

#### 4. Install TensorFlow GPU

Before we install TensorFlow GPU, we need to complete configuration of CUDA.

About configuration of CUDA, please refer to 【1.Preparation tutorial】

##### 1. Install pip

We need to input command:

**sudo apt-get install python3-pip python3-dev**

After installation, pip is version 9.01, you need to upgrade it to the latest version.

After upgrading, the pip version is 19.3.1. There will be a small bug after the upgrade, you need to manually change it.

We need to input command:

**python3 -m pip install --upgrade pip** #upgrade pip

```
jetson@jetson-desktop:~$ sudo apt-get install python3-pip python3-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  apt-cclone archdetect-deb bogl-bterm busybox-static cryptsetup-bin dpkg-repack gir1.2-timezon
  kwayland-data kwin-common kwin-data kwin-x11 libdebian-installer4 libkdecorations2-5v5 libkd
  libkf5completion5 libkf5declarative-data libkf5declarative5 libkf5doctools5 libkf5globalacce
  libkf5jobwidgets-data libkf5jobwidgets5 libkf5kcmutils-data libkf5kcmutils5 libkf5kiocore5 l
  libkf5package-data libkf5package5 libkf5plasma5 libkf5quickaddons5 libkf5solid5 libkf5solid5
  libkf5textwidgets5 libkf5waylandclient5 libkf5waylandserver5 libkf5xmlgui-bin libkf5xmlgui-d
  libkwineglutils11 libkwineglutils11 libllvm9 libqgsttools-plt libqt5designer5 libqt5help5
  libqt5multimediawidgets5 libqt5opengl5 libqt5quickwidgets5 libqt5sql5 libqt5test5 libxcb-com
  python3-pam python3-pyqt5 python3-pyqt5.qtsvg python3-pyqt5.qtwebkit python3-sip qml-module-
  tasksel-data
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  dh-python libpython3-dev libpython3.6-dev python-pip-whl python3-setuptools python3-wheel py
Suggested packages:
  python-setuptools-doc
The following NEW packages will be installed:
  dh-python libpython3-dev libpython3.6-dev python-pip-whl python3-dev python3-pip python3-set
```

**sudo vim /usr/bin/pip3** #Open pip3 file

Replace

```
from pip import main
```

```
if __name__ == '__main__':
```

```
    sys.exit(main())
```

to

```
from pip import __main__
```

```
if __name__ == '__main__':
```

```
    sys.exit(__main__.__main__())
```

After modification is complete, we need to input command:

**pip3 -V**

```
jetson@jetson-desktop:~$ pip3 -V
pip 20.2.4 from /usr/local/lib/python3.6/dist-packages/pip (python 3.6)
jetson@jetson-desktop:~$
```

## 2. We need to input command to install some software package:

### **sudo apt-get install python3-numpy**

(It is an extension library of Python language, which supports a large number of dimensional arrays and matrix operations, and also provides a large number of mathematical function libraries for array operations.)

### **sudo apt-get install python3-scipy**

(Scipy is a common software package used in the fields of mathematics, science, and engineering, which can handle interpolation, integration, optimization, image processing, numerical solution of ordinary differential equations, signal processing, etc.)

### **sudo apt-get install python3-pandas**

(Pandas is a tool based on NumPy, which is created to solve data analysis tasks. Pandas includes a large number of libraries and some standard data models, and provides the tools needed to efficiently operate large data sets. Pandas provides A large number of functions and methods that enable us to process data quickly and easily. )

### **sudo apt-get install python3-matplotlib**

(Matplotlib is a 2D plotting library for Python that generates publishing-quality graphics in a variety of hardcopy formats and a cross-platform interactive environment)

### **sudo apt-get install python3-sklearn**

(Simple and efficient data mining and data analysis tools)

## 3. Install TensorFlow GPU version

Check if CUDA is installed properly

We need to input command:

### **nvcc -V**

If you can see the CUDA version number, as shown below, it is installed correctly.

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2019 NVIDIA Corporation
Built on Wed Oct 23 21:14:42 PDT 2019
Cuda compilation tools, release 10.2, V10.2.89
```

1) Install the required package

We need to input command:

### **sudo apt-get install libhdf5-serial-dev hdf5-tools libhdf5-dev zlib1g-dev zip libjpeg8-dev liblapack-dev libblas-dev gfortran**

2) Install python dependency

### **sudo pip3 install -U numpy==1.16.1 future==0.18.2 mock==3.0.5 h5py==2.10.0**

**keras\_preprocessing==1.1.1 keras\_applications==1.0.8 gast==0.2.2 futures protobuf pybind11**

3) Install TensorFlow GPU version

We need to input the following command:

**sudo pip3 install --pre --extra-index-url**

**<https://developer.download.nvidia.com/compute/redist/jp/v44/tensorflow>**

4) Enter the following command to check whether tensorflow is successfully installed.

```
Python 3.6.9 (default, Oct 8 2020, 12:12:24)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
2020-11-04 09:49:32.772668: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.10.2
>>> tf.__version__
'2.3.0'
```

#### 4. Install Keras

Keras is a high-level neural network API written in Python. It can run with TensorFlow, CNTK, or Theano as the backend.

We can input the following command to install Keras:

**sudo pip3 install keras**

After the installation is complete, we can input **python3** and check the installation results. When we input **import keras**, the following prompts “using TensorFlow backend”, it proves that Keras is installed successfully and uses TensorFlow as backend.

```
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import keras
2020-10-27 14:01:25.163525: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.10.2
```

```
beckhans@Jetson:~$ python3
```

```
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
```

```
[GCC 8.2.0] on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>> import keras
```

```
Using TensorFlow backend.
```

```
>>>
```

If there is a problem like the one shown below, you need to update the version of numpy.

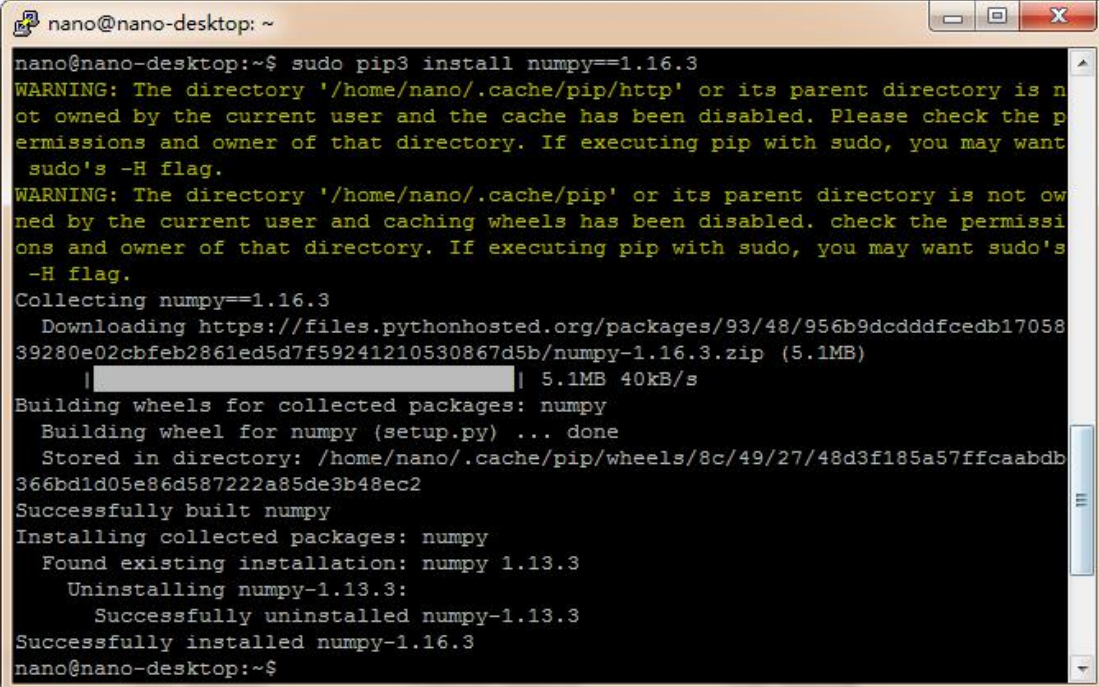
```
nano@nano-desktop:~$ python3
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
[GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import keras
Using TensorFlow backend.
ModuleNotFoundError: No module named 'numpy.core._multiarray_umath'
ImportError: numpy.core._multiarray_umath failed to import

The above exception was the direct cause of the following exception:

Traceback (most recent call last):
  File "<frozen importlib._bootstrap>", line 968, in _find_and_load
SystemError: <class '_frozen_importlib._ModuleLockManager'> returned a result with an error set
ImportError: numpy.core._multiarray_umath failed to import
ImportError: numpy.core._multiarray_umath failed to import
2019-05-10 17:27:32.651672: F tensorflow/python/lib/core/bfloat16.cc:675] Check failed: PyBfloat16_Type.tp_base != nullptr
Aborted (core dumped)
```

We can input the following command:

**sudo pip3 install numpy==1.16.3**



```
nano@nano-desktop: ~
nano@nano-desktop:~$ sudo pip3 install numpy==1.16.3
WARNING: The directory '/home/nano/.cache/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
WARNING: The directory '/home/nano/.cache/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting numpy==1.16.3
  Downloading https://files.pythonhosted.org/packages/93/48/956b9dcdddfcedb1705839280e02cbfeb2861ed5d7f59241210530867d5b/numpy-1.16.3.zip (5.1MB)
    | 5.1MB 40kB/s
Building wheels for collected packages: numpy
  Building wheel for numpy (setup.py) ... done
  Stored in directory: /home/nano/.cache/pip/wheels/8c/49/27/48d3f185a57ffcaabdb366bd1d05e86d587222a85de3b48ec2
Successfully built numpy
Installing collected packages: numpy
  Found existing installation: numpy 1.13.3
    Uninstalling numpy-1.13.3:
      Successfully uninstalled numpy-1.13.3
Successfully installed numpy-1.16.3
nano@nano-desktop:~$
```

## 5. Test TensorFlow

We can use vi to create a new python file name: tensorflowDemo.py and then copy the following code into it. After saving, run it with **python3 tensorflowDemo.py**. This section must be run in a graphical interface, because a chart will appear.

```
import tensorflow as tf
import numpy as np
```



```

import matplotlib.pyplot as plt

x_data = np.linspace(-0.5, 0.5, 200)[: , np.newaxis]
noise = np.random.normal(0, 0.02, x_data.shape)
y_data = np.square(x_data) + noise

x = tf.placeholder(tf.float32, [None, 1])
y = tf.placeholder(tf.float32, [None, 1])

# Input layer one neuron, output layer one neuron, middle 10 neurons
# First layer
Weights_L1 = tf.Variable(tf.random.normal([1, 10]))
Biases_L1 = tf.Variable(tf.zeros([1, 10]))
Wx_plus_b_L1 = tf.matmul(x, Weights_L1) + Biases_L1
L1 = tf.nn.tanh(Wx_plus_b_L1)

# Second layer
Weights_L2 = tf.Variable(tf.random.normal([10, 1]))
Biases_L2 = tf.Variable(tf.zeros([1, 1]))
Wx_plus_b_L2 = tf.matmul(L1, Weights_L2) + Biases_L2
pred = tf.nn.tanh(Wx_plus_b_L2)

# Loss function
loss = tf.reduce_mean(tf.square(y - pred))

# Train
train = tf.train.GradientDescentOptimizer(0.1).minimize(loss)

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(2000):
        sess.run(train, feed_dict={x: x_data, y: y_data})
        print("第{0}次, loss = {1}".format(i, sess.run(loss, feed_dict={x: x_data, y:
y_data})))
    pred_vaule = sess.run(pred, feed_dict={x: x_data})
    plt.figure()
    plt.scatter(x_data, y_data)
    plt.plot(x_data, pred_vaule, 'r-', lw=5)
    plt.show()

```

```

第1993次, loss = 0.000520005589351058
第1994次, loss = 0.0005199451697990298
第1995次, loss = 0.0005198844010010362
第1996次, loss = 0.0005198236322030425
第1997次, loss = 0.0005197632126510143
第1998次, loss = 0.0005197029677219689
第1999次, loss = 0.0005196425481699407

```

