

TensorFlow

www.huawei.com

Copyright © 2018 Huawei Technologies Co., Ltd. All rights reserved.





Objectives

- ◆ After completing this course, you will be able to:
 - Understand what TensorFlow is and its characteristics.
 - Master the basic knowledge of TensorFlow and the methods for environment setup.
 - Understand TensorFlow modules.
 - Master basic development steps using TensorFlow.
 - Understand other deep learning frameworks.

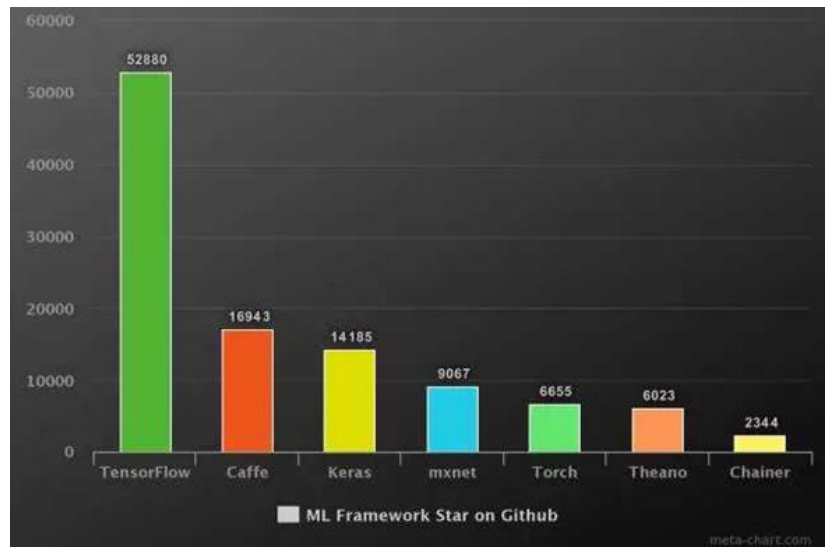


Contents

- 1. TensorFlow Overview**
2. TensorFlow Characteristics
3. TensorFlow Basics
4. TensorFlow Modules
5. TensorFlow Development Environment Setup
6. Basic Development Steps Using TensorFlow
7. Other Deep Learning Frameworks

What Is TensorFlow

- ◆ TensorFlow is Google's second-generation software library for dataflow programming.
- ◆ The TensorFlow framework supports various deep learning algorithms, as well as many computing platforms other than those for deep learning. The system stability is high.
- ◆ TensorFlow is open-source, which facilitates maintenance and update and improves the development efficiency.

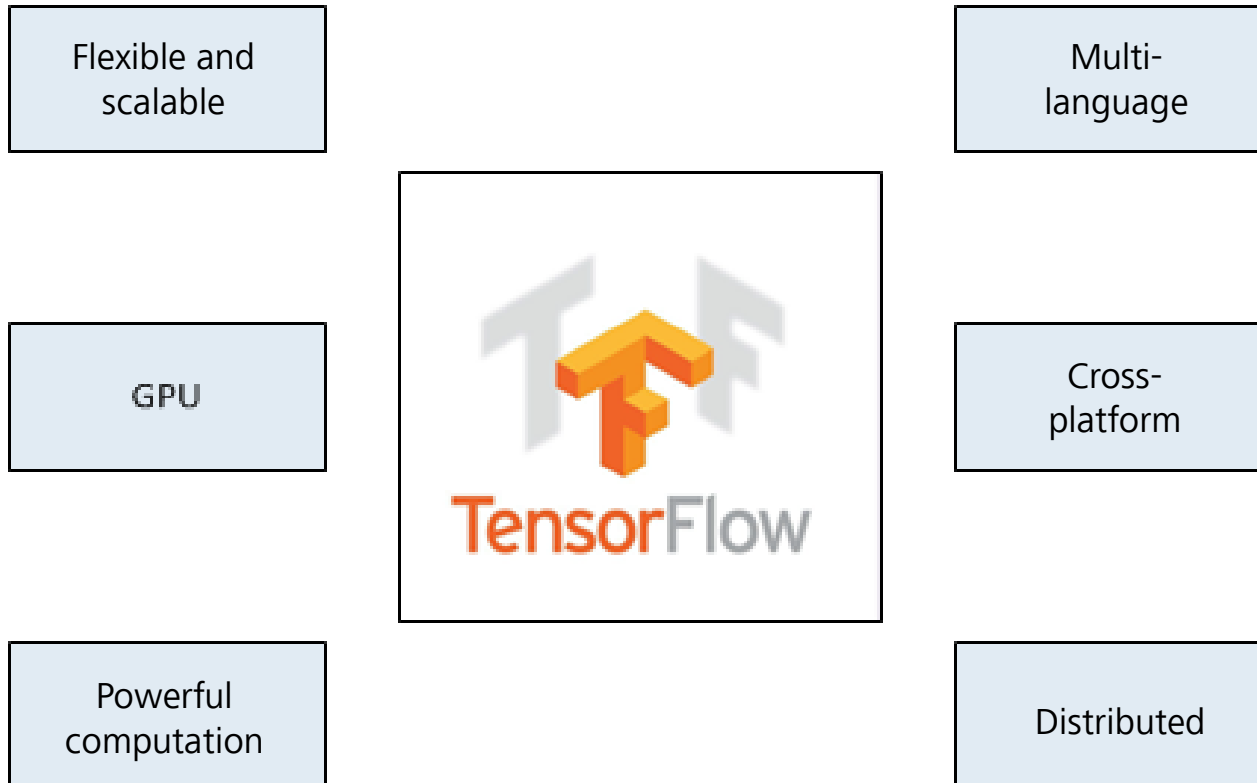




Contents

1. TensorFlow Overview
- 2. TensorFlow Characteristics**
3. TensorFlow Basics
4. TensorFlow Modules
5. TensorFlow Development Environment Setup
6. Basic Development Steps Using TensorFlow
7. Other Deep Learning Frameworks

TensorFlow Characteristics



What Can We Do Using TensorFlow (1)

- ◆ Self-driving cars
- ◆ Music creation
- ◆ Image recognition
- ◆ Speech recognition
- ◆ Language models
- ◆ Human activity recognition
- ◆ Automated theorem proving
- ◆ Gaming (for example, play MarioKart racing games), etc.

What Can We Do Using TensorFlow (2)



Artistic style transfer



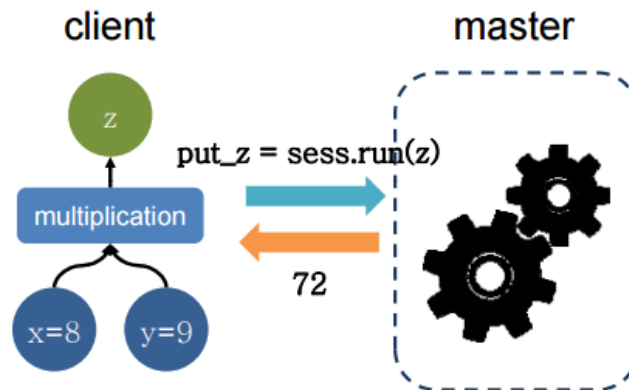
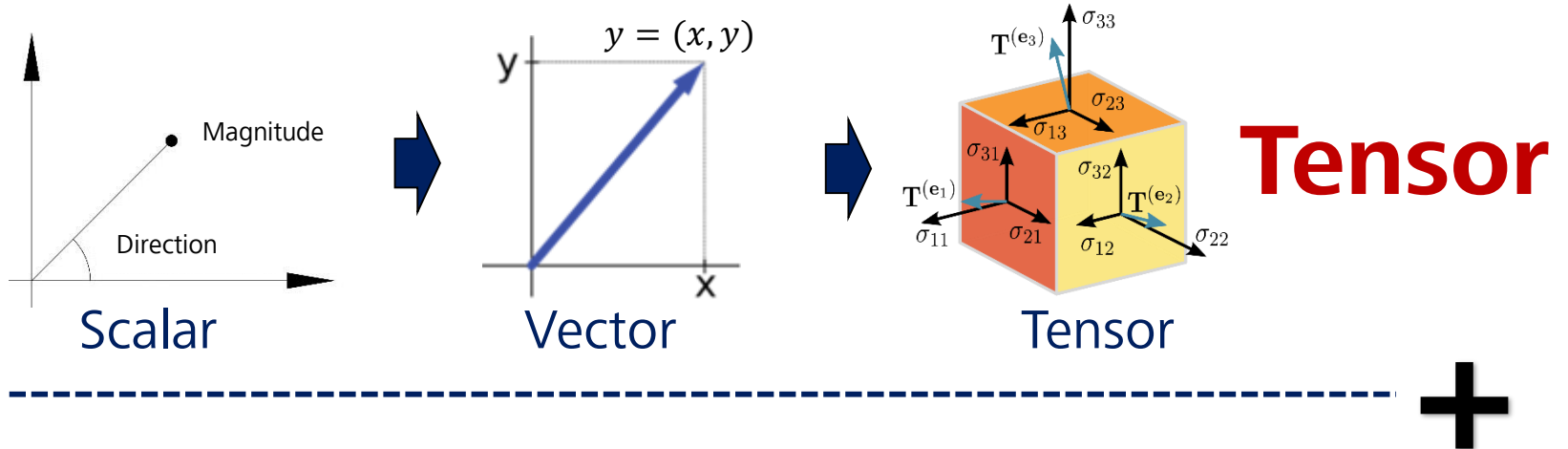
Facial recognition



Contents

1. TensorFlow Overview
2. TensorFlow Characteristics
- 3. TensorFlow Basics**
4. TensorFlow Modules
5. TensorFlow Development Environment Setup
6. Basic Development Steps Using TensorFlow
7. Other Deep Learning Frameworks

TensorFlow

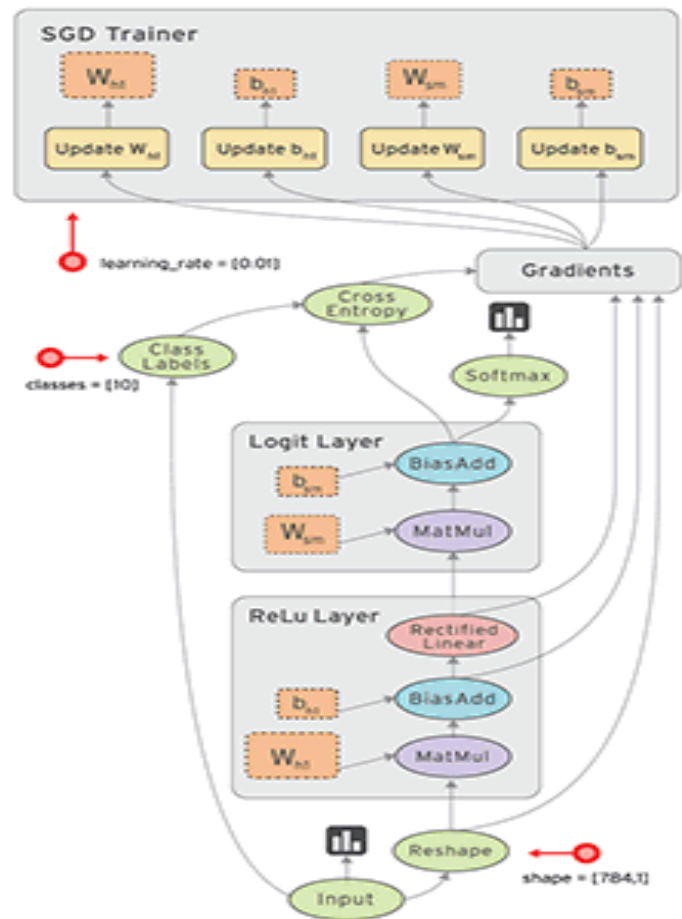


Data flow graph

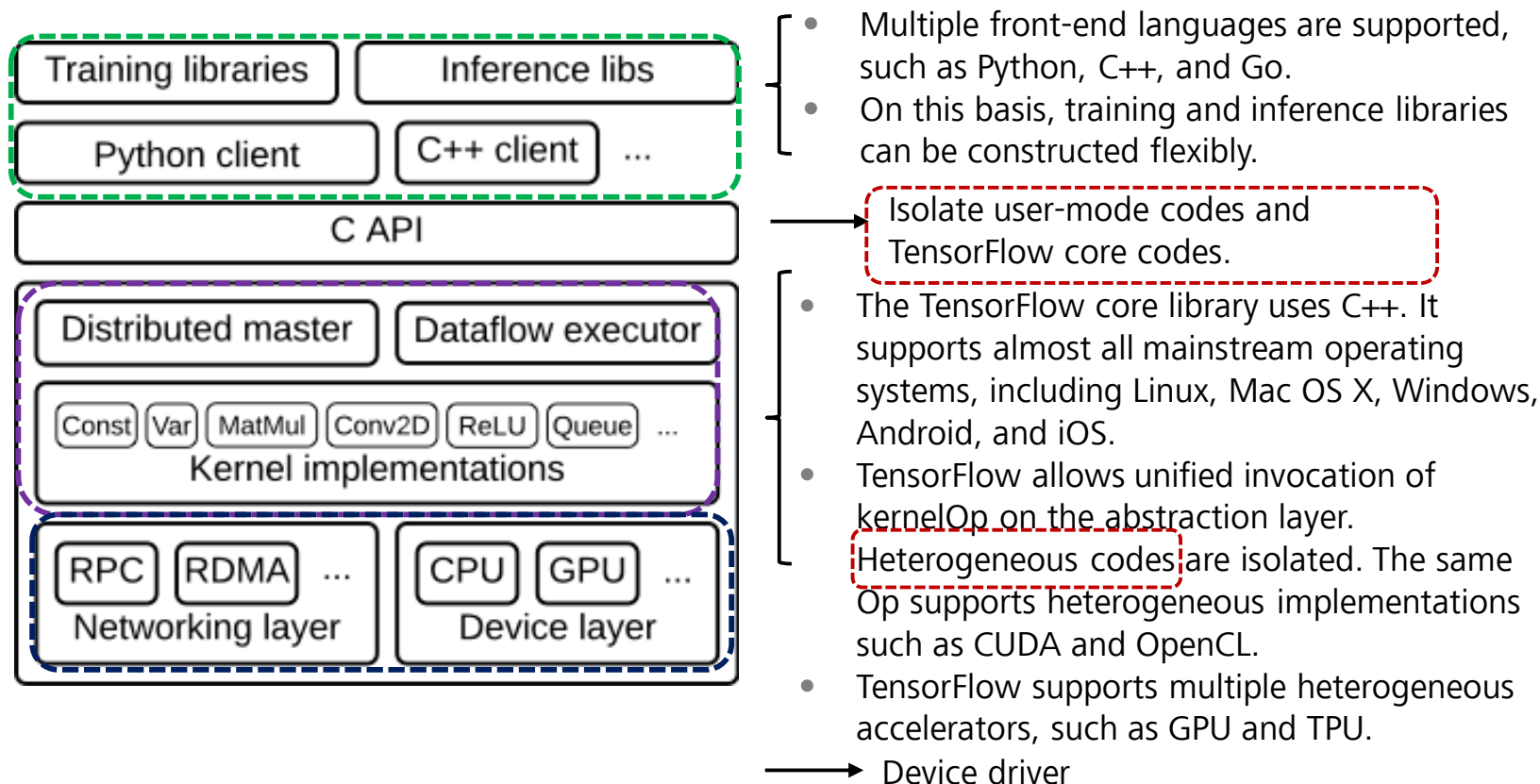
Flow

TensorFlow Computation Process

- ◆ TensorFlow is a programming system that describes computation in the form of a graph. It is a framework that implements and executes machine learning/deep learning algorithms in the form of tensors on graphs.



TensorFlow Architecture



Layer-by-layer breakdown and decoupling

Basic Concepts of TensorFlow

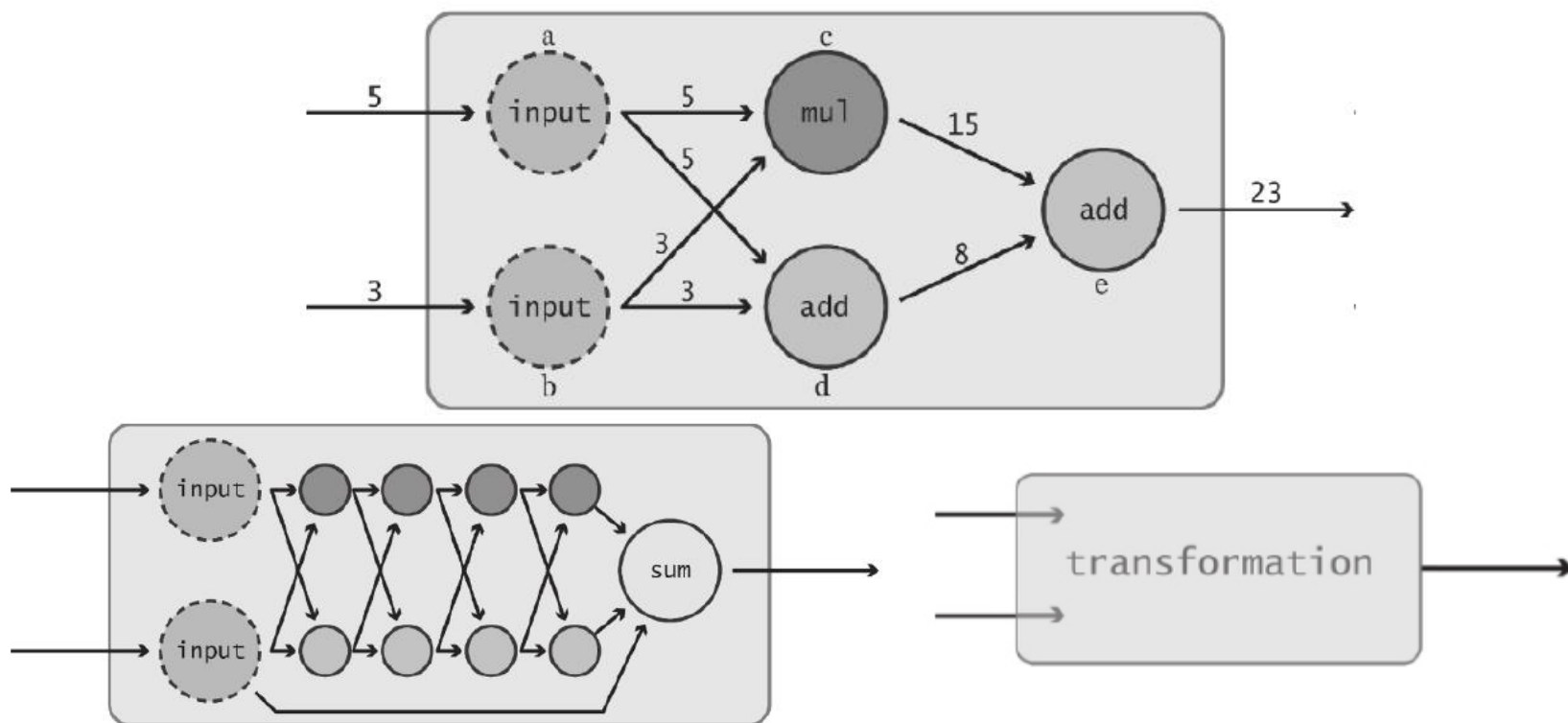
- ◆ Tensor
- ◆ Graph
 - Node
 - Edge
- ◆ Operator
- ◆ Session
 - Feed
 - Fetch
- ◆ Variable

Tensor

- ◆ Tensor is the primary data structure in TensorFlow programs. Tensors are N-dimensional (where N could be 1, 2, 3, 4, or very large) data structures. In a running graph, tensors are the data that flows between nodes.
- ◆ The data structure contained in a tensor is: name + shape + type.

Graph (Data Flow Graph)

- ◆ Node: A data operation (OP) in the TensorFlow graph
- ◆ Edge: Data and its dependency relationships

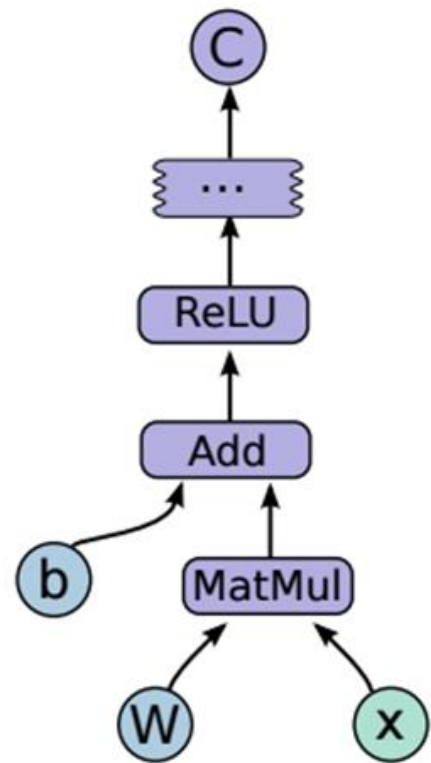


TensorFlow Operators

Category	Example
Element – wise mathematical Operations	Add, Sub, Div, Exp, Log, Greater, Less, Equal...
Array operations	Concat, Slice, Split, Constant, Rank, Shape, Shuffle...
Matrix operations	MatMul, MatrixInverse, MatrixDeterminant...
Stateful operations	Variable, Assign, AssignAdd...
Neural-net building blocks	SoftMax, Singmoid, ReLU, Convolution2D, MAMaxPool...
Checkpointing operations	Save, Restore...
Queue and synchronization operations	Enqueue, Dequeue, Mutex Acquire, Mutex Release...
Control flow operations	Merge, Switch, Enter, Leave, NextIteration

Session

- ◆ A session owns and manages all resources when the TensorFlow program is running.
- ◆ A client uses a session to interact with the TensorFlow system. Generally, an empty graph is generated when a session is created. Nodes and edges are added to the session to form a graph for execution.



Feed

- ◆ The feed mechanism directly connects the tensor to a node in the graph, and temporarily replaces the tensor value on the node. A feed is not created when the graph is formed. Rather, it is applied for when graph execution is triggered. That is, the feed data is initialized in the form of parameters when the "run" or "eval" command is executed. After the execution is complete, the replaced feed data disappears, but the behavior of the node defined in the graph does not change. Generally, the feed mechanism is used together with `tf.placeholder()`.

Fetch

- ◆ The fetch mechanism retrieves the result of an operation in the graph. The fetch application occurs when graph execution is triggered, not when the graph is generated. To fetch the tensor value of one or more nodes, you can call the `run()` method on the session object and use the list of the nodes to be fetched as parameters to execute the graph.

Variable

- ◆ A variable maintains state across multiple calls to run during graph execution. For example, in a neural network, it is used to identify coefficients such as w and b . The variable in TensorFlow is actually a variable object in Python.
- ◆ The initial value of a variable in TensorFlow can be set to a random number, a constant, or a number calculated based on the initial values of other variables.



Contents

1. TensorFlow Overview
2. TensorFlow Characteristics
3. TensorFlow Basics
- 4. TensorFlow Modules**
5. TensorFlow Development Environment Setup
6. Basic Development Steps Using TensorFlow
7. Other Deep Learning Frameworks

TensorFlow Modules (1)

- ◆ `tf.app`: A generic entry point script
- ◆ `tf.nn`: Supports neural networks.
- ◆ `tf.bitwise`: Used to manipulate the binary representations of integers.
- ◆ `tf.compat`: A function compatible with Python 2 and Python 3
- ◆ `tf.contrib`: Relatively complex; related to Monte Carlo integration, conditional random field, the builder, regularization, initialization, optimization, and feature columns.
- ◆ `tf.estimator`: An advanced tool for processing models
- ◆ `tf.errors`: Types of TensorFlow errors

TensorFlow Modules (2)

- ◆ `tf.nn`: Supports neural networks.
 - `tf.nn` is used to create RNN. It is the most commonly used module for constructing classical convolutional networks. It includes a sub-module, `rnn_cell`, which is used to construct recurrent neural networks.
- ◆ `tf.estimator`:
 - Used to create one or more input functions.
 - Defines the feature column of a model.
 - Instantiates an estimator to define feature columns and various types of hyperparameters.
 - Calls one or more methods on the estimator object and passes the appropriate input function as the data source.

TensorFlow Modules (3)

- ◆ `tf.layers`: The network layer encapsulates variables and operations on the variables.
 - For example, the full connection layer performs a weighted sum operation on the input and can use an optional activation function. The weight and bias of a connection are managed by network layer objects.
- ◆ `tf.contrib`: This module provides functions for computing streaming metrics.
 - The slim submodule is the most commonly used in this module. All the functions that are experimental or easy to change are in this module, which has rich functional modules.
 - The `tf.contrib` provides functions for computing streaming metrics: metrics computed on dynamically valued 'Tensors'. Each metric declaration returns a "value_tensor", an idempotent operation that returns the current value of the metric, and an "update_op", an operation that accumulates the information from the current value of the 'Tensors' being measured as well as returns the value of the "value_tensor".



Contents

1. TensorFlow Overview
2. TensorFlow Characteristics
3. TensorFlow Basics
4. TensorFlow Modules
- 5. TensorFlow Development Environment Setup**
6. Basic Development Steps Using TensorFlow
7. Other Deep Learning Frameworks

Setting Up the TensorFlow Environment on Windows (1)

- ◆ Operating systems: Windows/Mac/Linux
- ◆ Python 3
<https://www.python.org/downloads/release/python-350/>
- ◆ The Anaconda 3 (compatible with Python 3) contains the pip software.
- ◆ Installing TensorFlow
 - Installing the nightly package online (pip install tf-nightly)
 - Install the pure edition of TensorFlow (pip install tensorflow).
 - Install offline.

Setting Up the TensorFlow Environment on Windows (2)

- ◆ Installing the GPU version:
 - Install the CUDA software package (mapping with the TensorFlow version).
 - Install the cuDNN library (mapping with the TensorFlow version).
 - Test the graphics card.
 - Run the nvidia-smi command to view the graphics card information.
 - Check the CUDA version.
 - For installation on Linux and Mac, see online tutorials.

Setting Up the TensorFlow Environment on MacOS

- ◆ On the Mac OS X system, it is recommended that you install homebrew first and then run the brew install python command, so that you can use Python in homebrew to install TensorFlow. Another recommended method is to install TensorFlow in virtualenv.

```
# In the current version, only CPU is supported.  
$ pip install https://storage.googleapis.com/tensorflow/mac/tensorflow-0.5.0-py2-none-any.whl
```

Setting Up the TensorFlow Environment on Linux

- ◆ The simplest installation mode on Linux is to use pip.

```
# Only the CPU version is used.
```

```
$ pip install https://storage.googleapis.com/tensorflow/linux/cpu/tensorflow-0.5.0-cp27-none-linux_x86_64.whl
```

```
# Start the version supporting GPU. (The version can only be installed after installing the CUDA sdk.
```

```
$ pip install https://storage.googleapis.com/tensorflow/linux/gpu/tensorflow-0.5.0-cp27-none-linux_x86_64.whl
```

Toolkit That TensorFlow Depends On (1)

- ◆ Protocol Buffer is one of the main toolkits that TensorFlow depends on. It is developed by Google to process structured data.
 - Protocol Buffers serialize structured data into a binary stream.
 - The format of the predefined data is Schema.proto.
 - The ProtoBuf data is 10%-30% of XML, and the parsing time is 20-100 times faster than XML.

```
name: Zhang San
id: 12345
email: zhangsan@abc.com
```

Toolkit That TensorFlow Depends On (1)

- ◆ Bazel (automatic build tool)
 - Compared with the traditional Makefile and Ant, Bazel is faster, more scalable, and flexible.
 - The basic concept is workspace. It can be considered a folder that contains the source code required for software compilation and the soft-make address of the output.
 - Only the py_binary, py_library, and py_test compilation modes are supported.

```
-rw-rw-r-- root root 208 BUILD
-rw-rw-r-- root root 48  hello_lib.py
-rw-rw-r-- root root 47  hello_main.py
-rw-rw-r-- root root 0   WORKSPACE
```

Running TensorFlow

- ◆ Open the terminal in Python and enter the following command:

```
$ python

>>> import tensorflow as tf
>>> hello = tf.constant('Hello, TensorFlow!')
>>> sess = tf.Session()
>>> print sess.run(hello)
Hello, TensorFlow!
>>> a = tf.constant(10)
>>> b = tf.constant(32)
>>> print sess.run(a+b)
42
>>>
```

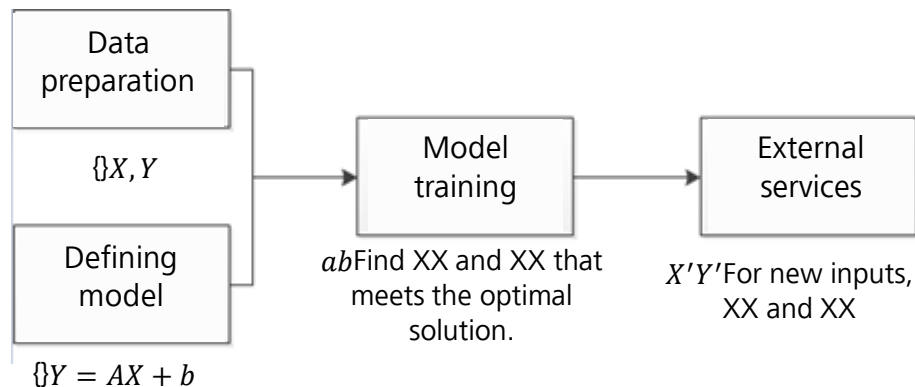



Contents

1. TensorFlow Overview
2. TensorFlow Characteristics
3. TensorFlow Basics
4. TensorFlow Modules
5. TensorFlow Development Environment Setup
- 6. Basic Development Steps Using TensorFlow**
7. Other Deep Learning Frameworks

Development Using TensorFlow

- ◆ Define the input node.
- ◆ Define "learning parameter" variables.
- ◆ Define the operation.
- ◆ Optimize functions and objectives.
- ◆ Initialize all variables.
- ◆ Perform model training to iterate and update parameters to the optimal solution.
- ◆ Test the model.
- ◆ Use the model.



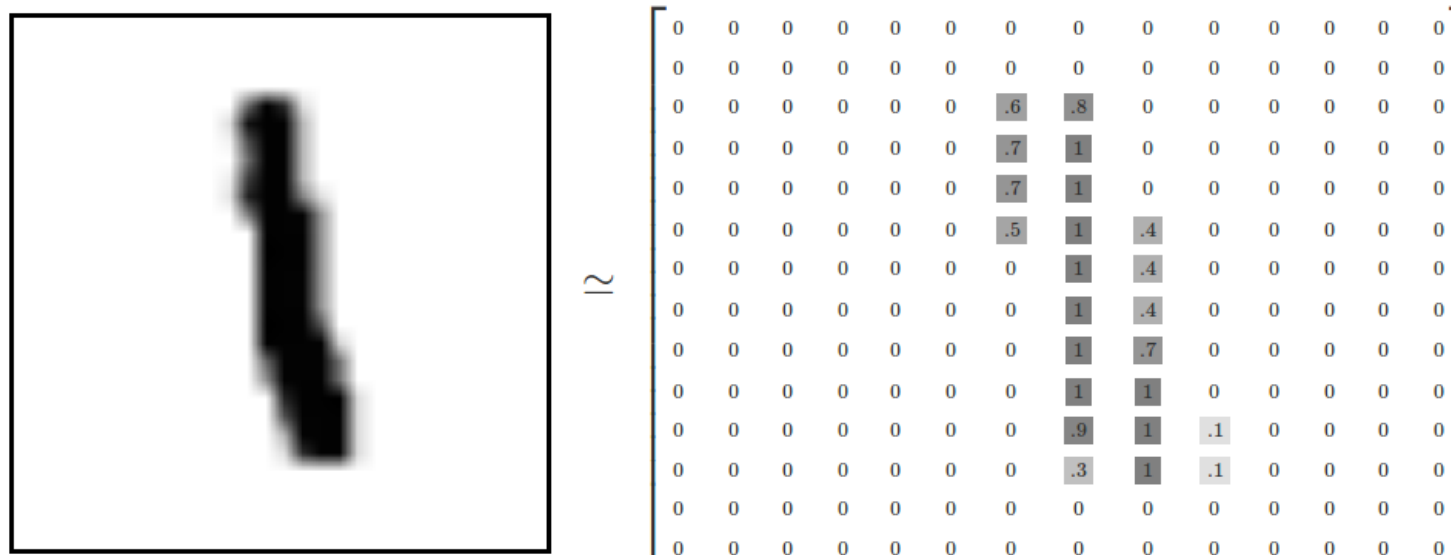
Preparing Data (1)

- ◆ MNIST is a classic problem in machine learning. The problem is solved by identifying 28 x 28 pixel grayscale images of handwritten digits as corresponding numbers ranging from 0 to 9.
- ◆ The MNIST dataset is an example of TensorFlow, you don't need to download it.



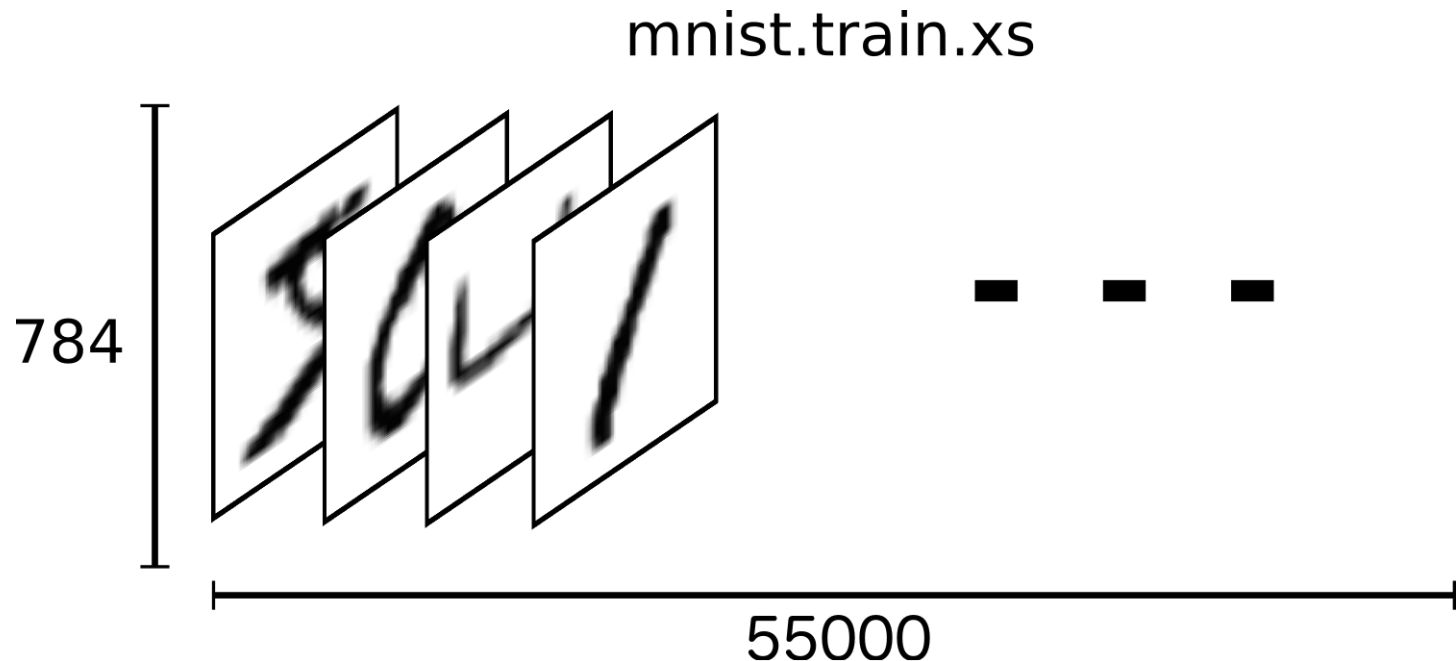
Preparing Data (2)

- ◆ An image is a matrix of 28 pixels x 28 pixels, which can be represented by a two-dimensional integer matrix of the same size.



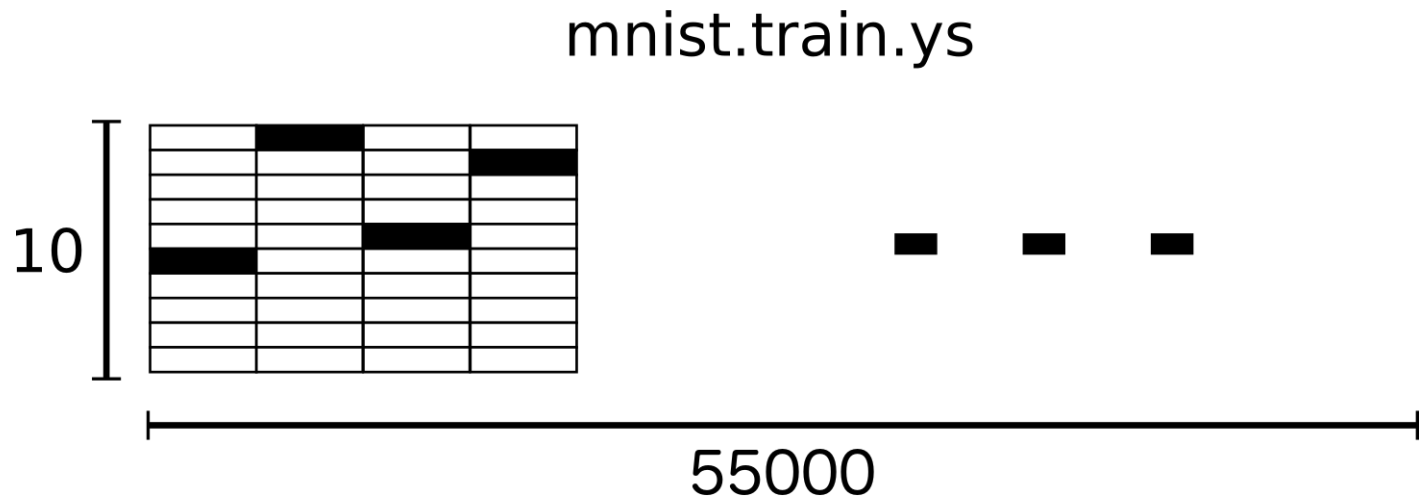
MNIST Dataset (1)

- ◆ The MNIST training dataset may be a 55000 x 784 tensor, that is, a multidimensional array. The first dimension represents the index of an image, and the second dimension represents the index of any pixel in the image (the pixel value in the tensor is between 0 and 1).



MNIST Dataset (2)

- ◆ `mnist.train.labels` is a two-dimensional array of 55000 x 10, as is shown in the following figure:



Defining the Input Node

- ◆ TensorFlow has the following methods for defining input nodes:
 - ◆ Defined by placeholders (commonly used)
 - ◆ Defined by dictionary types (used when there are many inputs)
 - ◆ Directly defined (seldom used)
- ◆ The input images are 550000 x 784 matrices. Therefore, create a placeholder x of [None, 784], and a placeholder y of [None and 10], and use the feed mechanism to input images and tags.

Defining Learning Parameters

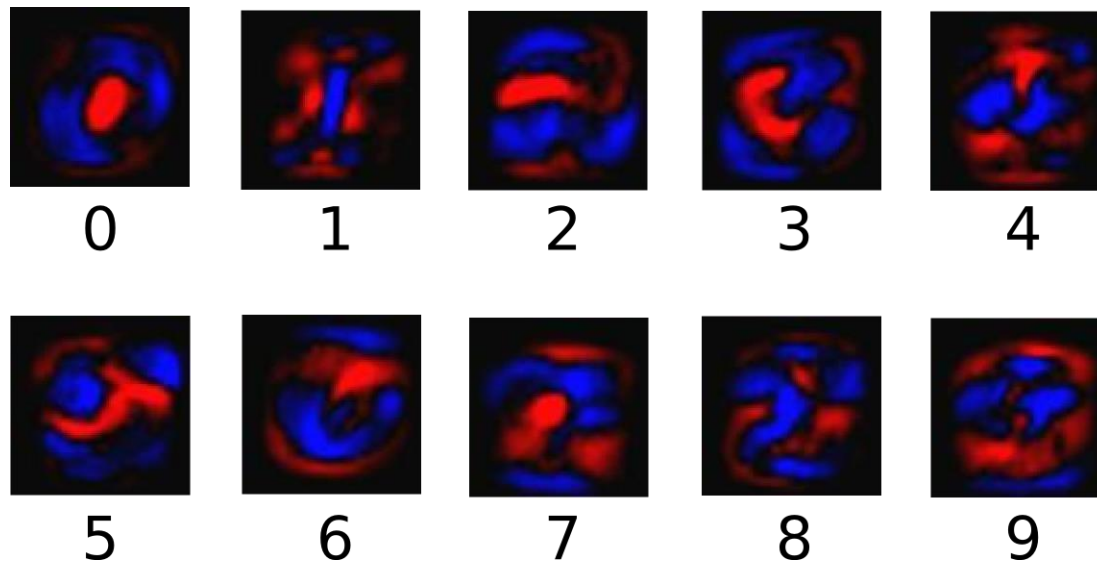
- ◆ Define "learning parameter" variables.
 - Directly defined
 - Defined by dictionary
- ◆ The learning parameters include weight values and bias values.
In TensorFlow, learning parameters are defined by variables.
- ◆ A variable represents a modifiable tensor, which is defined in the TensorFlow graph. Learning parameters defined by variables can be used for computing input values, or be modified in computation.

Defining the Operation

- ◆ The core process of creating a model determines the fitting effect of the model.
- ◆ Define the operation type by:
 - ◆ Defining the forward propagation model.
 - ◆ Defining the loss function (calculate the error between the output value and the target value and work in collaboration with backpropagation). The common loss functions in TensorFlow are mean square error and cross entropy.
 - ◆ Defining the backpropagation structure

Creating a Model

- ◆ The softmax model can be used to assign probabilities to different objects. Even after we train more elaborate models, the final step also needs to use softmax to assign probabilities.
- ◆ The image below shows the weight of each pixel in a picture that the model learns for a particular number class. Red represents negative weights and blue represents positive weights.



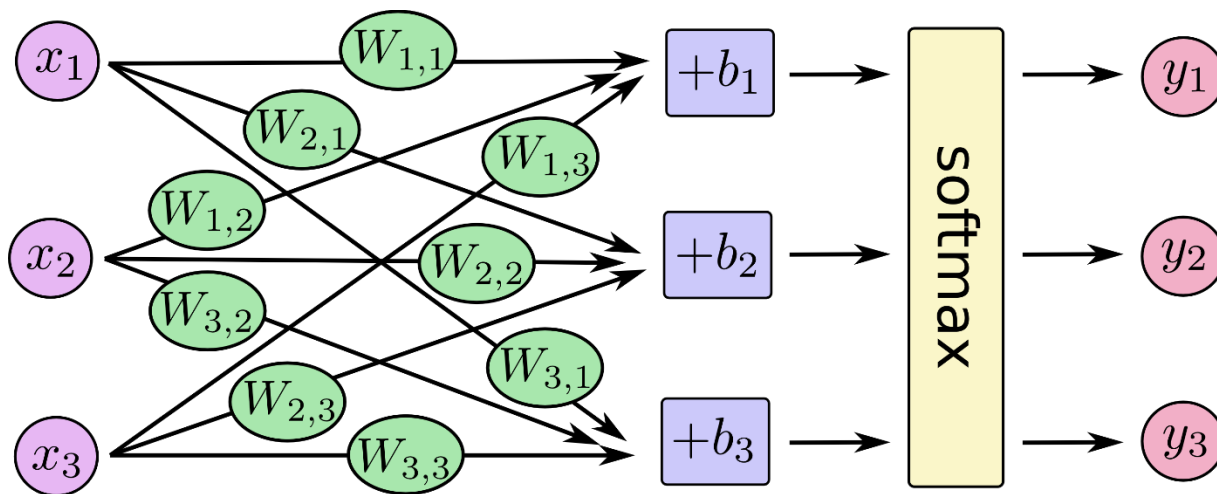
Defining a Model

- ◆ We also need to add an extra bias because the input tends to have some extraneous interference. So for a given input image x , the evidence proving that it represents the digit i can be expressed as: $evidence_i = \sum_j W_{i,j}x_j + b_i$
- ◆ Wherein W_i indicates weight. b_i indicates the bias of the numeric class i . j indicates the pixel index of a given image, x , and is used for summing the pixels. Then, use the softmax function to convert the evidence to the probability y .

$$y = softmax(evidence)$$

Regression Model (1)

- ◆ Add a bias to the weighted sum of the inputs x_i , and then input the biases to the softmax function.



Regression Model (2)

- ◆ In actual cases, the calculation process is expressed using vectors, as shown in the following figure:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax} \left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right)$$

Further, it can be written in a more compact way:

$$y = \text{softmax}(W_x + b).$$

Optimizing Functions and Objectives

- ◆ This process is completed in backpropagation. The backpropagation process is to transfer the error back along the reverse direction of forward propagation, involving L1 and L2 regularization, impulse adjustment, learning rate adaptation, and the Adam random gradient descent algorithm.

Initializing

- ◆ To create a model, you need to create a huge number of weights and biases. The weights in the model should be initialized with a small amount of noise to break the symmetry and avoid the zero gradient.
- ◆ To avoid repeated initialization during model creation, we define two functions for initialization.

```
def weight_variable(shape):  
    initial = tf.truncated_normal(shape, stddev=0.1)  
    return tf.Variable(initial)  
  
def bias_variable(shape):  
    initial = tf.constant(0.1, shape=shape)  
    return tf.Variable(initial)
```

Implementing the Model

- ◆ In order to train our model, we first need to define an indicator to evaluate this model is good. In fact, in machine learning, we usually define indicators to indicate that a model is bad, this indicator is called cost or loss, and then try to minimize this indicator. The cost function is the cross-entropy function.

Iterating and Training the Model to Obtain the Optimal Solution

- ◆ Here, we require TensorFlow to use a gradient descent algorithm to minimize the cross-entropy at a learning rate of 0.01. The gradient descent algorithm is a simple learning process. TensorFlow simply moves each variable little by little in a direction that keeps costs down.
- ◆ Then, start training the model, here we let the model cycle training 1000 times!

```
for i in range(1000):  
    batch_xs, batch_ys = mnist.train.next_batch(100)  
    sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})
```

Testing the Model

- ◆ Use the test data in MNIST to test the model. First, find out the labels that are correct. Since the tag is a vector consisting of 0 and 1, the maximum value, that is, 1, of the index position is located a category label. To determine the proportion of the correct predictors, we can convert the Boolean values into floating point numbers and then average them. we calculate the correctness of the learned model on the test dataset.
- ◆ For `tf.argmax(y, 1)`, the return of the model prediction for any input `x` to the tag value, and `tf.argmax(y_, 1)`, the representative of the correct label, we can use `tf.equal` to test whether the prediction is a true tag match (the same as the index position indicates a match).

```
correct_prediction = tf.equal(tf.argmax(y,1), tf.argmax(y_,1))
```

Evaluating the Model

- ◆ TensorFlow needs to add the model evaluation to the computation graph, and then call the model evaluation after model training.
- ◆ The model evaluation mainly involves the following indicators: average accuracy, identification time, and loss decrease.
- ◆ During model training, model evaluation helps gain insight into the model algorithms and provide prompt information to debug, improve, or modify the whole model. After training the model, the performance of the model needs to be evaluated quantitatively.

Using the Model (1)

- ◆ Save the model:
 - Create a saver and a path, and invoke the save command to save the parameters in the session.
- ◆ Use the model:
 - Replace the nodes of loss value with the output nodes. This is similar to what has been done in model testing.
- ◆ Read the model:
 - Read the model. Add images to the model for prediction, and display the images and their corresponding tags.

Using the Model (2)

- ◆ Read the model. Add two images to the model to predict results and display the labels corresponding to the two images.
- ◆ The definition of the network model remains unchanged during code execution. Create a new session. All operations are performed in the new session.

```
After 0 training step(s), validation accuracy using average model is 0.103
After 1000 training step(s), validation accuracy using average model is 0.9044
After 2000 training step(s), validation accuracy using average model is 0.9174
After 3000 training step(s), validation accuracy using average model is 0.9258
After 4000 training step(s), validation accuracy using average model is 0.93
After 5000 training step(s), validation accuracy using average model is 0.9346
After 6000 training step(s), validation accuracy using average model is 0.94
After 7000 training step(s), validation accuracy using average model is 0.9422
After 8000 training step(s), validation accuracy using average model is 0.9472
After 9000 training step(s), validation accuracy using average model is 0.9498
After 10000 training step(s), test accuracy using average model is 0.9475
```

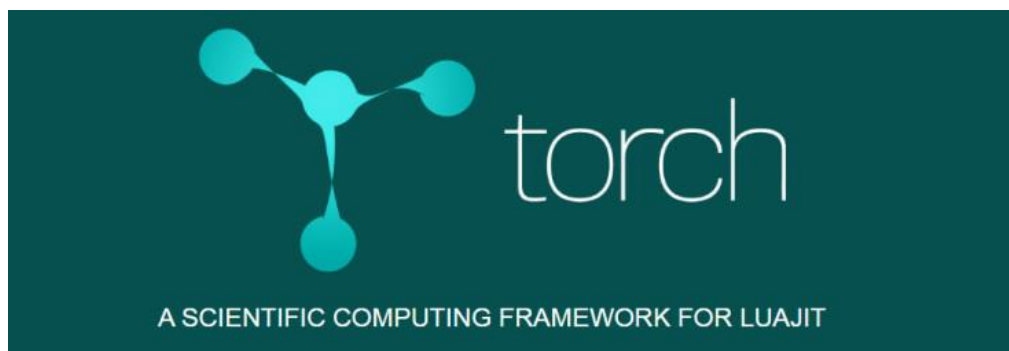


Contents

1. TensorFlow Overview
2. TensorFlow Characteristics
3. TensorFlow Basics
4. TensorFlow Modules
5. TensorFlow Development Environment Setup
6. Basic Development Steps Using TensorFlow
- 7. Other Deep Learning Frameworks**

Other Deep Learning Frameworks

- ◆ Theano
- ◆ Torch
- ◆ Keras
- ◆ DeepLearning4j
- ◆ Caffe
- ◆ MXNet
- ◆ CNTK



Hardware Supported by Deep Learning Frameworks

Property	Caffe	Neon	TensorFlow	Theano	Torch
Core	C++	Python	C++	Python	Lua
CPU	✓	✓	✓	✓	✓
Multi-threaded CPU	✓Blas	x Only data loader	✓Eigen	✓Blas, conv2D, limited OpenMP	✓Widely used
GPU	✓	✓customized Nvidia backend	✓	✓	✓
Multi-GPU	✓(only data parallel)	✓	✓Most flexible	x Experimental version available	✓
Nvidia cuDNN	✓	x	✓	✓	✓
Quick deploy. on standard models	✓Easiest	✓	✓	x Via secondary libraries	✓
Auto. gradient computation	✓	✓Supports Op-Tree	✓	✓Most flexible (also over loops)	✓

Advantages and Disadvantages of the Popular Deep Learning Frameworks

◆ Caffe:

- Advantages: Easy to get started, fast, modular, open, and supported by a good community.
- Disadvantages: The layer needs to be defined in C++, and the model needs to be defined using Protobuf. The configuration file of the Caffe cannot be programmed to adjust the hyperparameter. Caffe only supports single-machine multi-GPU training, but does not natively support distributed training.

◆ Torch:

- Advantages: The modeling is simple and highly modular, with fast and efficient GPU support. It uses LuaJIT to connect to C++ and numerical optimization programs. It can be embedded in the interfaces of iOS, Android, and FPGA.
- Disadvantages: Some interfaces are not comprehensive, requiring LuaJIT to use the Lua language. Therefore, Torch lacks universality.



Conclusion

- ◆ This chapter describes the concepts, characteristics, basic knowledge, and commonly used modules of TensorFlow. It then introduces how to set up the development environment and create TensorFlow programs. It also describes the basic development steps using TensorFlow and internal mechanisms of TensorFlow. In the end, it touches upon other deep learning frameworks.

Quiz

1. Which of the following are application scenarios of TensorFlow? ()
 - A. AlphaGo
 - B. Facial recognition
 - C. Artistic style transfer
 - D. Self-driving



More Information

- ◆ Huawei E-Learning website:
 - <http://support.huawei.com/learning/Index!toTrainIndex>
- ◆ Huawei Support case library:
 - <http://support.huawei.com/enterprise/servicecenter?lang=zh>

Thank You

www.huawei.com