**REDHAT6.5+CUDA7.5+CUDNN7.0(v3)+Caffe安装教程**

**---非root用户安装**

**前言：**

因为没有root权限，所以网上教程用sudo apt-get install或者yum install 来安装caffe依赖包都不可行，只能下载依赖包源码手动编译安装。且因为原生系统所带编译器和python版本较低，所以在开始之前需先安装高版本的gcc\g++、cmake。且为了方便以后编译pycaffe接口，可下载anaconda 2.7.x并配置python环境，在此推荐使用pyenv来管理不同python版本防止混用。

先列出安装caffe所需的依赖包，之后再一一讲解安装。本人安装配置如下:

REDHAT6.5 + gcc4.7.4(如果此后要编译matcaffe则gcc版本不能超过4.7.x) + cmake2.8.12（3.1版本以上编译gflags时有bug）+ anaconda2.7.12 + protobuf-2.5.0 + Boost\_1\_56\_0 + gflags-2.0(2.1版本的编译caffe时有bug) + glog-0.3.3 + OpenBLAS-0.2.14(也可用默认的atlas或者安装MKL) + hdf5-1.8.9 + OpenCV-2.4.13(2.4.10版本有bug，也可用3.0及以上版本，但可能对于一些早期项目编译时不支持) + lmdb.tgz + leveldb-1.7.0 + snappy-1.1.1 。

(注: 若你有root权限那么接下来的依赖包安装过程就可以直接跳过了，直接执行如下语句即可：

(1) Ubuntu下：

sudo apt-get install git

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 libatlas-base-dev

sudo apt-get install python-dev

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

(2):RHEL/Fedora?CentOS下：

sudo yum install protobuf-devl leveldb-devel snappy-devel opencv-devel boost-devel hdf5-devel atlas-devel

剩下的依赖，对于最新系统可以这样安装:

sudo yum install gflags-devel glog-devel lmdb-devel

最后再跳过后看安装CUDA的步骤再编译caffe即可。）

一：高版本编译器的安装

1.下载gcc-4.7.4.tar.gz，并依次执行如下语句：

tar zxvf gcc-4.7.4.tar.gz

cd gcc-\*

./contrib/download\_prerequisites

安装gcc需要5个组件，download\_prerequisites 的任务就是下载这些组件分别是：

cloog-0.18.1 gmp-4.3.2 isl-0.12.2 mpc-0.8.1 mpfr-2.4.2

如果遇到download\_prerequisites里面的地址无法访问

推荐自行下载这些组件到目录gcc-4.7.4/，解压。

然后将download\_prerequisites里面的wget全部注释掉，再执行 :

./contrib/download\_prerequisites

做好上面的准备就可以configure了，建议另建一个目录来存放编译文件，默认安装目录是 /usr/local/ 可以使用 --prefix 修改自定义路径。(例如我的根目录为/home/th, 先在该目录下建一个caffe\_den文件夹来放置以后装的这些依赖包) 然后执行如下语句:

mkdir build

cd build

../configure –-enable-checking=release --enable-languages=c,c++ --disable-multilib --prefix=/home/yourname/caffe\_den

make –j32

make install

然后在.bashrc中配置库文件和头文件路径，如下:

export LD\_LIBRARY\_PATH=/home/yourname/caffe\_den/lib64/:$LD\_LIBRARY\_PATH

export C\_INCLUDE\_PATH=/home/yourname/caffe\_den/include/:$C\_INCLUDE\_PATH

export CPLUS\_INCLUDE\_PATH=/home/yourname/caffe\_den/include/:$CPLUS\_INCLUDE\_PATH

最后执行source ~/.bashrc再执行gcc --version查看版本是否为4.7.4即可。

二：python环境的安装

1. 使用pyenv。 pyenv 的美好之处在于，他并没有使用将不同的 $PATH 植入不同的 shell 这种高耦合的工作方式，而是简单地在 $PATH 的最前面插入了一个垫片路径（shims）：~/.pyenv/shims:/usr/local/bin:/usr/bin:/bin。所有对 Python 可执行文件的查找都会首先被这个 shims 路径截获，从而架空了后面的系统路径。依次执行如下语句进行安装:

cd

git clone git://github.com/yyuu/pyenv.git .pyenv

echo ‘export PYENV\_ROOT=”$HOME/.pyenv”’>>~/.bash\_profile

echo ‘export PATH=”$PYENV\_ROOT/bin:$PATH”’>>~/.bash\_profile

echo 'eval "$(pyenv init -)"' >> ~/.bash\_profile

source ~/.bash\_profile

执行 pyenv versions可检测到所有python版本，处于激活状态的版本前以\*标示

使用 pyenv install -v 2.7.3 会安装想要的版本到 $PYENV\_ROOT/versions 路径下，

同理可使用pyenv uninstall 2.7.3 来卸载，并且以后每次增删了 Python 版本或带有可执行文件的包（如 pip）以后，都应该执行一次 pyenv rehash 。

设置全局的python版本: pyenv global 3.4.0

设置面向程序的本地版本(优先级比global高)：pyenv local 2.7.3

设置面向shell的python版本（优先级最高）： pyenv shell pypy-2.2.1

通过 pyenv shell --unset 可取消当前shell设定

**2. 安装anaconda2.7.12(推荐)**

上anaconda官网下载 Anaconda2-4.2.0-Linux-x86\_64.sh后执行:

bash ./anaconda2-4.2.0-Linux-x86\_64.sh

之后按命令行提示一路回车，其中是否写入路径需要输入yes,安装路径可手动输入，默认安装在/home/yourname目录下。

然后执行 vi ~/.bashrc 并写入如下语句:

export PATH=/home/th/anaconda2/bin:$PATH

最后执行 source ~/.bashrc 后再执行 pyenv local 2.7.12 ，再执行

python --version 查看版本即可。

（注：若只想用系统自带python则以后编译pycaffe接口时需注意环境变量路径是否正确，且需要按caffe-master/python下的requirements.txt里的要求安装所需的python模块。如在ubuntu下执行如下语句:

sudo apt-get install update

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

sudo apt-get install gfortran

sudo apt-get install GraphViz

sudo pip install pydot

sudo pip install –r ${CAFFE\_ROOT}/python/requirements.txt

三：caffe依赖包的安装

**1. 安装protobuffer**

tar zxvf protobuf-\*.tar.gz

cd protobuf-\*

./configure –prefix=/home/yourname/caffe\_den/

make –j(后面数字依电脑核数而定)

make install

cd python

python setup.py build

python setup.py install

这样就完成了protobuf的安装，为检验是否成功可用 ls ~/caffe\_den/bin/ 查看是否有protoc可执行文件。

再在./bashrc 目录下写入:

export PATH=/home/yourname/caffe\_den/bin/:$PATH

export LD\_LIBRARY\_PATH=/home/yourname/caffe\_den/lib:$LD\_LIBRARY\_PATH

别忘了每次修改./bashrc后需执行 source ~/.bashrc 使之生效。

**2. 安装Boost**

tar jxvf boost\_\*.tar.bz2

cd boost\_\*

./bootstrap.sh

./b2

cp –r boost/ /home/yourname/caffe\_den/include/

cp stage/lib/\* /home/yourname/caffe\_den/lib/

**3. 安装 gflags**

unzip gflags-2.0.zip

cd gflags-2.0/

mkdir build

cd build

cmake ..

ccmake ..

在这里会弹出ccmake配置界面，只需将BUILD\_SHARED\_LIBS改为ON(按Enter键) , CMAKE\_INSTALL\_PREFIX改为/home/yourname/caffe\_den 即可。

改完后先按c 键，再按 g键生成 makefile。再执行: make –j && make install

**4. 安装 glog (需先安装gflags，其他依赖包可并行安装)**

tar zxvf glog-0.3.3.tar.gz

cd glog-\*

./configure –prefix=/home/yourname/caffe\_den/

make –j && make install

**5. 安装 OpenBLAS**

tar zxvf OpenBLAS-\*.tar.gz

cd OpenBLAS-\*/

make –j32

make PREFIX=/home/yourname/caffe\_den install

这里要注意的是在make时可能会报错找不到 –lgfortran.一种原因是系统没安装fortran编译器，另一种极大可能是没找到动态库路径，这时可用

locate gfortran

找到/usr/lib下有libgfortran.so文件，然后可在/.bash\_profile文件中写入如下语句：

export PATH=/usr/bin/:$PATH

export LD\_LIBRARY\_PATH=/usr/lib:/usr/lib64/:$LD\_LIBRARY\_PATH

再 source ~/.bash\_profile ，然后make clean后重新make 即可。

**6. 安装 HDF5**

tar zxvf hdf5-1.8.9.tar.gz

cd hdf5-1.8.9/

./configure –prefix=/home/yourname/caffe\_den/

make –j && make install

**7. 安装 OPENCV**

unzip opencv-2.4.13.zip

cd opencv-\*

mkdir build

cd build

cmake ..

ccmake ..

make

make install

可根据需要自行编译opencv相关模块，因为caffe中用到的opencv中的功能并不多，仅限于图片读写、缩放等CPU上的模块。（注意WITH\_TBB模块在3.0以下版本要选择ON,WITH\_QT如果之前没装qt的话要选择OFF，再把CMAKE\_BUILD\_PREFIX改为/home/yourname/caffe\_den , 其他模块自行选择）。

**8. 安装 LMDB**

tar zxvf lmdb.tgz

cd ./libraries/liblmdb/

cp lmdb.h /home/yourname/caffe\_den/include/

cp liblmdb.so /home/yourname/caffe\_den/lib/

**9. 安装 LEVELDB**

tar zxvf leveldb-1.7.0.tar.gz

cd leveldb-\*

make –j

cp –r include.leveldb /home/yourname/caffe\_den/include/

cp libleveldb.so\* /home/yourname/caffe\_den/lib/

**10. 安装 Snappy**

tar zxvf snappy-1.1.1.tar.gz

cd snappy-1.1.1/

./configure –prefix=/home/yourname/caffe\_den/

make –j && make install

四：cpu版caffe的安装

至此cpu版的caffe依赖包全部安装完成，我们可在/home/yourname/caffe\_den/目录下执行 tree –d –L 2查看下我们安装的成果。

然后就需要在官网上下载最新版的caffe-master并解压至/home/yourname/目录下，再进入caffe-master目录并执行:

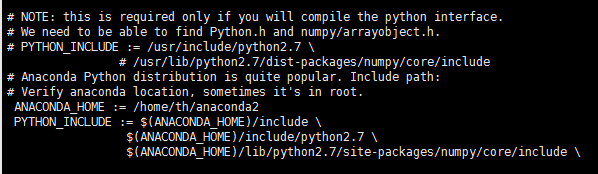
cp Makefile.config.example Makefile.config

再修改Makefile.config如下：

（1）将CPU\_ONLY :=1前的#删除

（2）修改BLAS :=open

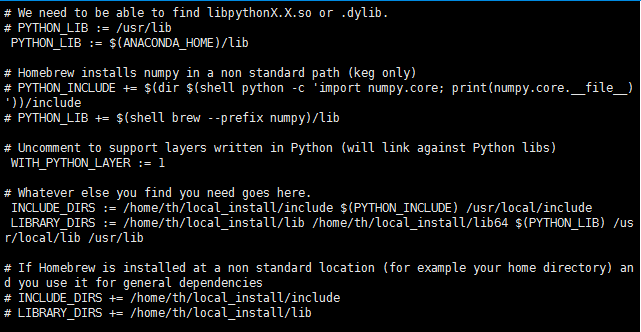
(3) python路径修改如下：



并且其后的PYTHON\_LIB注意要改为 $（ANACONDA\_HOME）/lib .且

WITH\_PYTHON\_LAYER :=1 前面的#去除掉。

最后INCLUDE\_DIRS和LIBRARY\_DIRS修改如下：



最后依次执行

make –j

make pycaffe

make test

make runtest

若没报错则说明CPU版caffe已安装成功。

然后在./bashrc目录下写入：

export

PYTHONPATH=/home/yourname/caffe-master/python:$PYTHONPATH

在python环境下能import caffe则说明pycaffe接口也成功编译了。

五：GPU驱动的安装（该部分需要root权限进行安装，目录在/usr/local）

如果是在自己电脑上安装，推荐使用.deb安装方式，且前面的依赖包也可用root权限进行快速安装（本人在ubuntu14.04系统上已成功安装），但是服务器上为防止出现安装时nvidia driver不匹配导致系统崩溃还是推荐使用.run的安装方式。

请先直接阅读NVIDIA官方安装手册 ***NVIDIA CUDA Installation Guide For Linux/Mac OS X/Windows***，可少踩坑。

1. 创建一个文件: vi /etc/modprobe.d/blacklist-nouveau.conf并写入如下内容：

blacklist nouveau

options nouveau modeset=0

按wq!保存

2. 进入字符界面（例如ubuntu则按ctrl+alt+F1进入tty1），然后输入用户名和密码后再输入如下命令:

sudo service lightdm stop

3. 安装GPU驱动，可在[***http://www.geforce.cn/drivers***](http://www.geforce.cn/drivers)网站找到对应的版本，进入到该文件目录下：

cd /.home/yourname

sudo sh linux\*\*\*.run

安装完后:

sudo service lightdm start

然后sudo reboot重启后再终端输入

cat /proc/driver/nvidia/version 验证驱动安装是否成功。

六：CUDA-7.5的安装

同样在字符界面运行：

sudo chmod +x ./cuda\_7.5\_\*\*\*\*.run

获得可执行权限后再将其解压为三个文件:

sudo sh cuda\*.run –noexec –target

分别为CUDA安装包、NVIDIA驱动、SAMPLE包的.run文件，这里我们就不需要再安装NVIDIA驱动了。

先安装CUDA:

sudo ./cuda-\*\*\*.run(刚解压出来的文件)

完成后在.bashrc下添加CUDA环境变量（分别export PATH和LD\_LIBRART\_PATH,同上即可，别忘了source使之生效！）

继续安装SAMPLE包，完成后进入samples/bin/x86\_64/linux/release并执行:

sudo ./deviceQuery

若出现显卡信息则CUDA安装成功.

七：CUDNN的安装

1. 官网上注册申请下载cudnn-7.5-linux-x64-v5.0-ga.tgz（地址https://developer.nvidia.com/cudnn
2. 到下载目录下执行如下命令：

$ tar -zxvf cudnn-7.5-linux-x64-v5.0-ga.tgz

$ cd cuda/include

$ sudo cp \*.h /usr/local/cuda/include/

$ cd ../lib64

$ sudo cp lib\* /usr/local/cuda/lib64/

$ cd /usr/local/cuda/lib64/

$ sudo chmod +r libcudnn.so.5.0.5

$ sudo ln -sf libcudnn.so.5.0.5 libcudnn.so.5

$ sudo ln -sf libcudnn.so.5 libcudnn.so

$ sudo ldconfig

八：Caffe的MATLAB接口配置（可选）

1. 下载matlab R2014a的iso镜像及破解文件（地址：http://pan.baidu.com/s/1qYJ9tNm#list/path=%2F）

2. 下载完成后将iso文件挂载到Linux

sudo mkdir /media/matlab

mount -o loop [path][filename].iso /media/matlab

cd /media/matlab

sudo ./install

3. 安装过程中使用readme.txt中的序列号

4. 破解：

(1) 安装完成后使用crack下的 license进行激活；

(2)将crack文件夹下的libmwservices.so copy到 /usr/local/MATLAB/R2014A/bin/glnxa64。

五、完成安装，命令行下使用sudo matlab即可启动使用。

九：Caffe的GPU版重编译

重新进入caffe-master并修改Makefile.config文件，将CPU\_ONLY:=1前的#加上，查看cuda路径是否正确，python接口和matlab接口可选，若要用则去除对应路径前的#并检查路径是否正确，然后执行：

make clean

make –j

make pycaffe(可选)

make matcaffe（可选）

十：总结

在此要多说一句的是，即使你完全按照我如上步骤安装也极有可能在中途遇到一些小错误，因为每个人的系统版本、依赖包版本都可能不一样导致编译时出bug，但只要你细心并善于查出错误和搜索解决方案定能成功编译caffe!接下来就运行个mnist的小示例练练手吧~祝好运！

遇到的问题：手动安装hdf5-1.8.9后编译caffe时报错：头文件和库文件版本不匹配，这是因为安装的anaconda的库中已有hdf5的1.8.17版本,此时只需conda uninstall hdf5后重新编译即可。若出现protobuf和protoc版本不匹配可能anaconda中已有protobuf，删除即可。