2023-2024学年第一学期本科生课程

《神经网络与深度学习》

第四节：卷积神经网络(CNN)

主讲人：戴金晟(副教授，博士生导师)

daijincheng@bupt.edu.cn

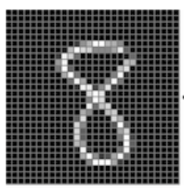
神经网络与深度学习课程组

北京邮电大学
Beijing University of Posts and Telecommunications

计算机眼中的图像

黑白图片可用一个矩阵表示

对于每个像素点，0表示最暗，255表示最亮



28 x 28
784 pixels

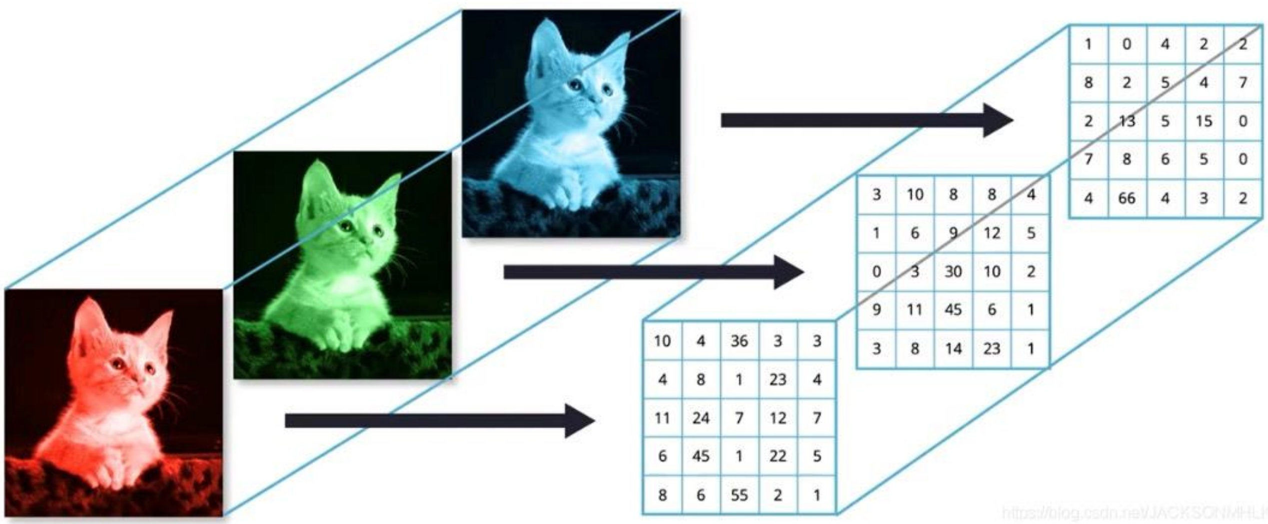
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	4	62	146	182	254	254	181	176	139	15	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	34	186	253	217	208	136	136	136	166	232	99	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	61	242	208	111	3	0	0	0	0	0	18	32	107	43	0	0	0	0	0	0
0	0	0	0	0	0	0	0	156	242	23	0	0	0	0	0	0	0	13	191	181	6	0	0	0	0	0	0
0	0	0	0	0	0	0	0	121	255	98	3	0	0	0	0	0	8	194	225	12	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	169	253	120	3	0	0	0	0	128	247	51	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	3	111	244	169	19	0	14	131	249	117	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	59	241	235	72	142	229	66	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	25	218	254	231	36	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	133	253	221	33	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	19	237	111	196	217	19	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	174	138	0	23	193	204	18	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	96	224	0	0	0	25	218	169	3	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	138	0	0	0	0	86	253	99	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	97	0	0	0	0	3	162	214	11	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	215	97	0	0	0	0	0	118	253	68	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	185	157	0	0	0	0	0	40	254	98	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	50	244	61	0	0	0	0	112	244	58	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	174	251	142	59	83	167	244	111	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	6	133	253	253	253	169	61	3	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28*28*1

计算机眼中的图像

彩色图片用三个矩阵表示

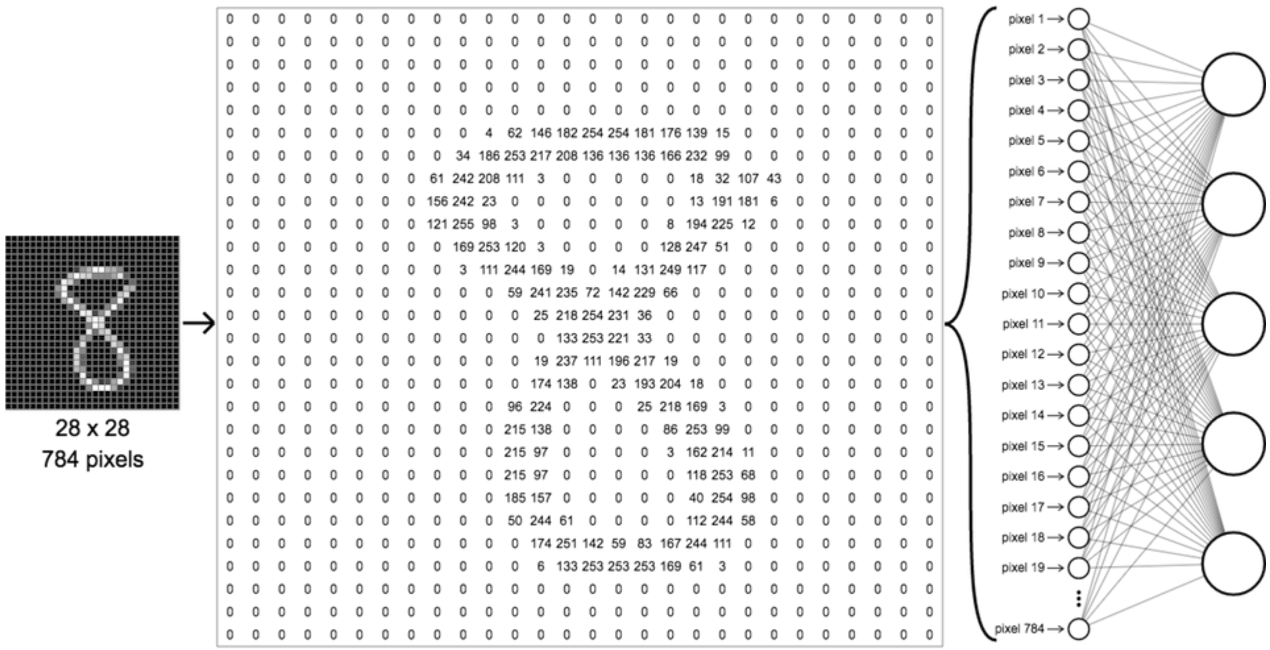
RGB 颜色模型：红，绿，蓝 三原色



5*5*3

全连接神经网络进行图像识别

- 图片的像素尺寸为28×28，$28\times28\times1=784$，即输入层的数据长度是 784



将28*28*1个输入层神经元拉直

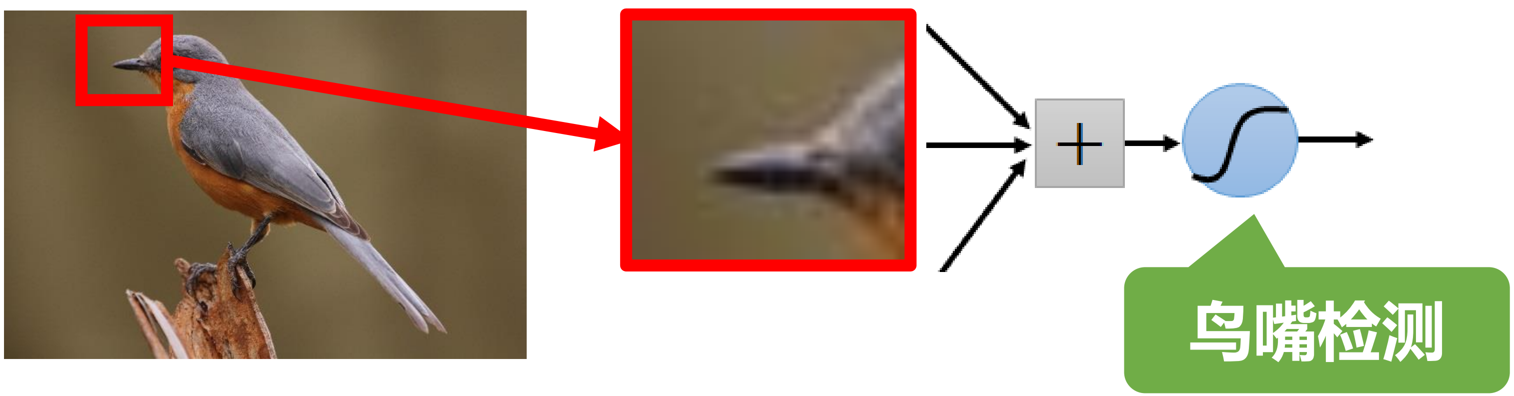
- 参数过多，计算量庞大，假设隐藏层神经元个数和输入层尺寸一致，权重 w 的个数是$(28\times28)^2=614656$！

观察现实世界的图像特点

□特点一：某些图样所在区域远小于整张图像

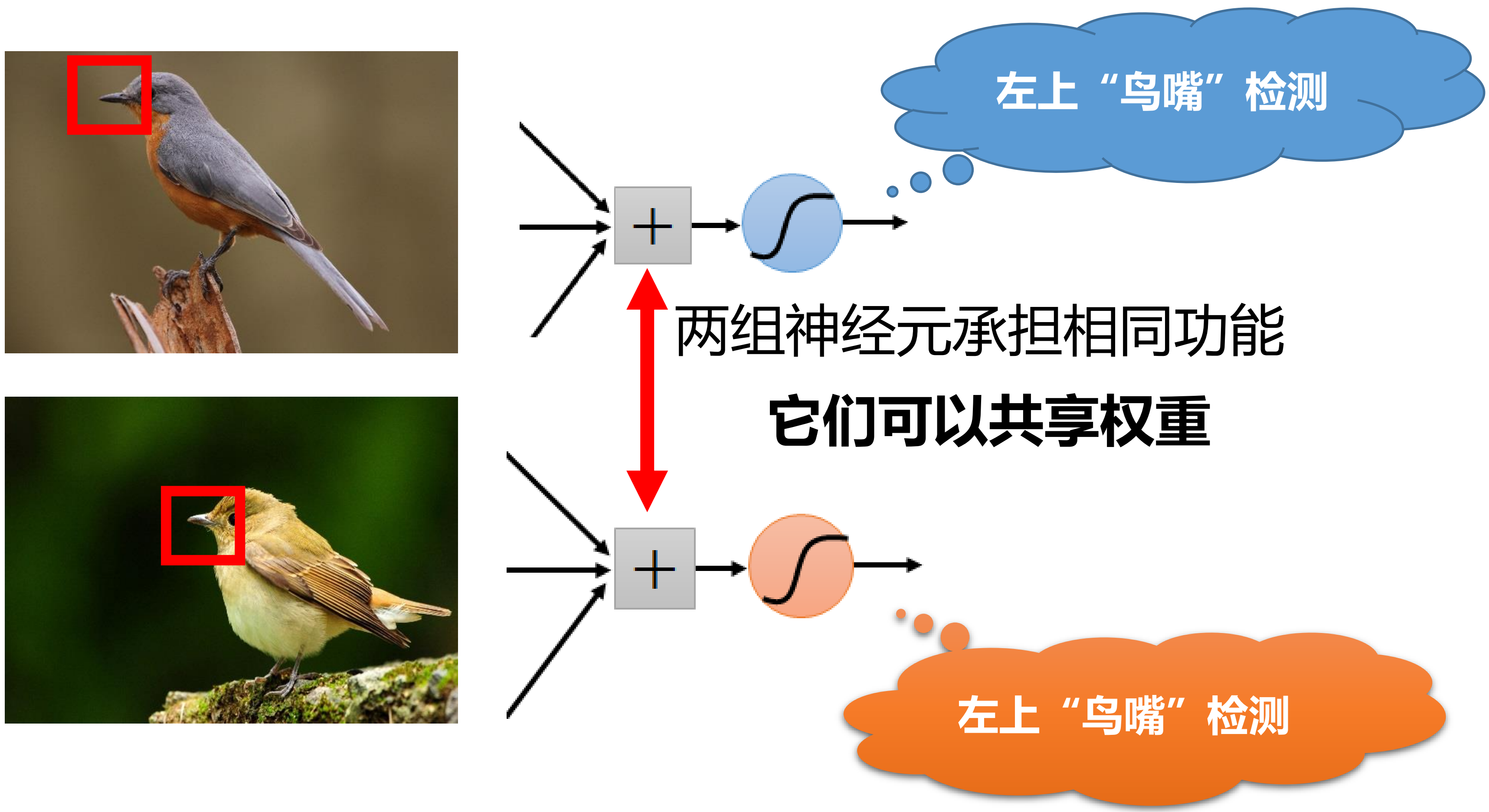
神经元无需“看到”整张图像去识别某一特定图样

局部连接减少参数数量



观察现实世界的图像特点

口特点二：同样的图样出现在不同区域



观察现实世界的图像特点

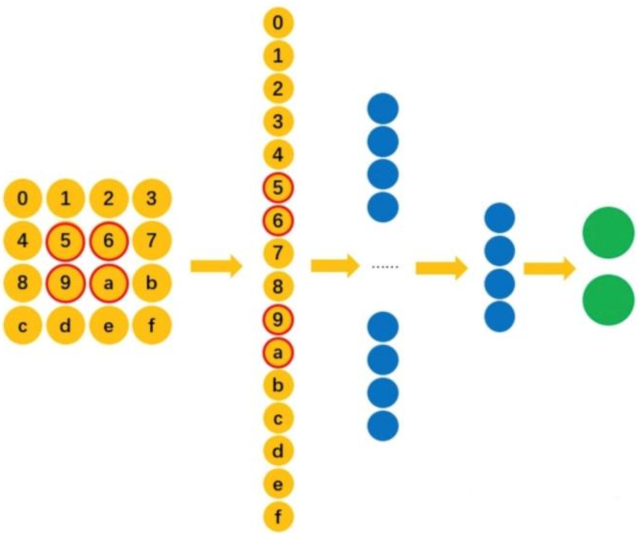
▫特点三：图像下采样可能不影响图像内容

下采样

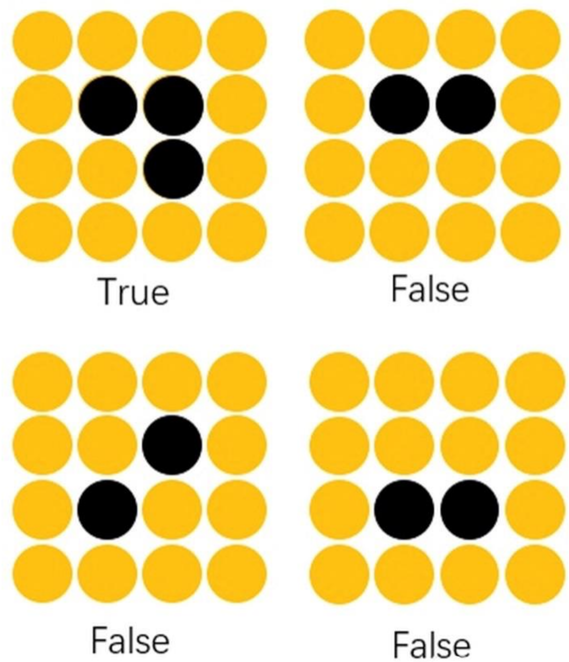
对图像可以下采样，不影响图像内容识别，由此有效降低网络参数数量

全连接神经网络进行图像识别

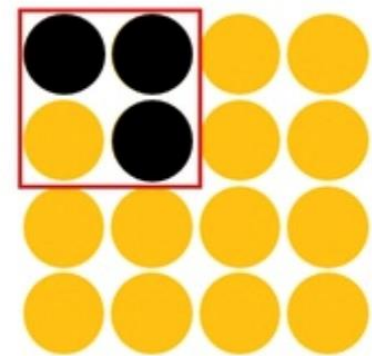
- 全连接的情况下，无法做到图片的形变识别，图像旋转或平移形变之后，MLP便无法识别



训练集



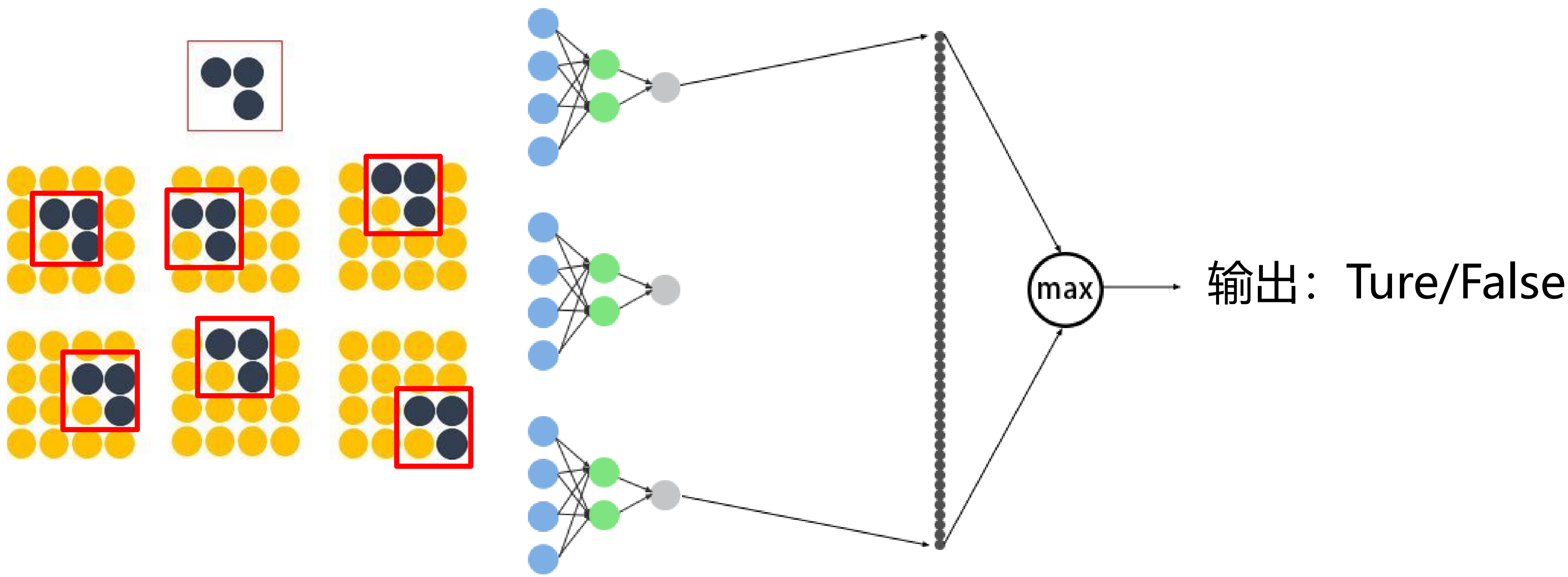
测试集



结果为False

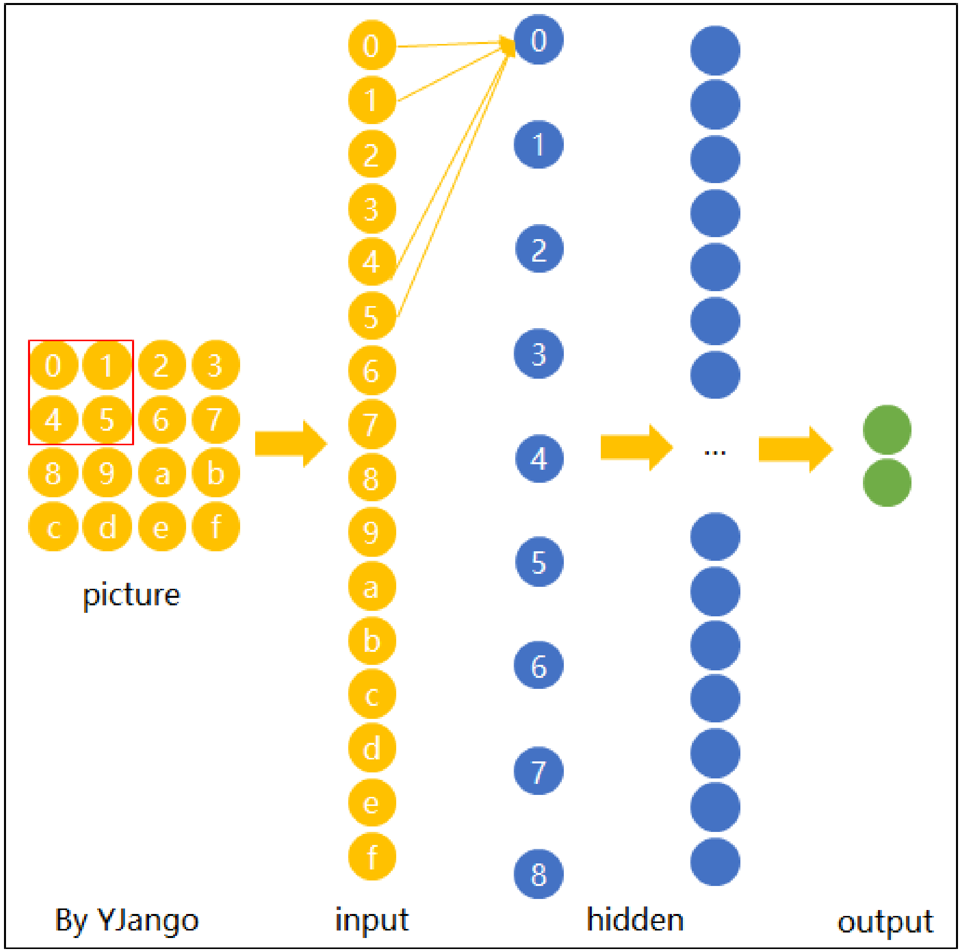
全连接神经网络进行图像识别的改进

- 改进之后，图片中任何位置的“横折”都可以识别



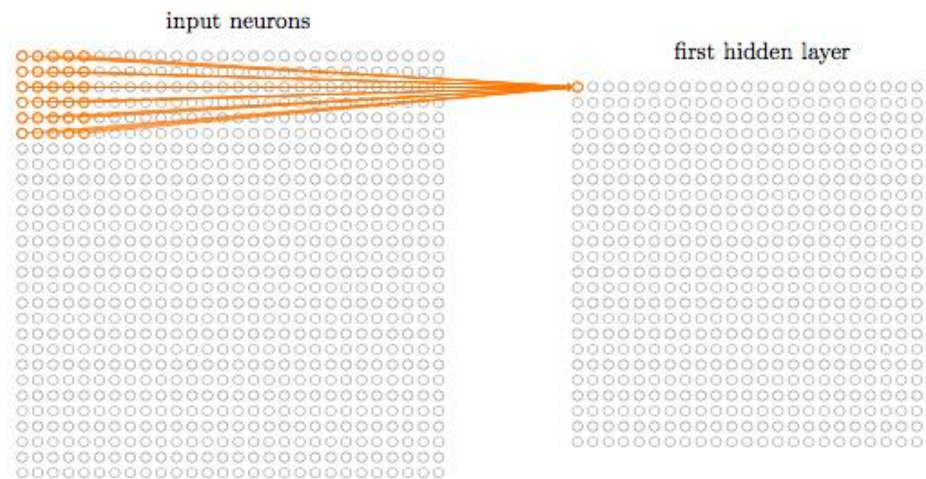
全连接神经网络进行图像识别的改进

- 先选择一个局部区域，用这个局部区域去扫描整张图片
- 局部区域所圈起来的所有节点会被连接到下一层的一个节点上



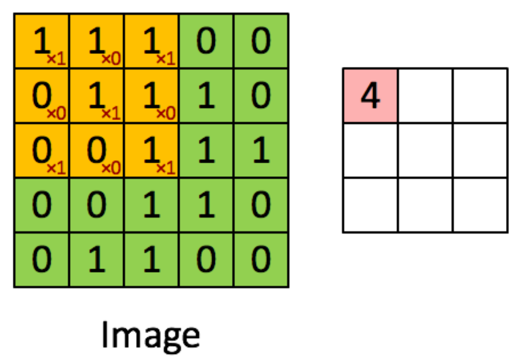
全连接神经网络进行图像识别的改进

- 下一层的每个神经元只需要与上一层部分神经元相连接
- 传统神经网络中一个神经元与全部的输入节点连接



局部连接

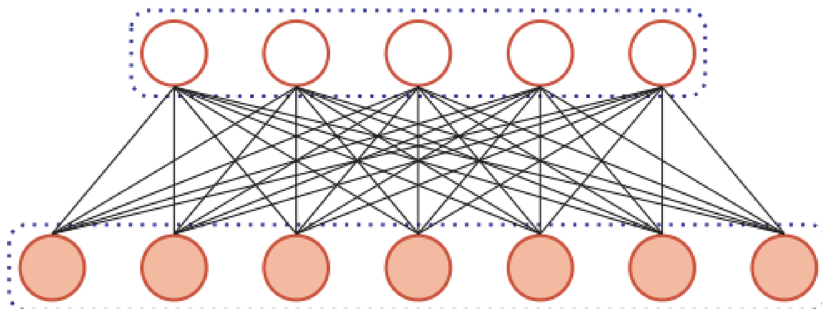
- **目标/特征在任何位置都可以被发现**
- **识别一个特征的网络权重是一样的**，则连接了一块局部和一个隐藏层神经元的每组权重的值都应该一样



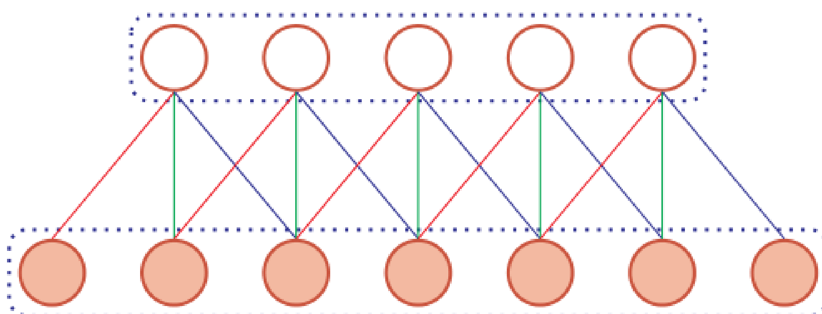
权值共享

卷积神经网络(Convolutional Neural Networks, CNN)

- 卷积神经网络(CNN)也是一种前馈神经网络，其特点是**每层的神经元节点只响应前一层局部区域范围内的神经元**



(a) 全连接层



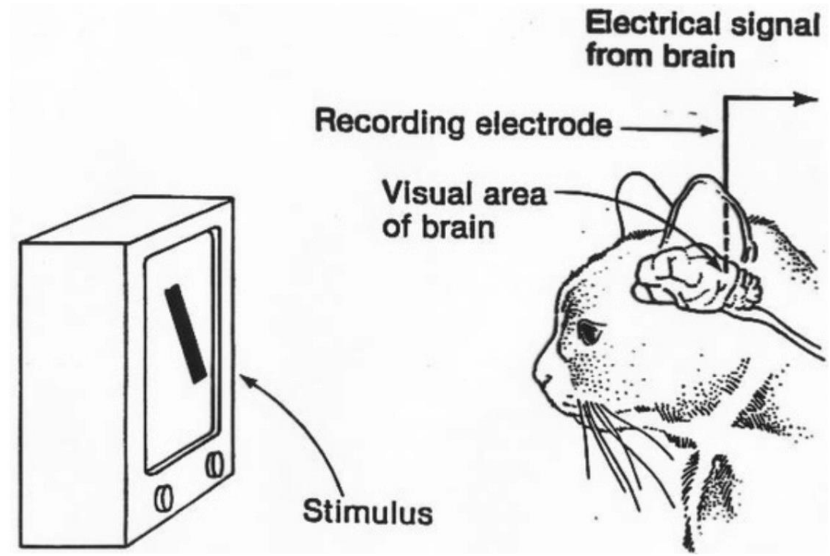
(b) 卷积层

感受野(Receptive Field)的概念

David Hunter Hubel

Torsten Wiesel

1981年诺贝尔生理学或医学奖



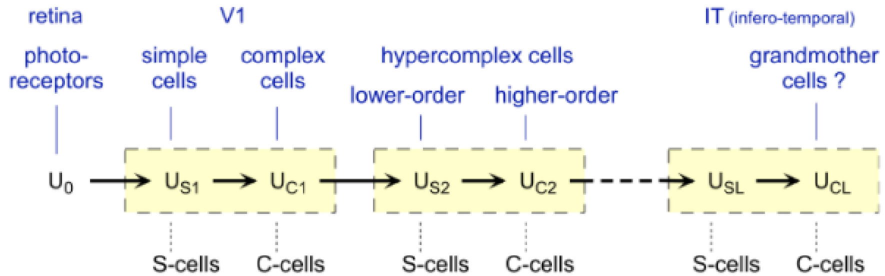
猫的神经元中的感受野

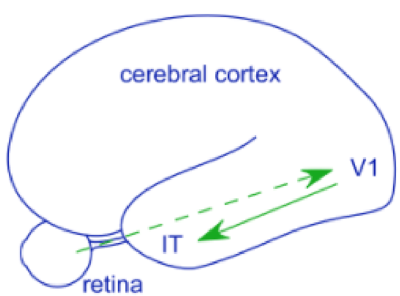
- **大脑对视觉信息的处理是分层级的，低级脑区可能处理边度、边缘的信息，高级脑区处理更抽象的信息，信息被一层一层提取出来往上传递进行处理**

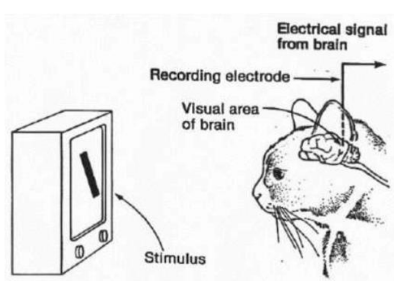
感受野(Receptive Field)的概念

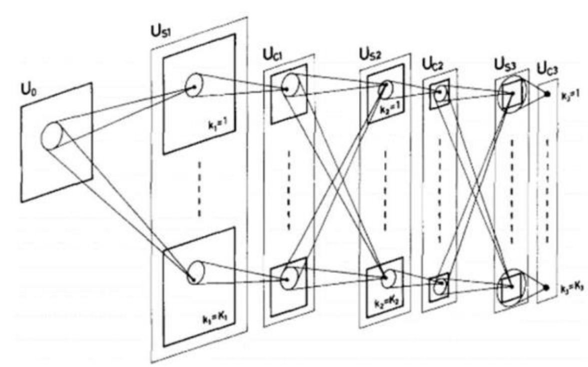
- 将脑神经科学的结构在做了计算机模拟
- 提出了现在 CNN 常用的 step-by-step 的卷积核
- **使用 ReLU 来给网络提供非线性**
- **采用平均池化来做下采样**
- **保证网络的平移不变性**
- **实现稀疏交互**

简单细胞，复杂细胞，低阶超复杂细胞，高阶超复杂细胞，祖母细胞

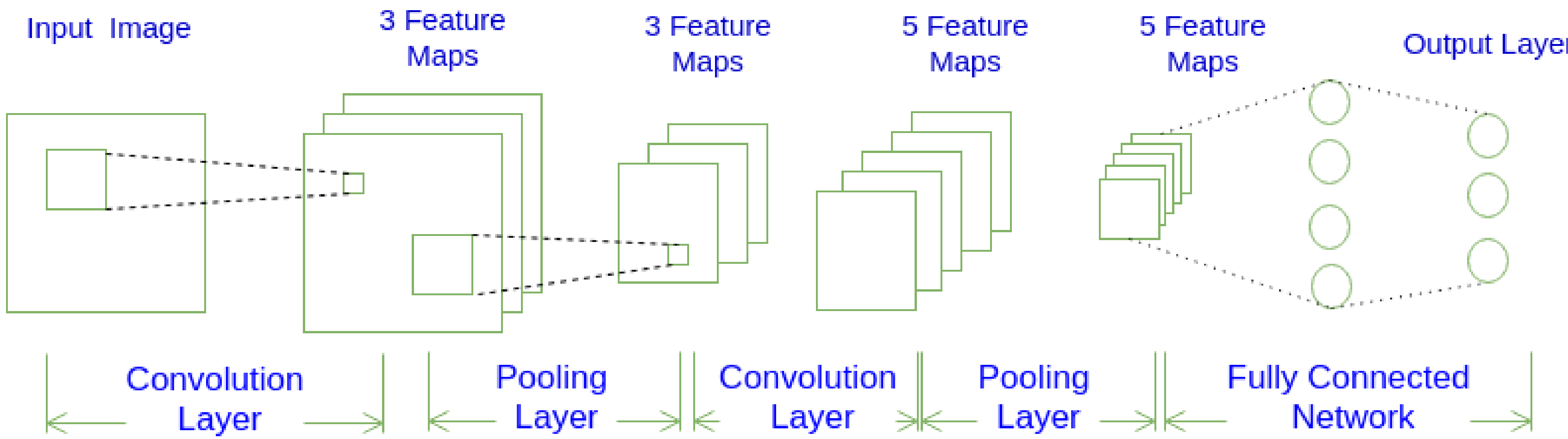








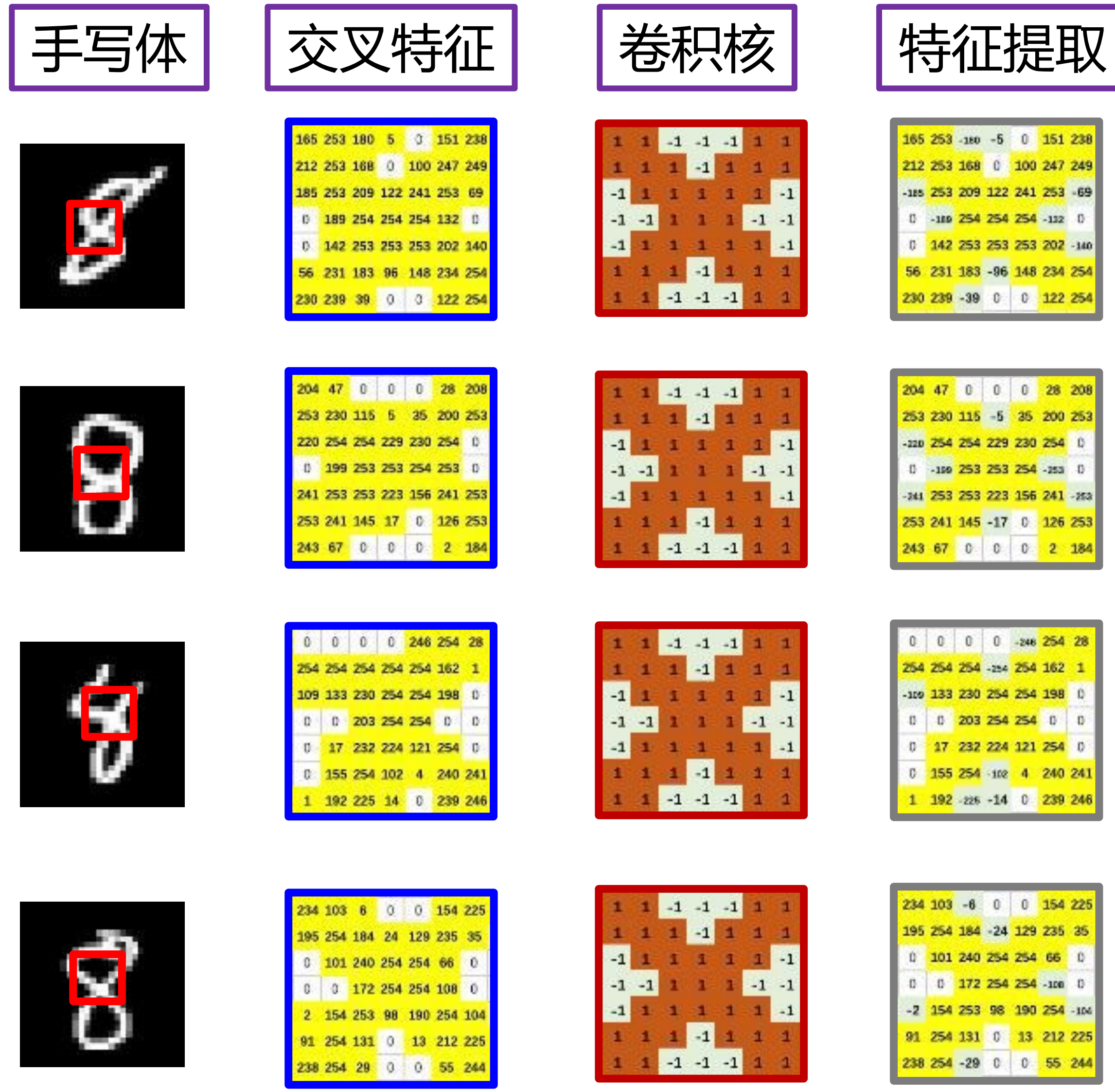
CNN的结构



输入	卷积层	池化层	卷积/池化组合	全连接网络层	输出
将原始图像进行规则化，例如统一大小	使用离散卷积计算处理提取图像特征	使用平均值或最大值将特征进行抽样降维	若干次卷积和池化的组合……	将卷积/池化后产生的特征表示映射到标记空间	产生期望的输出，例如分类结果

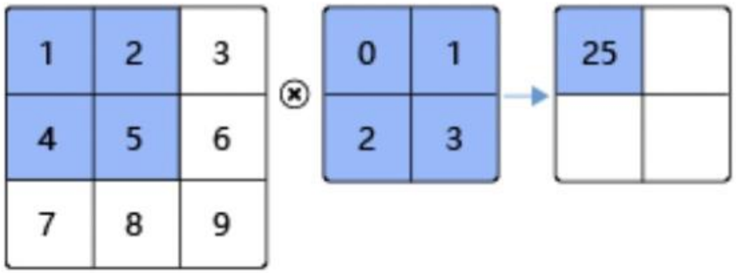
利用卷积核提取手写数字图像的特征

数字手写体

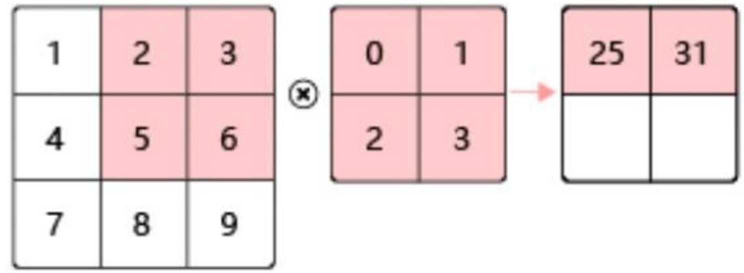


卷积操作

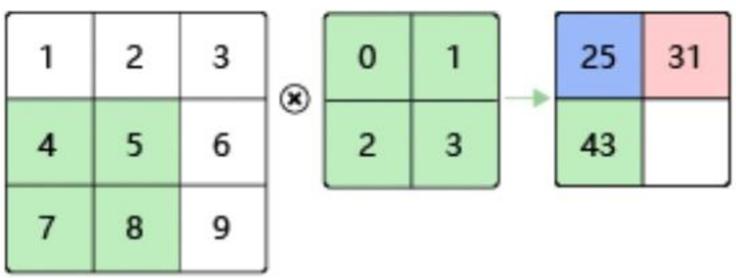
- 对像素矩阵做内积，**逐个元素相乘再求和的操作**就是卷积操作
- 卷积核的作用：**滤波/特征提取**



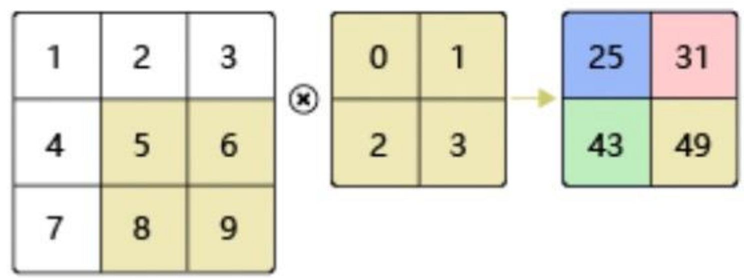
(a) 0x1+1x2+2x4+3x5=25



(b) 0x2+1x3+2x5+3x6=31



(c) 0x4+1x5+2x7+3x8=43



(d) 0x5+1x6+2x8+3x9=49

卷积操作

$$
\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 5 & 2 \\ 1 & 5 \end{bmatrix}
$$

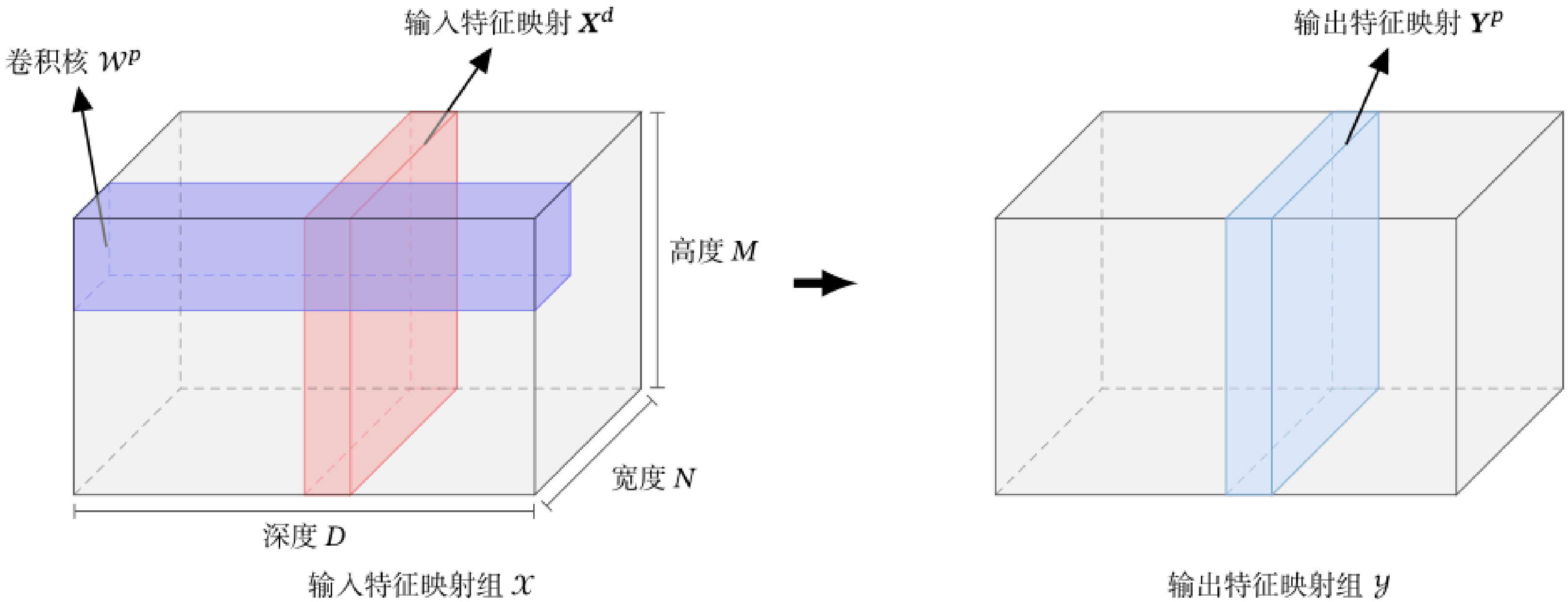
$$
\begin{bmatrix} x_{00} & x_{01} & x_{02} & x_{03} \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{20} & x_{21} & x_{22} & x_{23} \\ x_{30} & x_{31} & x_{32} & x_{33} \end{bmatrix} * \begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \\ w_{20} & w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} f_{00} & f_{01} \\ f_{10} & f_{11} \end{bmatrix}
$$

输入图像　　卷积核　　输出特征

$$
\begin{aligned}
&f_{0,0} \\
&= x_{00}w_{00} + x_{01}w_{01} \\
&+ x_{02}w_{02} + x_{10}w_{10} \\
&+ x_{11}w_{11} + x_{12}w_{12} \\
&+ x_{20}w_{20} + x_{21}w_{21} \\
&+ x_{22}w_{2,2}
\end{aligned}
$$

卷积操作

- 典型的卷积层为3维结构



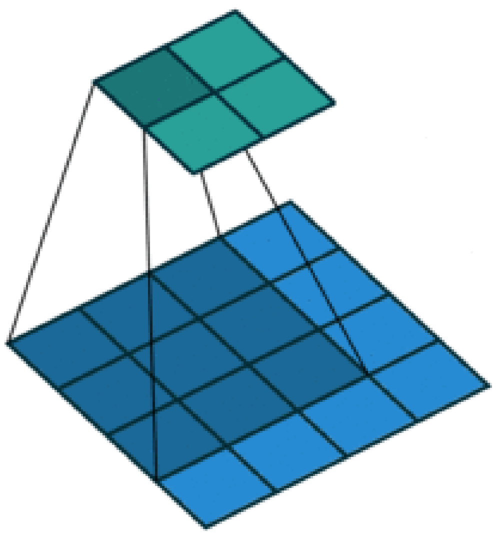
$$\boldsymbol{Z}^p = \boldsymbol{W}^p \otimes \boldsymbol{X} + b^p = \sum_{d=1}^{D} \boldsymbol{W}^{p,d} \otimes \boldsymbol{X}^d + b^p,$$

$$\boldsymbol{Y}^p = f(\boldsymbol{Z}^p).$$

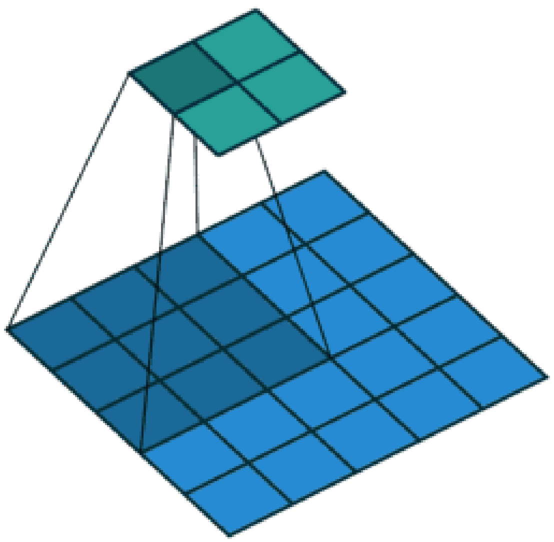
卷积步长

- 卷积步长也叫卷积步幅，英文名字是stride，代表卷积核每次移动的步幅，步长为1则卷积核每次移动1格，步长为2则卷积核每次移动2格

步长为1 (stride 1)



步长为2 (stride 2)



Padding操作

$$
\begin{bmatrix}
1 & 1 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 & 0 & 1
\end{bmatrix}
*
\begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1
\end{bmatrix}
=
\begin{bmatrix}
5 & 2 & 3 & 2 \\
1 & 5 & 2 & 3 \\
4 & 1 & 3 & 3 \\
1 & 5 & 3 & 4
\end{bmatrix}
$$

输入：6×6

卷积核大小：3×3

输出大小：$6 - 3 + 1 = 4$

4×4

- 卷积后的特征图越变越小，当卷积层数较多时，最终得到的特征图很小
- **输入矩阵边缘像素只被计算过一次，而中间像素被卷积计算多次，意味着丢失图像角落信息**
- 为解决这两个问题，**对输入图像进行 padding，即填充像素**

Padding操作

- 沿着图像边缘填充一层像素，填充之后，6×6的图像就被填充成了一个 8×8 的图像
- 用 3×3 的卷积核进行卷积计算，得到的特征图是 6×6 ，和原始输入尺寸一致
- **习惯上，用 0 进行填充**

$$
\begin{bmatrix}
0&0&0&0&0&0&0&0\\
0&1&1&1&1&0&0&0\\
0&1&1&0&1&1&0&0\\
0&1&0&1&0&0&0&0\\
0&0&1&0&1&1&1&0\\
0&0&0&1&1&0&1&0\\
0&0&1&1&1&0&1&0\\
0&0&0&0&0&0&0&0
\end{bmatrix}
*
\begin{bmatrix}
1&0&1\\
0&1&0\\
1&0&1
\end{bmatrix}
=
\begin{bmatrix}
2&2&3&2&1&1\\
2&5&2&3&2&0\\
2&1&5&2&3&2\\
0&4&1&3&3&1\\
2&1&5&3&4&2\\
0&1&2&2&2&1
\end{bmatrix}
$$

卷积核大小: $3*3$

输入：6×6 ，填充 $p=1$ ，1表示1层

输出大小：$6+2-3+1=6$

6×6

卷积计算输出大小计算公式

0	0	0	0	0	0	0
0	1	1	1	1	0	0
0	1	1	0	1	1	0
0	1	0	1	0	0	0
0	0	1	0	1	1	0
0	0	0	1	1	0	0
0	0	0	0	0	0	0

输入：5×5，填充 $p = 1$

*

1	0	1
0	1	0
1	0	1

卷积核大小：3×3

步长：2

=

2	2	1
3	5	2
1	3	1

输出大小：$\frac{5+2\times1-3}{2} + 1 = 3$

3×3

- 假设输入大小为 $n \times n$，卷积核大小为 $f \times f$，填充层数为 p，步长为 s，则输出大小为：

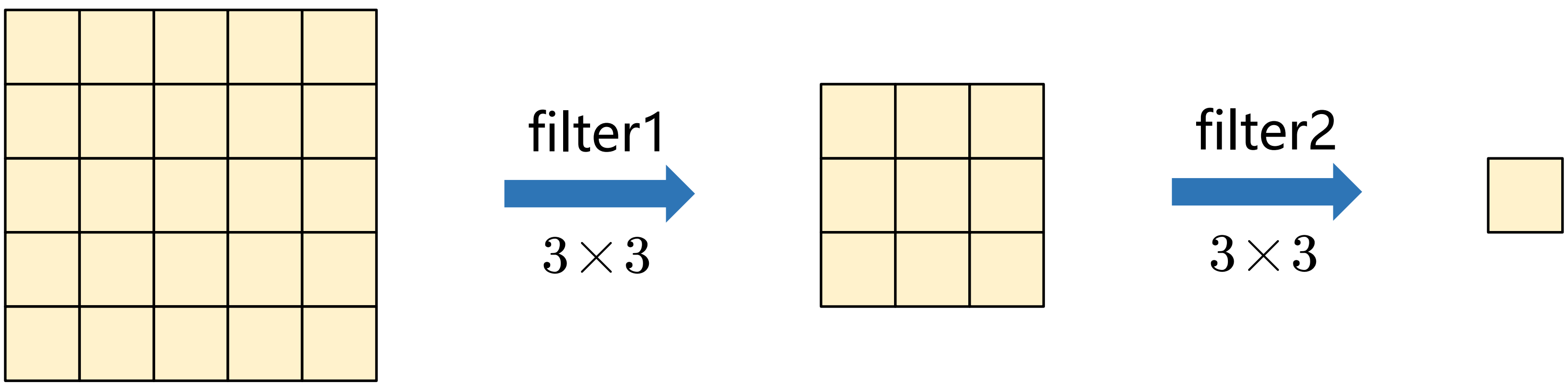
$$Output\ Size = \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

感受野大小计算

- **感受野(Receptive Field)的定义：**卷积神经网络每一层输出的特征图(feature map)上的像素点映射回输入图像上的区域大小。通俗点的解释是，特征图上一点，相对于原图的大小，也是卷积神经网络特征所能看到输入图像的区域

感受野大小计算

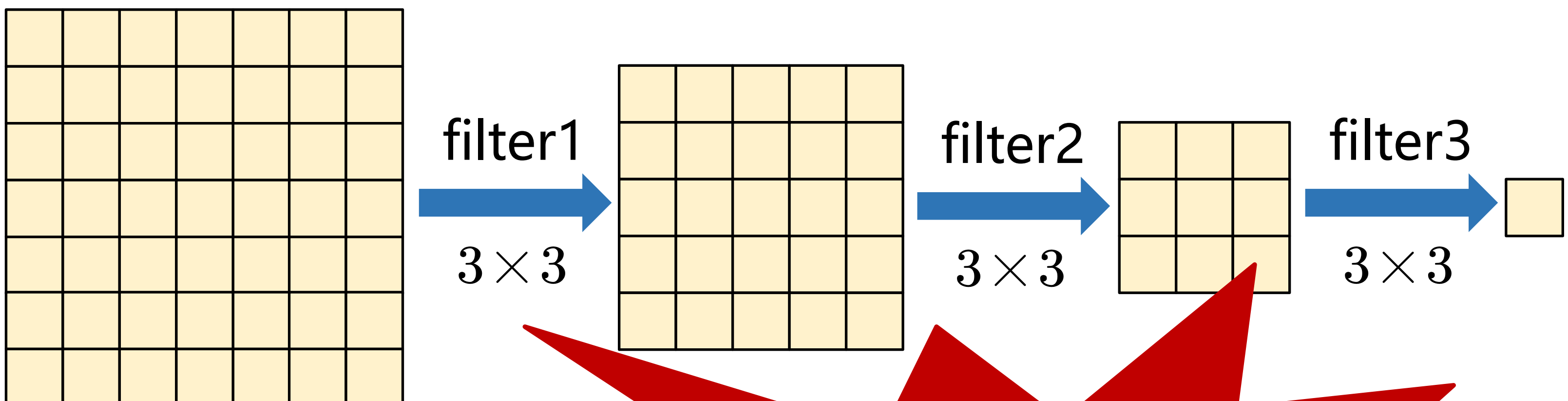
- 若输入图像的尺寸大小是 5×5，经过两次 3×3 的卷积 （其中stride=1，padding=0）后，其感受野大小为 5×5



根据卷积计算公式：$N=(W-F+2P)/S+1$ ，得到第一次卷积后的图像大小为 3×3 ，第二次卷积后的图像大小为 1×1

感受野大小计算

- 若输入图像的尺寸大小是 7×7，经过三次 3×3 的卷积（其中stride=1，padding=0）后，其感受野大小为 7×7



根据卷积计算公式：… 的第
一次卷积后的图像大小为 … 像大
小为 3×3，第三次卷积后的 …

随着卷积核的增多（即网络的加深），感受野会越来越大

感受野大小计算

- **感受野大小的计算方式是从最后一层feature map开始，往下往上的计算方法**，即先计算最深层在前一层上的感受野，然后以此类推逐层传递到第一层
- 感受野大小的计算不考虑padding的大小
- 最后一层的特征图感受野的大小等于其卷积核的大小
- 第i层特征图的感受野大小和第i层的卷积核大小和步长有关系，同时也与第$(i+1)$层特征图的感受野大小有关。**（为什么？）**

□感受野大小计算公式

$$RF_i = (RF_{i+1} - 1) \times stride_i + K_{size_i}$$

其中，RF_i 表示 i 层感受野大小，i 表示当前特征层的层数，$stirde$ 是卷积的步长，K_{size} 是本层卷积核的大小。

感受野大小计算

Layer Type	Kernel Size	Stride
Input		
Conv1	3*3	1
Pool1	2*2	2
Conv2	3*3	1
Pool2	2*2	2
Conv3	3*3	1
Conv4	3*3	1
Pool3	2*2	2

从最后一层的Pool3池化层开始计算感受野：

pool3：RF=2（最后一层池化层输出特征图的感受野大小等于卷积核的大小）

conv4：RF=（2-1）*1+3=4

conv3：RF=（4-1）*1+3=6

pool2：RF=（6-1）*2+2=12

conv2：RF=（12-1）*1+3=14

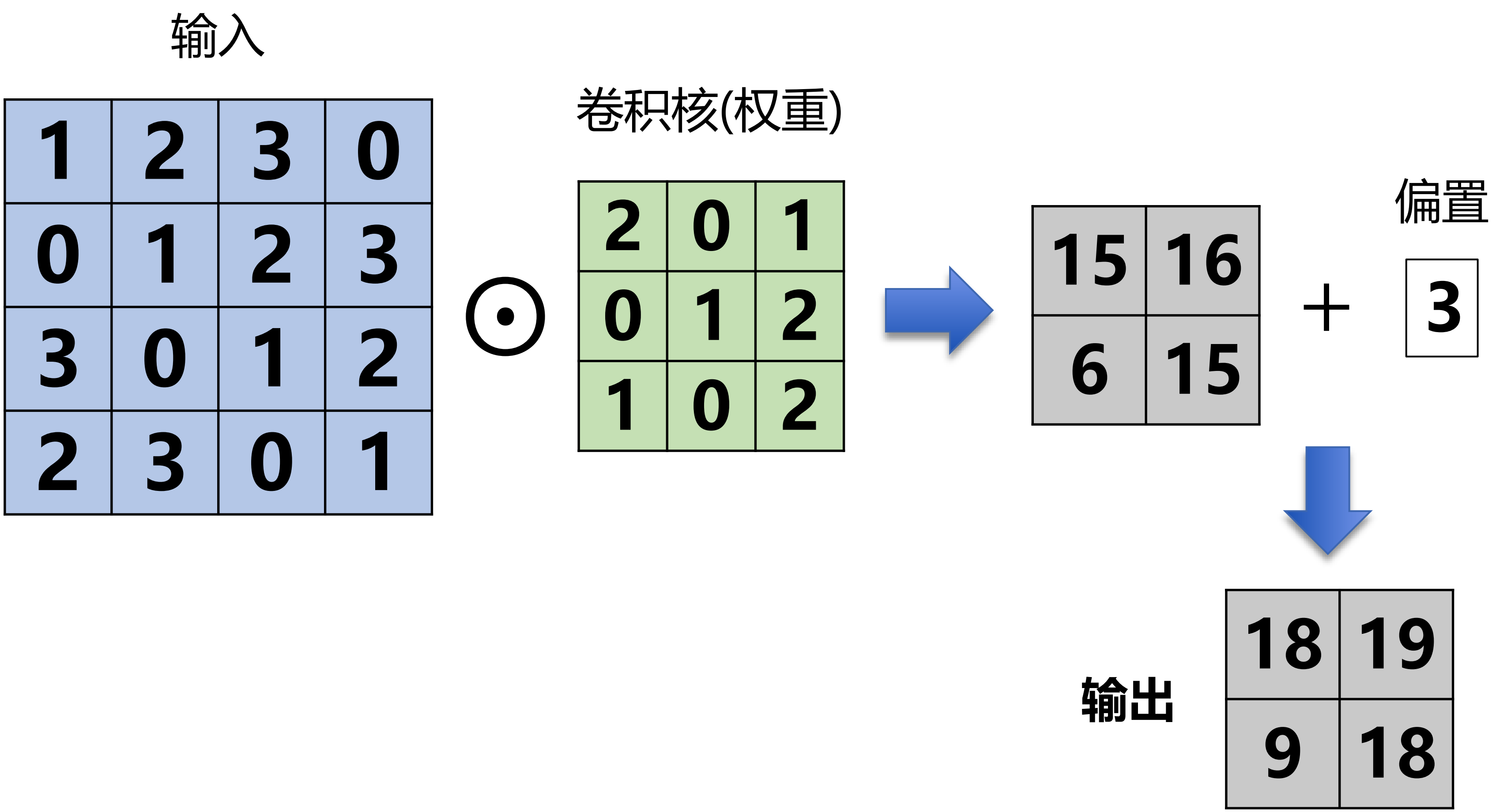
pool1：RF=（14-1）*2+2=28

conv1：RF=（28-1）*1+3=30

因此，pool3输出的特征图在输入图片上的感受野为30*30

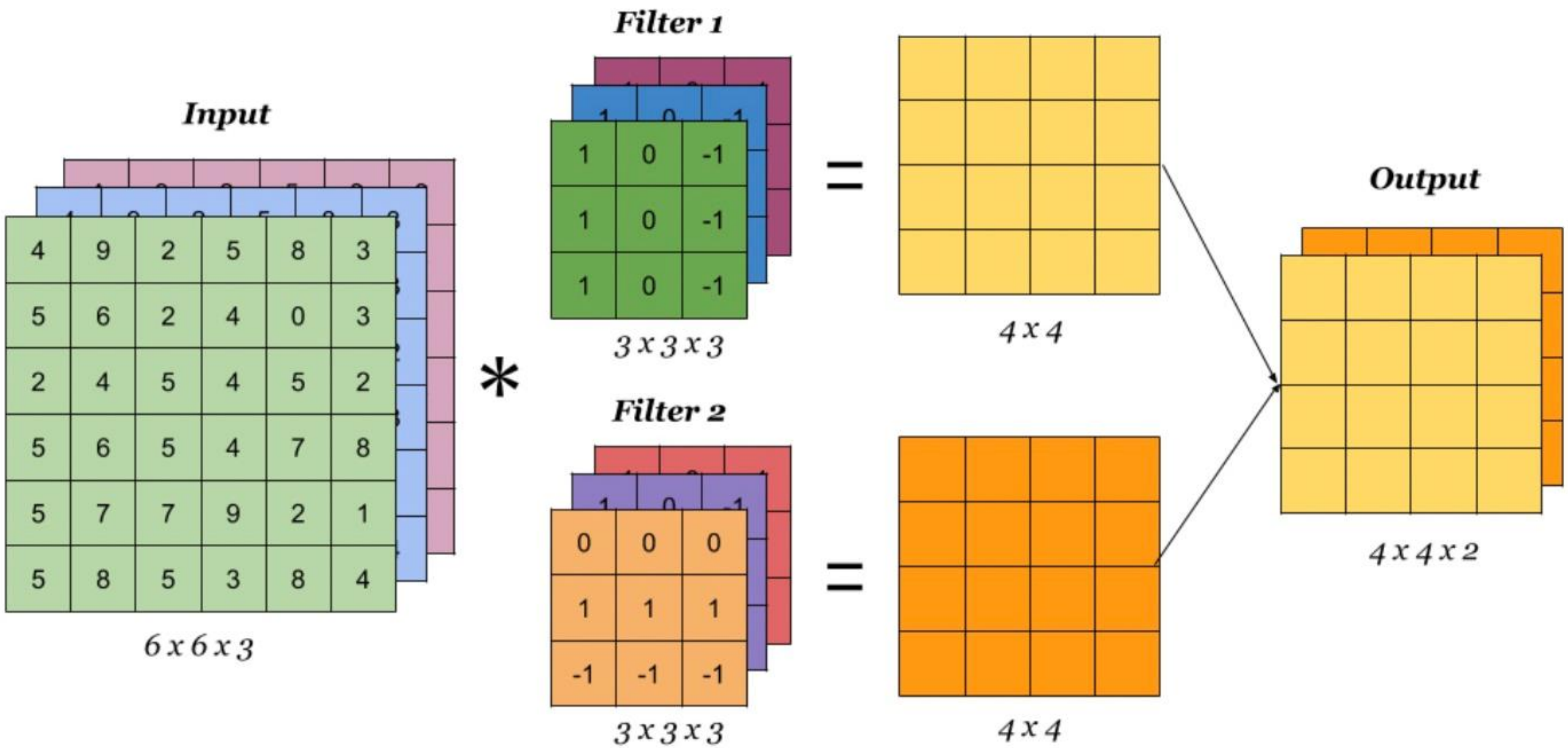
卷积核偏置

- 偏置的存在是为了更好的拟合数据，每个卷积核有一个偏置



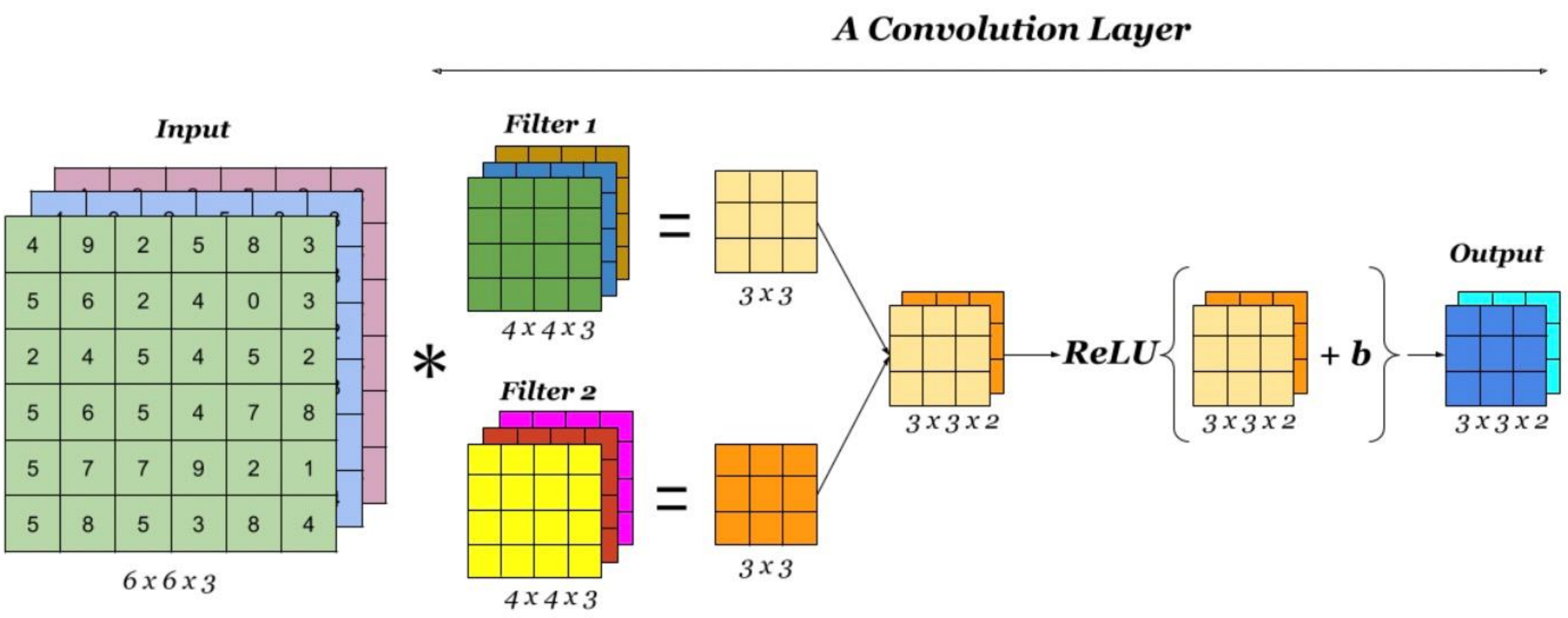
使用多个卷积核提取不同特征

- 使用卷积从一个图像中提取多个特征时，可以使用多个卷积核而不是仅使用一个，但是**所有卷积核的大小必须相同**



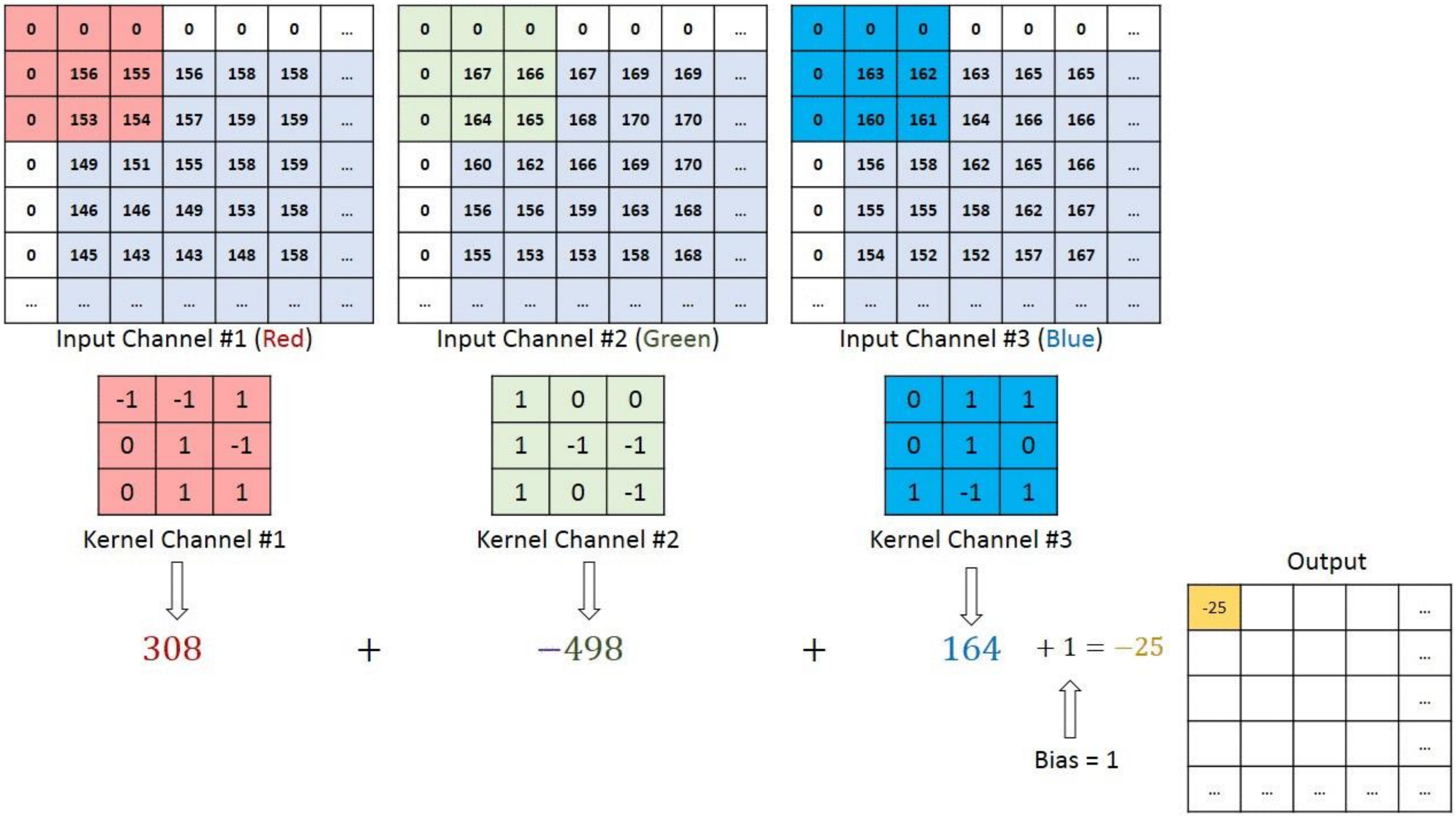
卷积层激活函数

- 激活函数是卷积层的最后一个组成部分，可增加输出中的非线性



RGB图像的卷积操作

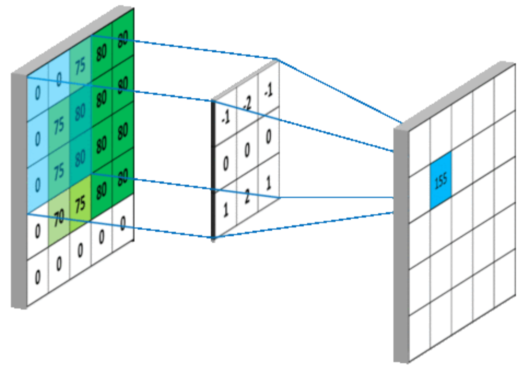
- 每层特征图与相应的卷积核进行卷积计算，将每层卷积结果进行加和，经过卷积运算后，三层特征图变成了一层

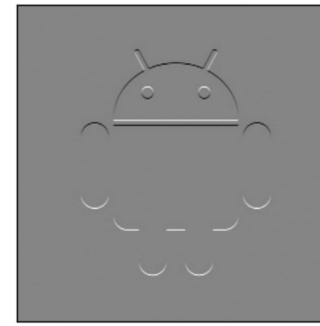


不同卷积核的效果

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

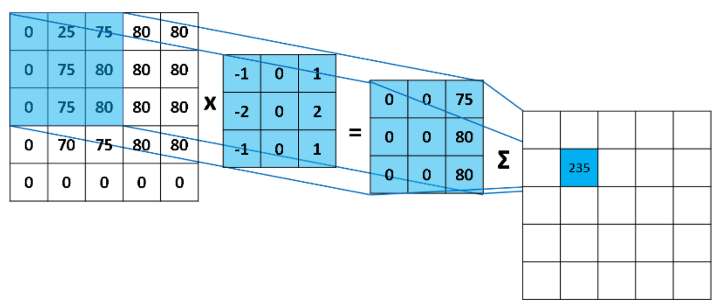
发现水平直线

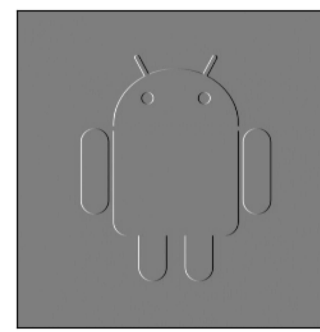




-1	0	1
-2	0	2
-1	0	1

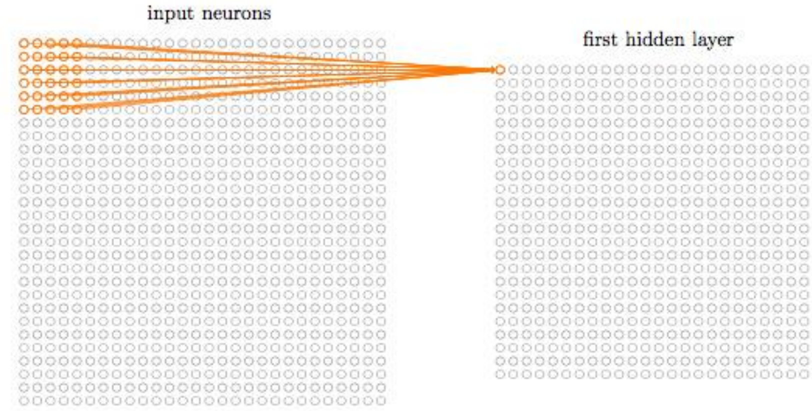
发现垂直直线



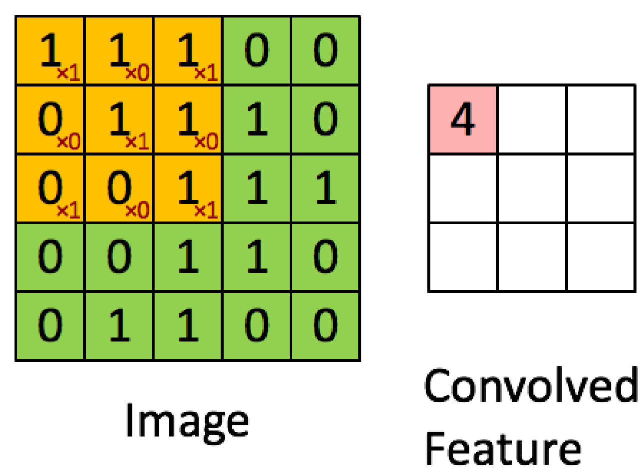


总结：卷积层的特点

- **局部连接：** 在卷积层中每一个神经元都只与前一层中某个局部窗口内的神经元相连，构成一个局部连接网络，卷积层和前一层之间的连接数大大减少



- **权重共享：** 卷积核 w 对于第 i 层的所有神经元都是相同的，权重共享可以理解为一个卷积核只捕捉输入数据中的一个特定的局部特征，因此如果要提取多种特征就要使用多种不同的卷积核



现实世界的图像

拍摄的车牌

平移

旋转

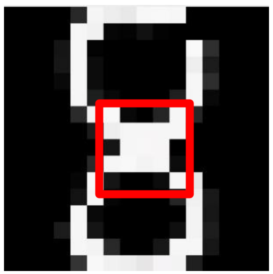
遮挡

扭曲变形

尺度改变

如何突出卷积后的数字特征？

手写体　交叉特征　卷积核　卷积结果

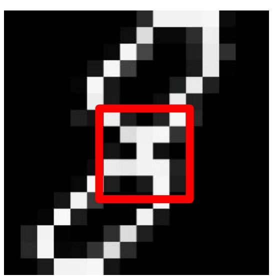


241	0	0	0	241
229	241	242	242	243
0	240	242	242	0
242	242	237	241	0
0	0	0	0	242

1	-1	-1	-1	1
-1	1	-1	1	-1
-1	-1	1	-1	-1
-1	1	-1	1	-1
1	-1	-1	-1	1

241	0	0	0	241
-229	241	-242	242	-243
0	-240	242	-242	0
-242	242	-237	241	0
0	0	0	0	242

结果：646



230	0	0	0	240
0	205	232	240	27
0	0	236	0	0
225	219	238	237	0
0	0	0	239	0

1	-1	-1	-1	1
-1	1	-1	1	-1
-1	-1	1	-1	-1
-1	1	-1	1	-1
1	-1	-1	-1	1

230	0	0	0	240
0	205	-232	240	-27
0	0	236	0	0
-225	219	-238	237	0
0	0	0	-239	0

结果：257

计算机如何突出千奇百怪的数字的特征？

池化后的图像

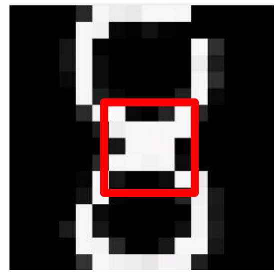
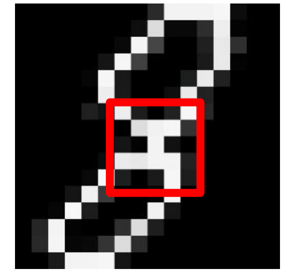
手写体

↓

池化操作

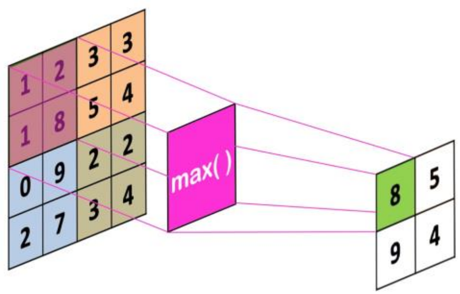
↓

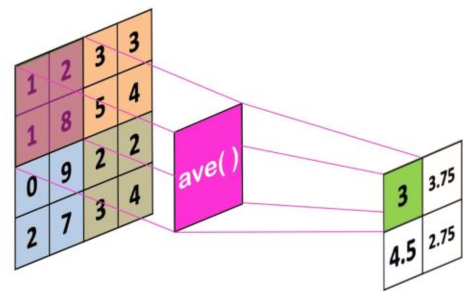
输出图像

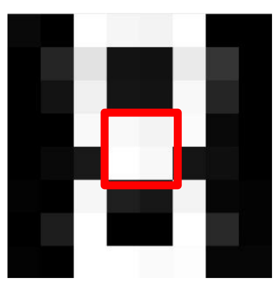
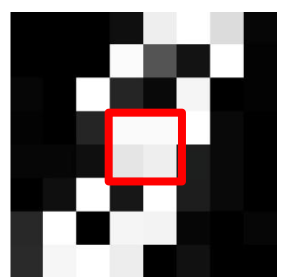
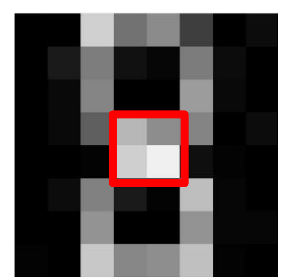
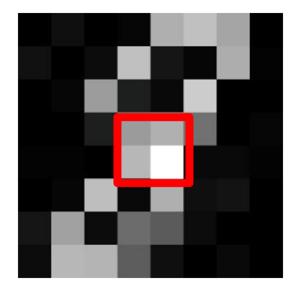



最大池化

平均池化

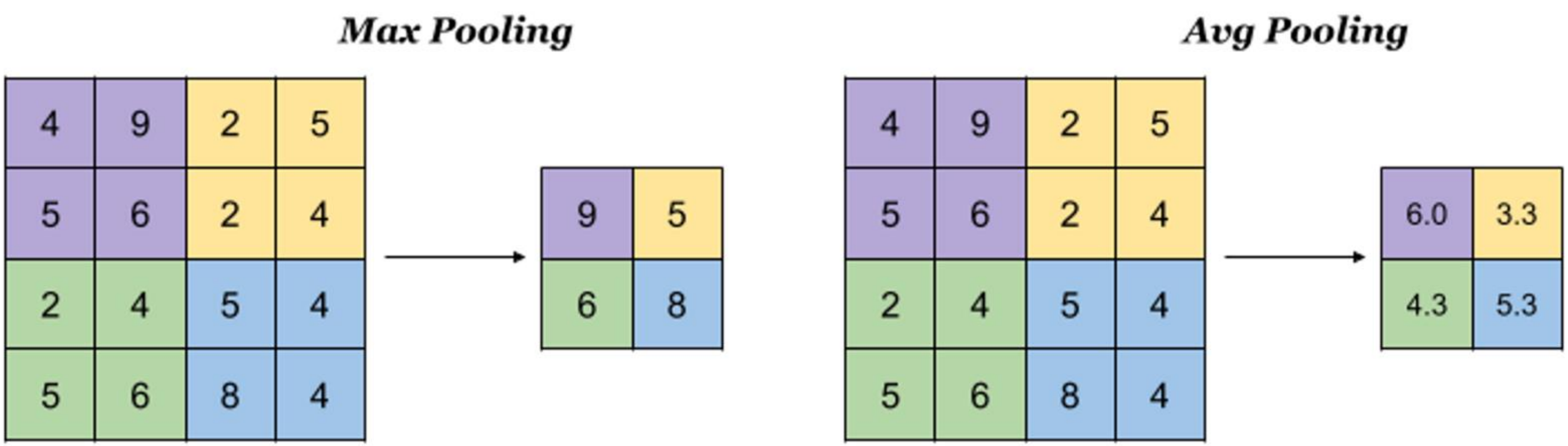




池化操作

- **池化层的本质是降采样**
- 最大池化通过获取邻域内特征的最大值来实现，**能够抑制网络参数误差造成估计均值偏移的现象，特点是更好的提取纹理信息**
- 平均池化通过邻域内特征数值求平均来实现，**能够抑制由于邻域大小受限造成估计值方差增大的现象，特点是对背景的保留效果更好**



池化后的图像

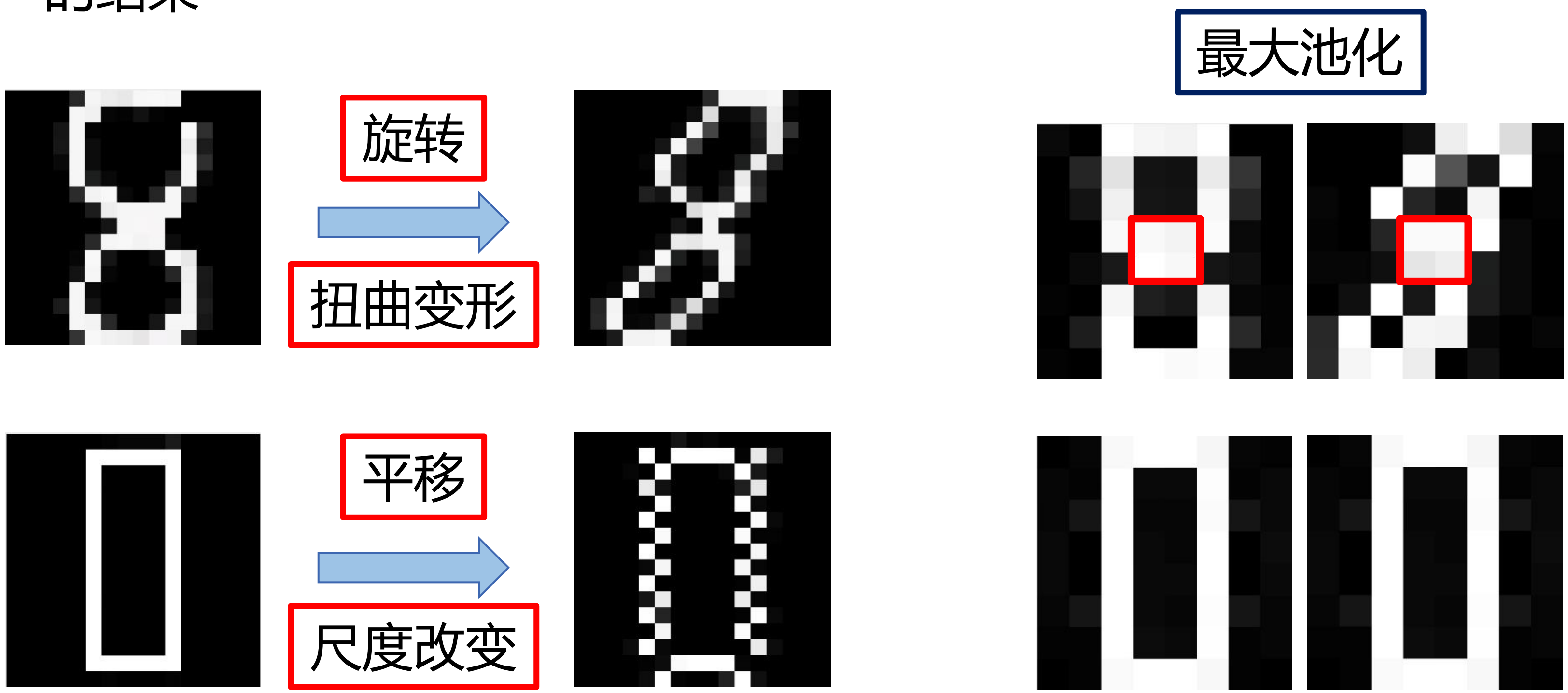
原图

最大池化

平均池化

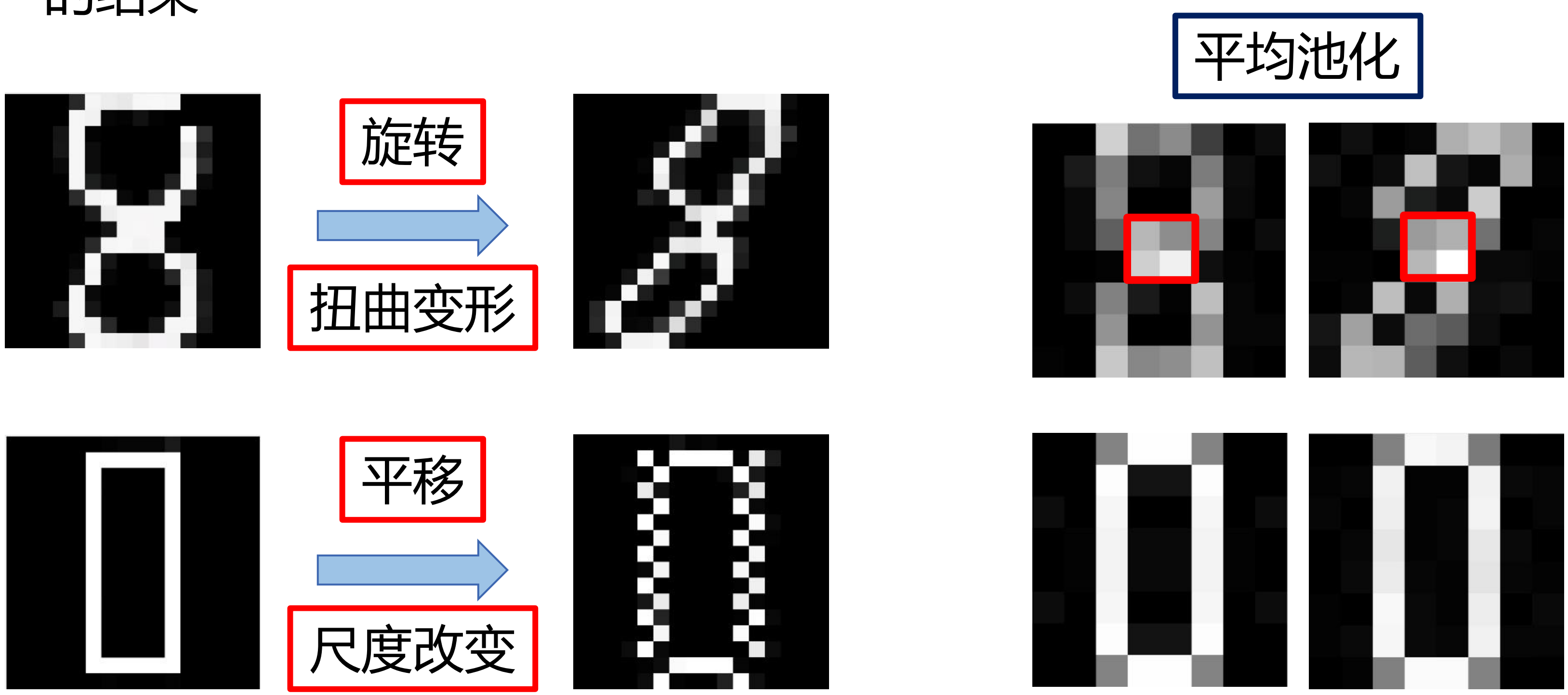
池化的特点

- 池化只是从目标区域中取最大值或者平均值，所以没有要学习的参数
- **通道数不发生改变，即不改变特征图的数量**
- 利用图像局部相关性的原理，**对图像进行下采样，这样对微小的位置变化具有鲁棒性**，当输入的数据发生微小偏差时，池化仍会返回相同的结果



池化的特点

- 池化只是从目标区域中取最大值或者平均值，所以没有要学习的参数
- **通道数不发生改变，即不改变特征图的数量**
- 利用图像局部相关性的原理，**对图像进行下采样，这样对微小的位置变化具有鲁棒性**，当输入的数据发生微小偏差时，池化仍会返回相同的结果



池化层步长

- 每次移动的步幅，步长为 1 则每次移动 1 格，步长为 2 则每次移动 2 格

1	0	1	0
2	1	3	2
1	2	0	4
2	2	3	0

2	3	3
2	3	4
2	3	4

池化操作窗口：2 × 2，步长为1

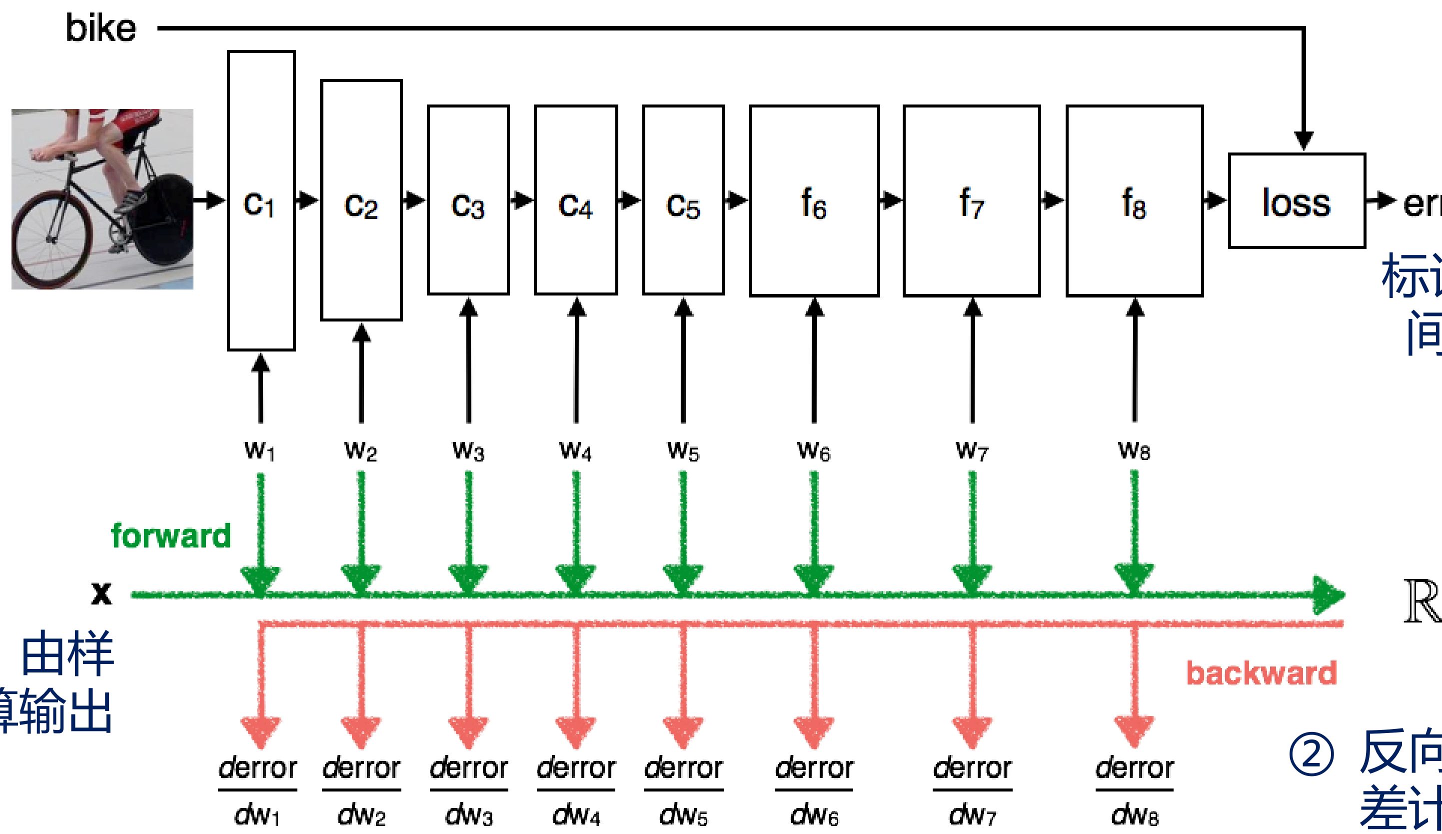
1	0	1	0
2	1	3	2
1	2	0	4
2	2	3	0

2	3
2	3

池化操作窗口：2 × 2 ，步长为2

CNN模型训练

标记样本

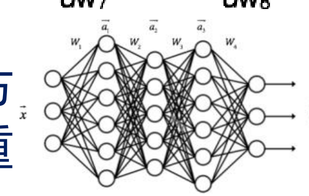


标记与输出间的误差

① 正向：由样本计算输出

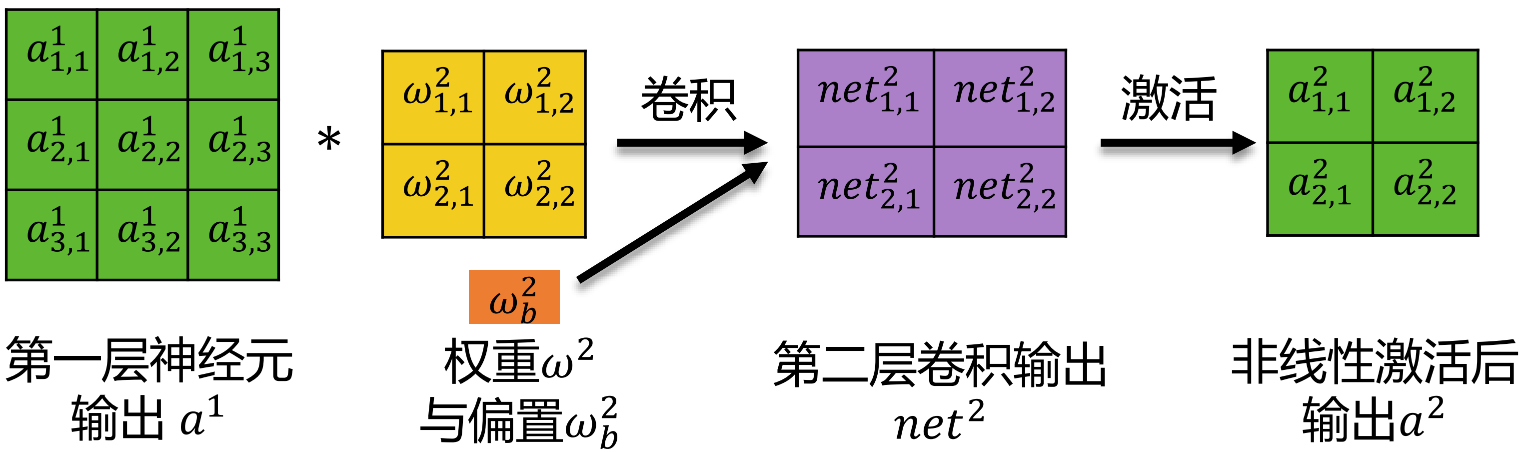
② 反向：由误差计算残差

③ 根据误差更新节点间连接的权重



④ 多轮迭代

卷积网络的梯度反向传播



卷积运算　对输入进行卷积操作得到第一层的卷积输出

$$net^2 = conv(\omega^2, a^1) + \omega^2_b$$

$$net^2_{1,1} = \omega^2_{1,1}a^1_{1,1} + \omega^2_{1,2}a^1_{1,2} + \omega^2_{2,1}a^1_{2,1} + \omega^2_{2,2}a^1_{2,2} + \omega^2_b$$
$$net^2_{1,2} = \omega^2_{1,1}a^1_{1,2} + \omega^2_{1,2}a^1_{1,3} + \omega^2_{2,1}a^1_{2,2} + \omega^2_{2,2}a^1_{2,3} + \omega^2_b$$
$$net^2_{2,1} = \omega^2_{1,1}a^1_{2,1} + \omega^2_{1,2}a^1_{2,2} + \omega^2_{2,1}a^1_{3,1} + \omega^2_{2,2}a^1_{3,2} + \omega^2_b$$
$$net^2_{2,2} = \omega^2_{1,1}a^1_{2,2} + \omega^2_{1,2}a^1_{2,3} + \omega^2_{2,1}a^1_{3,2} + \omega^2_{2,2}a^1_{3,3} + \omega^2_b$$

a^2 表示对第一层的神经元 net^2 进行非线性激活，

f 为激活函数

$$a^2 = f(net^2)$$

Refer: https://www.zybuluo.com/hanbingtao/note/485480

卷积网络的梯度反向传播

反向传播

考察权重$\omega_{i,j}$对损失函数E_d的影响，即计算$\frac{\partial E_d}{\partial \omega_{i,j}}$，

以计算对$\omega_{1,1}^2$的偏导为例

$$\frac{\partial E_d}{\partial \omega_{1,1}^2} = \frac{\partial E_d}{\partial net^2} \frac{\partial net^2}{\partial \omega_{1,1}^2}$$

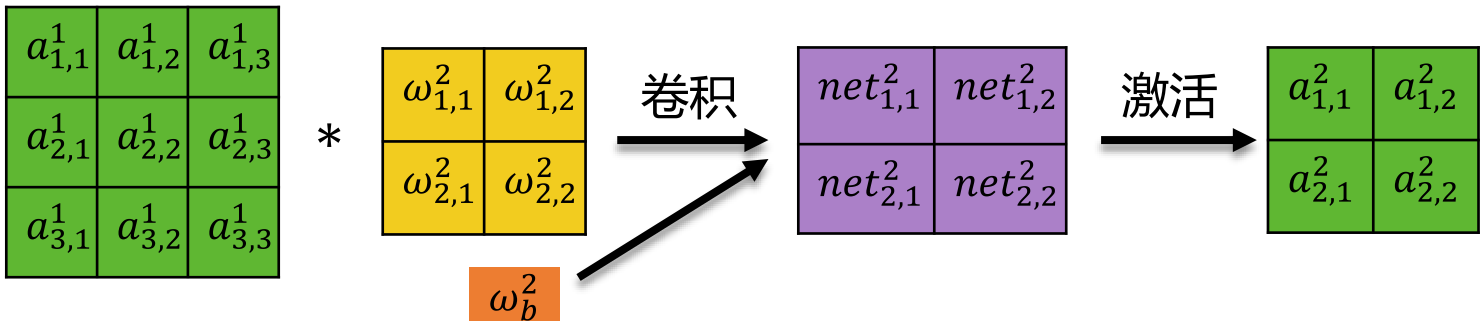
根据全导数公式，计算$\frac{\partial E_d}{\partial \omega_{1,1}^2}$需要把每个偏导数都加起来

$$\frac{\partial E_d}{\partial net_{1,1}^2} \boxed{\frac{\partial net_{1,1}^2}{\partial \omega_{1,1}^2}} + \frac{\partial E_d}{\partial net_{1,2}^2} \boxed{\frac{\partial net_{1,2}^2}{\partial \omega_{1,1}^2}} + \frac{\partial E_d}{\partial net_{2,1}^2} \boxed{\frac{\partial net_{2,1}^2}{\partial \omega_{1,1}^2}} + \frac{\partial E_d}{\partial net_{2,2}^2} \boxed{\frac{\partial net_{2,2}^2}{\partial \omega_{1,1}^2}}$$

$$= \frac{\partial E_d}{\partial net_{1,1}^2} a_{1,1}^1 + \frac{\partial E_d}{\partial net_{1,2}^2} a_{1,2}^1 + \frac{\partial E_d}{\partial net_{2,1}^2} a_{2,1}^1 + \frac{\partial E_d}{\partial net_{2,2}^2} a_{2,2}^1$$

Refer: https://www.zybuluo.com/hanbingtao/note/485480

卷积网络的梯度反向传播



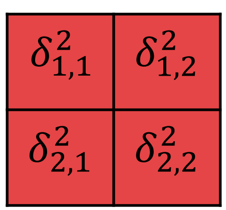
第一层神经元输出 a^1

权重ω^2与偏置ω^2_b

第二层卷积输出 net^2

非线性激活后输出a^2

定义损失函数对卷积层输出的偏导数为 $\delta^2_{i,j}$（$\delta^2_{1,1}$，$\delta^2_{1,2}$，$\delta^2_{2,1}$，$\delta^2_{2,2}$），也被称为误差敏感项



sensitivity map
2*2

$$\delta^2_{i,j} = \frac{\partial E_d}{\partial net^2_{i,j}} = \frac{\partial E_d}{\partial a^2_{i,j}} \frac{\partial a^2_{i,j}}{\partial net^2_{i,j}}$$

Refer: https://www.zybuluo.com/hanbingtao/note/485480

卷积网络的梯度反向传播

反向传播

$$\begin{bmatrix} \delta^2_{1,1} & \delta^2_{1,2} \\ \delta^2_{2,1} & \delta^2_{2,2} \end{bmatrix} \rightarrow \begin{bmatrix} \frac{\partial E_d}{\partial \omega^2_{1,1}} & \frac{\partial E_d}{\partial \omega^2_{1,2}} \\ \frac{\partial E_d}{\partial \omega^2_{2,1}} & \frac{\partial E_d}{\partial \omega^2_{2,2}} \end{bmatrix}$$

$$\frac{\partial E_d}{\partial \omega^2_{1,1}} = \delta^2_{1,1} a^1_{1,1} + \delta^2_{1,2} a^1_{1,2} + \delta^2_{2,1} a^1_{2,1} + \delta^2_{2,2} a^1_{2,2}$$

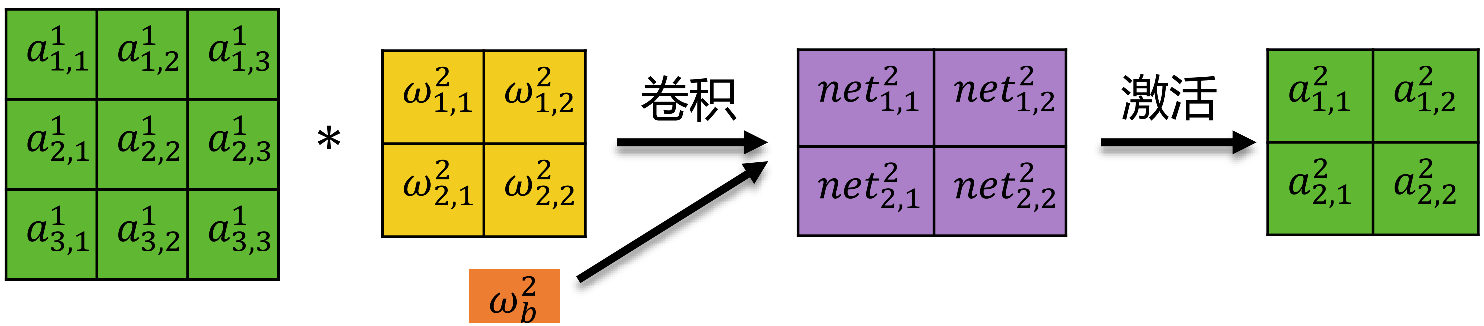
$$\frac{\partial E_d}{\partial \omega^2_{1,2}} = \delta^2_{1,1} a^1_{1,2} + \delta^2_{1,2} a^1_{1,3} + \delta^2_{2,1} a^1_{2,2} + \delta^2_{2,2} a^1_{2,3}$$

$$\frac{\partial E_d}{\partial \omega^2_{2,1}} = \delta^2_{1,1} a^1_{2,1} + \delta^2_{1,2} a^1_{2,2} + \delta^2_{2,1} a^1_{3,1} + \delta^2_{2,2} a^1_{3,2}$$

$$\frac{\partial E_d}{\partial \omega^2_{2,2}} = \delta^2_{1,1} a^1_{2,2} + \delta^2_{1,2} a^1_{2,3} + \delta^2_{2,1} a^1_{3,2} + \delta^2_{2,2} a^1_{3,3}$$

由此，可以推出各权重的梯度计算公式为 $\frac{\partial E_d}{\partial \omega^2_{m,n}} = \sum_i \sum_j \delta^2_{i,j} a^1_{i+m-1,j+n-1}$

卷积网络的梯度反向传播



求对偏置项 ω^2_b 偏导

$$\frac{\partial E_d}{\partial \omega^2_b} = \frac{\partial E_d}{\partial net^2_{1,1}}\frac{\partial net^2_{1,1}}{\partial \omega^2_b} + \frac{\partial E_d}{\partial net^2_{1,2}}\frac{\partial net^2_{1,2}}{\partial \omega^2_b} + \frac{\partial E_d}{\partial net^2_{2,1}}\frac{\partial net^2_{2,1}}{\partial \omega^2_b} + \frac{\partial E_d}{\partial net^2_{2,2}}\frac{\partial net^2_{2,2}}{\partial \omega^2_b}$$

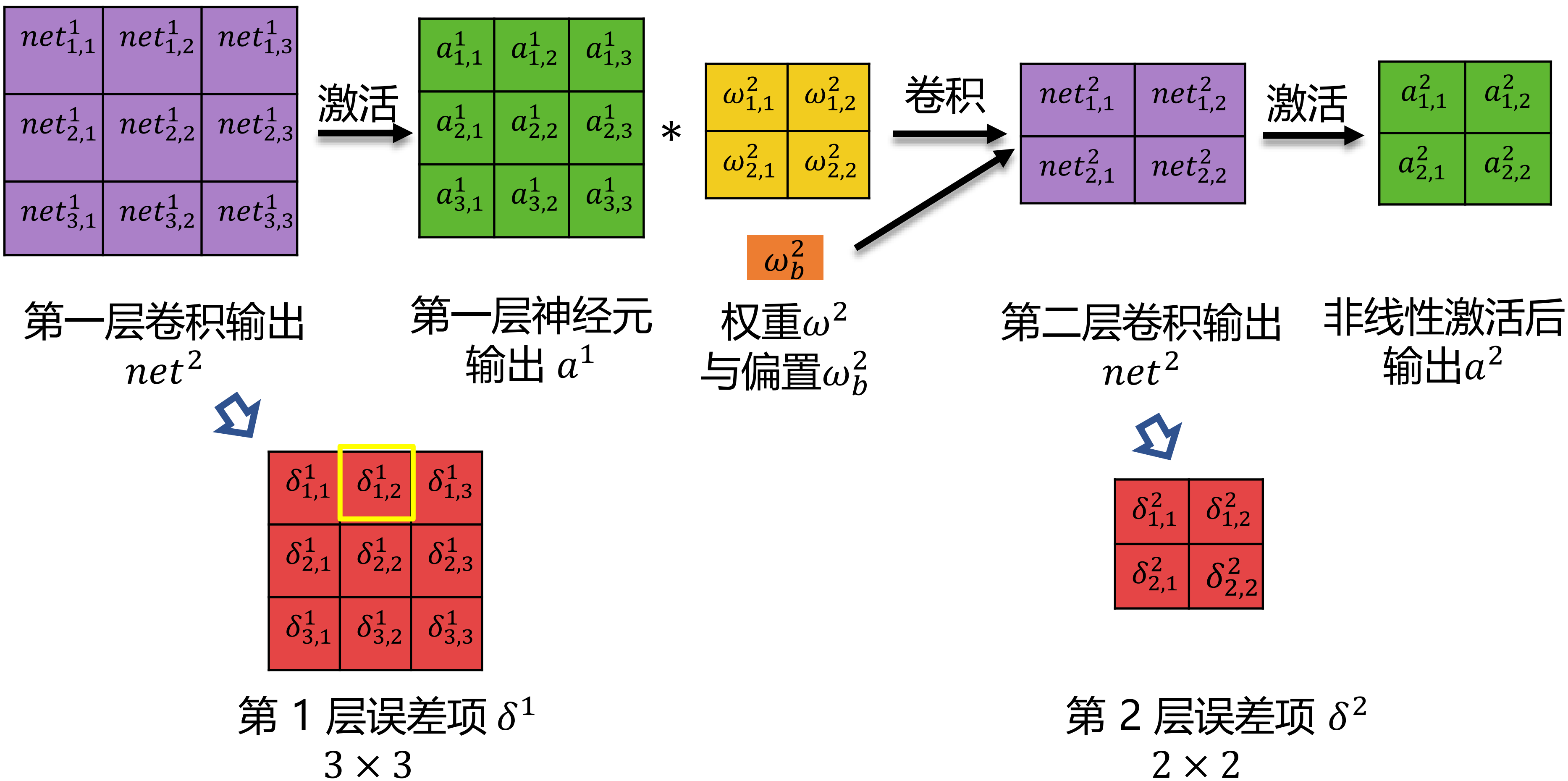
$$= \delta^2_{1,1} + \delta^2_{1,2} + \delta^2_{2,1} + \delta^2_{2,2}$$

$$= \sum_i \sum_j \delta^2_{i,j}$$

可以得到对偏置项求偏导就是 sensitivity map 所有误差敏感项之和

Refer：*https://www.zybuluo.com/hanbingtao/note/485480*

卷积网络的梯度反向传播



已知第 2 层误差敏感项 δ^2，计算第 1 层误差敏感项 δ^1 的传播

卷积网络的梯度反向传播

以第1层第1行第2列神经元的误差敏感项
$\delta_{1,2}^1$的传播为例：

$$\delta_{1,2}^1=\frac{\partial E_d}{\partial net_{1,2}^1}=\frac{\partial E_d}{\partial a_{1,2}^1}\frac{\partial a_{1,2}^1}{\partial net_{1,2}^1}$$

其中，$$\frac{\partial E_d}{\partial a_{1,2}^1}=\frac{\partial E_d}{\partial net_{1,1}^2}\frac{\partial net_{1,1}^2}{\partial a_{1,2}^1}+\frac{\partial E_d}{\partial net_{1,2}^2}\frac{\partial net_{1,2}^2}{\partial a_{1,2}^1}$$

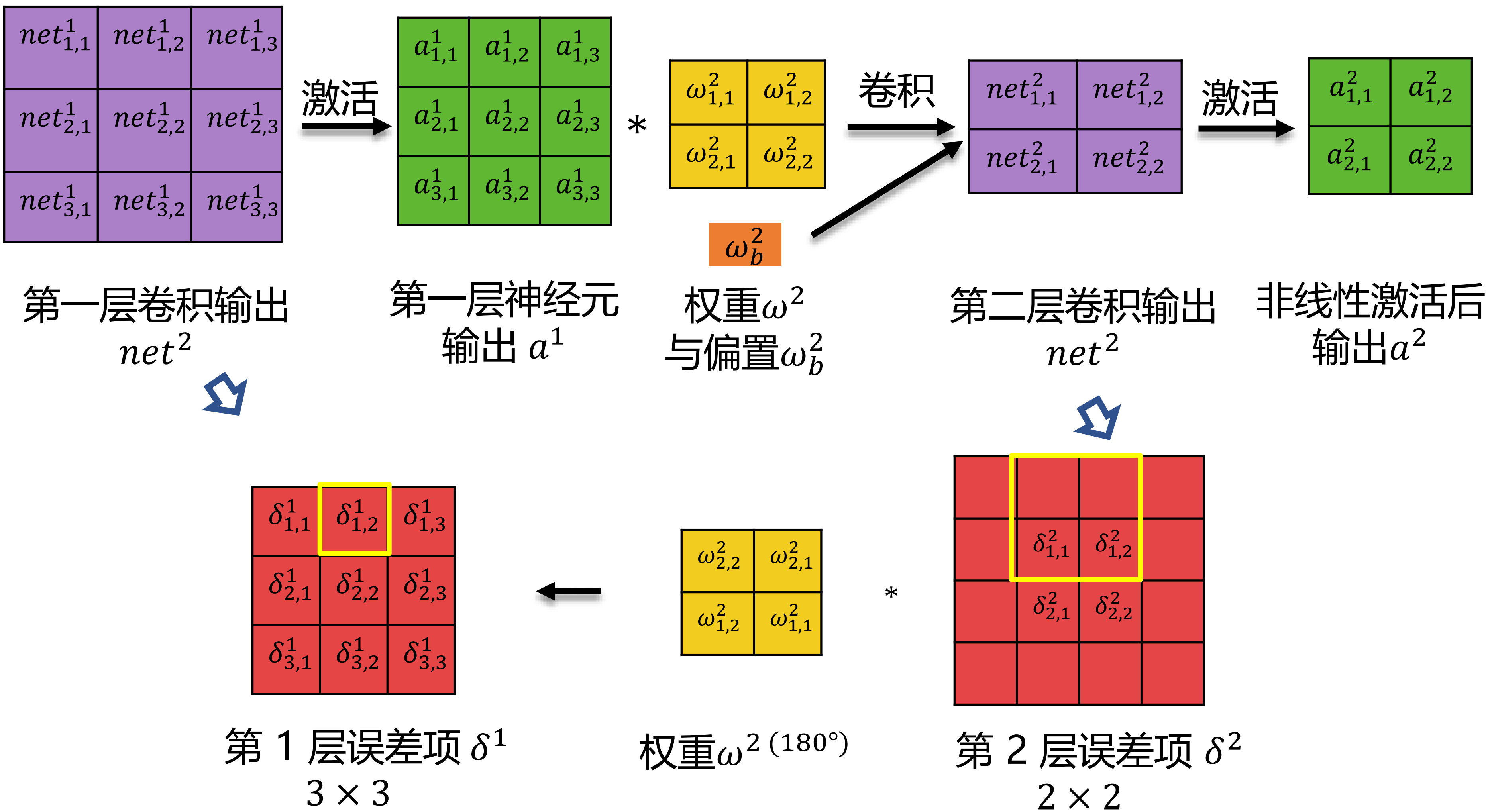
$$=\delta_{1,1}^2\omega_{1,2}^2+\delta_{1,2}^2\omega_{1,1}^2$$

$$\frac{\partial a_{1,2}^1}{\partial net_{1,2}^1}=f'(net_{1,2}^1)$$

所以得到

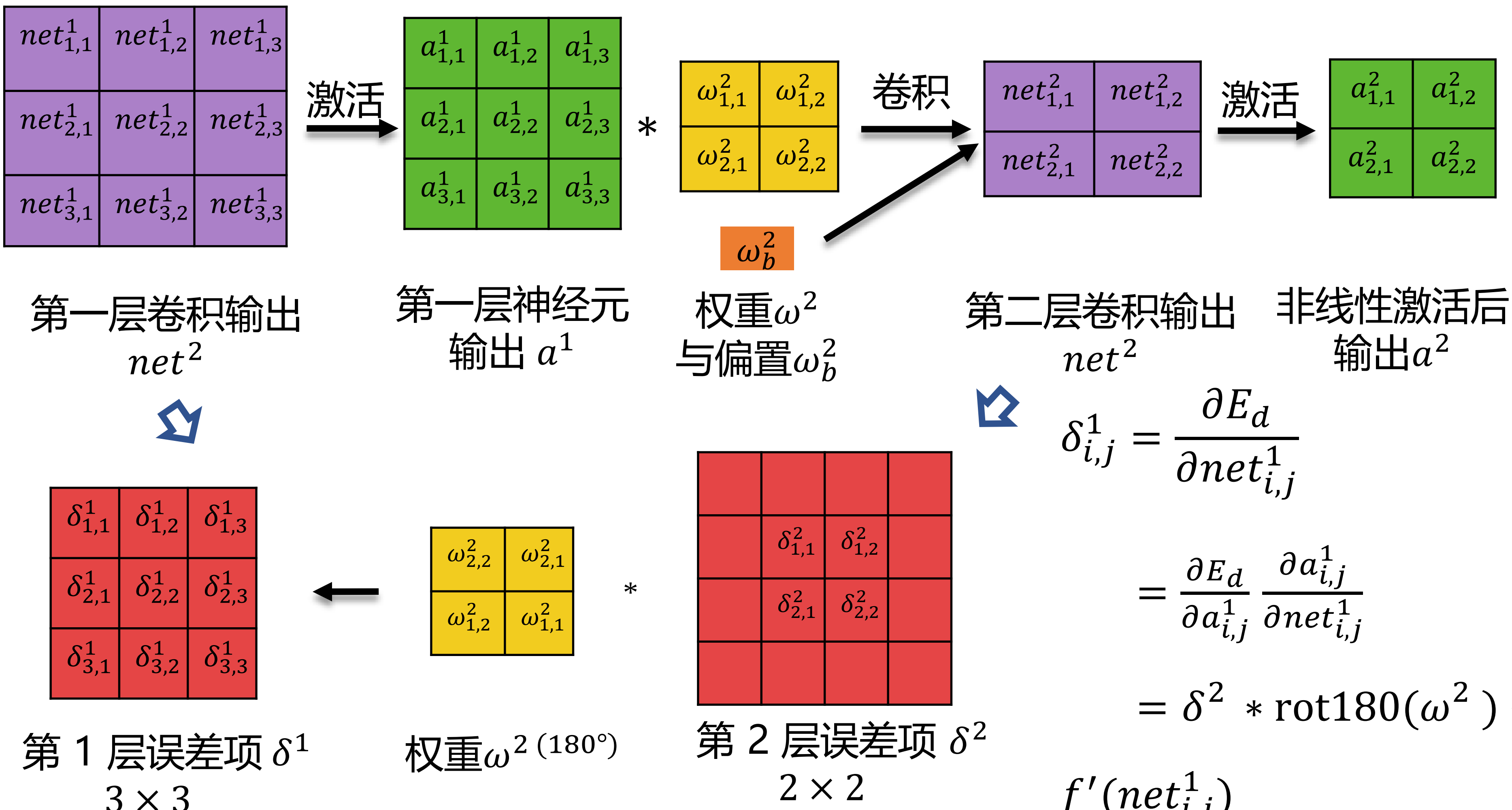
$$\delta_{1,2}^1=\delta_{1,1}^2\omega_{1,2}^2 f'(net_{1,2}^1)+\delta_{1,2}^2\omega_{1,1}^2 f'(net_{1,2}^1)$$

卷积网络的梯度反向传播



已知第 2 层误差敏感项 δ^2，计算第 1 层误差敏感项 δ^1 的传播

卷积网络的梯度反向传播

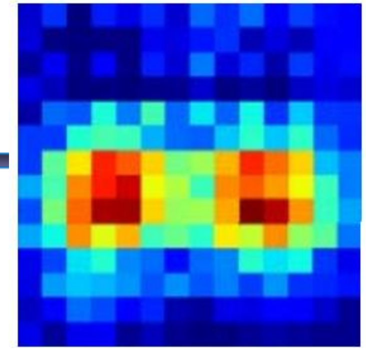


$$\delta^1_{i,j} = \frac{\partial E_d}{\partial net^1_{i,j}} = \frac{\partial E_d}{\partial a^1_{i,j}} \frac{\partial a^1_{i,j}}{\partial net^1_{i,j}} = \delta^2 * \mathrm{rot180}(\omega^2) \cdot f'(net^1_{i,j})$$

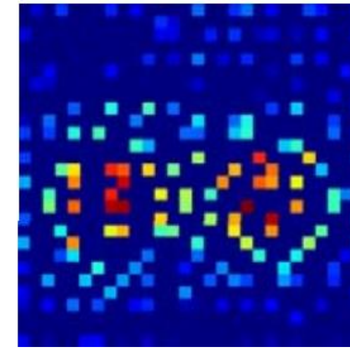
类比 $\delta^1_{1,2}$ 的计算公式，可以推出第一层第i行第j列神经元的误差敏感项传播公式为：

逆池化操作

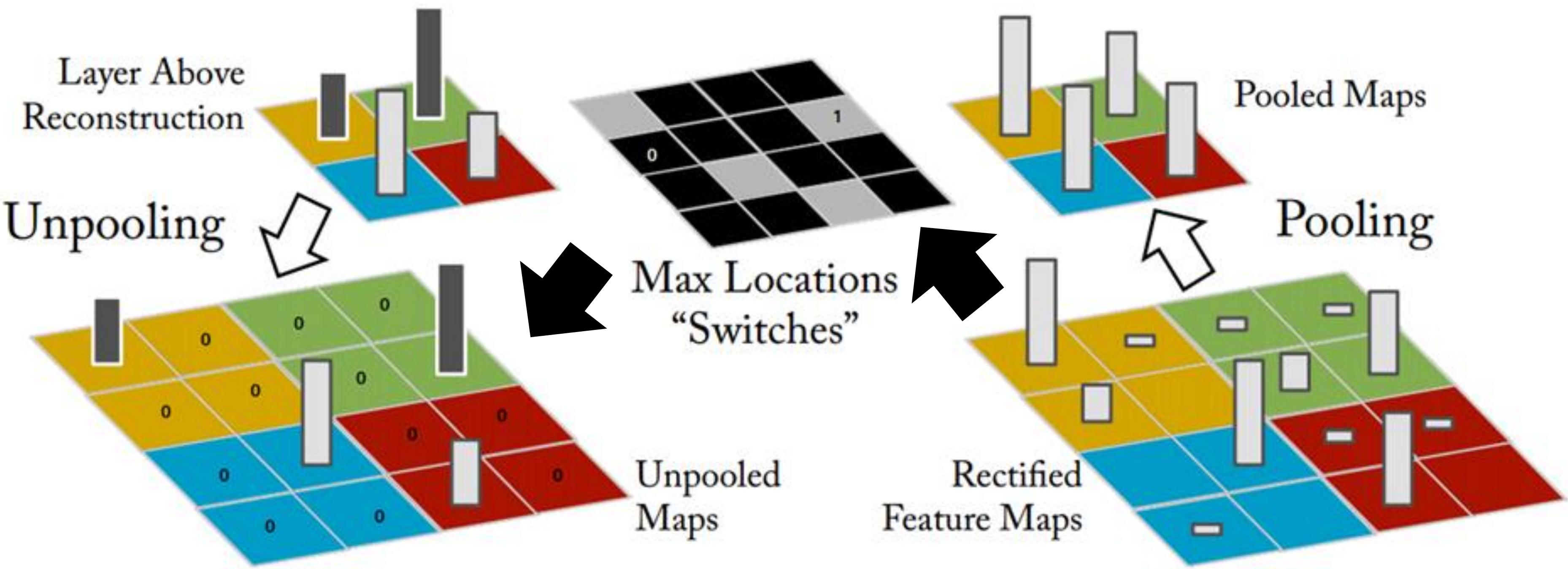
□Unpooling操作



14 x 14



28 x 28



也可以是简单重复，放置于所有位置

解卷积操作