（3）环境变量配置

打开~/.bashrc文件：

sudo gedit ~/.bashrc

将以下内容写入到~/.bashrc尾部：

export PATH=/usr/local/cuda-7.5/bin${PATH:+:${PATH}}

export LD\_LIBRARY\_PATH=/usr/local/cuda-7.5/lib64${LD\_LIBRARY\_PATH:+:${LD\_LIBRARY\_PATH}}

在/etc/plsrofile文件中添加CUDA环境变量：

sudo gedit /etc/profile

打开文档都在文档结尾加上下面两句：

1 PATH=/usr/local/cuda/bin:$PATH

2 export PATH

保存后, 执行下列命令, 使环境变量立即生效：

source /etc/profile

同时添加lib库路径，在 /etc/ld.so.conf.d/新建文件 cuda.conf

sudo gedit /etc/ld.so.conf.d/cuda.conf

在文中加入下面内容：

/usr/local/cuda/lib64

执行下列命令使之立刻生效：

sudo ldconfig

(4)测试CUDA的samples

1 cd /usr/local/cuda-7.5/samples/1\_Utilities/deviceQuery

2 make

3 sudo ./deviceQuery

如果显示一些关于GPU的信息，则说明安装成功。

nvcc –V 命令可查看版本

（1）安装cython, python-opencv,easydict

pip install cython

pip install easydict

apt-get install python-opencv

1、安装OpenCV的依赖包

[编译器] > sudo apt-get install build-essential

[必须安装] > sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev

[可选择安装] > sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev

2、  从源代码构建OpenCV

安装完Python、与Python相关的软件包、cmake后就可构建OpenCV了。首先从

 https://github.com/opencv/opencv/archive/3.0.0-beta.zip下载源代码并解压，然后将其移动到存放在终端上的解压文件夹的文件夹中。

然后运行以下命令:

>> mkdir build

>> cd build

利用下面的cmake命令进行编译设置

>> cmake –D CMAKE\_BUILD\_TYPE=Release –D CMAKE\_INSTALL\_PREFIX=/usr/local ..

>> make

>> make install

若在python下运行opencv库，必须安装如下

sudo apt-get install python-opencv

完成上述流程即安装完成

3、进行测试，检查opencv是否安装成功

下述代码中要注意，图片与代码文件要在一个文件夹下，否则要注明图片路径。

（2）下载py-faster-rcnn

# Make sure to clone with --recursive

git clone --recursive https://github.com/rbgirshick/py-faster-rcnn.git

如图：

（3）进入py-faster-rcnn/lib

   执行make

如图：

（4）进入py-faster-rcnn\caffe-fast-rcnn

执行 cp Makefile.config.example Makefile.config

然后，配置Makefile.config文件，可参考我的配置：Makefile.config文件

配置好Makefile.config文件后，执行：

make -j8 && make pycaffe

如图：

（5）下载VOC2007数据集

提供一个百度云地址：http://pan.baidu.com/s/1mhMKKw4

解压，然后，将该数据集放在py-faster-rcnn\data下，用你的数据集替换VOC2007数据集。（替换Annotations，ImageSets和JPEGImages）

（用你的Annotations，ImagesSets和JPEGImages替换py-faster-rcnn\data\VOCdevkit2007\VOC2007中对应文件夹）

（6）下载ImageNet数据集下预训练得到的模型参数（用来初始化）

提供一个百度云地址：http://pan.baidu.com/s/1hsxx8OW

解压，然后将该文件放在py-faster-rcnn\data下

下面是训练前的一些修改。

1.py-faster-rcnn/models/pascal\_voc/ZF/faster\_rcnn\_alt\_opt/stage1\_fast\_rcnn\_train.pt修改

layer {

name: 'data'

type: 'Python'

top: 'data'

top: 'rois'

top: 'labels'

top: 'bbox\_targets'

top: 'bbox\_inside\_weights'

top: 'bbox\_outside\_weights'

python\_param {

module: 'roi\_data\_layer.layer'

layer: 'RoIDataLayer'

param\_str: "'num\_classes': 16" #按训练集类别改，该值为类别数+1

}

}

layer {

name: "cls\_score"

type: "InnerProduct"

bottom: "fc7"

top: "cls\_score"

param { lr\_mult: 1.0 }

param { lr\_mult: 2.0 }

inner\_product\_param {

num\_output: 16 #按训练集类别改，该值为类别数+1

weight\_filler {

type: "gaussian"

std: 0.01

}

bias\_filler {

type: "constant"

value: 0

}

}

}

layer {

name: "bbox\_pred"

type: "InnerProduct"

bottom: "fc7"

top: "bbox\_pred"

param { lr\_mult: 1.0 }

param { lr\_mult: 2.0 }

inner\_product\_param {

num\_output: 64 #按训练集类别改，该值为（类别数+1）\*4

weight\_filler {

type: "gaussian"

std: 0.001

}

bias\_filler {

type: "constant"

value: 0

}

}

}

2.py-faster-rcnn/models/pascal\_voc/ZF/faster\_rcnn\_alt\_opt/stage1\_rpn\_train.pt修改

layer {

name: 'input-data'

type: 'Python'

top: 'data'

top: 'im\_info'

top: 'gt\_boxes'

python\_param {

module: 'roi\_data\_layer.layer'

layer: 'RoIDataLayer'

param\_str: "'num\_classes': 16" #按训练集类别改，该值为类别数+1

}

}

3.py-faster-rcnn/models/pascal\_voc/ZF/faster\_rcnn\_alt\_opt/stage2\_fast\_rcnn\_train.pt修改

layer {

name: 'data'

type: 'Python'

top: 'data'

top: 'rois'

top: 'labels'

top: 'bbox\_targets'

top: 'bbox\_inside\_weights'

top: 'bbox\_outside\_weights'

python\_param {

module: 'roi\_data\_layer.layer'

layer: 'RoIDataLayer'

param\_str: "'num\_classes': 16" #按训练集类别改，该值为类别数+1

}

}

layer {

name: "cls\_score"

type: "InnerProduct"

bottom: "fc7"

top: "cls\_score"

param { lr\_mult: 1.0 }

param { lr\_mult: 2.0 }

inner\_product\_param {

num\_output: 16 #按训练集类别改，该值为类别数+1

weight\_filler {

type: "gaussian"

std: 0.01

}

bias\_filler {

type: "constant"

value: 0

}

}

}

layer {

name: "bbox\_pred"

type: "InnerProduct"

bottom: "fc7"

top: "bbox\_pred"

param { lr\_mult: 1.0 }

param { lr\_mult: 2.0 }

inner\_product\_param {

num\_output: 64 #按训练集类别改，该值为（类别数+1）\*4

weight\_filler {

type: "gaussian"

std: 0.001

}

bias\_filler {

type: "constant"

value: 0

}

}

}

4.py-faster-rcnn/models/pascal\_voc/ZF/faster\_rcnn\_alt\_opt/stage2\_rpn\_train.pt修改

layer {

name: 'input-data'

type: 'Python'

top: 'data'

top: 'im\_info'

top: 'gt\_boxes'

python\_param {

module: 'roi\_data\_layer.layer'

layer: 'RoIDataLayer'

param\_str: "'num\_classes': 16" #按训练集类别改，该值为类别数+1

}

}

5.py-faster-rcnn/models/pascal\_voc/ZF/faster\_rcnn\_alt\_opt/faster\_rcnn\_test.pt修改

layer {

name: "cls\_score"

type: "InnerProduct"

bottom: "fc7"

top: "cls\_score"

inner\_product\_param {

num\_output: 16 #按训练集类别改，该值为类别数+1

}

}

layer {

name: "bbox\_pred"

type: "InnerProduct"

bottom: "fc7"

top: "bbox\_pred"

inner\_product\_param {

num\_output: 64 #按训练集类别改，该值为（类别数+1）\*4

}

}

6.py-faster-rcnn/lib/datasets/pascal\_voc.py修改

（1）

class pascal\_voc(imdb):

def \_\_init\_\_(self, image\_set, year, devkit\_path=None):

        imdb.\_\_init\_\_(self, 'voc\_' + year + '\_' + image\_set)

        self.\_year = year

        self.\_image\_set = image\_set

        self.\_devkit\_path = self.\_get\_default\_path() if devkit\_path is None \

                            else devkit\_path

        self.\_data\_path = os.path.join(self.\_devkit\_path, 'VOC' + self.\_year)

self.\_classes = ('\_\_background\_\_', # always index 0

'你的标签1','你的标签2',你的标签3','你的标签4'

)

上面要改的地方是

修改训练集文件夹：

self.\_data\_path = os.path.join(self.\_devkit\_path, 'VOC'+self.\_year)

用你的数据集直接替换原来VOC2007内的Annotations，ImageSets和JPEGImages即可,以免出现各种错误。

修改标签：

self.\_classes = ('\_\_background\_\_', # always index 0

'你的标签1','你的标签2','你的标签3','你的标签4'

)

修改成你的数据集的标签就行。

（2）

cls = self.\_class\_to\_ind[obj.find('name').text.lower().strip()]

这里把标签转成小写，如果你的标签含有大写字母，可能会出现KeyError的错误，所以建议标签用小写字母。

（去掉lower应该也行）

建议训练的标签还是用小写的字母，如果最终需要用大写字母或中文显示标签，可参考：

http://blog.csdn.net/sinat\_30071459/article/details/51694037

7.py-faster-rcnn/lib/datasets/imdb.py修改

该文件的append\_flipped\_images(self)函数修改为：

def append\_flipped\_images(self):

num\_images = self.num\_images

widths = [PIL.Image.open(self.image\_path\_at(i)).size[0]

for i in xrange(num\_images)]

for i in xrange(num\_images):

boxes = self.roidb[i]['boxes'].copy()

oldx1 = boxes[:, 0].copy()

oldx2 = boxes[:, 2].copy()

boxes[:, 0] = widths[i] - oldx2 - 1

print boxes[:, 0]

boxes[:, 2] = widths[i] - oldx1 - 1

print boxes[:, 0]

assert (boxes[:, 2] >= boxes[:, 0]).all()

entry = {'boxes' : boxes,

'gt\_overlaps' : self.roidb[i]['gt\_overlaps'],

'gt\_classes' : self.roidb[i]['gt\_classes'],

'flipped' : True}

self.roidb.append(entry)

self.\_image\_index = self.\_image\_index \* 2

这里assert (boxes[:, 2] >= boxes[:, 0]).all()可能出现AssertionError，具体解决办法参考：

http://blog.csdn.net/xzzppp/article/details/52036794

！！！为防止与之前的模型搞混,训练前把output文件夹删除（或改个其他名），还要把py-faster-rcnn/data/cache中的文件和

py-faster-rcnn/data/VOCdevkit2007/annotations\_cache中的文件删除（如果有的话）。

至于学习率等之类的设置，可在py-faster-rcnn/models/pascal\_voc/ZF/faster\_rcnn\_alt\_opt中的solve文件设置，迭代次数可在py-faster-rcnn\tools的train\_faster\_rcnn\_alt\_opt.py中修改：

max\_iters = [80000, 40000, 80000, 40000]

分别为4个阶段（rpn第1阶段，fast rcnn第1阶段，rpn第2阶段，fast rcnn第2阶段）的迭代次数。可改成你希望的迭代次数。

如果改了这些数值，最好把py-faster-rcnn/models/pascal\_voc/ZF/faster\_rcnn\_alt\_opt里对应的solver文件（有4个）也修改，stepsize小于上面修改的数值。

8.开始训练

进入py-faster-rcnn，执行：

./experiments/scripts/faster\_rcnn\_alt\_opt.sh 0 ZF pascal\_voc

这样，就开始训练了。

9.测试

将训练得到的py-faster-rcnn\output\faster\_rcnn\_alt\_opt\\*\*\*\_trainval中ZF的caffemodel拷贝至py-faster-rcnn\data\faster\_rcnn\_models（如果没有这个文件夹，就新建一个），然后，修改：

py-faster-rcnn\tools\demo.py,主要修改：

CLASSES = ('\_\_background\_\_',

'你的标签1', '你的标签2', '你的标签3', '你的标签4')

改成你的数据集标签；

NETS = {'vgg16': ('VGG16',

'VGG16\_faster\_rcnn\_final.caffemodel'),

'zf': ('ZF',

'ZF\_faster\_rcnn\_final.caffemodel')}

上面ZF的caffemodel改成你的caffemodel。

im\_names = ['1559.jpg','1564.jpg']

改成你的测试图片。(测试图片放在py-faster-rcnn\data\demo中)

10.结果

在py-faster-rcnn下，

执行：

./tools/demo.py --net zf

或者将默认的模型改为zf：

parser.add\_argument('--net', dest='demo\_net', help='Network to use [vgg16]',

choices=NETS.keys(), default='vgg16')

修改：

default='zf'

执行：

./tools/demo.py

﻿﻿

---------------------

作者：小咸鱼\_

来源：CSDN

原文：https://blog.csdn.net/sinat\_30071459/article/details/51332084

版权声明：本文为博主原创文章，转载请附上博文链接！

1.安装相关依赖项（直接打开终端一条条执行即可）

sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libhdf5-serial-dev protobuf-compiler

sudo apt-get install --no-install-recommends libboost-all-dev

sudo apt-get install libopenblas-dev liblapack-dev libatlas-base-dev

sudo apt-get install libgflags-dev libgoogle-glog-dev liblmdb-dev

---------------------

作者：A\_Z666666

来源：CSDN

原文：https://blog.csdn.net/A\_Z666666/article/details/72853346

版权声明：本文为博主原创文章，转载请附上博文链接！

.安装opencv3.1

首先肯定是先安装依赖了，官方列出了一些：

sudo apt-get install build-essential

sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev

sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev

sudo apt-get install --assume-yes libopencv-dev libdc1394-22 libdc1394-22-dev libjpeg-dev libpng12-dev libtiff5-dev libjasper-dev libavcodec-dev libavformat-dev libswscale-dev libxine2-dev libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev libv4l-dev libtbb-dev libqt4-dev libfaac-dev libmp3lame-dev libopencore-amrnb-dev libopencore-amrwb-dev libtheora-dev libvorbis-dev libxvidcore-dev x264 v4l-utils unzip

sudo apt-get install ffmpeg libopencv-dev libgtk-3-dev python-numpy python3-numpy libdc1394-22 libdc1394-22-dev libjpeg-dev libpng12-dev libtiff5-dev libjasper-dev libavcodec-dev libavformat-dev libswscale-dev libxine2-dev libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev libv4l-dev libtbb-dev qtbase5-dev libfaac-dev libmp3lame-dev libopencore-amrnb-dev

---------------------

作者：A\_Z666666

来源：CSDN

原文：https://blog.csdn.net/A\_Z666666/article/details/72853346

版权声明：本文为博主原创文章，转载请附上博文链接！