ubuntu14.04 + cuda8.0（GTX1080）+ matlab2014a + caffe 安装

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

1、 ubuntu14.04 安装

分区：usr/local 300G

boot 200M

/ 根目录 300G

交换空间 20G （内存 16G）

其余都是/home

重装之后，下载 NVIDIA-Linux-x86_64-367.27.run，下载地址为：

http://www.geforce.cn/drivers/results/104314

将其与 cuda_8.0.27_linux.run（下载地址为：

https://developer.nvidia.com/cuda-toolkit 不过需要注册）拷到 home/zhou 下。

https://developer.nvidia.com/cuda-75-downloads-archive cuda7.5 下载网址

（注意：不要下载 cuda-repo-ubuntu1404-8-0-rc_8.0.27-1_amd64.deb 文件，.deb 文件会重新安装显卡驱动，造成与之前安装的显卡驱动冲突，使安装出错）

2、 NVIDIA 驱动安装

1)把 Nouveau 驱动加入黑名单（必须有，不然会出错）

参考：http://blog.csdn.net/misiter/article/details/7652731

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

在文件后面加入 blacklist nouveau，重启。

2) Ctrl+alt+F1 进入字符界面，关闭图形界面

sudo service lightdm stop //必须有，不然会安装失败

3) 安装 nvidia driver

sudo chmod a+x NVIDIA-Linux-x86_64-367.44.run //获取权限

sudo ./NVIDIA-Linux-x86_64-367.44.run //安装驱动

Accept

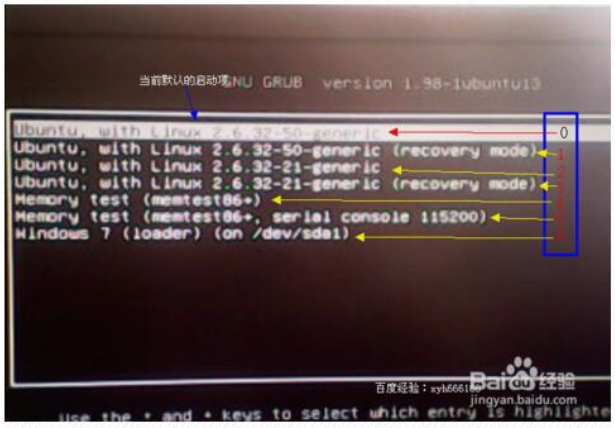
Continue installation

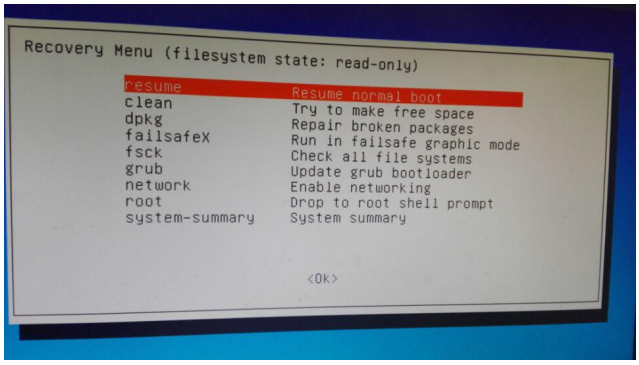
安装完成之后

sudo service lightdm start

图形界面出现，然后关机，回到图形界面，继续。

ps: 如果显卡驱动安装错误 导致不能进入登录界面 或者背景没有 停留在显示输入密码界面





3、 cuda8.0 安装

1）在终端运行指令 `sudo sh cuda_8.0.27_linux.run`

选择

```
Do you accept the previously read EULA?
accept/decline/quit: accept
    Install NVIDIA Accelerated Graphics Driver for Linux-x86_64 361.62?
(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/zhou ]:
   Installing the CUDA Toolkit in /usr/local/cuda-8.0 …
Missing recommended library: libGLU.so
Missing recommended library: libX11.so
Missing recommended library: libXi.so
Missing recommended library: libXmu.so
   Installing the CUDA Samples in /home/zhou …
Copying samples to /home/zhou/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/zhou, 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 with the name of this run file:
sudo .run -silent -driver
   Logfile is /tmp/cuda_install_2961.log
```

安装完成，但是缺少一些库。

2）安装所缺少的库

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

3) 设置环境变量

在终端输入这两句：

```
export PATH=/usr/local/cuda-8.0/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-8.0/lib64:$LD_LIBRARY_PATH
```

然后修改文件中环境变量设置

```
sudo gedit /etc/profile
```

输入上面 export 的两句

```
export PATH=/usr/local/cuda-8.0/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-8.0/lib64:$LD_LIBRARY_PATH
```

保存，退出。

```
sudo ldconfig //环境变量立即生效
```

4) 验证安装是否完成

```
nvidia-smi
```

显示：

```
nvcc --version
```

显示：

```
nvcc -V nvcc: NVIDIA (R) Cuda compiler driver Copyright
(c) 2005-2016
NVIDIA Corporation Built on Wed_May__4_21:01:56_CDT_2016
Cuda
compilation tools, release 8.0, V8.0.26
```

6) 测试 cuda 的 samples

```
cd '/home/zhou/NVIDIA_CUDA-8.0_Samples'
make
```

7) 安装 cudnn5.0

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：

```
tar -zxvf cudnn-8.0-linux-x64-v5.0-ga.tgz
```

显示以下信息：

```
*cuda/include/cudnn.h
cuda/lib64/libcudnn.so
cuda/lib64/libcudnn.so.5
cuda/lib64/libcudnn.so.5.0.5
cuda/lib64/libcudnn_static.a*
```

继续执行以下指令：

```
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
sudo chmod a+r /usr/local/cuda/lib64/libcudnn*
```

8）验证一下

cuda 的 samples 里面有个 deviceQuery 运行之后会显示信息，最后一行出行 pass 说明成功啦。

4、matlab 安装

1）下载 matlab for linux 如果压缩文件有两个 part A 和 part B 解压其中一个把.iso 文件拷贝出来就行。

2）然后按照以前的步骤来，不过这次没有创建快捷方式，因为根本没用过==。

下载完成后将 iso 文件挂载到 Linux

```
sudo mkdir /media/matlab
```

```
mount -o loop MATHWORKS_R2014A.iso /media/matlab
cd /media/matlab
sudo ./install
```

进行安装，安装路径：/usr/local/MATLAB/R2014a

PS：需要注意的是，我下载到的文件里面 readme.txt 里没有序列号，所以我就随便填了一个

选项：不使用 Internet 安装

序列号： 12345-67890-09876-54321

默认路径：/usr/local/MATLAB/R2014a

勾选从默认启动路径创建符号链接（实现在任意位置运行 matlab 启动程序）

3）激活，选择离线激活模式，选择 crack 文件夹下的 license_405329_R2014a.lic 文件进行激活。

4）破解 将 crack 文件夹下的 libmwservices.so 复制到 /usr/local/MATLAB/R2014A/bin/glnxa64

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

5、 caffe 安装

这里是跟着官网教程来的

1) 安装依赖项

```
sudo apt-get update
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
```

2) BLAS 安装

```
sudo apt-get install libatlas-base-dev
```

3) 安装 pycaffe 接口所需要的依赖项

```
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
```

4) 继续安装依赖项

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

5） opencv3.0 安装

参照 http://ouxinyu.github.io/Blogs/20151108001.html

（1）http://pan.baidu.com/s/1qX1uFHa 密码:wysa

使用他提供的修改版的安装包，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）

（2）切换到文件保存的文件夹（/home/zhou/opencv3.0），然后安装依赖项

```
sudo sh dependencies.sh
```

（3）安装 OpenCV3.0

正常安装之前，需要进行修改，为避免 make 到 72%出现错误，graphcuts.cpp 文件中许多变量没有声明，因为 opencv3.0 还不支持 cuda8.0。

google 之后发现,但是有个同学已经对其进行修改。

请参考：

https://github.com/opencv/opencv/pull/6510/commits/10896129b39655e19e4e7c529153cb5c2191a1db

```
cd  /home/zhou/opencv3.0/OpenCV/opencv-3.0.0/modules/cudalegacy/src
sudo gedit graphcuts.cpp
```

进行修改

然后保存并退出。

```
sudo sh opencv3_0_0.sh
```

然后开始正常安装，并开始 make。

运行正常，至此 opencv3.0 安装成功！！！

或者 sudo sh opencv2_4_10.sh

sudo gedit opencv2_4_10.sh 添加 -D CUDA_GENERATION=Auto

6.caffe 编译

下载 caffe-master，解压到/home/zhou 下，我重新命了个名字，caffe

然后 配置 Makefile.config 文件，执行指令：

cp Makefile.config.example Makefile.config

我的配置文件如下：

```
## Refer to http://caffe.berkeleyvision.org/installation.html
# Contributions simplifying and improving our build system are welcome!

# cuDNN acceleration switch (uncomment to build with cuDNN).
  USE_CUDNN := 1

# CPU-only switch (uncomment to build without GPU support).
# CPU_ONLY := 1

# uncomment to disable IO dependencies and corresponding data layers
 USE_OPENCV := 1
 USE_LEVELDB := 1
 USE_LMDB := 1
#如果要用 opencv levelldb lmdb 则全部改成 1

# uncomment to allow MDB_NOLOCK when reading LMDB files (only if necessary)
#   You should not set this flag if you will be reading LMDBs with any
#   possibility of simultaneous read and write
# ALLOW_LMDB_NOLOCK := 1

# Uncomment if you're using OpenCV 3
  OPENCV_VERSION := 3

# To customize your choice of compiler, uncomment and set the following.
# N.B. the default for Linux is g++ and the default for OSX is clang++
# CUSTOM_CXX := g++
```

```
# CUDA directory contains bin/ and lib/ directories that we need.
CUDA_DIR := /usr/local/cuda
# On Ubuntu 14.04, if cuda tools are installed via
# "sudo apt-get install nvidia-cuda-toolkit" then use this instead:
# CUDA_DIR := /usr

# CUDA architecture setting: going with all of them.
# For CUDA < 6.0, comment the *_50 lines for compatibility.
CUDA_ARCH := -gencode arch=compute_20,code=sm_20 \
       -gencode arch=compute_20,code=sm_21 \
       -gencode arch=compute_30,code=sm_30 \
       -gencode arch=compute_35,code=sm_35 \
       -gencode arch=compute_50,code=sm_50 \
       -gencode arch=compute_50,code=compute_50

# BLAS choice:
# atlas for ATLAS (default)
# mkl for MKL
# open for OpenBlas
BLAS := atlas
# Custom (MKL/ATLAS/OpenBLAS) include and lib directories.
# Leave commented to accept the defaults for your choice of BLAS
# (which should work)!
# BLAS_INCLUDE := /path/to/your/blas
# BLAS_LIB := /path/to/your/blas

# Homebrew puts openblas in a directory that is not on the standard search
path
# BLAS_INCLUDE := $(shell brew --prefix openblas)/include
# BLAS_LIB := $(shell brew --prefix openblas)/lib

# This is required only if you will compile the matlab interface.
# MATLAB directory should contain the mex binary in /bin.
  MATLAB_DIR := /usr/local/MATLAB/R2014a
 # MATLAB_DIR := /Applications/MATLAB_R2012b.app

# NOTE: this is required only if you will compile the python interface.
# We need to be able to find Python.h and numpy/arrayobject.h.
PYTHON_INCLUDE := /usr/include/python2.7 \
       /usr/lib/python2.7/dist-packages/numpy/core/include
# Anaconda Python distribution is quite popular. Include path:
# Verify anaconda location, sometimes it's in root.
# ANACONDA_HOME := $(HOME)/anaconda
```

```
# PYTHON_INCLUDE := $(ANACONDA_HOME)/include \
        # $(ANACONDA_HOME)/include/python2.7 \
        #
$(ANACONDA_HOME)/lib/python2.7/site-packages/numpy/core/include \

# Uncomment to use Python 3 (default is Python 2)
# PYTHON_LIBRARIES := boost_python3 python3.5m
# PYTHON_INCLUDE := /usr/include/python3.5m \
#                 /usr/lib/python3.5/dist-packages/numpy/core/include

# We need to be able to find libpythonX.X.so or .dylib.
PYTHON_LIB := /usr/lib
# PYTHON_LIB := $(ANACONDA_HOME)/lib

# Homebrew installs numpy in a non standard path (keg only)
# PYTHON_INCLUDE += $(dir $(shell python -c 'import numpy.core;
print(numpy.core.__file__)'))/include
# PYTHON_LIB += $(shell brew --prefix numpy)/lib

# Uncomment to support layers written in Python (will link against Python
libs)
  WITH_PYTHON_LAYER := 1

# Whatever else you find you need goes here.
INCLUDE_DIRS := $(PYTHON_INCLUDE) /usr/local/include
LIBRARY_DIRS := $(PYTHON_LIB) /usr/local/lib /usr/lib

# If Homebrew is installed at a non standard location (for example your
home directory) and you use it for general dependencies
# INCLUDE_DIRS += $(shell brew --prefix)/include
# LIBRARY_DIRS += $(shell brew --prefix)/lib

# Uncomment to use `pkg-config` to specify OpenCV library paths.
# (Usually not necessary -- OpenCV libraries are normally installed in
one of the above $LIBRARY_DIRS.)
# USE_PKG_CONFIG := 1

# N.B. both build and distribute dirs are cleared on `make clean`
BUILD_DIR := build
DISTRIBUTE_DIR := distribute

# Uncomment for debugging. Does not work on OSX due to
https://github.com/BVLC/caffe/issues/171
# DEBUG := 1
```

```
# The ID of the GPU that 'make runtest' will use to run unit tests.
TEST_GPUID := 0

# enable pretty build (comment to see full commands)
Q ?= @
```

```
make all -j16
make test -j16
make runtest -j16
```

make runtest –j16 时出现错误： libcudart.so.8.0: cannot open shared object file: No such file or directory

解决办法：在终端再次运行

```
export LD_LIBRARY_PATH=/usr/local/cuda-8.0/lib64:$LD_LIBRARY_PATH
```

再次运行 `make runtest -j16`

编译 Python 和 Matlab 用到的 caffe 文件

```
make pycaffe -j16
make matcaffe -j16
```

到这里就编译完成啦！！！！可以自己用里面的例子去嗨，去浪啦！！！

7、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 支持多种数据格式输入网络，包括 Image(.jpg, .png 等)，leveldb，lmdb，HDF5 等，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。

附录：第 6 和第 7 用到的指令可以写成.sh 文件，一次运行，因为某条指令可能会运行很长时间。

mynote.sh

```
cd ..
cd caffe
rm -rf build
mkdir build
make clean  #把之前编译的去掉
```

```
make all -j16
make test -j16
export LD_LIBRARY_PATH=/usr/local/cuda-8.0/lib64:$LD_LIBRARY_PATH
make runtest -j16
make pycaffe -j16
make matcaffe -j16
#1. 数据预处理
sh data/mnist/get_mnist.sh
#2. 重建 lmdb 文件。Caffe 支持多种数据格式输入网络，包括 Image(.jpg, .png 等)，
leveldb，lmdb，HDF5 等，根据自己需要选择不同输入吧。
sh examples/mnist/create_mnist.sh
sh examples/mnist/train_lenet.sh
```