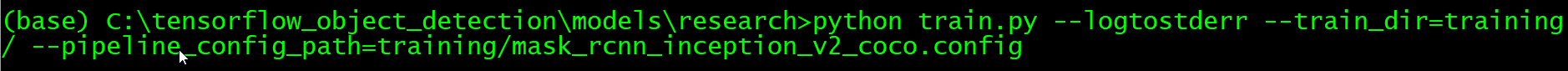
Move the “**master\_rcnn\_inception\_v2\_coco**” folder from C:\tensorflow\_object\_detection to C:\tensorflow\_object\_detection\models\research





All the red box cells contain the path that I am using. It may change based on your preferences.

Now let’s start the training process.



Training has started

You can view the progress of the training job by using TensorBoard. To do this, open a new instance of Anaconda Prompt, activate the tensorflow1 virtual environment, change to the C:\tensorflow1\models\research\object\_detection directory, and issue the following command:

(objectdetectionfashion) C:\tensorflow1\models\research\object\_detection>tensorboard --logdir=training

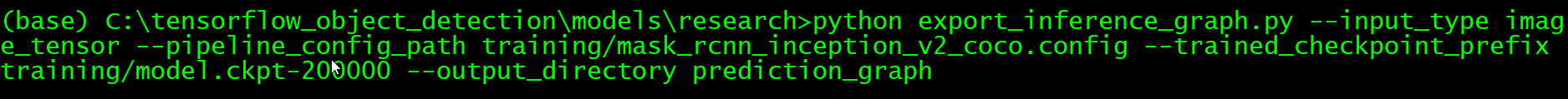
Now we can see that checkpoint files are generated on the training folder.

Now we need to convert the last ckpt numbered file to the pb model we need.

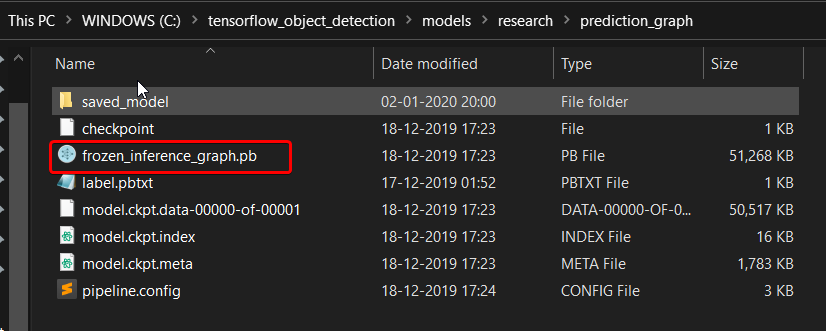
For conversion

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

In our case



Now finally prediction\_graph folder is generated.



Now you can use this pb model to predict.

**Now lets predict : -**

Lets open the **object\_detection\_tutorial.ipynb** notebook and do some changes



**Don’t Run the Download model cell of the notebook.**

And After that keep some one or two images for testing in **test\_images** folder of **object\_detection**.

Naming convention should be followed like **image1.jpg** then **image2.jpg**

**Now execute all the cells of the notebook and your test pictures will be displayed at the last cell.**