《新一代人工智能：从深度学习到大模型》

卷积神经网络原理

主讲人 张重生

Part 04-1

Introduction to
Convolutional Neural Networks

卷积神经网络—卷积运算 示例讲解

通过示例，讲解卷积神经网络的卷积操作、原理及其特点

一、图像的卷积操作（1）二维图像/二维矩阵

在某个图像（矩阵）A上的每个滑动窗口W，与另一个矩阵B，进行点乘操作、并求和

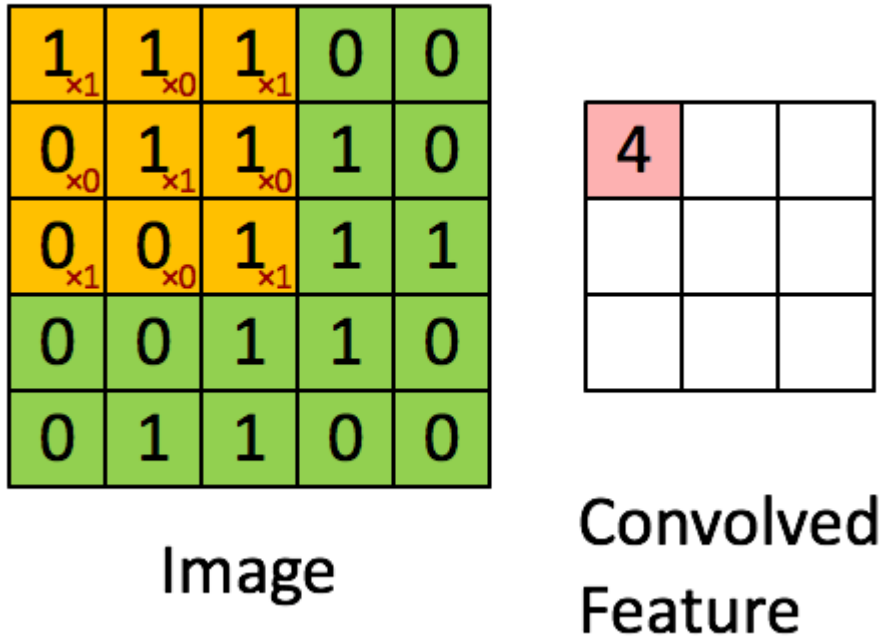
矩阵A

W

1	**1**	**1**	**0**	**0**
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

矩阵B

1	**0**	**1**
0	1	0
1	0	1



矩阵B称为二维卷积核，此处是3×3尺寸

1	**0**	**1**
0	1	0
1	0	1

矩阵B

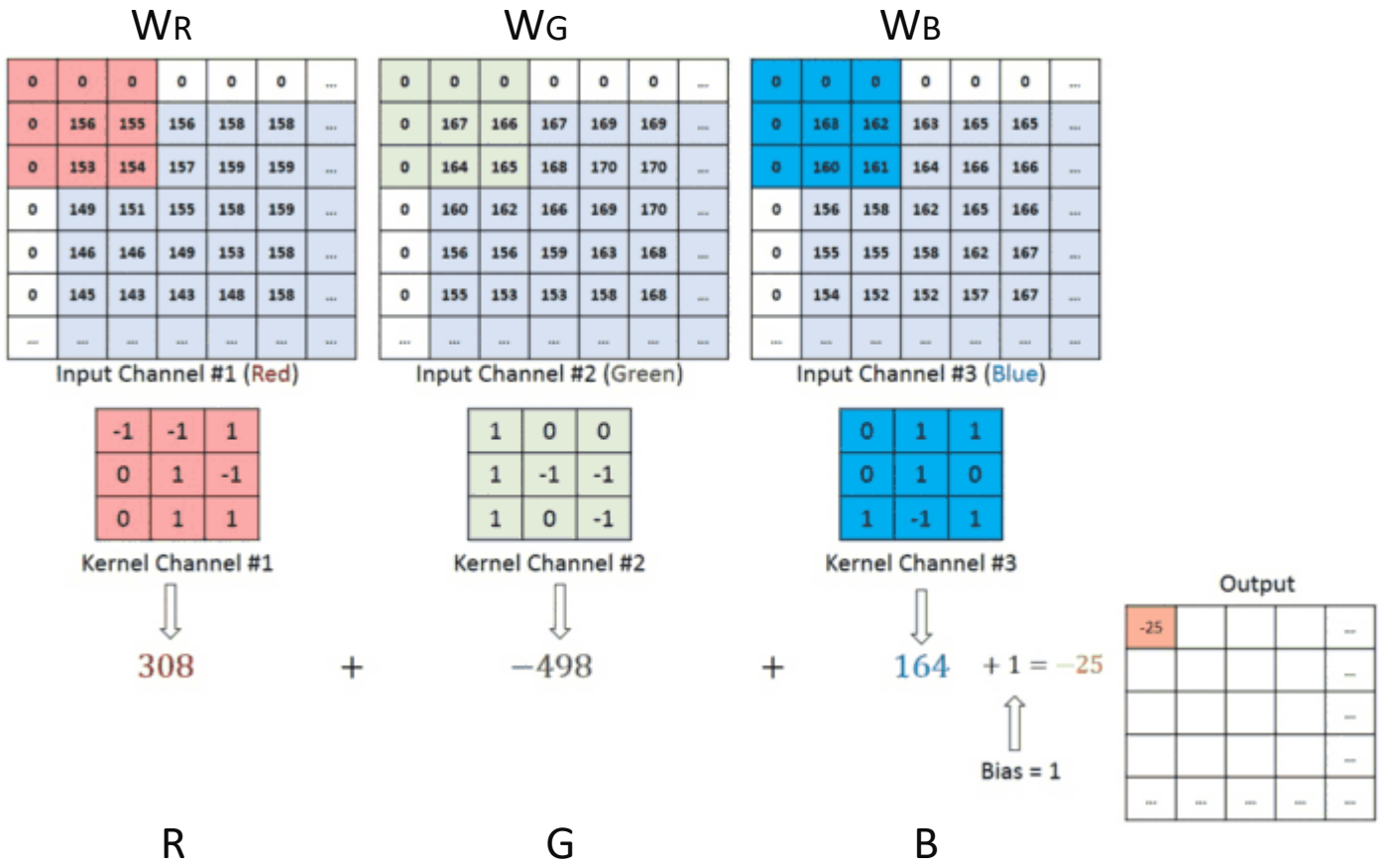
详见本书教材第51-52页

一、图像的卷积操作（2）三维图像/矩阵 三维卷积核

某个图像A为彩色图像，它包含了三个通道RGB三个矩阵； 与这三个通道依次对应有三个矩阵R G B

图像A上的每个滑动窗口W，对应三个通道上的矩阵，窗口W中的三个矩阵分别为W_R, W_G, W_B
将其分别与矩阵R、 G、 B 进行点乘操作、并求和

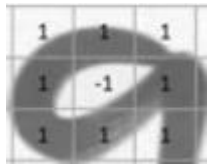
矩阵[R G B]合称为三维卷积核
此处是3×3×3

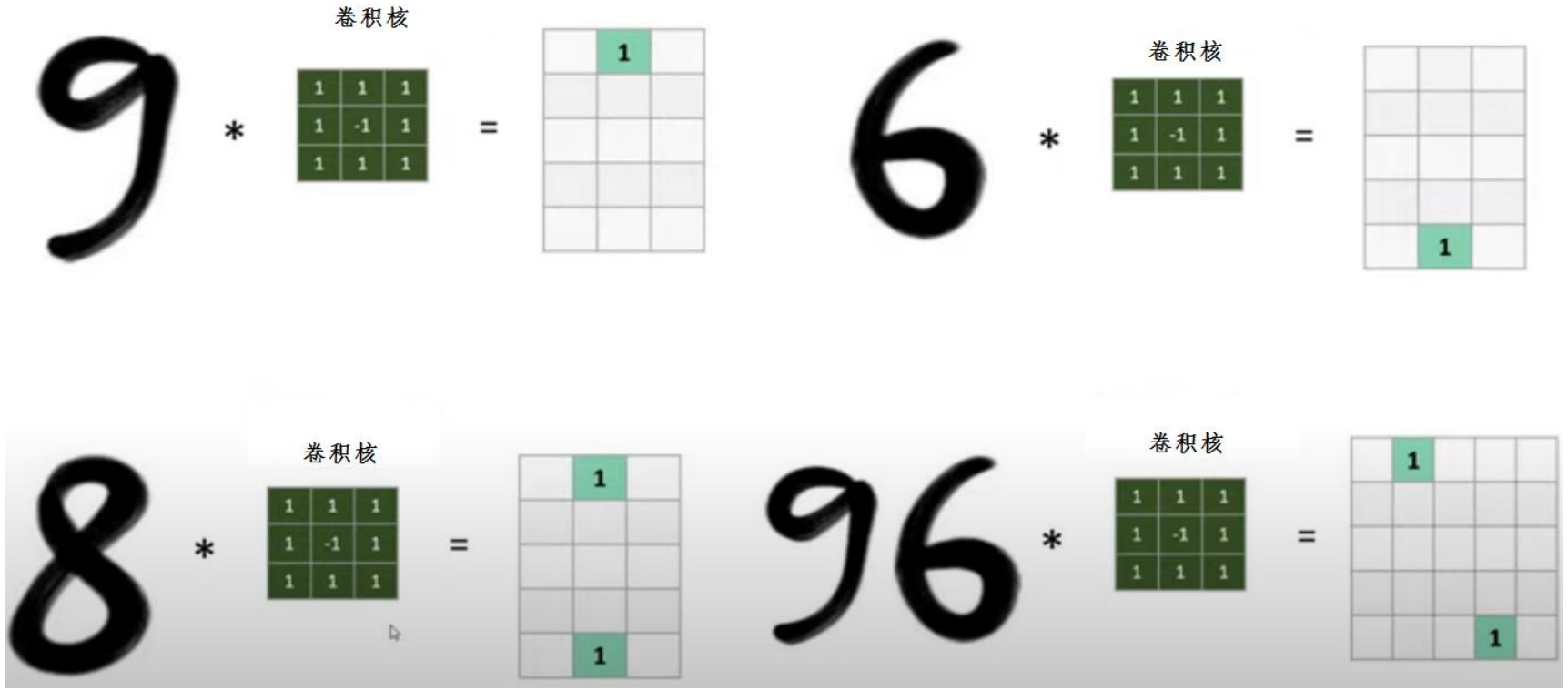


二、卷积核的物理意义（1）

左图中，9的笔画部分，在矩阵中的像素点的值都为1；否则为-1

问题： 一个图像(数字)， 是否包含9的上部笔画部分？

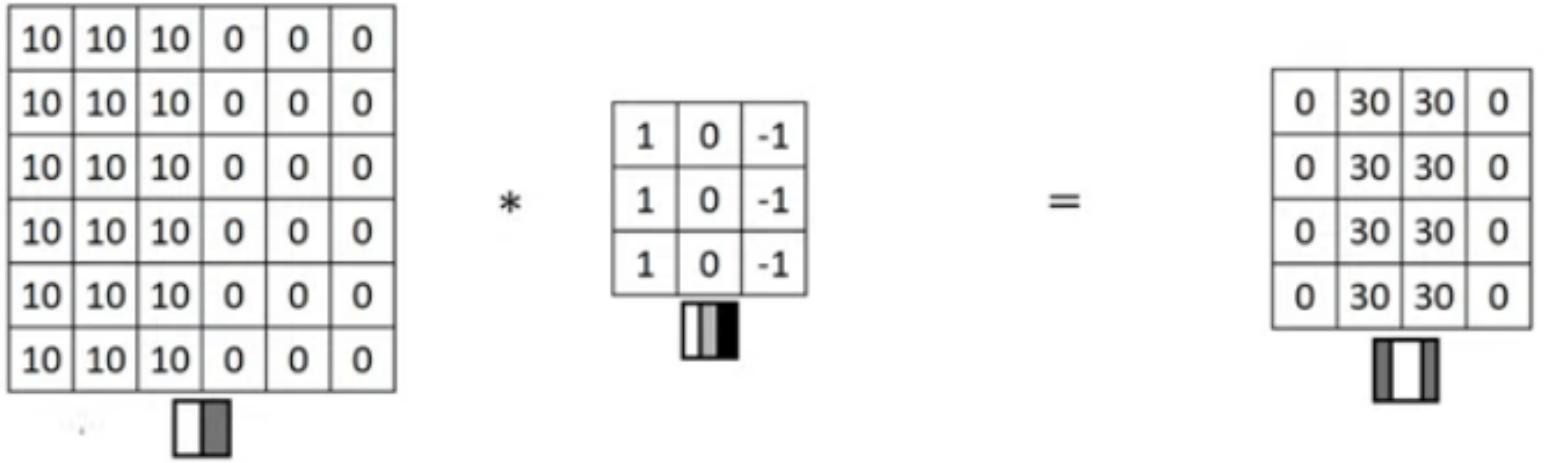




1	1	1
1	-1	1
1	1	1

卷积核相当于某种特征的矩阵表示
滤波器

二、卷积核的物理意义（2）



一个图像(矩阵)， 是否包含某种特征?

卷积核相当于某种特征的矩阵表示，如边缘特征滤波器

Sobel卷积核(Y)

1	**2**	**1**
0	0	0
-1	-2	-1

高斯滤波器

$$\frac{1}{2\pi\sigma * \sigma} e^{-\frac{x*x+y*y}{2\sigma*\sigma}}$$

Scharr卷积核(Y)

-3	**-10**	**-3**
0	0	0
3	10	3

-1	**0**	**1**
-2	0	2
-1	0	1

Sobel卷积核(X)
索伯算子

1	**1**	**1**
1	-8	1
1	1	1

Laplacian卷积核
拉普拉斯算子

-3	**0**	**3**
-10	0	10
-1	0	3

Scharr卷积核(X)
沙尔算子

二、卷积核的物理意义（3）

边缘特征1

-1	-1	5
-1	-1	5
-1	-1	5

边缘特征2

5	5	5
-1	-1	-1
-1	-1	-1

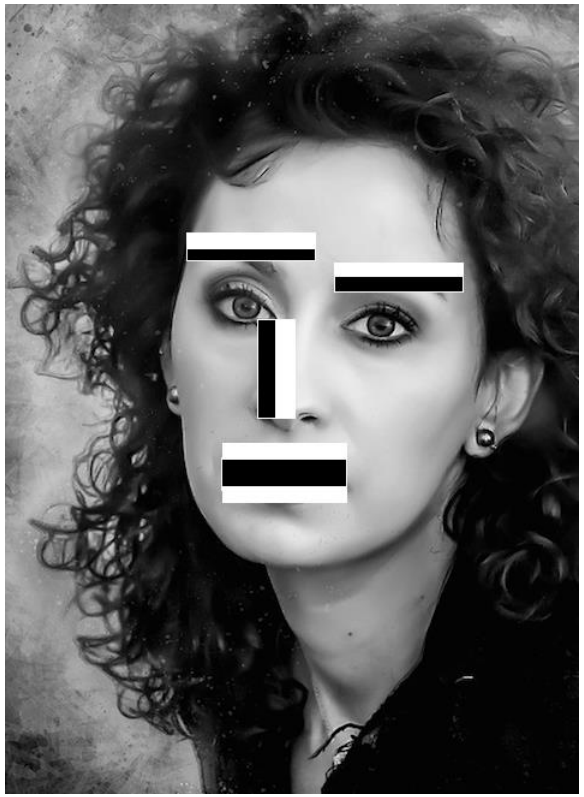
直线特征

-1	5	-1
-1	5	-1
-1	5	-1

斜线特征

5	-1	-1
-1	5	-1
-1	-1	5

Harr特征/卷积核



人脸检测

滑动窗口，每个窗口应用Harr卷积核

参考文献:
Paul A. Viola, Michael J. Jones: Rapid Object Detection using a Boosted Cascade of Simple Features. CVPR (1) 2001: 511-518

著名的Voila&Jones人脸检测算法，基于Harr特征/卷积核 + AdaBoost算法级联分类

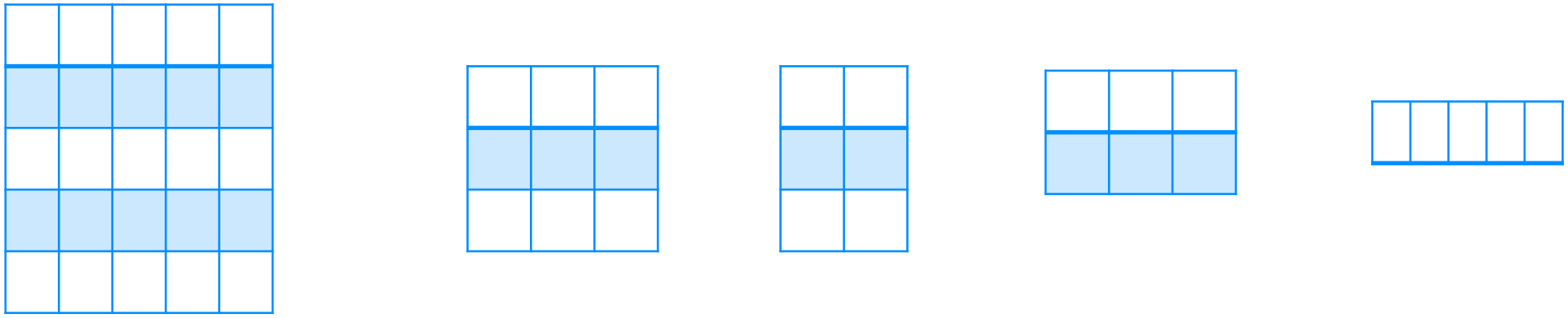
三、卷积神经网络中卷积核的特点

卷积神经网络中卷积核的特点：

（1）**卷积核中的数值是深度学习自动学习**的，无须用户事先指定
通过误差反向传播，进行卷积核中的值的自动更新

（2）**用户仍需指定卷积的尺寸（大小）和数量**

（3）同一个/同一层矩阵中的元素/神经元，共享卷积核中的权值



四、卷积神经网络与人工神经网络的区别与联系（1）

（1）卷积神经网络的各卷积层中的神经元，都不是全连接，而是局部连接
卷积神经网络，只有全连接层(FC)，神经元之间才是全连接

而人工神经网络中的相邻两层的神经元，都是全连接的

（2）**卷积神经网络中，同一层的神经元是共享卷积核的，即共享权值**

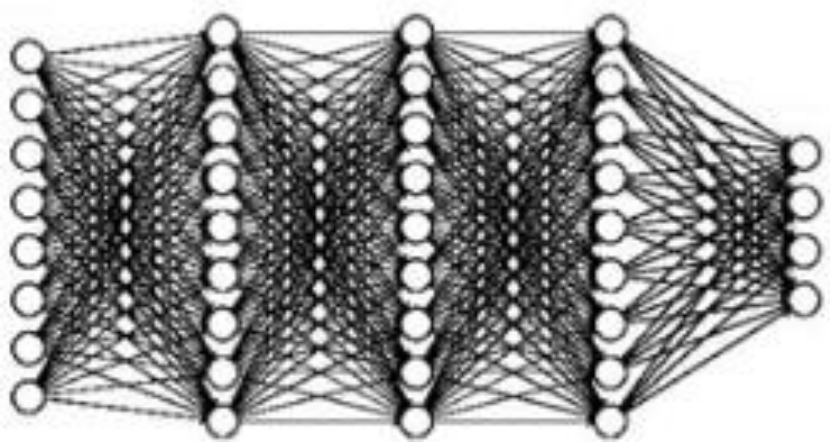
而人工神经网络中相邻两层的神经元，全部相连接，每个连接均有权值

（3）由于人工神经网络的相邻层的神经元是全连接的，
对于图像特征，如1024*1024的图像，第一层有100万个神经元（像素点）
带来了巨大的参数和运算量，卷积神经网络共享卷积核大大缓解该问题

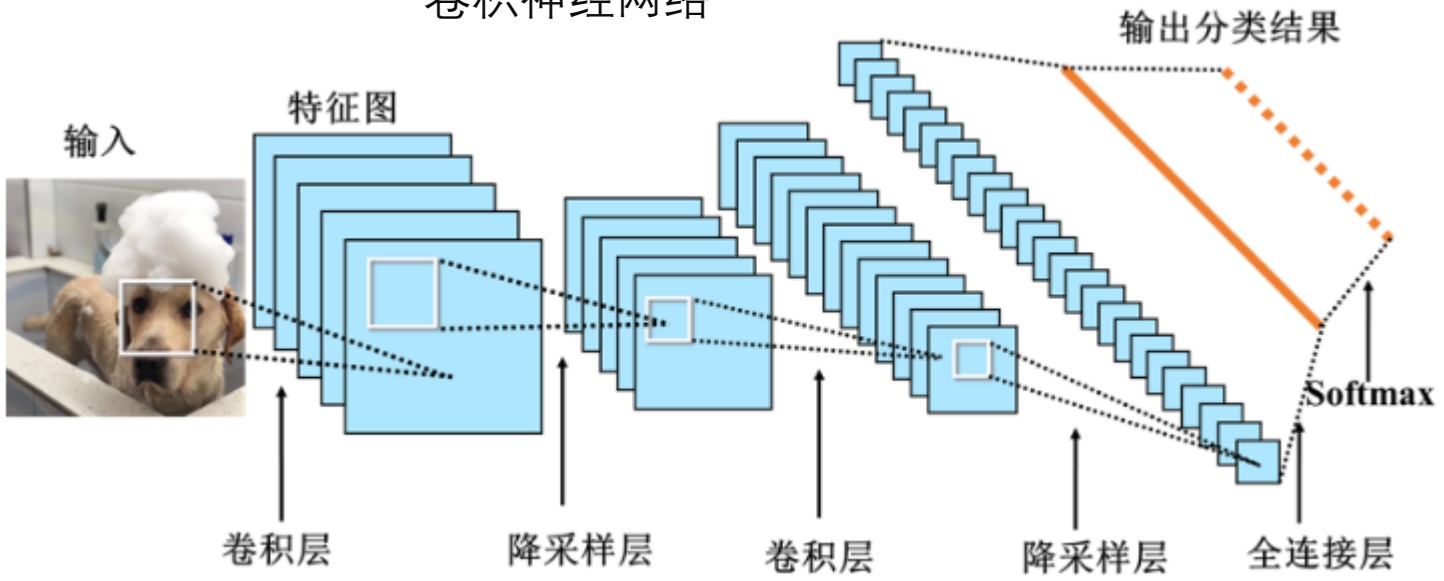
（4）人工神经网络，无法直接体现、利用神经元之间的空间邻域特征和信息

四、卷积神经网络与人工神经网络的区别与联系（2）

全连接神经网络



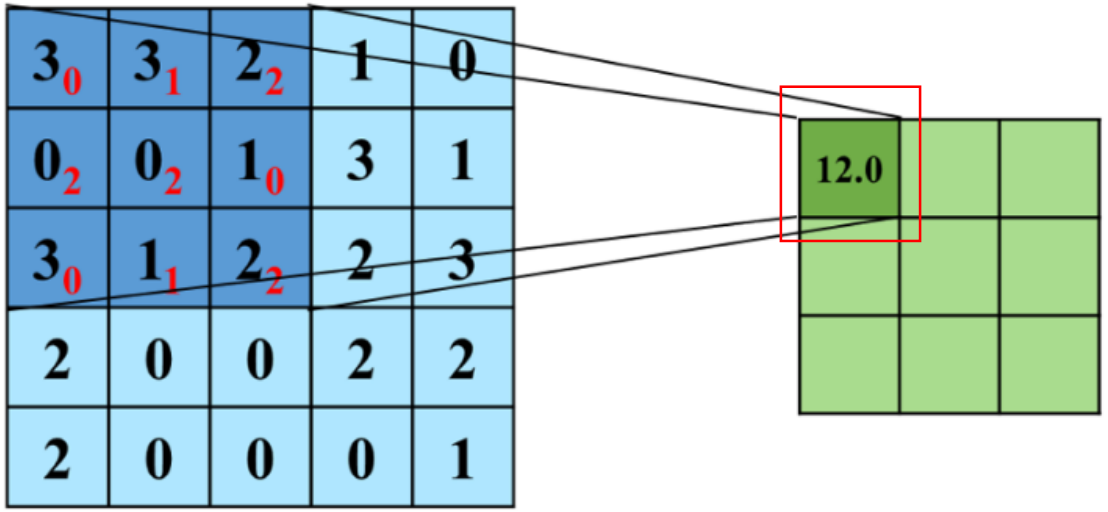
卷积神经网络



卷积神经网络中，
下一层某个神经元的值只与上一层的
某个子空间域（窗口）中的神经元连接

因此是部分连接，而不是全连接

尤其是图像是1024*1024时，
上一层神经元数量是百万级，而卷积操作中
下一层的某个神经元只与其中的9个神经元连接
而在人工神经网络中与上一层所有神经元连接



卷积操作
相当于
基于滤波器
的图像操作

Part 04-2

The Principles of
Convolutional Neural Networks

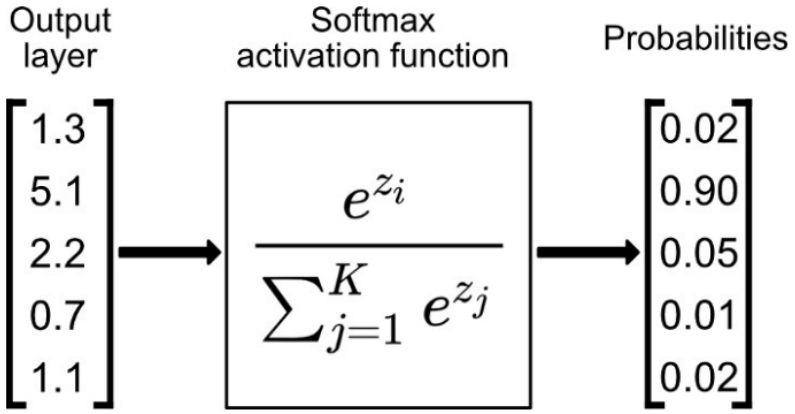
卷积神经网络—原理及示例

ReLU激活函数，卷积神经网络结构，Softmax激活函数，交叉熵损失

0 准备知识：Softmax激活函数和ReLU激活函数

Softmax激活函数 = $\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$

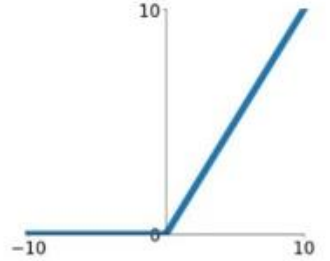
令原始值z_i：1，　　2，　　4
则激活后：　0.04201　　0.114195　　0.84379



ReLU激活函数

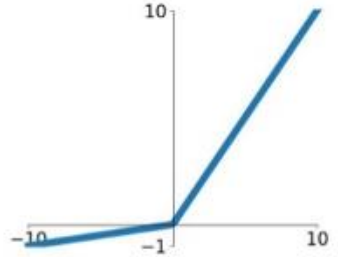
ReLU

$\max(0, x)$



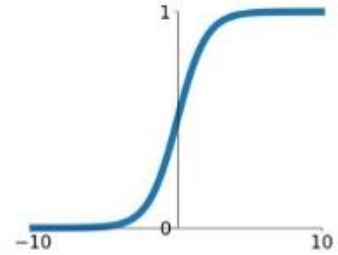
Leaky ReLU

$\max(0.1x, x)$



Sigmoid

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



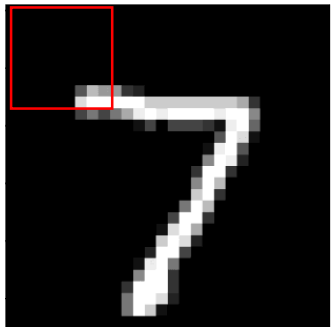
一、卷积神经网络计算过程（1）一个卷积核

⊗ σ μ

卷积神经网络顺序迭代地执行如下步骤：卷积操作→激活操作→降采样操作……卷积操作→激活操作→降采样操作

Conv1 ReLU Maxpool Conv2 ReLU Maxpool

3×3滑动窗口 3×3卷积核 卷积操作 图像26×26像素 激活操作 降采样 13×13 卷积操作 激活操作 降采样



1	**1**	**1**
1	-8	1
1	1	1

0	**0**	**0**	**0**	**0**	…		
0	0	0	0	0	…		
…							
0	0	1	1	1	1	…	
0	0	0	0	0	1		
					1		
					1		
					…		

σ

0	**0**	**0**	**0**	**0**	**0**	**0**
0	0	0	0	0	0	0
0	1	1	1	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	0	0	0

新卷积核

⊗ σ μ

图像28×28像素

步幅 Stride = 1, 2, …

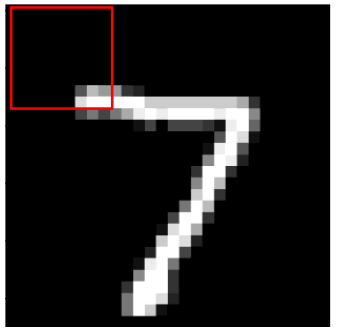
一、卷积神经网络计算过程（2）多个卷积核

⊗ σ μ池化

卷积神经网络顺序迭代地执行如下步骤： 卷积操作→激活操作→降采样操作……卷积操作→激活操作→降采样操作

Conv1 三个新矩阵 ReLU Maxpool Conv2 ReLU Maxpool

3×3滑动窗口 3×3卷积核 卷积操作 图像26×26像素 激活操作 降采样 卷积操作 激活操作 降采样



图像28×28像素

1	**1**	**1**
1	-8	1
1	1	1

-1	**0**	**1**
-2	0	2
-1	0	1

1	**2**	**1**
0	0	0
-1	-2	-1

三个卷积核

0	**0**	**0**	**0**	**0**	…		
0	0	0	0	0	…		
…							
0	0	1	1	1	1	…	
0	0	0	0	0	1		
					1		
					1		
					…		

…

三个卷积后的新矩阵 26×26

σ

0	**0**	**0**	**0**	**0**	**0**	**0**
0	0	0	0	0	0	0
0	1	1	1	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	1	0	0
0	0	0	0	0	0	0

三个激活后的矩阵 26×26

新卷积核若干个

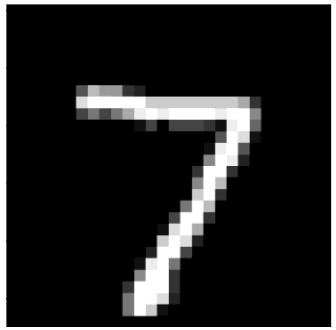
⊗ σ μ

三个降采样后的矩阵 13×13

二、一个简单的卷积神经网络示例（1）

① ② ③ ④ ⑤

28×28输入图像



3×3卷积核

1	**1**	**1**
1	-8	1
1	1	1

-1	**0**	**1**
-2	0	2
-1	0	1

1	**2**	**1**
0	0	0
-1	-2	-1

Conv_1
三个26×26矩阵

ReLU

Maxpool_1
下采样 三个13×13矩阵

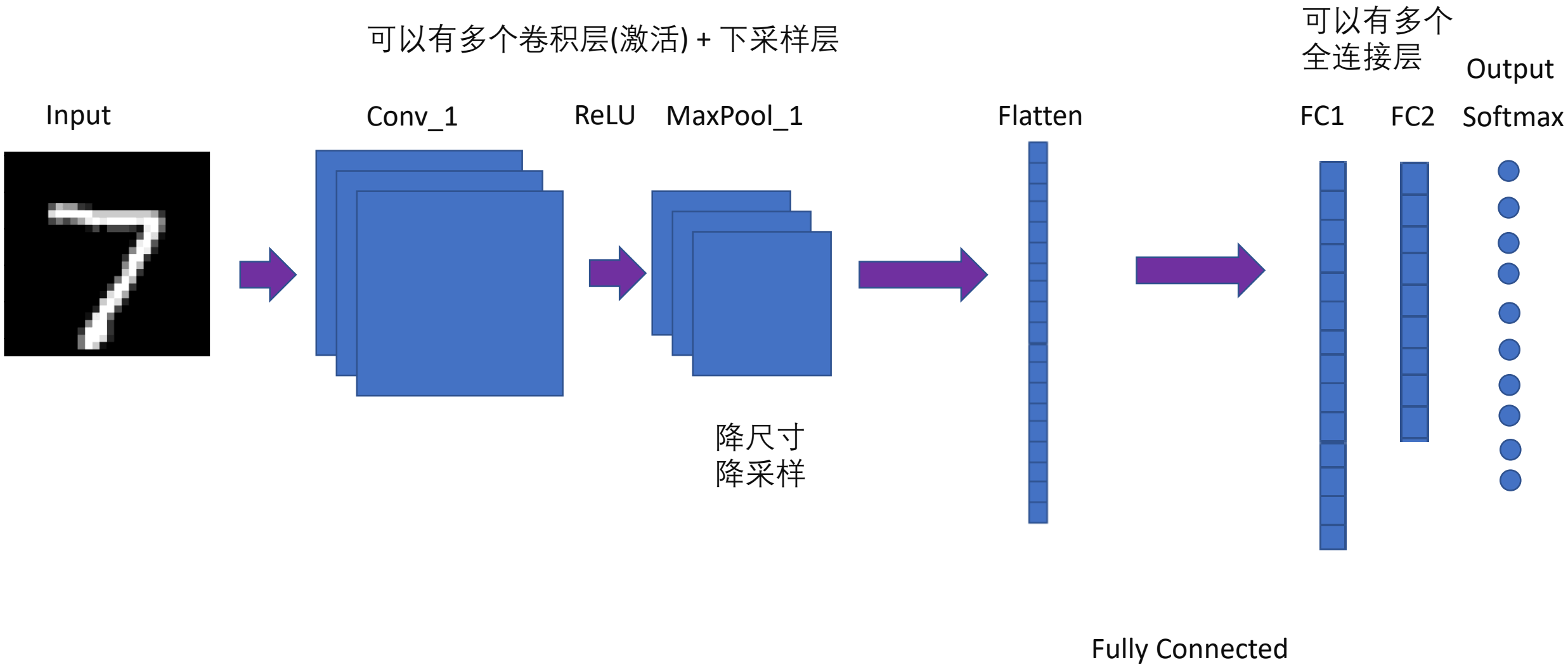
Flatten拉平

Output输出层
Softmax激活
预测属于0-9的概率

稀疏连接(部分连接)

全连接

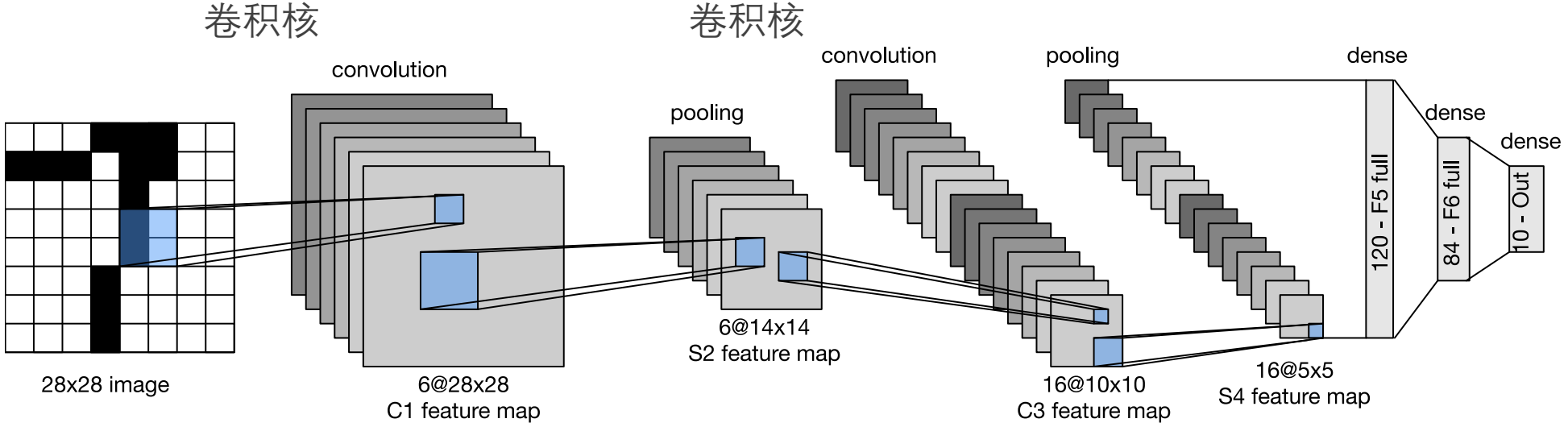
二、一个简单的卷积神经网络示例（2）

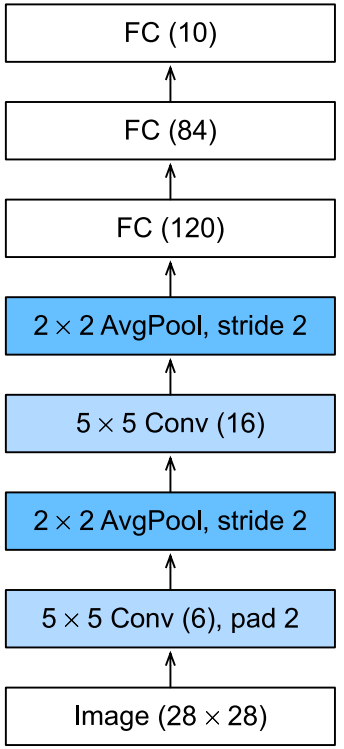


二、一个简单的卷积神经网络示例（3）

6个5×5
卷积核

16个5×5 （*现当作5×5 ×6的多维卷积核*）
卷积核





LeNet神经网络

LeNet (1989):
两个卷积层，两个池化层，两个全连接层，一个输出层
AvgPool，Sigmoid激活函数

两个卷积层都是5×5窗口
两个全连接层的神经元数量分别为120和84

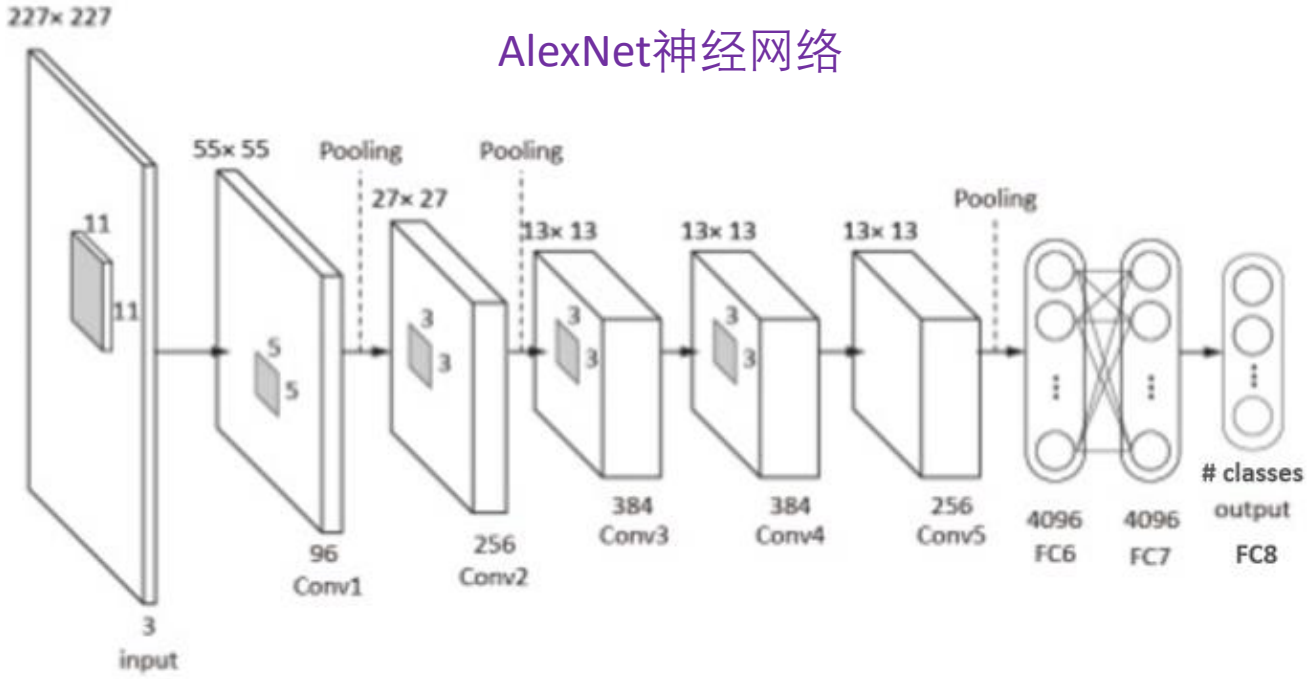
Valid: $w_o = \text{ceil}(\frac{|W|-w+1}{stride})$ 无padding

Same: $w_o = \text{ceil}(\frac{|W|}{stride})$ 有padding

详见本书教材第58-60页

二、一个简单的卷积神经网络示例（4）

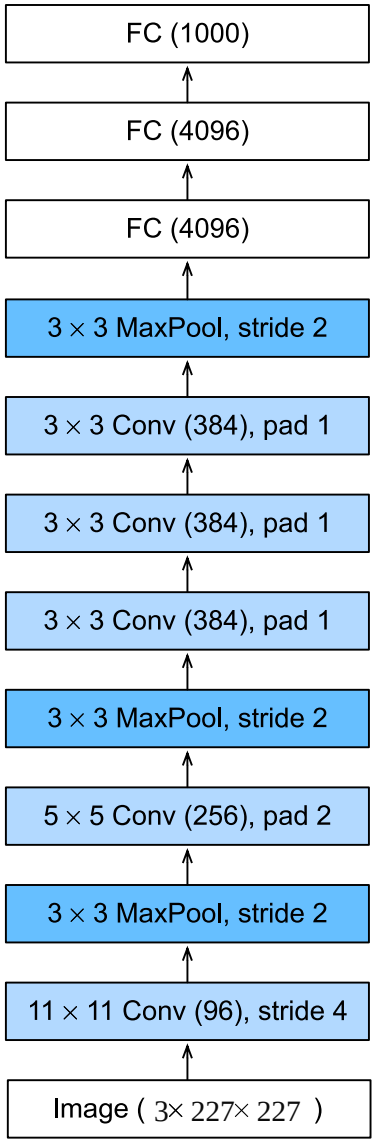
AlexNet神经网络



AlexNet (2012年ImageNet竞赛冠军)：
五个卷积层，三个池化层，两个全连接层，一个输出层
MaxPool，ReLU激活函数

第一层卷积11×11窗口，以后5×5， 3×3
两个全连接层的神经元数量均为4096

两个全连接层之间使用了Dropout技术，随机一半隐层节点值为0
(因为当模型的参数太多，而训练样本太少时，易产生过拟合)



Valid:

$w_o = \text{ceil}\left(\frac{|W| - w + 1}{stride}\right)$

Same:

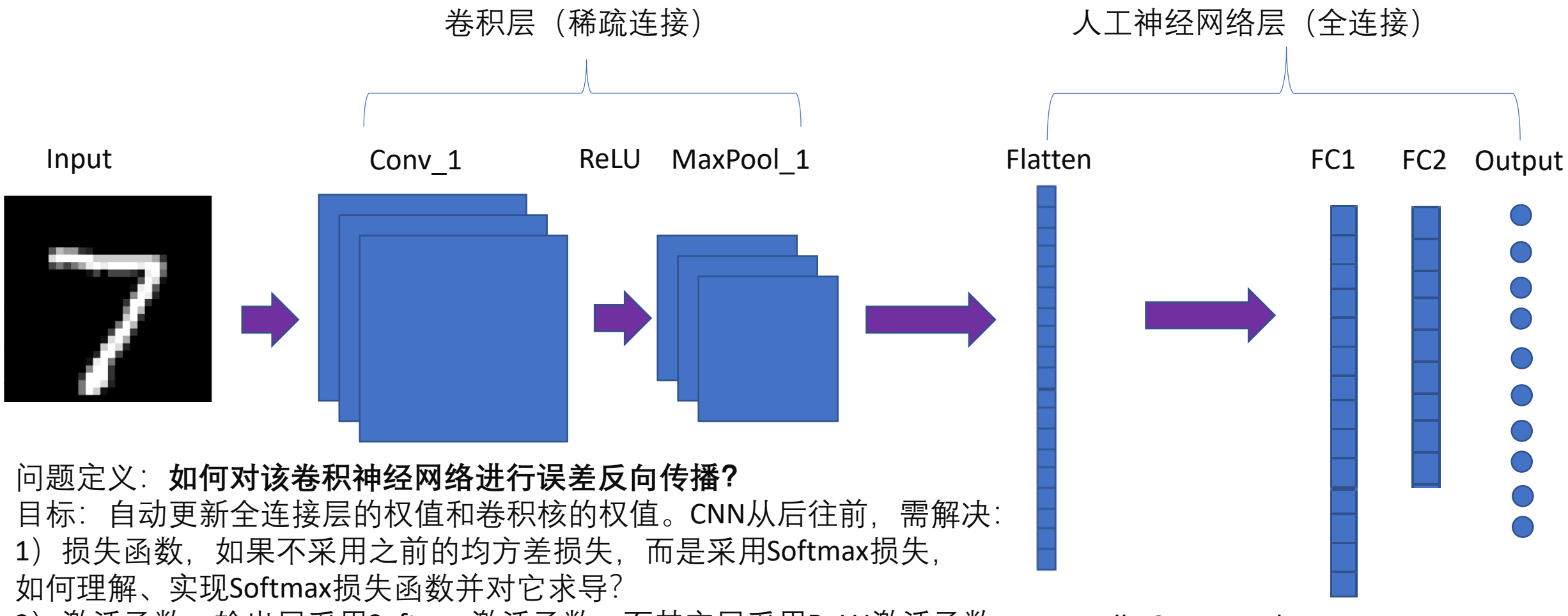
$w_o = \text{ceil}\left(\frac{|W|}{stride}\right)$

Part 04-3

Backpropagation of
Convolutional Neural Networks

卷积神经网络—误差反向传播原理

0 问题定义：卷积神经网络的误差反向传播



问题定义：**如何对该卷积神经网络进行误差反向传播?**

目标：自动更新全连接层的权值和卷积核的权值。CNN从后往前，需解决：

1）损失函数，如果不采用之前的均方差损失，而是采用Softmax损失，如何理解、实现Softmax损失函数并对它求导?

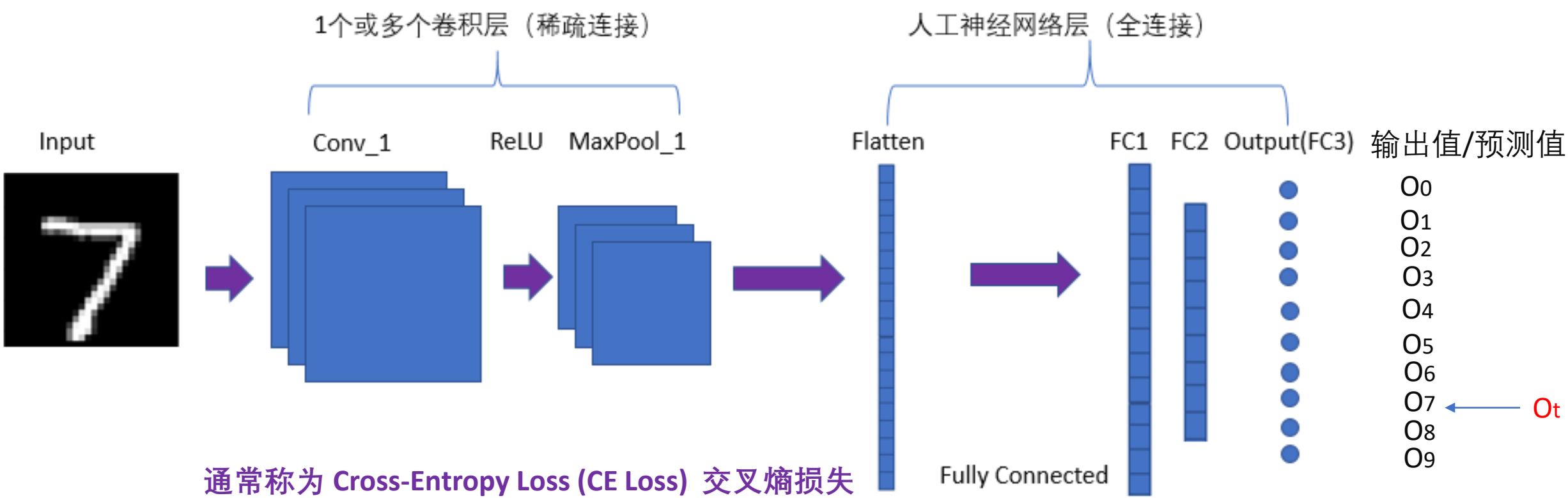
2）激活函数，输出层采用Softmax激活函数，而其它层采用ReLU激活函数，那么， Softmax激活函数和ReLU激活函数如何求导?

3）卷积神经网络中的全连接层的权值和神经元的值，如何求导?

4）下采样层（池化层）如何求导?

5） 卷积层神经元，和卷积核中的权值，如何求导?

一、Softmax损失函数（1）公式与原理



通常称为 Cross-Entropy Loss (CE Loss) 交叉熵损失

Softmax Loss，又称为Softmax Cross-Entropy Loss。对输出层的每个神经元，先使用Softmax激活函数进行激活；再使用交叉熵计算损失。

Softmax Loss: $L = -\log(O_t)$，这其实是Cross-Entropy Loss的公式

其中，t是ground truth index, 输入样本的类别对应的顺序编号

如当前输入图像是7的图像，其真实类别是7，类别7在待预测的十个类中的编号是7，故t=7

而O_t是该类别t(当前输入样本对应的真实类别)对应的预测值。期望情况下，O_t=1，或越接近于1，越好。

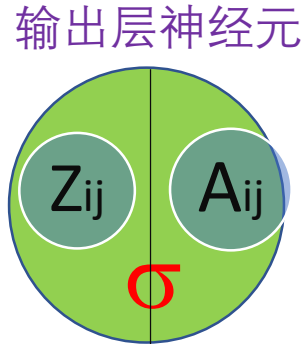
当 O_t=1 时，$L = -\log(O_t) = -\log(1) = 0$，即损失为0，没有损失; O_t = 0.9 时,$L = -\log(O_t) = -\log(0.9) = 0.1054$；$-\log(0.8) = 0.2231$

因此，Softmax Loss虽然公式简单，但能较好地反映神经网络模型的预测结果的准确程度（反过来说，即损失值的大小）。

一、Softmax损失函数（2）

每个输出层神经元都用Softmax激活

每个输出层的神经元，也要进行激活，因此，存在激活前的神经元的值Z_{ij}，和激活后的神经元值A_{ij}。
目前普遍使用Softmax激活函数。
Cross Entropy Loss又称为Softmax Cross Entropy Loss (Softmax交叉熵损失)



Softmax Loss

$L = -\log(O_t)$

t是输入样本对应的真实类别/真实标签在所有类别中的编号/索引值，O_t是该类别对应的预测值。

Cross Entropy Loss

$L = \sum_{i=0}^{9}(-O_i * \log(P_i))$，其它类的期望概率$O_i$是0（除了t之外的其它类），因此，最后 $L = -\log(O_t)$

举例

假定，共有三个类，则输出层有三个神经元；经Softmax 激活后，假定激活后的三个神经元的值，即神经网络模型预测出来的属于三个类的概率分别是0.9503，0.0473，0.0024
假定，当前输入样本对应的真实类别是第一个类，那么此时O_t为0.9503

使用Softmax Loss

$L = -\log(0.9503) = 0.0509$

详见本书教材第62-63页

一、Softmax损失函数（3）Softmax Loss的偏导值计算

如果使用Softmax Loss, 则损失函数的公式为： L = -log(Ot)

激活后神经元

$\frac{d}{dO_t}L = -\frac{1}{O_t}$， Ot神经网络在当前输入图像的真实类别t上的预测值

而对于其它类别k， k !=t, O_k与损失L无关

Softmax激活函数

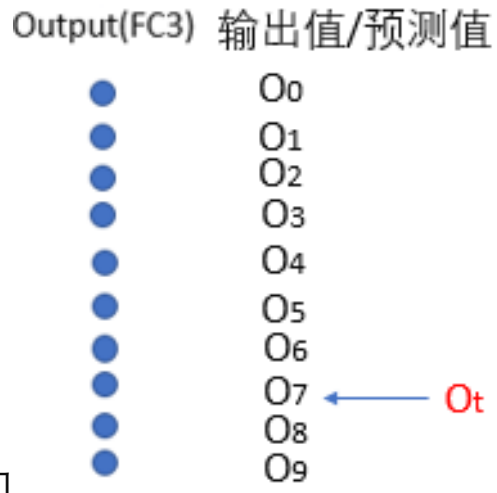
回忆Softmax激活函数, 其公式为： Softmax激活函数 = $\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$

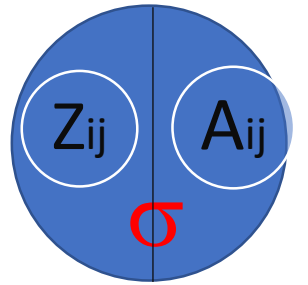
其中， Zj是输出层激活前的某个神经元的值

当分子的i=t时， Softmax激活函数的对应输出值就是Ot

由于分母的输出层是各个神经元激活前的值的自然数次方，再求和

因此， Ot与输出层的各个激活前的神经元都有函数关系。





激活前神经元

$\frac{d}{dZ_i}L = \frac{d}{dO_t}L * \frac{d}{dZ_i}O_t = \frac{d}{dO_t}L * \frac{d}{dZ_i}\left(\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}\right)$

O_t与Z_i有关

若i = t, 则 $\frac{d}{dZ_i}L = \frac{d}{dO_t}L * \frac{e^{z_t}*\left(\sum_{j=0}^{9} e^{z_j}\right) - e^{z_t}*e^{z_t}}{\left(\sum_{j=0}^{9} e^{z_j}\right)^2}$

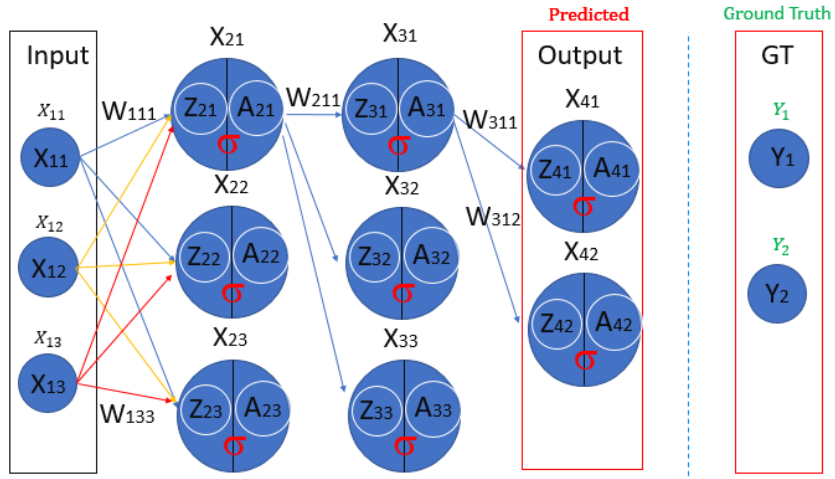
$= \frac{d}{dO_t}L * \left(\frac{e^{z_t}}{\sum_{j=0}^{9} e^{z_j}} - \frac{e^{z_t}*e^{z_t}}{\left(\sum_{j=0}^{9} e^{z_j}\right)^2}\right) = -\frac{1}{O_t} * (O_t - (O_t)^2) = O_t - 1$

O_t与Z_i无关

若i != t, 则 $\frac{d}{dZ_i}L = \frac{d}{dO_t}L * e^{z_t} * \frac{d}{dZ_i}\left(\frac{1}{\sum_{j=0}^{9} e^{z_j}}\right)$

$= \frac{d}{dO_t}L * e^{z_t} * \left(-\frac{e^{z_i}}{\left(\sum_{j=0}^{9} e^{z_j}\right)^2}\right) = \frac{d}{dO_t}L * \left(-\frac{e^{z_t}}{\sum_{j=0}^{9} e^{z_j}}\right) * \left(\frac{e^{z_i}}{\sum_{j=0}^{9} e^{z_j}}\right)$

$= -\frac{1}{O_t} * (-O_t) * O_i = O_i$



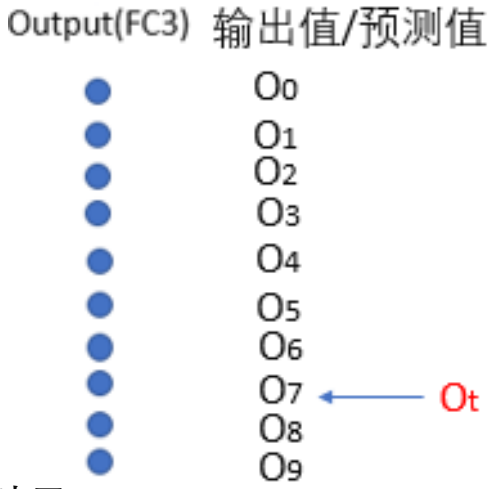
分数求导 $\left(\frac{u}{v}\right)' = \frac{u'v - uv'}{v^2}$

一、Softmax损失函数（4）Softmax Loss的偏导值计算

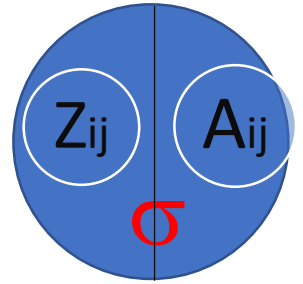
使用Softmax Loss， L = -log(Ot)， Ot神经网络在当前输入图像的真实类别t上的预测值

L = -log(Ot)

总损失L与输出层的各个激活前的神经元的偏导关系为：



输出层神经元



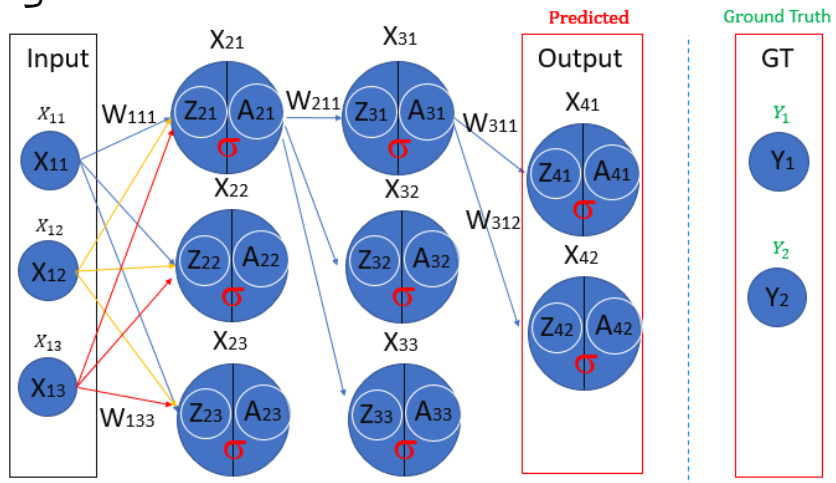
Softmax Loss
输出层
激活前神经元
偏导公式

若i = t， 则 $\frac{d}{dZ_i}L = O_t - 1$

若i != t， 则 $\frac{d}{dZ_i}L = O_i$

t为当前输入图像的真实类别对应的编号， t 在所有类别中的索引号

到这里为止，使用Softmax Loss作为损失函数，
我们求出来了输出层的各个激活前的神经元的偏导值DZij



Softmax Loss
输出层
激活前神经元
偏导值求解

```
# 用户根据当前输入图像的真实标签指定t的值
for j in range(0,10)
    DZij = Oj
DZit = Ot - 1
```

二、ReLU激活函数（1）使用ReLU激活函数神经元偏导值计算

前情提要 & 问题引入

前文

总损失，使用的是Softmax Loss

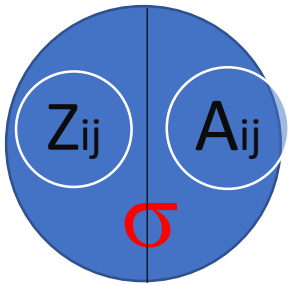
输出层神经元，使用的是Softmax激活函数

本文

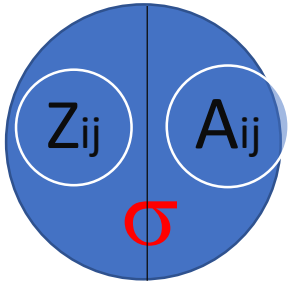
隐层神经元，　使用ReLU激活函数　$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

因此，隐层神经元，若使用ReLU激活函数，激活前的神经元的偏导值如何计算？

输出层神经元
Softmax激活



隐层神经元
ReLU激活



ReLU激活

$Z_{ij} >= 0$，　则ReLU激活后，其值保持不变，即$A_{ij} = Z_{ij} >= 0$；

$Z_{ij} < 0$，　则ReLU激活后，其值为0.　　即$A_{ij} = 0$.

ReLU偏导

$Z_{ij} >= 0$，　则 $\frac{d}{dZ_{ij}} A_{ij} = 1$

$Z_{ij} < 0$，　则 $\frac{d}{dZ_{ij}} A_{ij} = 0$

链式求导

$Z_{ij} >= 0$，　则 $\frac{d}{dZ_{ij}} L = \frac{d}{dA_{ij}} L * \frac{d}{dZ_{ij}} A_{ij} = \frac{d}{dA_{ij}} L$

$Z_{ij} < 0$，　则 $\frac{d}{dZ_{ij}} L = 0$

因此，使用ReLU激活函数，
激活前的神经元的值>=0，则其偏导值与激活后神经元偏导一样
激活前的神经元的值<0，　则其相对于总损失L的偏导值为0.

二、ReLU激活函数（2）使用ReLU激活函数元素偏导值计算

ReLU激活函数

$$f(x)=\begin{cases}0 & \text{for} \quad x<0\\ x & \text{for} \quad x\geq 0\end{cases}$$

ReLU偏导求解

$Z_{ij} >= 0$ ， 则 $\frac{d}{dZ_{ij}}L=\frac{d}{dA_{ij}}L*\frac{d}{dZ_{ij}}A_{ij}=\frac{d}{dA_{ij}}L$

$Z_{ij} < 0$ ， 则 $\frac{d}{dZ_{ij}}L=0$

ReLU激活重要结论

使用ReLU激活函数，

激活前的神经元的值>=0，则其偏导值与激活后神经元一样，即 $\frac{d}{dZ_{ij}}L=\frac{d}{dA_{ij}}L$

Z_{ij} >=0　　激活前和激活后的偏导值完全相同，故仅需计算后者

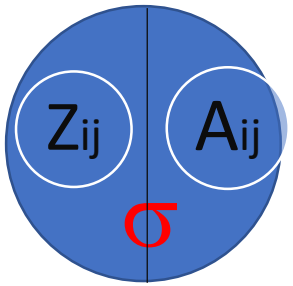
且$A_{ij}=Z_{ij}$

激活前的神经元的值<0，则其相对于总损失L的偏导值为0.

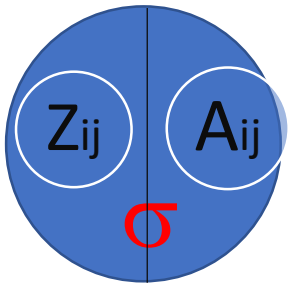
$Z_{ij} < 0$　　激活前的偏导值为0，不需要计算。

且$A_{ij}=0$

输出层神经元
Softmax激活



隐层神经元
ReLU激活



详见本书教材第64-65页

卷积层中：矩阵中的元素值便是上面说的神经元值

二、ReLU激活函数（3）使用ReLU激活函数神经元偏导值计算

Softmax Loss
隐层神经元
ReLU激活
偏导值求解

ReLU激活函数，神经元求偏导值

```
# ReLU激活，对于第i个层神经网络（且为隐层），求激活前和激活后的偏导值
# 注：需要先计算DAij

for j in 1...Ni
    if Zij < 0:
        DZij = 0
        Aij = 0
    else:
        DZij = DAij
```

上述算法，局限于当前层的神经元的偏导值计算，但是无法计算得到DA_{ij}

下面，计算DA_{ij}时要区分：

全连接层的神经元
卷积层的神经元

$$Z_{41}=A_{31}*W_{311}+A_{32}*W_{321}+A_{33}*W_{331}+B_{41}$$

三、全连接层神经元偏导值计算，使用ReLU激活函数（1）

使用Softmax Loss, 损失函数的公式为： L = -log(Ot)

输出层 FC3 层

输出层激活后的神经元和激活前的神经元的偏导值，上面已经得出，即$\frac{d}{dO_i}$L, $\frac{d}{dZ_{ij}}$L，i是最后一层的层编号N

输出层前一层 FC2 层

输出层的前一层，是隐层，即第N-1层，对应图中FC2
由于该层FC2与输出层FC3是全连接，因此，FC2层的每个神经元与FC3层的所有神经元进行连接，即全连接

FC2 层激活后神经元求偏导

与人工神经网络的误差反向传播原理类似，此时i=N-1

$$\frac{d}{dA_{ij}}L = \sum_{k=1}^{N_{i+1}}\left(\frac{d}{dZ_{(i+1)k}}L * W_{ijk}\right) = \sum_{k=1}^{N_{i+1}}(DZ_{(i+1)k} * W_{ijk})$$

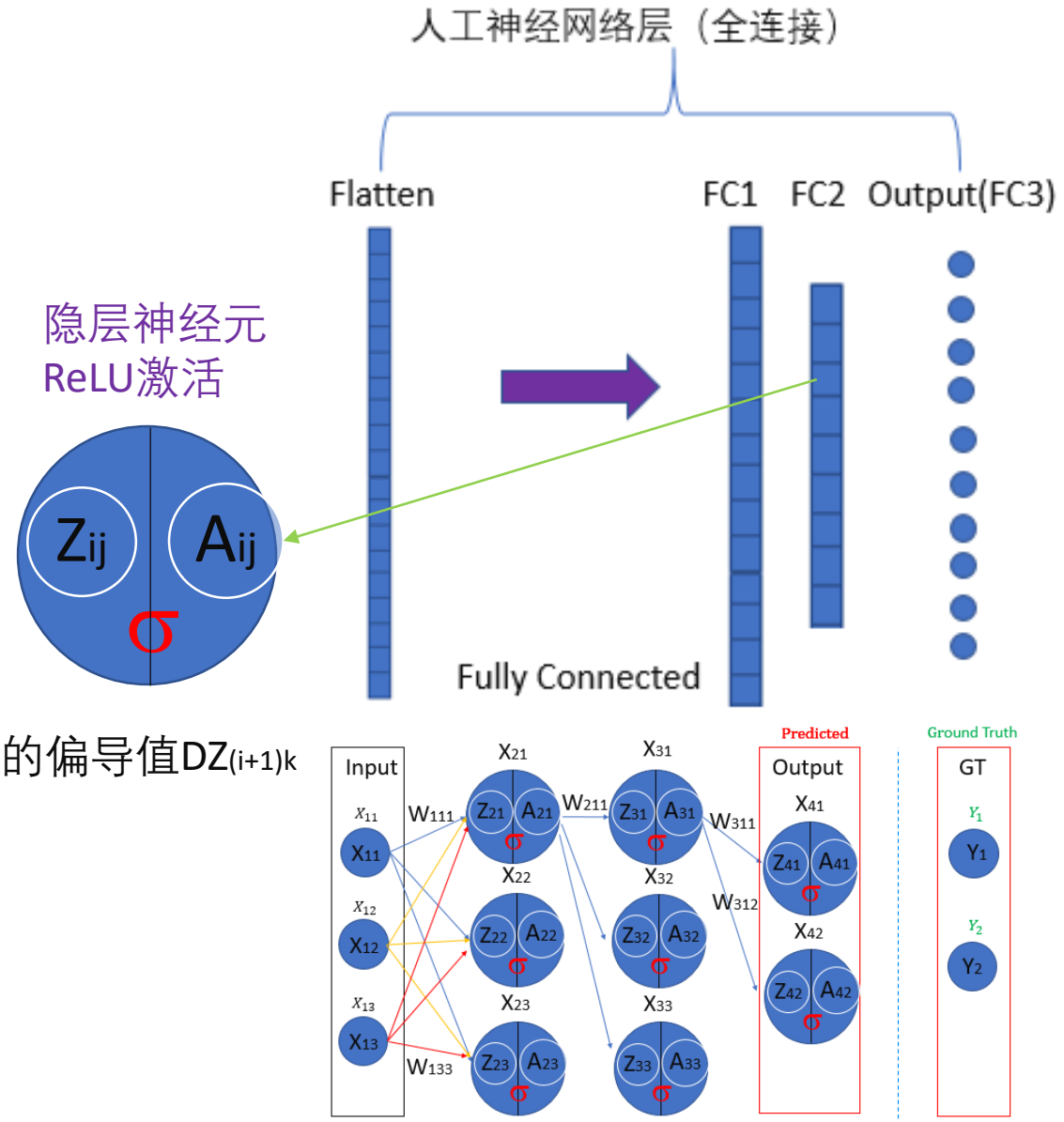
神经元Xij，下一层神经网络的每个激活前的神经元X(i+1)k的偏导值DZ(i+1)k乘以Xij与X(i+1)k之间的权值Wijk，再求和

FC2 层激活前神经元求偏导

Zij >= 0 ， 则$\frac{d}{dZ_{ij}}L = \frac{d}{dA_{ij}}L$

Zij < 0 ， 则$\frac{d}{dZ_{ij}}L = 0$

使用ReLU激活函数



$$Z_{41}=A_{31}*W_{311}+A_{32}*W_{321}+A_{33}*W_{331}+B_{41}$$

三、全连接层神经元偏导值计算，使用ReLU激活函数（2）

FC2 层
激活后
神经元
求偏导

与人工神经网络的误差反向传播原理类似，此时i=N-1

$$\frac{d}{dA_{ij}}L=\sum_{k=1}^{N_{i+1}}\left(\frac{d}{dZ_{(i+1)k}}L*W_{ijk}\right)=\sum_{k=1}^{N_{i+1}}(DZ_{(i+1)k}*W_{ijk})$$

神经元X_{ij}，下一层神经网络的每个激活前的神经元$X_{(i+1)k}$的偏导值$DZ_{(i+1)k}$乘以X_{ij}与$X_{(i+1)k}$之间的权值W_{ijk}，再求和

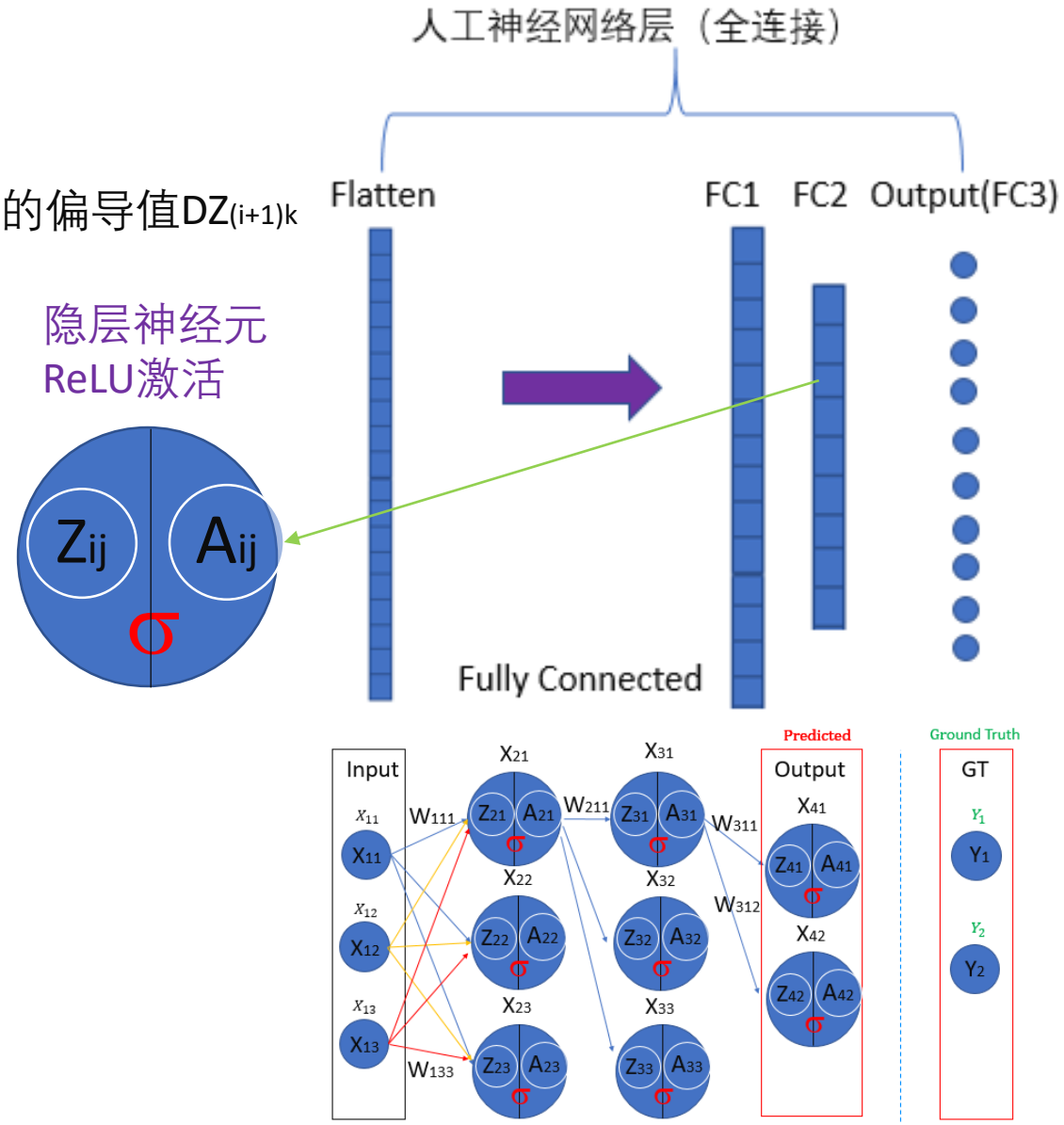
FC2 层
FC3 层
权值
求偏导

$$\frac{d}{dW_{ijk}}L=\frac{d}{dZ_{(i+1)k}}L*A_{ij}=DZ_{(i+1)k}*A_{ij}$$

FC2 层
FC3 层
偏移量
求偏导

$$\frac{d}{dB_{(i+1)k}}L=\frac{d}{dZ_{(i+1)k}}L=DZ_{(i+1)k}$$

$$\frac{d}{dB_{ij}}L=\frac{d}{dZ_{ij}}L=DZ_{ij}$$



$$Z_{41}=A_{31}*W_{311}+A_{32}*W_{321}+A_{33}*W_{331}+B_{41}$$

三、全连接层神经元偏导值计算，使用ReLU激活函数（3）

激活后神经元 A_{ij} 求偏导

$$\frac{d}{dA_{ij}}L = \sum_{k=1}^{N_{i+1}}(DZ_{(i+1)k} * W_{ijk})$$

激活前神经元 Z_{ij} 求偏导

$Z_{ij} >= 0$ ， 则 $\frac{d}{dZ_{ij}}L = \frac{d}{dA_{ij}}L$

$Z_{ij} < 0$ ， 则 $\frac{d}{dZ_{ij}}L = 0$

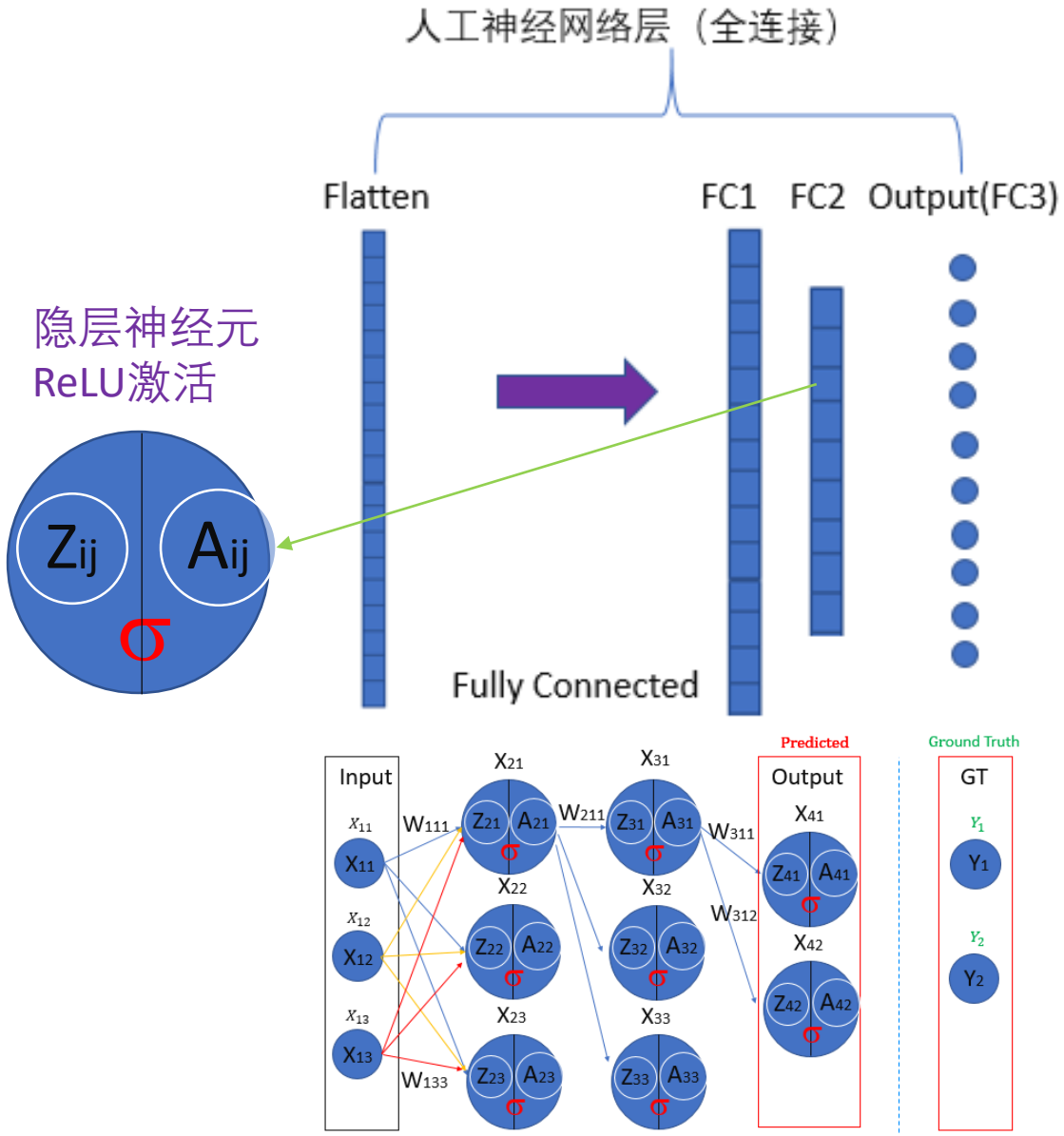
ReLU 激活函数

权值 W_{ijk} 求偏导

$$\frac{d}{dW_{ijk}}L = DZ_{(i+1)k} * A_{ij}$$

偏移量 B_{ij} 求偏导

$$\frac{d}{dB_{ij}}L = DZ_{ij}$$



$$Z_{41}=A_{31}*W_{311}+A_{32}*W_{321}+A_{33}*W_{331}+B_{41}$$

三、全连接层神经元偏导值计算，使用ReLU激活函数（4）

Softmax Loss
全连接层
隐层神经元

激活前激活后
偏导值求解
权值求偏导

卷积神经网络中，全连接层的神经元求偏导值

```
# 对于第i个层神经网络（且为隐层），全连接层
# 注：事先已计算得到下一层神经元的DZ(i+1)j

for j in range(1, 1+Ni)
    for k in range(1, 1+Ni+1)
        DAij += DZ(i+1)k * W_ijk
        DWijk = DZ(i+1)k * A_ij

    if Zij < 0:
        DZij = 0
    else:
        DZij = DAij

    DBij = DZij
```

(1) 激活后神经元求偏导值

(2) 相邻层神经元之间的权值求偏导值

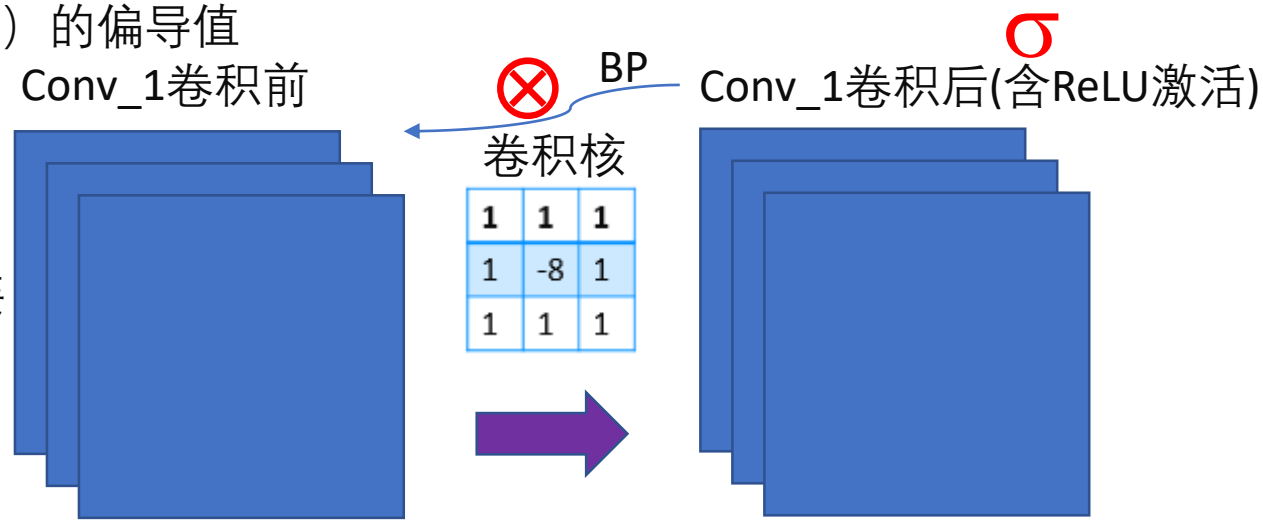
(3) 激活前神经元求偏导值

(4) 当前神经元对应的偏移量求偏导值

四、卷积层神经元偏导值计算，使用ReLU激活函数（1）

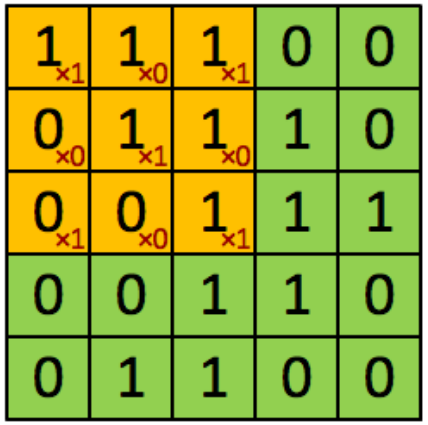
问题定义

卷积神经网络中，卷积操作之后，得到的矩阵（神经元），对应的偏导值已通过前面的反向传播得到
现在需要计算卷积前的矩阵（中的元素值/神经元）的偏导值

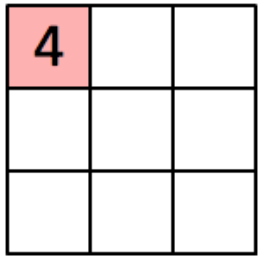


偏导求解主要难点

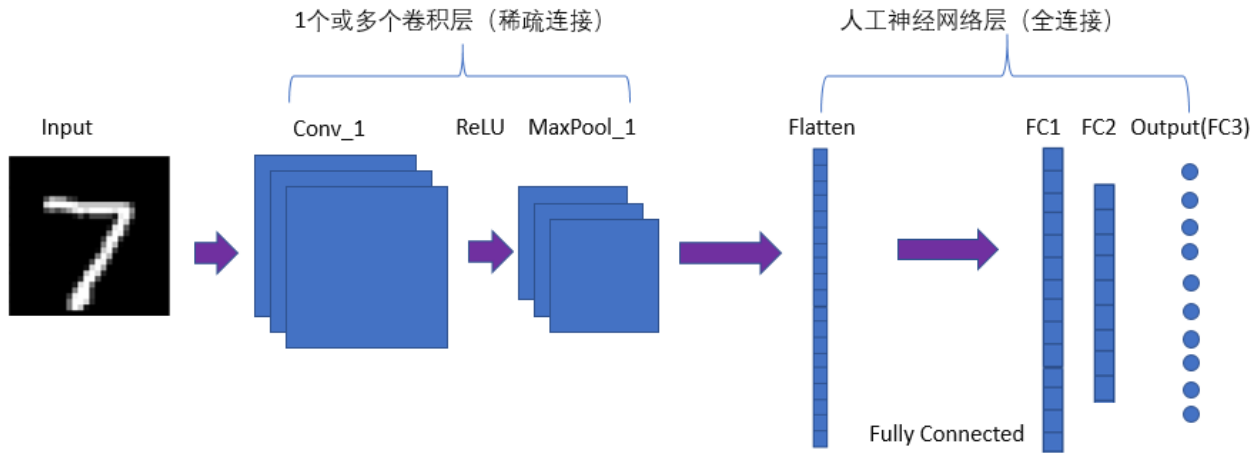
卷积神经网络不同于人工神经网络之处：
1）上一层的神经元只与下一层的少数神经元连接
属于稀疏连接，而非全连接，
2）权值共享，且权值都存在于卷积核矩阵中，
一个权值可以被多个连接所共用。



Image



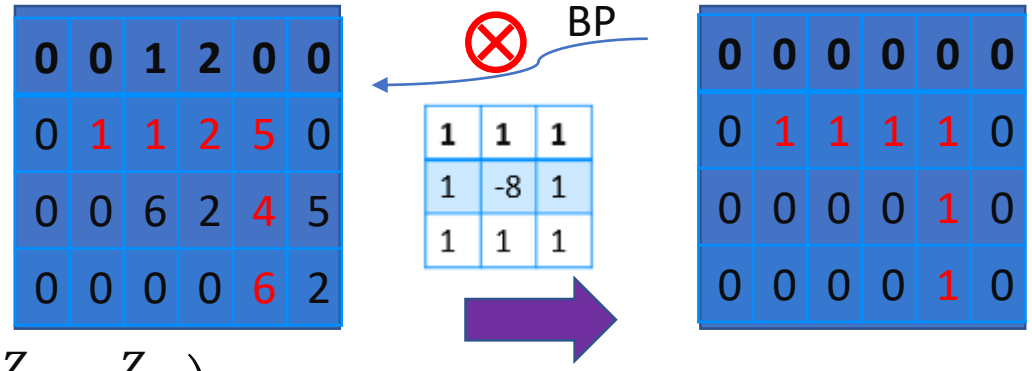
Convolved Feature



四、卷积层神经元偏导值计算，使用ReLU激活函数（2）

问题定义

卷积神经网络中，卷积操作之后，得到的矩阵（神经元），对应的偏导值已通过前面的反向传播得到
现在需要计算卷积前的矩阵（神经元）的偏导值



偏导求解

卷积神经网络的卷积操作：

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} \otimes \begin{pmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{pmatrix} = \begin{pmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{pmatrix}$$

卷积后

$$Z_{11}= A_{11}*W_{11}+ A_{12}*W_{12} + A_{13}*W_{13} +A_{21}*W_{21}+ A_{22}*W_{22} + A_{23}*W_{23} + A_{31}*W_{31}+ A_{32}*W_{32} + A_{33}*W_{33} + b$$
$$Z_{12}= A_{12}*W_{11}+ A_{13}*W_{12} + A_{14}*W_{13} +A_{22}*W_{21}+ A_{23}*W_{22} + A_{24}*W_{23} + A_{32}*W_{31}+ A_{33}*W_{32} + A_{34}*W_{33} + b$$
$$Z_{21}= A_{21}*W_{11}+ A_{22}*W_{12} + A_{23}*W_{13} +A_{31}*W_{21}+ A_{32}*W_{22} + A_{33}*W_{23} + A_{41}*W_{31}+ A_{42}*W_{32} + A_{43}*W_{33} + b$$
$$Z_{22}= A_{22}*W_{11}+ A_{23}*W_{12} + A_{24}*W_{13} +A_{32}*W_{21}+ A_{33}*W_{22} + A_{34}*W_{23} + A_{42}*W_{31}+ A_{43}*W_{32} + A_{44}*W_{33} + b$$

$$\frac{d}{dW_{11}}L = A_{11}*\frac{d}{dZ_{11}}L + A_{12}*\frac{d}{dZ_{12}}L + A_{21}*\frac{d}{dZ_{21}}L + A_{22}*\frac{d}{dZ_{22}}L = A_{11}*DZ_{11} + A_{12}*dZ_{12} + A_{21}*DZ_{21} + A_{22}*DZ_{22}$$
$$\frac{d}{dW_{12}}L = A_{12}*\frac{d}{dZ_{11}}L + A_{13}*\frac{d}{dZ_{12}}L + A_{22}*\frac{d}{dZ_{21}}L + A_{23}*\frac{d}{dZ_{22}}L = A_{12}*DZ_{11} + A_{13}*dZ_{12} + A_{22}*DZ_{21} + A_{23}*DZ_{22}$$
$$\frac{d}{dW_{22}}L = A_{22}*\frac{d}{dZ_{11}}L + A_{23}*\frac{d}{dZ_{12}}L + A_{32}*\frac{d}{dZ_{21}}L + A_{33}*\frac{d}{dZ_{22}}L = A_{22}*DZ_{11} + A_{23}*dZ_{12} + A_{32}*DZ_{21} + A_{33}*DZ_{22}$$

四、卷积层神经元偏导值计算，使用ReLU激活函数（3）

卷积操作

卷积神经网络的卷积操作：

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} \otimes \begin{pmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{pmatrix} = \begin{pmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{pmatrix}$$

卷积后

权值偏导

$$\begin{pmatrix} DW_{11} & DW_{12} & DW_{13} \\ DW_{21} & DW_{22} & DW_{23} \\ DW_{31} & DW_{32} & DW_{33} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} \otimes \begin{pmatrix} DZ_{11} & DZ_{12} \\ DZ_{21} & DZ_{22} \end{pmatrix}$$

$$Z_{11} = A_{11}*W_{11} + A_{12}*W_{12} + A_{13}*W_{13} + A_{21}*W_{21} + A_{22}*W_{22} + A_{23}*W_{23} + A_{31}*W_{31} + A_{32}*W_{32} + A_{33}*W_{33} + b$$
$$Z_{12} = A_{12}*W_{11} + A_{13}*W_{12} + A_{14}*W_{13} + A_{22}*W_{21} + A_{23}*W_{22} + A_{24}*W_{23} + A_{32}*W_{31} + A_{33}*W_{32} + A_{34}*W_{33} + b$$
$$Z_{21} = A_{21}*W_{11} + A_{22}*W_{12} + A_{23}*W_{13} + A_{31}*W_{21} + A_{32}*W_{22} + A_{33}*W_{23} + A_{41}*W_{31} + A_{42}*W_{32} + A_{43}*W_{33} + b$$
$$Z_{22} = A_{22}*W_{11} + A_{23}*W_{12} + A_{24}*W_{13} + A_{32}*W_{21} + A_{33}*W_{22} + A_{34}*W_{23} + A_{42}*W_{31} + A_{43}*W_{32} + A_{44}*W_{33} + b$$

$$\frac{d}{dW_{11}}L = A_{11}*\frac{d}{dZ_{11}}L + A_{12}*\frac{d}{dZ_{12}}L + A_{21}*\frac{d}{dZ_{21}}L + A_{22}*\frac{d}{dZ_{22}}L = A_{11}*DZ_{11} + A_{12}*dZ_{12} + A_{21}*DZ_{21} + A_{22}*DZ_{22}$$
$$\frac{d}{dW_{12}}L = A_{12}*\frac{d}{dZ_{11}}L + A_{13}*\frac{d}{dZ_{12}}L + A_{22}*\frac{d}{dZ_{21}}L + A_{23}*\frac{d}{dZ_{22}}L = A_{12}*DZ_{11} + A_{13}*dZ_{12} + A_{22}*DZ_{21} + A_{23}*DZ_{22}$$
$$\frac{d}{dW_{22}}L = A_{22}*\frac{d}{dZ_{11}}L + A_{23}*\frac{d}{dZ_{12}}L + A_{32}*\frac{d}{dZ_{21}}L + A_{33}*\frac{d}{dZ_{22}}L = A_{22}*DZ_{11} + A_{23}*dZ_{12} + A_{32}*DZ_{21} + A_{33}*DZ_{22}$$

四、卷积层神经元偏导值计算，使用ReLU激活函数（4）

卷积操作

卷积后

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} \otimes \begin{pmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{pmatrix} = \begin{pmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{pmatrix}$$

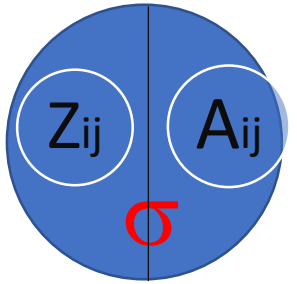
权值偏导

$$\begin{pmatrix} DW_{11} & DW_{12} & DW_{13} \\ DW_{21} & DW_{22} & DW_{23} \\ DW_{31} & DW_{32} & DW_{33} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} \otimes \begin{pmatrix} DZ_{11} & DZ_{12} \\ DZ_{21} & DZ_{22} \end{pmatrix}$$

神经元值偏导

卷积前

$$\begin{pmatrix} DA_{11} & DA_{12} & DA_{13} & DA_{14} \\ DA_{21} & DA_{22} & DA_{23} & DA_{24} \\ DA_{31} & DA_{32} & DA_{33} & DA_{34} \\ DA_{41} & DA_{42} & DA_{43} & DA_{44} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & DZ_{11} & DZ_{12} & 0 & 0 \\ 0 & 0 & DZ_{21} & DZ_{22} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} W_{33} & W_{32} & W_{31} \\ W_{23} & W_{22} & W_{21} \\ W_{13} & W_{12} & W_{11} \end{pmatrix}$$



神经元值偏导

激活前

zij >= 0 ， 则$dz_{ij} = DA_{ij}$

zij < 0 ， 则$dz_{ij} = 0$

偏移量值偏导

$$db = DZ_{11} + DZ_{12} + DZ_{21} + DZ_{22}$$

四、卷积层神经元偏导值计算，使用ReLU激活函数（5）

卷积层神经元偏导值计算，公式汇总

权值偏导

$$\begin{pmatrix} DW_{11} & DW_{12} & DW_{13} \\ DW_{21} & DW_{22} & DW_{23} \\ DW_{31} & DW_{32} & DW_{33} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix} \otimes \begin{pmatrix} DZ_{11} & DZ_{12} \\ DZ_{21} & DZ_{22} \end{pmatrix}$$

神经元值偏导

激活后

$$\begin{pmatrix} DA_{11} & DA_{12} & DA_{13} & DA_{14} \\ DA_{21} & DA_{22} & DA_{23} & DA_{24} \\ DA_{31} & DA_{32} & DA_{33} & DA_{34} \\ DA_{41} & DA_{42} & DA_{43} & DA_{44} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & DZ_{11} & DZ_{12} & 0 & 0 \\ 0 & 0 & DZ_{21} & DZ_{22} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{pmatrix}^{\text{rot180}}$$

偏移量值偏导

$$db = DZ_{11} + DZ_{12} + DZ_{21} + DZ_{22}$$

神经元值偏导

激活前

$z_{ij} >= 0$，　则$dz_{ij} = DA_{ij}$

$z_{ij} < 0$，　则$dz_{ij} = 0$

详见本书教材第61-72页

五、池化层神经元偏导值计算 MaxPooling

池化前的卷积层神经元求偏导值

令Z_{ij} 是池化层中的某个神经元的值
找到Z_{ij} 在Conv_1矩阵中的对应池化窗口PW，如2*2

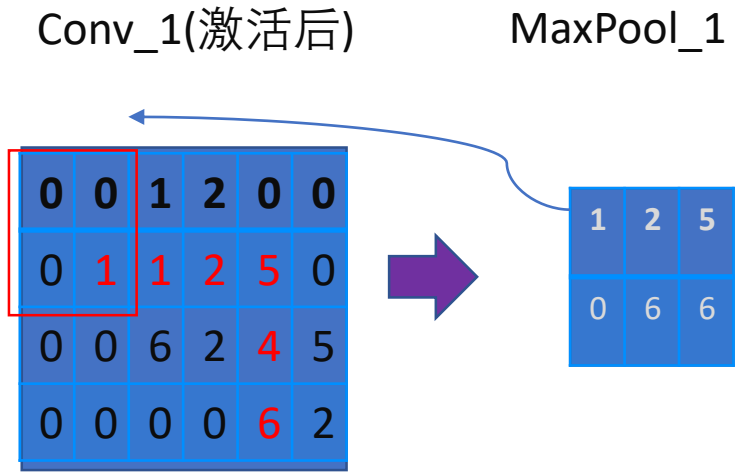
对该池化窗口PW中的每个元素A_{ij}

If A_{ij} 是当前窗口中的最大值，

$$\frac{d}{dA_{ij}}L = \frac{d}{dZ_{ij}}L$$ ，即DAij= DZij

Else

$$\frac{d}{dA_{ij}}L = 0$$ ，即 DAij = 0



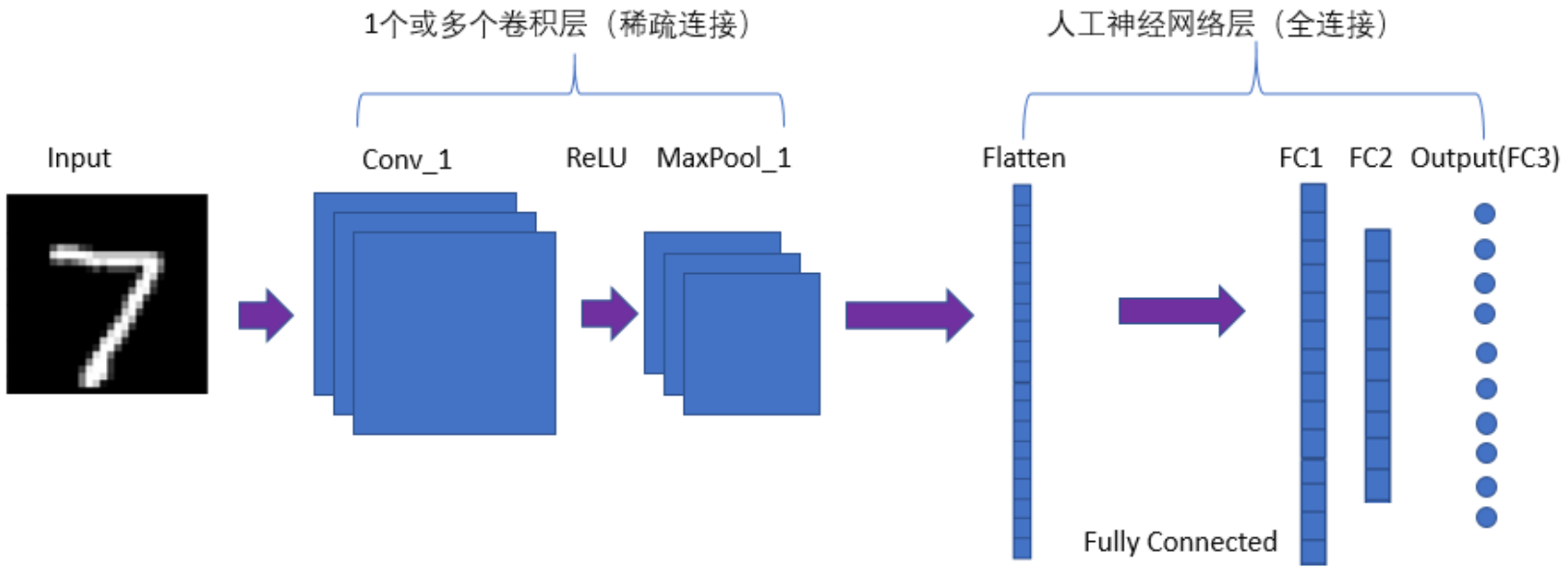
如果是AvgPooling，则池化前的卷积层中，每个池化窗口中的元素的偏导值是 ¼*DZij

六、卷积神经网络的整体误差反向传播过程

求导顺序是从后往前，从输出层，到人工神经网络层到，卷积神经网络层

1. 定义总的损失函数公式，如Softmax Loss
2. 对于输出层，每个激活后的神经元（即预测值），求其相对于总损失的偏导值；
3. 对于输出层，每个激活前的神经元，求其相对于总损失的偏导值；Softmax激活函数
4. 对于全连接层的每个神经元，分别求其激活后和激活前的偏导值，及每个连接的权值；使用ReLU激活
5. 对于池化层，对池化前的卷积层的神经元求偏导值；使用ReLU激活
6. 对于卷积层，对卷积前的矩阵中的神经元求偏导值，求卷积核中的权值的偏导值；使用ReLU激活

若步骤4的全连接层，步骤5的池化和步骤6的卷积操作有多组，则分别重复执行。



作业 卷积神经网络的误差反向传播推导