环境：Ubuntu14.04（中文版） + GT 705

主要内容：安装Theano

安装NVIDIA驱动：NVIDIA-Linux-x86\_64-375.66.run

　　安装CUDA：Cuda\_8.0.61\_375.26\_linux.run

　　安装CUDNN：Cudnn-8.0-linux-x64-v5.0-ga.tar

因为是新手，这个使用GPU进行加速安装了快三天，各种问题折腾得很是烦躁，好在坚持下来了，希望后面也不会出问题。把的安装过程记下，怕后面有突发情况……

注意事项：

1. 先把theano安装好，省去在安装驱动的过程中的各种包的一些apt-get；
2. 安装NVIDIA驱动和CUDA的时候都需要禁用nouveau和X服务；
3. 最好先不要更新系统，不要安装搜狗输入法之类的东西，会出现中文乱码等等问题。

参考：

1. theano的安装比较早，详细的参考目录不太记得了，先贴几个：

<http://blog.csdn.net/xuezhisdc/article/details/47065475>

<http://blog.csdn.net/TriLoo/article/details/52785947?_t_t_t=0.3240865101106465>

<http://www.cnblogs.com/linhao-0204/p/6148842.html>

<http://blog.csdn.net/a1311543690/article/details/48878843>

1. NVIDIA和CUDAA和CUDNN的参考目录：

<http://blog.csdn.net/zafir_410/article/details/73188228?utm_source=itdadao&utm_medium=referral>

<http://blog.csdn.net/u012759136/article/details/53355781>

<http://www.jianshu.com/p/d72905e9bec5>

<http://blog.csdn.net/u012177034/article/details/61614497>

<http://blog.csdn.net/zizi7/article/details/72954295>

<https://zhidao.baidu.com/question/2205362345149227068.html>

<http://blog.sina.com.cn/s/blog_9db078090102vdvx.html>

<http://www.linuxidc.com/Linux/2016-11/137561.htm>

1. Theano的GPU测试

<http://blog.csdn.net/lliuyi_zhanghao/article/details/51233492>

<http://blog.csdn.net/wonengguwozai/article/details/52740674?locationNum=13>

**安装Theano**

写在前面：安装Theano，就需要先安装好numpy和scipy；

要安装numpy和scipy，就需要ATLAS；

要安装ATLAS，就需要安装BLAS和LAPACK;

安装顺序：BLAS → LAPACK → ATLAS → numpy → scipy → Theano

注意：如果你的numpy和scipy是通过apt-get安装的，那么它们的单元测试可能会通不过！！

如果numpy或scipy不能通过测试，就需要卸载，然后重新按照本文介绍的顺序安装。

# 卸载numpy

sudo apt-get remove python-numpy

# 卸载scipy

sudo apt-get remove python-scipy

第一步：安装各种包

# 安装gfortran,后面编译过程中会用到

sudo apt-get install gfortran

# 安装blas,Ubuntu下对应的是libopenblas，其它操作系统可能需要安装其它版本的blas——这是个OS相关的。

sudo apt-get install libopenblas-dev

# 安装lapack，Ubuntu下对应的是liblapack-dev，和OS相关。

sudo apt-get install liblapack-dev

# 安装atlas，Ubuntu下对应的是libatlas-base-dev，和OS相关。

sudo apt-get install libatlas-base-dev

sudo apt-get install python-dev

sudo apt-get install python-devel

sudo apt-get install python-pip

sudo pip install atlas

sudo pip install nose

第二步：安装numpy和scipy

# 安装numpy

sudo pip install numpy

# 测试numpy

# 测试通过才能进行下一步～～

python -c "import numpy;numpy.test()"

# 安装scipy

sudo pip install scipy

# 测试scipy

# 测试通过才能进行下一步～～

python -c "import scipy;scipy.test()"

# 安装nose

sudo apt-get install python-nose

# 测试

python

>>> import nose

>>> import numpy

>>> import scipy

>>> numpy.test('full')

等待。。。。

>>> scipy.test('full')

第三步：安装其他库

为了安装Theano，最后还需要安装一些库，可以参考[官方教程](http://deeplearning.net/software/theano/install_ubuntu.html#install-ubuntu)

sudo apt-get install python-dev

sudo apt-get install python-pip

sudo apt-get install python-nose

sudo apt-get install g++

sudo apt-get install git

第四步：安装Theano

# 安装Theano

sudo pip install Theano

用下面命令对安装的Theano进行[**测试**](http://lib.csdn.net/base/softwaretest)：

$python -c "import theano; theano.test()"

运行时可能会出现错误：

...no module named nose-parameterized...

则安装nose-parameterized:

$pip install nose-parameterized

程序能运行且没有error即可。   
运行的时候，貌似要以Root身份运行才不会出错。

**安装NVIDIA驱动**

写在前面：差一点就先安装CUDA再安驱动了，正常顺序是先驱动在CUDA，本打算这次要是再失败就倒着来，因为有个博主这么说来着：在安装了cuda之后再安装nvidia驱动, （http://blog.csdn.net/u012759136/article/details/53355781）可能这和网上“流传”的教程有些出入，但是长得帅的人都这样做了，换句话说，这样做的话会避免出现一些奇怪的问题。比如：tf安装好了出现的缺少libcuda.so.1。万幸这次成功了。

主要参考：<http://blog.csdn.net/zizi7/article/details/72954295>

还有一篇博文里介绍了三种安装方法，第一种就是这种，他主要也是用的第一种，后面两种要是感兴趣也可以试试：

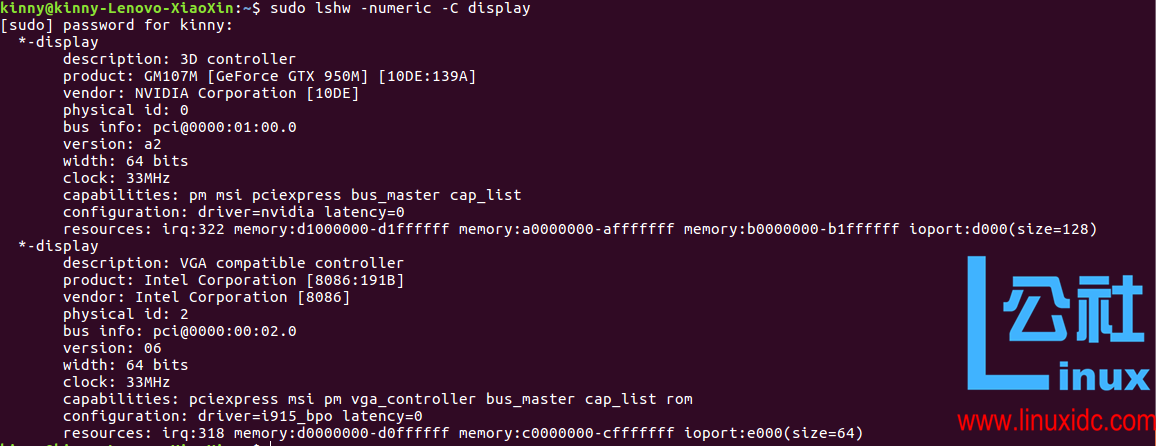
<http://blog.csdn.net/zafir_410/article/details/73188228?utm_source=itdadao&utm_medium=referral>

**大致的思路是：**禁用ubuntu默认的开源驱动nouveau -> 禁用桌面X服务 -> 安装驱动 -> 开启X服务

第零步：前期准备，找到适合的正确的驱动＆＆以及卸载掉原有驱动

方法一：首先检查你的NVIDIA VGA card model

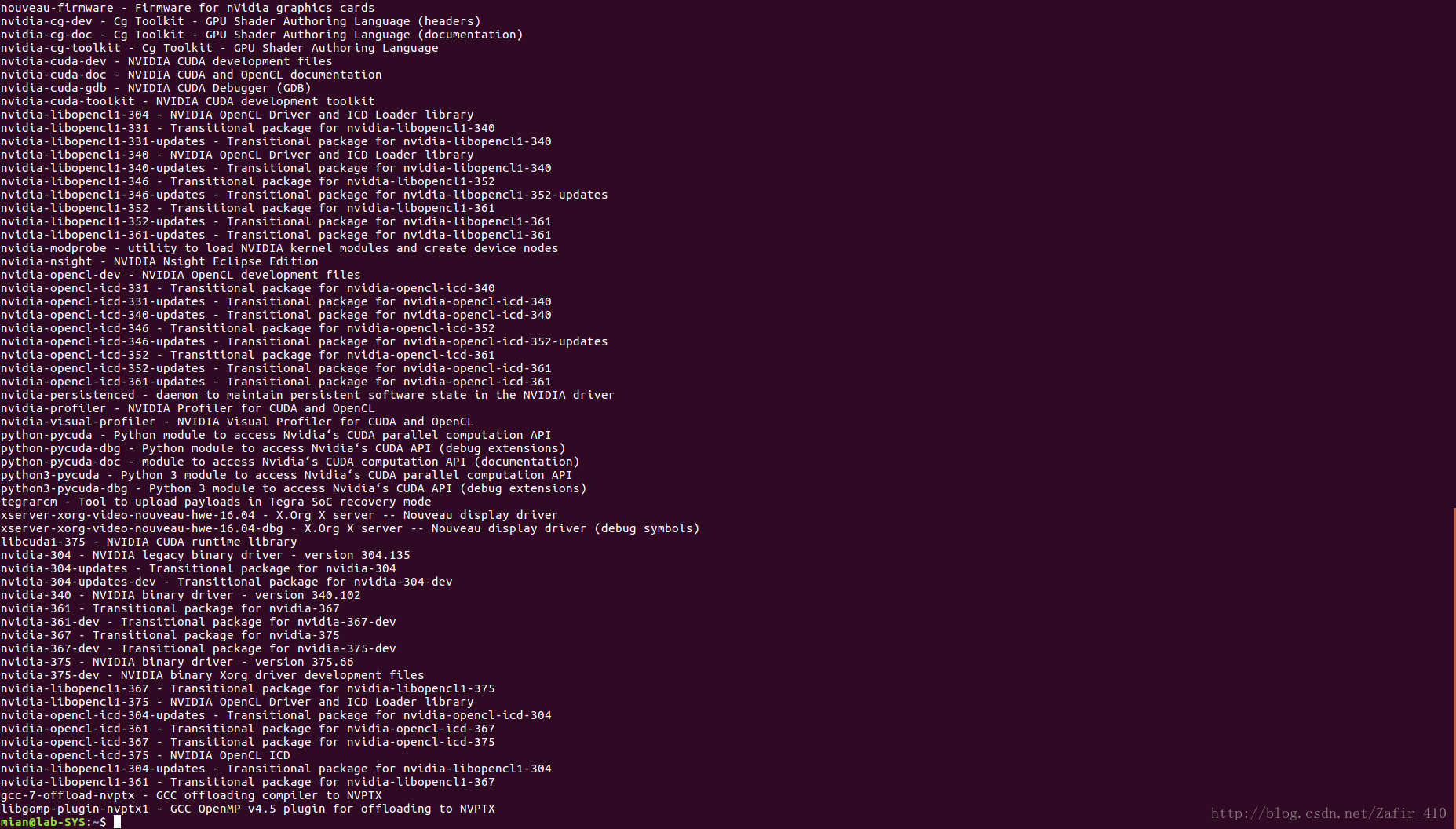
sudo lshw -numeric -C display

  
可以看到你的显卡信息，比如博主的就是 product: GM107M [GeForce GTX 950M] [10DE:139A]，然后去[NVDIA driver search page](http://www.nvidia.com/Download/index.aspx)搜索你的显卡需要的驱动型号

方法二：你首先知道你该装那个版本的驱动，不然也是徒劳的，方式如下：

sudo apt-cache search nvidia\*

结果如下：



可以看到博主的电脑最高支持nvidia-375.66的驱动。有了版本就好办了。下载了NVIDIA-[**Linux**](http://lib.csdn.net/base/linux)-x86\_64-375.66.run （根据你的版本号下载），具体的安装过程如下（可参考师弟的博客：<http://blog.csdn.net/u012759136/article/details/53355781>）：

卸载掉原有驱动

sudo apt-get purge nvidia\*

**这个也和重要，我的有一台服务器没有这个操作，还真不行**。如果使用的是runfile，则使用–uninstall 参数，当然runfile安装的时候会卸载掉之前的驱动，所以可以不用手动去卸载。

第一步：禁用开源驱动nouveau和X服务

**1）禁用nouveau**   
编辑黑名单文件

sudo gedit /etc/modprobe.d/blacklist.conf

在文末添加

blacklist vga16fb

blacklist nouveau

blacklist rivafb

blacklist nvidiafb

blacklist rivatv

【还看到过另外一个禁用的方法，我两种都弄了，也不知道那种起了效果：

（参考：http://blog.csdn.net/zafir\_410/article/details/73188228?utm\_source=itdadao&utm\_medium=referral）

新建blacklist-nouveau.conf文件，输入命令：

sudo gedit /etc/modprobe.d/blacklist-nouveau.conf

往文件中写入：

blacklist nouveau

blacklist lbm-nouveau

options nouveau modeset=0

alias nouveau off

alias lbm-nouveau off

保存并退出。这一步是为了禁掉Ubuntu自带开源驱动nouveau。之后sudo reboot重启系统。在终端执行命令：

lsmod | grep nouveau

查看nouveau模块是否被加载。如果什么都没输出，则执行下一步。】

更新状态

sudo update-initramfs -u

重启系统，检查nouveau是否禁用成功（没有输出）

lsmod | grep nouveau

**2）禁用X服务**

sudo service lightdm stop

此时桌面黑屏，切换到命令行界面（ctrl+alt+f1）

输入登录名：xuling 密码：＊＊＊＊＊＊

第二步：安装驱动

cd进入文件夹下，给 runfile 增加 执行权限

sudo chmod a+x NVIDIA-Linux-x86\_64-375.66.run

安装

sudo ./NVIDIA-Linux-x86\_64-375.66.run -no-opengl-files

**注意：** a. 使用 ./ 的执行方式   
　　　 b. 添加安装参数**no-opengl-file**（只安装驱动，不安装OpenGL）   
　　　 　（如果安装OpenGL会导致X服务启动失败的循环登录）   
　　　 ｃ.对于UEFI启动的主板，需要进 Bios **关闭 secure boot**

**在安装驱动的时候，有一步问你”Would you like to run the nvidia-xconfig utility to automatically update your X configuration file…”什么的，选择 No。**

sudo ./NVIDIA-Linux-x86\_64-375.20.run --no-opengl-files –no-x-check –no-nouveau-check

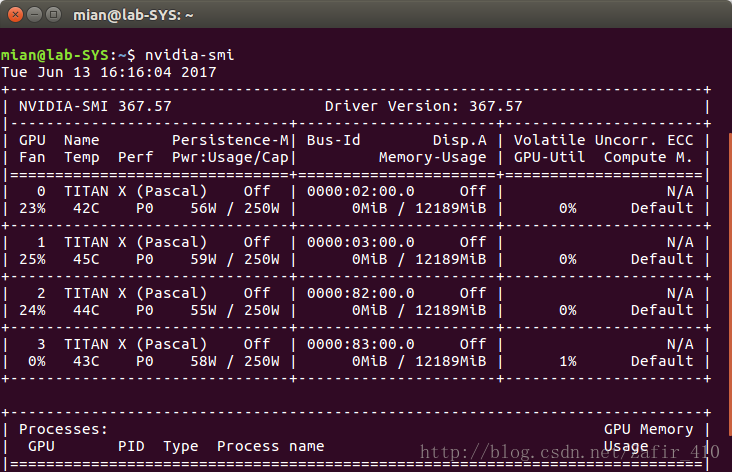
* –no-x-check 安装驱动时不检查X服务
* –no-nouveau-check 安装驱动时不检查nouveau

重启电脑，进行**验证** ：因为这种方式安装的驱动，显示还是用的集显，因此系统信息里图形选项仍然是 intel\* nvidia-prime 也无法找到独显，所以使用 nvidia-smi 可以看到显卡信息，说明安装成功

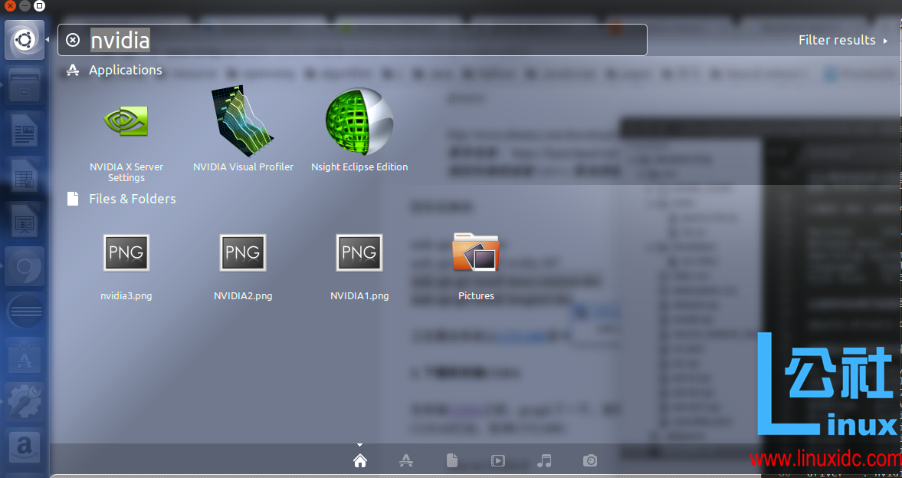
输入命令：

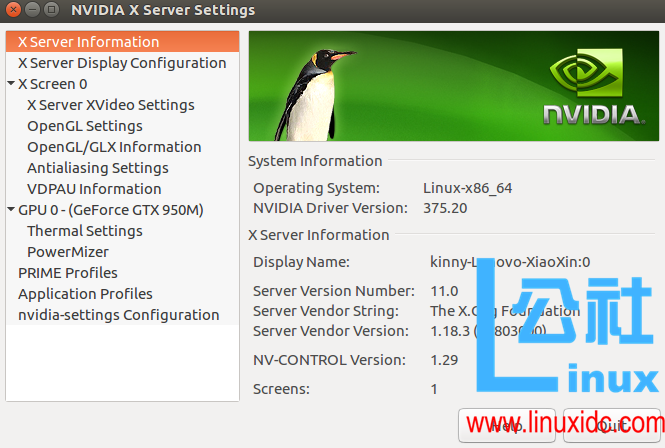
nvidia-smi

然后出现：



安装完成之后，重启电脑，驱动应该就完成了！你也可以在dashboard上搜索nvidia，看到像 NVIDIA X Server Settings的东西，就说明安装驱动成功了，接下来就是安装cuda8了。





小技巧提示，如果你装驱动失败，进不去系统了，可能不知道怎么做，就直接系统重装，这是一种方式，但是最简单的方式是，进文本tty1模式，输入命令：

sudo /etc/init.d/lightdm stop

sudo ./NVIDIA-Linux-x86\_64-375.20.run --uninstall

sudo /etc/init.d/lightdm start

系统回到从前，继续做打不死的小强，继续试试。

**安装CUDA**

注意：cuda的安装与nvidia驱动的安装思路类似，需要首先关闭图形界面，安装完成后再开启。一个关键点是，使用runfile的方式（deb的方式会再次安装nvidia驱动）

第一步：禁用开源驱动nouveau和X服务

同上一步的步骤

第二步：安装ＣＵＤＡ

cd进入文件夹下，给 runfile 增加 执行权限

sudo chmod a+x cuda\_8.0.61\_375.26\_linux.run

安装

sudo sh cuda\_8.0.61\_375.26\_linux.run

注意：安装过程会有多次输入提示，装cuda的时候不要再装驱动了，“**是否安装 nvidia driver”一项选 no**，其他 yes，具体选项如下：

Description

This package includes over 100+ CUDA examples that demonstrate

various CUDA programming principles, and efficient CUDA

implementation of algorithms in specific application domains.

The NVIDIA CUDA Samples License Agreement is available in

Do you accept the previously read EULA?

accept/decline/quit: accept

Install NVIDIA Accelerated Graphics Driver for Linux-x86\_64 367.48?

(y)es/(n)o/(q)uit: **n**

Install the CUDA 8.0 Toolkit?

(y)es/(n)o/(q)uit: y

Enter Toolkit Location

[ default is /usr/local/cuda-8.0 ]:

Do you want to install a symbolic link at /usr/local/cuda?

(y)es/(n)o/(q)uit: y

Install the CUDA 8.0 Samples?

(y)es/(n)o/(q)uit: y

Enter CUDA Samples Location

[ default is /home/c302 ]:

Installing the CUDA Toolkit in /usr/local/cuda-8.0 ...

Installing the CUDA Samples in /home/c302 ...

Copying samples to /home/c302/NVIDIA\_CUDA-8.0\_Samples now...

Finished copying samples.

===========

= Summary =

===========

Driver: Not Selected

Toolkit: Installed in /usr/local/cuda-8.0

Samples: Installed in /home/kinny, but missing recommended libraries

Please make sure that

- PATH includes /usr/local/cuda-8.0/bin

- LD\_LIBRARY\_PATH includes /usr/local/cuda-8.0/lib64, or, add /usr/local/cuda-8.0/lib64 to /etc/ld.so.conf and run ldconfig as root

To uninstall the CUDA Toolkit, run the uninstall script in /usr/local/cuda-8.0/bin

Please see CUDA\_Installation\_Guide\_Linux.pdf in /usr/local/cuda-8.0/doc/pdf for detailed information on setting up CUDA.

\*\*\*WARNING: Incomplete installation! This installation did not install the CUDA Driver. A driver of version at least 361.00 is required for CUDA 8.0 functionality to work.

To install the driver using this installer, run the following command, replacing <CudaInstaller> with the name of this run file:

sudo <CudaInstaller>.run -silent -driver

Logfile is /tmp/cuda\_install\_17494.log

第三步：设置环境变量

重启系统，编辑文件 ~/.bashrc，

sudo gedit ~/.bashrc

在文末添加路径

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

export LD\_LIBRARY\_PATH=/usr/local/cuda8.0/lib64:$LD\_LIBRARY\_PATH

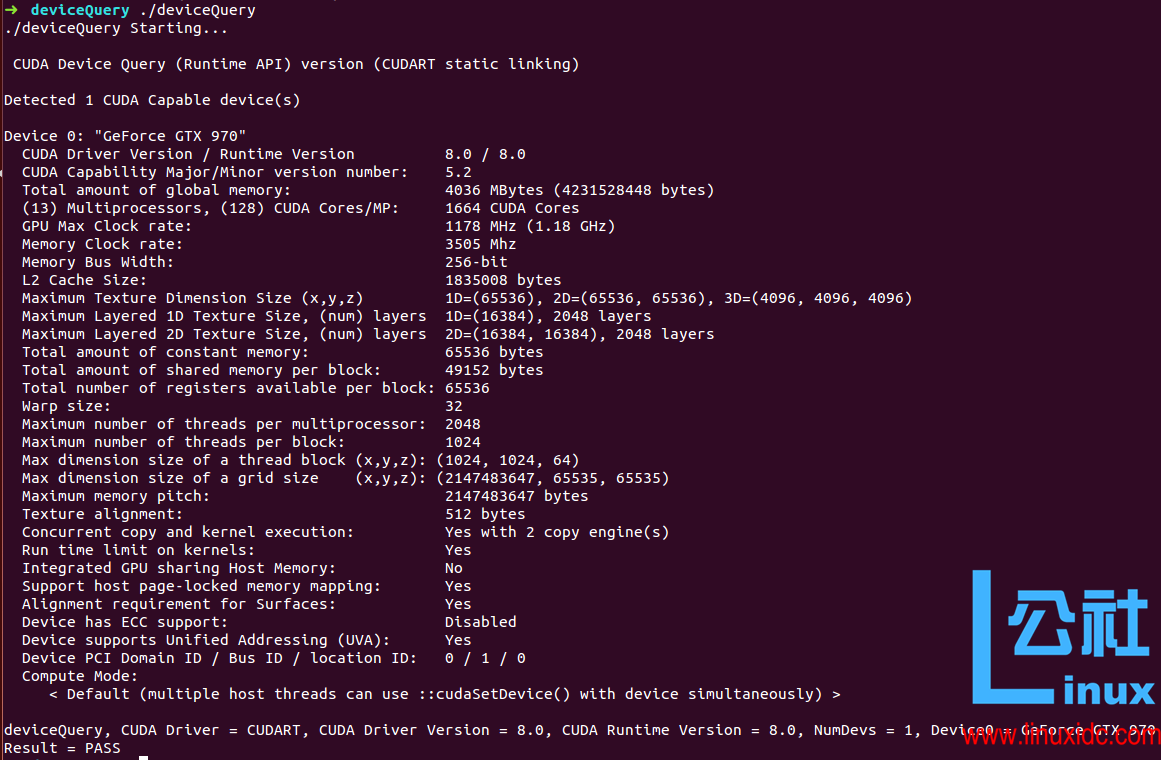
生效：

sudo source ./bashrc

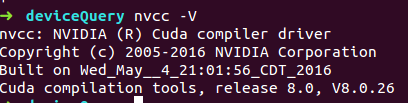
第四步：验证安装正确

进入 ~/NVIDIA\_CUDA-8.0\_Samples，执行 make （需要大概10分钟）   
进入 ~/NVIDIA\_CUDA-8.0\_Samples/bin/x86\_64/linux/release ，执行

./deviceQuery



执行：$ nvcc -V，结果如下：



禁忌：

1. 安装CUDA时一定使用runfile文件，这样可以进行选择。不再选择安装驱动，以及**在弹出xorg.conf时选择NO**
2. 不要使用ubuntu设置中附加驱动中驱动

无法进入桌面的问题：如果出现无法进入桌面的问题，这是因为驱动修改了xorg的配置，可执行一下命令：

cd /usr/share/X11/xorg.conf.d/   
sudo mv nvidia-drm-outputclass.conf nvidia-drm-outputclass.conf.bak

提醒：cuda安装时会将源文件装在/usr/local/cuda-8.0下，然后会创建一个/usr/local/cuda的链接，所以在安装tensorflow设置cuda和cudnn路径以及拷贝cudnn文件到cuda文件夹下时，注意使用真实的路径

出现问题：

1、循环登录   
有时候重启电脑之后进入图形界面输入用户密码却怎么都进不了桌面，又重新弹出登录界面。   
解决办法：重装显卡驱动   
首先，需要事先在NVIDIA官网下载与自己显卡对应的图形驱动（cuda对显卡有一定的要求，基本上得是n卡才能保证一定跑的起来），保存在电脑某个自己找得到的地方。   
遭遇循环登录时，ctrl+f1进入shell主界面，执行以下操作：

cd 存放驱动的文件夹

sudo service lightdm stop #关闭图形界面

sudo sh NVIDIA\_Linux-xxxxxxx.ru #选择accept、continue 、OK等

sudo service lightdm start

2、查询不到nvcc   
装好cuda后利用nvcc –version查询nvcc版本，出现以下提示信息：   
Theprogram ‘nvcc’ is currently not installed. You can install it by typing:   
sudoapt-get install nvidia-cuda-toolkit   
注意：不要执行提示中的命令   
执行sudoapt-get install nvidia-cuda-toolkit后系统会自动安装某个低版本的cuda,当时我手贱，这么试了一下，就装上了cuda5.5，而且找不到对应文件夹，没法删除。

**安装深度学习库cuDNN**

第一步：解压

tar xvzf cudnn-8.0-linux-x64-v5.1.tgz

第二步：COPY

把相应的文件拷贝到cuda的对应目录。(注意是cuda, 不是cuda-8.0)

然后将 include 和 lib64 里的文件分别 copy 到 /usr/local/cuda/include/ 以及/usr/local/cuda/lib64/目录中

（一定是你自己安装的目录如/usr/local/cuda-8.0）,不过正确安装的话，ubuntu一般就会有软链接/usr/local/cuda -> /usr/local/cuda-8.0/

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

sudo cp cuda/lib64/libcudnn\* /usr/local/cuda/lib64

sudo chmod a+r /usr/local/cuda/include/cudnn.h /usr/local/cuda/lib64/libcudnn\*

第三步：建立软链接

最后然后 cd 到 /usr/local/cuda/lib64/，为 libcudnn.so.5 和 libcudnn.so.5.1.10 建立软链接

sudo ln -sf libcudnn.so.5.1.5 libcudnn.so.5

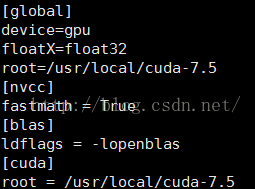
sudo ln -sf libcudnn.so.5 libcudnn.so

**测试Theano**

先**配置主目录下~/.theanorc 内容如下：**

sudo gedit ~/.theanorc

[global]  
device=gpu  
floatX=float32  
root=/usr/local/cuda-8.0  
[nvcc]  
fastmath = True  
[blas]  
ldflags = -lopenblas  
[cuda]  
root = /usr/local/cuda-8.0



在[Ubuntu](http://www.linuxidc.com/topicnews.aspx?tid=2)下安装完Theano以及cuda后，可以使用如下程序来测试你当前是否使用了GPU：

from theano import function, config, shared, sandbox

import theano.tensor as T

import numpy

import time

vlen = 10 \* 30 \* 768 # 10 x #cores x # threads per core

iters = 1000

rng = numpy.random.RandomState(22)

x = shared(numpy.asarray(rng.rand(vlen), config.floatX))

f = function([], T.exp(x))

print(f.maker.fgraph.toposort())

t0 = time.time()

for i in range(iters):

r = f()

t1 = time.time()

print("Looping %d times took %f seconds" % (iters, t1 - t0))

print("Result is %s" % (r,))

if numpy.any([isinstance(x.op, T.Elemwise) for x in f.maker.fgraph.toposort()]):

print('Used the cpu')

else:

print('Used the gpu')

假设将上述代码存放在test\_gpu.py中，运行test\_gpu.py，

Python test\_gpu.py

如果输出如下结果：

[Elemwise{exp,no\_inplace}(<TensorType(float32, vector)>)]

Looping 1000 times took 3.06635117531 seconds

Result is [ 1.23178029 1.61879337 1.52278066 ..., 2.20771813 2.29967761

1.62323284]

Used the cpu

则说明当前使用的是CPU，并没有使用GPU。   
若出现类似如下结果：

Using gpu device 0: GeForce GTX 580

[GpuElemwise{exp,no\_inplace}(<CudaNdarrayType(float32, vector)>), HostFromGpu(GpuElemwise{exp,no\_inplace}.0)]

Looping 1000 times took 0.638810873032 seconds

Result is [ 1.23178029 1.61879349 1.52278066 ..., 2.20771813 2.29967761

1.62323296]

Used the gpu

这说明当前使用了GPU，并且告诉了我们当前使用的是哪个GPU。

也可以用下面的语句来测试

Sudo THEANO\_FLSGS=mode=FAST\_RUN,device=cpu,floatX=float32 python check1.py

Sudo THEANO\_FLSGS=mode=FAST\_RUN,device=gpu,floatX=float32 python check1.py

**安装pyyaml**   
打开终端输入命令：

Sudo pip install pyyaml

可以看到还有两个可选的库，建议安装HDF5和h5py，至于cuDNN根据你自己的情况而定。   
打开终端输入命令：

#先

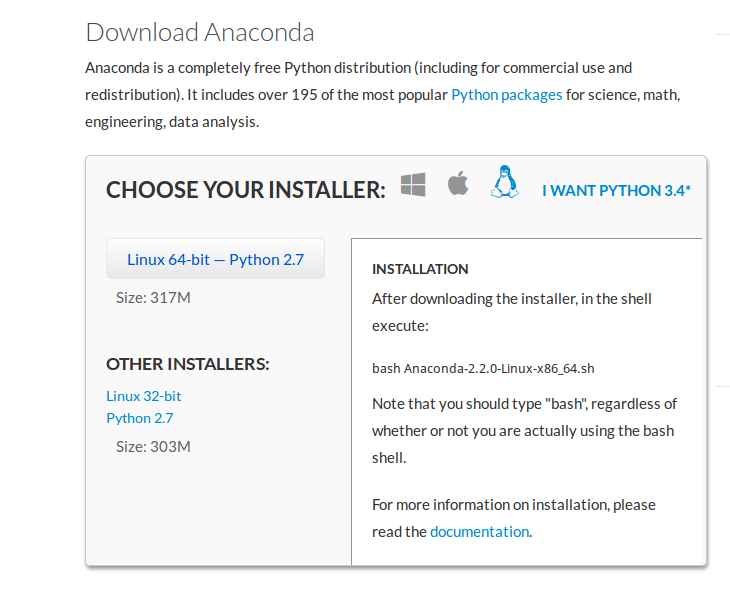
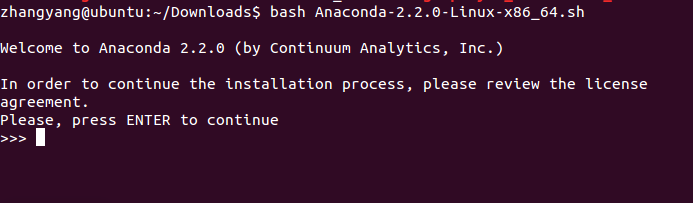
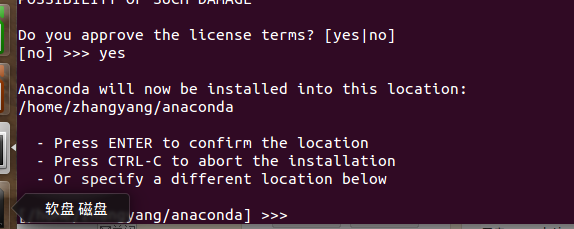
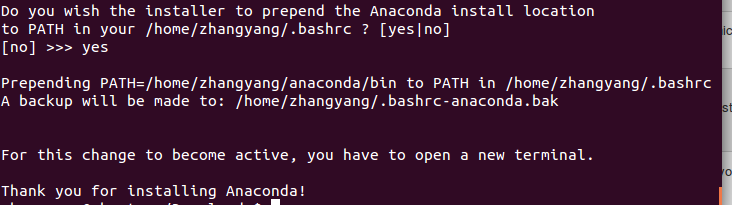
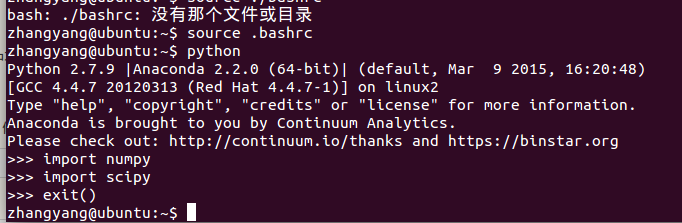
Sudo apt-get install libhdf5-serial-dev

#后

sudo pip install h5py

到这里所有的库已经安装完毕。

**安装Anaconda**

1、 到官网<http://continuum.io/downloads>下载anaconda。   
  
选择linux64-bit-python2.7   
2、 安装anaconda，在终端输入:cd ~/Downloads;bash Anaconda-2.2.0-[**Linux**](http://lib.csdn.net/base/linux)-x86\_64.sh   
  
回车后，是许可文件，接收许可，anaconda将安装在~/anaconda下：   
  
直接回车即可。   
3、 最后会询问是否把anaconda的bin添加到用户的环境变量中，选择yes：   
  
4、 在终端输入[**Python**](http://lib.csdn.net/base/python)发现依然是gnome自带的[**python**](http://lib.csdn.net/base/python)版本，这是因为.bashrc的更新还没有生效，命令行输入：source ~/.bashrc   


**创建新的python环境**

安装好后，我们需要创建一个新的python环境，这样我们所有的keras程序都安装在这个独立的环境，具体我们用到了conda这个命令：可以参考这个：http://www.tuicool.com/articles/vyyA7rB

conda create -n keras\_env python=2.7 numpy scipy

(创建一个keras\_env的环境，这个环境带了python2.7 numpy scipy)

最后使用这个环境我们要把这个环境active起来，注意后面所有的pip安装theano的命令都在这个kears\_env下执行

#激活环境

Source activate keras\_env

如果有什么库少了可以参考安装下面一些库，因为毕竟我们的环境有些少许出入：

sudo apt-get install python-numpy python-scipy python-dev python-pip python-nose g++ libopenblas-dev git

#在该环境下安装keras

Pip install theano

Pip install tensorflow-gpu

Pip install keras

#。。。。。等等其他的

补充：

关于tensorflow，网上好多都说用源码来安装，其实只要版本选择正确的话不用源码安装，还是非常容易的，所以一定要和自己安装的cuda和cudnn的版本对应。   
比如我安装了cuda 8.0 和 cudnn v5 , 则根据[tensorflow的官网](https://www.tensorflow.org/get_started/os_setup)的说明。

The GPU version works best with Cuda Toolkit 8.0 and cuDNN v5. Other versions are supported (Cuda toolkit >= 7.0 and cuDNN >= v3) only when installing from sources.

就是说如果不是cuda 8.0 和 cudnn v5的话，应该使用源码方式进行安装才行。幸运的是，我安装的正好就是cuda 8.0 和 cudnn v5，所以直接使用pip install 的方式就行了。因为我的是Ubuntu, 所以只需要执行下面一行命令就搞定了。

$ pip install tensorflow-gpu

安装好keras后，我们需要把keras的backend设置为theano(keras默认是用tf)，具体的方法是

查询到 ~/.keras/keras.json

{

"image\_dim\_ordering": "th",

"epsilon": 1e-07,

"floatx": "float32",

"backend": "theano"

}



参考：<http://keras-cn.readthedocs.io/en/latest/backend/>



{

"image\_dim\_ordering": "th",

"epsilon": 1e-07,

"floatx": "float32",

"backend": "theano"

}

8.设置keras的GPU模式：

vim 编辑 ～/.theanorc 输入以下字符：

[global]

device=gpu

floatX=float32

root=/usr/local/cuda

[nvcc]

fastmath = True

[blas]

ldflags = -lopenblas

[cuda]

root = /usr/local/cuda

[lib]

cnmem=0.8

[nvcc]

optimizer\_including=cudnn [global]

device=gpu

floatX=float32

root=/usr/local/cuda

[nvcc]

fastmath = True

[blas]

ldflags = -lopenblas

[cuda]

root = /usr/local/cuda

[lib]

cnmem=0.8

[nvcc]

optimizer\_including=cudnn

这里我安装了cudnn，如果没有安装就不需要指定最后的cudnn，但是cuda一定需要指定



关闭环境

Source deactivate keras\_env

开始使用先打开一个。。。。。。（我也不知道它是啥）

Anaconda-navigator

最上方的Application里面选择做使用的keras\_env【可以设多个环境】

在home里可以选择下载spyder，打开

在environment里面可以选择下载缺少的包，这里也需要确定环境是keras\_env（直接激活keras\_env，然后用pip下载更快）

直接在对话框里面import试试是否成功即可。

Tips：

提示缺少PIL的时候选择：

Source activate keras\_env

Pip install image

#这个时候就没有提示PIL缺少