



西安电子科技大学
XIDIAN UNIVERSITY

仿真FGSM对抗攻击

Mengdie Huang
November 22, 2019



1 搭建深度学习环境

2 Github获取FGSM源码

3 Github获取数据集

4 输入训练集

5 输入测试集



目标：用TensorFlow生成对抗样本

作者信息

Anish Athalye: MIT在读博士生, 对分布式系统、系统安全及人工智能感兴趣。



学术: <http://www.anish.io/>

Email: aathalye@mit.edu

Github: <https://github.com/anishathalye>

本文由北邮@爱可可-爱生活老师推荐, 阿里云云栖社区组织翻译。

文章原标题《A Step-by-Step Guide to Synthesizing Adversarial Examples》, 作者: Anish Athalye,



Deep Learning/ Machine learning/ Artificial Intelligence的关系

Deep Learning \subseteq Machine learning \rightarrow Artificial Intelligence

Artificial Intelligence: 让机器具有学习和认知的能力。

Machine learning: 实现AI的一种方法。

Step 1 特征提取

将原始数据转化成特征向量

Step 2 逻辑回归

训练已经进行过数据标注（手动打过标签）的数据集，得到特征向量每个特征的权重。

Step 3 特征权重构成预测模型

Deep Learning: 解决如何设计特征的难题，让算法从数据中自动学习特征。

DL是实现ML的一种技术。



一、什么是TensorFlow

1. TensorFlow由Google开发，是Google的第二代机器学习系统，2015年开源。

说文解字：以张量在计算图上流动的方式实现和执行机器学习算法的框架。

↓
Tensor

↓
Flow

(1) 额外说明：只要可以将计算表示成数据流图，就可以使用TensorFlow，换句话说它不止是机器学习库，而是一个基于数据流图的科学计算库。

(2) TensorFlow可以自动求微分，所以反向传播的梯度求解就可以自动化。

(3) 语言：

硬件好的条件下，支持Python实验；

硬件欠缺条件下，支持C++实验。

2. 其他机器学习框架：Caffe、Keras(凯拉斯)、mxnet、Torch(火炬)、Theano(茶野)、Chainer(链条机)



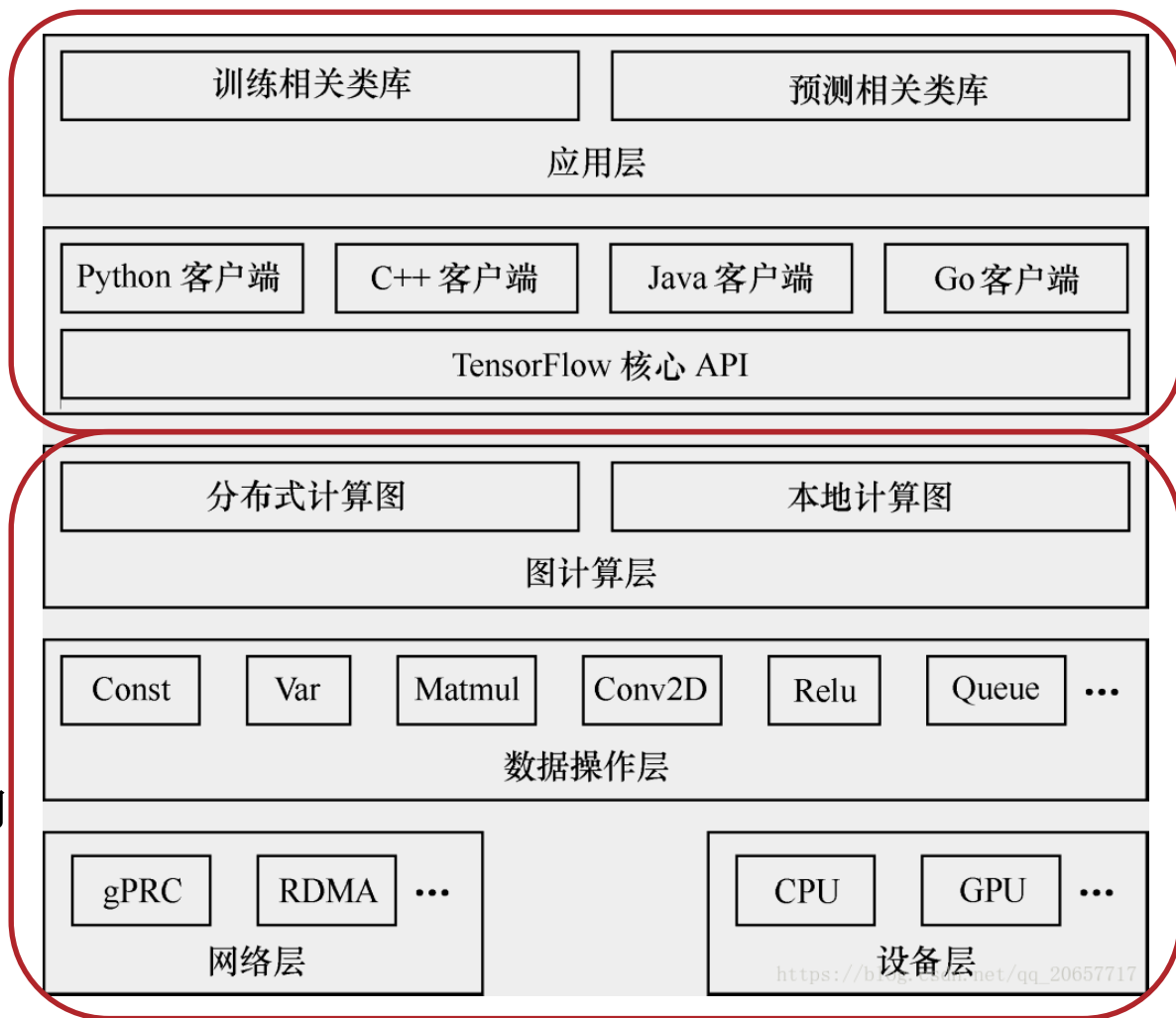
一、什么是TensorFlow

3. TensorFlow架构

前端：构造计算图
提供多种语言接口

前端→session桥梁→后端

后端：执行计算图
采用C++实现
调用CPU\GPU等内核完成具体计算。





一、什么是TensorFlow

4. 编程模式：符号式编程

(1) 命令式编程 (Imperative)：明确输入变量，根据程序逻辑逐步运算。

以前我编写的都是命令式编程。例如

```
import numpy as np
```

```
a = np.ones(10)
```

```
b = np.ones(10) * 2
```

```
c = b * a
```

```
d = c + 1
```



一、什么是TensorFlow

4. 编程模式：符号式编程

(2) 符号式编程 (Symbolic)：将计算过程抽象为计算图，所有input节点、compute节点、output节点均符号化处理。例如

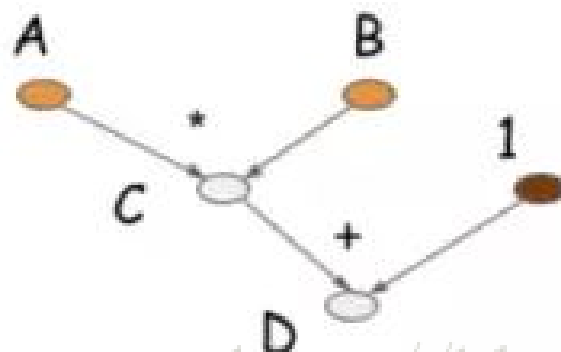
```
A = Variable('A')
```

```
B = Variable('B')
```

```
C = B * A
```

```
D = C + Constant(1)
```

构建出右边含有
5个OP的计算图



```
# compiles the function//编译上述定义的函数
```

```
f = compile(D)
```

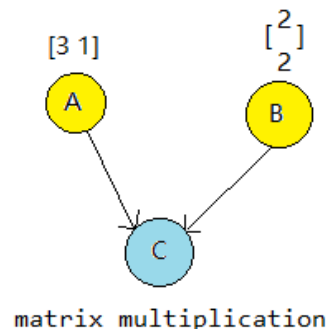
```
d = f(A=np.ones(10), B=np.ones(10)*2)
```




一、什么是TensorFlow

5. TensorFlow编程中的参数概念

(1) 图Graph: 表示计算流程



TensorFlow Python库有一个默认图 (default graph), OP构造器可以为其增加节点(OP)。

例如, 构建一个包含3个OP的计算图:

```
import tensorflow as tf

# Create a Constant op that produces a 1x2 matrix. The op is added as a node to the default
graph.

# The value returned by the constructor represents the output of the Constant op.
matrix1 = tf.constant([[3., 3.]])

# Create another Constant op that produces a 2x1 matrix.
matrix2 = tf.constant([[2.],[2.]])

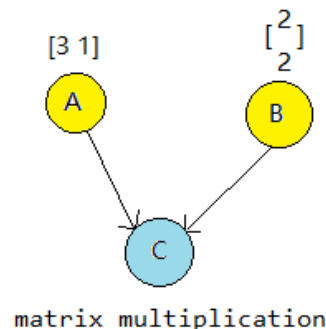
# Create a Matmul op that takes 'matrix1' and 'matrix2' as inputs.
# The returned value, 'product', represents the result of the matrix multiplication.
product = tf.matmul(matrix1, matrix2)
```



一、什么是TensorFlow

5. TensorFlow编程中的参数概念

(2) 会话Session: 在Session中执行Graph。



执行计算图的第一步是创建Session对象（如命名为sess）。

```
# Launch the default graph.
sess = tf.Session()

# To run the matmul op we call the session 'run()' method, passing 'product' which represents
# the output of the matmul op. This indicates to the call that we want to get the output of
# the matmul op back.
# All inputs needed by the op are run automatically by the session. They typically are run in
# parallel.
# The call 'run(product)' thus causes the execution of three ops in the graph: the two
# constants and matmul.
# The output of the op is returned in 'result' as a numpy `ndarray` object.
result = sess.run(product)
print(result)
# ==> [[ 12.]]
# Close the Session when we're done.
sess.close()
```

最后关闭Session释放资源，可以直接用with语句

```
with tf.Session() as sess:
    result = sess.run([product])
    print(result)
```



一、什么是TensorFlow

5. TensorFlow编程中的参数概念

(3) 张量Tensor: 表示数据。图片、语音等数据都是以Tensor的形式表示。

● Tensor : $[T_1, T_2, T_3, \dots, T_n]$, T 可以是单个数字、矩阵。

数据类型可包括: 0维数值 一维矢量 二维矩阵 N维数组

Tensor维度看Tensor的最左边有几个左中括号。

● Tensor的形状: $[D_0, D_1, \dots, D_{n-1}]$

形状的中括号中有几个数字, 就代表这个张量是几维张量。

形状的第一个元素要看张量最外边的中括号中有几个元素。

1 #维度为0的标量

$[1, 2, 3]$ #维度为1, 一维向量

$[[1, 2], [3, 4]]$ #维度为2, 二维矩阵

$[[[1, 2], [3, 4]], [[1, 2], [3, 4]]]$ #维度为3, 3维空间矩阵)

1 # 形状为 $[]$

$[1, 2, 3]$ # 形状为 $[3]$

$[[1, 2], [3, 4]]$ # 形状为 $[2, 2]$

$[[[1, 2], [3, 4]], [[1, 2], [3, 4]]]$ # 形状为 $[2, 2, 2]$)



一、什么是TensorFlow

5. TensorFlow编程中的参数概念

(4) 变量Variable: 用于存储和更新参数。

换句话说，机器学习就是用一系列Variables表示一个统计模型，训练过程中不断更新，训练完成后用这些Variable构成的模型进行预测。



```
# Create a Variable, that will be initialized to the scalar value 0.
state = tf.Variable(0, name="counter")

# Create an Op to add one to `state`.
one = tf.constant(1)
new_value = tf.add(state, one)
update = tf.assign(state, new_value)

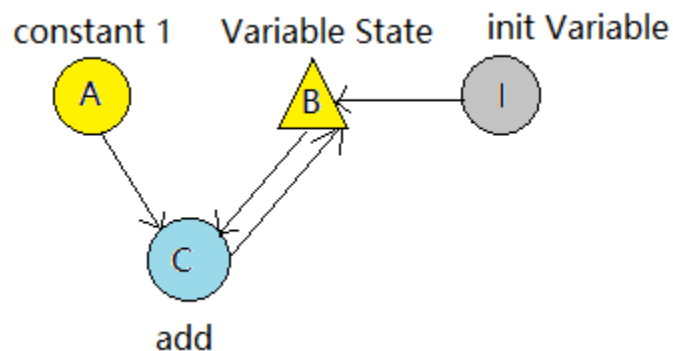
# Variables must be initialized by running an `init` Op after having launched the graph.
# We first have to add the `init` Op to the graph.
init_op = tf._all_variables()

# Launch the graph and initialize and run the ops.
with tf.Session() as sess:
    # Run the 'init' op
    sess.run(init_op)

    # Print the initial value of 'state'
    print(sess.run(state))

    # Run the op that updates 'state' and print 'state'.
    for _ in range(3):
        sess.run(update)
        print(sess.run(state))

# output: 0 1 2 3
```





一、什么是TensorFlow

5. TensorFlow编程中的参数概念

(5) feed和fetch：赋值或获取值。

```
# feed机制实现从外部导入数据；
input1 = tf.placeholder(tf.float32)
input2 = tf.placeholder(tf.float32)
output = tf.mul(input1, input2)
with tf.Session() as sess:
    print(sess.run([output],
        feed_dict={input1:[7.], input2:[2.]})
# output: [array([ 14.], dtype=float32)]
```

```
# fetch机制实现获取Operation的输出
input1 = tf.constant([3.0])
input2 = tf.constant([2.0])
input3 = tf.constant([5.0])
intermed = tf.add(input2, input3)
mul = tf.mul(input1, intermed)
with tf.Session() as sess:
    result = sess.run([mul, intermed])
    print(result)
# output:
[array([ 21.], dtype=float32), array([ 7.],
dtype=float32)]
```

特点：都要执行session的run函数。



一、什么是TensorFlow

5. TensorFlow编程中的参数概念

(6) 操作Operation(OP): 图中的节点。

OP input: 0~multi Variable

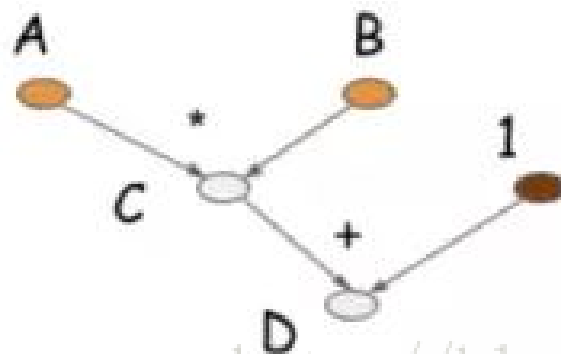
OP output: 0~multi Variable

源OP: 不需要任何input的OP, 如Constant

构建Graph的第一步就是创造源OP,

源OP的输出 (即OP构造器的返回值)

被传递给其它OP做运算。





二、TensorFlow环境搭

Step 1 安装Anaconda3 64位

install后，打开cmd

input: conda list, 查看Anaconda已经安装了哪些包。

```
C:\Users\hmd>conda list
# packages in environment at D:\Tools\Python\Anaconda3:
#
# Name                        Version            Build Channel
ipyw_jlab_nb_ext_conf        0.1.0              py37_0 https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
alabaster                     0.7.12             py37_0 https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
anaconda                      2019.10            py37_0 https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
```

input: conda --version, 查看Anaconda的版本 (4.7.12)

```
C:\Users\hmd>conda --version
conda 4.7.12
```

input: python, 查看Anaconda默认的python版本 (3.7.4)

```
C:\Users\hmd>python
Python 3.7.4 (default, Aug 9 2019, 18:34:13) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32

Warning:
This Python interpreter is in a conda environment, but the environment has
not been activated. Libraries may fail to load. To activate this environment
please see https://conda.io/activation
```




二、TensorFlow环境搭

Step 1 安装Anaconda3 64位

input:

```
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/  
conda config --add channels https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main/  
conda config --set show_channel_urls yes  
conda update --all  
更新所有包。
```



二、TensorFlow环境搭

Step 2 安装Tensorflow 64位()

创建虚拟环境：Tensorflow，为虚拟环境指定Python版本3.6

conda create -n environment_name python=version

```
(base) C:\windows\system32>conda create -n tensorflow python=3.6
Collecting package metadata (current_repodata.json): done
Solving environment: done

Proceed ([y]/n)? y

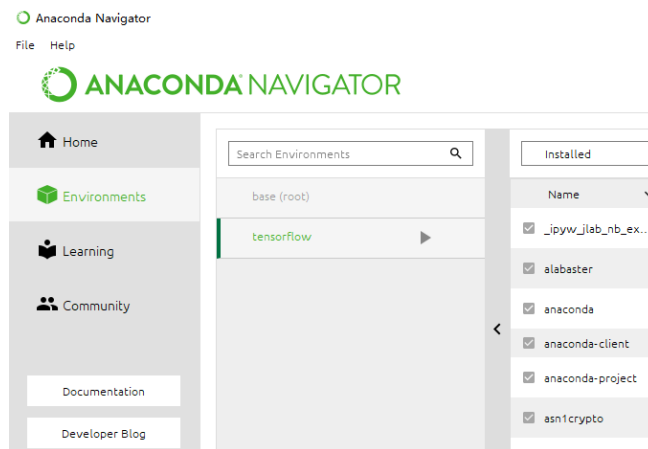
Downloading and Extracting Packages
python-3.6.2            31.5 MB  #####
#
```

激活虚拟环境：

conda activate environment_name

关闭虚拟环境

conda deactivate





二、TensorFlow环境搭

Step 2 安装Tensorflow 64位()

input: python -m pip install --upgrade pip, 更新pip

input: pip install tensorflow -i

<https://pypi.tuna.tsinghua.edu.cn/simple> 安装CPU版本

还可以: pip install tensorflow-gpu -i

<https://pypi.tuna.tsinghua.edu.cn/simple> 安装GPU版本

```
Successfully built absl-py wrapt opt-einsum gast termcolor  
ERROR: google-auth 1.7.1 has requirement setuptools>=40.3.0, but you'll have setuptools 36.4.0 which is incompatible.  
ERROR: tensorboard 2.0.1 has requirement setuptools>=41.0.0, but you'll have setuptools 36.4.0 which is incompatible.
```

装好后检查tensorflow的版本 (V2) 和环境下的python版本(3.6.2):

```
VERSION  
2.0.0  
  
FILE  
d:\tools\python\anaconda3\envs\tensorflow\lib\site-packages\tensorflow\__init__.py  
  
(tensorflow) C:\Users\hmd>python --version  
Python 3.6.2 :: Continuum Analytics, Inc.
```



二、TensorFlow环境搭

Step 2 安装Tensorflow 64位()

测试cpu版本的TensorFlow :

```
input: python
```

```
import tensorflow as tf
```

```
hello = tf.constant('Hello, TensorFlow!') #初始化一个TensorFlow常量
```

```
sess = tf.Session() #启动一个会话
```

```
print(sess.run(hello))
```



二、TensorFlow环境搭

Step 2 安装Tensorflow 64位()

报错:

(1) AVX2: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2

(2) 没有session: AttributeError: module 'tensorflow' has no attribute 'Session'

解决办法: 卸载tensorflowV2.0.0, 重装tensorflowV1.2.1

input:: pip uninstall tensorflow

pip install tensorflow==1.2.1 -i https://pypi.tuna.tsinghua.edu.cn/simple

help (tf)

```
VERSION
1.2.1

FILE
d:\tools\python\anaconda3\envs\tensorflow\lib\site-packages\tensorflow\__init__.py
```



二、TensorFlow环境搭

Step 2 安装Tensorflow 64位()

input: python

```
import tensorflow as tf
```

```
hello = tf.constant('Hello, TensorFlow!') #初始化一个TensorFlow常量
```

```
sess = tf.Session() #启动一个会话
```

```
print(sess.run(hello))
```

成功输出测试代码！再把warning debug掉。

```
b'Hello, TensorFlow!'
```

Debug warning: 重装numpy版本。



二、TensorFlow环境搭

Step 2 安装Tensorflow 64位()

input: pip show numpy, 查看numpy版本 (1.17.4), 卸载, 更新至 (1.14.0)

```
(base) C:\Users\hmd>activate tensorflow
(tensorflow) C:\Users\hmd>pip show numpy
Name: numpy
Version: 1.17.4
Summary: NumPy is the fundamental package for array computing with Python.
Home-page: https://www.numpy.org
Author: Travis E. Oliphant et al.
Author-email: None
License: BSD
Location: d:\tools\python\anaconda3\envs\tensorflow\lib\site-packages
Requires:
Required-by: tensorflow, tensorboard, opt-einsum, Keras-Preprocessing, Keras-Applications, h5py
```

input: pip show setuptools, 查看setuptools版本 (36.4.0), 卸载, 更新至 (41.2.0)



二、TensorFlow环境搭

Step 2 安装Tensorflow 64位()

测试cpu版本的TensorFlow :

```
input: python
```

```
import os
```

```
os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
```

```
import tensorflow as tf
```

```
hello = tf.constant('Hello, TensorFlow!') #初始化一个TensorFlow常量
```

```
sess = tf.Session() #启动一个会话
```

```
print(sess.run(hello))
```

```
(tensorflow) C:\Users\hmd>python
Python 3.6.2 |Continuum Analytics, Inc.| (default, Jul 20 2017, 12:30:02) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
>>> import tensorflow as tf
>>> hello = tf.constant('Hello, TensorFlow!') #初始化一个TensorFlow的常量
>>> sess = tf.Session() #启动一个会话
>>> print(sess.run(hello))
b'Hello, TensorFlow!'
```




二、TensorFlow环境搭

Step 2 总结

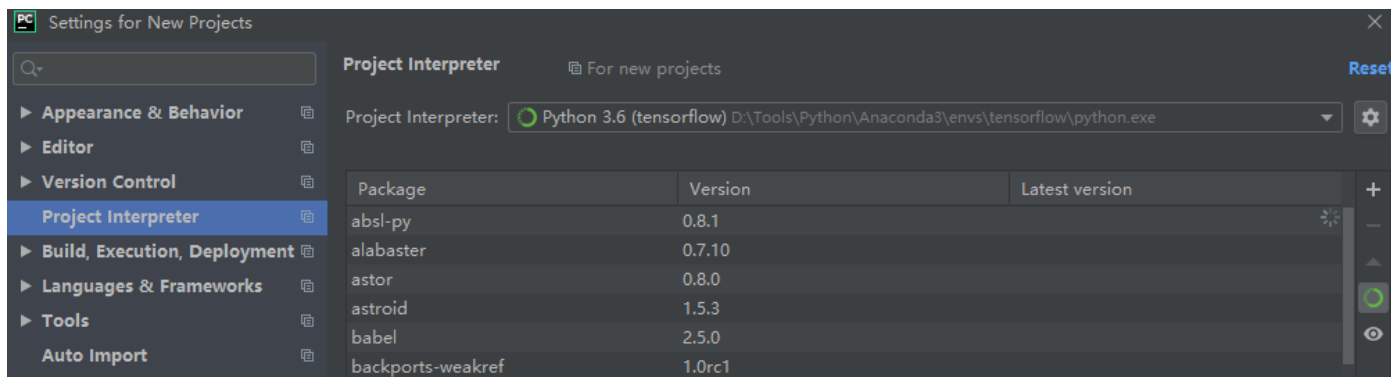
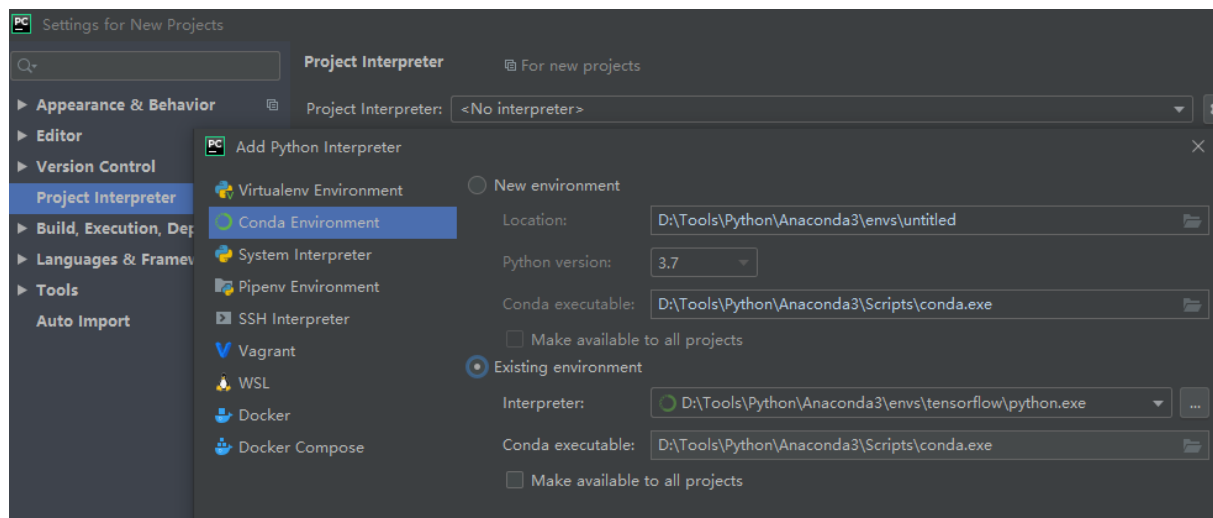
TensorFlow安装过程中，版本匹配很重要，列出以下成功的版本组合：

- (1) Anaconda Version: 4.7.12
- (2) TensorFlow Version: 1.2.1
- (3) TensorFlow中的Python Version: 3.7.4
- (4) Numpy Version: 1.14.0
- (5) Setuptools Version: 36.4.0



二、TensorFlow环境搭

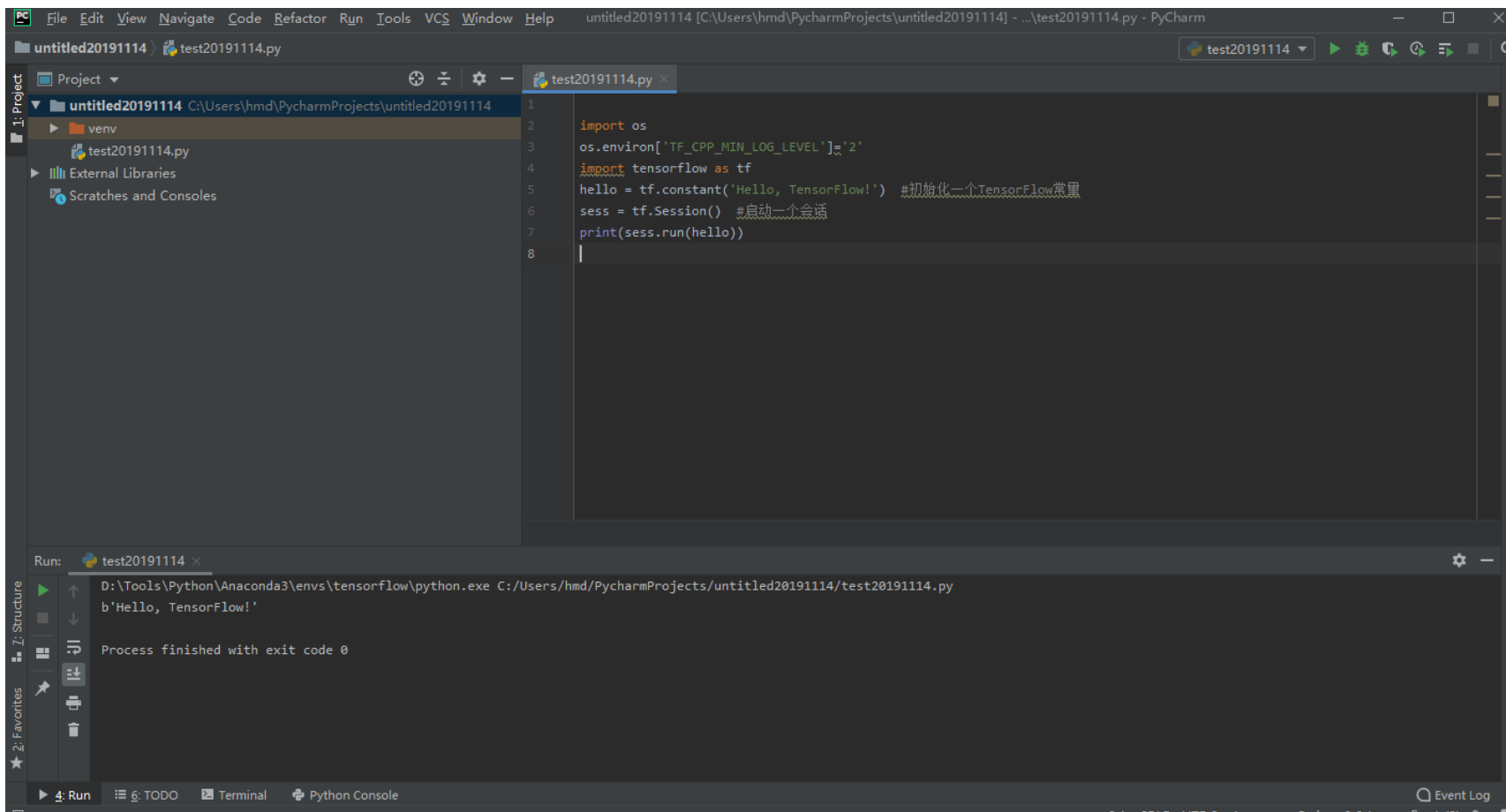
Step 3 安装Pycharm，在pycharm中引入tensorflow





二、TensorFlow环境搭

Step 3 安装Pycharm，在pycharm中引入tensorflow



```
1 import os
2 os.environ['TF_CPP_MIN_LOG_LEVEL']='2'
3 import tensorflow as tf
4 hello = tf.constant('Hello, TensorFlow!') #初始化一个TensorFlow常量
5 sess = tf.Session() #启动一个会话
6 print(sess.run(hello))
7
8
```

Run: test20191114

D:\Tools\Python\Anaconda3\envs\tensorflow\python.exe C:/Users/hmd/PycharmProjects/untitled20191114/test20191114.py
b'Hello, TensorFlow!'

Process finished with exit code 0



西安电子科技大学
XIDIAN UNIVERSITY

Thanks

