武漢大學

Google

TensorFlow开发环境搭建

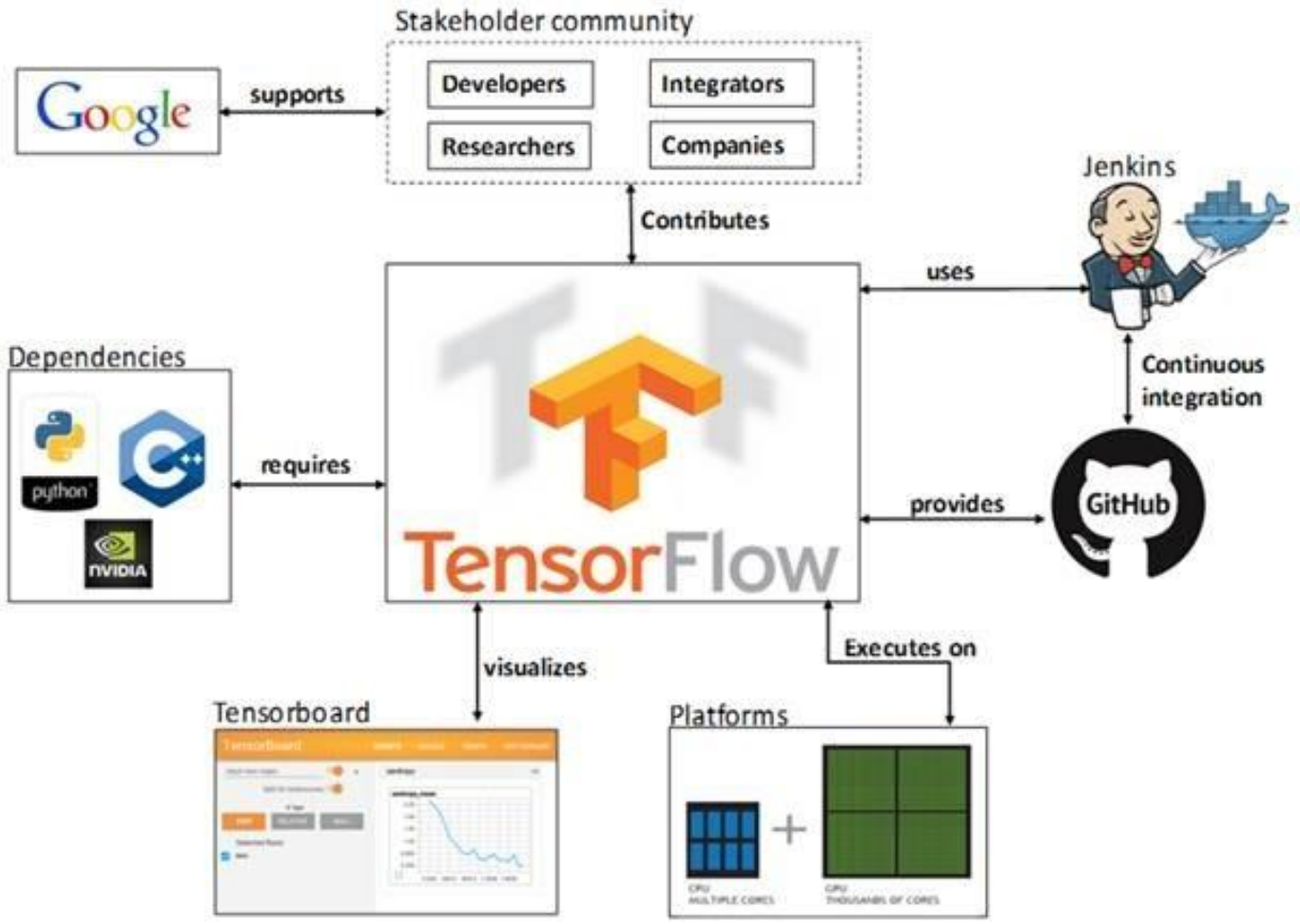
武汉大学 谷歌 联合实验室

目录

- TensorFlow运行环境
- Windows环境安装
- Linux环境安装
  - CPU Version
  - GPU Version
- Mac环境安装

TensorFlow运行环境

- TensorFlow™ 是一个采用数据流图（data flow graphs），用于数值计算的开源软件库



为什么选择tensorflow?

Library	Rank	Overall	Github	Stack Overflow	Google Results
tensorflow	1	10.87	4.25	4.37	2.24
keras	2	1.93	0.61	0.83	0.48
caffe	3	1.86	1.00	0.30	0.55
theano	4	0.76	-0.16	0.36	0.55
pytorch	5	0.48	-0.20	-0.30	0.98
sonnet	6	0.43	-0.33	-0.36	1.12
mxnet	7	0.10	0.12	-0.31	0.28
torch	8	0.01	-0.15	-0.01	0.17
cntk	9	-0.02	0.10	-0.28	0.17
dlib	10	-0.60	-0.40	-0.22	0.02
caffe2	11	-0.67	-0.27	-0.36	-0.04
chainer	12	-0.70	-0.40	-0.23	-0.07
paddlepaddle	13	-0.83	-0.27	-0.37	-0.20
deeplearning4j	14	-0.89	-0.06	-0.32	-0.51
lasagne	15	-1.11	-0.38	-0.29	-0.44
bigdl	16	-1.13	-0.46	-0.37	-0.30
dynet	17	-1.25	-0.47	-0.37	-0.42
apache singa	18	-1.34	-0.50	-0.37	-0.47
nvidia digits	19	-1.39	-0.41	-0.35	-0.64
matconvnet	20	-1.41	-0.49	-0.35	-0.58
tflearn	21	-1.45	-0.23	-0.28	-0.94
nervana neon	22	-1.65	-0.39	-0.37	-0.89
opennn	23	-1.97	-0.53	-0.37	-1.07

左表显示了标准化后的分数，其中值 1 表示高于平均值的一个标准偏差（平均值为 0）。 例如，Caffe 在 Github 中的活动是一个高于平均水准的标准差，而 torch 接近平均水平

该排名基于三个同等重要的部分：Github（star 和 fork），Stack Overflow（标签和问题）和 Google 搜索结果（总和以及季度增长率）

所有数据均截止于于 2017 年9月14日

https://github.com/thedataincubator/data-science-blogs/blob/master/output/deep_learning_data.csv

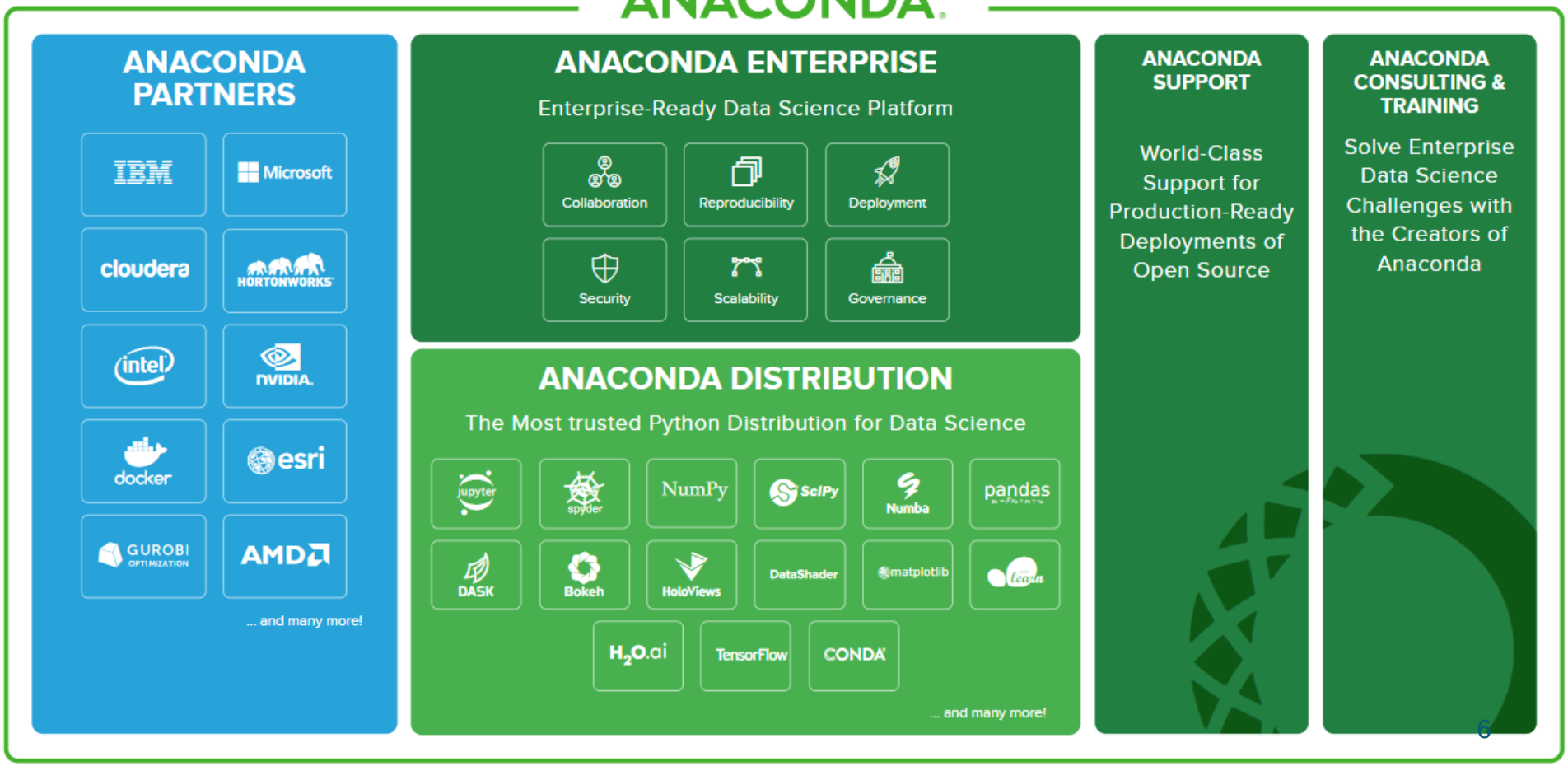
安装TensorFlow

- 安装方式

- ✓ 基于pip 工具
- ✓ 从源代码安装
- ✓ 基于Anaconda工具

Anaconda

- Anaconda是一个用于科学计算的Python发行版，支持 Linux, Mac, Windows系统，提供了包管理与环境管理的功能，可以很方便地解决多版本python 并存、切换以及各种第三方包安装问题



2. 在Windows安装TensorFlow开发环境

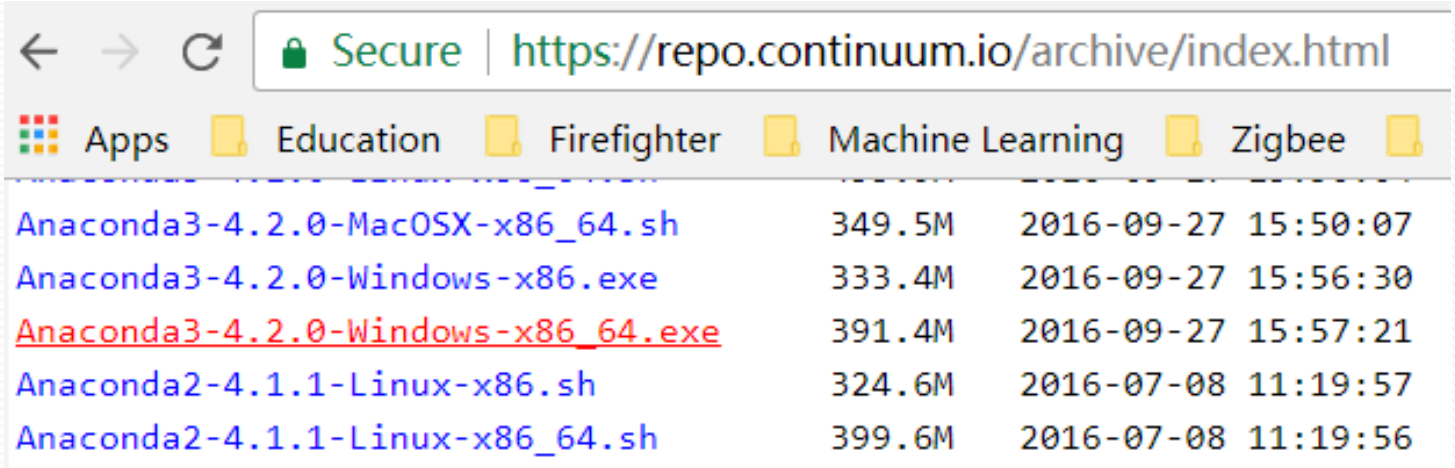
- Anaconda3.4.2.0 (Python 3.5.x) 下载
- Anaconda3 安装
- 国内镜像添加
- TensorFlow环境变量配置
- TensorFlow安装
- TensorFlow环境测试

2. 在Windows安装TensorFlow开发环境

➢ 2.1 Anaconda下载

https://repo.continuum.io/archive/index.html (Anaconda3 4.2.0)

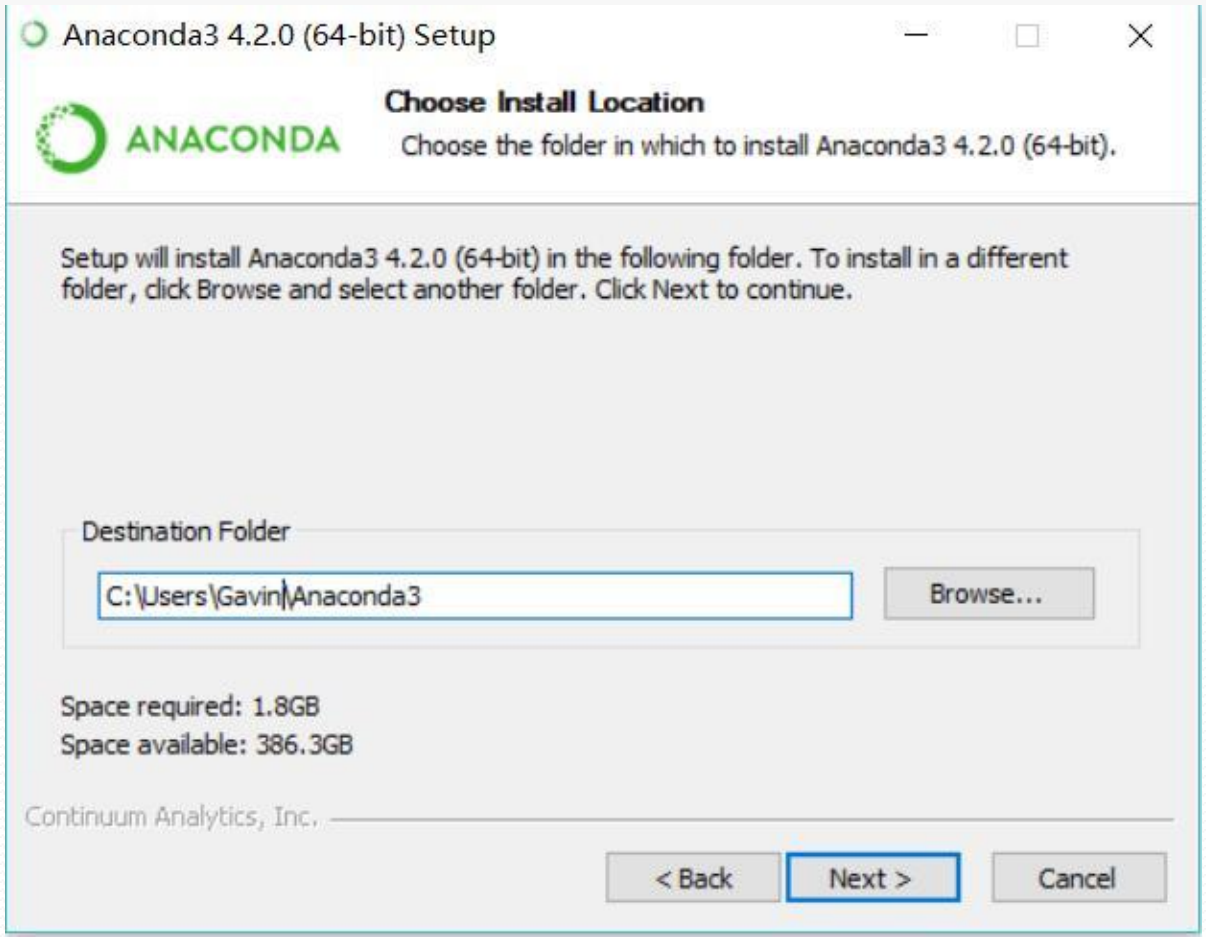
- 由于目前Windows下 TensorFlow仅对Python3.5.x 适配，所以不要下载最新的Anaconda



2. 在Windows安装TensorFlow开发环境

➢ 2.2 Anaconda安装

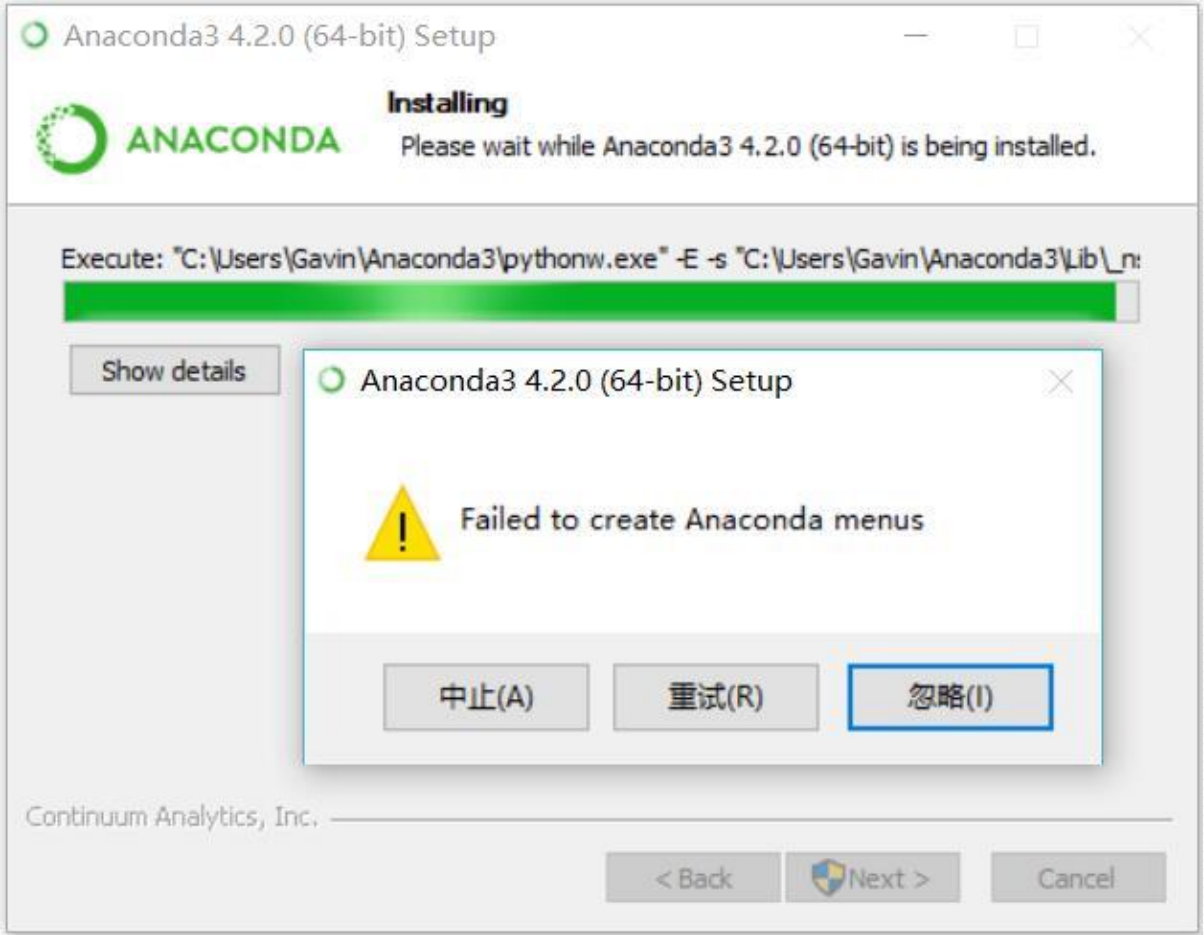
- 以管理员身份运行
- 安装路径不要有空格



2. 在Windows安装TensorFlow开发环境

➢ 2.2 Anaconda安装

• 可能出现的问题1 未能生成菜单 一般是由于安装路径空格 或之前安装过 并且未删除路径环境变量



2. 在Windows安装TensorFlow开发环境

➢ 2.2 Anaconda安装

• 可能出现的问题1 未能生成菜单 一般是由于安装路径空格 或之前安装过未删除路径环境变量

•解决办法 切换到Anaconda3目录，输入python .\lib_nsis.py mkmenus并运行

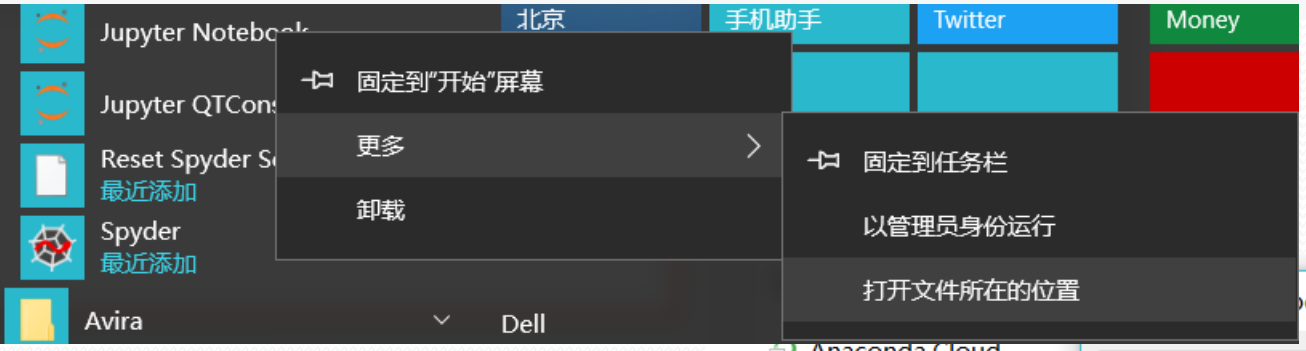
```
C:\Anaconda3>python .\lib\_nsis.py mkmenus
Processed C:\Anaconda3\Menu\anaconda_web.json successfully.
Processed C:\Anaconda3\Menu\console_shortcut.json successfully.
Processed C:\Anaconda3\Menu\ipython.json successfully.
Processed C:\Anaconda3\Menu\navigator_shortcut.json successfully.
Processed C:\Anaconda3\Menu\notebook.json successfully.
Processed C:\Anaconda3\Menu\qtconsole.json successfully.
Processed C:\Anaconda3\Menu\spyder_shortcut.json successfully.

C:\Anaconda3>
```

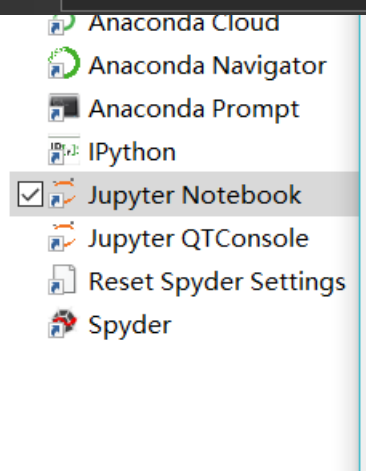
2. 在Windows安装TensorFlow开发环境

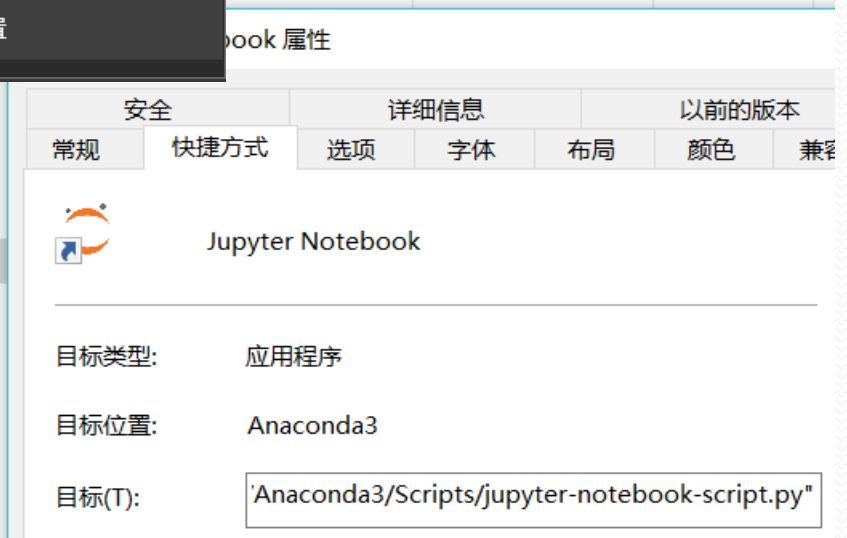
➢ 2.2 Anaconda安装

- 可能出现的问题2 能生成菜单 但个别项如 Jupyter等 运行后出现控制台黑屏然后闪退
  一般是由于启动项快捷方式路径冗余

•解决办法 右键点击启动项图标打开所在位置，删除属性中多余项





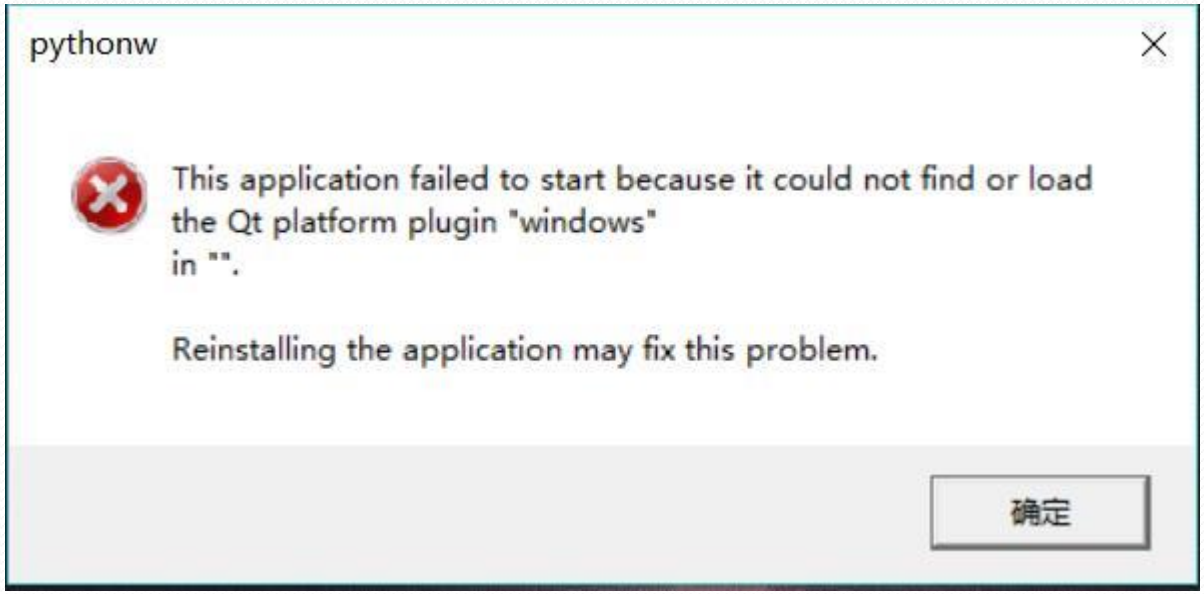


2. 在Windows安装TensorFlow开发环境

➢ 2.2 Anaconda安装

• 可能出现的问题3 能生成菜单 但QTConsole运行时提示插件未找到

• 解决办法 将\Anaconda3\Library\plugins目录下的platforms文件夹拷贝到\Anaconda3后重新打开Anaconda Navigator



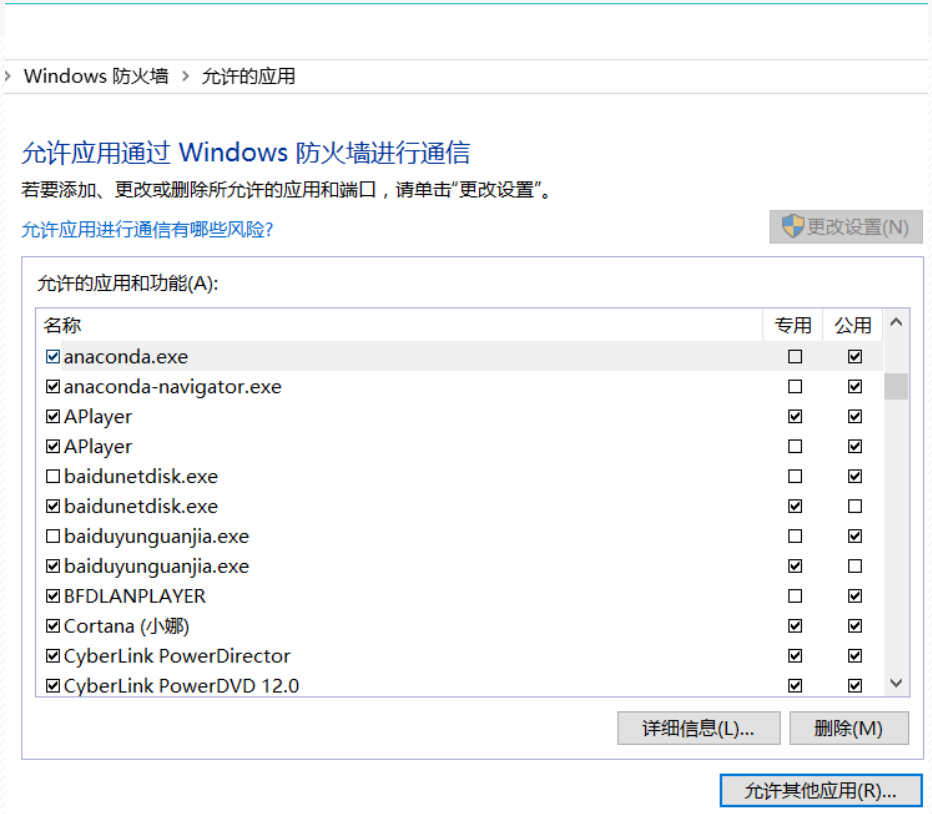
C:\Anaconda3\Library\plugins

2. 在Windows安装TensorFlow开发环境

➢ 2.2 Anaconda安装

- 可能出现的问题4 能生成菜单 但Spyder运行时一闪而过 无法运行

- 解决办法 1. 设置防火墙，解除对Spyder的阻止；
  2.把用户变量中 pythonpath改为path



2. 在Windows安装TensorFlow开发环境

➢ 2.2 Anaconda安装

- 可能出现的问题5 能生成菜单 但Navigator运行时一闪而过 无法运行

- 解决办法 1. 设置防火墙，解除对Navigator的阻止；
  2. 不启用Navigator 也不影响各个插件的使用

2. 在Windows安装TensorFlow开发环境

➢ 2.3 Anaconda添加依赖包国内镜像

•运行 Anaconda Prompt 命令行界面，输入以下命令
• conda --version 查看是否安装正确
• conda list 查看已安装包

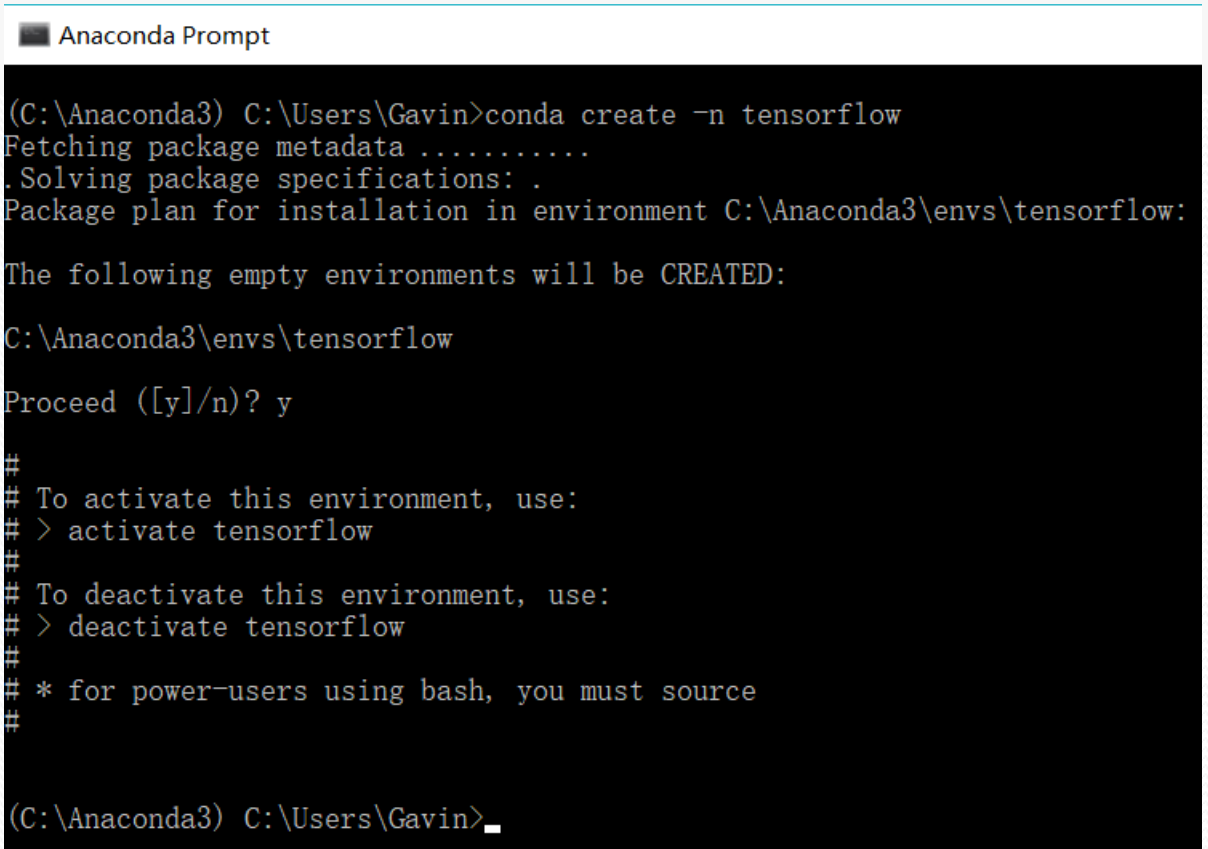
•conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
设置搜索时显示通道地址，输入以下命令
conda config --set show_channel_urls yes

2. 在Windows安装TensorFlow开发环境

➢ 2.4 TensorFlow环境变量配置

- 在Anaconda prompt 下输入

```
conda create -n tensorflow
```



在Windows安装TensorFlow开发环境

➢ 2.4 TensorFlow环境变量配置

- 激活Anaconda环境
- 在Anaconda prompt 下输入

activate tensorflow

若正确激活，则在命令行提示符前显示环境变量

```
(C:\Anaconda3) C:\Users\Gavin>activate tensorflow

(tensorflow) C:\Users\Gavin>_
```

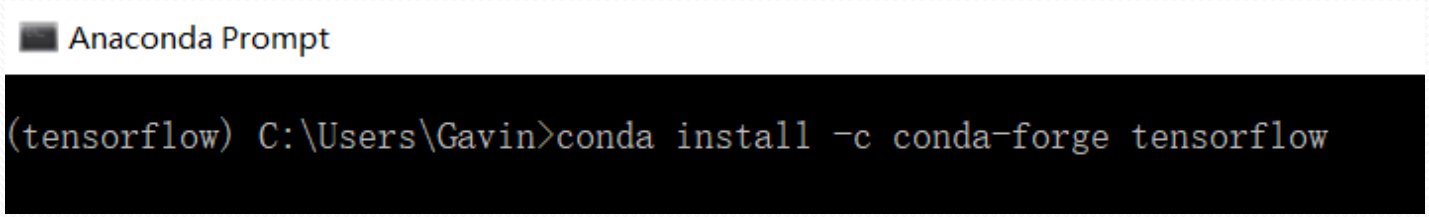
在Windows安装TensorFlow开发环境

➢ 2.5 TensorFlow安装

- 在Anaconda Prompt下输入(建议管理员模式运行)
- conda install -c conda-forge tensorflow

或者

- pip install tensorflow



在Windows安装TensorFlow开发环境

➢ 2.5 TensorFlow安装

- 在Anaconda Prompt下输入 (建议管理员模式运行)
- conda install -c conda-forge tensorflow

或者

- pip ir

Anaconda Prompt - conda install -c conda-forge tensorflow

```
The following NEW packages will be INSTALLED:

    backports:          1.0-py36_1          conda-forge
    backports.weakref:  1.0rc1-py36_1       conda-forge
    bleach:             1.5.0-py36_0        conda-forge
    certifi:            2017.7.27.1-py36_0  conda-forge
    html5lib:           0.9999999-py36_0    conda-forge
    markdown:           2.6.9-py36_0        conda-forge
    mkl:                2017.0.3-0          https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free
    mock:               2.0.0-py36_0        conda-forge
    numpy:              1.13.1-py36_0       https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free
    pbr:                3.1.1-py36_0        conda-forge
    pip:                9.0.1-py36_0        conda-forge
    protobuf:           3.4.0-py36_vc14_1   conda-forge                                    [vc14]
    python:             3.6.3-1             conda-forge
    setuptools:         36.6.0-py36_1       conda-forge
    six:                1.11.0-py36_1       conda-forge
    tensorflow:         1.3.0-py36_0        conda-forge
    vs2015_runtime:     14.0.25420-0        conda-forge
    webencodings:       0.5-py36_0          conda-forge
    werkzeug:           0.12.2-py_1         conda-forge
    wheel:              0.30.0-py_1         conda-forge
    wincertstore:       0.2-py36_0          conda-forge
    zlib:               1.2.11-vc14_0       conda-forge                                    [vc14]

Proceed ([y]/n)?
```

在Windows安装TensorFlow开发环境

➢ 2.5 TensorFlow安装

•官方网址安装 在Anaconda Prompt下输入

•CPU: **pip install --ignore-installed --upgrade**
https://storage.googleapis.com/tensorflow/windows/cpu/
tensorflow-1.4.0-cp35-cp35m-win_amd64

GPU: **pip install --ignore-installed --upgrade**
https://storage.googleapis.com/tensorflow/windows/gpu/
tensorflow-1.4.0-cp35-cp35m-win_amd64

一般情况下，官网无法访问或速度很慢

```
Successfully installed bleach-1.5.0 enum34-1.1.6 html5lib-0.9999999 markdown-2.6.9 numpy-1.13.3 protobuf-3.5.0 tensorflo
w-1.4.0 tensorflow-tensorboard-0.4.0rc3
You are using pip version 8.1.2, however version 9.0.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

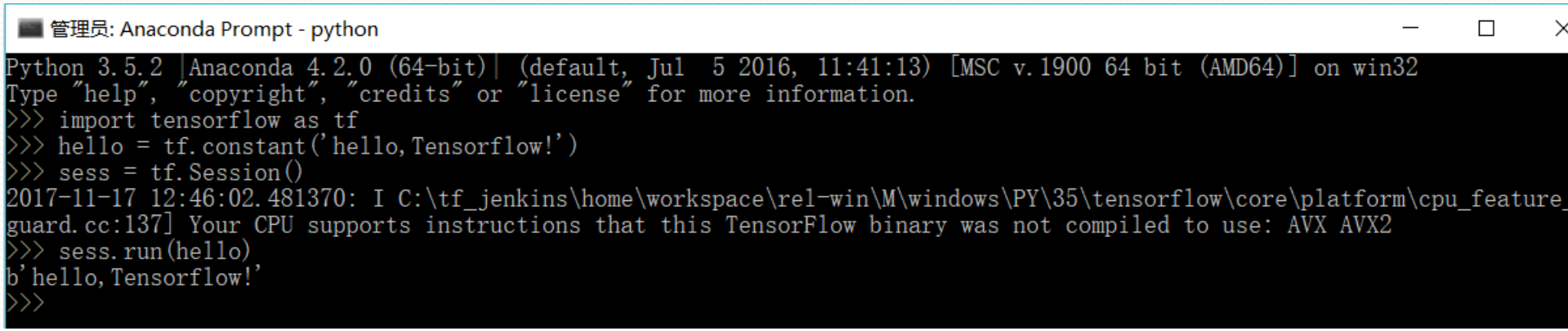
在Windows安装TensorFlow开发环境

➢ 2.6 TensorFlow环境测试

•在Anaconda Prompt下输入Python 启动运行时环境，输入代码

```
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
sess.run(hello)        # print(sess.run(hello))
```

能打印出 hello，TensorFlow！ 说明成功



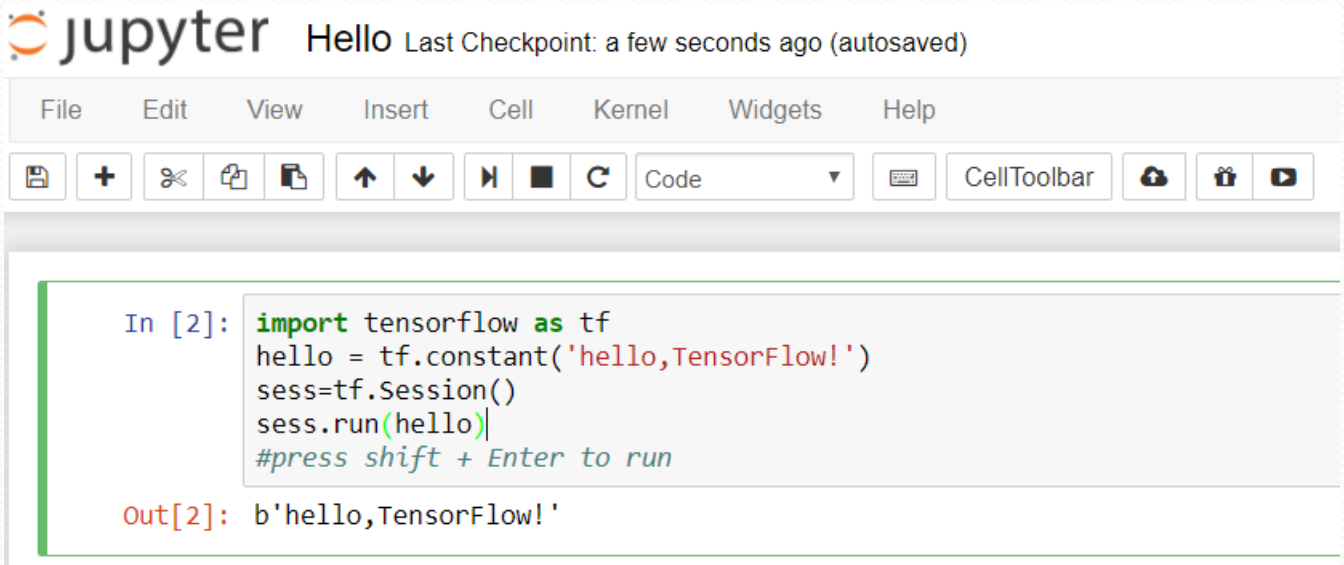
在Windows安装TensorFlow开发环境

➢ 2.6 TensorFlow环境测试

•启动Jupyter，输入代码

```
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
sess.run(hello)        # print(sess.run(hello))
#按 shift +enter 运行程序
```

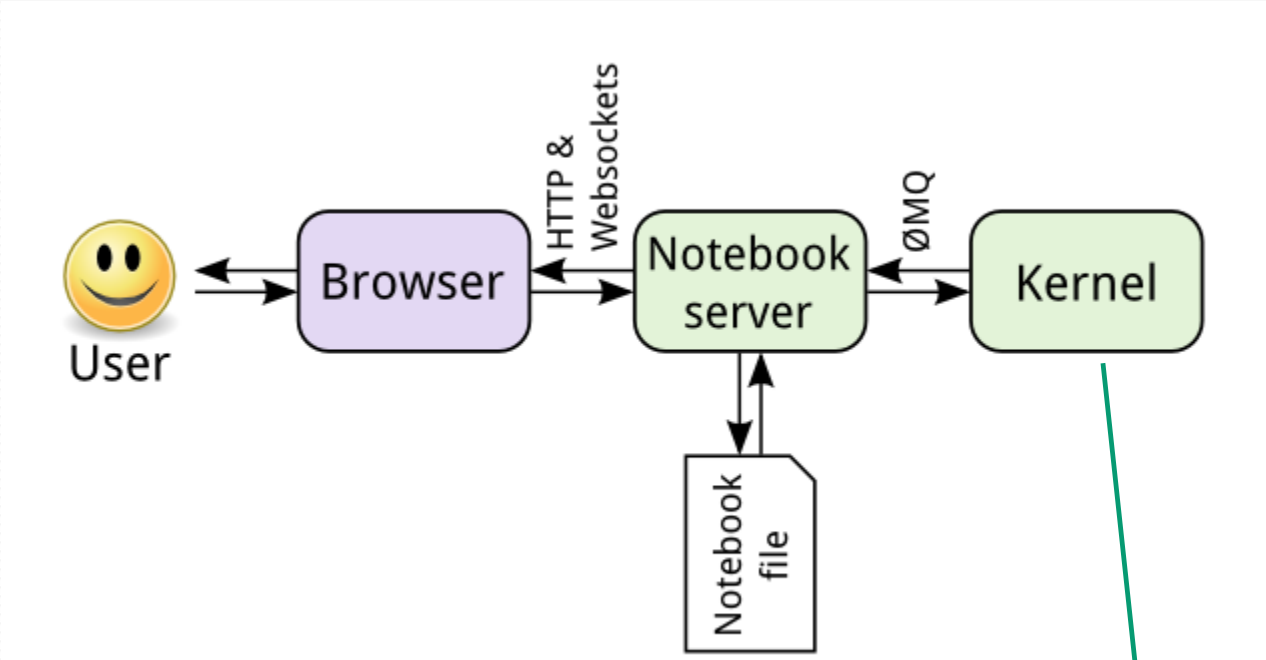
能打印出 hello，TensorFlow！ 说明成功



在Windows安装TensorFlow开发环境

➢ 2.6 TensorFlow环境测试

•数据报告工具Jupyter 的工作机制



目录

- TensorFlow运行环境
- Windows环境安装
- Linux环境安装
  - CPU Version
  - GPU Version
- Mac环境安装

Linux下安装TensorFlow开发环境

- Anaconda下载并安装

https://repo.continuum.io/archive/index.html (如果python为3.6 选最新版本的Anaconda)

- 国内镜像地址配置

- conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/

```
# 设置搜索时显示通道地址
conda config --set show_channel_urls yes
```

- 环境配置 在自带终端下(shell) 下输入

```
$ conda create -n tensorflow
```

- 激活Anaconda环境

```
$ source activate tensorflow
(tensorflow)$  # Your prompt should change
```

Linux下安装TensorFlow开发环境

- 安装TensorFlow

```
(tensorflow)$ pip install --ignore-installed --upgrade $TF_PYTHON_URL
```

python 3.4 3.5 3.6 版本
cpu: https://storage.googleapis.com/tensorflow/linux/cpu/
tensorflow-1.4.0-py3-none-any.whl

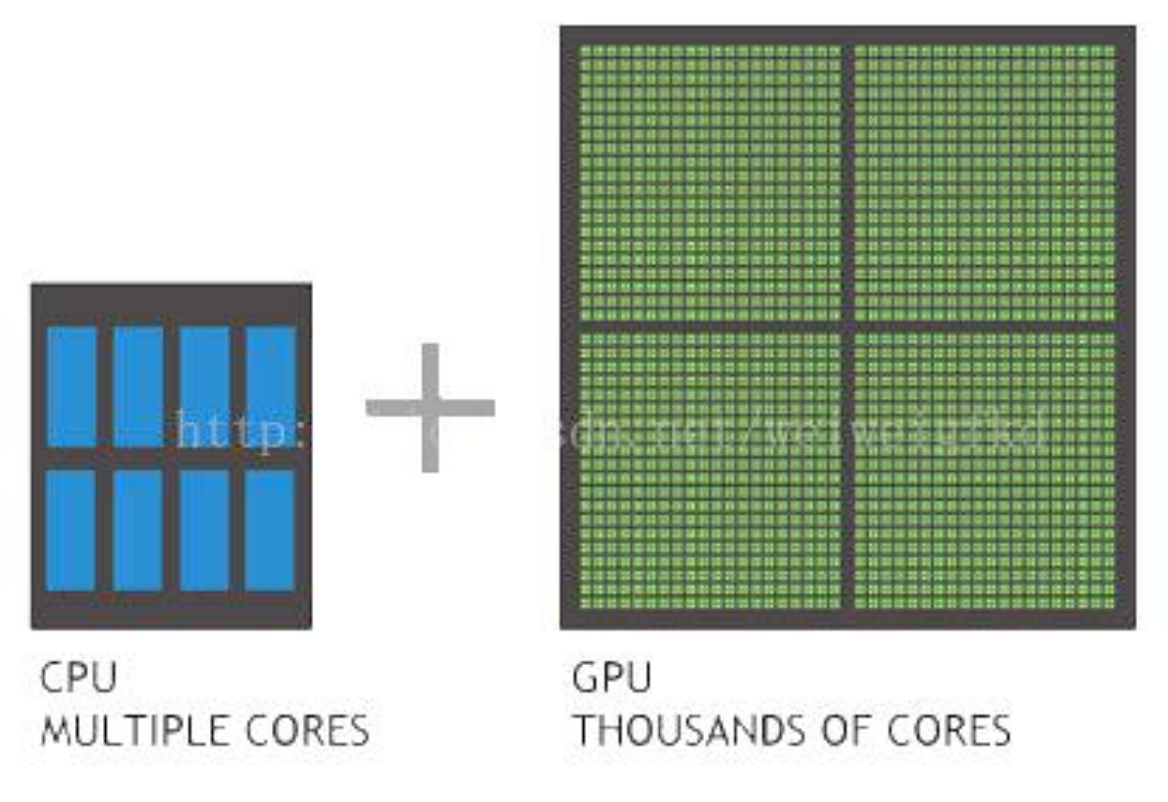
目录

- TensorFlow运行环境
- Windows环境安装
- Linux环境安装
  - CPU Version
  - GPU Version
- Mac环境安装

GPU版本的Tensorflow安装、配置与测试

GPU加速原理

当前最顶级的CPU只有4核或者6核，模拟出8个或者12个处理线程来进行运算，但是普通级别的GPU就包含了成百上千个处理单元，高端的甚至更多，这对于多媒体计算中大量的重复处理过程有着天生的优势。而CNN中的卷积操作则正是这种大量的、重复的计算。



硬件需求

NVIDIA GPU Compute Capability >= 3.5

GeForce Desktop Products

GPU	Compute Capability
NVIDIA TITAN Xp	6.1
NVIDIA TITAN X	6.1
GeForce GTX 1080 Ti	6.1
GeForce GTX 1080	6.1
GeForce GTX 1070	6.1
GeForce GTX 1060	6.1
GeForce GTX 1050	6.1
GeForce GTX TITAN X	5.2

GeForce Notebook Products

GPU	Compute Capability
GeForce GTX 1080	6.1
GeForce GTX 1070	6.1
GeForce GTX 1060	6.1
GeForce GTX 980	5.2
GeForce GTX 980M	5.2
GeForce GTX 970M	5.2
GeForce GTX 965M	5.2
GeForce GTX 960M	5.0

软件选择

描述	版本
Ubuntu	16.04 LTS
Cuda ToolKit	8.0
Cudnn	5.1
Tensorflow	1.2.0
Anaconda	2
Python	2.7

CUDA与CUDNN

CUDA(Compute Unified Device Architecture)
是英伟达公司推出的一种基于新的并行编程模型和指令集架构的通用计算架构，它能利用英伟达GPU的并行计算引擎，比CPU更高效的解决许多复杂计算任务。

CUDNN(CUDA Deep Neural Network library)
是用于深度神经网络的GPU加速库。它强调性能、易用性和低内存开销。NVIDIA cuDNN可以集成到更高级别的机器学习框架中，如加州大学伯克利分校的CAFFE框架，Google的Tensorflow框架。

NVIDIA驱动安装

1. 注销Ubuntu的图形界面，并关闭lightdm服务 ：

sudo service lightdm stop

2. 执行命令安装对应版本驱动：

sudo apt-get install nvidia-367

3. 根据提示安装完毕后重启：

sudo reboot

注意：尽量安装与Cuda兼容的驱动版本

CUDA 的安装

1. 同样注销Ubuntu的图形界面，并关闭lightdm服务 ；

sudo service lightdm stop

2. 禁用系统自带的nouveau显卡驱动；

3. 在NVIDIA官网下载8.0版本的cuda(run格式)，切换到相关目录运行：

sudo sh ./cuda_8.0.44_linux.run

a.在安装过程中，选择不安装NVIDIA驱动和OpenGL库，否则会遇到登录循环问题。需要重新卸载驱动。
b.如果遇到磁盘空间不足问题的报错，可以通过在命令后面追加参数 --tmpdir=/home/ 设置临时目录的方式解决。

CUDA 的安装

4. 配置cuda环境变量（即通过sudo gedit ~/.bashrc中追加）：

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

5. 查看cuda安装版本：

```
wgz@lazysheep:~$ nvcc -V
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2016 NVIDIA Corporation
Built on Sun_Sep__4_22:14:01_CDT_2016
Cuda compilation tools, release 8.0, V8.0.44
wgz@lazysheep:~$
```

CUDA 的测试

1. 编译Cuda的示例文件，等待10分钟左右，完成无报错即可：

```
wgz@lazysheep:~$ cd NVIDIA_CUDA-8.0_Samples/
wgz@lazysheep:~/NVIDIA_CUDA-8.0_Samples$ ls
0_Simple     2_Graphics  4_Finance      6_Advanced       common    Makefile
1_Utilities  3_Imaging   5_Simulations  7_CUDALibraries  EULA.txt
wgz@lazysheep:~/NVIDIA_CUDA-8.0_Samples$ make
make[1]: Entering directory '/home/wgz/NVIDIA_CUDA-8.0_Samples/0_Simple/clock'
"/usr/local/cuda-8.0"/bin/nvcc -ccbin g++ -I../../common/inc  -m64    -gencode a
rch=compute_20,code=sm_20 -gencode arch=compute_30,code=sm_30 -gencode arch=comp
ute_35,code=sm_35 -gencode arch=compute_37,code=sm_37 -gencode arch=compute_50,c
ode=sm_50 -gencode arch=compute_52,code=sm_52 -gencode arch=compute_60,code=sm_6
0 -gencode arch=compute_60,code=compute_60 -o clock.o -c clock.cu
```

2. 运行示例中的deviceQurey，如果结果为pass则说明安装成功，见下页图示：

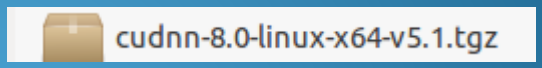
Cuda 的测试

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

Detected 1 CUDA Capable device(s)

Device 0: "GeForce GTX 1050 Ti"
  CUDA Driver Version / Runtime Version          8.0 / 8.0
  CUDA Capability Major/Minor version number:    6.1
  Total amount of global memory:                 4037 MBytes (4232904704 bytes)
  ( 6) Multiprocessors, (128) CUDA Cores/MP:     768 CUDA Cores
  GPU Max Clock rate:                            1392 MHz (1.39 GHz)
  Memory Clock rate:                             3504 Mhz
  Memory Bus Width:                              128-bit
  L2 Cache Size:                                 1048576 bytes
  Maximum Texture Dimension Size (x,y,z)         1D=(131072), 2D=(131072, 65536)
, 3D=(16384, 16384, 16384)
  Maximum Layered 1D Texture Size, (num) layers  1D=(32768), 2048 layers
  Maximum Layered 2D Texture Size, (num) layers  2D=(32768, 32768), 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 2 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 Domain ID / Bus ID / location ID:   0 / 1 / 0
  Compute Mode:
     < Default (multiple host threads can use ::cudaSetDevice() with device simu
ltaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 8.0, CUDA Runtime Versi
on = 8.0, NumDevs = 1, Device0 = GeForce GTX 1050 Ti
Result = PASS
```

Cudnn 的安装

1. 解压下载好的cudnn包，会生成cuda/include和cuda/lib64：

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



2. 将cuda/include目录中的cudnn.h文件拷贝到/usr/local/cuda-8.0/include/目录下：

   sudo cp cuda/include/cudnn.h /usr/local/cuda-8.0/include

3. 将cuda/lib64目录中的库拷贝到/usr/local/cuda-8.0/lib64/目录下；

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

4. 赋予相关权限；

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

Anaconda 的安装与配置

1. 运行下载好的.sh文件根据提示进行安装：

```
sudo ./Anaconda2-5.0.0.1-Linux-x86_64.sh
```

2. 创建新的python解释环境（这里只是起名是tensorFlow，叫什么都可以）：

```
conda create  -n tensorflow python=2.7
```

3. 激活新的解释环境：

```
source activate tensorflow
```

4. 在新的解释器中通过pip安装tensorflow包及相关依赖：

```
pip install --ignore-installed --upgrade tensorflow-gpu=1.2
```

```
Successfully installed backports.weakref-1.0.post1 bleach-1.5.0 funcsigs-1.0.2 h
tml5lib-0.9999999 markdown-2.6.9 mock-2.0.0 numpy-1.13.3 pbr-3.1.1 protobuf-3.4.
0 setuptools-36.5.0.post20170921 six-1.11.0 tensorflow-gpu-1.2.0 werkzeug-0.12.2
 wheel-0.30.0
```

5. 安装完成后退出当前解释环境；

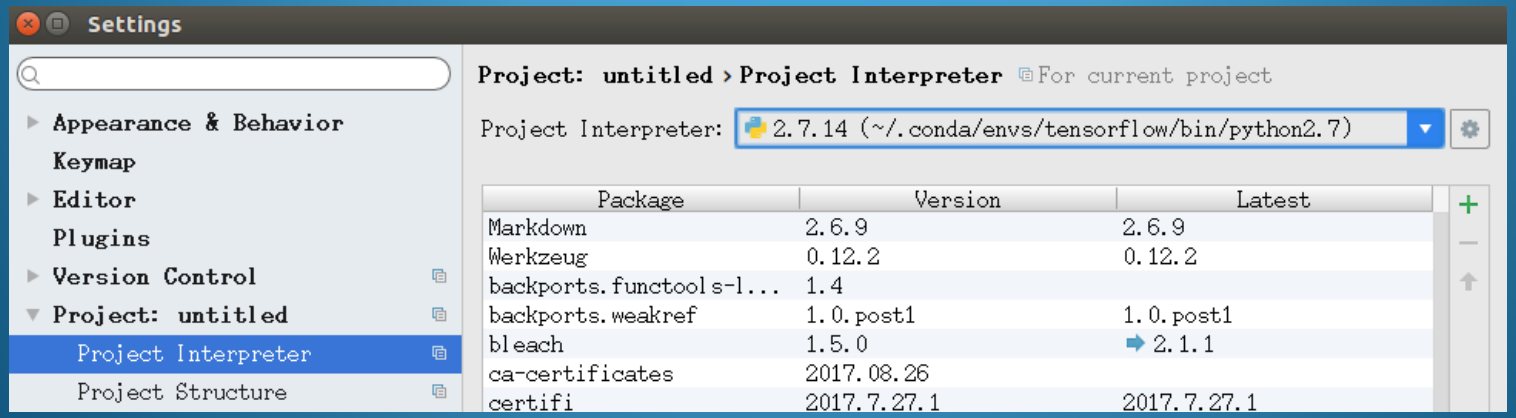
```
source deactivate tensorflow
```

Pycharm 的安装与配置

1. 运行下载好的Pycharm/bin中的.sh文件根据提示进行安装。

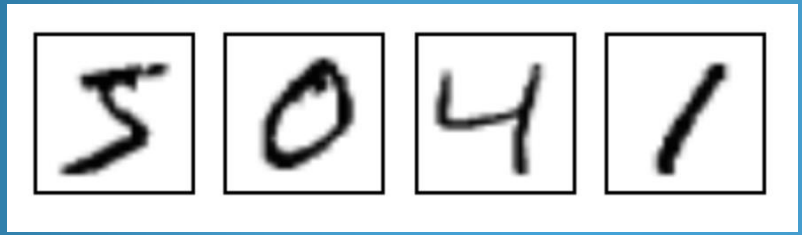
2. 很关键的一步：
选择上一步自定义的Python解释器，可以在环境中通过which python命令查看Python解释器目录：

```
(tensorflow) wgz@lazysheep:~$ which python
/home/wgz/.conda/envs/tensorflow/bin/python
```



Run MNIST Demo

以官方文档中的MNIST手写字体识别demo为例进行运行，测试TensorFlow的安装。同时对CPU和GPU下的模型迭代时间进行对比。

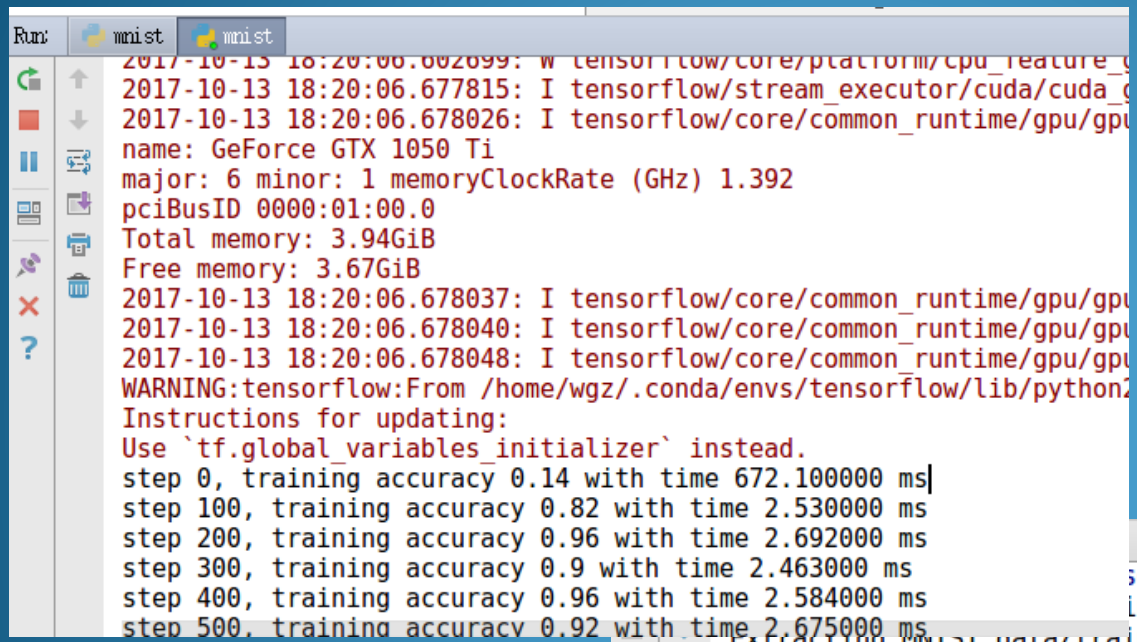


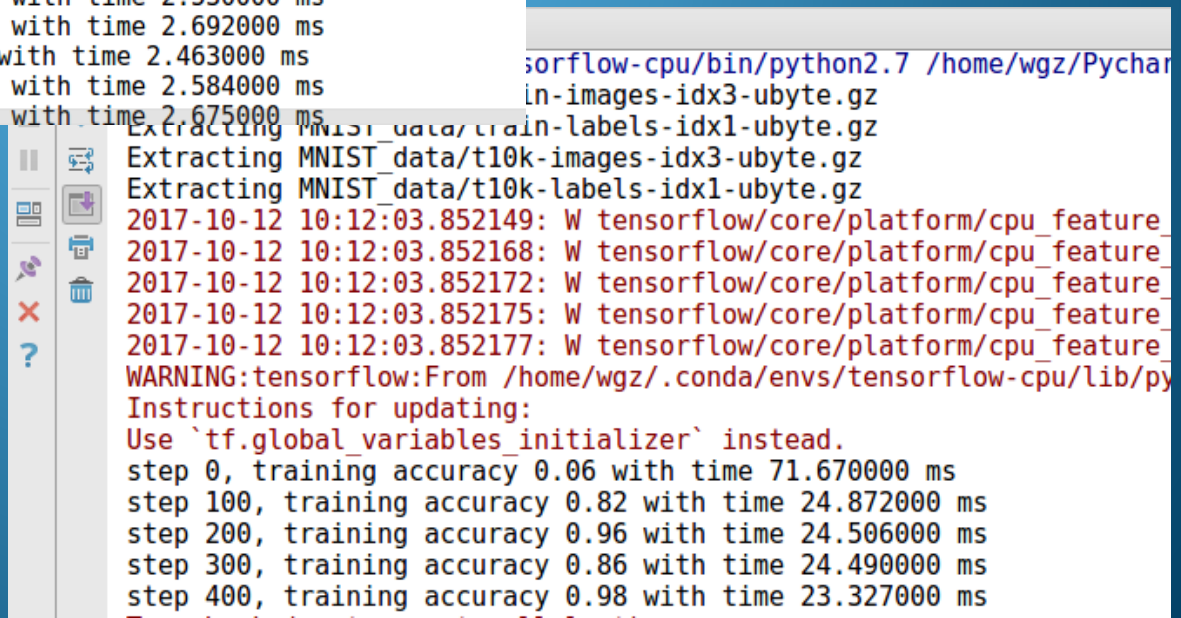
文件	内容
train-images-idx3-ubyte.gz	训练集图片 - 55000 张 训练图片, 5000 张 验证图片
train-labels-idx1-ubyte.gz	训练集图片对应的数字标签
t10k-images-idx3-ubyte.gz	测试集图片 - 10000 张 图片
t10k-labels-idx1-ubyte.gz	测试集图片对应的数字标签

Run MNIST Demo

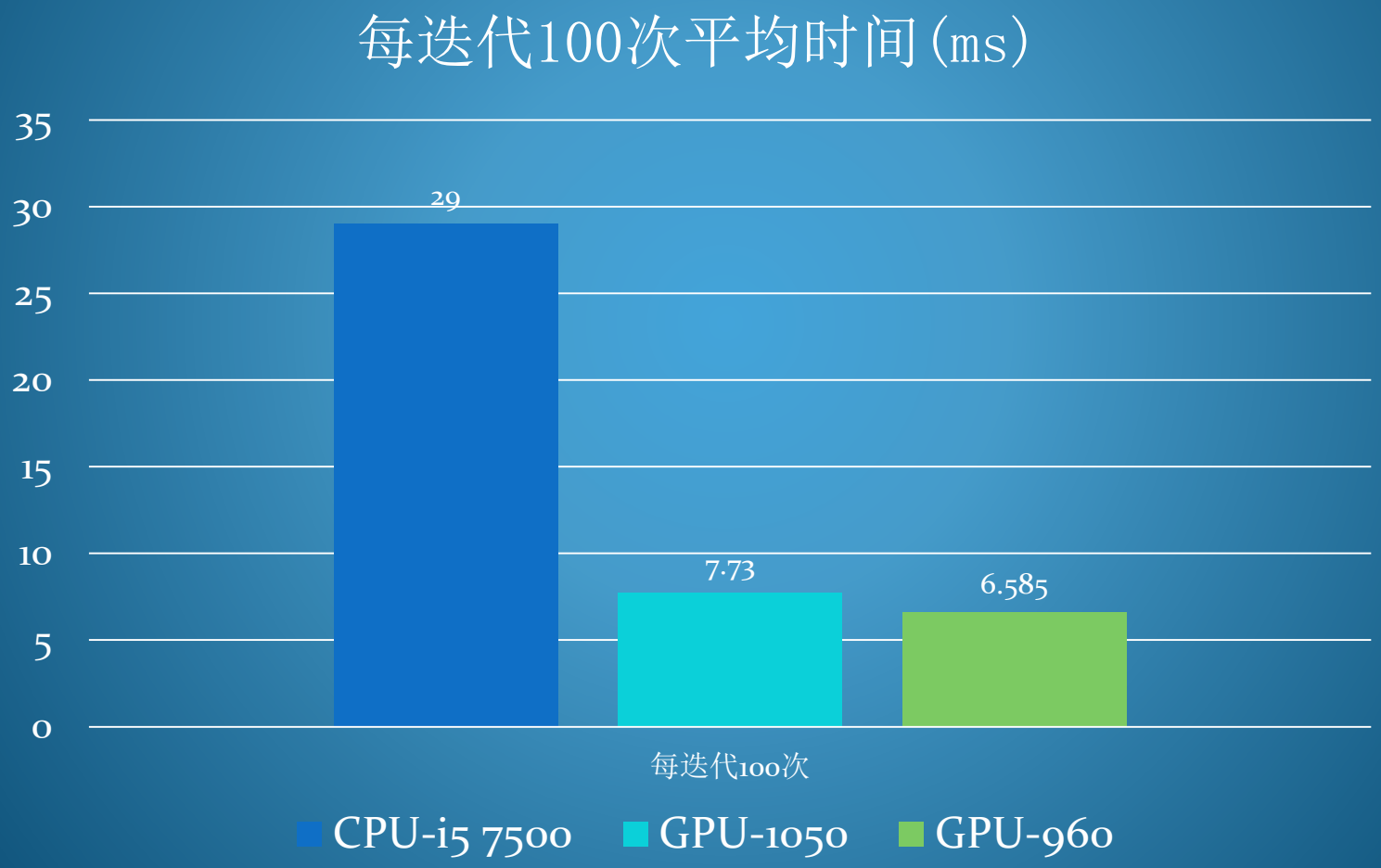
```
Every 1.0s: nvidia-smi                                  Thu Oct 12 10:06:25 2017

Thu Oct 12 10:06:25 2017
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 375.66                 Driver Version: 375.66                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX 105...  Off  | 0000:01:00.0      On |                  N/A |
| 31%   51C    P0    66W /  75W |    539MiB /  4036MiB |     91%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID  Type  Process name                               Usage      |
|=============================================================================|
|    0       966    G   /usr/lib/xorg/Xorg                           117MiB |
|    0      1903    G   compiz                                       115MiB |
|    0      2196    G   fcitx-qimpanel                                 6MiB |
|    0      2270    G   /usr/lib/firefox/firefox                       1MiB |
|    0     23569    C   .../wgz/.conda/envs/tensorflow/bin/python2.7   295MiB |
+-----------------------------------------------------------------------------+
```

Run MNIST Demo

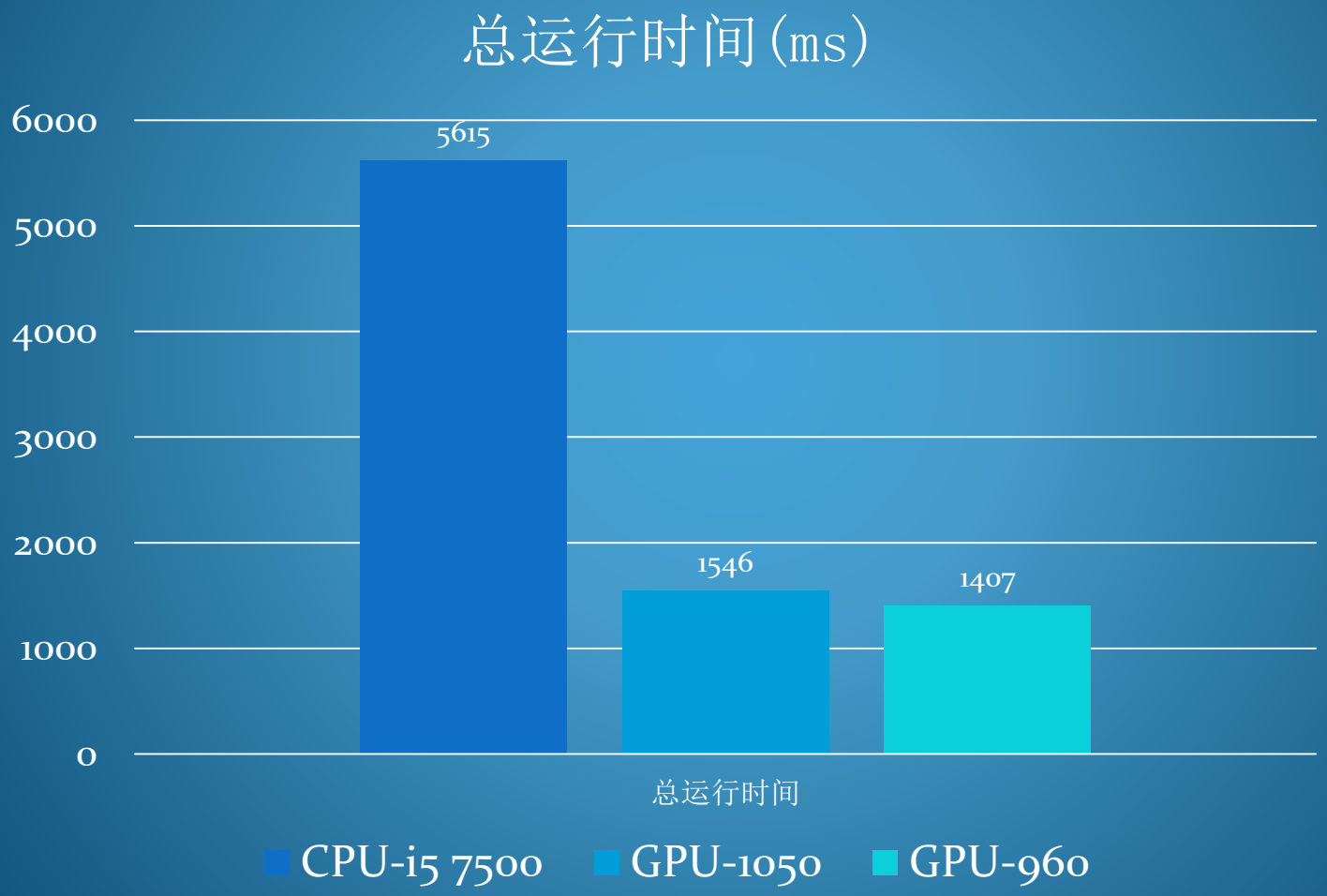




Run MNIST Demo



Run MNIST Demo



目录

- TensorFlow运行环境
- Windows环境安装
- Linux环境安装
  - CPU Version
  - GPU Version
- Mac环境安装

MAC下安装TensorFlow开发环境

- Anaconda下载

https://repo.continuum.io/archive/index.html

(如果python为3.6 选最新版本的Anaconda)

- 在自带终端下(shell) 下输入

```
$ conda create -n tensorflow
```

MAC下安装TensorFlow开发环境

- 激活Anaconda环境

```
$ source activate tensorflow
(tensorflow)$  # Your prompt should change
```

- 安装TensorFlow

```
(tensorflow)$ pip install --ignore-installed --upgrade $TF_PYTHON_URL
```

python 3.4 3.5 3.6
cpu: https://storage.googleapis.com/tensorflow/mac/cpu/tensorflow-1.4.0-py3-none-any.whl

gpu: https://storage.googleapis.com/tensorflow/mac/gpu/tensorflow_gpu-1.4.0-py3-none-any.whl

Thanks