Lab 8 – TensorFlow (cont.)

www.huawei.com

HUAWEI

Contents

Common Modules of TensorFlow 2.x (1)

- tf: Functions in the tf module are used to perform common arithmetic operations, such as **tf.abs** (calculating an absolute value), **tf.add** (adding elements one by one), and **tf.concat** (concatenating tensors). Most operations in this module can be performed by NumPy.
- **tf.errors**: error type module of TensorFlow
- **tf.data**: implements operations on datasets.
  - Input pipes created by tf.data are used to read training data. In addition, data can be easily input from memories (such as NumPy).
- **tf.distributions**: implements various statistical distributions.
  - The functions in this module are used to implement various statistical distributions, such as Bernoulli distribution, uniform distribution, and Gaussian distribution.

Common Modules of TensorFlow 2.x (2)

- **tf.io.gfile:** implements operations on files.
  - Functions in this module can be used to perform file I/O operations, copy files, and rename files.
- **tf.image:** implements operations on images.
  - Functions in this module include image processing functions. This module is similar to OpenCV, and provides functions related to image luminance, saturation, phase inversion, cropping, resizing, image format conversion (RGB to HSV, YUV, YIQ, or gray), rotation, and sobel edge detection. This module is equivalent to a small image processing package of OpenCV.
- **tf.keras:** a Python API for invoking Keras tools.
  - This is a large module that enables various network operations.

Keras Interface

- TensorFlow 2.x recommends Keras for network building. Common neural networks are included in **Keras.layers**.

Common Keras Methods and Interfaces

- The following describes common methods and interfaces of **tf.keras** by focusing on code. The main content is as follows:
  - **Dataset processing**: datasets and preprocessing
  - **Neural network model creation**: Sequential, Model, Layers...
  - **Network compilation**: compile, Losses, Metrics, and Optimizers
  - **Network training and evaluation**: fit, fit_generator, and evaluate

Contents

TensorFlow Environment Setup in Windows 10

- Environment setup in Windows 10:
  - Operating system: Windows 10
  - pip software built in Anaconda 3 (adapting to Python 3)
  - TensorFlow installation:
    - Open Anaconda Prompt and run the pip command to install TensorFlow.
    - Run pip install TensorFlow in the command line interface.

```
(base) C:\Users\ThinkPad>pip install tensorflow
Requirement already satisfied: tensorflow in d:\vs\anaconda3_64\lib\site-packages (1.14.0)
Requirement already satisfied: astor>=0.6.0 in c:\users\thinkpad\appdata\roaming\python\python36\site-packages (from ten
sorflow) (0.8.0)
Requirement already satisfied: keras-preprocessing>=1.0.5 in c:\users\thinkpad\appdata\roaming\python\python36\site-pack
ages (from tensorflow) (1.1.0)
Requirement already satisfied: six>=1.10.0 in d:\vs\anaconda3_64\lib\site-packages (from tensorflow) (1.11.0)
Requirement already satisfied: keras-applications>=1.0.6 in c:\users\thinkpad\appdata\roaming\python\python36\site-packa
ges (from tensorflow) (1.0.8)
Requirement already satisfied: grpcio>=1.8.6 in d:\vs\anaconda3_64\lib\site-packages (from tensorflow) (1.23.0)
Requirement already satisfied: gast>=0.2.0 in d:\vs\anaconda3_64\lib\site-packages (from tensorflow) (0.3.2)
Requirement already satisfied: tensorflow-estimator<1.15.0rc0,>=1.14.0rc0 in c:\users\thinkpad\appdata\roaming\python\py
thon36\site-packages (from tensorflow) (1.14.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\thinkpad\appdata\roaming\python\python36\site-packages (from
 tensorflow) (1.1.0)
```

TensorFlow Environment Setup in Ubuntu/Linux

- The simplest way for installing TensorFlow in Linux is to run the pip command.

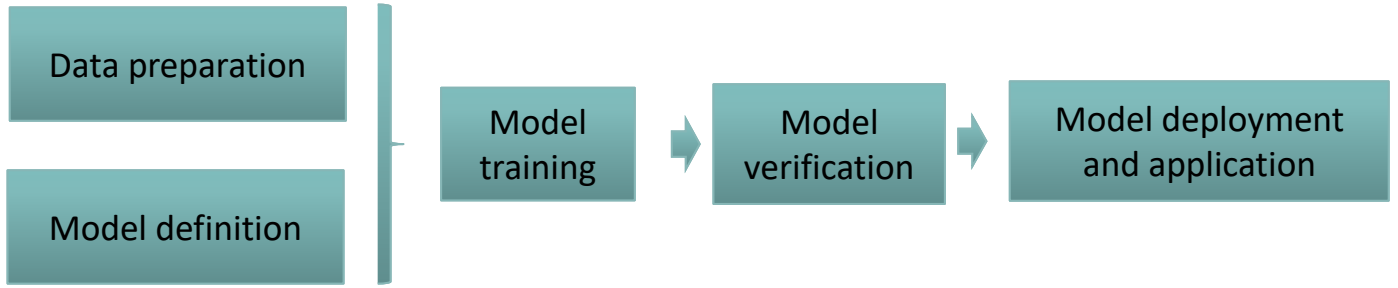
```
czy@czy-System-Product-Name:~$ pip install tensorflow==2.0.0-alpha0
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting tensorflow==2.0.0-alpha0
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/29/39/f99185d3913
afcfe1dcdb0629c2ffc4ecfb0e4c14ca210d620e56c/tensorflow-2.0.0a0-cp36-cp36m-
nux1_x86_64.whl (79.9MB)
    37% |                                | 29.8MB 98.5MB/s eta 0:00:01
```

- pip command: pip install TensorFlow==2.1.0

Contents

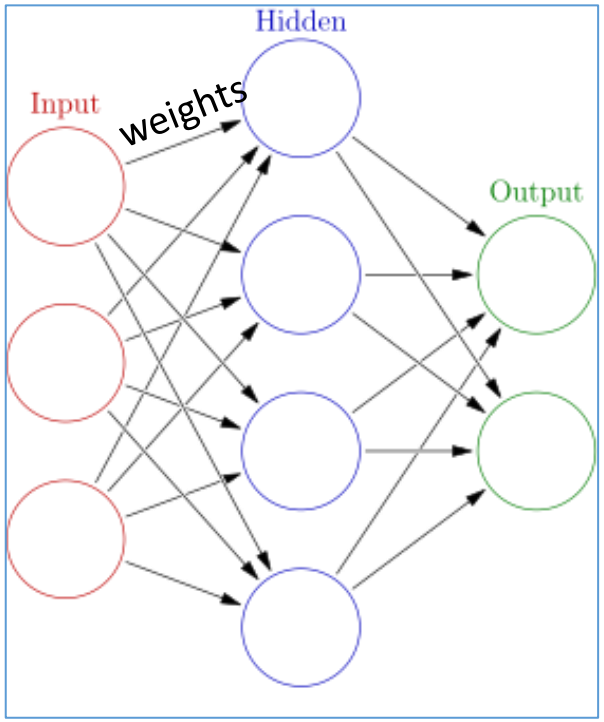
TensorFlow Development Process

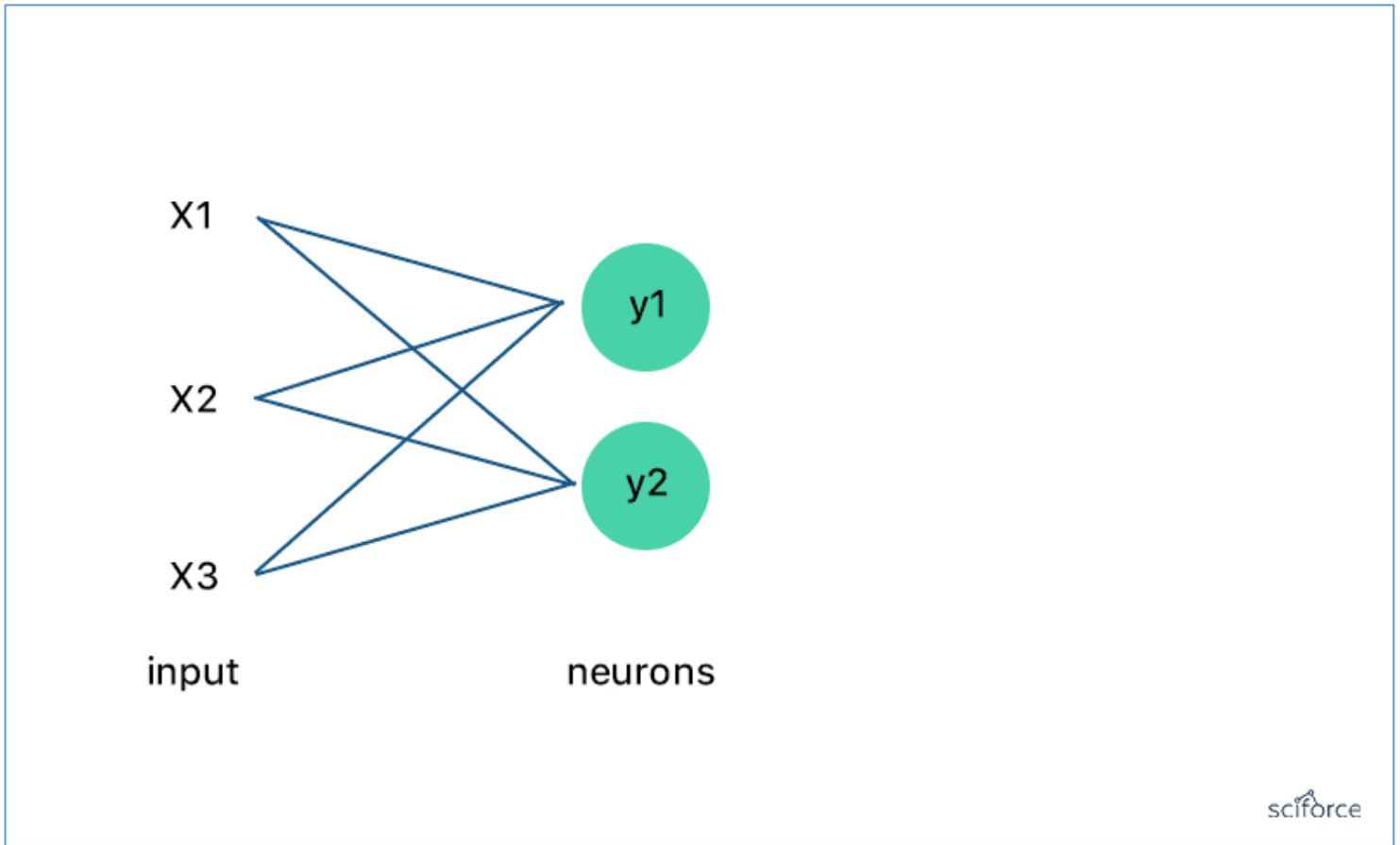
- Data preparation
  - Data exploration
  - Data processing
- Network construction
  - Defining a network structure.
  - Defining loss functions, selecting optimizers, and defining model evaluation indicators.
- Model training and verification
- Model saving
- Model restoration and invoking



Artificial Neural Network (ANN)

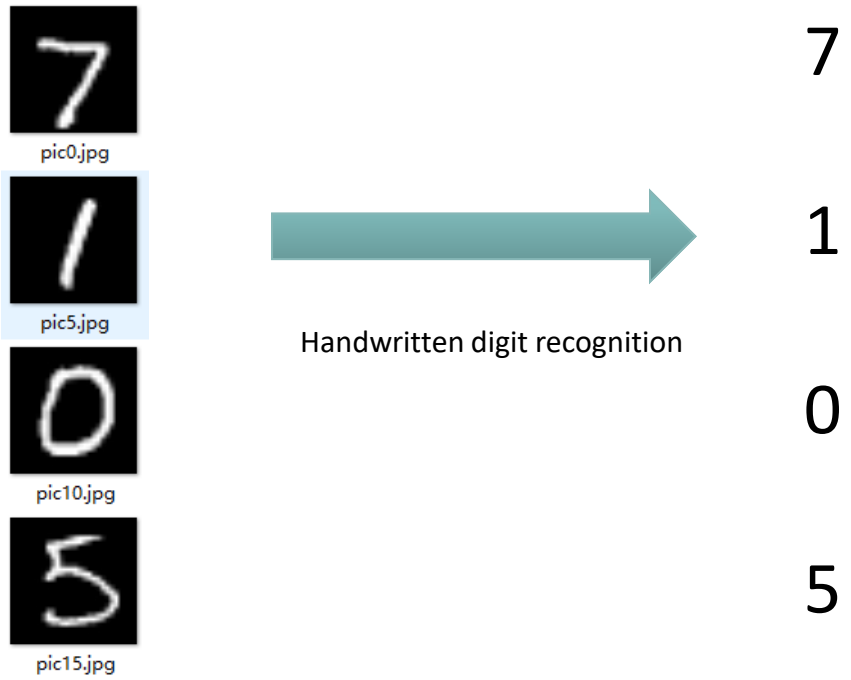
- Similar to linear regression, a simple ANN provides a supervised learning algorithm.
- ANN loosely model the neurons in a biological brain that transfer signals to each other. The following figures show how a general ANN looks like and how its output is calculated.





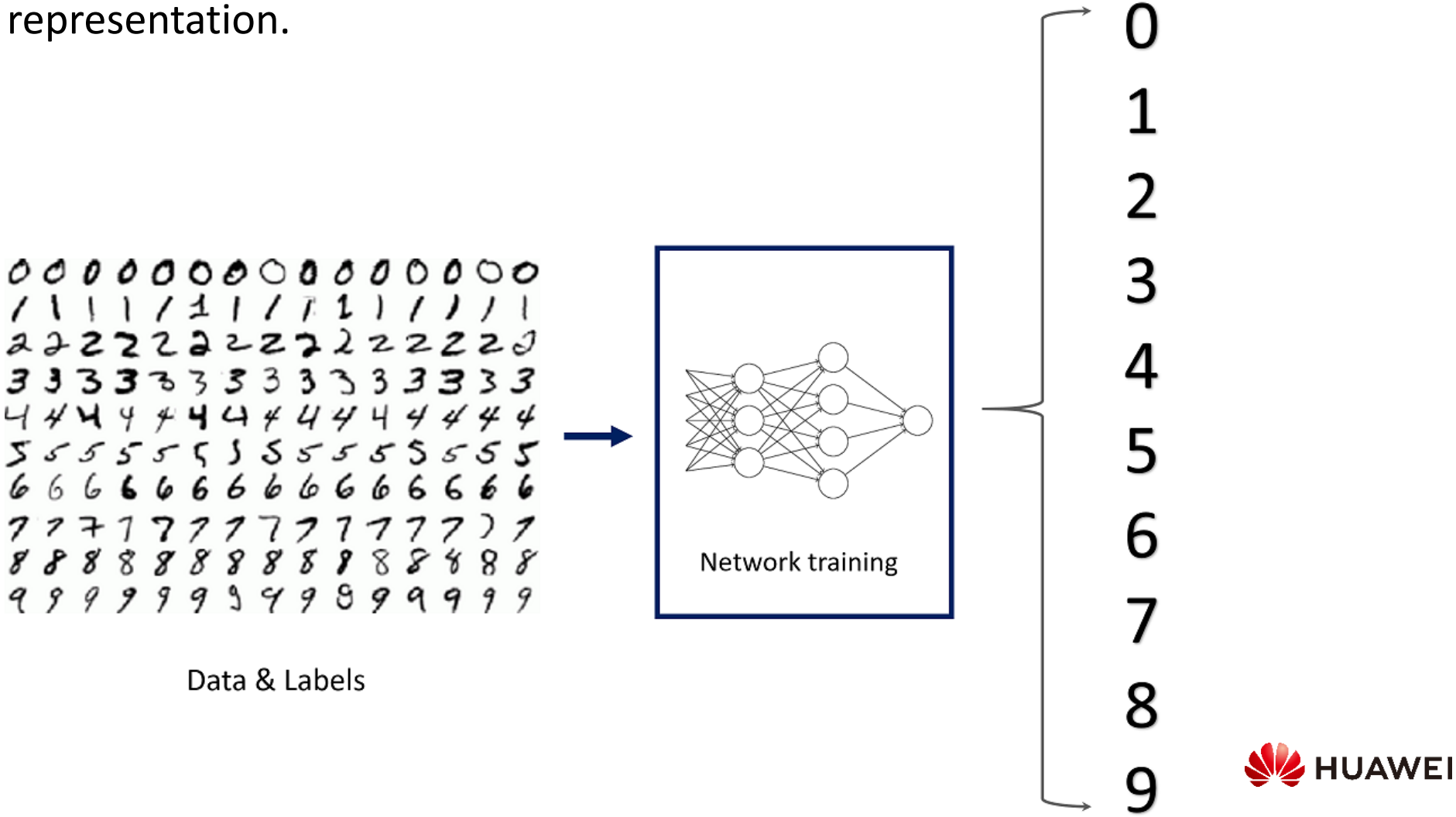
Project Description (1)

- Handwritten digit recognition is a common image recognition task where computers recognize text in handwritten images. Different from printed fonts, handwriting of different people has different sizes and styles, making it difficult for computers to recognize handwriting. This project applies deep learning and TensorFlow tools to train and build models based on **the MNIST handwriting dataset**.



Project Description (2)

- We will be solving the classification task and try to recognize the actual digit from its handwritten representation.



Data & Labels

Data Preparation

- MNIST datasets
  - Download the MNIST datasets from http://yann.lecun.com/exdb/mnist/.
  - The MNIST datasets consist of a training set and a test set.
    - Training set: 60,000 handwriting images and corresponding labels
    - Test set: 10,000 handwriting images and corresponding labels

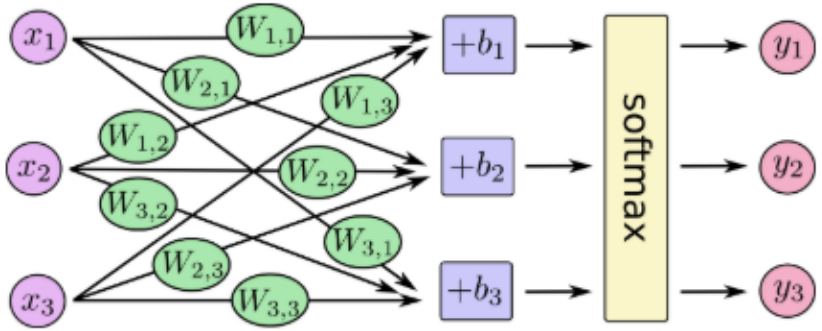
Examples

Corresponding labels

[0,0,0,0,0,1, 0,0,0,0]	[0,0,0,0,0, 0,0,0,0,1]	[0,0,0,0,0,0 ,0,1,0,0]	[0,0,0,1,0, 0,0,0,0,0]	[0,0,0,0,1,0, 0,0,0,0]

Network Structure Definition

- The process of model establishment is the core process of network structure definition.
- The network operation process defines how model output is calculated based on input.



- Matrix multiplication and vector addition are used to express the calculation process of softmax.

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

Network Compilation (1)

- To feed the network with the training data, we need to flatten the digit images.
- Depending on the phase (training or testing), different examples will be pushed through the classifier.
- The training process will be based on the labels while comparing them to the current predictions.

Network Compilation (2)

Model compilation involves the following two parts:

- **Loss function selection**
  - In machine learning/deep learning, an indicator needs to be defined to indicate whether a model is proper. This indicator is called cost or loss, and is minimized as far as possible.
  - We would like to minimize the difference between the network predictions and actual labels' values. In deep learning, we often use a technique called Cross entropy to define the loss.
- **Gradient descent method**
  - A loss function is constructed for an original model needs to be optimized by using an optimization algorithm, to find optimal parameters and further minimize a value of the loss function. Among optimization algorithms for solving machine learning parameters, the gradient descent-based optimization algorithm (Gradient Descent) is usually used.
  - Gradient descent optimiser will work in several steps adjusting the values of the ANN weights.

```
Optimizer = optimizers.Adam(0.001)
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=Optimizer,
              metrics=['accuracy'])
```

Model Training

- Training process:
  - All training data is trained through batch iteration or full iteration. In the experiment, all data is trained 10 times.
  - In TensorFlow, model.fit is used for training, where epoch indicates the number of training iterations.

```
model.fit (x_train, y_train, batch_size=128, epochs=10, verbose=1)
```

```
Epoch 1/10
469/469 [==============================] - 5s 10ms/step - loss: 0.4196 - accuracy: 0.8800
Epoch 2/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0861 - accuracy: 0.9747
Epoch 3/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0527 - accuracy: 0.9829
Epoch 4/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0376 - accuracy: 0.9885
Epoch 5/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0304 - accuracy: 0.9903
Epoch 6/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0242 - accuracy: 0.9925
Epoch 7/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0208 - accuracy: 0.9927
Epoch 8/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0147 - accuracy: 0.9949
Epoch 9/10
469/469 [==============================] - 5s 10ms/step - loss: 0.0192 - accuracy: 0.9938
Epoch 10/10
469/469 [==============================] - 4s 10ms/step - loss: 0.0137 - accuracy: 0.9954
<tensorflow.python.keras.callbacks.History at 0x7fef86191b90>
```

Model Evaluation

- You can test the model using the test set, compare predicted results with actual ones, and find correctly predicted labels, to calculate the accuracy of the test set.

```
model.evaluate(x_test, y_test, verbose=0)
```

```
10000/10000 [==============================] - 0s 42us/sample - loss: 0.0779 - categorical_accuracy: 0.9764
[0.07786676207473502, 0.9764]
```

Loss value

Accuracy

More Information

Official TensorFlow website: https://tensorflow.google.cn

Thank you.

把数字世界带入每个人、每个家庭、
每个组织，构建万物互联的智能世界。

Bring digital to every person, home, and organization for a fully connected, intelligent world.

HUAWEI