Steps for yolov4 custom training

For yolo custom object detection we need the large dataset of object that we want to detect. Here large means 7000 to 9000 for one class. It is for demo purpose so I took only 500 images, but recommended is around 8000. Once our dataset is ready now it is time to annotate it. Image annotation means we label each and every class to its corresponding class. It is just like we teach a child that it is a cat or it is dog. In similar way we teach to computer using image annotation technique. For image annotation we will use one tool that can be downloaded from github using below link. Installation steps are given in that link. Go through this link.

<https://github.com/tzutalin/labelImg>

Once image annotation is done now it is time to split dataset into train and validation part, and convert that train and validation dataset folder into .zip. Once all these things done now make two file “obj.names” and “obj.data”. In obj.name write the custom class for that you want to train yolo model. Write every class name in new line. Now open “obj.data” and enter the following details, no of classes, train and validation dataset folder path, “obj.names” file path and backup folder path.

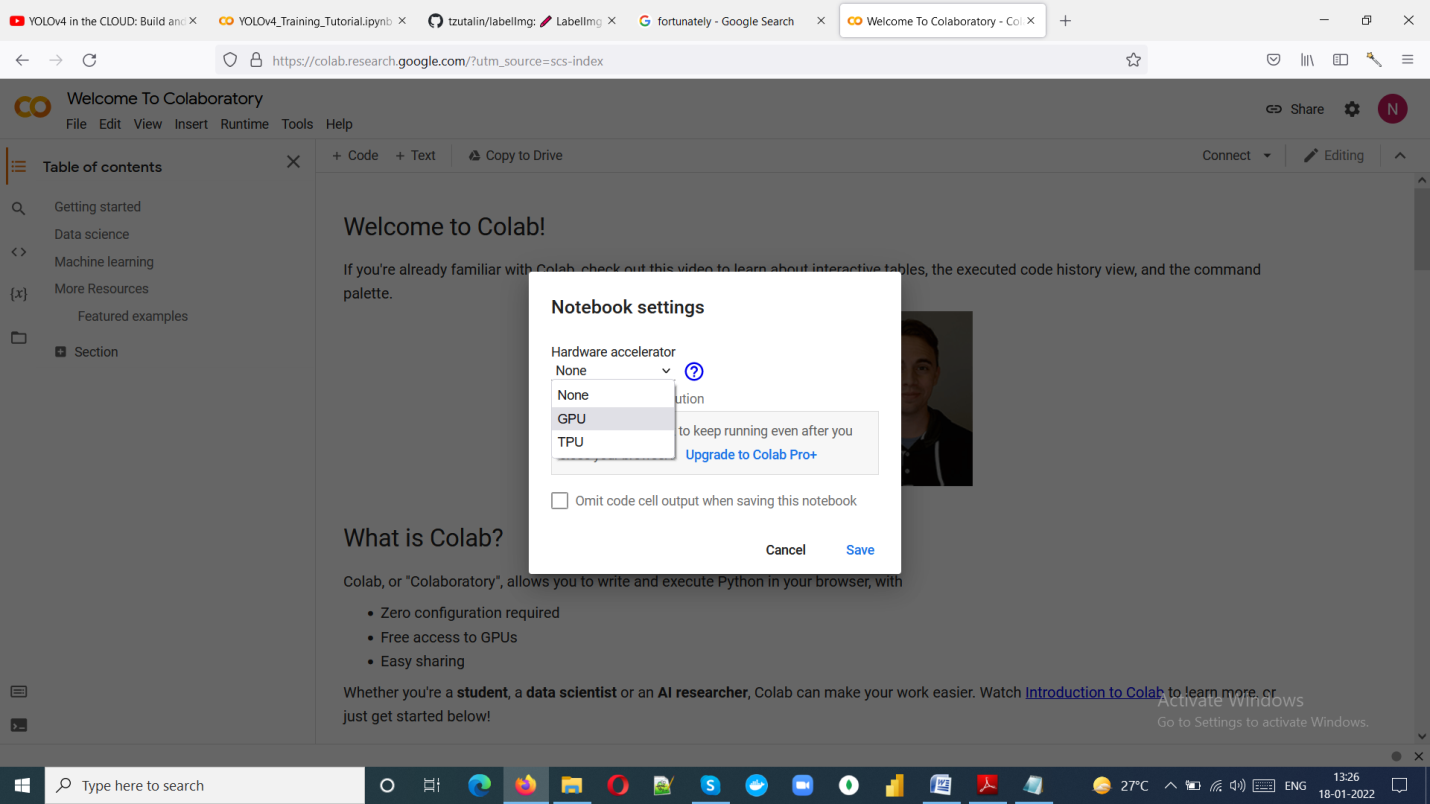
Once it is done we will prepare our environment for training our model for that we need a laptop with GPU. But unfortunately I don’t have GPU in my laptop. For that I will use google colab, it provides free GPU for limited time for free. It also has some paid plan.

Now go to google colab and choose new notebook.

Select run time type as GPU as shown below figure.

Go to run time and open change run time type

Select hardware accelerator as GPU



Now it is time to run different command to train our model

Note: Run each python command in new cell

Now connect(mount) google colab with google drive for that run following command

from google.colab import drive

drive.mount('/content/gdrive')

Change working directory by following command

import os

os.chdir("/content/gdrive/MyDrive")

print(os.getcwd())

Now clone darknet repo by following command

# clone darknet repo

!git clone https://github.com/AlexeyAB/darknet

Now time to make some necessary changes for enabling opencv and GPU

# change makefile to have GPU and OPENCV enabled

%cd darknet

!sed -i 's/OPENCV=0/OPENCV=1/' Makefile

!sed -i 's/GPU=0/GPU=1/' Makefile

!sed -i 's/CUDNN=0/CUDNN=1/' Makefile

!sed -i 's/CUDNN\_HALF=0/CUDNN\_HALF=1/' Makefile

Now Verify the CUDA by following command

# verify CUDA

!/usr/local/cuda/bin/nvcc --version

Now it is time to run make command

# make darknet (builds darknet so that you can then use the darknet executable file to run or train object detectors)

!make

After that run this command

# this creates a symbolic link so that now the path /content/gdrive/My\ Drive/ is equal to /mydrive

!ln -s /content/gdrive/My\ Drive/ /mydrive

!ls /mydrive

Create new folder in mydrive names as yolov4 and upload following file in that folder

generate\_train.py, generate\_test.py, obj.names, obj.data, test.zip i.e. validation dataset, obj.zip i.e. train dataset, yolov4-obj.cfg and create one folder backup

All above we can find in link given below

<https://github.com/theAIGuysCode/YOLOv4-Cloud-Tutorial/tree/master/yolov4>

After that run following command

!cp /mydrive/yolov4/obj.zip ../

!cp /mydrive/yolov4/test.zip ../

Now unzip obj.zip and test.zip by following command

# unzip the datasets and their contents so that they are now in /darknet/data/ folder

!unzip ../obj.zip -d data/

!unzip ../test.zip -d data/

Now it is time to make some changes in configuration file for that run following command and open file text editor

# download cfg to google drive and change its name

!cp cfg/yolov4-custom.cfg /mydrive/yolov4/yolov4-obj.cfg

I recommend having batch = 64 and subdivision=16 for ultimate result.

Change the max batches to (no of classes \* 2000) but it should not be less than 6000.

Now change the steps to 0.8\*max batches and 0.9\*max batches

Note: step will have two value as stated above

Now search of classes and change it your no of classes. In our case 2)

Change the filter few line above to classes as bellow

Filter value = (no of classes +5)\*3. In our case it is 21. i.e. (2+5)\*3 is 21

Now upload this updated file to cfg folder inside darknet directory by following command

# upload the custom .cfg back to cloud VM from Google Drive

!cp /mydrive/yolov4/yolov4-obj.cfg ./cfg

To generate train.txt and test.txt run following commands

# upload the generate\_train.py and generate\_test.py script to cloud VM from Google Drive

!cp /mydrive/yolov4/generate\_train.py ./

!cp /mydrive/yolov4/generate\_test.py ./

!python generate\_train.py

!python generate\_test.py

Now verify the train.txt and test.txt files by running following command

# verify that the newly generated train.txt and test.txt can be seen in our darknet/data folder

!ls data/

Here we are using transfer learning technique for training our model. In transfer learning we use pretrained weight and change the output layer of that pretrained model’s weight file as per our requirement.

From the link given below download the pretrained model weight file.

!wget https://github.com/AlexeyAB/darknet/releases/download/darknet\_yolo\_v3\_optimal/yolov4.conv.137

Training will take few hours if you sit idle for 30 min then google colab’s VM will be turned off, to avoid this we will a script in javascript for that go to page inspect and then console and paste following code.

function ClickConnect(){

console.log("Working");

document

.querySelector('#top-toolbar > colab-connect-button')

.shadowRoot.querySelector('#connect')

.click()

}

setInterval(ClickConnect,60000)

Now from below command we can start training of custom yolo model

# train your custom detector! (uncomment %%capture below if you run into memory issues or your Colab is crashing)

# %%capture

!./darknet detector train data/obj.data cfg/yolov4-obj.cfg yolov4.conv.137 -dont\_show -map

If the notebook is crashed then one can restart the of model by following command

# kick off training from where it last saved

!./darknet detector train data/obj.data cfg/yolov4-obj.cfg /mydrive/yolov4/backup/yolov4-obj\_last.weights -dont\_show

Calculate accuracy of model by following command (in this case mAP score)

!./darknet detector map data/obj.data cfg/yolov4-obj.cfg /mydrive/yolov4/backup/yolov4-obj\_6000.weights

Once model got trained completely download this model use this model on local machine for testing. For that python script will be required, you can get sample python script from link given below

<https://github.com/nabeel-313/yolov4/blob/main/image_detection.py>

Resource that helped to write this article

<https://www.youtube.com/watch?v=mmj3nxGT2YQ&t=1976s>

<https://colab.research.google.com/drive/1_GdoqCJWXsChrOiY8sZMr_zbr_fH-0Fg?usp=sharing#scrollTo=2UoSBmTFlNep>