

第五部分

实战 TensorFlow 手写体数字识别



扫描二维码

试看/购买《TensorFlow 快速入门与实战》视频课程

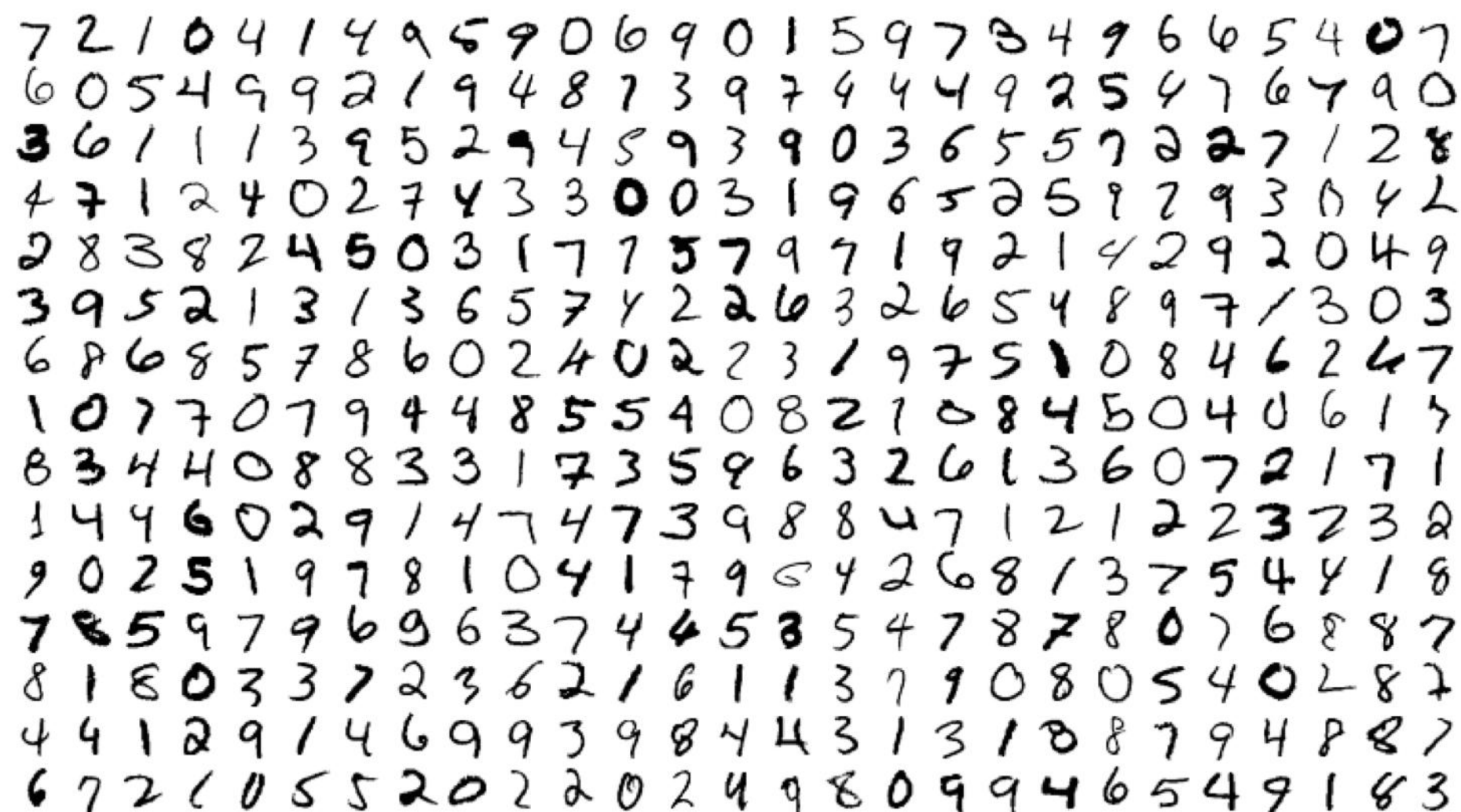
第五部分 目录

- 手写体数字 MNIST 数据集介绍
- MNIST Softmax 网络介绍
- 实战 MNIST Softmax 网络
- MNIST CNN 网络介绍
- 实战 MNIST CNN 网络

手写体数字 MNIST 数据集介绍

MNIST 数据集介绍

[MNIST](#) 是一套手写体数字的图像数据集，包含 60,000 个训练样例和 10,000 个测试样例，由纽约大学的 Yann LeCun 等人维护。



获取 MNIST 数据集

THE MNIST DATABASE of handwritten digits

Yann LeCun, Courant Institute, NYU

Corinna Cortes, Google Labs, New York

Christopher J.C. Burges, Microsoft Research, Redmond

The MNIST database of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting.

Four files are available on this site:

<u>train-images-idx3-ubyte.gz</u> :	training set images (9912422 bytes)
<u>train-labels-idx1-ubyte.gz</u> :	training set labels (28881 bytes)
<u>t10k-images-idx3-ubyte.gz</u> :	test set images (1648877 bytes)
<u>t10k-labels-idx1-ubyte.gz</u> :	test set labels (4542 bytes)

MNIST 手写体数字介绍

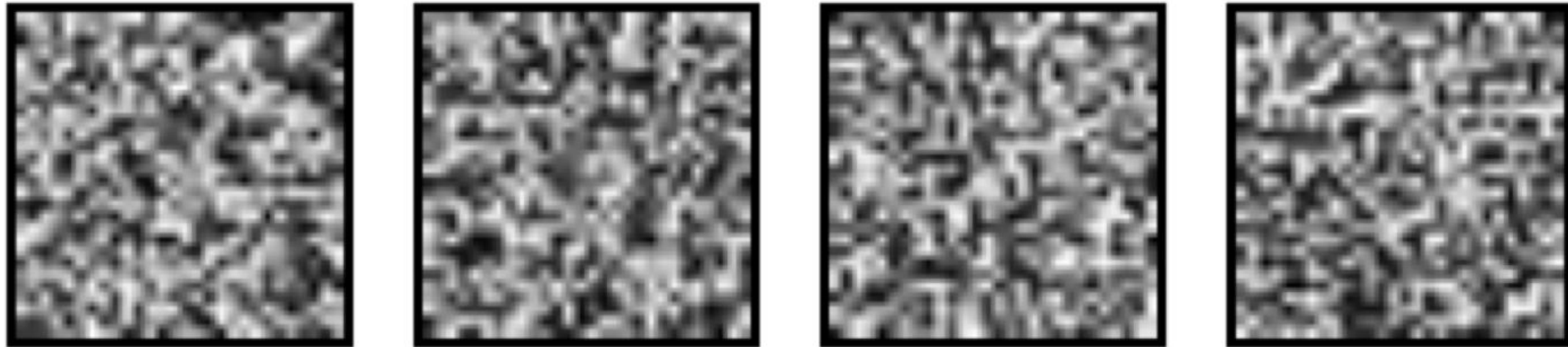
MNIST 图像数据集使用形如 $[28, 28]$ 的二阶数组来表示每个手写体数字，数组中的每个元素对应一个像素点，即每张图像大小固定为 28×28 像素。



MNIST 手写体数字介绍

由于每张图像的尺寸都是28x28像素，为了方便连续存储，我们可以将形如 $[28, 28]$ 的二阶数组“摊平”成形如 $[784,]$ 的一阶数组，可以表示 $256 * 256 * \dots * 256 = 256^{784}$ 张不同的图像。

但这些图像并非每一张都代表有效的手写体数字，其中绝大部分都是如下的噪声图：



下载和读取 MNIST 数据集

一个曾广泛使用（如 chapter-2/basic-model.ipynb），如今被废弃的（**deprecated**）方法：

```
In [7]: import tensorflow as tf

from tensorflow.examples.tutorials.mnist import input_data

# 导入数据
mnist = input_data.read_data_sets('./mnist/dataset/')
```

```
WARNING:tensorflow:From /Users/Patient/tf-course/py3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:262: extract_images (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
```

```
Instructions for updating:
```

```
Please use tf.data to implement this functionality.
```

```
Extracting ./mnist/dataset/train-images-idx3-ubyte.gz
```

```
WARNING:tensorflow:From /Users/Patient/tf-course/py3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:267: extract_labels (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
```

```
Instructions for updating:
```

```
Please use tf.data to implement this functionality.
```

```
Extracting ./mnist/dataset/train-labels-idx1-ubyte.gz
```

```
Extracting ./mnist/dataset/t10k-images-idx3-ubyte.gz
```

```
Extracting ./mnist/dataset/t10k-labels-idx1-ubyte.gz
```

```
WARNING:tensorflow:From /Users/Patient/tf-course/py3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:290: DataSet.__init__ (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
```

```
Instructions for updating:
```

```
Please use alternatives such as official/mnist/dataset.py from tensorflow/models.
```


下载和读取 MNIST 数据集

一个曾广泛使用（如 chapter-2/basic-model.ipynb），如今被废弃的（**deprecated**）方法：

```
In [7]: import tensorflow as tf

        from tensorflow.examples.tutorials.mnist import input_data

        # 导入数据
        mnist = input_data.read_data_sets('./mnist/dataset/')
```

```
WARNING:tensorflow:From /Users/Patient/tf-course/py3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:262: extract_images (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
```

```
Instructions for updating:
```

```
Please use tf.data to implement this functionality.
```

```
Extracting ./mnist/dataset/train-images-idx3-ubyte.gz
```

```
WARNING:tensorflow:From /Users/Patient/tf-course/py3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:267: extract_labels (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
```

```
Instructions for updating:
```

```
Please use tf.data to implement this functionality.
```

```
Extracting ./mnist/dataset/train-labels-idx1-ubyte.gz
```

```
Extracting ./mnist/dataset/t10k-images-idx3-ubyte.gz
```

```
Extracting ./mnist/dataset/t10k-labels-idx1-ubyte.gz
```

```
WARNING:tensorflow:From /Users/Patient/tf-course/py3/lib/python3.6/site-packages/tensorflow/contrib/learn/python/learn/datasets/mnist.py:290: DataSet.__init__ (from tensorflow.contrib.learn.python.learn.datasets.mnist) is deprecated and will be removed in a future version.
```


```
Instructions for updating:
```

```
Please use alternatives such as official/mnist/dataset.py from tensorflow/models.
```




tf.contrib.learn 模块已被废弃

需要注意的是，tf.contrib.learn 整个模块均已被废弃：

Branch: r1.12 ▾ tensorflow / tensorflow / contrib / learn / README.md Find file Copy path

 martinwicke Deprecate tf.contrib.learn. c7caa2d on Feb 28, 2018

1 contributor

144 lines (120 sloc) | 6.94 KB Raw Blame History   

EVERYTHING IN THIS DIRECTORY IS DEPRECATED.

Using functions or classes will result in warnings.

Instructions for converting to current alternatives are included in the warnings. A high-level overview is below.

使用 Keras 加载 MNIST 数据集

```
tf.keras.datasets.mnist.load_data(path='mnist.npz')
```

Arguments:

- **path**: 本地缓存 MNIST 数据集 (mnist.npz) 的相对路径 (~/.keras/datasets)

Returns:

Tuple of **Numpy** arrays: `(x_train, y_train), (x_test, y_test)`.

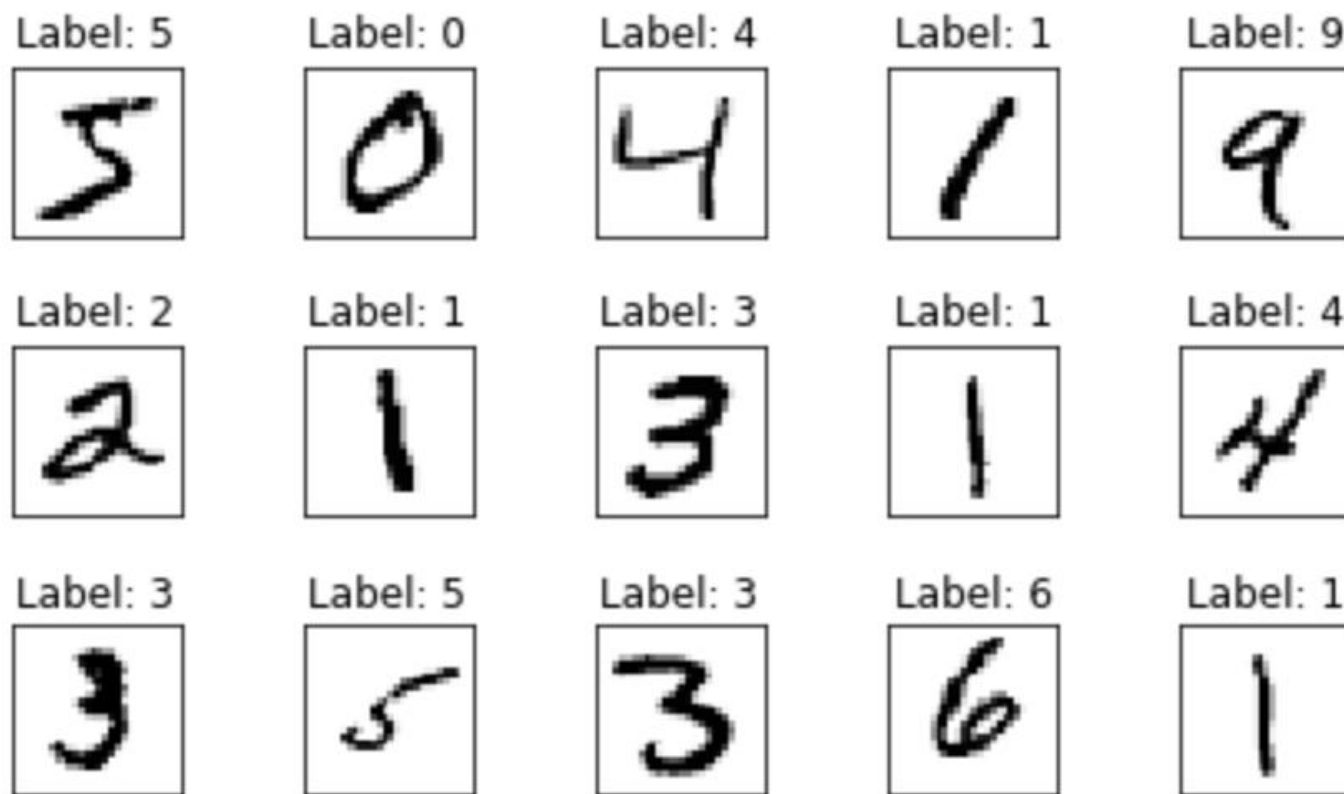
```
In [5]: from keras.datasets import mnist  
  
(x_train, y_train), (x_test, y_test) = mnist.load_data('mnist/mnist.npz')
```

```
In [9]: print(x_train.shape, y_train.shape)  
        print(x_test.shape, y_test.shape)  
  
(60000, 28, 28) (60000,)  
(10000, 28, 28) (10000,)
```

MNIST 数据集 样例可视化

```
In [58]: import matplotlib.pyplot as plt

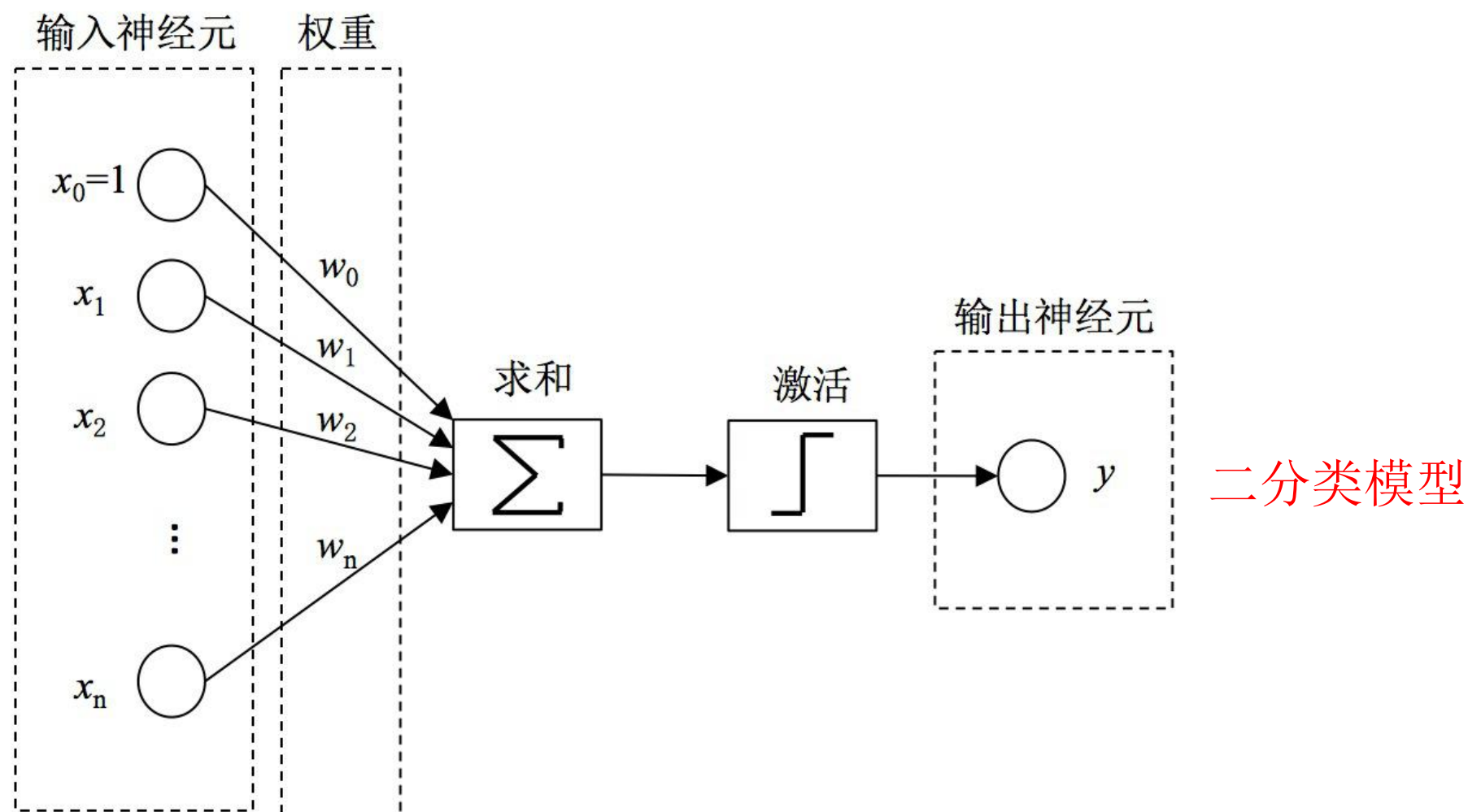
fig = plt.figure()
for i in range(15):
    plt.subplot(3,5,i+1) # 绘制前15个手写体数字, 以3行5列子图形式展示
    plt.tight_layout() # 自动适配子图尺寸
    plt.imshow(x_train[i], cmap='Greys') # 使用灰色显示像素灰度值
    plt.title("Label: {}".format(y_train[i])) # 设置标签为子图标题
    plt.xticks([]) # 删除x轴标记
    plt.yticks([]) # 删除y轴标记
```



MNIST Softmax 网络介绍

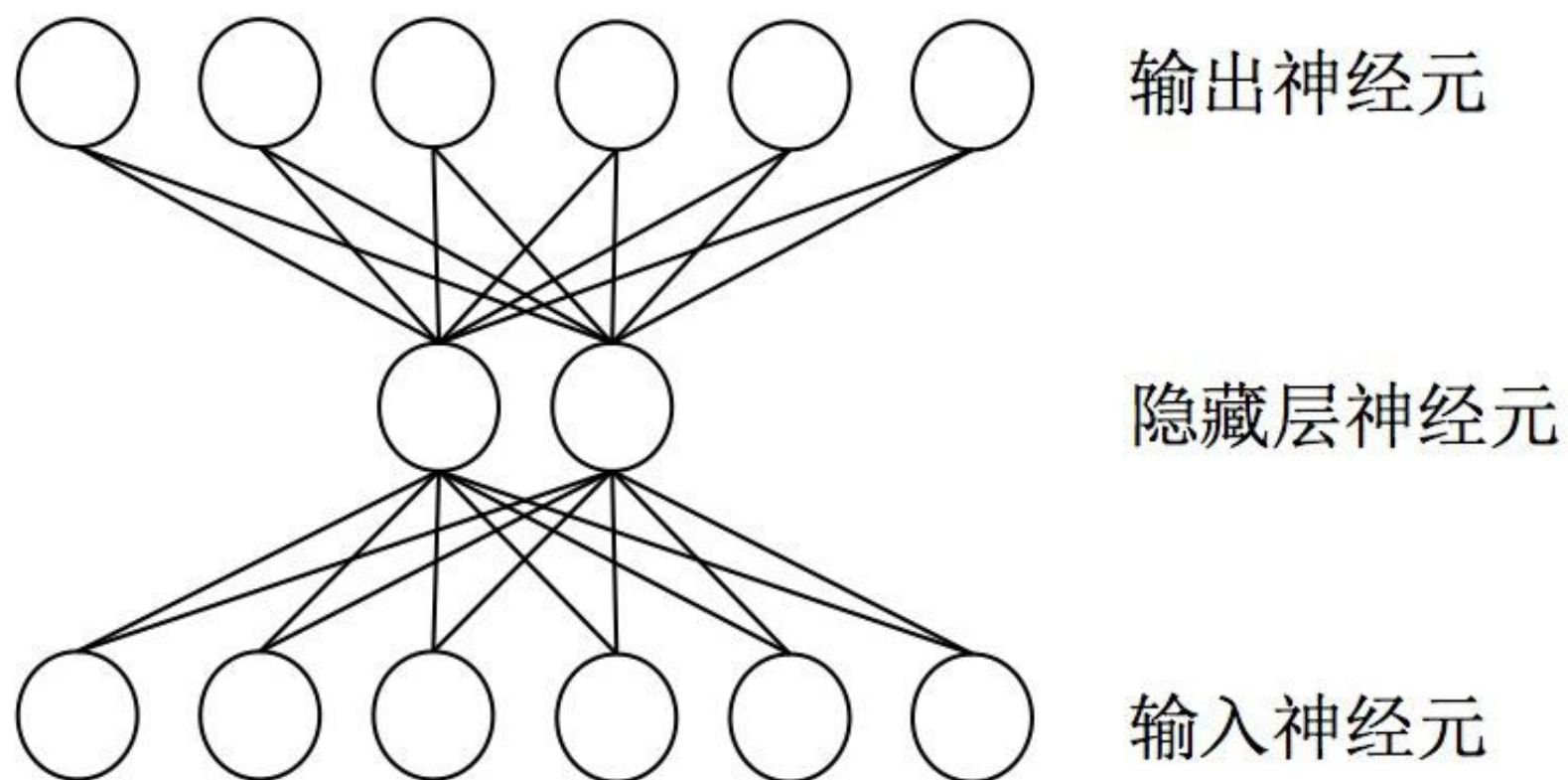
感知机模型

1957年，受 Warren McCulloch 和 Walter Pitts 在神经元建模方面工作的启发，心理学家 Frank Rosenblatt 参考大脑中神经元信息传递信号的工作机制，发明了神经感知机模型 **Perceptron**。

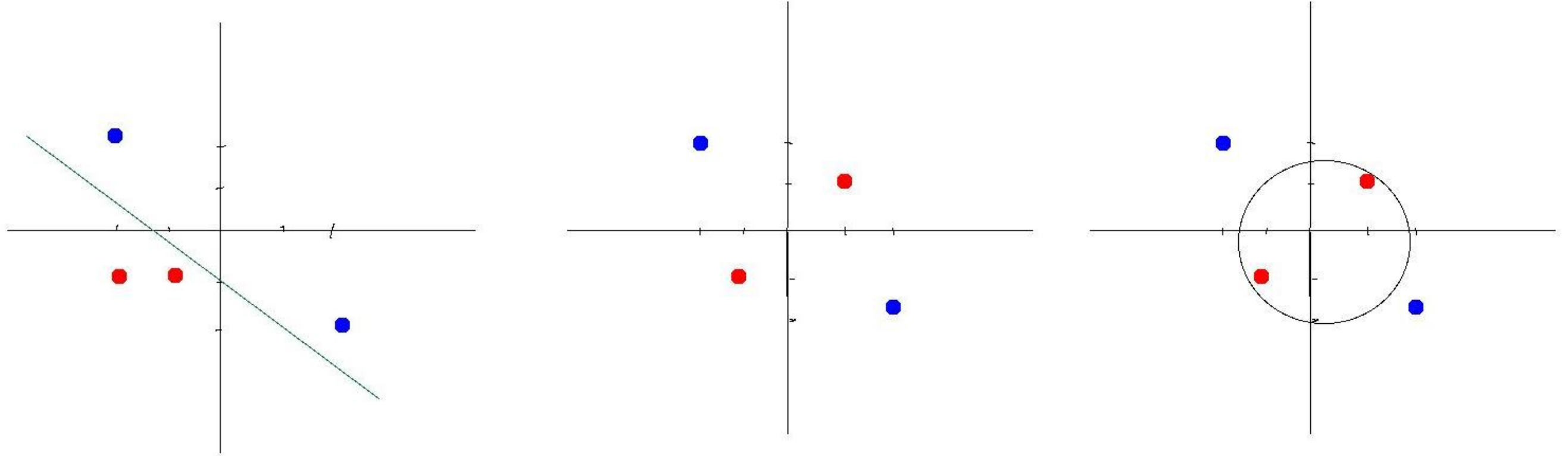


神经网络

在机器学习和认知科学领域，人工神经网络（ANN），简称神经网络（NN）是一种模仿生物神经网络（动物的中枢神经系统，特别是大脑）的结构和功能的数学模型或计算模型，用于对函数进行估计或近似。神经网络是**多层神经元的连接**，上一层神经元的输出，作为下一层神经元的输入。



线性不可分

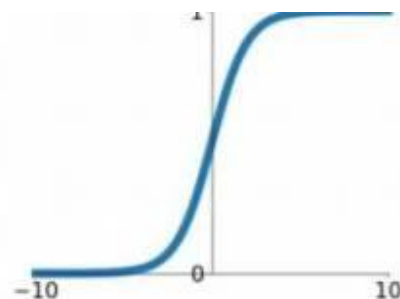


激活函数（Activation Function）

为了实现神经网络的非线性建模能力，解决一些线性不可分的问题，我们通常使用激活函数来引入非线性因素。激活函数都采用非线性函数，常用的有Sigmoid、tanh、ReLU等。

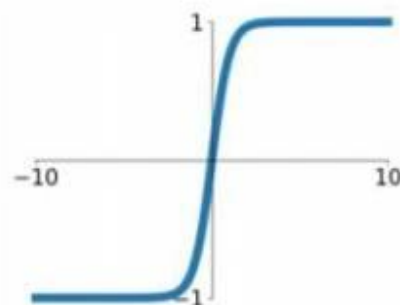
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



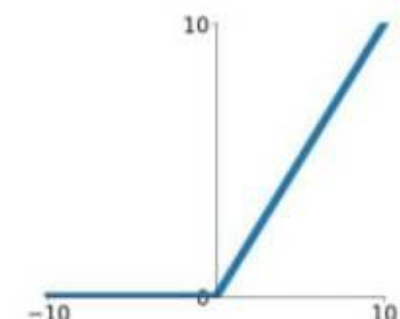
tanh

$$\tanh(x)$$



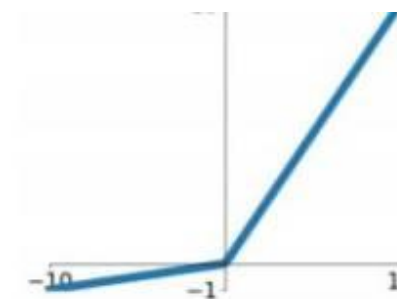
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

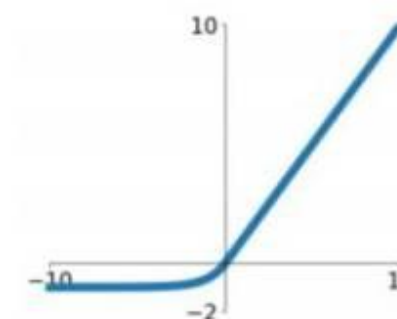


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

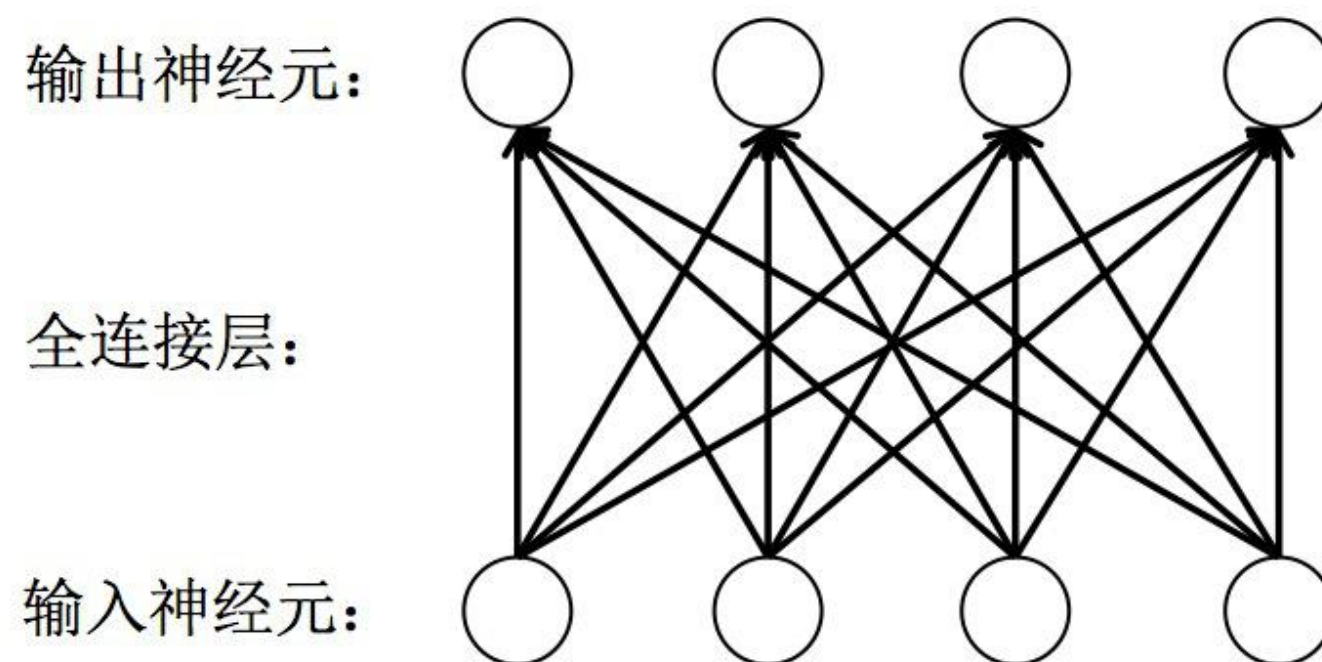
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



全连接层（fully connected layers, FC）

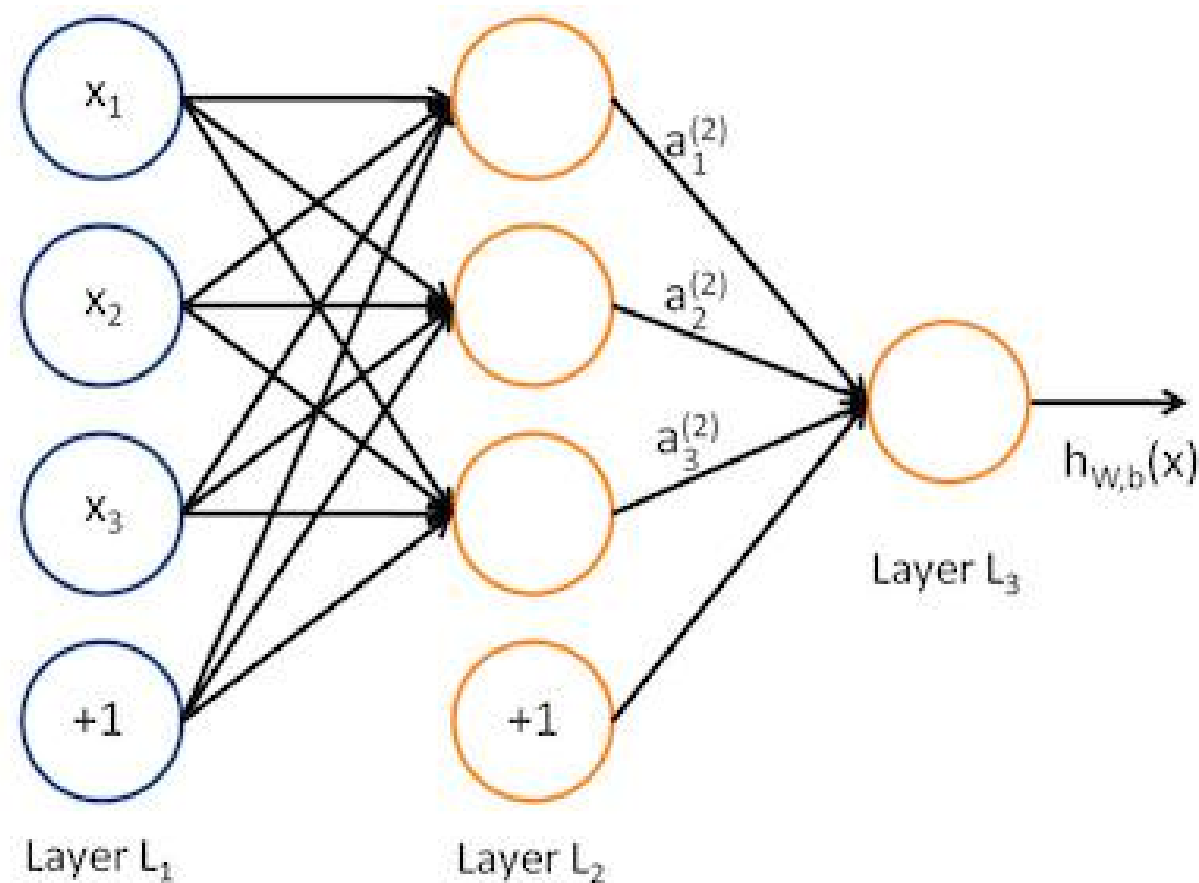
全连接层是一种对输入数据直接做线性变换的线性计算层。它是神经网络中最常用的一种层，用于学习输出数据和输入数据之间的变换关系。全连接层可作为特征提取层使用，在学习特征的同时实现特征融合；也可作为最终的分类层使用，其输出神经元的值代表了每个输出类别的概率。



前向传播

符号定义：

- L 为网络层数， w 和 b 为模型参数， X 为输入数据
- x_i : 第1层第 i 个神经元的输入
- $a_i^{(l)}$: 第 l 层第 i 个神经元的输出 (当 $l = 1$ 时, $a_i^{(1)} = x_i$)
- $w_{ij}^{(l)}$: 第 l 层第 j 个神经元到第 $l + 1$ 层第 i 个神经元的权重
- $b_i^{(l)}$: 第 l 层第 i 个神经元的偏置
- $h_{w,b}(X)$: 神经网络 (假设函数) 输出数据
- $(\vec{W}, \vec{B}) = (w^{(l)}, b^{(l)}), l = 1, \dots, L$



前向传播

3 层神经网络 计算过程：

$$a_1^{(2)} = f(w_{11}^{(1)}x_1 + w_{12}^{(1)}x_2 + w_{13}^{(1)}x_3 + b_1^{(1)})$$

$$a_2^{(2)} = f(w_{21}^{(1)}x_1 + w_{22}^{(1)}x_2 + w_{23}^{(1)}x_3 + b_2^{(1)})$$

$$a_3^{(2)} = f(w_{31}^{(1)}x_1 + w_{32}^{(1)}x_2 + w_{33}^{(1)}x_3 + b_3^{(1)})$$

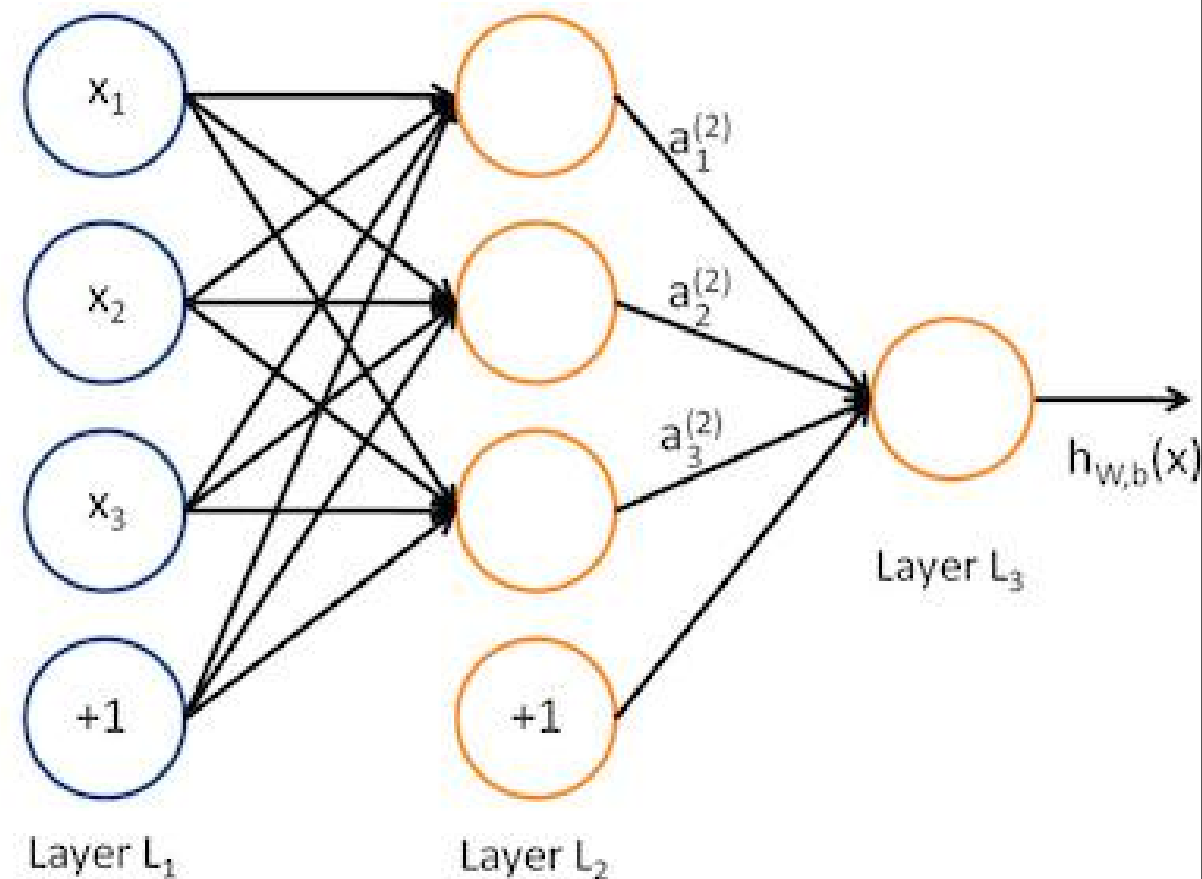
$$h_{w,b}(X) = a_3^{(3)} = f(w_{11}^{(2)}a_1^{(2)} + w_{12}^{(2)}a_2^{(2)} + w_{13}^{(2)}a_3^{(2)} + b_1^{(2)})$$

简化形式：

$$z_i^{(l)} = \sum_{j=1}^n w^{(l-1)} a^{(l-1)} + b^{(l-1)}$$

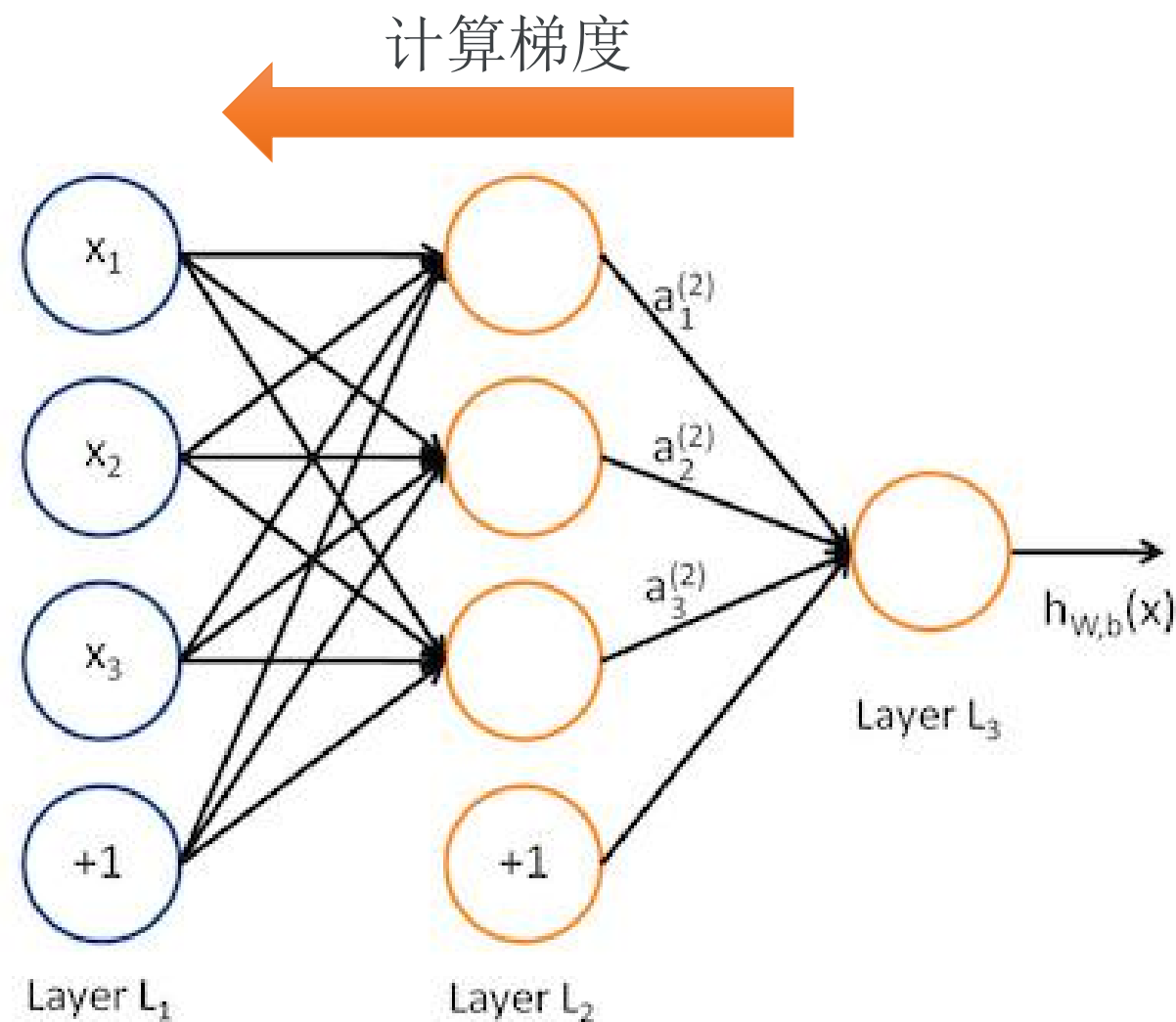
$$a_i^{(l)} = f(z_i^{(l)}), i = 1, \dots, n$$

$$h_{w,b}(X) = a_3^{(3)} = f(z_3^{(3)})$$



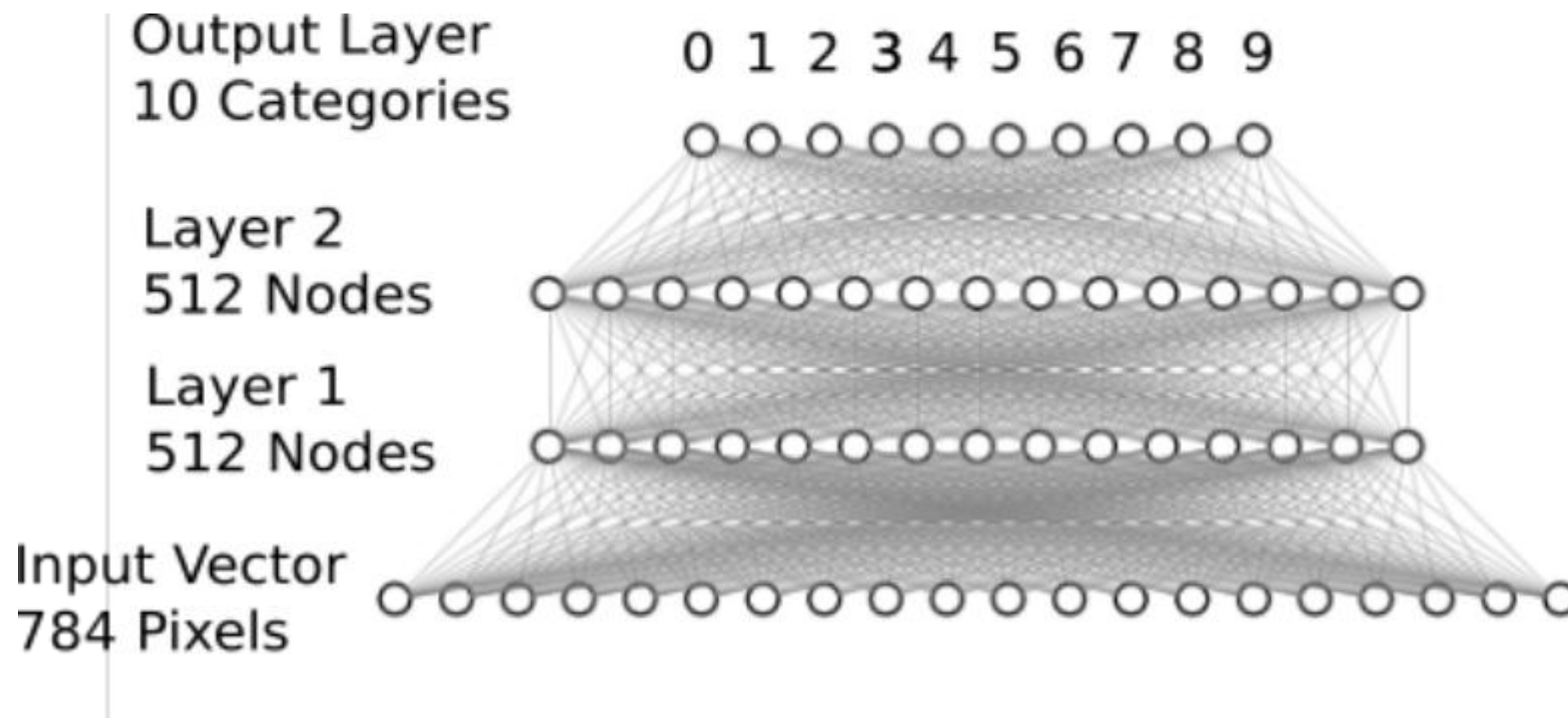
后向传播 (Back Propagation, BP)

BP算法的基本思想是通过损失函数对模型参数进行求导，并根据复合函数求导常用的“链式法则”将不同层的模型参数的梯度联系起来，使得计算所有模型参数的梯度更简单。BP算法的思想早在 1960s 就被提出来了。直到1986年，David Rumelhart 和 Geoffrey Hinton 等人发表了一篇后来成为经典的论文，清晰地描述了BP算法的框架，才使得BP算法真正流行起来，并带来了神经网络在80年代的辉煌。



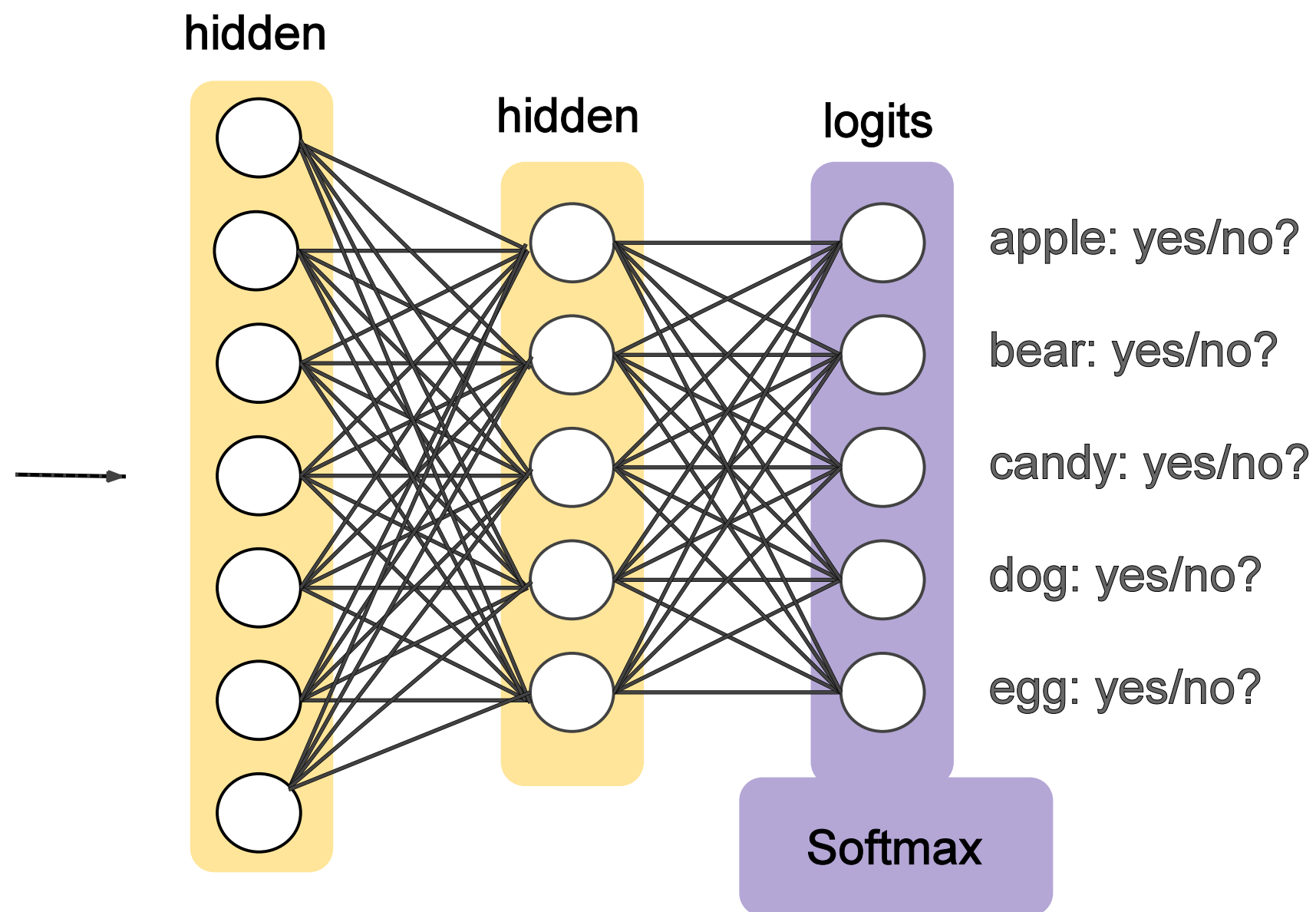
MNIST Softmax 网络

将表示手写体数字的形如 [784] 的一维向量作为输入；中间定义2层 512 个神经元的隐藏层，具备一定模型复杂度，足以识别手写体数字；最后定义1层10个神经元的全联接层，用于输出10个不同类别的“概率”。



实战 MNIST Softmax 网络

MNIST Softmax 网络层

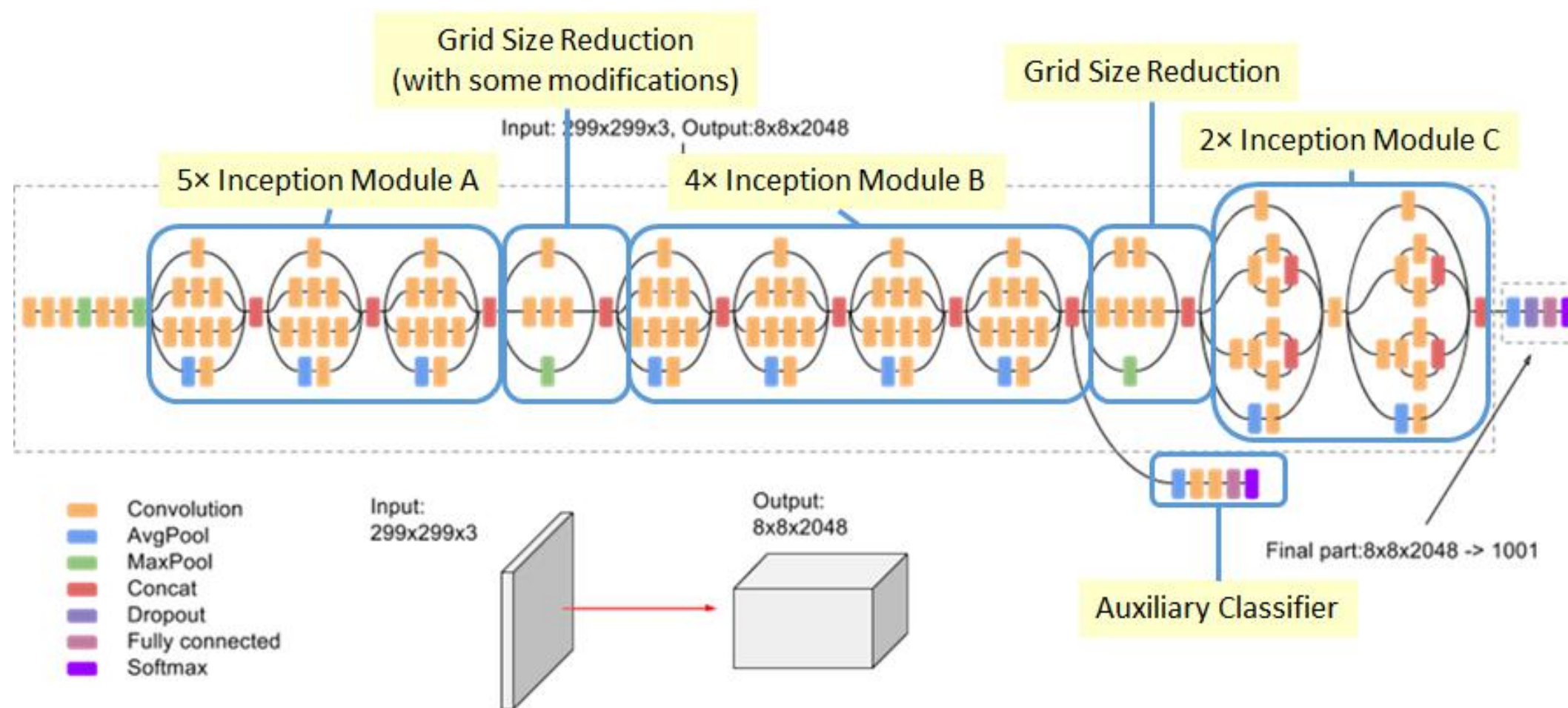


Try it

MNIST CNN 网络介绍

CNN 简介

CNN模型是一种以卷积为核心的前馈神经网络模型。20世纪60年代，Hubel和Wiesel在研究猫脑皮层中用于局部敏感和方向选择的神经元时发现其独特的网络结构可以有效地降低反馈神经网络的复杂性，继而提出了卷积神经网络（Convolutional Neural Networks，简称CNN）。



卷积 (Convolution)

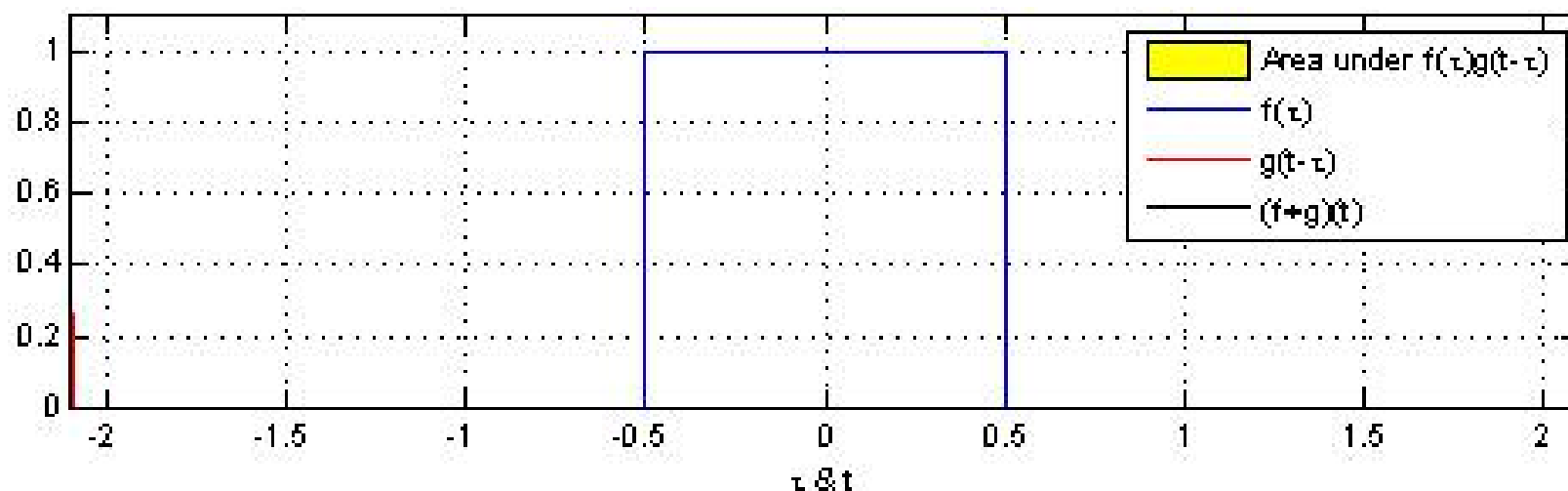
卷积是分析数学中的一种基础运算，其中对输入数据做运算时所用到的函数称为卷积核。

设: $f(x)$, $g(x)$ 是 \mathbb{R} 上的两个可积函数，作积分：

$$\int_{-\infty}^{\infty} f(\tau)g(x - \tau) d\tau$$

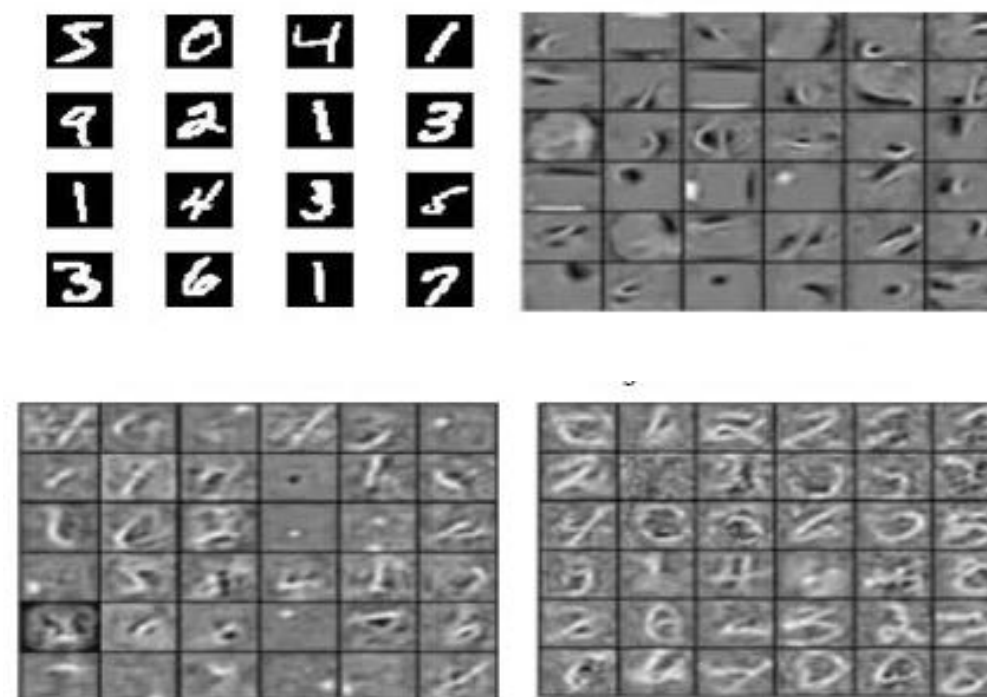
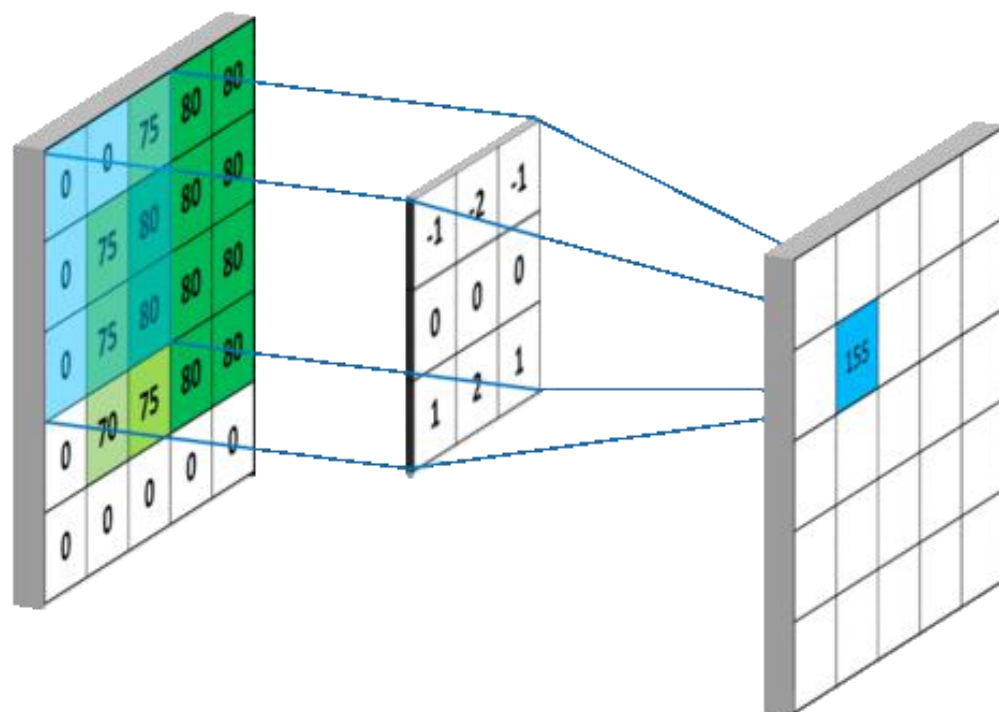
可以证明，关于几乎所有的实数 x ，上述积分是存在的。这样，随着 x 的不同取值，这个积分就定义了一个如下的新函数，称为函数 f 与 g 的卷积

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{\mathbb{R}^n} f(\tau)g(t - \tau) d\tau$$



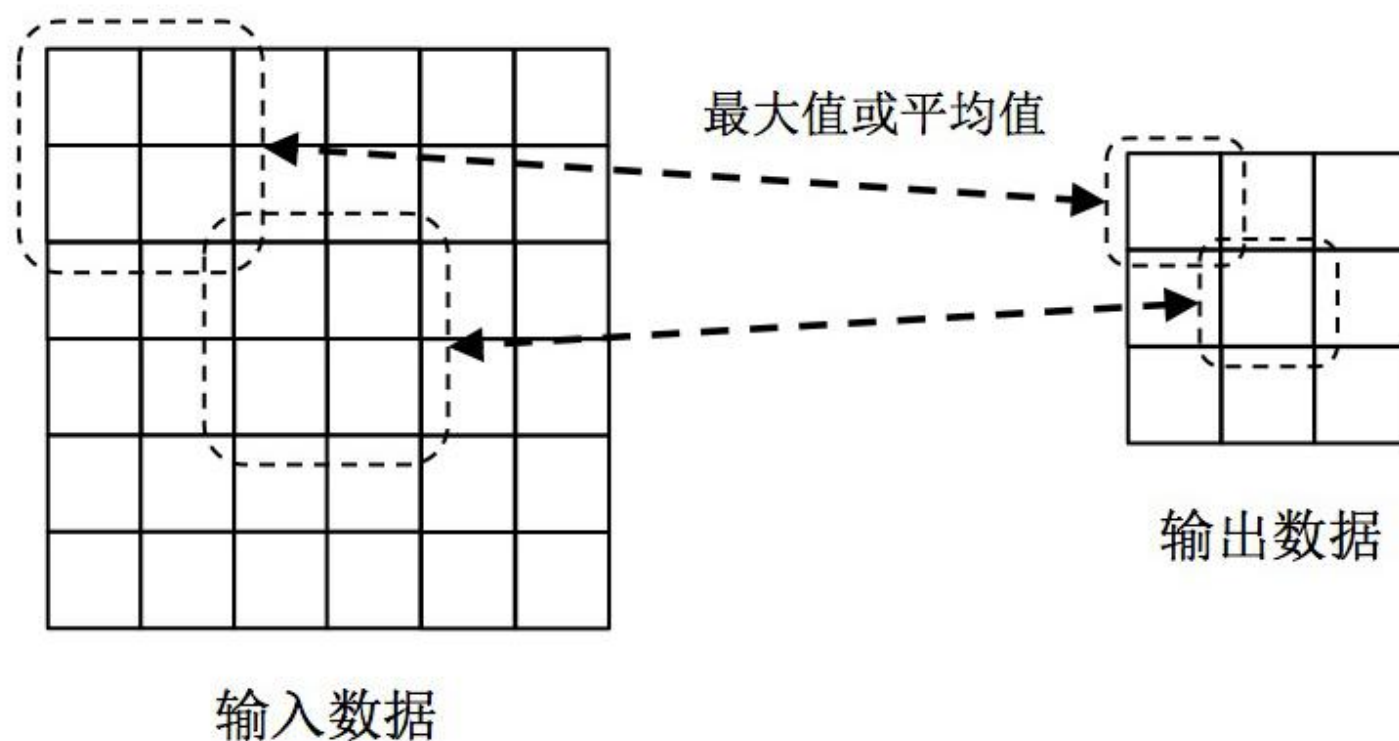
卷积层 (Convolutional Layer, conv)

卷积层是使用一系列卷积核与多通道输入数据做卷积的线性计算层。卷积层的提出是为了利用输入数据（如图像）中特征的局域性和位置无关性来降低整个模型的参数量。卷积运算过程与图像处理算法中常用的空间滤波是类似的。因此，卷积常常被通俗地理解作为一种“滤波”过程，卷积核与输入数据作用之后得到了“滤波”后的图像，从而提取出了图像的特征。



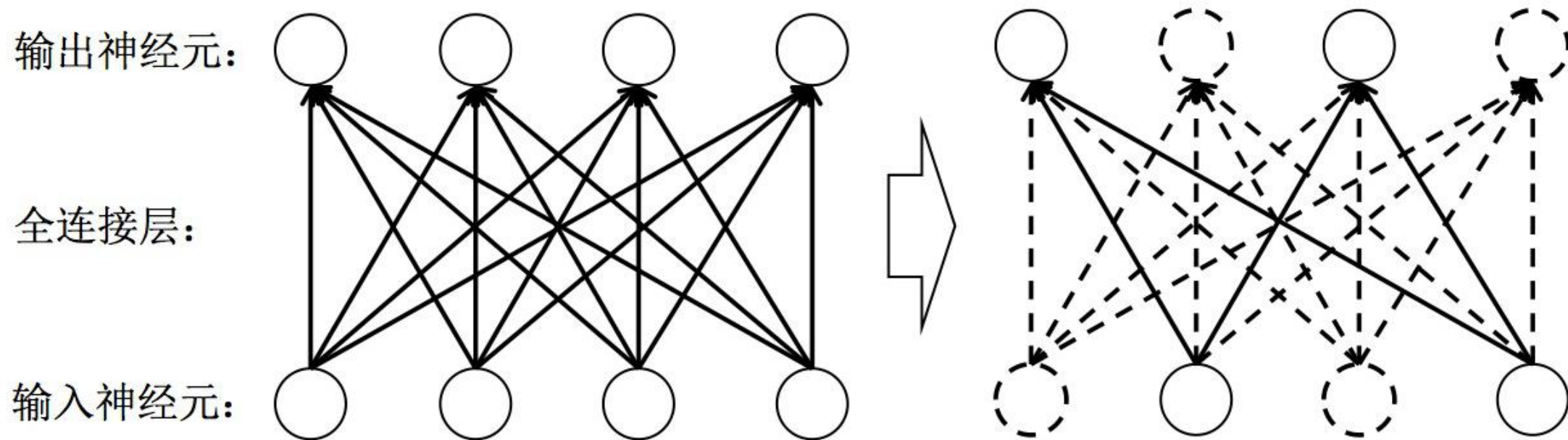
池化层 (Pooling)

池化层是用于缩小数据规模的一种非线性计算层。为了降低特征维度，我们需要对输入数据进行采样，具体做法是在一个或者多个卷积层后增加一个池化层。池化层由三个参数决定：（1）池化类型，一般有最大池化和平均池化两种；（2）池化核的大小 k ；（3）池化核的滑动间隔 s 。下图给出了一种的池化层示例。其中， 2×2 大小的池化窗口以2个单位距离在输入数据上滑动。在池化层中，如果采用最大池化类型，则输出为输入窗口内四个值的最大值；如采用平均池化类型，则输出为输入窗口内四个值的平均值



Dropout 层

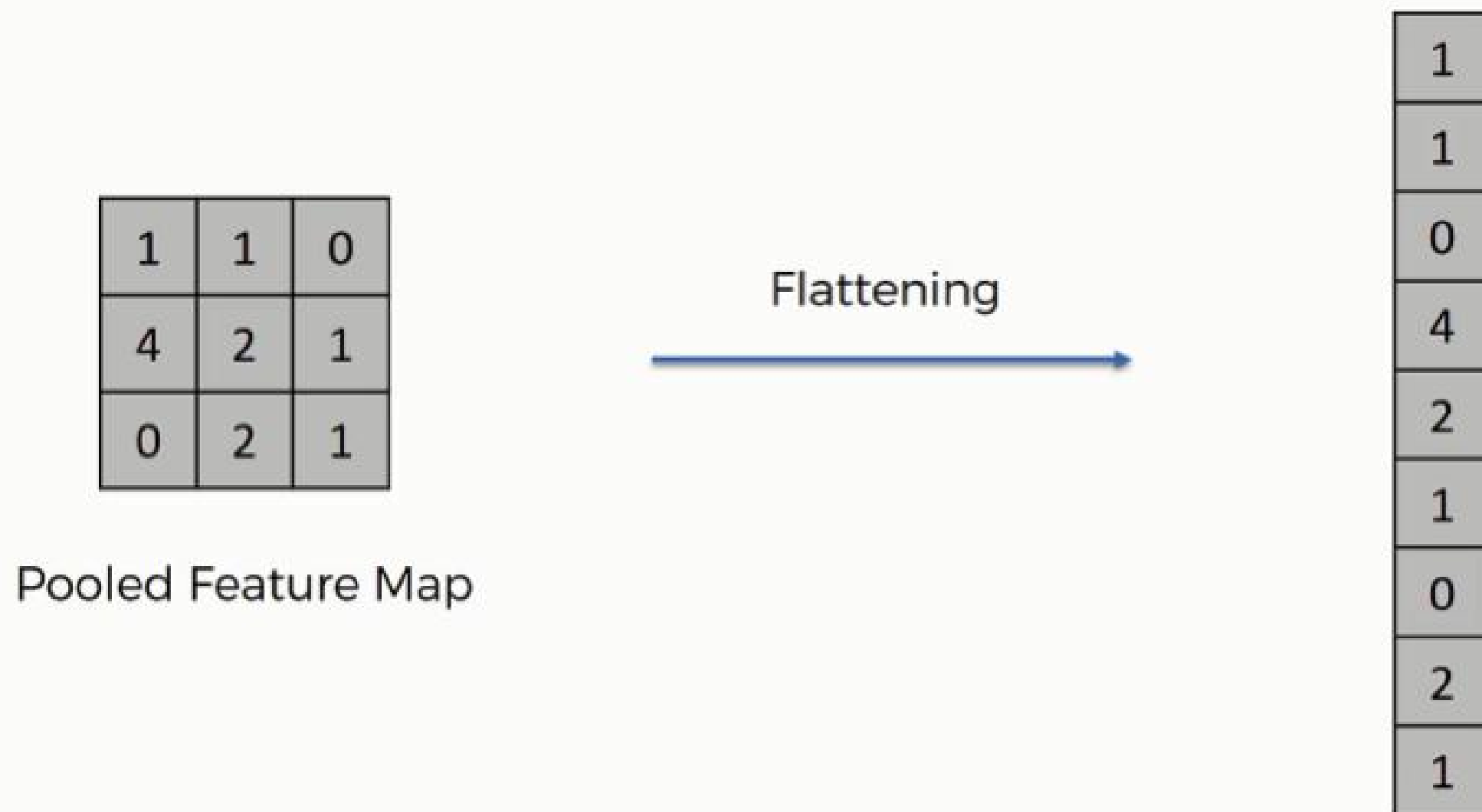
Dropout 是常用的一种正则化方法，Dropout层是一种正则化层。全连接层参数量非常庞大（占据了CNN模型参数量的80%~90%左右），发生过拟合问题的风险比较高，所以我们通常需要一些正则化方法训练带有全连接层的CNN模型。在每次迭代训练时，将神经元以一定的概率值暂时随机丢弃，即在当前迭代中不参与训练。



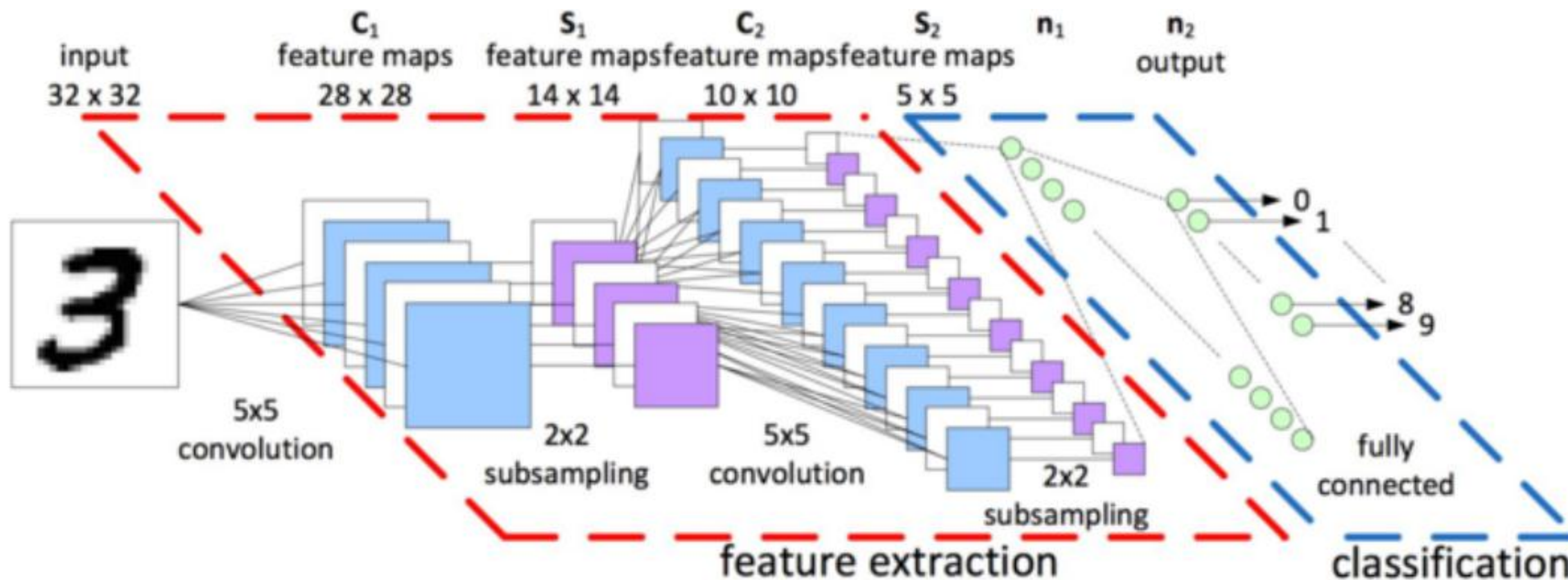
Flatten

将卷积和池化后提取的特征摊平后输入全连接网络，这里与 MNIST softmax 网络的输入层类似。

MNIST CNN 输入特征，MNIST Softmax 输入原图。



MNIST CNN 示意图



实战 MNIST CNN 网络

Try it



扫描二维码

试看/购买 《TensorFlow 快速入门与实战》 视频课程