buntu 14.04+cuda 7.5+caffe安装配置（20160622）

主要参考:

<http://blog.csdn.net/ubunfans/article/details/47724341>

<http://www.cnblogs.com/karac/p/5456134.html>

电脑配置：

ubuntu 14.04 64bit

内存: 32G

显卡: GTX650 ti

CPU : i7 4770

软件版本：

CUDA 7.5

caffe 当天从github下载的版本

Caffe 安装配置步骤：

一， 安装开发所需的依赖包

$ sudo apt-get install build-essential # basic requirement

$ sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libboost-all-dev libhdf5-serial-dev libgflags-dev libgoogle-glog-dev liblmdb-dev protobuf-compiler #required by caffe

二，安装CUDA 7.5

1. 离线.deb安装：官网下载，地址:https://developer.nvidia.com/cuda-downloads

安装之前请先进行md5校验，确保下载的安装包完整

1. 切换到下载的deb所在目录，执行下边的命令

$ sudo dpkg -i cuda-repo-ubuntu1404-7-5-local\_7.5-18\_amd64.deb

$ sudo apt-get update

$ sudo apt-get install cuda

1. 设置cuda的环境变量
2. 使用如下命令：

$ sudo vim /etc/profile

在/etc/profile中添加CUDA环境变量末尾添加如下内容：

PATH=/usr/local/cuda/bin:$PATH

export PATH

1. 保存后, 执行下列命令, 使环境变量立即生效

$ source /etc/profile

1. 同时需要添加lib库路径： 使用

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

在 /etc/ld.so.conf.d/加入cuda.conf文件, 内容如下

/usr/local/cuda/lib64

1. 保存后，执行下列命令使之立刻生效

$ sudo ldconfig

1. 然后重启电脑：sudo reboot

三，安装cuDNN

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

五，安装CUDA SAMPLE

1. 进入/usr/local/cuda/samples, 执行下列命令来编译samples：

$ cd /usr/local/cuda/samples

$ sudo make all -j8

1. 整个过程大概10分钟左右, 全部编译完成后，进入/usr/local/cuda/samples/bin/x86\_64/linux/release , 运行deviceQuery

$ cd /usr/local/cuda/samples/bin/x86\_64/linux/release

$ ./deviceQuery

如果出现类似如下显卡信息， 则驱动及显卡安装成功：

./deviceQuery Starting...

CUDA Device Query (Runtime API) version (CUDART static linking)

Detected 1 CUDA Capable device(s)

Device 0: "GeForce GTX 670"

  CUDA Driver Version / Runtime Version          6.5 / 6.5

  CUDA Capability Major/Minor version number:    3.0

  Total amount of global memory:                 4095 MBytes (4294246400 bytes)

  ( 7) Multiprocessors, (192) CUDA Cores/MP:     1344 CUDA Cores

  GPU Clock rate:                                1098 MHz (1.10 GHz)

  Memory Clock rate:                             3105 Mhz

  Memory Bus Width:                              256-bit

  L2 Cache Size:                                 524288 bytes

  Maximum Texture Dimension Size (x,y,z)         1D=(65536), 2D=(65536, 65536), 3D=(4096, 4096, 4096)

  Maximum Layered 1D Texture Size, (num) layers  1D=(16384), 2048 layers

  Maximum Layered 2D Texture Size, (num) layers  2D=(16384, 16384), 2048 layers

  Total amount of constant memory:               65536 bytes

  Total amount of shared memory per block:       49152 bytes

  Total number of registers available per block: 65536

  Warp size:                                     32

  Maximum number of threads per multiprocessor:  2048

  Maximum number of threads per block:           1024

  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)

  Max dimension size of a grid size    (x,y,z): (2147483647, 65535, 65535)

  Maximum memory pitch:                          2147483647 bytes

  Texture alignment:                             512 bytes

  Concurrent copy and kernel execution:          Yes with 1 copy engine(s)

  Run time limit on kernels:                     Yes

  Integrated GPU sharing Host Memory:            No

  Support host page-locked memory mapping:       Yes

  Alignment requirement for Surfaces:            Yes

  Device has ECC support:                        Disabled

  Device supports Unified Addressing (UVA):      Yes

  Device PCI Bus ID / PCI location ID:           1 / 0

  Compute Mode:

     < Default (multiple host threads can use ::cudaSetDevice() with device simultaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 6.5, CUDA Runtime Version = 6.5, NumDevs = 1, Device0 = GeForce GTX 670

Result = PASS

NOTE:上边的显卡信息是从别的地方拷过来的，我的GTX650ti显卡不是这些信息，如果没有这些信息，那肯定是安装不成功，找原因吧！

六，安装Intel MKL 或Atlas

我没有MKL，装的Atlas，安装命令：

$ sudo apt-get install libatlas-base-dev

七，安装OpenCV

1. 下载并编译OpenCV（官网原版OpenCV：http://opencv.org/）， 或者使用修改版的安装包(百度云盘链接: http://pan.baidu.com/s/1qX1uFHa 密码:wysa)（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，然后安装依赖项：(进入 “Install-OpenCV-master” 文件夹下)

$ sudo sh Ubuntu/dependencies.sh

1. 切换目录Ubuntu/3.0/安装OpenCV 3.0.0：

$ cd Ubuntu/3.0/

$ sudo sh opencv3\_0\_0.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。。。

八，安装Caffe所需要的Python环境

1. 按caffe官网的推荐使用Anaconda，去Anaconda官网下 安装包(下载python2.7 的Anaconda2-4.0.0-Linux-x86\_64.sh)
2. 切换到文件所在目录，执行

$ bash Anaconda2-4.0.0-Linux-x86\_64.sh

NOTE:后边的文件名按自己下的版本号更改，整个安装过程请选择默认,安装路径为 /home/username/anaconda (username为自己系统的用户名)

1. 添加Anaconda Library Path：使用

$ sudo vim /etc/ld.so.conf

在/etc/ld.so.conf最后加入以下路径 (NOTE:下边的wuyi要替换为自己的用户名)：

/home/wuyi/anaconda/lib

1. 使用

$ vim ~/.bashrc

在~/.bashrc最后添加下边路径：

  export LD\_LIBRARY\_PATH="/home/username/anaconda/lib:$LD\_LIBRARY\_PATH"

1. 退出运行如下命令使配置文件生效:

$ sudo ldconfig

九，安装python依赖库

1. [去caffe的github下载caffe源码包](https://github.com/BVLC/caffe)（<https://github.com/BVLC/caffe> ）
2. ﻿﻿进入caffe-master下的python目录，执行如下命令

$ sudo apt-get install python-pip #如果没有安装pip则需要执行该命令

$ for req in $(cat requirements.txt); do pip install $req; done

十，编译Caffe

1. 进入caffe-master目录，复制一份Makefile.config.examples：

$ cp Makefile.config.example Makefile.config

1. 修改配置文件，如果前边和该安装教程一致，那么配置文件应该是这个样子的：

## 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 := 0

# USE\_LEVELDB := 0

# USE\_LMDB := 0

# 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 #这里根据自己安装的加速库而定

# 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/wuyi/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 /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

# 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 ?= @

1. 保存退出，然后编译：

$ make all -j8  #4是指cpu核心数

$ make test

$ make runtest

十一，编译Python wrapper

$ make pycaffe

十二， 添加~/caffe/python到$PYTHONPATH：

$ vim ~/.bashrc

在末尾添加：

export PYTHONPATH=”/home/wuyi/caffe-master/python:$PYTHONPATH”

$ sudo ldconfig # 使之生效

**重启系统使环境变量完全生效**

# 十三，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

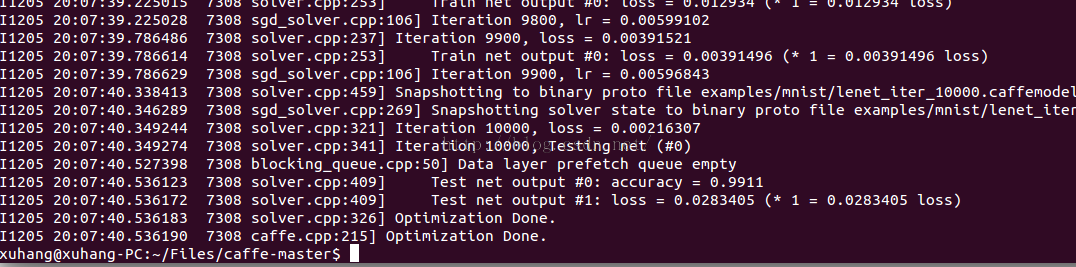
2. 重建lmdb文件。Caffe支持多种数据格式输入网络，包括Image(.jpg, .png等)，leveldb，lmdb，HDF5等，根据自己需要选择不同输入吧。

$ 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



注意：

1.每个账户下的～/.bashrc不一样，安装caffe时在～/.bashrc中设置了环境配置，因此必须进入root账户，然后运行python才是Anaconda环境的python，此时才能导入caffe

2.在wuyi账户中已设置了～/.bashrc中的环境变量，因此在wuyi账户中也可以用

在python中import caffe报错问题解决：

1. ImportError: numpy.core.multiarray failed to import

解决：进入caffe安装目录下的python文件夹，用pip uninstall numpy卸载numpy，重新安装pip install numpy

2.Intel MKL FATAL ERROR: Cannot load libmkl\_avx2.so or libmkl\_def.so

解决：输入这两条命令：

conda install nomkl numpy scipy scikit-learn numexpr

conda remove mkl mkl-service

3. ImportError: No module named google.protobuf.internal

解决参考：http://blog.csdn.net/woainiwss/article/details/50585665

4. ImportError: No module named caffe

解决：查看 /etc/profile中是否添加了export PYTHONPATH=/home/xuhang/Fiels/caffe-master/python:$PYTHONPATH， 如果没有添加就添加到末尾，然后运行source /etc/profile使配置文件生效