> 现在 Tensorflow 已经出了 1.0 的正式版，此教程更新至此版本。
>
> 很早之前就出了 caffe 的安装步骤教程，迟迟没有出 tf 的教程的原因是之前装了很多次，但是完全装好 tf 只有一次，而且还是迷迷糊糊装好的，所以不能保证步骤一定能够复现。今天又试了一次，终于大功告成。

- 首先，要安装 **CUDA** 以及 **CUDNN**，教程在此，我已经在 caffe 的安装中总结过，此处不再赘述。
- Tensorflow的官方教程中有提供了四种安装方式，第一种 virtualenv 的方法很简单，但我尝试了一下之后，安装完成在 python 的交互式环境中输入 import tensorflow 遇到 segmentation fault，于是我改为 Install from Source。
- 需要注意的是，在安装过程中，建议开启控制台代理，下载速度会快很多。

Clone the TensorFlow repository

Start the process of building TensorFlow by cloning a TensorFlow repository.

To clone **the latest** TensorFlow repository, issue the following command:

```
$ git clone https://github.com/tensorflow/tensorflow
```

The preceding `git clone` command creates a subdirectory named `tensorflow`. After cloning, you may optionally build a **specific branch** (such as a release branch) by invoking the following commands:

```
$ cd tensorflow
$ git checkout Branch # where Branch is the desired branch
```

For example, to work with the `r1.0` release instead of the master release, issue the following command:

```
$ git checkout r1.0
```

Next, you must prepare your environment for Linux or Mac OS

Prepare environment for Mac OS

Before building TensorFlow, you must install the following on your system:

- bazel
- TensorFlow Python dependencies.
- optionally, NVIDIA packages to support TensorFlow for GPU.

Install bazel

If bazel is not installed on your system, install it now by following these directions.

Install python dependencies

To install TensorFlow, you must install the following packages:

- six
- numpy, which is a numerical processing package that TensorFlow requires.
- wheel, which enables you to manage Python compressed packages in the wheel (.whl) format.

You may install the python dependencies using pip. If you don't have pip on your machine, we recommend using homebrew to install Python and pip as documented here. If you follow these instructions, you will not need to disable SIP.

After installing pip, invoke the following commands:

```
$ sudo pip install six numpy wheel
```

Optional: install TensorFlow for GPU prerequisites

If you do not have brew installed, install it by following these instructions.

After installing brew, install GNU coreutils by issuing the following command:

```
$ brew install coreutils
```

If you want to compile tensorflow and have XCode 7.3 and CUDA 7.5 installed, note that Xcode 7.3 is not yet compatible with CUDA 7.5. To remedy this problem, do either of the following:

- Upgrade to CUDA 8.0.
- Download Xcode 7.2 and select it as your default by issuing the following command:

```
$ sudo xcode-select -s /Application/Xcode-7.2/Xcode.app
```

NOTE: Your system must fulfill the NVIDIA software requirements described in one of the following documents:

- Installing TensorFlow on Linux
- Installing TensorFlow on Mac OS

Configure the installation

The root of the source tree contains a bash script named `configure`. This script asks you to identify the pathname of all relevant TensorFlow dependencies and specify other build configuration options such as compiler flags. You must run this script *prior* to creating the pip package and installing TensorFlow.

If you wish to build TensorFlow with GPU, `configure` will ask you to specify the version numbers of Cuda and cuDNN. If several versions of Cuda or cuDNN are installed on your system, explicitly select the desired version instead of relying on the system default.

Here is an example execution of the `configure` script. Note that your own input will likely differ from our sample input:

```
$ cd tensorflow  # cd to the top-level directory created
$ ./configure
Please specify the location of python. [Default is /usr/bin/python]:
/usr/bin/python2.7
Please specify optimization flags to use during compilation when bazel
option "--config=opt" is specified [Default is -march=native]:
Do you wish to use jemalloc as the malloc implementation? [Y/n]
jemalloc enabled
Do you wish to build TensorFlow with Google Cloud Platform support? [y/N]
No Google Cloud Platform support will be enabled for TensorFlow
Do you wish to build TensorFlow with Hadoop File System support? [y/N]
No Hadoop File System support will be enabled for TensorFlow
Do you wish to build TensorFlow with the XLA just-in-time compiler
(experimental)? [y/N]
No XLA JIT support will be enabled for TensorFlow
Found possible Python library paths:
  /usr/local/lib/python2.7/dist-packages
  /usr/lib/python2.7/dist-packages
Please input the desired Python library path to use.  Default is
[/usr/local/lib/python2.7/dist-packages]
Using python library path: /usr/local/lib/python2.7/dist-packages
Do you wish to build TensorFlow with OpenCL support? [y/N] N
No OpenCL support will be enabled for TensorFlow
Do you wish to build TensorFlow with CUDA support? [y/N] Y
CUDA support will be enabled for TensorFlow
Please specify which gcc should be used by nvcc as the host compiler.
[Default is /usr/bin/gcc]:
Please specify the Cuda SDK version you want to use, e.g. 7.0. [Leave
empty to use system default]: 8.0
Please specify the location where CUDA 8.0 toolkit is installed. Refer to
README.md for more details. [Default is /usr/local/cuda]:
Please specify the cuDNN version you want to use. [Leave empty to use
system default]: 5
Please specify the location where cuDNN 5 library is installed. Refer to
README.md for more details. [Default is /usr/local/cuda]:
Please specify a list of comma-separated Cuda compute capabilities you
want to build with.
```

```
You can find the compute capability of your device at:
https://developer.nvidia.com/cuda-gpus.
Please note that each additional compute capability significantly
increases your build time and binary size.
[Default is: "3.5,5.2"]: 3.0
Setting up Cuda include
Setting up Cuda lib
Setting up Cuda bin
Setting up Cuda nvvm
Setting up CUPTI include
Setting up CUPTI lib64
Configuration finished
```

If you told `configure` to build for GPU support, then `configure` will create a canonical set of symbolic links to the Cuda libraries on your system. Therefore, every time you change the Cuda library paths, you must rerun the `configure` script before re-invoking the `bazel build` command.

Note the following:

- Although it is possible to build both Cuda and non-Cuda configs under the same source tree, we recommend running `bazel clean` when switching between these two configurations in the same source tree.
- If you don't run the `configure` script *before* running the `bazel build` command, the `bazel build` command will fail.

Build the pip package

To build a pip package for TensorFlow with CPU-only support, invoke the following command:

```
$ bazel build --config=opt
//tensorflow/tools/pip_package:build_pip_package
```

To build a pip package for TensorFlow with GPU support, invoke the following command:

```
$ bazel build --config=opt --config=cuda
//tensorflow/tools/pip_package:build_pip_package
```

Tip: By default, building TensorFlow from sources consumes a lot of RAM. If RAM is an issue on your system, you may limit RAM usage by specifying `--local_resources 2048,.5,1.0` while invoking `bazel`.

The `bazel build` command builds a script named `build_pip_package`. Running this script as follows will build a `.whl` file within the `/tmp/tensorflow_pkg` directory:

```
$ bazel-bin/tensorflow/tools/pip_package/build_pip_package
/tmp/tensorflow_pkg
```

Install the pip package

Invoke `pip install` to install that pip package. The filename of the `.whl` file depends on your platform. For example, the following command will install the pip package for TensorFlow 1.0.1 on Linux:

```
$ sudo pip install /tmp/tensorflow_pkg/tensorflow-1.0.1-py2-none-any.whl
```

NOTE on gcc version 5: the binary pip packages available on the TensorFlow website are built with gcc4 that uses the older ABI. To make the library compatible with the older abi you have to add `-cxxopt="-D_GLIBCXX_USE_CXX11_ABI=0"`

Validate your installation

Validate your TensorFlow installation by doing the following:

1. Start a terminal.
2. Change directory (`cd`) to any directory on your system other than the `tensorflow` subdirectory from which you invoked the `configure` command.
3. Invoke python:

```
$ python
```

1. Enter the following short program inside the python interactive shell:

```
>>> import tensorflow as tf
```

hello = tf.constant('Hello, TensorFlow!') sess = tf.Session() print(sess.run(hello))

```
  If the Python program outputs the following, then the installation is
successful and you can begin writing TensorFlow programs. (If you are new to
TensorFlow, see *Getting Started with TensorFlow*):

  ```shell
  Hello, TensorFlow!
```

If the system generates an error message instead of a greeting, see the next section.

Common installation problems

The installation problems you encounter typically depend on the operating system. See the "Common installation problems" section of one of the following guides:

- Installing TensorFlow on Linux
```

- Installing TensorFlow on Mac OS

Beyond the errors documented in those two guides, the following table notes additional errors specific to building TensorFlow. Note that we are relying on Stack Overflow as the repository for build and installation problems. If you encounter an error message not listed in the preceding two guides or in the following table, search for it on Stack Overflow. If Stack Overflow doesn't show the error message, ask a new question on Stack Overflow and specify the `tensorflow` tag.

Stack Overflow Link	Error Message
42013316	`ImportError: libcudart.so.8.0: cannot open shared object file: No such file or directory`
42013316	`ImportError: libcudnn.5: cannot open shared object file: No such file or directory`
35953210	Invoking `python` or `ipython` generates the following error: `ImportError: cannot import name pywrap_tensorflow`

以下步骤是历史版本，可以忽略。

- 从 github 上克隆 tf 的仓库，然后切换至项目的根目录下：

```
➜ /Users/poodar/Libs/tensorflow git:(master) ✗ >./configure
~/Libs/tensorflow ~/Libs/tensorflow
Please specify the location of python. [Default is /Users/poodar/anaconda/bin/python]:
Do you wish to build TensorFlow with Google Cloud Platform support? [y/N] N
No Google Cloud Platform support will be enabled for TensorFlow
Do you wish to build TensorFlow with Hadoop File System support? [y/N] y
Hadoop File System support will be enabled for TensorFlow
Found possible Python library paths:
  /Users/poodar/anaconda/lib/python2.7/site-packages
  /Users/poodar/Libs/caffe/python
Please input the desired Python library path to use.  Default is [/Users/poodar/anaconda/lib/python2.7/site-packages]

Using python library path: /Users/poodar/anaconda/lib/python2.7/site-packages
Do you wish to build TensorFlow with OpenCL support? [y/N] N
No OpenCL support will be enabled for TensorFlow
Do you wish to build TensorFlow with CUDA support? [y/N] y
CUDA support will be enabled for TensorFlow
Please specify which gcc should be used by nvcc as the host compiler. [Default is /usr/bin/gcc]:
Please specify the CUDA SDK version you want to use, e.g. 7.0. [Leave empty to use system default]: 8.0
Please specify the location where CUDA 8.0 toolkit is installed. Refer to README.md for more details. [Default is /usr/local/cuda]: 5
Invalid path to CUDA 8.0 toolkit. 5/lib/libcudart.8.0.dylib cannot be found
Please specify the CUDA SDK version you want to use, e.g. 7.0. [Leave empty to use system default]: 8.0
Please specify the location where CUDA 8.0 toolkit is installed. Refer to README.md for more details. [Default is /usr/local/cuda]:
Please specify the Cudnn version you want to use. [Leave empty to use system default]: 5
Please specify the location where cuDNN 5 library is installed. Refer to README.md for more details. [Default is /usr/local/cuda]:
Please specify a list of comma-separated Cuda compute capabilities you want to build with.
You can find the compute capability of your device at: https://developer.nvidia.com/cuda-gpus.
Please note that each additional compute capability significantly increases your build time and binary size.
[Default is: "3.5,5.2"]: 3.5
INFO: Starting clean (this may take a while). Consider using --expunge_async if the clean takes more than several minutes.
.......
____Loading package: tensorflow/cc
____Loading package: tensorflow/core/util/tensor_bundle
____Downloading http://bazel-mirror.storage.googleapis.com/github.com/google/protobuf/archive/008b5a228b37c054f46ba478ccafa5e855cb16db.tar.gz: 40,679 bytes
____Downloading http://bazel-mirror.storage.googleapis.com/github.com/google/protobuf/archive/008b5a228b37c054f46ba478ccafa5e855cb16db.tar.gz: 92,087 bytes
____Downloading http://bazel-mirror.storage.googleapis.com/github.com/google/protobuf/archive/008b5a228b37c054f46ba478ccafa5e855cb16db.tar.gz: 195,041 bytes
____Downloading http://bazel-mirror.storage.googleapis.com/github.com/google/protobuf/archive/008b5a228b37c054f46ba478ccafa5e855cb16db.tar.gz: 284,629 bytes
____Downloading http://bazel-mirror.storage.googleapis.com/github.com/google/protobuf/archive/008b5a228b37c054f46ba478ccafa5e855cb16db.tar.gz: 455,613 bytes
```

需要注意的是，MBP 上的 GT 750M 支持的 compute capabilities 是 3.0，所以在最后的配置中输入 3.0。

- 使用 bazel 安装下载好的 whl

- Mac GPU: Python 2 (build history) / Python 3 (build history)

值得注意的是，需要选择文件名形如 tensorflow_gpu-0.12.0rc0-cp27-cp27m-macosx_10_11_intel.whl 的文件，可以从 **build history** 中找到。

如果不出什么意外，到此就已经安装成功，是不是很简单？哈哈.....

安装成功后，进入 ipython 输入 import tensorflow 则会看到：

```
In [1]: import tensorflow
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcublas.8.0.dylib locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcudnn.5.dylib locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcufft.8.0.dylib locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcuda.1.dylib locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcurand.8.0.dylib locally

In [2]:
```

当然，不出意外的话，你是会遇到很多问题的，不要着急，慢慢一个错一个错的解决就好了，要有耐心。

<持续更新中>