本文以Ubuntu 18.04 64bit为例。CPU-ONLY为例（实际训练环境别考虑用CPU，还是要用CUDA&NVIDIA显卡加成的，CUDA请自行配置，也就是下面有几处小的地方不同而已，基本上CPU的搞定了CUDA的也不难搞）。

我自己折腾时的一些参考资料：

<http://www.cnblogs.com/justinzhang/p/5386837.html>

<https://blog.csdn.net/gaohuazhao/article/details/60871424>

<https://blog.csdn.net/qq_14975217/article/details/51495844>

等等。遇到问题建议先自行Google。

首先习惯开工之前先update&upgrade一波：

sudo apt-get update #更新软件源列表

sudo apt-get dist-upgrade #把所有软件包升级到最新版

安装python (python2)及pip：

sudo apt-get install python python-pip python-numpy

安装一些基本的开发环境（我猜包含编译工具链 gcc make之类的）

sudo apt-get install build-essential cmake

安装一些caffe的依赖组件：

sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libboost-all-dev libhdf5-serial-dev libgflags-dev libgoogle-glog-dev liblmdb-dev protobuf-compiler libopenblas-dev libatlas-base-dev libblas-dev liblapack-dev

安装git：

sudo apt-get install git

下载opencv3.1.0：

wget <https://github.com/opencv/opencv/archive/3.1.0.zip>

然后解压：

unzip 3.1.0.zip

或者访问https://opencv.org/releases.html 下3.1.0版本的source，然后自行解压

（直连可能会比较慢，请自行准备梯子，或者直接解压我的压缩包）

（安利一个东西叫proxychains以及tsocks，这两个是在linux下可以让一些程序 强行走代理，配置方法请自行Google）

Git clone 下载py-faster-rcnn：

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

（同直连可能比较慢，请自备梯子，或直接解压我的压缩包）

进入opencv目录，比如：

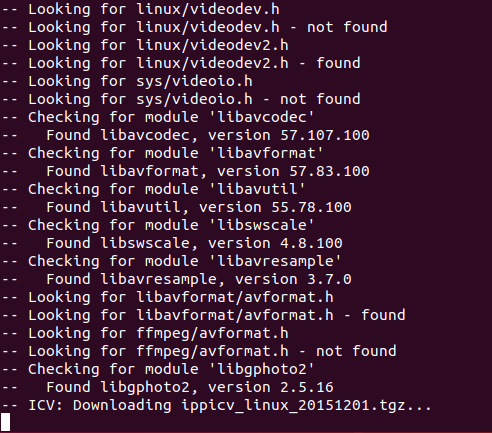
cd opencv-3.1.0

（cd就是进入目录的命令，这个看你自己把opencv放在了哪里）

然后准备编译opencv：

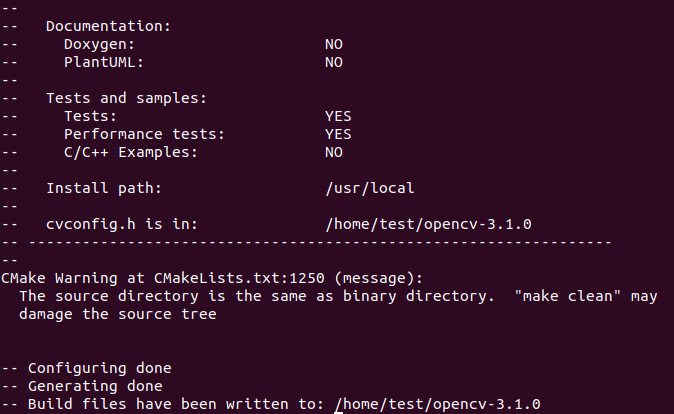
cmake -DBUILD\_TIFF=ON

这个过程如下图所示，会下载东西：



（如果卡在这里太久的话，请检查网络连接，或准备梯子）

像下图这样（没有报错，一切都done）的话，就可以准备编译了



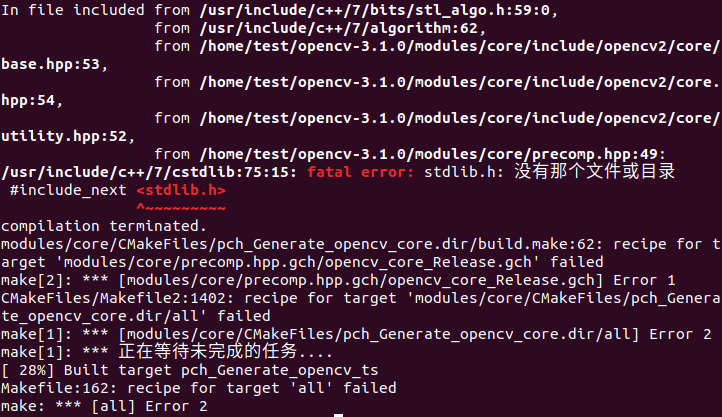
执行：

make

sudo make install

（小tips：make可以开启多线程，比如你的是4核或总共4线程的CPU 可以执行make -j4以提高编译速度，-j几，就是开几个线程，根据自己的CPU核心数、线程数定）

如果出现如下图的错误（stdlib.h）：

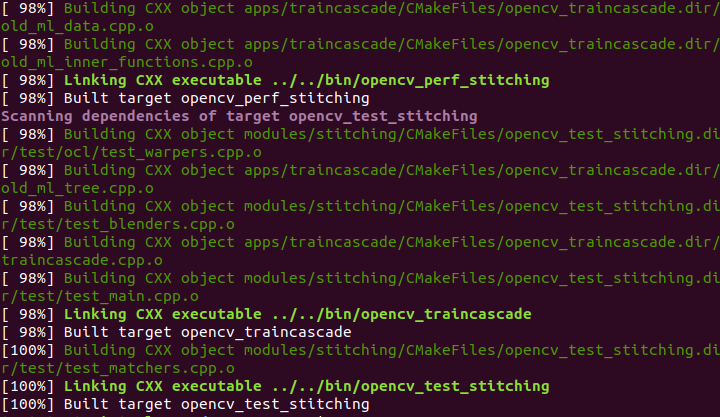


则重新执行cmake：

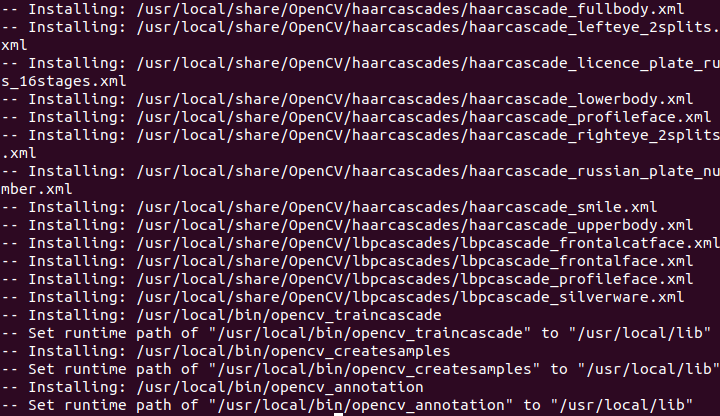
cmake -DBUILD\_TIFF=ON -DENABLE\_PRECOMPILED\_HEADERS=OFF

再重新make

像下图这样100%没报错就可以继续 sudo make install了：



像下图这样 sudo make install没报错的话：



就说明opencv编译并安装好了

接下来就是编译caffe

进入py-faster-rcnn/caffe-fast-rcnn/：

cd py-faster-rcnn/caffe-fast-rcnn/

由example复制一份makefile:

cp Makefile.config.example Makefile.config

修改Makefile.config:

把CPU\_ONLY打开 （把前面的#去掉）（大约在第8行）

**（注意如果用GPU用CUDA的话请不要打开CPU\_ONLY）**



把OPENCV\_VERSION := 3打开（大约在21行）



把WITH\_PYTHON\_LAYER := 1打开（大约在第87行）



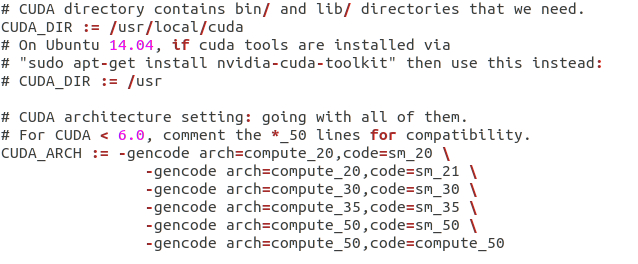
**注：如果使用CUDA（NVIDIA GPU加速）的话，需要配置下面这部分CUDA信息**

**CUDA\_DIR是你的CUDA安装路径，如果是从软件包安装的话试着保持默认即可，如果不行，试着到/usr/local里找找CUDA到底装在哪个文件夹里（有可能是带版本号的文件夹名，比如CUDA-8.0）**

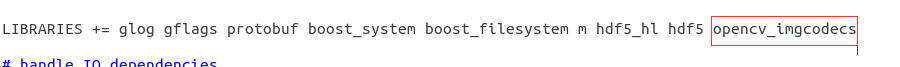
**CUDA\_ARCH与显卡有关，建议直接删掉compute\_20 30那些小的那几行（有可能新的显卡不支持较低的compute等级）。然后依据体你的显卡来写相应的compute。（具体可以查NVIDIA官方，或者直接Google之。）**

**比如我的1050TI就是加上**

**-gencode arch=compute\_61,code=compute\_61**



修改Makefile，大约第181行，LIBRARIES最后加上opencv\_imgcodecs



安装python依赖包

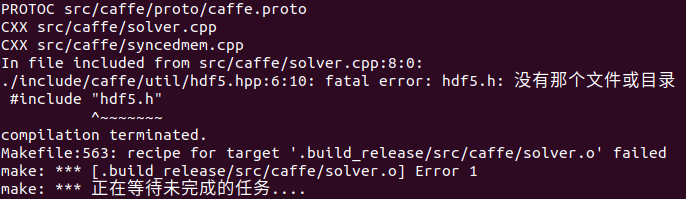
在py-faster-rcnn/caffe-fast-rcnn/python目录中执行

for req in $(cat requirements.txt); do pip install $req; done

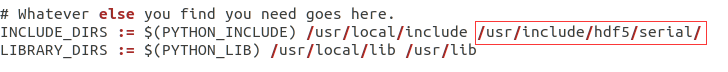
（注意：pip的源可能有点烂，或者撞墙，如果太慢请考虑换源或者搭梯子，pip如何换源请自行Google）

然后就可以在caffe-fast-rcnn这个目录下执行make编译了

注：如果在make caffe-fast-rcnn过程中出现找不到hdf5.h，如下图所示：



则修改Makefile.config，在INCLUDE\_DIRS（约第90行）后面加入/usr/include/hdf5/serial/

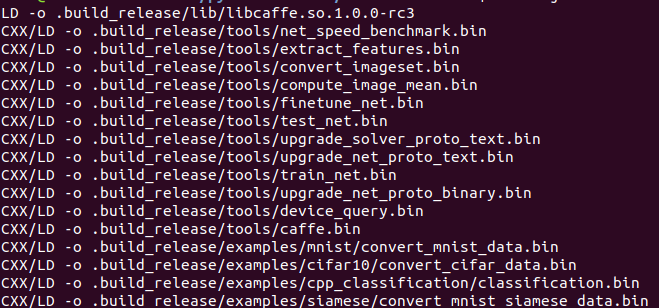


修改Makefile，在大约第181行，把 hdf5\_hl 和hdf5修改为hdf5\_serial\_hl 和 hdf5\_serial



然后继续make试试

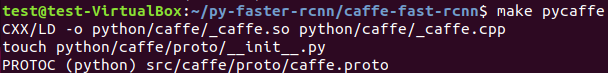
最后像下图这样



就说明caffe 编译好了

然后就可以

make pycaffe



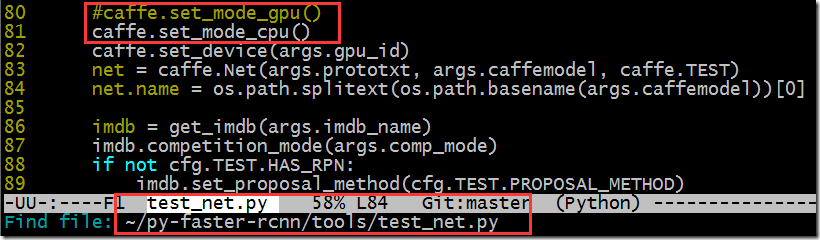
如果是使用CPUONLY（不使用GPU的），还需要做以下更改：

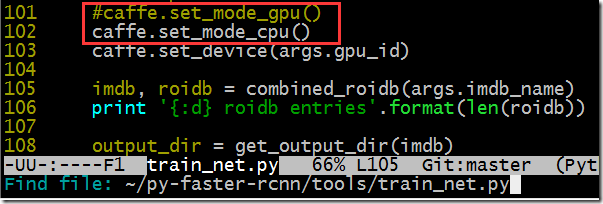
————————————————————————————————

修改py-faster-rcnn/lib/fast\_rcnn/config.py的如下内容：

IMG_256

py-faster-rcnn/tools/test\_net.py和 py-faster-rcnn/tools/train\_net.py的caffe.set\_mode\_gpu()修改为caffe.set\_mode\_cpu()

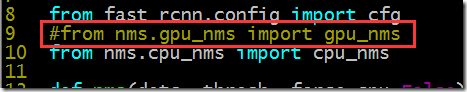




在py-faster-rcnn/lib/setup.py中，含有'nms.gpu\_nms’的部分去掉，去掉后的部分内容如下：



把py-faster-rcnn/lib/fast\_rcnn/nms\_wrapper.py中下图那行注释掉



————————————————————————————————

最后在py-faster-rccn/lib中执行make

然后分别执行py-faster-rcnn/data/scripts中的3个sh脚本下载模型。（这是dropbox的资源，所以需要梯子）

就可以在py-faster-rcnn/tools下执行

python demo.py --cpu

（注：如果是用GPU的，则不用加--cpu）

查看效果。