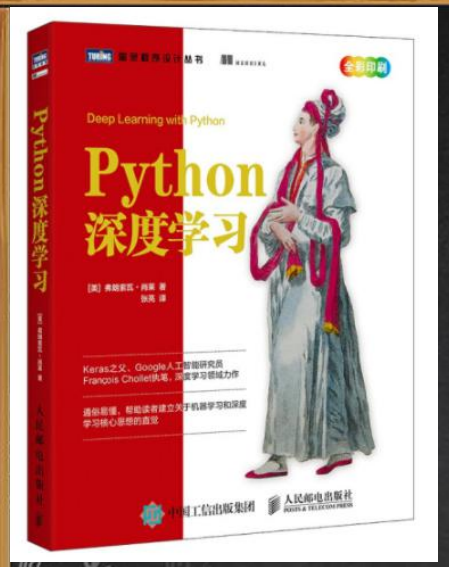


张量运算

欢迎关注B站：祥水村东头

祥水村东头@AI识堂

《Python 深度学习》读书笔记系列课程(对应2.3节 pp29-36)

本次胶片内容、及涉及相关代码均可移步至Github进行下载

我的代码 Github 地址：
https://github.com/david-cai/Reading-Note-for-Chollet-of-Deep-Learning-with-Python

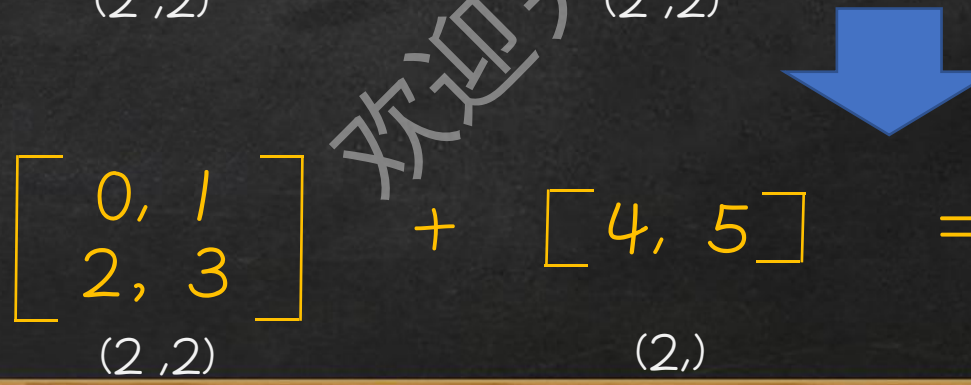
欢迎关注B站：祥永科东头

目录

欢迎关注B站：祥永村东头

1 逐元素运算（element-wise）

该运算独立地应用于张量中的每个元素，即不同张量的元素之间的点对点运算，非常适合大规模并行实现。



如果两个张量形状不一样，如何计算？

欢迎关注B站：祥求讨教

2 张量的广播

当两个形状（shape）不同的张量运算时，较小张量会沿着新增加的轴不断复制，使其维度（ndim）与较大形状张量保持一致。

$$\begin{bmatrix} 0, 1 \\ 2, 3 \end{bmatrix} + \begin{bmatrix} 4, 5 \\ 4, 5 \end{bmatrix} = \begin{bmatrix} 4, 6 \\ 6, 8 \end{bmatrix}$$

(2 ,2)　　(2,2)

-新增轴
-沿着新轴复制

$$\begin{bmatrix} 4, 5 \end{bmatrix}$$

(2,)

*特别注意：多维不同形状的张量可运算的前提条件
一个张量的形状为（a, b, c, d... k, l, m, n,）
另一个张量的形状为（n,）或（m, n）或（l, m, n,）.....
较小形状张量会沿着从a到m轴进行复制、从a到l轴、从a到k轴

欢迎关注B站：祥水计东头

3 乘积与点积（tensor product）

乘积

- 当两个张量shape一致时，两个张量逐元素相乘
- 当两个张量shape不一致时，按照广播的规则进行运算

点积/内积/数量积

- 前提：假如第一个张量包含n个数轴，且（n-1）轴包含x个元素，则第二个张量的第（n-2）轴上也应包含x个元素
- 形状(2,3）的张量可以与（3,5）的张量进行点积运算
- 形状（2,3,4）的张量可以与(5,4,6)的张量进行点积运算
- 元素先乘积后求和

欢迎关注B站：祥永的东头

3 乘积与点积（tensor product）

乘积

- 示例：

$$\begin{bmatrix} 0, & 1 \\ 2, & 3 \end{bmatrix} * \begin{bmatrix} 4, & 5 \\ 6, & 7 \end{bmatrix}$$

(2 ,2)　　(2 ,2)

相乘

$$0 \times 4 = \begin{bmatrix} 0, & 5 \\ 12, & 21 \end{bmatrix}$$

(2,2)

点积/内积/数量级

- 示例：

$$\begin{bmatrix} 0, & 1 \\ 2, & 3 \end{bmatrix} \text{ dot } \begin{bmatrix} 4, & 5 \\ 6, & 7 \end{bmatrix}$$

(2 ,2)　　(2 ,2)

先乘后和

$$0 \times 4 + 1 \times 6 = \begin{bmatrix} 6, & 7 \\ 26, & 31 \end{bmatrix}$$

(2,2)

欢迎关注B站：祥永村东头

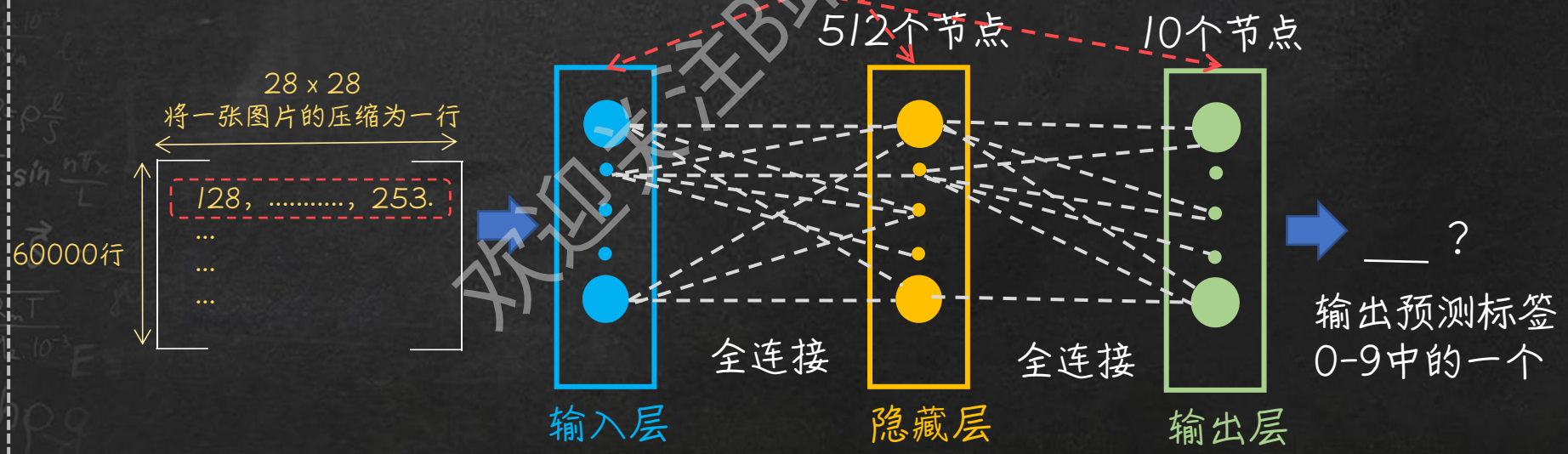
4 实现神经网络运算

回顾一下2.1节中所搭建的神经网络模型，张量是如何运算的？

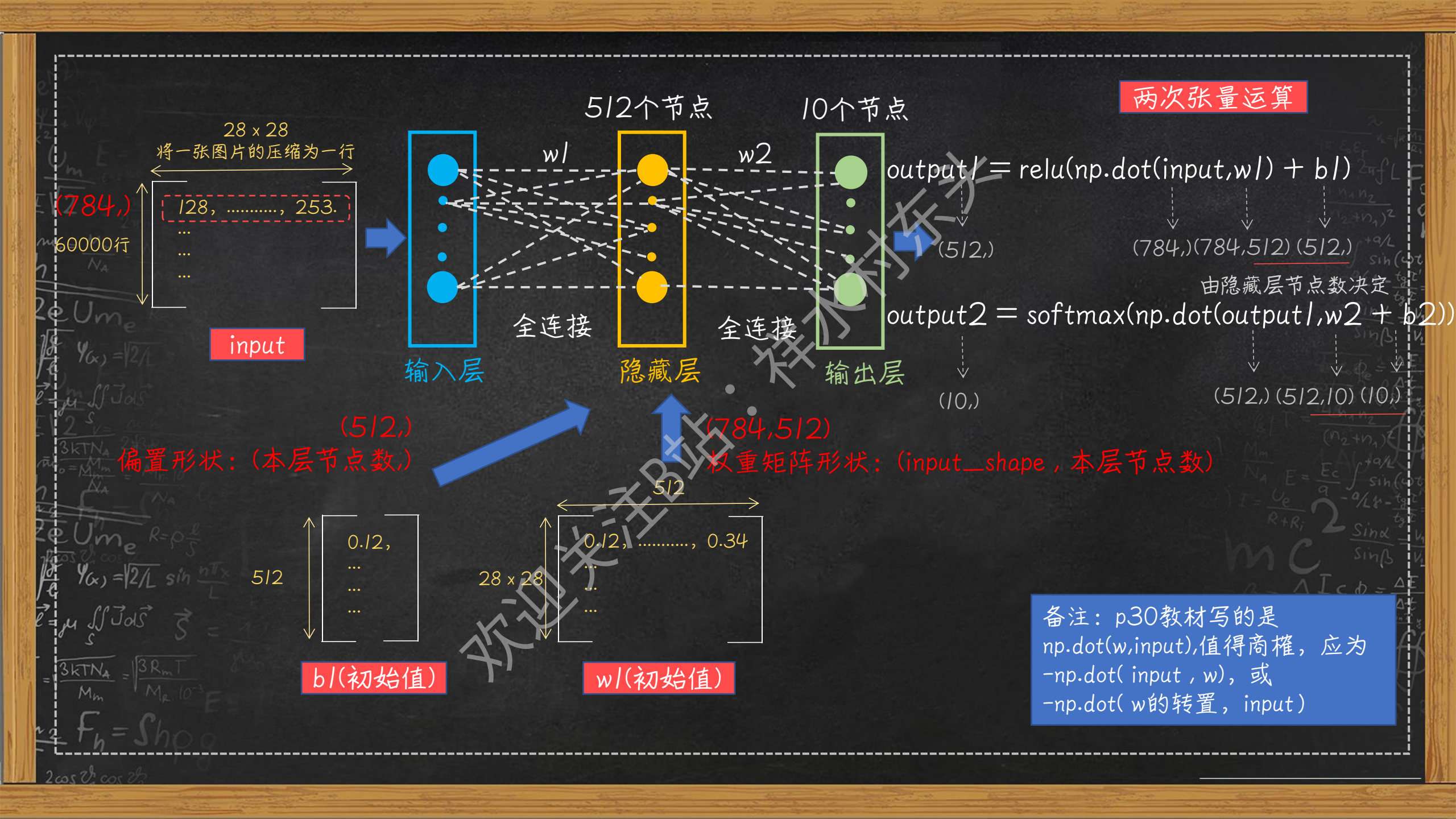
```
#初始化神经网络模型
network = models.Sequential()

#叠加首层——全连接层,首层还需要指定input_shape
network.add(layers.Dense(512,activation = 'relu', input_shape = (28 * 28 , )))

#叠加第二层——全连接层，注意是多分类问题，所以激活函数应使用softmax
network.add(layers.Dense(10, activation = 'softmax'))
```



欢迎关注B站：样长村东头


两次张量运算
28 x 28
将一张图片的压缩为一行
(784,)
60000行
128, ..., 253.
input
512个节点
10个节点
w1
w2
全连接
全连接
输入层
隐藏层
输出层
output1 = relu(np.dot(input,w1) + b1)
(784,)(784,512) (512,)
由隐藏层节点数决定
output2 = softmax(np.dot(output1,w2 + b2))
(512,) (512,10) (10,)
(512,)
(10,)
(784,512)
偏置形状：(本层节点数,)
权重矩阵形状：(input_shape，本层节点数)
512
0.12, ..., 0.34
28 x 28
512
0.12,
b1(初始值)
w1(初始值)
欢迎关注B站：祥永村东头
备注：p30教材写的是
np.dot(w,input),值得商榷，应为
-np.dot(input , w)，或
-np.dot(w的转置， input)

4 实现神经网络运算

先看输入层、第1个Dense层

```
#叠加首层——全连接层,首层还需要指定input_shape
network.add(layers.Dense(512,activation = 'relu', input_shape = (28 * 28 , )))
```

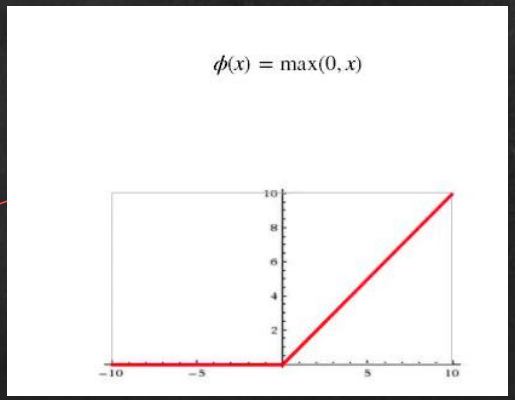
DIY内部运算实现

```
# 第1个Dense层数学运算表达式 output_1 = relu( np.dot(input * w1) + b1 )
#初始化输入，向量（784，）
inp= np.random.random((28 * 28, ))

#初始化权重参数，矩阵（784，512）
w1 = np.random.random((28*28 ， 512))

#偏置参数 ，向量(512，)
b1 = np.random.random((512， ))
print('权重参数张量的shape：' + str(b1.shape))

#应用relu激活函数，得到第1个Dense层输出
#输出为向量(512,)
outp1 = naive_relu(np.dot(inp,w1) + b1)
```



激活函数作用：将所有数转换为非负数

欢迎关注B站：祥永村东头

4 实现神经网络运算

再第2个Dense层

```
#叠加第二个dense层
network.add(layers.Dense(10,activation = 'softmax'))
```

DIY内部运算实现

```
# 第2个Dense层数学运算表达式，类同第1层，output_2 = softmax(np.dot(w2, output_1) + b2)
#初始化权重参数,矩阵（512,10）
w2 = np.random.random((512,10))

#偏置参数，向量（10,）
b2 = np.random.random((10, ))


.....
#应用tf自带的激活函数softmax，得到第2个Dense层输出
#输出为向量(10,)
outp2 = tf.nn.softmax(np.dot(outp1 , w2) + b2, axis=0)
```

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

激活函数作用：将所有数转换为概率值且概率和为1

欢迎关注B站：祥永科东头

张量形状（10，）
根据输出结果找到最大概率对应的标签类别

```
第2个dense层输出形状：Tensor("Softmax_2:0", shape=(10,), dtype=float64)
第2个dense层输出结果：[0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
```

```
Tensor("ArgMax_3:0", shape=(), dtype=int64)
预测数字为：9
```

模型预测标签类别为9

```
#选择概率最大的标签类别作为训练结果
outp_final = tf.argmax(outp3,axis = 0)
```

欢迎关注B站：祥水村东头

“人生之旅历途甚长，所争决不在一年半日，
万不可因此着急失望，招精神上之萎靡”

——梁启超致梁思成家书
献给正在逐梦路上努力奔跑的你我他

欢迎关注B站：行水航东头

感谢聆听

THANK YOU

本次胶片内容、及涉及相关代码均可移步至Github进行下载
感谢您的投币三连！

我的代码 Github 地址：
https://github.com/david-cai/Reading-Note-for-Chollet-of-Deep-Learning-with-Python

欢迎关注B站：祥永相伴头