How to train your custom dataset using YOLO-V4 by:
By Subroto Singha

Step#1: clone the Alexey github repo
https://github.com/AlexeyAB/darknet.git

Step#2: Change `Makefile` flag GPU to 1 to use the GPU and set cuda=1

Step#3: Then `compile` the darknet using make command in Linux and cmake in Windows

Step#4: Follow the steps below for **customizing the architecture of the neural network:**

- Copy yolov4-custom.cfg to yolo-obj.cfg (To avoid changing the existing cfg conflict)
- Change line batch to `batch=64`
- Change line subdivisions to `subdivisions=16`
- Change line max_batches to (`classes*2000` but not less than number of training images, but not less than number of training images and not less than `6000`), i.e. `max_batches=6000` if you train for 3 classes
- Change line steps to 80% and 90% of max_batches, i.e. `steps=4800,5400`
- Set network size `width=416 height=416` or any value multiple of 32
- Change line `classes=80` to your number of objects in each of 3 `[yolo]`-layers Line numbers: **1146,1058,970**
- Change [`filters=255`] to filters=(classes + 5)x3 in the 3 `[convolutional]` before each `[yolo]` layer, keep in mind that it only has to be the last `[convolutional]` before each of the `[yolo]` layers Line numbers:**1139,1051,963**

- So if `classes=1` then should be `filters=18`. If `classes=2` then write `filters=21`.
- Turn off mosaic flag as we are not using OpenCV mosaic=0

Step#5: Create file `obj.names` in the directory `darknet\data\`, with objects names - each in new line
> ***Linux command:*** touch obj.names

Step#6: Create file `obj.data` in the directory `darknet\data\`, containing (where classes = number of objects):

```
classes= 2
train  = data/train.txt
valid  = data/test.txt
names = data/obj.names
backup = backup/
```

- Linux command: touch obj.data

Step#7: Create obj folder in the directory `darknet\data\`

> **Linux Command:** mkdir obj

Step#8: Create train and test folders in the directory `darknet\data\obj\`

> ***Linus Command:* mkdir train test**

Step#9: Put **image-files (.jpg) and .txt** files in the directory `darknet\data\obj\train\`

Step#10: Create file **train.txt** and **test.txt** in directory `darknet\data\`, for example:

`data/obj/img1.jpg`

`data/obj/img2.jpg`

`data/obj/img3.jpg`

Linux Command: To add filenames in front of the path for jpg files and save it to train.txt file:
ls *.jpg | xargs -i echo data/obj/train/{ } > train.txt

Step#11: Download pre-trained weights for the convolutional layers and put to the directory `\darknet\`

* for `yolov4.cfg, yolov4-custom.cfg` (162 MB): yolov4.conv.137 (Google drive mirror yolov4.conv.137)

Step#12: Start training by using the command line:

To train on Linux use command: `./darknet detector train data/obj.data cfg/yolo-obj.cfg yolov4.conv.137`

Step#13: After **training is complete** - check trained weights `yolo-obj_final.weights` from path `\darknet\backup\`

Step#14: Testing using below command:

`./darknet detector test data/obj.data cfg/yolov4- custom.cfg backup/yolov4-custom_last.weights`

Enter path:

data/obj/test/green_turtle_50.JPG
data/obj/test/green_turtle_51.JPG
……

//Download Predictions: /home/subroto_singha4/alexeynew1/darknet/predictions.jpg

- To check accuracy mAP@IoU=50: darknet.exe detector map data/obj.data yolo-obj.cfg backup\yolo-obj_7000.weights

- To check accuracy mAP@IoU=75: darknet.exe detector map data/obj.data yolo-obj.cfg backup\yolo-obj_7000.weights -iou_thresh 0.75