4.4 TensorFlow AI framework

4.4.1 About TensorFlow

TensorFlow is an open source software library that uses data flow graphs for numerical calculations. The nodes in the graph represent mathematical operations, while the edges of the graph represent the multidimensional data array (tensor) flowing between them. This flexible architecture allows you to deploy computing to one or more CPUs or GPUs on your desktop, server or mobile device without having to rewrite your code.

The TensorFlow User Guide provides a detailed overview and describes how to use and customize the TensorFlow deep learning framework. TensorFlow was originally developed by researchers and engineers at the Google Brain team of Google Machine Intelligence Research to conduct machine learning and deep neural network (DNN) research, which is versatile enough to be used in a variety of other areas.

4.4.2 Install and use TensorFlow

Below we demonstrate the use of TensorFlow with a simple example of nonlinear regression.

(This needs to be demonstrated on the main system side of the Jetbot robot car with the screen. After the program is executed, a window will pop up to display the running result)

```
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
# First level
```

```
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 level
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))

# Training function
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("{0}, 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()
```

NVIDIA Deep Learning Official TensorFlow User Guide:

https://docs.nvidia.com/deeplearning/frameworks/tensorflow-user-guide/index.htm

TensorFlow official learning website:

https://www.tensorflow.org/learn?hl=zh_cn