Step by Step Guide to Training YOLO V2 On Pascal VOC, and Custom Dataset on GPU, Ubuntu 16.04

This is a step by step guide which takes you from installing the Requirements for setting up the environment for YOLO to training YOLO on custom dataset. The instructions can be split up into three different sections. In the first section i will walk you through installing the required softwares which includes installing the NVIDIA drivers,CUDA,OPENCV and other supporting softwares.

Then in section two, we will go through installation of the darknet and YOLO v2. Then in the section 3, i will guide you through how to tune the Custom dataset at hand and in section 4, Will guide you through fine tuning the dataset for training on YOLO V2. This includes detailed explanation on annotating the images in your dataset using the Microsoft annotation tool. In the Error troubleshooting section , we will go through troubleshooting some of the most common issues encountered while installing and running the YOLO network.

SECTION 1:Requirements

Operating system: ubuntu 16.04

Software requirements: The following are the requirements before starting to work on the darknet platform.

1. NVIDIA binary driver
2. CUDA 8.0
3. OPENCV 3.0
4. Python 2.7

SECTION 2:YOLO v2 training on Pascal VOC

The main installation page for yolo in itself is a good reference for installing the YOLO V2 or YOLO V2 on the system. Please visit the link for installing darkent, train YOLO and other useful resources to play with different YOLO v2 Models.

https://pjreddie.com/darknet/yolo/

But to go through the installation without going to a different webpage. You can follow the instructions below to start training the YOLO v2 on Pascal VOC.

Note:

1. **If you want to go through the training process of yolo please follow the steps.**
2. **If you are just looking to play with YOLOv2 without training. Jump to running YOLOv2 on pretrained weights for detection.**
   Jump to: Running YOLOv2 using the pretrained detection weights:

Installing Darknet Platform for YOLO 9000:

For installing darknet on your system go to the directory in the terminal in which you want to install darknet and run the following commands.

```
git clone https://github.com/pjreddie/darknet
cd darknet
make
```

Downloading the Pascal VOC:

Once the darknet is installed now to download the pascal VOC create a directory where you want to store all the pascal VOC data. Then run the following,

```
wget https://pjreddie.com/media/files/VOCtrainval_11-May-2012.tar
wget https://pjreddie.com/media/files/VOCtrainval_06-Nov-2007.tar
wget https://pjreddie.com/media/files/VOCtest_06-Nov-2007.tar
tar xf VOCtrainval_11-May-2012.tar
tar xf VOCtrainval_06-Nov-2007.tar
tar xf VOCtest_06-Nov-2007.tar
```

Once you extract there will be VOCdevkit/ folder in the folder you created that contains all the VOC 2012 and 2007 data.

Generating labels for the Pascal data:

For each image in the pascal VOC we need the annotation to be in the following format

```
<object-class> <x> <y> <width> <height>
```

So we use the labels.py tool comes with YOLOv2 to generate the labels for the images. Go to the VOCdevkit directory and run the following,

```
wget https://pjreddie.com/media/files/voc_label.py
python voc_label.py
```

After some time this generates the labels for all the images in directory inside the folder. You should be able to see something like this in the directory,

```
ls
2007_test.txt   VOCdevkit
2007_train.txt  voc_label.py
2007_val.txt    VOCtest_06-Nov-2007.tar
2012_train.txt  VOCtrainval_06-Nov-2007.tar
2012_val.txt    VOCtrainval_11-May-2012.tar
```

After the above operation we have the set of images to train from each Year of Pascal VOC i.e 2007_train.txt and 2012_train.txt. Now we will obtain a file with list of all the images that needs to be trained on YOLO v2 by running this in the terminal,

cat 2007_train.txt 2007_val.txt 2012_*.txt > train.txt

Modifying config file and training YOLO:

Go to the darknet directory where the darknet is installed and open the voc.data file in the editor of your choice and change the directory information by changing the location to look for pascal VOC datasets

```
1 classes= 20
2 train  = <path-to-voc>/train.txt
3 valid  = <path-to-voc>2007_test.txt
4 names = data/voc.names
5 backup = backup
```

The path-to-voc should be replaced with the directory information

To download the convolutional weights which are obtained by pretraining on imagenet download the weights by running the following in the darknet directory.

wget https://pjreddie.com/media/files/darknet19_448.conv.23

Note: This is not an optional step. The weights downloaded here are for the classification layer not for the detection layer. I suggest for downloading the weights for the classification layer. This is given the reason that training the classification net actually takes long time.

Before starting to train create a folder with name as in the darknet directory “backup”. The YOLOv2 while getting trained will store a backup of weights in this directory

Now to start training the detector network run the following from the darknet directory,

./darknet detector train cfg/voc.data cfg/yolo-voc.cfg darknet19_448.conv.23

The training will be taking few hours of time based on your GPU.

Once you are done with training the YOLO v2 you can test it using,

./darknet test cfg/voc.data cfg/yolo-voc.cfg backup/<final-weights> <test_image/test_video>

Replace the final-weights with the name of the final weights file obtained after training the YOLOv2. Which is located in the backup folder you created earlier. You can just replace the test_image with the location of image you want to test or with the video file you want to test on.

Optional method without training the YOLOv2 at all: (skip this if you have trained weights in previous steps)
Running YOLOv2 using the pretrained detection weights:

Download the pretrained weights from the following link and move the file to the darknet directory and run this,

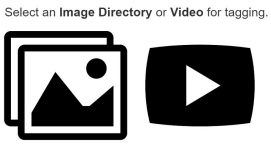
./darknet test cfg/voc.data cfg/yolo-voc.cfg <final-weights> <test_image/test_video>

Replace the final weights with the file you just downloaded and also replace the test_image with the image or video you want to test on.

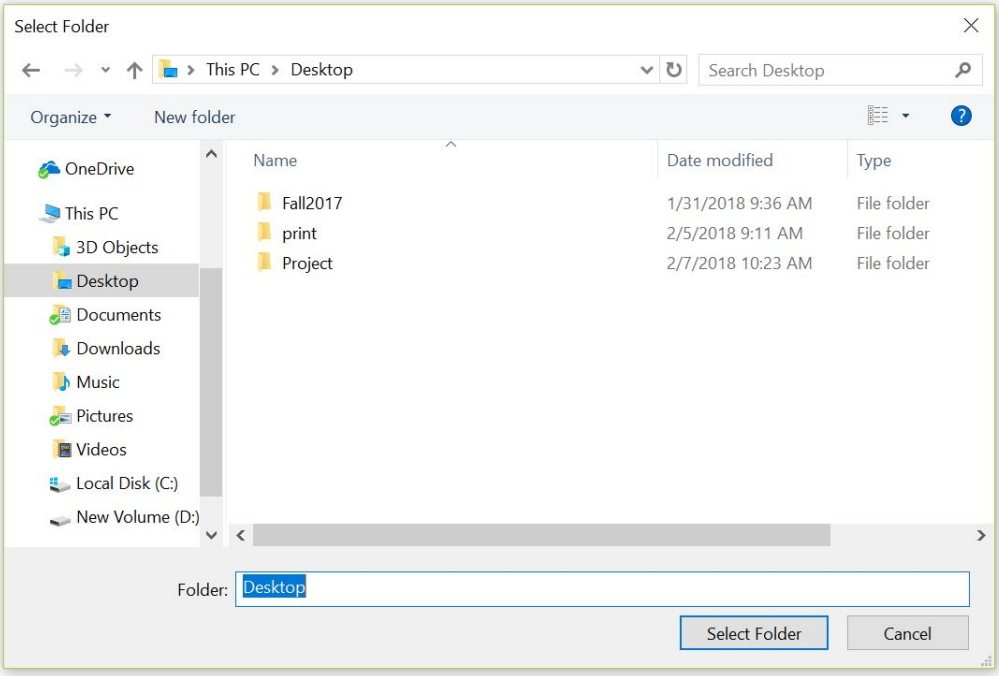
SECTION 3: Creating Custom Dataset for training on YOLO v2
Creating the Custom dataset for training YOLO9000:
Note: The reason for choosing the microsoft VOTT for annotation is because it is able to output the annotations in various widely used formats. Which can directly fit into the Training models without much preprocessing.
Annotating Images:
Windows annotating: Here the tagging is carried out on windows platform using Microsoft VOTT tool. If you are comfortable with tagging on windows please download the software from the following link.
Linux annotating tool:https://github.com/AlexeyAB/Yolo_mark
I haven't tried annotating on this tool. But if you want to carry annotations in the same file i would suggest giving it a try**.**
Let's continue with the VOTT on windows:

1. **Download the Microsoft Visual object tagging tool from this link.**

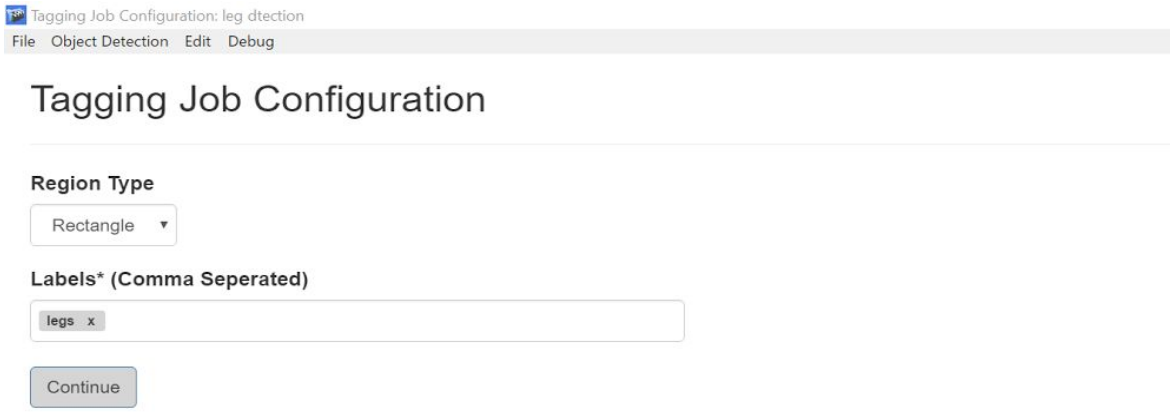
https://github.com/Microsoft/VoTT/releases/download/v1.0.8/VOTT-win32-x64.zip
Now run the exe file in the extracted zip folder. The following window should pop up



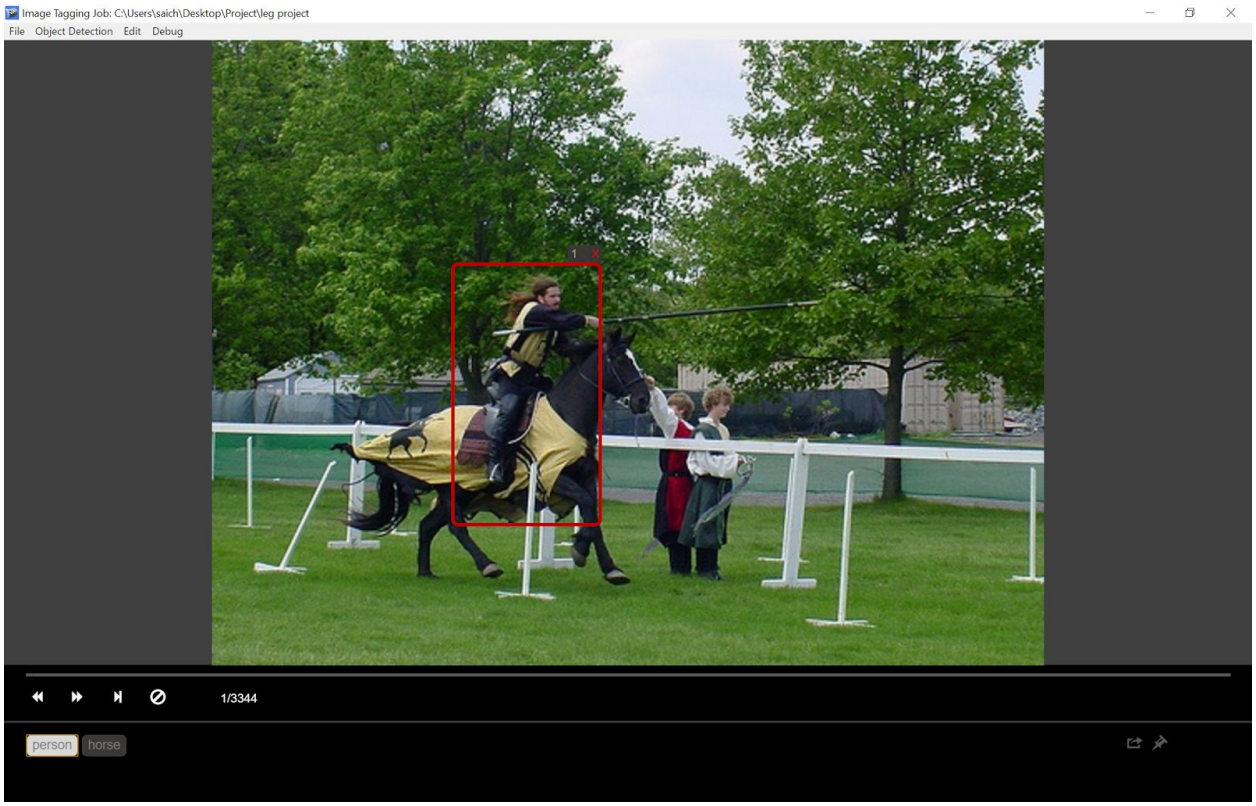
Now The above window pops up and asks to select the image or the video file directory that contains the dataset that needs annotation. Click on the respective icon.



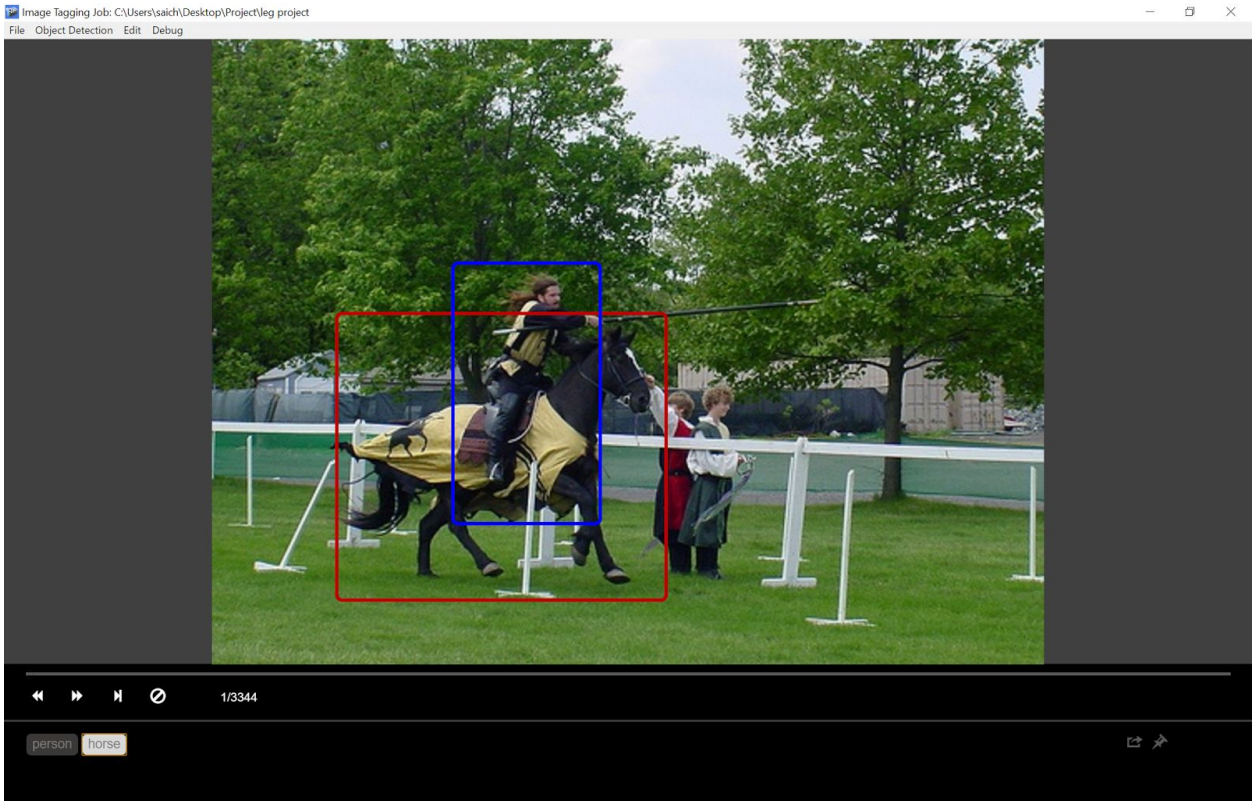
Choose the directory where the images are located. Once you are done with that the following window should pop up asking you for the details of the annotations.



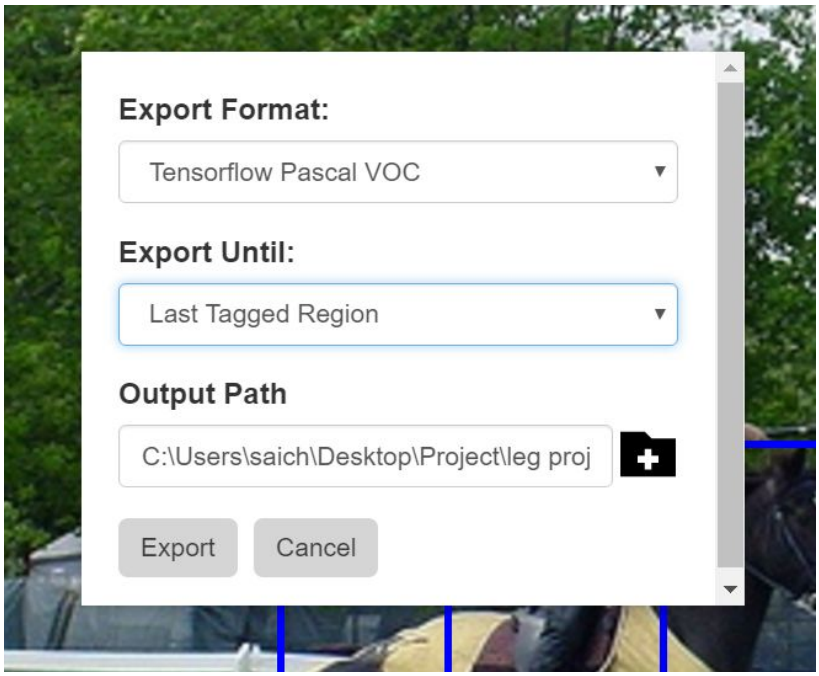
Now choose the shape of the annotation by using the drop down menu. Then in the label section create labels for all the set of classes you want to annotate in your dataset. For example if you want two classes in your dataset persons and horses
Type in Persons and hit enter then,
horses and hit enter
Once you are done with specifying the parameters for annotations hit Continue



Select the rectangle region you want to annotate and select the label for the object. Then draw the bounding box for the second object and and label it as shown below.



Continue this process for all the classes in the image for all the images in the dataset.
Once you finished annotating all the images
In the Toolbar:
Object detection>Export Tags



Once you are here you have four options for exporting the annotations. As i have trained the YOLO v2 on Pascal VOC i choose the Tensorflow Pascal VOC format for the export annotation type.

I suggest to research and choose the output annotation format carefully based on your deep neural network requirements.

Section 4: Tuning the Custom dataset and make it ready for training on YOLO v2:

Now the annotation file in the Tensorflow pascal VOC format should have the following folders namely,

Annotations
ImageSets
JPEGImages

Move the above folders into a new folder and name the new folder as **VOC2007** . Then move this to a new folder named **VOCdevkit**

The important thing to note is that in the ImageSets directory we will have only the train and validation info

VOCdevkit/VOC2007/ImageSets/Main looks something like this (if you have just one class for your dataset, in my class “legs” is the only class)

test.txt
val.txt
legs_train.txt

legs_val.txt

The above should be changed to make it look something like
test.txt
val.txt
trainval.txt
test.txt
legs_train.txt
legs_val.txt
legs_trainval.txt
legs_test.txt
So overall we should be creating four new files (if you have just one class) trainval.txt,test.txt,legs_trainval.txt,legs_test.txt. Let us now see on what will be the contents of this new files.

test.txt

We get the contents for test.txt from the train.txt. Let us suppose we will be using 80:20 ratio for training and testing the YOLO v2. Then we will **MOVE** certain percent of the contents of train.txt to test.txt such that ratio of number of test instances to train val instances is 0.20.

Trainval.txt

Now **COPY** the remaining of data from test.txt and val.txt to the trainval.txt

The procedure for obtaining the contents of files below is just the same as above except her instead of using train.txt and val.txt we use the legs_train.txt, legs_val.txt to create the new files

legs_trainval.txt

legs_test.txt

I have actually a thought of automating this step by using a algorithm. Will update this once i have one such algorithm

After repeating the above steps for all the classes in the dataset follow these steps.
The voc_label.py file downloaded needs editing before running the following step.

Modify,

sets=[('2012', 'train'), ('2012', 'val'), ('2007', 'train'), ('2007', 'val'), ('2007', 'test')]

To

sets=[('2007', 'train'), ('2007', 'val'), ('2007', 'test')]

Then,

classes = ["aeroplane", "bicycle", "bird", "boat", "bottle", "bus", "car", "cat", "chair", "cow", "diningtable", "dog", "horse", "motorbike", "person", "pottedplant", "sheep", "sofa", "train", "tvmonitor"]

To

classes = ["legs","another_class",.....]

Then comment out
Line 35: **difficult = obj.find('difficult').text**

Then change, line 37,38,
if cls not in classes or int(difficult) == 1:
Continue
to,
if cls not in classes: # or int(difficult) == 1:
continue

Thats it with editing the file if you have .jpg images
If that is not the case change the .jpg in the line 53 to required format
list_file.write('%s/VOCdevkit/VOC%s/JPEGImages/%s.jpg\n'%(wd, year, image_id))

You will be moving the voc_label.py into the annotation output directory of Microsoft VOTT and then run the steps for Generating labels

1. **Generating labels for the Pascal data:**

Once done with generating labels do the following, do all the steps exactly the same way explained except the training step.
We have to modify the yolo-voc.cfg before starting the training
Line 3: change batch=1 to batch=64
Line 4: change subdivisions=1 to subdivisions=8 (increase the value to **16** if you are having cuda:out of memory error)
Line 237: in the last convolutional layer change the filter value= 120 to ((5)+(no of classes))*5
If no of classes=1.
Then filter value=30
Line 244: change the classes=20 to the number of classes in your dataset
Now use the modified yolo-voc.cfg file in the training step in the link below,

2. **Modifying config file and training YOLO:**

Use the new yolo-voc.cfg
./darknet detector train cfg/voc.data cfg/yolo-voc.cfg darknet19_448.conv.23
DEBUGGING ERRORS:
CONCLUSION: