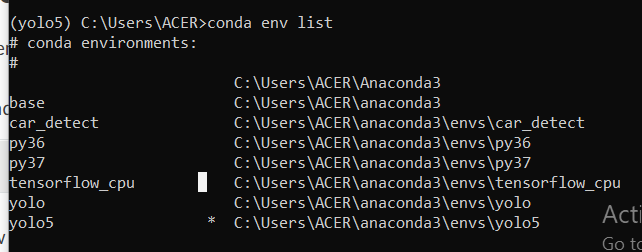
**Step 1** ( very important ): Install required package and module

This project run the virtual conda environment named yolo5



Every requirement for version of libraries and packages and modules are specified in the file requirements.txt:

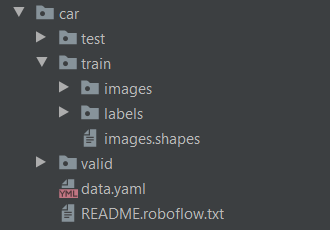
# pip install -U -r requirements.txt  
Cython  
numpy==1.17  
opencv-python  
torch>=1.4  
matplotlib  
pillow  
tensorboard  
PyYAML>=5.3  
torchvision  
scipy  
tqdm  
git+https://github.com/cocodataset/cocoapi.git#subdirectory=PythonAPI

To install all of these, just run “pip install -U -r requirements.txt”

Step 2: Prepare data

Để train được yolo5 thì yêu cầu đặt ra là tất cả các bức ảnh đều có hình vuông, tức là width và height bằng nhau, và hơn nữa size của tất cả ảnh phải giống nhau

Ví dụ đây là data của chúng ta



Thì tất cả các bức ảnh trong tệp car/train/images đều có width và height là 416 và 416

416 được chọn vì nó là bội số của 32, nên sẽ hỗ trợ trong việc train model.

Vấn đề nữa là annotation, chúng ta cần annotation dạng yolo5-pytorch,

Tức là ứng với mỗi bức ảnh sẽ có một file txt tương ứng và data lưu ở dạng yolo5-pytorch format.

Ví dụ



Với các field là class, x\_center,y\_center,width,height

Chú ý rằng x\_center,y\_center,width,height đều đã được normalized để nằm trong khoảng 0-1

Tức là x\_center = (real x\_center)/(width of image)

Y\_center = (real y\_center)/(height of image)

Width = (real width)/(width of image)

Height = (real height)/(height of image)

Tuy nhiên, dataset của ta thường có các bức ảnh với size khác nhau, và vấn đề đặt ra là resize các ảnh này đồng thời update file annotation.

Để giải quyết vấn đề này, ta có thể sử dụng roboflow

Đầu tiên là ta cần có folder chứa các bức ảnh của ta, giả sử folder tên là images

Tiếp theo ta cần đặt tất các file annotation vào trong folder này.

Và roboflow sẽ hỗ trợ các dạng annotation sau:

1. Pascal voc xml

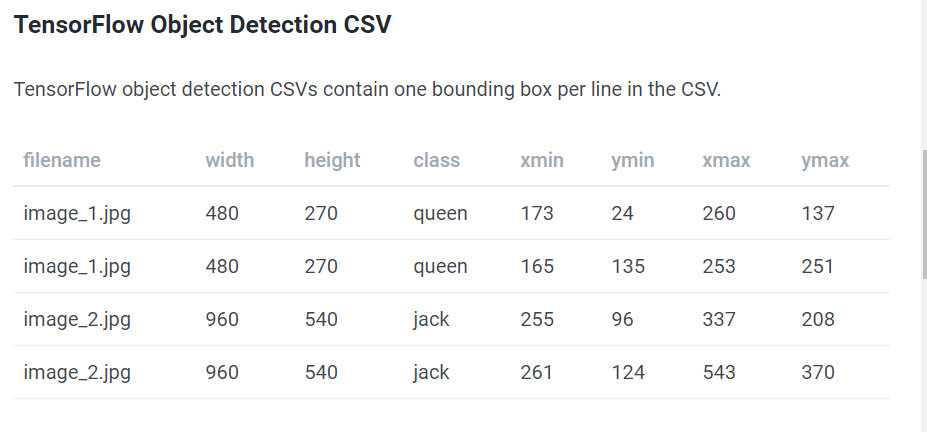
Tức là ứng với mỗi bức ảnh sẽ có một file xml tương ứng

1. Coco json

Thực ra hỗ trợ dạng này chủ yếu là để khai thác được coco dataset

1. Tensorflow object detection csv:

Tức là file được lưu dưới dạng như sau:



Và còn một vài dạng ít phổ biến khác

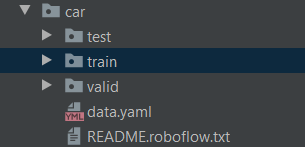
Tất cả đều được liệt kê ở đây:

[https://docs.roboflow.ai/adding-data/object-detection#tensorflow-object-detection-csv](https://docs.roboflow.ai/adding-data/object-detection" \l "tensorflow-object-detection-csv)

Sau khi đã tải data lên, thì chọn resize, ta nên resize thành 416\*416 là đẹp nhất, nếu muốn cỡ khác thì nên là a\*a với a là bội số của 32, và nên chọn a sát với kích thước thật của data ban đầu nhất.

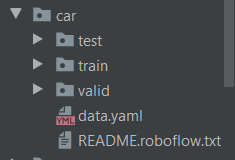
Sau khi đã tạo data xong, ở bước tải xuống, phải chọn dạng là yolov-pytorch

Sau đó, tải xuống được một file zip, giải nén đi và ta được một tệp như sau:



Step 3: Adjust the config file

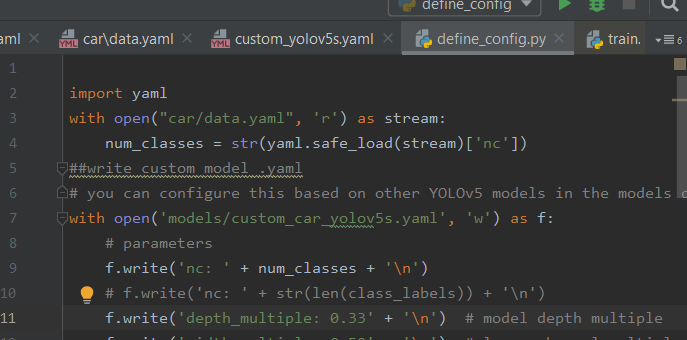
With each dataset, it would have a file data.yaml



We have to create another file yaml basing on this data.yaml file

To do that, we need to parse the path of our data.yaml file to file define\_config.py and define the name of output yaml file in define\_config.py to run it.

For example, with our example:



In the file <name of folder containing data>/data.yaml ( for example , car/data.yaml), we need to provide corresponding data of our dataset.

Step 4: Training

In this project, we can run this line in the terminal of pycharm to train:

python train.py --img 416 --batch 16 --epochs 1 --data car/data.yaml --cfg models/custom\_car\_yolov5s.yaml --name yolov5s\_results --nosave --cache --name car

Step 5: Testing

After training, the weight would be saved in directory weights

For example, in our project, it is weights/last\_yolov5s\_results.pt

Then, for this project, if we want to run, we can run this on the terminal:

python detect.py --weights weights/last\_car.pt --img 416 --conf 0.4 --source inference/images

here, the folder inference/images would contain images for testing and the size of images must be 416\*416 because we have trained our data on image with size 416\*416. We can get this folder images by using the images in folder dataset/car/test/images

And running this would output images with predicted boxes in the folder inference/output