**Yolov3\_darknet 本地安装配置**

汪雄 2018/09/25

此教程参考：

<https://github.com/AlexeyAB/darknet>

1. 去github上下载Yolov3文件

<https://github.com/AlexeyAB/darknet.git>

在linux系统下，输入

**git clone <https://github.com/AlexeyAB/darknet.git>**

采坑记录：

由于之前是window系统下已经将该文件下载好，就直接拷贝到linux系统下，结果运行demo的时候，报错

1. compiling with cuda

修改Makefile文件：

GPU = 0改为 GPU = 1

Make编译命令

1.编译项目

命令：make

1. 清理编译过的项目

Make clean

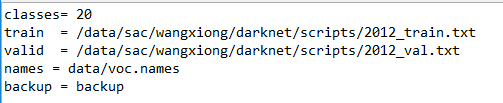
**Yolov使用**

1. **训练模型**

需要准备好以下几项数据：

1. **Data文件**

voc.data:该文件夹存放的是



分类数目

训练数据集文件夹

测试数据集文件夹

每一类对应的类别名称

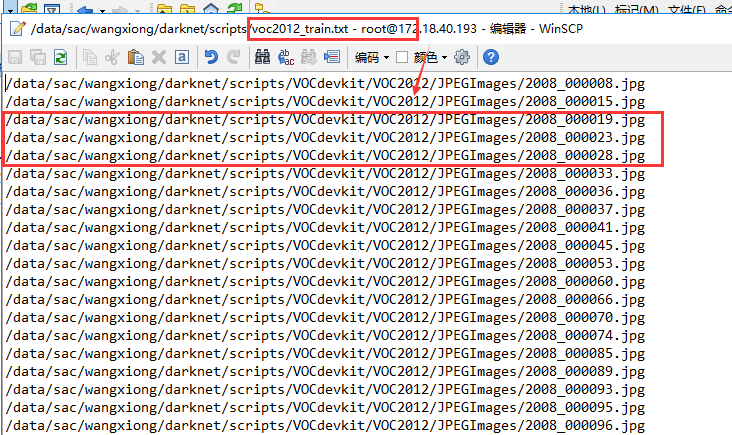
模型数据存储的文件夹

**补充说明：**

train：训练集中的图片的绝对地址

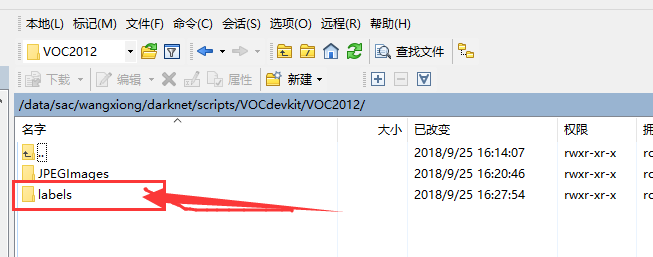
Valid：验证集中图片的绝对地址

如图所示：



**疑问：**只提供了训练的图片路径，那标注数据在哪呢？

在图片存放的文件夹的同一级目录里，有一个labels文件夹，相应的标注数据就存放在该文件里面，如下图所示



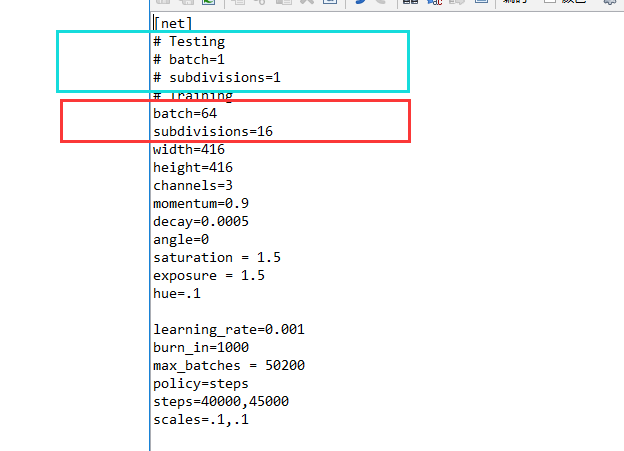
1. **CFG文件**

训练自己的分类，配置文件如何配置，请参考：

<https://github.com/AlexeyAB/darknet>

How to train(to detect your custom objects):

yolov3-voc.cfg:该文件是训练中模型参数的配置文件



分为训练和测试。

在训练时，将红线框中的打开

在测试时，将青线框中的打开

1. 预训练好的模型

下载预训练好的文件 darknet53.conv.74

下载地址：

<https://pjreddie.com/media/files/darknet53.conv.74>

1. **开始训练数据**

参考网友：jieshaoxiansen

网址：<https://blog.csdn.net/jieshaoxiansen/article/details/81281481>

**命令：**

**单GPU训练：**

./darknet detector train cfg/voc.data cfg/yolov3-voc.cfg darknet53.conv.74

**多GPU训练：**

./darknet detector train cfg/voc.data cfg/yolov3-voc.cfg darknet53.conv.74 -gpus 0,1,2,3

CPU 训练：

./darknet -nogpu detector train cfg/voc.data cfg/yolov3-voc.cfg darknet53.conv.74

1. **测试模型**

**在测试模型之前，需要将配置文件 yolov3-voc.cfg改为测试模型**

**命令：**

./darknet detector test cfg/voc.data cfg/yolov3-voc.cfg backup/yolov3-voc\_10000.weights data/dog.jpg

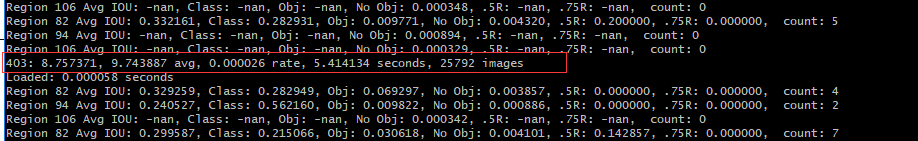
**Yolov3模型训练输出参数解读：**

参考网友：magic428

链接： <https://blog.csdn.net/gzj2013/article/details/82285511>

**Batch输出：**

训练过程中输出截图

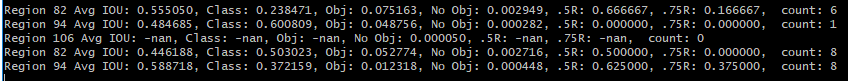


红线框中参数代表的意思解读：

* 403：指示当前训练的迭代次数
* 8.757371：总体的Loss（损失值）
* 9.743887 avg：是平均Loss
* 0.000026 rate:代表当前的学习率
* 5.414134 seconds:表示当前批次训练花费的总时间
* 25792 images： 表示403\*64，表示到目前为止，参与训练的图片的总量

**Subdivision输出：**

训练过程中输出截图



以上信息表示三个不同尺度（82,94,106）上预测的不同大小的框的参数

* 82卷积层为最大的预测尺度，使用较大的mask，但是可以预测出较小的物体
* 94卷积层为中间的预测尺度，使用中等的mask
* 106卷积层为最小的预测尺度，使用较小的mask，可以预测出较大的物体

上述输出信息的各个参数含义是（主要观察某一尺度上的参数），下面就Region 82分析

* Region 82 Avg IOU:0.446188：表示在当前subdivision内的图片的平均IOU， 代表预测的矩形和真实目标的交集与并集之比
* Class：0.503023：标注物理分类的正确率，期望该值趋近于1（这个数怎么计算的？）
* Obj：0.052774：越接近1越好
* No Obj：0.002716：期望该值越来越小，但不为零
* .5R:0.5000：以IOU=0.5为阈值时的recall/count值，是当前模型在所有subdivision图片中检测的正样本与实际的正样本的比值
* .7R:0.000 以IOU = 0.7为阈值时的recall/count值
* Count：所有当前subdivision图片（本例中一共16张）中包含正样本的图片的数量，当该值为nan时，说明该批次没有预测到正样本

**Yolov3 预测结果输出**

1. **日志输出（输出重定向）：**

参考网友：Gotta-C

网址：<https://blog.csdn.net/cgt19910923/article/details/80783614>

**命令：**

./darknet detector train cfg/voc.data cfg/yolov3-voc.cfg darknet53.conv.74 -gpus 0, 1 2>1 | tee -a train\_yolov3.log

**命令解析：**

| tee train\_yolov3.log：是linux中的命令，将终端输出，输出到文本文件

-a：以附加的形式添加进来

将多个命令的输出都输出到文件

命令：script-

退出该模式：exit

1. **linux远程连接断开后，继续运行程序**

参考网友：热血-文宗

网址：<https://blog.csdn.net/zhang5207892/article/details/78934256>

命令参数：

Screen -S name #创建一个名为name的会话

关闭该远程连接后，再次进入该虚拟终端

Screen -r name

查看所有的screen

Screen -ls

退出screen

exit

删除会话

Screen -wipe name

参考手册：

<http://man.linuxde.net/screen>



1. **将预测的结果输出为txt**

**命令：**

./darknet detector test data/voc.data cfg/yolov3-voc.cfg yolov3.weights data/dog.jpg -ext\_output

1. **批量处理并且将数据保存到文件中**

**命令：**

./darknet detector test data/voc.data cfg/yolov3-voc.cfg yolov3-voc.weight -dont\_show -ext\_output < data/train.txt > result.txt

Yolov3进行预测模型

Demo

./darknet detector cfg/yolov3.cfg yolov3.weights data/dog

./darknet detector train cfg/pod.data cfg/yolov3\_pod.cfg darknet53.conv.74 -gpus 0,1 | tee yolov3\_train\_pod.log

1. **调用Python接口来批量处理图片文件**

Yolov3训练POD数据

1. 数据准备

服务：

Ssh [root@172.18.40.193](mailto:root@172.18.40.193)

密码：CNDPlab96321(^#@!

进入目录：

cd /data/sac/darknet

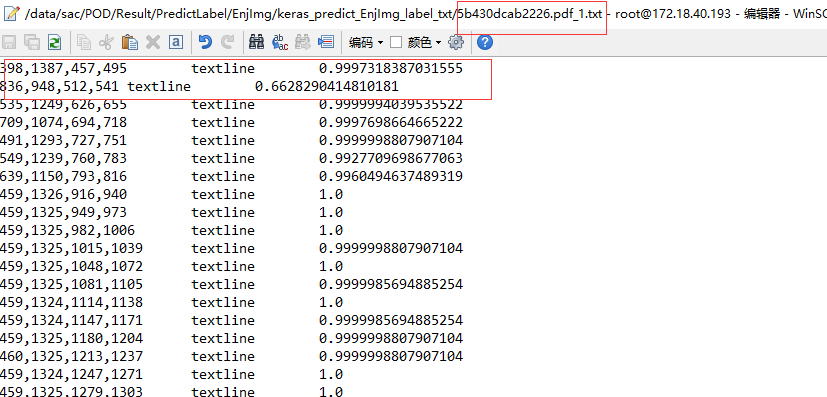
用darknbet测试数据：

Demo :

./darknet detector test pod.data yolo-pod.cfg yolo-pod\_19500.weights -dont\_show -ext\_output < /data/sac/POD/DataSet/ICDAR2013/img\_path.txt > yolov2\_predict\_ICDAR2013\_Table.txt

任务要求：

1. 将预测的结果 yolov2\_predict\_ICDAR2013\_Table.txt 文件解析出来，输出结果要求如下：



1. 将/data/sac/POD/DataSet/ICDAR2013/formulaTest 数据用yolov2模型跑出结果，得到txt数据。

附录：

服务器上：用pod2这个python环境，该python环境已经配好了需要用的库

激活该环境的指令：

source activate pod2

基于Keras的Yolov3本地训练测试：

1. 下载地址：

<https://github.com/qqwweee/keras-yolo3>

1. 使用说明：

Test:

python yolo\_video.py --image

#进入图像检测模式

#使用的是默认的anchor和classes

"anchors\_path": 'model\_data/yolo\_anchors.txt',

"classes\_path": 'model\_data/coco\_classes.txt',

Train：

python train.py

#使用的数据集的格式：

1. 将Darknet训练好的weights文件转换为keras的h5文件

研究转换脚本 convert.py

python convert.py -w yolov3.cfg yolov3.weights model\_data/yolo\_weight.h5

#将yolov3.weights转换成keras下的h5文件