Ubuntu16.04+cuda8+caffe安装

**电脑配置**

系统：Ubuntu16.04

GPU:NVIDIA GTX 750 Ti

安装内容：

NVIDIA驱动：375.20

CUDA：8.0.

cuDNN：5.1

Opencv：3.1

python

**安装过程**

**1.安装相关依赖项**

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

**2.安装NVIDIA驱动**

（1）查询NVIDIA驱动

首先去官网(http://www.nvidia.com/Download/index.aspx?lang=en-us)查看适合自己显卡的驱动：

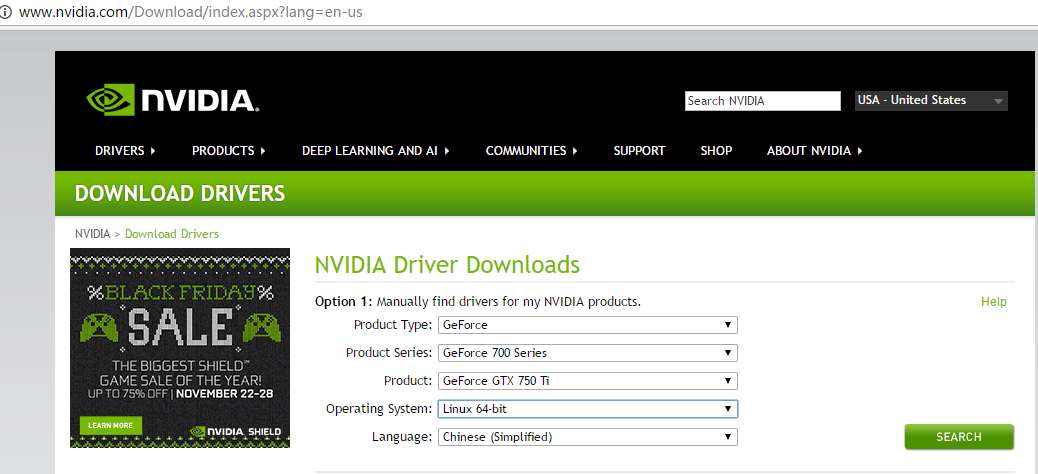


图1.显卡驱动查询

例如本人电脑的显卡驱动如下图：

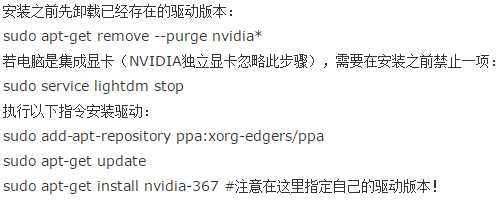


图2.显卡驱动版本

（2）安装驱动

照如下方法并没有成功，因为总是sudo apt-get update失败：

=================================



========================================

换用另一种方法：

直接在ubuntu系统设置，软件和更新里面，选择中国的服务器源刷新之后，点击附加驱动选项，在Nvidia Corporation选择361.42(强迫症必须安装最新的)，然后点击应用更改，下载安装完之后重启

安装完成之后输入以下指令进行验证：

sudo nvidia-smi

若列出了GPU的信息列表则表示驱动安装成功。

**3.安装CUDA**

CUDA是NVIDIA的编程语言平台，想使用GPU就必须要使用cuda。

（1）下载CUDA

首先在官网上(https://developer.nvidia.com/cuda-downloads)下载CUDA：

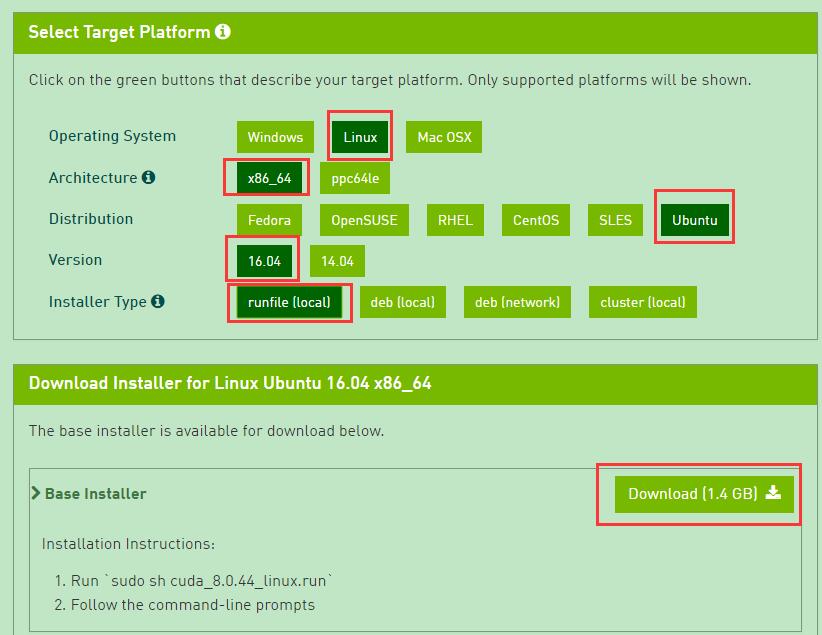


图3.CUDA下载

（2）安装CUDA

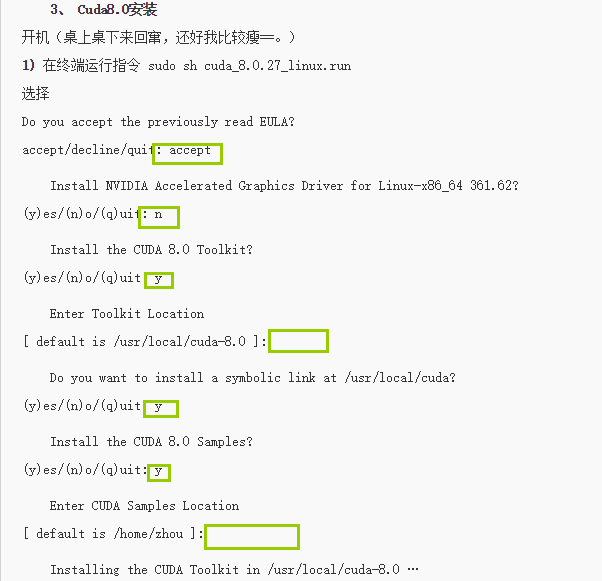
下载完成后执行以下命令：

sudo sh cuda\_8.0.44\_linux.run

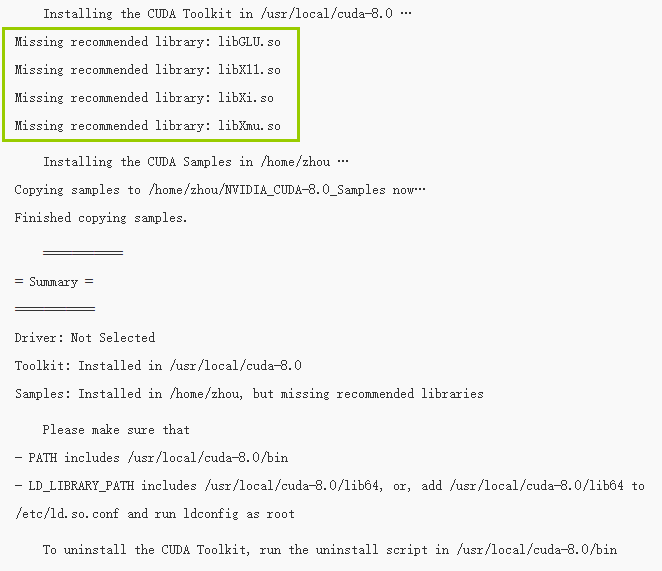
注意：执行后会有一系列提示让你确认，但是注意，有个让你选择是否安装nvidia361驱动时，一定要选择否：

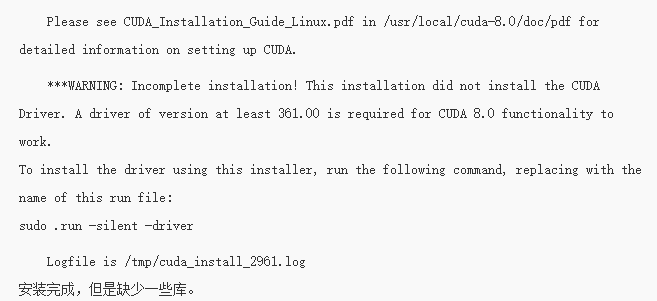
Install NVIDIA Accelerated Graphics Driver for Linux-x86\_64 361.62?

因为前面我们已经安装了更加新的nvidia367，所以这里不要选择安装。其余的都直接默认或者选择是即可。



可能会出现这样的问题：





（3）提示缺少一些库

就安装所缺少的库

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

这个因为网络问题，可能要安装很久。  
还是有线好点

（4）然后安装vim

sudo apt-get install vim

安装完成

（5）设置环境变量

打开~/.bashrc文件：

sudo vim ~/.bashrc

将以下内容写入到~/.bashrc尾部：

export PATH=/usr/local/cuda-8.0/bin${PATH:+:${PATH}}

export LD\_LIBRARY\_PATH=/usr/local/cuda-8.0/lib64${LD\_LIBRARY\_PATH:+:${LD\_LIBRARY\_PATH}}

保存退出（按Esc，输入命令:Wq,表示保存退出）

然后刷新

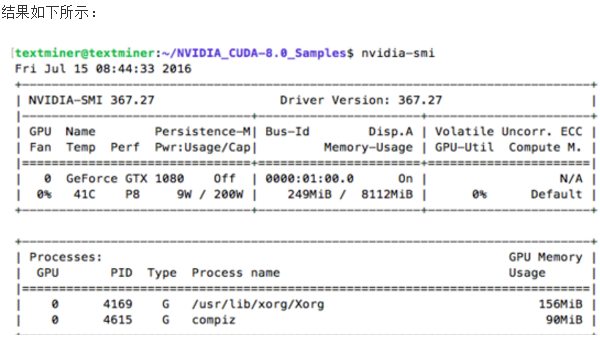
输入：



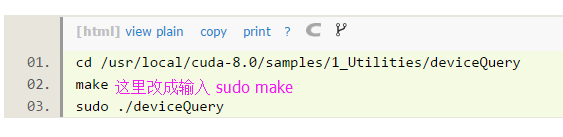
（6）测试cuda

命令行回到最初文件夹，运行





测试cuda的sample

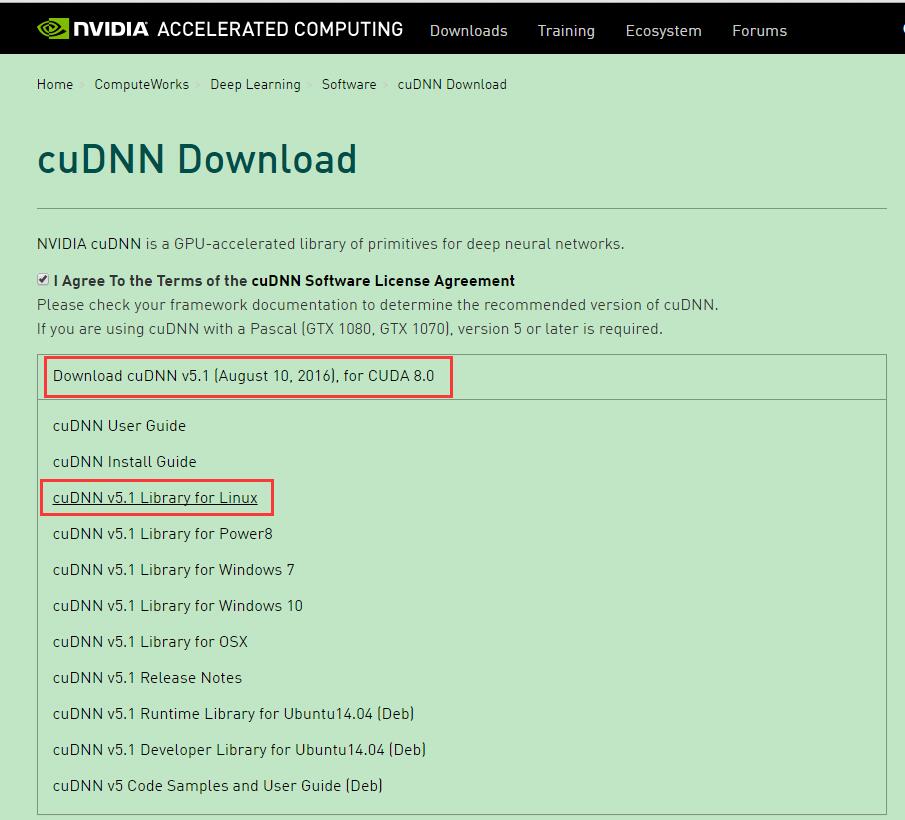


如果显示的是一些关于GPU的信息，则说明安装成功了。(最后显示是ＰＡＳＳ不是Faile!)

**4.配置cuDNN**

cuDNN是GPU加速计算深层神经网络的库。

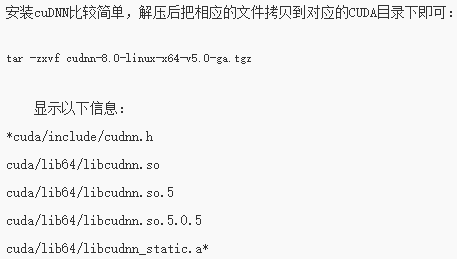
首先去官网(https://developer.nvidia.com/rdp/cudnn-download)下载cuDNN,可能需要注册一个账号才能下载。



安装cuDNN比较简单，解压后把相应的文件拷贝到对应的CUDA目录下即可：

解压：

tar –zxvf cudnn-8.0-linux-x64-v5.1.tgz



cd进入cuDNN5.1解压之后的include目录：

cd cuda/include

在命令行进行如下操作

sudo cp cudnn.h /usr/local/cuda/include #复制头文件

再将cd进入lib64目录下的动态文件进行复制和链接：



sudo cp lib\* /usr/local/cuda/lib64/ #复制动态链接库

cd /usr/local/cuda/lib64/

（以下命令要根据不同版本的cudnn进行修改，不然后面caffe安装会有很多这边的问题）

sudo rm -rf libcudnn.so libcudnn.so.5 #删除原有动态文件

sudo chmod u=rwx,g=rx,o=rx libcudnn.so.5.1.5 #修改文件权限

sudo ln -s libcudnn.so.5.1.5 libcudnn.so.5 #生成软衔接

sudo ln -s libcudnn.so.5 libcudnn.so #生成软链接

更新链接库：

**sudo ldconfig**



**5.安装opencv3.1**

从官网(http://opencv.org/downloads.html)下载Opencv,并将其解压到你要安装的位置，假设解压到了/home/用户名/opencv（如我是 /home/vision/opencv）。

安装前准备，创建编译文件夹：

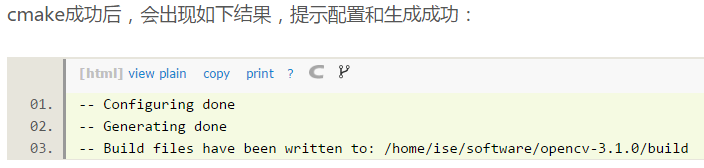
cd ~/opencv （我是cd到了 /home/vision/opencv/opencv-3.1.0）

mkdir build

cd build

配置：

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





#-j8表示并行计算，根据自己电脑的配置进行设置，配置比较低的电脑可以将数字改小或不使用，直接输make。  
可能出现下面的报错，如下图改过来就好：



以上只是将opencv编译成功，还没将opencv安装，需要运行下面指令进行安装：

sudo make install

**6 配置Python环境**

（1）将终端cd到要安装caffe的位置。

（2）从github上获取caffe：

git clone https://github.com/BVLC/caffe.git

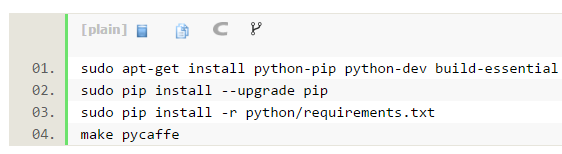
注意：若没有安装Git，需要先安装Git：

sudo apt-get install git

进入python文件夹：

cd python

在终端执行下面的命令：





**8.配置caffe**

（1）将终端cd到要安装caffe的位置。

（2）从github上获取caffe：

git clone https://github.com/BVLC/caffe.git

注意：若没有安装Git，需要先安装Git：

sudo apt-get install git

（3）因为make指令只能make Makefile.config文件，而Makefile.config.example是caffe给出的makefile例子，因此，首先将Makefile.config.example的内容复制到Makefile.config：

sudo cp Makefile.config.example Makefile.config

（4）打开并修改配置文件：

sudo gedit Makefile.config #打开Makefile.config文件

根据个人情况修改文件：

a.若使用cudnn，则

将

#USE\_CUDNN := 1

修改成：

USE\_CUDNN := 1

b.若使用的opencv版本是3的，则

将

#OPENCV\_VERSION := 3

修改为：

OPENCV\_VERSION := 3

c.若要使用python来编写layer，则

将

#WITH\_PYTHON\_LAYER := 1

修改为

WITH\_PYTHON\_LAYER := 1

d.重要的一项 :

将# Whatever else you find you need goes here.下面的

INCLUDE\_DIRS := $(PYTHON\_INCLUDE) /usr/local/include

LIBRARY\_DIRS := $(PYTHON\_LIB) /usr/local/lib /usr/lib

修改为：

INCLUDE\_DIRS := $(PYTHON\_INCLUDE) /usr/local/include /usr/include/hdf5/serial

LIBRARY\_DIRS := $(PYTHON\_LIB) /usr/local/lib /usr/lib /usr/lib/x86\_64-linux-gnu /usr/lib/x86\_64-linux-gnu/hdf5/serial

这是因为ubuntu16.04的文件包含位置发生了变化，尤其是需要用到的hdf5的位置，所以需要更改这一路径.

（5）修改makefile文件

打开makefile文件，做如下修改：

将：

NVCCFLAGS +=-ccbin=$(CXX) -Xcompiler-fPIC $(COMMON\_FLAGS)

替换为：

NVCCFLAGS += -D\_FORCE\_INLINES -ccbin=$(CXX) -Xcompiler -fPIC $(COMMON\_FLAGS)

（6）编辑/usr/local/cuda/include/host\_config.h

将其中的第115行注释掉：

将

#error-- unsupported GNU version! gcc versions later than 4.9 are not supported!

改为

//#error-- unsupported GNU version! gcc versions later than 4.9 are not supported!

（7）编译

make all -j8 #-j根据自己电脑配置决定

编译过程中可能会出现如下错误：

错误1：

"libcudart.so.8.0 cannot open shared object file: No such file or directory"

解决办法是将一些文件复制到/usr/local/lib文件夹下：

#注意自己CUDA的版本号！

sudo cp /usr/local/cuda-8.0/lib64/libcudart.so.8.0 /usr/local/lib/libcudart.so.8.0 && sudo ldconfig

sudo cp /usr/local/cuda-8.0/lib64/libcublas.so.8.0 /usr/local/lib/libcublas.so.8.0 && sudo ldconfig

sudo cp /usr/local/cuda-8.0/lib64/libcurand.so.8.0 /usr/local/lib/libcurand.so.8.0 && sudo ldconfig

错误2：

Usr/bin/Id:cannot find –lxxx错误

通常在软件编译时出现的usr/bin/ld:cannot find –lxxx的错误，主要的原因是库文件并没有导入到ld检索目录中

解决方式：

确认库文件是否存在，比如-l123, 在/usr/lib, /usr/local/lib,或者其他自定义的lib下有无lib123.so, 如果只是存在lib123.so.1, 那么可以通过ln -sv lib123.so.1 lib123.so，建立一个连接重建lib123.so.

（8）测试

make test -j4

make runtest -j4

如果运行之后出现下图，说明caffe配置成功。

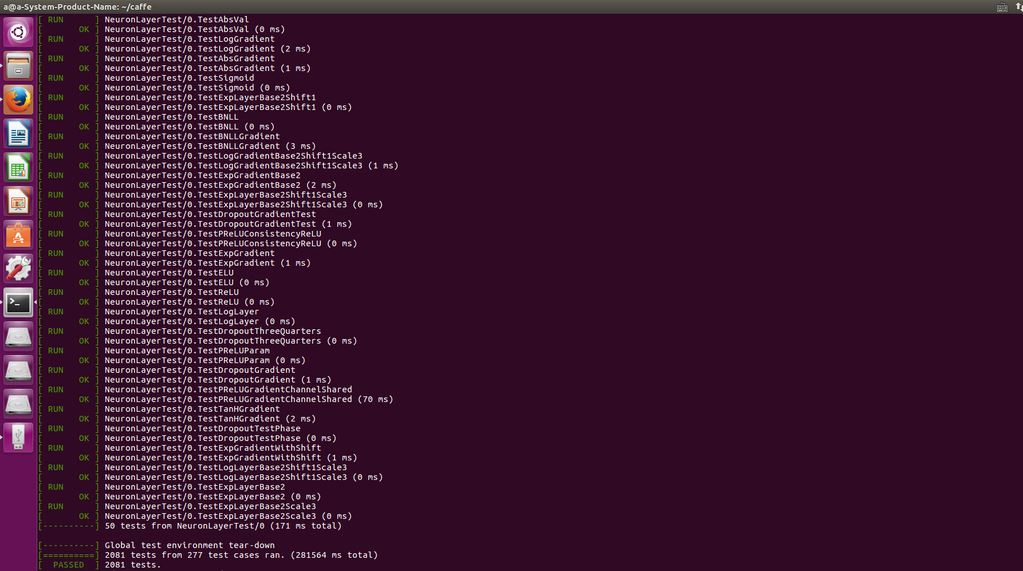


图6.caffe测试结果

到此caffe配置完毕！

**9. MNIST数据集测试**

配置caffe完成后，我们可以利用MNIST数据集对caffe进行测试，过程如下：

1.将终端定位到Caffe根目录

cd ~/caffe

2.下载MNIST数据库并解压缩

./data/mnist/get\_mnist.sh

caffe/data/mnist/目录下会多出训练集图片、训练集标签、测试集图片和测试集标签等4个文件

3.将其转换成Lmdb数据库格式

./examples/mnist/create\_mnist.sh

必须要在caffe根目录下运行create\_mnist.sh

此时在/caffe/examples/mnist/目录下生成mnist\_test\_lmdb和mnist\_train\_lmdb两个LMDB格式的训练集和测试集

4.训练网络

LeNet-5模型描述在/caffe/examples/mnist/lenet\_train\_test.prototxt

Solver配置文件在/caffe/examples/mnist/lenet\_solver.prototxt

执行文件在/caffe/examples/mnist/train的lenet.sh

当前位置在caffe根目录下

终端输入

./examples/mnist/train\_lenet.sh

即可开始训练，准确率一般都有0.99以上

训练的时候可以看到损失与精度数值，如下图：

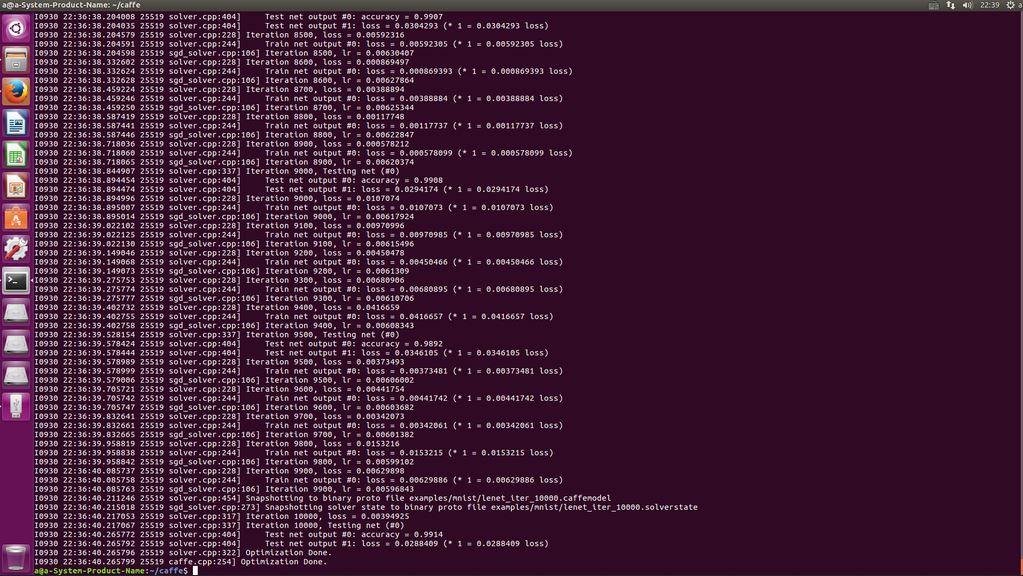


图7.MNIST数据集训练

可以看到最终训练精度是0.9914。

十、备份系统

见 blog.chriscabin.com/os/linux/Ubuntu/1016.html

备份文件放在了 /media 目录下

参考链接：

http://www.linuxdiyf.com/linux/24659.html 大部分参考这里 opencv（有warning，查了好久没成功，忽略了）

http://blog.csdn.net/g0m3e/article/details/51420565 安装驱动是照这里的第一个方法

http://www.2cto.com/os/201607//528798.html cuda安装

http://www.cnblogs.com/djiankuo/p/5886605.html 测试cuda

http://blog.csdn.net/zhongshijunacm/article/details/52824894 测试cuda的sample

http://blog.csdn.net/leo\_xu06/article/details/53010900 配置python

http://www.cnblogs.com/platero/p/4118139.html NVIDIA CuDNN 安装说明 重要！！

http://www.mamicode.com/info-detail-1003750.html 解决 usr/bin/ld:cannot find -lxxx的错误的思路（caffe 配置中 make runtest出现的错）

http://blog.csdn.net/lee\_j\_r/article/details/52693724 mnist测试

http://blog.csdn.net/gwzz1228/article/details/34424179 如何解决make: Nothing to be done for `all' 的方法

其他参考

http://blog.csdn.net/leo\_xu06/article/details/53010900