> 2017.04.04 更新日志：昨天电脑不知道出了什么问题，OpenCV 用不了了，我重装发现安装不了，然后发现Caffe也用不了了，无奈，重装。本来这个安装过程我已经很熟练了，但是这次安装遇到了很多之前没有遇到过的问题，弄了一下午终于安装好了，感觉应该再更新一波教程了。
>
> 2016.12.19 更新日志：我作死将系统升级到了10.12，然后又乖乖的重装了系统回到 10.11.6。环境需要重新配置，而这时的 CUDA 大版本已经来到了 8.0，因此现在更新这一教程至最新版本，同时使用了 Intel Parallel Studio Composer XE 2017 替代默认的 BLAS。

昨天看了一篇论文：

A Neural Algorithm of Artistic Style，主要内容是如何将一张图片的风格和另一张图片的内容结合起来的问题。是不是听着很熟？是的，最近很火的 **PRISMA** 就是做这个的。因为这是一个很有趣的内容，我就尝试动手做了一下，花了大约一个下午和一个晚上的时间，大功告成。

这篇 blog 的主要内容是总结我在 Mac 上安装 caffe 的详细过程，希望对需要的人有所帮助。

> Hardware&Software：
>
> - MacBook Pro (Retina, 15-inch, Mid 2014)
> - 2.5 GHz Intel Core i7
> - 16 GB 1600 MHz DDR3
> - OS X 10.11.6
> - Intel Iris Pro 1536 MB + NVIDIA GeForce GT 750M 2048MB

首先需要安装 caffe 需要的环境依赖：

- **Homebrew**

  ```
  /usr/bin/ruby -e "$(curl -fsSL
  https://raw.githubusercontent.com/Homebrew/install/master/install)"
  ```

- **安装 XCode 7.3.1 和 Command_Line_Tools_OS_X_10.11_for_Xcode_7.3.1**
- **CUDA** & **CUDNN**

  CUDA 是 NVIDIA 推出的一种集成技术，是该公司对 GPGPU 的正式名称。通过这个技术，用户可以利用 NVIDIA 的 GeForce 8 以后的 GPU 和教新的 Quadro GPU 进行计算。也是首次可以利用 GPU 作为 C-编译器的开发环境。CUDA可以兼容 OpenCL 或者自家的 C-编译器。无论是CUDA C-语言或是 OpenCL，指令最终都会被驱动程序转换成 PTX 代码，交由显示核心计算。更详细的介绍可以点击这里。

  我安装的 CUDA ToolKit 的版本是 CUDA 8.0，工具包中包含了 CUDA Driver，因此无需另外单独下载。CUDA 7 之前的版本需要 libstdc++，而 OS X 10.9+ 的编译器是 clang++，标准库是 libc++。需要更老版本的同学可以去官网查询详细的步骤。

Setup GPU for Mac

If you plan to build with GPU support you will need to make sure you have GNU coreutils installed via homebrew:

```
$ brew install coreutils
```

Next you will need to make sure you have a recent CUDA Toolkit installed by either downloading the package for your version of OSX directly from NVIDIA or by using the Homebrew Cask extension:

```
$ brew tap caskroom/cask
$ brew cask install cuda
```

Once you have the CUDA Toolkit installed you will need to setup the required environment variables by adding the following to your `~/.bash_profile`:

```
export CUDA_HOME=/usr/local/cuda
export DYLD_LIBRARY_PATH="$DYLD_LIBRARY_PATH:$CUDA_HOME/lib"
export PATH="$CUDA_HOME/bin:$PATH"
```

Finally, you will also want to install the CUDA Deep Neural Network (cuDNN v5) library which currently requires an Accelerated Computing Developer Program account. Once you have it downloaded locally, you can unzip and move the header and libraries to your local CUDA Toolkit folder:

```
$ sudo mv include/cudnn.h /Developer/NVIDIA/CUDA-8.0/include/
$ sudo mv lib/libcudnn* /Developer/NVIDIA/CUDA-8.0/lib
$ sudo ln -s /Developer/NVIDIA/CUDA-8.0/lib/libcudnn* /usr/local/cuda/lib/
```

To verify the CUDA installation, you can build and run deviceQuery to make sure it passes.

```
$ cp -r /usr/local/cuda/samples ~/cuda-samples
$ pushd ~/cuda-samples
$ make
$ popd
$ ~/cuda-samples/bin/x86_64/darwin/release/deviceQuery
```

安装完成之后，需要添加环境变量：

```
export PATH=/usr/local/cuda/bin:$PATH
export DYLD_LIBRARY_PATH=/usr/local/cuda/lib:$DYLD_LIBRARY_PATH
```

- **Library Path**

  为了使编译成功，我们需要指定 $DYLD_FALLBACK_LIBRARY_PATH，用于为 caffe 提供 CUDA, Python 和其他 libraries 的路径。e.g.

```
/usr/local/cuda/lib:$HOME/anaconda/lib:/usr/local/lib:/usr/lib
```

- ANACONDA

  我是用的是 Anaconda2，其自带最新版的 Python，也已经安装好了hdf5，caffe 官方也推荐我们使用 anaconda 作为我们的 python 环境。

  安装完成后，将 anaconda 添加到环境变量（路径因你的安装路径不同而异）：

  ```
  export PATH=~/anaconda/bin:$PATH
  ```

接下来开始安装 caffe 需要的通用依赖：

如果你像我一样使用的是 Anaconda Python，需要对 OpenCV formula 进行一下修改，在命令行中输入 ***brew edit opencv***，然后将

```
args << "-DPYTHON#{py_ver}_LIBRARY=#{py_lib}/libpython2.7.#{dylib}"
args << "-DPYTHON#{py_ver}_INCLUDE_DIR=#{py_prefix}/include/python2.7"
```

替换为

```
args << "-DPYTHON_LIBRARY=#{py_prefix}/lib/libpython2.7.dylib"
args << "-DPYTHON_INCLUDE_DIR=#{py_prefix}/include/python2.7"
```

接下来开始安装依赖：

```
brew install -vd snappy leveldb gflags glog szip lmdb
# need the homebrew science source for OpenCV and hdf5
brew tap homebrew/science
brew install hdf5 opencv
```

继续安装其他依赖：

```
# with Python pycaffe needs dependencies built from source
brew install --build-from-source --with-python -vd protobuf
brew install --build-from-source -vd boost boost-python
# without Python the usual installation suffices
brew install protobuf boost
```

- BLAS

  BLAS 已经通过 Accelerate / vecLib Framework 安装好。如果你想要更快的 CPU 计算或更好的稳定性，可以选择安装 OpenBLAS 或者 MKL。如果你也是在校大学生，可以使用学生邮箱下载Intel Parallel XE 申请下载安装。记得如果此处不使用默认设置，需要在后面的Makefile.config中设置 *BLAS := MKL*。

- **安装 Intel Parallel Studio Composer XE 2017**
  1. 使用学生邮箱注册，收到下载链接后下载，并使用给定的激活码完成激活
  2. 安装完成后，可以在 / opt / intel 目录下找到安装文件
  3. 在编辑 Makefile.config 中，将 BLAS 设置为 mkl，然后将 BLAS include path 和 BLAS lib 设置好。

     ```
     BLAS := mkl
     BLAS_INCLUDE := /opt/intel/mkl/include
     BLAS_LIB := /opt/intel/mkl/lib
     ```

     还有一个细节是，需要

     ```
     cd /opt/intel/mkl/lib/
     sudo ln -s . /opt/intel/mkl/lib/intel64
     ```

     因为在编译Caffe时Caffe会从MKL的intel64目录中去搜索mkl的库，但是在安装MKL后，MKL的lib目录下并没有intel64这个目录，所以需要建立一个intel64目录到lib目录的软链接。

- **别忘了在后面的caffe编译时的 Makefile.config 中取消 *USE_CUDNN := 1* 的注释。**

以上配置完成后，我们便可以开始进行 **caffe 的编译**了。

- 从 Github 上克隆 Caffe 的源代码
- Caffe 可以使用 make 或者 cmake 安装，make 由官方支持，cmake 由社区支持。我是用的是 make，因此下面主要介绍如何使用 make 进行编译。
  - 首先，修改 Makefile.config，如果你使用系统自带的 Python，不修改也是可以的，如果像我一样使用的 Anaconda Python，就需要将该文件中的关于使用 A_Python 部分的注释去掉，别忘了也得将使用系统自带 Python 的部分注释掉。

    ```
    cp Makefile.config.example Makefile.config
    # Adjust Makefile.config (for example, if using Anaconda Python, or if cuDNN is desired)
    make all
    make test
    make runtest
    make pycaffe
    make pytest
    ```

    - For CPU & GPU accelerated Caffe, no changes are needed.
    - For cuDNN acceleration using NVIDIA's proprietary cuDNN software, uncomment the `USE_CUDNN := 1` switch in `Makefile.config`. cuDNN is sometimes but not always faster than Caffe's GPU acceleration.
    - For CPU-only Caffe, uncomment `CPU_ONLY := 1` in `Makefile.config`.
    - To compile the Python and MATLAB wrappers do `make pycaffe` and `make`

`matcaffe` respectively. Be sure to set your MATLAB and Python paths in `Makefile.config` first!

- **Distribution**: run `make distribute` to create a `distribute` directory with all the Caffe headers, compiled libraries, binaries, etc. needed for distribution to other machines.
- **Speed**: for a faster build, compile in parallel by doing `make all -j8` where 8 is the number of parallel threads for compilation (a good choice for the number of threads is the number of cores in your machine).
- 在运行 make pytest 时，会遇到很多问题
  - Exception: “dot” not found in path in python on mac

    这个问题需要 Install graphviz on Mac OSX
  - ImportError: No module named google.protobuf.internal

    这个需要安装 pip install protobuf
  -

- 编译完成后，需要添加 $PYTHONPATH

```
export PYTHONPATH=/Users/poodar/Libs/caffe/python:$PYTHONPATH
```

至此，安装已经大功告成了，你可以通过 MNIST tutorial 或者 ImageNet model tutorial 开始你的 caffe 之旅了！

TroubleShoots:

- 如果遇到‘Unknown locale： utf-8’ 错误，在环境变量中加入

```
export LC_ALL=en_US.UTF-8
export LANG=en_US.UTF-8
```

- 如果在上面的编译过程的最后一步的测试时遇到

  Check failed: error == cudaSuccess (35 vs. 0) CUDA driver version is insufficient for CUDA runtime version

  错误，只需要更新一下你的CUDA driver 到最新版本即可，目前是8.0， 这个的版本不需要跟CUDA toolkit的版本保持一致。
- 如果在安装boost-python或boost这两个编译pycaffe的依赖的时候，下载速度过慢，可以通过两种方式解决
  - 手动下载文件， 然后复制到对应的目录
  - 直接输入 brew install boost boost-python 安装即可，省时省力。
- ImportError: No module named google.protobuf.internal

  重新安装 protobuf： pip install protobuf

- TypeError: **init**() got an unexpected keyword argument 'syntax'

  卸载 protobuf 后重新安装 pip install protobuf。这时会卸载掉旧版的2.6，安装新版的3.1版本的protobuf。
- 如果出现 ld: can't map file, errno=22 file '/usr/local/cuda/lib' for architecture x86_64，要么是 blas 没有配置好，要么是 CUDA 没有配置好
- 参考链接 Some notes about building Caffe RC3 with Mac OS X 10.11.3, Anaconda, CUDA 7.5, cuDNN 4, Intel MKL and MATLAB R2015b

<持续更新中>