[Fast RCNN ubuntu下安装笔记](http://blog.csdn.net/tangwei2014/article/details/45442275)

微软研究院大神Ross Girshick新作：fase CNN，测试速度比RCNN快200倍。

文章：http://arxiv.org/pdf/1504.08083v1.pdf

代码：[https://github.com/rbgirshick/fast-rcnn](https://github.com/rbgirshick/fast-rcnn" \t "_blank)

安装笔记：

第一步：

严格按照安装步骤，特别是第一个：

git clone --recursive <https://github.com/rbgirshick/fast-rcnn.git>

不要尝试着自己下载安装，不然很多东西下载不全面

第二步：Build the Cython modules

     cd $FRCN\_ROOT/lib

     make

     如果出现问题：

     ImportError: No module named Cython.Distutils

     解决： 先 sudo pip install cython  然后再make

第三步：在caffe-fast-rcnn目录下， 修改：

     # In your Makefile.config, make sure to have this line uncommented

     WITH\_PYTHON\_LAYER:=1

     然后make -j16 && make pycaffe

第四步：

     cd $FRCN\_ROOT

     ./data/scripts/fetch\_fast\_rcnn\_models.sh

如果遇到问题，可能要：

     sudo pip install  easydict

        sudo pip install protobuf

# 研究Fast rcnn代码

原文：[http://arxiv.org/pdf/1504.08083.pdf](http://arxiv.org/pdf/1504.08083.pdf" \t "_blank)

代码：[https://github.com/rbgirshick/fast-rcnn](https://github.com/rbgirshick/fast-rcnn" \t "_blank)

**1.准备工作**

1.1 软件准备

    首先，需要安装Caffe和pycaffe。

    caffe原作者网页：<http://caffe.berkeleyvision.org/installation.html>

    欧新宇师兄的caffe安装说明：<http://ouxinyu.github.io/Blogs/20140723001.html>

    注意：必须在 Makefile.config配置文件中打开Python层支持。

    # In your Makefile.config, make sure to have this line uncommented

    WITH\_PYTHON\_LAYER := 1

    其次，可能需要Python安装包：cython，python-opencv，easydict

    先装一个python包管理器pip：

    sudo apt-get install python-pip

    再装那三个包：

    sudo pip install cython

    #sudo pip install python-opencv

    sudo pip install easydict

    再次，可能需要MATLAB，主要用于对PASCAL voc数据集的评估。

1.2 硬件准备

    对于训练较小的网络（CaffeNet, VGG\_CNN\_M\_1024），至少需要3G内存的GPU（如：Titan，K20，K40...）

    对于训练VGG16，至少需要一个K40（约11G内存），这里我们就不考虑了。

**2.安装（用于demo）**

2.1 从github上clone到Fast RCNN的仓库。最好就直接这么clone，不要自己去下载，不然还满麻烦的。

    # Make sure to clone with --recursive

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

2.2 生成Cython模块（下面的$FRCN\_ROOT都是指fast-rcnn的解压位置）

    cd $FRCN\_ROOT/lib

    make

2.3 生成Caffe和pycaffe

    cd $FRCN\_ROOT/caffe-fast-rcnn

    # Now follow the Caffe installation instructions here:

    #   http://caffe.berkeleyvision.org/installation.html

    # If you're experienced with Caffe and have all of the requirements installed

    # and your Makefile.config in place, then simply do:

    make -j8 && make pycaffe

2.4 下载Fast RCNN检测器

    cd $FRCN\_ROOT

    ./data/scripts/fetch\_fast\_rcnn\_models.sh

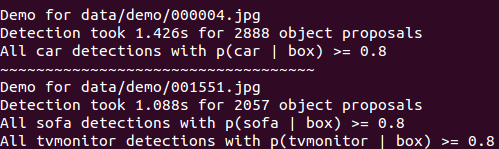
**3.运行demo**

3.1 Python版

    cd $FRCN\_ROOT

    ./tools/demo.py

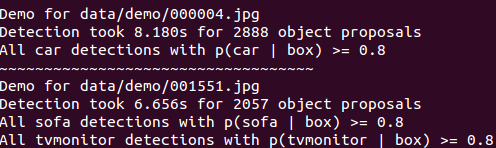
可能我安装了cudnn，所以即使2G内存的GPU也是可以的。

[](http://photo.blog.sina.com.cn/showpic.html#blogid=855a82cd0102vnjq&url=http://album.sina.com.cn/pic/002rpujjgy6SYw5VTob3f)

    如果用CPU模式，就是

    cd $FRCN\_ROOT

    ./tools/demo.py --cpu

[](http://photo.blog.sina.com.cn/showpic.html#blogid=855a82cd0102vnjq&url=http://album.sina.com.cn/pic/002rpujjgy6SYw75UQBe5)

显然是慢很多的。效果图如下所示：  

[](http://photo.blog.sina.com.cn/showpic.html#blogid=855a82cd0102vnjq&url=http://album.sina.com.cn/pic/002rpujjgy6SYwcsZkMdc)

    demo中是用VGG16网络，在PASCAL VOC2007上训练的模型来执行检测的，这个模型比较大，如果把caffe弄崩溃了，可以换一个小一点的网络，其实还更快一点，如

    ./tools/demo.py --net caffenet

    或者

    ./tools/demo.py --net vgg\_cnn\_m\_1024

    或者就用CPU模式好了。

3.2 MATLAB版（暂时没找到编译好的caffe，现在先不管）

    在matlab文件夹下打开matlab，下面是我的matlab的安装地址。

    cd $FRCN\_ROOT/matlab

/usr/local/MATLAB/R2014a/bin/matlab  # wait for matlab to start...

    把$FRCN\_ROOT/caffe-fast-rcnn/matlab下的caffe文件夹拷贝到$FRCN\_ROOT/matlab中，为防止内存不够，我们还是以CaffeNet为例，把fast-rcnn-demo.m中的所有VGG16改为CaffeNet。在matlab命令行下输入命令：

    >> fast\_rcnn\_demo

[](http://photo.blog.sina.com.cn/showpic.html#blogid=855a82cd0102vnjq&url=http://album.sina.com.cn/pic/002rpujjgy6SYwemWVL15)

3.3 一些获取object proposal的算法代码

    Selective Search: [original matlab code](http://disi.unitn.it/~uijlings/MyHomepage/index.php#page=projects1), [python wrapper](https://github.com/sergeyk/selective_search_ijcv_with_python)

    EdgeBoxes: [matlab code](https://github.com/pdollar/edges)

    GOP and LPO: [python code](http://www.philkr.net/)

    MCG: [matlab code](http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/mcg/)

    RIGOR: [matlab code](http://cpl.cc.gatech.edu/projects/RIGOR/)

**4.准备数据集**

4.1 首先要下载训练集、验证集、测试集，例子是VOC2007。资源在墙外，将给出百度云盘中的地址。

    wget http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/VOCtrainval\_06-Nov-2007.tar

    wget http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/VOCtest\_06-Nov-2007.tar

    wget http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/VOCdevkit\_08-Jun-2007.tar

4.2 提取所有压缩包到同一个下面称为$VOCdevkit的文件夹下。

    tar xvf VOCtrainval\_06-Nov-2007.tar

    tar xvf VOCtest\_06-Nov-2007.tar

    tar xvf VOCdevkit\_08-Jun-2007.tar

    要有这些基本的目录：

    $VOCdevkit/                           # development kit

    $VOCdevkit/VOCcode/                   # VOC utility code

    $VOCdevkit/VOC2007                    # image sets, annotations, etc.

4.3 创建对VOC2007数据集的symlink，也就是链接FRCN\_ROOT和VOC2007的目录。

    cd $FRCN\_ROOT/data

    ln -s $VOCdevkit VOCdevkit2007

    这个方法非常好，因为别的工程里面也可能用到这个数据集，这样就不用多次拷贝了，节省了很多存储空间，windows下面就没有。

4.4 可以再用同样的办法得到VOC2010和2012的数据集，如果有需要的话。

4.5 下载预先用selective search计算好的object proposal。

    cd $FRCN\_ROOT

    ./data/scripts/fetch\_selective\_search\_data.sh

    会下载到$FRCN\_ROOT/data下，解压后是一个名为selective\_search\_data的文件夹。

4.6 下载预先训练好的ImageNet模型。

    cd $FRCN\_ROOT

    ./data/scripts/fetch\_imagenet\_models.sh

    下载到三个模型，分别是CaffeNet (model S), VGG\_CNN\_M\_1024 (model M), and VGG16 (model L)，会下载到$FRCN\_ROOT/data下，解压后是一个名为imagenet\_models的文件夹。

**5.模型的训练与测试**

5.1 训练模型

    训练Fast R-CNN检测器，以在VOC2007上训练一个CaffeNet 的网络为例。

    ./tools/train\_net.py --gpu 0 --solver models/CaffeNet/solver.prototxt --weights data/imagenet\_models/CaffeNet.v2.caffemodel

    这里我出现了EnvironmentError: MATLAB command 'matlab' not found. Please add 'matlab' to your PATH.这种错误，说明没把matlab的路径添加到环境变量中，下面的语句设置环境变量：

    export PATH=$PATH:"/usr/local/MATLAB/R2014a/bin"

    又提示说ImportError: No module named yaml，那就下载安装一个：

    sudo apt-get install python-yaml

    再次运行代码就可以了。如果显示内存不够，可以用nvidia-smi随时查看内存使用情况。每10000次迭代会生成一个model，结果存放在output文件夹中。

    训练VGG\_CNN\_M\_1024网络时，会提示说内存不够，就把$FRCN\_ROOT/lib/fast\_rcnn下的config.py中每个minibatch所用的图片由2改为1，如果还不行，说明GPU内存太小，只能换GPU了。

    ./tools/train\_net.py --gpu 0 --solver models/VGG\_CNN\_M\_1024/solver.prototxt --weights data/imagenet\_models/VGG\_CNN\_M\_1024.v2.caffemodel

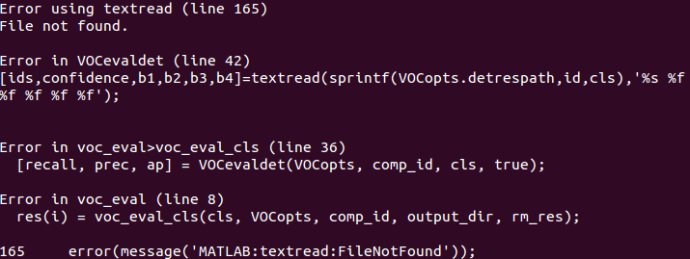
    训练 VGG16网络，据作者说，即使把每个minibatch所用的图片由2改为1，也需要将近5G的GPU内存，3G以上内存的可以尝试一下，cudnn可能在一定程度上起到了优化作用。

5.2 测试模型

    在自己的模型还没有训练好，或者训练得不够好的时候，可以试试作者提供的模型：

    ./tools/test\_net.py --gpu 0 --def models/CaffeNet/test.prototxt  --net data/fast\_rcnn\_models/caffenet\_fast\_rcnn\_iter\_40000.caffemodel

    在测试的时候一直报下面这样的错，困扰了很久，找到原因后觉得自己蠢哭了。

[](http://photo.blog.sina.com.cn/showpic.html#blogid=855a82cd0102vnjq&url=http://album.sina.com.cn/pic/002rpujjgy6SYwgbK0j73)

    把VOCevaldet中相应文件名输出来，发现问题出在VOCinit上，我们现在是在测试，把

[【深度学习】研究Fast <wbr>rcnn代码](http://photo.blog.sina.com.cn/showpic.html#blogid=855a82cd0102vnjq&url=http://album.sina.com.cn/pic/002rpujjgy6SYwiGLV35d)  
  
第一句注释掉，第二句取消注释。

    下面再测试自己的模型：

    ./tools/test\_net.py --gpu 0 --def models/CaffeNet/test.prototxt  --net output/default/voc\_2007\_trainval/caffenet\_fast\_rcnn\_iter\_40000.caffemodel

    测试的结果也在output文件夹中。

5.3 用全连接层压缩的SVD来压缩FRCNN模型

    ./tools/compress\_net.py --def models/CaffeNet/test.prototxt --def-svd models/CaffeNet/compressed/test.prototxt --net output/default/voc\_2007\_trainval/caffenet\_fast\_rcnn\_iter\_40000.caffemodel

    压缩后的模型和压缩前的模型是放在一起的，只是名字不一样，在output下的相应文件夹下。再测试这个压缩后的模型：

    ./tools/test\_net.py --gpu 0 --def models/CaffeNet/compressed/test.prototxt --net output/default/voc\_2007\_trainval/vcaffenet\_fast\_rcnn\_iter\_40000\_svd\_fc6\_1024\_fc7\_256.caffemodel

    好像是快了一些吧，反正这也不是重点。

**附录**

1.$FRCN\_ROOT/experiments/scripts下的这些脚本可以再现作者论文中的实验，有兴趣的话可以试一下。

2.日志文件下载地址：[Experiment logs](http://www.cs.berkeley.edu/~rbg/fast-rcnn-data/fast_rcnn_experiments.tgz)

3.PASCAL VOC的一些检测结果

[voc\_2007\_test\_results\_fast\_rcnn\_caffenet\_trained\_on\_2007\_trainval.tgz](http://www.cs.berkeley.edu/~rbg/fast-rcnn-data/voc_2007_test_results_fast_rcnn_caffenet_trained_on_2007_trainval.tgz)[voc\_2007\_test\_results\_fast\_rcnn\_vgg16\_trained\_on\_2007\_trainval.tgz](http://www.cs.berkeley.edu/~rbg/fast-rcnn-data/voc_2007_test_results_fast_rcnn_vgg16_trained_on_2007_trainval.tgz)[voc\_2007\_test\_results\_fast\_rcnn\_vgg\_cnn\_m\_1024\_trained\_on\_2007\_trainval.tgz](http://www.cs.berkeley.edu/~rbg/fast-rcnn-data/voc_2007_test_results_fast_rcnn_vgg_cnn_m_1024_trained_on_2007_trainval.tgz)    [voc\_2012\_test\_results\_fast\_rcnn\_vgg16\_trained\_on\_2007\_trainvaltest\_2012\_trainval.tgz](http://www.cs.berkeley.edu/~rbg/fast-rcnn-data/voc_2012_test_results_fast_rcnn_vgg16_trained_on_2007_trainvaltest_2012_trainval.tgz)[voc\_2012\_test\_results\_fast\_rcnn\_vgg16\_trained\_on\_2012\_trainval.tgz](http://www.cs.berkeley.edu/~rbg/fast-rcnn-data/voc_2012_test_results_fast_rcnn_vgg16_trained_on_2012_trainval.tgz)

4.[Fast R-CNN VGG16 model](http://www.cs.berkeley.edu/~rbg/fast-rcnn-data/voc12_submission.tgz)trained on VOC07 train,val,test union with VOC12 train,val

最后，为什么自己用selective search生成的bounding boxes就再难再现demo里面那么好的效果，而作者目前也没给出回应。