Xin-Yu Ou（欧新宇）
教育的根是苦的，但是其果是甜的。



Caffe + Ubuntu 15.04 + CUDA 7.0 新手安装配置指南

返回

特别感谢在学习和使用Caffe和CNN的过程中，超级大神Yanqing Jia, Eric Tzeng, Evan Shelhamer, Ross Girshick, Sergey Karayev, Sergio Gadarrama给予的帮助。

特别说明：

0. Caffe 官网地址：http://caffe.berkeleyvision.org/

1. 本文为作者亲自实验完成，但仅限用于学术交流使用，使用本指南造成的任何不良后果由使用者自行承担，与本文作者无关，谢谢！为保证及时更新，转载请标明出处，谢谢！

2. 本文旨在为新手提供一个参考，请高手勿要吐槽，有暴力倾向者，请绕道，谢谢！

3. 本文使用2015年5月5日下载的caffe-master版本，运行平台为：Ubuntu 15.04，CUDA7.0，cuDNN v2(以前一直是cuDNN R1)，OpenCV 3.0.0rc1。

4. 安装过程，因为平台不同、设备不同、操作者不同，会遇到各种奇怪的问题和报错信息，请善用Caffe官网的Issues和caffe-user论坛,以及Google和Baidu。参考本指南，请下载最新版caffe-master，新版本很多文件已经变更。

5. 最后更新时间：2015年5月6日。本次更新主要是在新版本的软件进行部署，并修正了过去的一些bug，保留了原来14.04下安装的部分步骤。

洋洋洒洒一大篇，就没截图了，经过几个月的使用，使用Caffe还是建议在Ubuntu系统下完成，因为不需要太多的编程，Windows版需要一定的编程基础，且因为没有官方Release版，所以更新和FixBug都麻烦一些.这里有个建议是，如果将来要做大数据集，最好事先给Linux留多点空间，比如Imagenet，估计500G都不为过。不过也可以全部使用软链接链接到Windows的NTFS磁盘，我后期实验都是使用这个方法。另外，请阅读完，至少一个部分再进行动手操作，避免多余的工作，写作能力有限，尽请见谅。

新版的各种软件，安装起来还是有一定的困难的，所以请大家使用的时候要有耐心。不过Ubuntu15.04相对于14.04有两个进步，一是窗口不会在无聊的闪烁了（也可能是驱动问题）；二是访问网页时，不会再半天都无法响应。

这篇安装指南，适合零基础，新手操作，请高手勿要吐槽！

简单介绍一下：Caffe，一种Convolutional Neural Network的工具包，和Alex的cuda-convnet功能类似，但各有特点。都是使用C++ CUDA进行底层编辑，Python进行实现，原作主要部署于Ubuntu，也有大神发布了Windows版，但其他相关资料较少，不适合新手使用，所以还是Ubuntu的比较适合新手。（相对而言）

本文主要包含5个部分，包括：

- 第一部分 **Linux**安装
- 第二部分 **nVidia**驱动和**CUDA Toolkit**的安装和调试（***.deb**方法，特别推荐）
- 第二部分 **nVidia**驱动和**CUDA Toolkit**的安装和调试（***.run**方法）
- 第三部分 **Python**安装和调试
- 第四部分 **Matlab**安装和调试
- 第五部分 **Caffe**的安装和测试

第一部分 Linux安装

Linux的安装，如果不是Linux粉，只是必须，被迫要用它来作作科研什么的，建议安装成双系统，网上方法很多，这里我就不详细写了，安装还算是傻瓜

式的，和windows的过程类似，至于语言，如果觉得难度还不够大的话，完全可以装E文版的，甚至日文，德文~~~，我是装的简体中文版，我总共用分出的500G的空间来安装Ubuntu 14.04，这个版本是最新的版本，有个好处是，可以直接访问Windows8.1的NTFS分区，不用做额外的操作，而且支持中文，例如：$ cd /media/yourname/分区名字/文件夹名，当然GUI就更方便了.

我的分区设置如下：

根分区：\ 100G，

Swap交换分区：16G ，这里，我设置和我的内存一样，据说小于16G的内存，就设置成内存的1.5-2倍

boot分区：200M

Home分区：剩余的空间，鉴于Imagenet，PASCAL VOC之类的大客户，建议500G，至少300G以上。

PS：解决启动分区错误

基本上，重装起来，都会破坏原来的启动分区表，还原Windows分区的一个简单办法：

$ sudo gedit etc/default/grub

设置：GRUB_DEFAULT = 2

$ sudo update-grub

该方法适用于安装双系统后，"看得到Linux，看不到Windows"的情况，反过来的话，请大家自己百度吧。

PS：关于我的笔记本的特例，仅供类似设备的参考

笔记本配置：技嘉P35X v3，i7-4720HQ@2.6G/16G/NVidia GTX 980 4G/Intel HD 4600/128G SSD*2 + 2T SATA *2

我的两组硬盘SSD和SATA分别做Raid 0，目的是合并逻辑分区，没有考虑冗余备份问题，最后的状态是2个逻辑硬盘块256G SSD + 4T SATA，用的GPA分区，最后导致利用Ubuntu的GRUB启动界面找不到Windows分区。所以上面的方法失效。不过，可以通过笔记本的F12和Bios设置来实现启动分区的选择，并且我用Linux的机会很少，所以也就如此处理了，Linux高手可以自己折腾一下Grub启动。Ubuntu 15.04安装在SATA逻辑分区，SSD分区安装Windows8.1。

第二部分两种安装方法，任选其一即可，推荐第二种方法**(*.deb方法)**。

第二部分：nVidia驱动和CUDA Toolkit的安装和调试（*.deb方法）

PS：特别推荐*.deb的方法，目前已提供离线版的deb文件，该方法比较简单，不需要切换到tty模式。这里以CUDA 7.0为例。

一、CUDA Repository

获取CUDA安装包,安装包请自行去NVidia官网下载。

$ sudo dpkg -i cuda-repo-ubuntu1410-7-0-local_7.0-28_amd64.deb

$ sudo apt-get update

二、CUDA Toolkit

$ sudo apt-get install -y cuda

三、Environment Variables

$ export CUDA_HOME=/usr/local/cuda-7.0

$ export LD_LIBRARY_PATH=${CUDA_HOME}/lib64

$ PATH=${CUDA_HOME}/bin:${PATH}

$ export PATH

第二部分：nVidia驱动和CUDA Toolkit的安装和调试（*.run方法）

PS：这里其实可以参考nVidia 官方提供的CUDA安装手册，非常相近，32页的，不过是全英文的，我就是参考这个文档完成后面的配置和验证工作。https://developer.nvidia.com/rdp/cuda-65-rc-toolkit-download#linux。一般要输入你的用户名和密码，就是下载6.5的那个账号。

一、Verify You Have a CUDA-Capable GPU

执行下面的操作，然后验证硬件支持GPU CUDA，只要型号存在于https://developer.nvidia.com/cuda-gpus，就没问题了

$ lspci | grep -i nvidia

二、Verify You Have a Supported Version of Linux

$ uname -m && cat /etc/*release

重点是“x86_64”这一项，保证是x86架构，64bit系统

三、Verify the System Has gcc Installed

$ gcc --version

没有的话就先安装吧，这个是必须的用来编译CUDA Toolkit，不过Ubuntu 14.04是默认有的

四、Download the NVIDIA CUDA Toolkit

下载地址：https://developer.nvidia.com/cuda-toolkit

验证地址：https://developer.nvidia.com/rdp/cuda-rc-checksums

$ md5sum filename

例如：md5sum cuda_6.5.11_rc_linux_64.run ，这个文件的正确 md5 = a47b0be83dea0323fab24ca642346351

这个感觉蛮重要，我第一次安装的时候md5就没通过，强制安装，结果就有问题，后面重新下载了再安装了一次

五、Handle Conflicting Installation Methods

根据官网介绍，之前安装的版本都会有冲突的嫌疑

所以，之前安装的Toolkit和Drievers就得卸载，屏蔽，等等

六、Graphical Interface Shutdown

退出GUI，也就是X-Win界面，操作方法是：同时按：CTRL+ALT+F1（F2-F6），切换到TTY1-6命令行模式。

关闭桌面服务：

$ sudo stop lightdm

七、Interaction with Nouveau

Nouveau是一个开源的显卡驱动，Ubuntu 14.04 默认安装了，但是它会影响nVidia驱动的安装，所以只有请他回老家了，sorry!

$ sudo vi /etc/modprobe.d/nvidia-graphics-drivers.conf

写入：blacklist nouveau

保存并退出: wq!

检查：**$ cat nvidia-graphics-drivers.conf**

$ sudo vi /etc/default/grub

末尾写入：rdblacklist=nouveau nouveau.modeset=0

保存并退出: wq!

检查：**$ cat /etc/default/grub**

八、**Installation CUDA 6.5**

切换到cuda_6.5.11_rc_linux_64.run 所在的目录，然后执行安装命令：

$ sudo sh cuda_6.5.11_rc_linux_64.run

再次提醒，安装前一定要执行 md5sum ，至于如果发现md5检测不一致，怎么办？别逗了，去nVidia重新下载就行了，地球人都知道，别无限循环就好 ^_^!

这里会一路问你各种问题，基本上就是Accept-yes-Enter-yes-Enter-yes-Enter， 其实就是让你接受协议，然后安装的默认位置确认等等，recruit就别自定义安装位置了，默认才是天堂。

九、**Extra Libraries**

安装一些必要的库文件，譬如：OpenGL (e.g., Mesa), GLU, GLUT, and X11 (including Xi, Xmu, and GLX).

$ sudo apt-get install freeglut3-dev build-essential libx11-dev libxmu-dev libxi-dev libgl1-mesa-glx libglu1-mesa libglu1-mesa-dev

十、驱动装完了，可以回到**GUI**界面了，也可以继续留这里玩文本。。。

$ sudo start lightdm

十一、**POST-INSTALLATION ACTIONS**

1. Environment Setup

$ export PATH=/usr/local/cuda-6.5/bin:$PATH

$ export LD_LIBRARY_PATH=/usr/local/cuda-6.5/lib64:$LD_LIBRARY_PATH

PS：如果出现安装失败，重启系统，重新安装一遍基本都可以解决，实在不行就卸载原来的驱动再安装一遍。

a. 卸载现有驱动

$ sudo nvidia-installer --uninstall

b. 重装CUDA Toolkit

$ sudo sh cuda_6.5.11_rc_linux_64.run

好了，到这里所有nVidia CUDA的安装就结束了，下面看看Caffe如何安装

第三部分 Python安装和调试

1. 安装**IDE**运行环境

选择一个适合你的IDE运行环境，我是用的是Spyder，因为它内置了 iPython 环境，Caffe有不少的程序是基于 iPython 环境完成的。安装方法很简单，直接在Ubuntu软件中心搜索“spyder”即可安装。

2. iPython NoteBook 安装

另外一个比较推荐的方法是使用iPyhthon NoteBook（基于浏览器的Python IDE），特别是适合需要用Python做教程的老师们，可以直接导出.py, .ipynb, html格式，安装步骤如下：

$ sudo apt-get install -y ipython-notebook pandoc

启动(自动打开浏览器):

$ ipython nootbook

一个简单的使用iPython NoteBook生成的html的例子：examples_notebook.html example_notebook.ipynb

3. 配置和编译**pycaffe**（见第五部分）

第四部分 Matlab安装和调试

1.下载

由于该软件为商业软件，请大家自行寻找，安装学习，并确保不使用做商业目的，下载24小时删除.....

2. 预准备

选择Mathworks.Matlab.R2014a.Unix.iso - 右键 - 使用磁盘映像挂载器打开”

进入装载的虚拟光盘，拷贝全部文件至home/Matlab 文件夹

(PS：我的原则是能GUI就GUI，喜欢CMD的可以参照执行)

复制Crack/install.jar至 home/Matlab/java/jar/ 并覆盖源文件

$ sudo cp install.jar /home/Matlab/java/jar/

3. 授权安装文件夹

$ chmod a+x Matlab -R

4. 安装

$ sudo ./install

选项：不使用Internet安装

序列号： 12345-67890-12345-67890

默认路径：/usr/local/MATLAB/R2014a

激活文件：license_405329_R2014a.lic

拷贝 libmwservices.so 至 /usr/local/MATLAB/R2014a/bin/glnxa64

$ sudo cp libmwservices.so /usr/local/MATLAB/R2014a/bin/glnxa64/

安装完毕，程序默认启动路径：

$ sh /usr/local/MATLAB/R2014a/bin/matlab

5.解决编译器**gcc/g++**版本问题。（这里因为折腾了一会，所以只做参考，基本流程就**2**步，有问题，大家可以自己尝试。）

因为Ubuntu 15.04的gcc/g++版本是4.9.2，而Matlab 2014a（2015a）的版本是4.7.x所以在使用matla调用mex文件的时候，基本上都会报错，根据报错信息，考虑如下两步解决方案。

1. 降级安装**gcc/g++**版本为**4.7.x**

(1). 下载gcc/g++ 4.7.x

$ sudo apt-get install -y gcc-4.7

$ sudo apt-get install -y g++-4.7

(2). 链接gcc/g++实现降级

$ cd /usr/bin

$ sudo rm gcc

$ sudo ln -s gcc-4.7 gcc

$ sudo rm g++

$ sudo ln -s g++-4.7 g++

2. 暴力引用新版本GLIBCXX_3.4.20

$ sudo cp /usr/lib/x86_64-linux-gnu/libstdc++.so.6.0.20 /usr/local/MATLAB/R2014a/sys/os/glnxa64/libstdc++.so.6.0.20

$ sudo mv libstdc++.so.6 libstdc++.so.6.backup

$ sudo ln -s libstdc++.so.6.0.20 libstdc++.so.6

$ sudo ldconfig -v

通过命令“strings /usr/local/MATLAB/R2014a/sys/os/glnxa64/libstdc++.so.6 | grep GLIBCXX_” 可以看一下，是否已经成功包含了GLIBCXX_3.4.20，如果已经存在，基本上就成功了。

6.编译Matlab用到的caffe文件（见第五部分）

第五部分 Caffe的安装和测试

对于Caffe的安装严格遵照官网的要求来：http://caffe.berkeleyvision.org/installation.html

一、安装BLAS

这里可以选择（ATLAS，MKL或者OpenBLAS），我这里使用MKL，首先下载并安装英特尔® 数学内核库 Linux* 版MKL，下载链接是：https://software.intel.com/en-us/intel-education-offerings， 请下载Student版，先申请，然后会立马收到一个邮件（里面有安装序列号），打开照着下载就行了。下载完之后，要把文件解压到home文件夹(或直接把tar.gz文件拷贝到home文件夹,为了节省空间，安装完记得把压缩文件给删除喔～)，或者其他的ext4的文件系统中。

接下来是安装过程，先授权，然后安装：

$ tar zxvf parallel_studio_xe_2015.tar.gz （如果你是直接拷贝压缩文件过来的）

$ chmod a+x parallel_studio_xe_2015 -R

$ sudo ./install_GUI.sh

二、MKL与CUDA的环境设置

1. 新建intel_mkl.conf， 并编辑之：

$ sudo gedit /etc/ld.so.conf.d/intel_mkl.conf

/opt/intel/lib/intel64

/opt/intel/mkl/lib/intel64

2. 新建cuda.conf，并编辑之：

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

/usr/local/cuda/lib64

/lib

3. 完成lib文件的链接操作，执行：

$ sudo ldconfig -v

三、安装OpenCV 3.0.0

1. 下载并编译OpenCV（官网原版OpenCV：http://opencv.org/）， 或者使用本站提供的修改版的安装包 Install-OpenCV-master （下面的安装方式使用

该包完成，安装包修改了dependencies.sh文件并增加了OpenCV 3.0.0的安装文件，同时保留了原来的2.3x和2.4x版）

2. 切换到文件保存的文件夹，然后安装依赖项：

$ sudo sh Ubuntu/dependencies.sh

3. 切换目录Ubuntu\3.0\安装OpenCV 3.0.0rc1：

$ sudo sh opencv3_0_0-rc1.sh

保证网络畅通，因为软件需要联网这里时间较长，请耐心等待。。。，

四、安装其他依赖项

1. Google Logging Library（glog），下载地址：https://code.google.com/p/google-glog/，然后解压安装：

$ tar zxvf glog-0.3.3.tar.gz

$ ./configure

$ make

$ sudo make install

如果没有权限就chmod a+x glog-0.3.3 -R，或者索性 chmod 777 glog-0.3.3 -R，装完之后，这个文件夹就可以kill了。

2. 其他依赖项，确保都成功

$ sudo apt-get install -y libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libboost-all-dev libhdf5-serial-dev

$ sudo apt-get install -y libgflags-dev libgoogle-glog-dev liblmdb-dev protobuf-compiler

五、安装Caffe并测试

1. 安装pycaffe必须的一些依赖项：

$ sudo apt-get install -y python-numpy python-scipy python-matplotlib python-sklearn python-skimage python-h5py python-protobuf python-leveldb python-networkx python-nose python-pandas python-gflags Cython ipython

$ sudo apt-get install -y protobuf-c-compiler protobuf-compiler

2. 安装配置nVidia cuDNN 加速Caffe模型运算

a. 安装cuDNN

该改版本caffe-master默认支持cudnn-6.5-linux-x64-v2，使用cudnn-6.5-linux-R1会报错，安装前请去先官网下载最新的cuDNN。

$ sudo cp cudnn.h /usr/local/include

$ sudo cp libcudnn.so /usr/local/lib

$ sudo cp libcudnn.so.6.5 /usr/local/lib

$ sudo cp libcudnn.so.6.5.48 /usr/local/lib

b. 链接cuDNN的库文件

$ sudo ln -sf /usr/local/lib/libcudnn.so.6.5.48 /usr/local/lib/libcudnn.so.6.5

$ sudo ln -sf /usr/local/lib/libcudnn.so.6.5 /usr/local/lib/libcudnn.so

$ sudo ldconfig -v

3. 切换到Caffe-master的文件夹，生成Makefile.config配置文件，执行：

$ cp Makefile.config.example Makefile.config

4. 配置Makefile.config文件（仅列出修改部分）

a. 启用CUDNN，去掉"#"（目前caffe-master仍然只支持R1版本）

```
USE_CUDNN := 1
```

b. 启用GPU，添加注释"#"

```
# CPU_ONLY := 1
```

c. 配置一些引用文件（增加部分主要是解决新版本下，HDF5的路径问题）

```
INCLUDE_DIRS := $(PYTHON_INCLUDE) /usr/local/include /usr/lib/x86_64-linux-gnu/hdf5/serial/include

LIBRARY_DIRS := $(PYTHON_LIB) /usr/local/lib /usr/lib /usr/lib/x86_64-linux-gnu/hdf5/serial
```

d. 启用Intel Parallel Studio XE 2015 Professional Edition for C++ Linux

```
BLAS := mkl
```

e. 配置路径，实现caffe对Python和Matlab接口的支持

```
PYTHON_LIB := /usr/local/lib

MATLAB_DIR := /usr/local/MATLAB/R2014a
```

5. 配置Makefile文件（实现对OpenCV 3.x的支持）

查找“Derive include and lib directories”一节，修改“LIBRARIES +=”的最后一行，增加opencv_imgcodecs

```
opencv_core opencv_highgui opencv_imgproc opencv_imgcodecs
```

6. 编译caffe-master！！！"-j8"是使用CPU的多核进行编译，可以极大地加速编译的速度，建议使用。

```
$ make all -j8

$ make test -j8

$ make runtest -j8
```

编译**Python**和**Matlab**用到的**caffe**文件

```
$ make pycaffe -j8

$ make matcaffe -j8
```

六、使用**MNIST**数据集进行测试

Caffe默认情况会安装在$CAFFE_ROOT，就是解压到那个目录，例如：$ home/username/caffe-master，所以下面的工作，默认已经切换到了该工作目录。下面的工作主要是，用于测试Caffe是否工作正常，不做详细评估。具体设置请参考官网：
http://caffe.berkeleyvision.org/gathered/examples/mnist.html

1. 数据预处理

```
$ sh data/mnist/get_mnist.sh
```

2. 重建lmdb文件。Caffe支持三种数据格式输入网络，包括Image(.jpg, .png等)，leveldb，lmdb，根据自己需要选择不同输入吧。

```
$ sh examples/mnist/create_mnist.sh
```

生成mnist-train-lmdb 和 mnist-train-lmdb文件夹，这里包含了lmdb格式的数据集

3. 训练mnist

```
$ sh examples/mnist/train_lenet.sh
```

至此，Caffe安装的所有步骤完结，下面是一组简单的数据对比，实验来源于MNIST数据集，主要是考察一下不同系统下CPU和GPU的性能。可以看到明显的差别了，虽然MNIST数据集很简单，相信复杂得数据集，差别会更大，Ubuntu+GPU是唯一的选择了。

测试平台1：i7-4770K/16G/GTX 770/CUDA 6.5

MNIST Windows8.1 on CPU：620s

MNIST Windows8.1 on GPU：190s

MNIST Ubuntu 14.04 on CPU：270s

MNIST Ubuntu 14.04 on GPU：160s

MNIST Ubuntu 14.04 on GPU with cuDNN：30s

Cifar10_full on GPU wihtout cuDNN：73m45s = 4428s （*Iteration 70000*）

Cifar10_full on GPU with cuDNN：20m7s = 1207s （*Iteration 70000*）

测试平台2：技嘉P35X v3，i7-4720HQ@2.6G/16G/NVidia GTX 980 4G

MNIST Ubuntu 15.04 on GPU with cuDNN：33s

对比测试1：2*E5-2620(12CPUs)/128G/Tesla K20M/CUDA5.5/CentOS 6.4

MNIST CentOS 6.4 on GPU：294s

对比测试2：Tesla K40M/CUDA6.5/ubuntu 14.04

MNIST on GPU with cuDNN：30s

对比测试3：GTX 660/CUDA6.5/ubuntu 14.04

MNIST on GPU with cuDNN：49s

对比试验1是一个不太公平的测试，毕竟性能差很大，很可能不单单是由Tesla K20s 和GTX 770带来的，也可能是因为CentOS或者是CUDA5.5(without cuDNN)的影响，但总体上的结论和Caffe官网的 reference performance numbers 一致，对于普通用户：GTX的性价比高很多。对比试验2展现了Tesla K40的强大性能，相信对于复杂图像，它应该有更强劲的表现。（感谢香港城市大学 Ph.D Jingjing、南京理工大学 Ph.D JinLu、华中科技大学 MS LiuMaolin 提供的测试环境和测试数据。）

返回