信陽師範学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

第14章 神经网络与深度学习

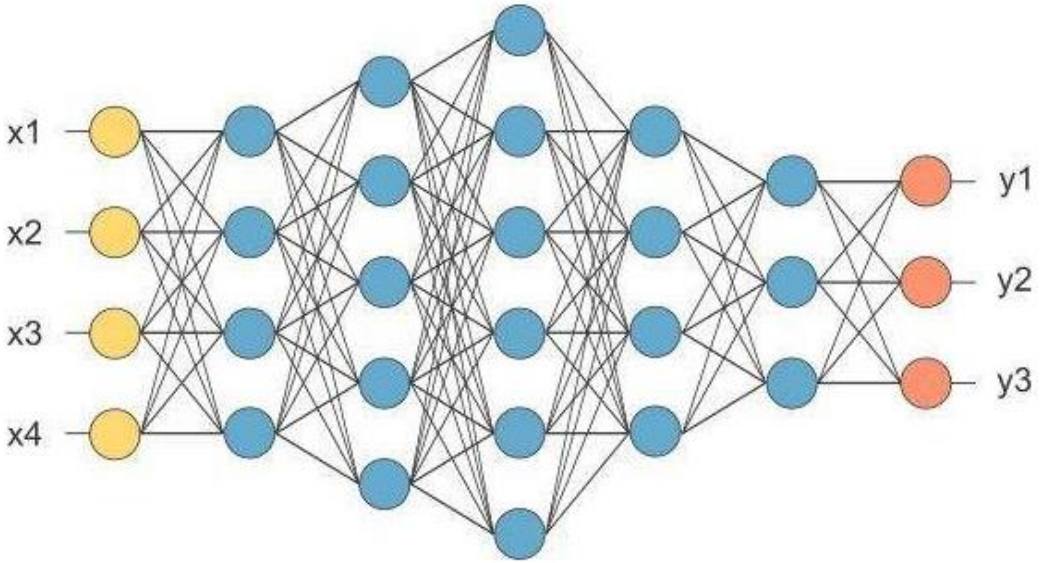
讲授人：牛言涛

日期：2020年5月23日

目录

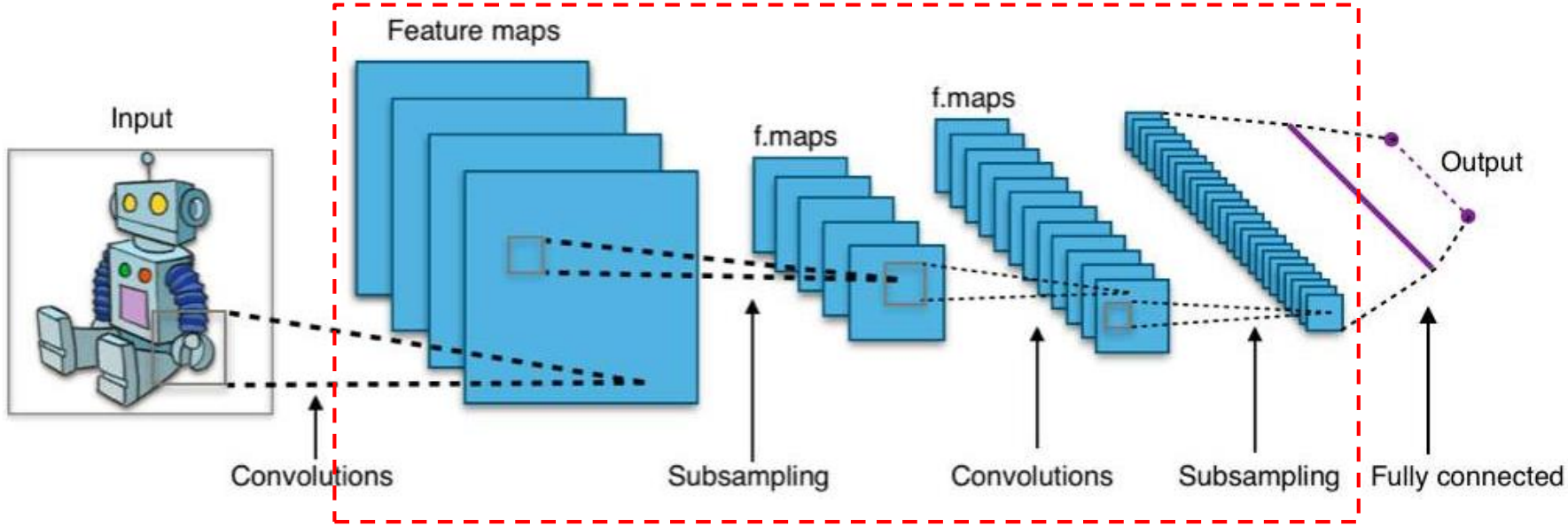
CONTENTS

1. 卷积神经网络概述



普通深度神经网络，隐藏层层数增多而已。

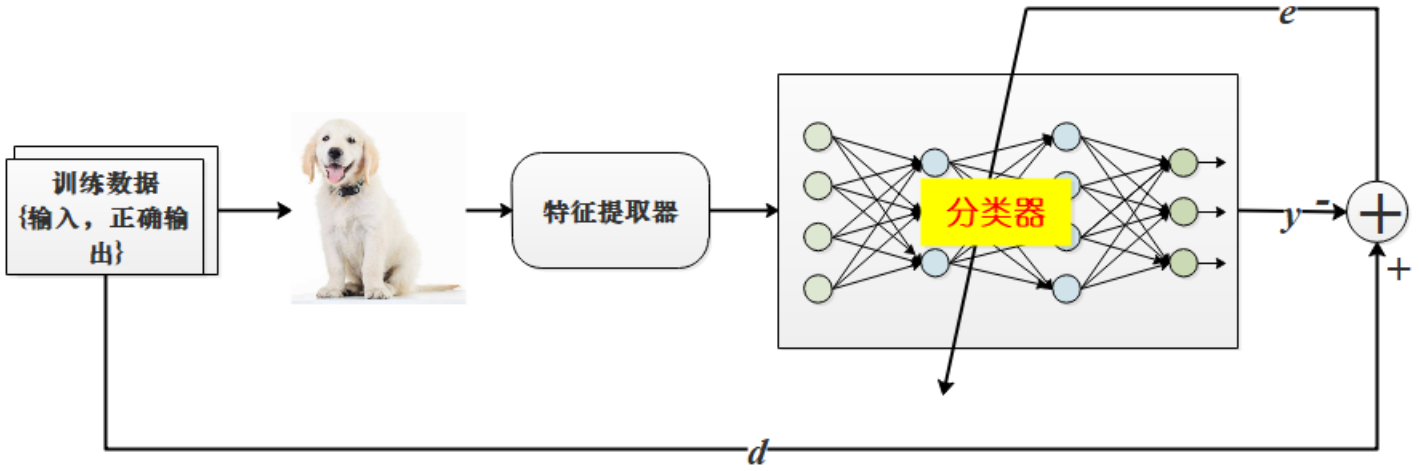
卷积神经网络：输入层、卷积层、激活函数、池化层、全连接层。



1. 卷积神经网络概述

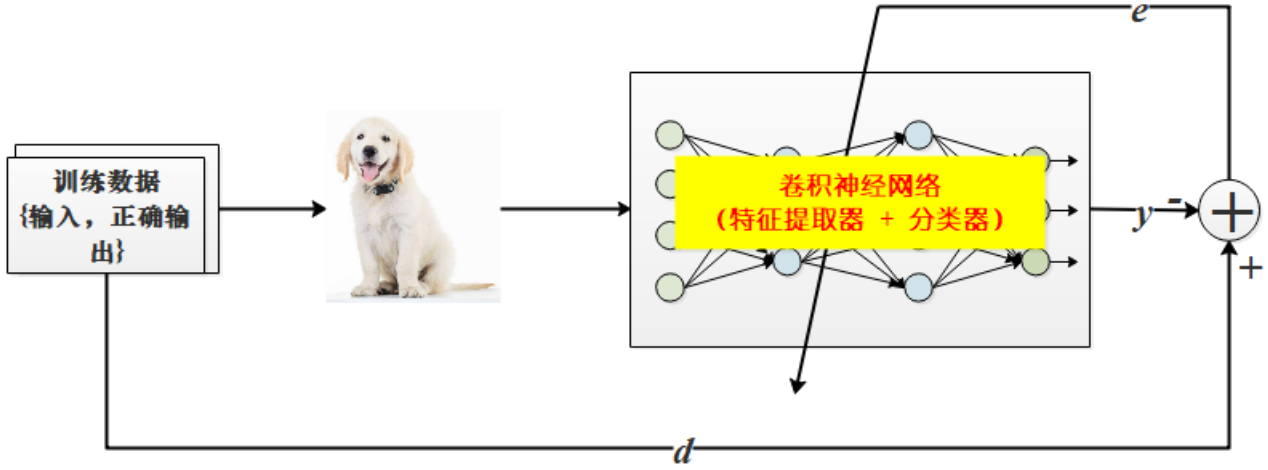
- 深度神经网络的重要性在于它打开了通向知识分层处理的复杂非线性模型和系统性方法的大门。卷积神经网络(ConvNet)是专门做图像识别的深度神经网络。从2012年开始，它引领着大多数的计算机视觉领域。
- 卷积神经网络不只是有很多隐藏层的深度神经网络，它也是一种模仿大脑视觉皮质进行图像处理和识别图像的深度网络。
- 图像识别基本上算是一种分类问题，卷积神经网络输出层通常采用多分类神经网络。然而，直接将原始图像用于图像识别而不考虑识别方法将会导致很差的结果。为了对比图像特征，需要提取处理图像。如第4节、第5节所使用的MNIST手写识别数字原始图像，分类结果较为有效，这是因为它们只是简单的黑白色块。
- 图像处理：特征提取器。

1. 卷积神经网络概述

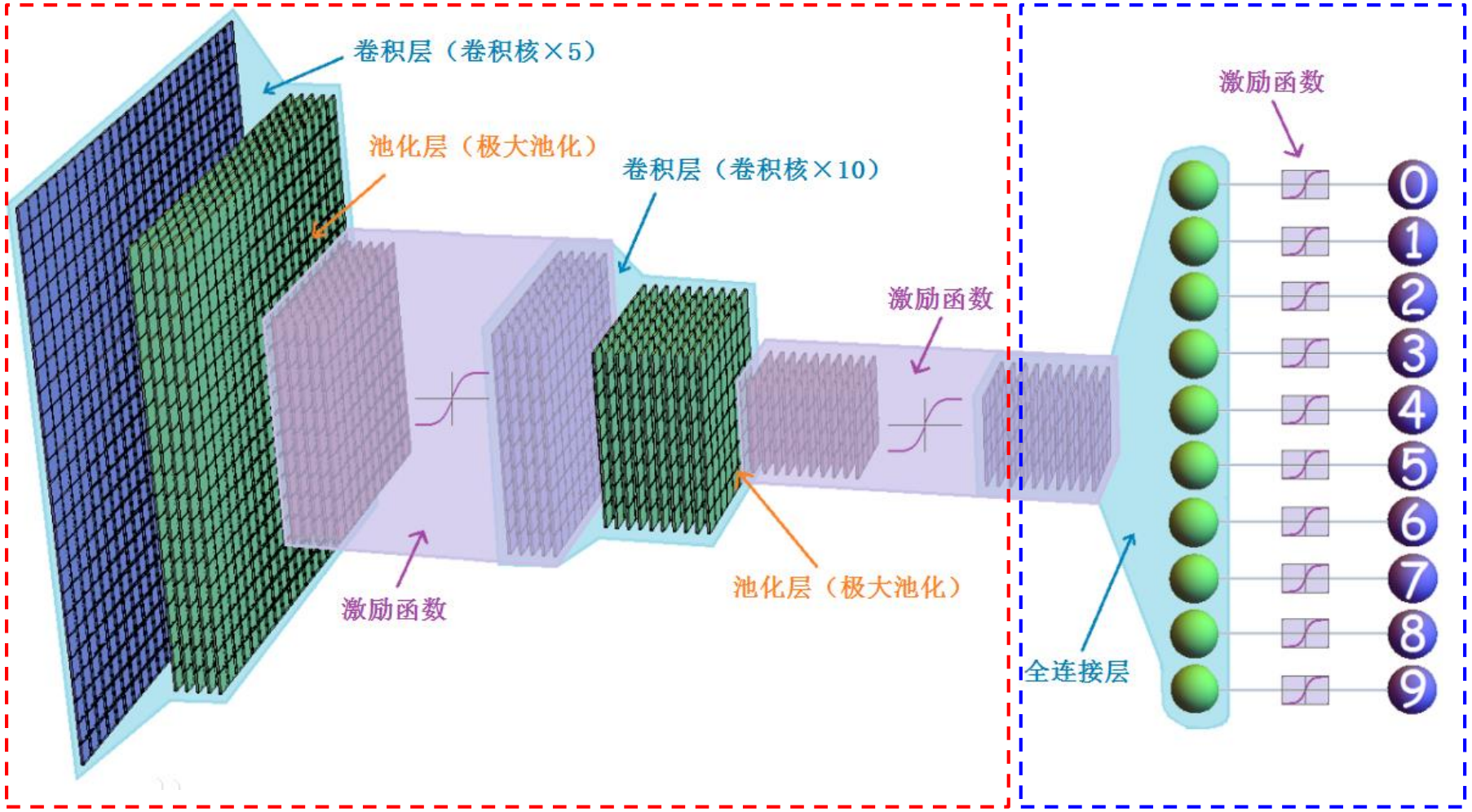


在神经网络之前，特征提取器根据使用领域不同，被设计成具有不同功能。它可能产生不同水平的学习性能。但是，这些特征提取器与机器学习是相互独立的。

卷积神经网络在训练过程中自动生成特征提取器，而不是由人工设计，它由一些特殊的神经网络类型组成，这些神经网络的权重是在训练过程中确定的。将人工设定的特征提取转变为自动生成特征提取是卷积神经网络的主要特点和优势。



2. 卷积神经网络的架构



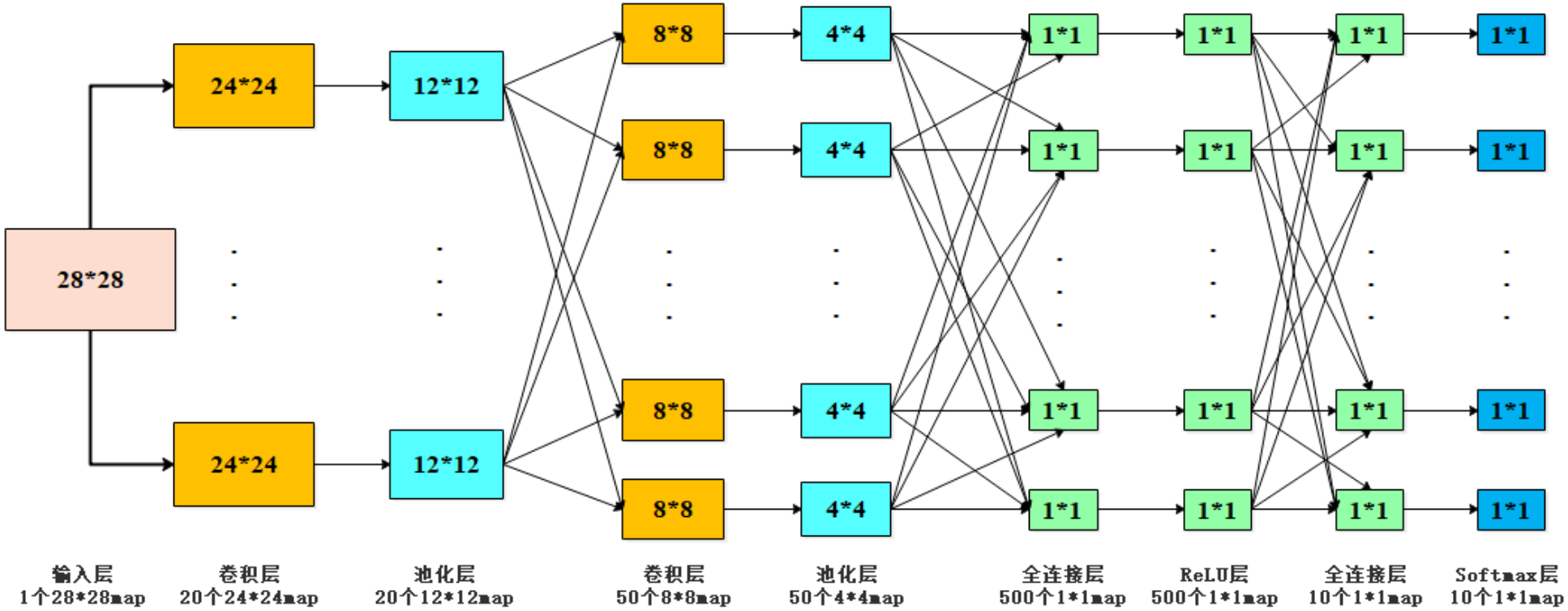
特征提取神经网络

分类神经网络

CNN包含提取输入图像特征的神经网络和另外一个进行图像分类的神经网络。特征提取神经网络的层数越深，图像识别的效果越好，而其代价是训练过程比较困难，这也是CNN曾经不实用和被遗忘的原因。

2. 卷积神经网络的架构

- 特征提取神经网络包括大量成对的卷积层和池化层。卷积层运算进行图像旋转，可以想象为数字过滤器的集合。池化层将邻近的像素合成为单个像素，因此它能降低图像的维度。分类神经网络通常采用普通的多分类神经网络。
- 卷积神经网络的层级结构：①数据输入层/ Input layer；②卷积计算层/ CONV layer；③ReLU激励层 / ReLU layer；④池化层 / Pooling layer；⑤全连接层 / FC layer。



3. 卷积层

- 卷积层生成的新图像叫做特征映射，它突出了原始图像的独特特征，它与其他神经网络的运算方法迥然不同。卷积层不使用连接权重与加权和，相反，它采用转换图像的过滤器，称之为“卷积过滤器”，当图像通过卷积过滤器之后会生成特征映射。
- 卷积核类似于人眼对物体进行扫描，其中有离散型卷积和连续性卷积：

连续型卷积 $y(t)=\int_{-\infty}^{\infty}x(p)h(t-p)dp=x(t)h(t)$

离散型卷积 $y(n)=\sum_{t=-\infty}^{\infty}x(t)h(n-t)=x(n)h(n)$

$$h_{记忆}(t)=f_{认知}(t)*g_{遗忘}(t)$$
$$=\int_{0}^{+\infty}f_{认知}(\tau)g_{遗忘}(t-\tau)d\tau$$

- 神经网络中的卷积操作都属于离散卷积，其实际上是一个线性运算，而不是真正意义上的卷积操作，相应的卷积核也可以称为滤波器。$x(t)$表示输入信号， $h(t)$表示滤波器或卷积核。卷积核大小确定了图像中参与运算子区域的大小。卷积核上的参数可以当成权重，就是说，每个像素点对最后的卷积结果的投票能力，权值越大，投票能力也就越大。

https://www.zhihu.com/question/22298352，卷积的通俗理解。

卷积在图像处理中的应用

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

conv_test.m

```
1 —    img = imread('flower.jpg');
2 —    w1 = [0 0 0;0 1 0;0 0 0];
3 —    w2 = [-1 -1 -1;-1 8 -1;-1 -1 -1];
4 —    w3 = [0 -1 0;-1 5 -1;0 -1 0];
5 —    w4 = ones(3,3)/9;
6 —    w5 = [-2 -1 0;-1 1 1;0 1 2];
7 —    f1 = imfilter(img,w1,'conv');
8 —    f2 = imfilter(img,w2,'conv');
9 —    f3 = imfilter(img,w3,'conv');
10 —   f4 = imfilter(img,w4,'conv');
11 —   f5 = imfilter(img,w5,'conv');
12 —   subplot(2,3,1)
13 —   imshow(img);title('原始图像');
14 —   subplot(2,3,2)
15 —   imshow(f1);title('同一化卷积后的图像');
16 —   subplot(2,3,3)
17 —   imshow(f2);title('边界检测卷积后的图像');
18 —   subplot(2,3,4)
19 —   imshow(f3);title('锐化卷积后的图像');
20 —   subplot(2,3,5)
21 —   imshow(f4);title('均值模糊化卷积后的图像');
22 —   subplot(2,3,6)
23 —   imshow(f5);title('浮雕卷积后的图像');
```

原始图像

同一化卷积后的图像

边界检测卷积后的图像

锐化卷积后的图像

均值模糊化卷积后的图像

浮雕卷积后的图像

卷积在图像处理中的应用——图像锐化滤波器Sharpness Filter

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

```
img = imread('dog2.png');
w1 = [-1 -1 -1;-1 9 -1;-1 -1 -1];
w2 = [-1 -1 -1 -1 -1 ;-1 2 2 2 -1;-1 2 8 2 -1;-1 2 2 2  -1;-1 -1 -1 -1 -1];
w3 = [1 1 1;1 -7 1;1 1 1];
image1 = imfilter(img,w1,'conv');
image2 = imfilter(img,w2,'conv');
image3 = imfilter(img,w3,'conv');
subplot(2,2,1)
imshow(img);title('原始图像');
subplot(2,2,2)
imshow(image1);title('锐利图像');
subplot(2,2,3)
imshow(image2);title('核加大（更精细）');
subplot(2,2,4)
imshow(image3);title('更强调边缘');
```

原始图像

锐利图像

核加大（更精细）

更强调边缘

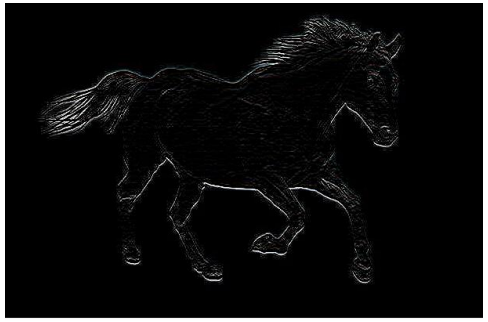
卷积在图像处理中的应用——边缘检测Edge Detection

```
img = imread('horse.jpg');
w1 = [-1 -1 -1;-1 9 -1;-1 -1 -1];
w2 = [0 0 0 0 0 ;0 0 0 0 0;-1 -1 2 0 0;0 0 0 0 0 ;0 0 0 0 0];
w3 = [0 0 -1 0 0 ;0 0 -1 0 0;0 0 4 0 0;0 0 -1 0 0 ;0 0 -1 0 0];
w4 = [-1 0 0 0 0;0 -2 0 0 0;0 0 6 0 0;0 0 0 -2 0 ;0 0 0 0 -1];
image1 = imfilter(img,w1,'conv');
image2 = imfilter(img,w2,'conv');
image3 = imfilter(img,w3,'conv');
image4 = imfilter(img,w4,'conv');
subplot(2,2,1)
imshow(image1);title('所有方向边缘');
subplot(2,2,2)
imshow(image2);title('水平边缘');
subplot(2,2,3)
imshow(image3);title('垂直边缘');
subplot(2,2,4)
imshow(image4);title('45°边缘');
```

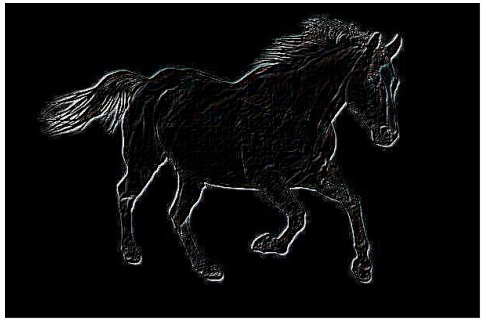
所有方向边缘

水平边缘

垂直边缘



45°边缘

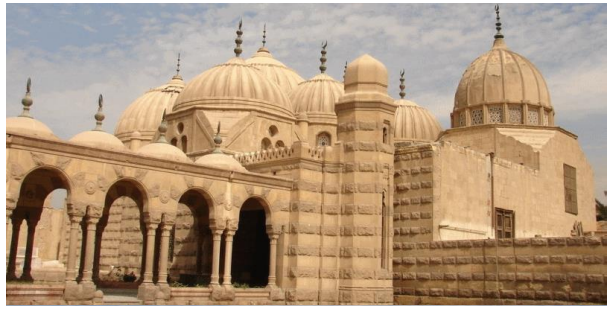


卷积在图像处理中的应用——浮雕Embossing Filter

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

```
img = imread('build.png');
w1 = [-1 -1 0;-1 0 1;0 1 1];
w2 = [-1 -1 -1 -1 0 ;-1 -1 -1 0 1;-1 -1 0 1 1;-1 0 1 1 1;0 1 1 1 1];
w3 = [2 0 0;0 -1 0;0 0 -1];
image1 = imfilter(img,w1,'conv');
image2 = imfilter(img,w2,'conv');
image3 = imfilter(img,w3,'conv');
subplot(2,2,1)
imshow(img);title('原图像');
subplot(2,2,2)
imshow(image1);title('45度的浮雕滤波器');
subplot(2,2,3)
imshow(image2);title('加大的滤波器');
subplot(2,2,4)
imshow(image3);title('非对称');
```

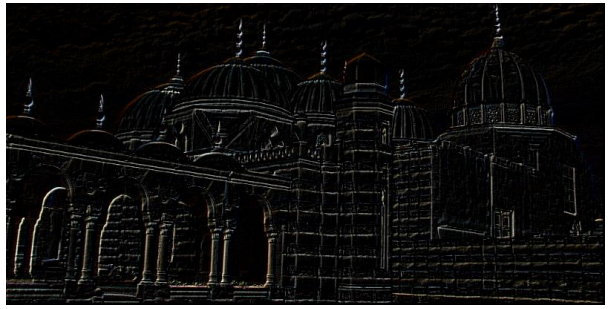
原图像



45度的浮雕滤波器

加大的滤波器

非对称



卷积在图像处理中的应用——运动模糊Motion Blur

```
img = imread('build.png');
%运动模糊可以通过只在一个方向模糊达到，例如9x9
的运动模糊滤波器。这个效果就好像，摄像机是从左
上角移动的右下角。
w1 = eye(9)/9;
image1 = imfilter(img,w1,'conv');
subplot(2,1,1)
imshow(img);
title('原图像');
subplot(2,1,2)
imshow(image1);
title('运动模糊Motion Blur');
```

原图像

运动模糊Motion Blur

卷积在图像处理中的应用——均值模糊Box Filter (Averaging)

```
img = imread('dog2.png');
w1 = [0 0.2 0;0.2 0 0.2;0 0.2 0];
w2 = [0 0 1 0 0 ;0 1 1 1 0;1 1 1 1 1;0 1 1 1 0;0 0 1 0 0]/13;
w3 = ones(5)/25;
image1 = imfilter(img,w1,'conv');
image2 = imfilter(img,w2,'conv');
image3 = imfilter(img,w3,'conv');
subplot(2,2,1)
imshow(img);title('原图像');
subplot(2,2,2)
imshow(image1);title('均值模糊');
subplot(2,2,3)
imshow(image2);title('加大的滤波器');
subplot(2,2,4)
imshow(image3);title('均值模糊');
```

原图像

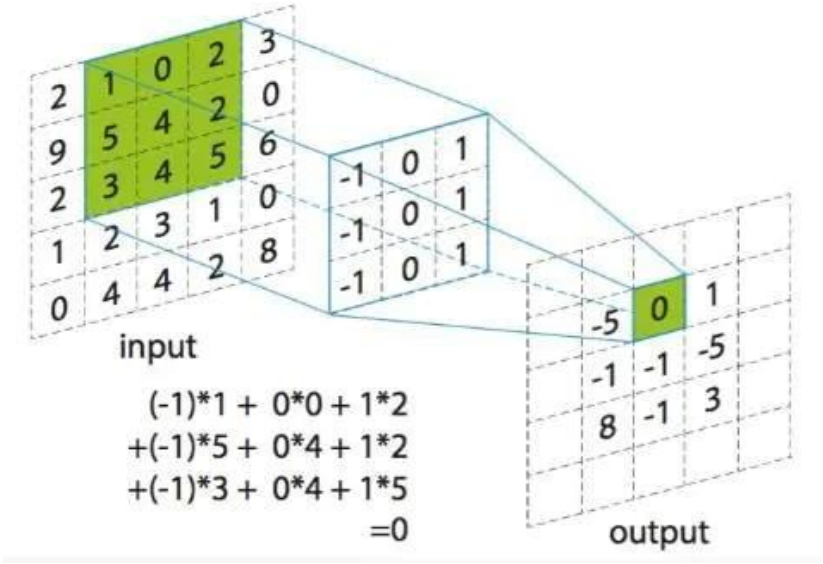
均值模糊

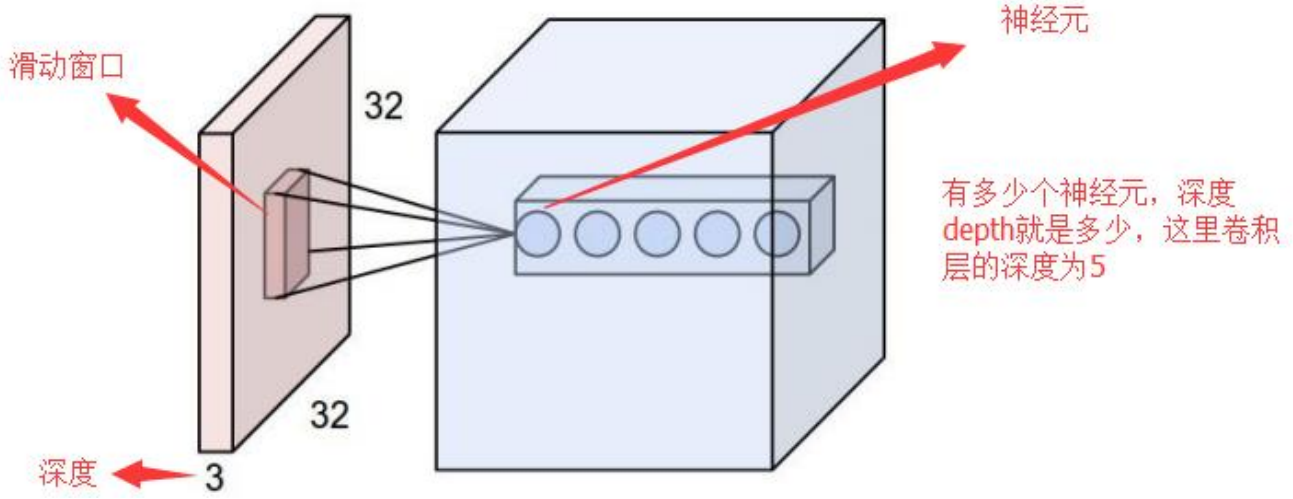
加大的滤波器

均值模糊

3. 卷积层

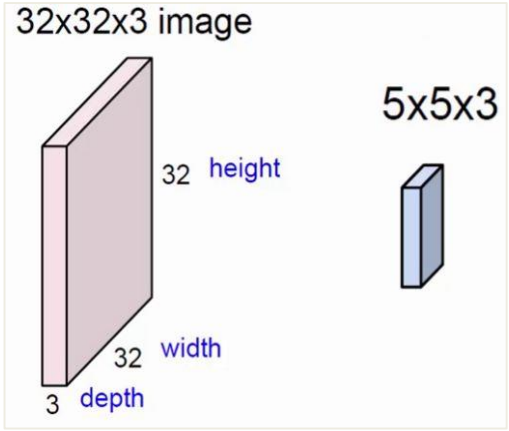
- 卷积层两个关键操作：
  - 局部关联：每个神经元可看做一个滤波器(filter)，第i层的每一个神经元都只和第$i-1$层的一个局部窗口内的神经元相连，构成一个局部连接网络。
  - 窗口(receptive field)滑动， filter对局部数据计算。
- 卷积层相关概念：
  - 深度/depth；步长/stride（窗口一次滑动的长度）；填充值/zero-padding

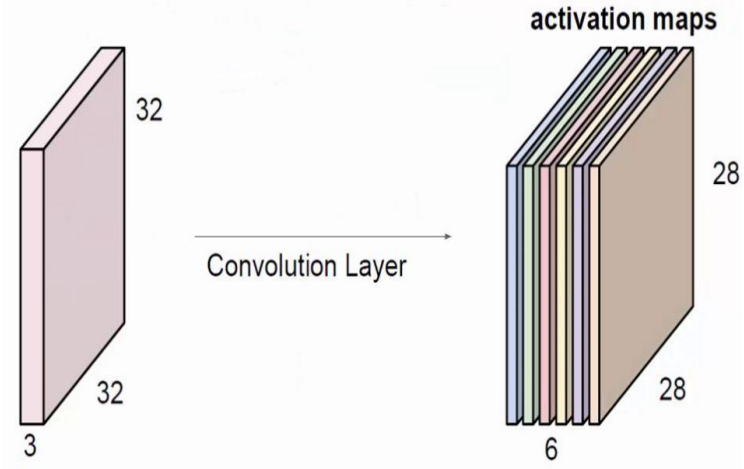




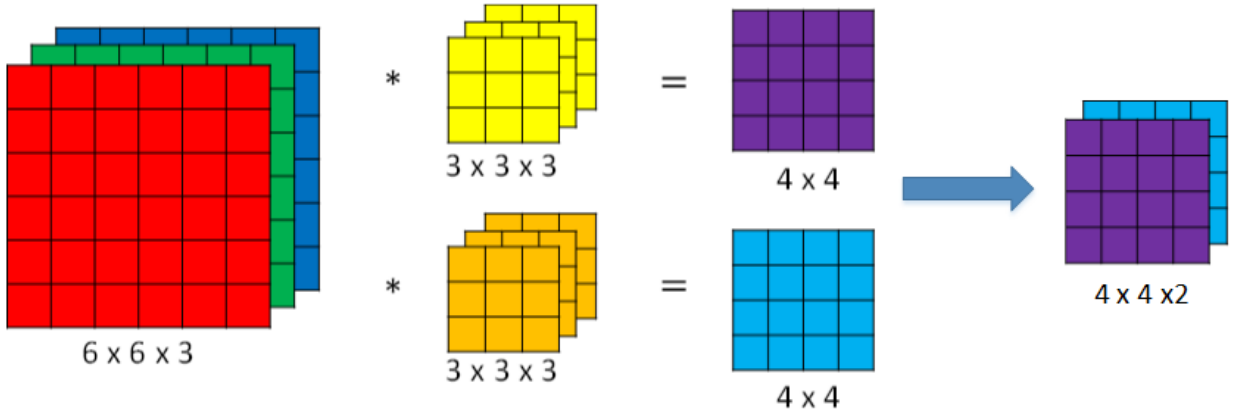
3. 卷积层

左图左方块是输入层，尺寸为32*32的3通道图像，小方块是filter（卷积核），尺寸为5*5，深度为3。将输入层划分为多个区域，用filter在输入层做卷积运算，得到一个深度为1的特征图。右图：一般使用多个filter分别进行卷积，最终得到多个特征图。



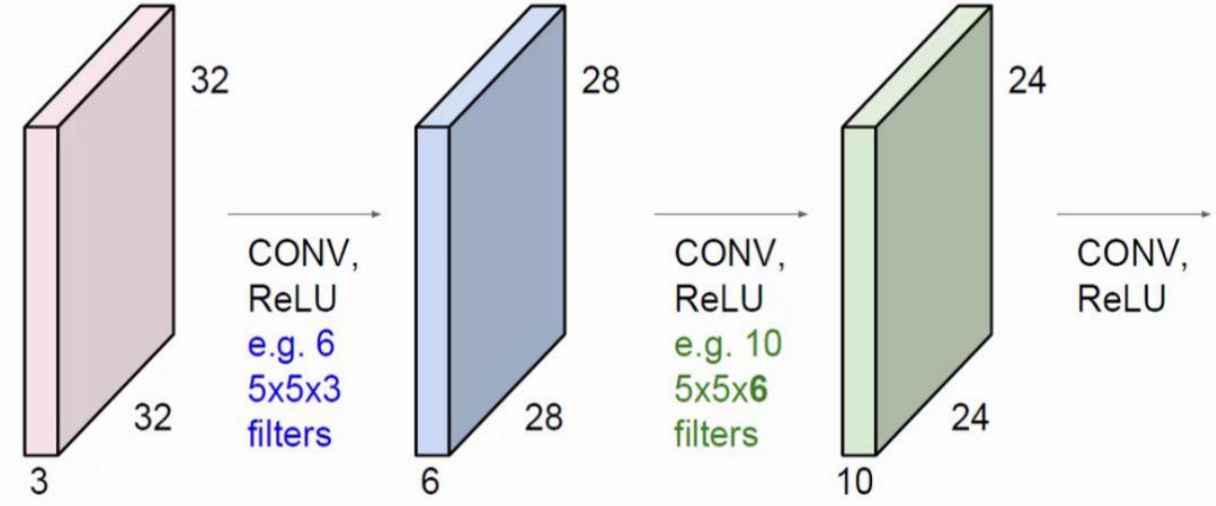


如右下图：为了进行多个卷积运算，实现更多边缘检测，可以增加更多的滤波器组。例如设置第一个滤波器组实现垂直边缘检测，第二个滤波器组实现水平边缘检测。不同滤波器组卷积得到不同的输出，个数由滤波器组决定。



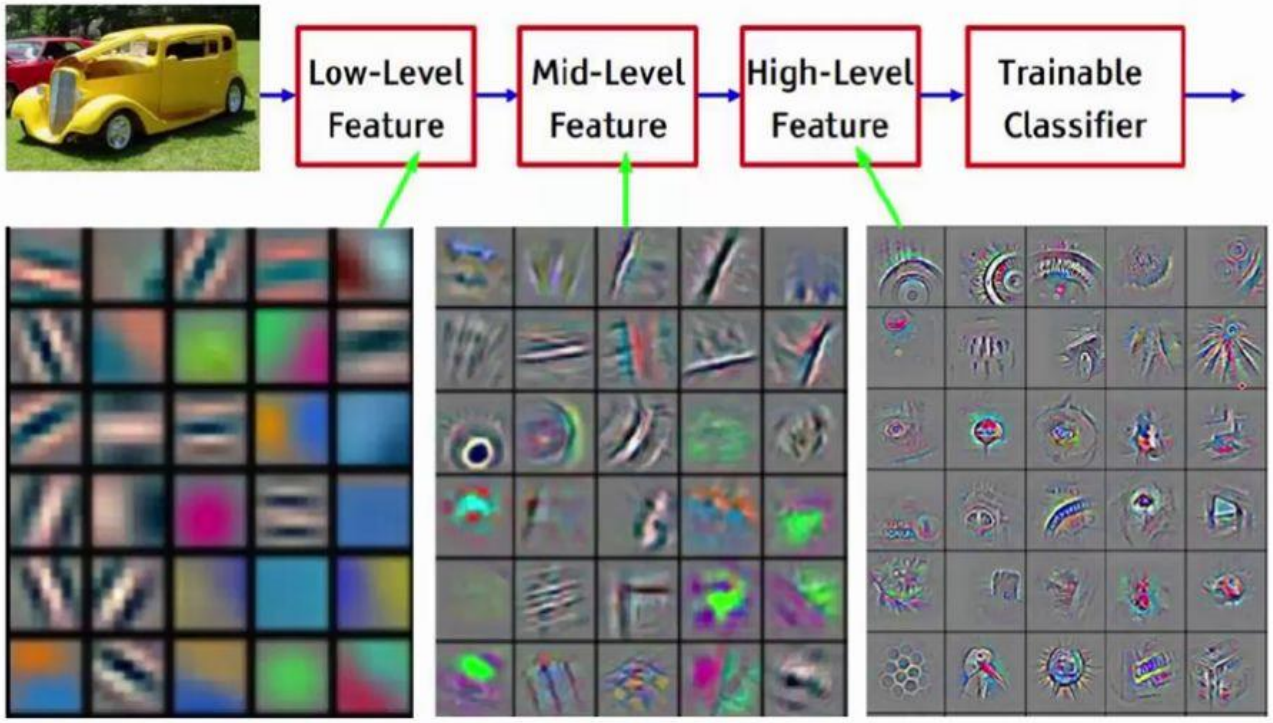
3. 卷积层

- 卷积网络的核心思想是将：局部感受野、权值共享以及时间或空间亚采样(池化)这三种结构思想结合起来获得了某种程度的位移、尺度、形变不变性。
- 卷积不仅限于对原始输入的卷积。蓝色方块是在原始输入上进行卷积操作，使用了6个filter得到了6个提取特征图。绿色方块还能对蓝色方块进行卷积操作，使用了10个filter得到了10个特征图。每一个filter的深度必须与上一层输入的深度相等。



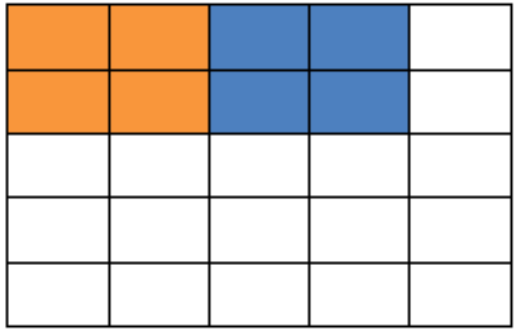
3. 卷积层

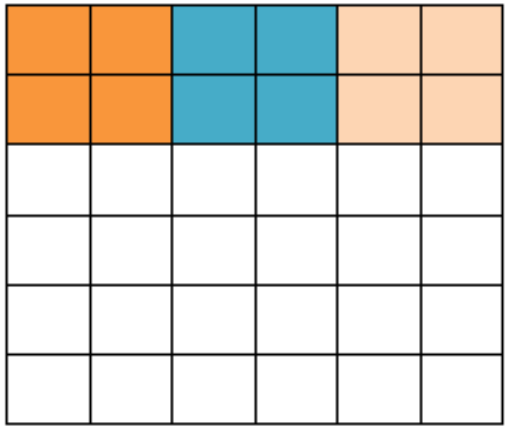
- 第一次卷积可以提取出低层次的特征；第二次卷积可以提取出中层次的特征；第三次卷积可以提取出高层次的特征。
- 特征是不断进行提取和压缩的，最终能得到比较高层次特征，简言之就是对原式特征一步又一步的浓缩，最终得到的特征更可靠。利用最后一层特征可以做各种任务：比如分类、回归等。

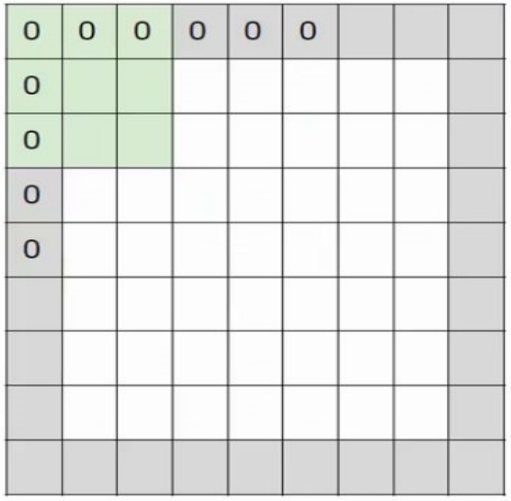


3. 卷积层

- 填充值：如下左图，一个5*5的图片（一个格子一个像素），滑动窗口取2*2，步长取2，那么还剩下1个像素没法滑完，怎么办？
- 在原先的矩阵加了一层填充值，使得变成6*6的矩阵，那么窗口就可以刚好把所有像素遍历完。这就是填充值的作用。



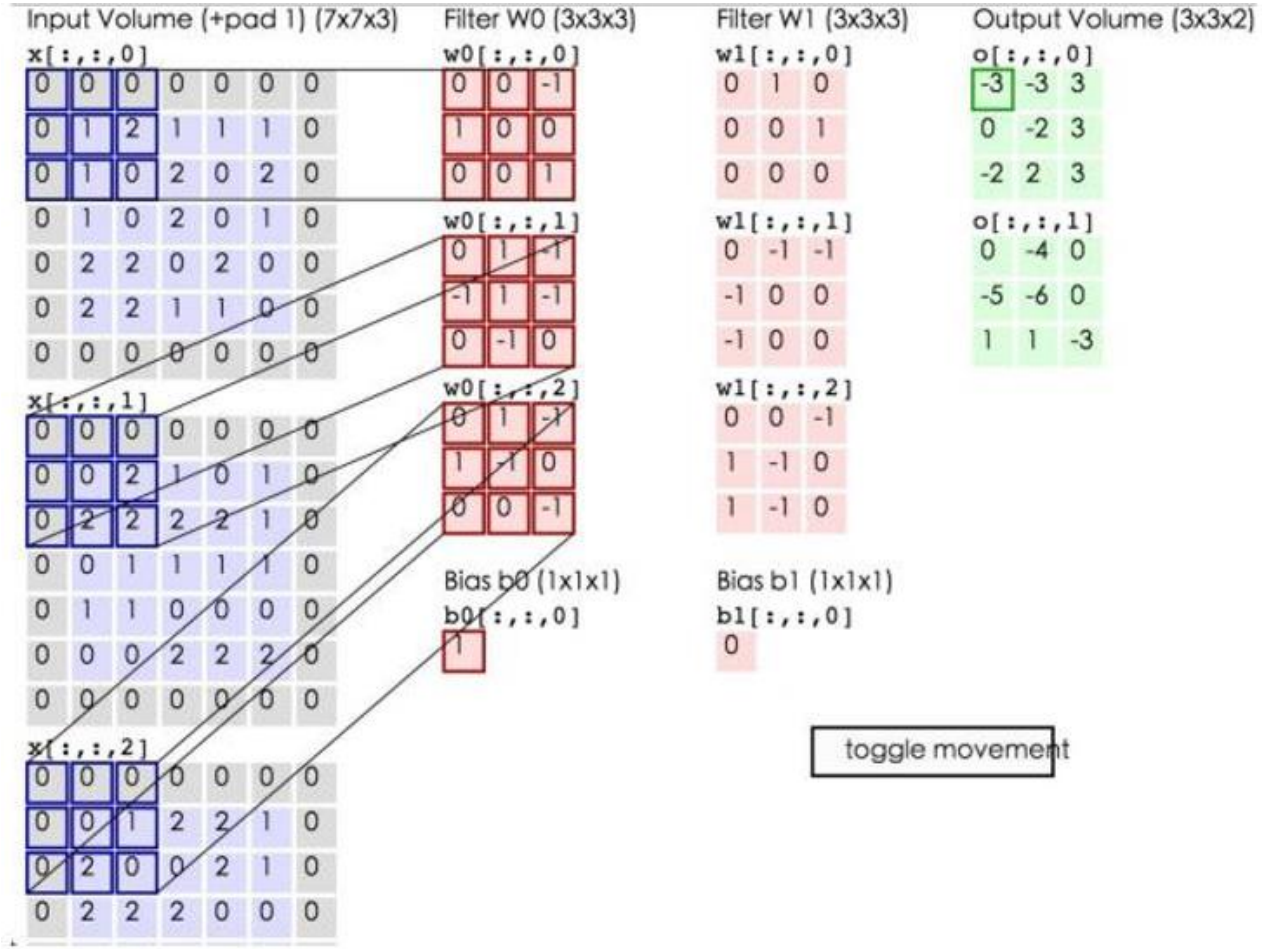




滑动的步长叫stride记为S。S越小，提取的特征越多，但是S一般不取1，主要考虑时间效率的问题。S也不能太大，否则会漏掉图像上的信息。

(1)卷积的计算

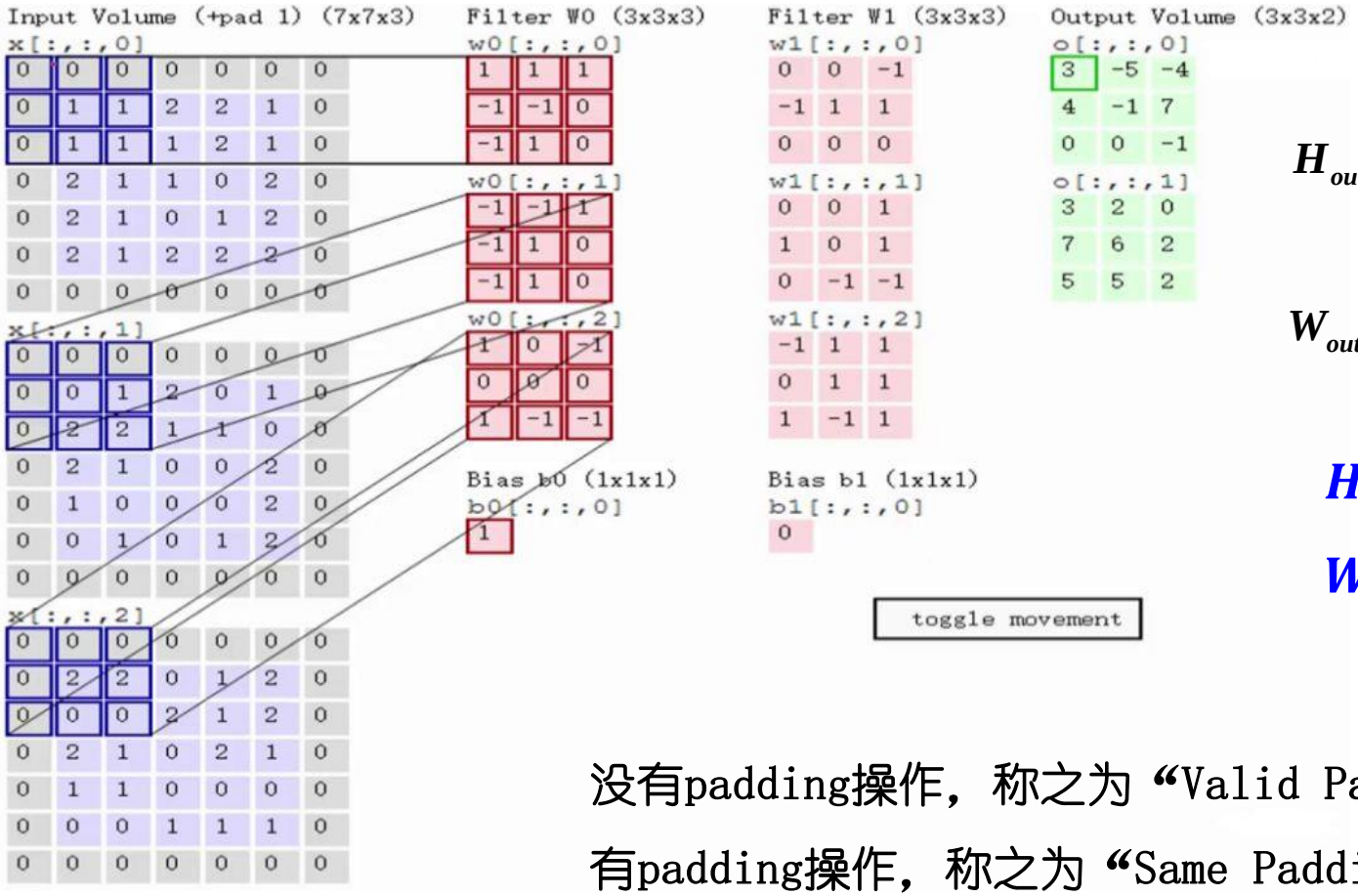
蓝色矩阵周围有一圈灰色的框，即填充值。左区域的三个大矩阵是原式图像的输入，RGB三个通道用三个矩阵表示，大小为7*7*3。



- 粉色矩阵就是卷积层的神经元，这里表示有两个神经元(w0,w1)；绿色矩阵就是经过卷积运算后的输出矩阵，这里的步长设置为2。
- Filter W0、W1表示1个卷积核，尺寸为3*3，深度为3（三个矩阵）。因为卷积中用了2个卷积核，因此该卷积层结果的输出深度为2（绿色矩阵有2个）。
- Bias b0是Filter W0的偏置项，Bias b1是Filter W1的偏置项。

(1)卷积的计算

卷积结果输出维度的计算



$$H_{out}=\left\lfloor\frac{H_{in}+2H_{padding}-H_{kernel}}{H_{stride}}\right\rfloor+1=\frac{5+2-3}{2}+1=3$$

$$W_{out}=\left\lfloor\frac{W_{in}+2W_{padding}-W_{kernel}}{W_{stride}}\right\rfloor+1=\frac{5+2-3}{2}+1=3$$

H_{out}垂直维度补零高度

W_{out}垂直维度步幅大小

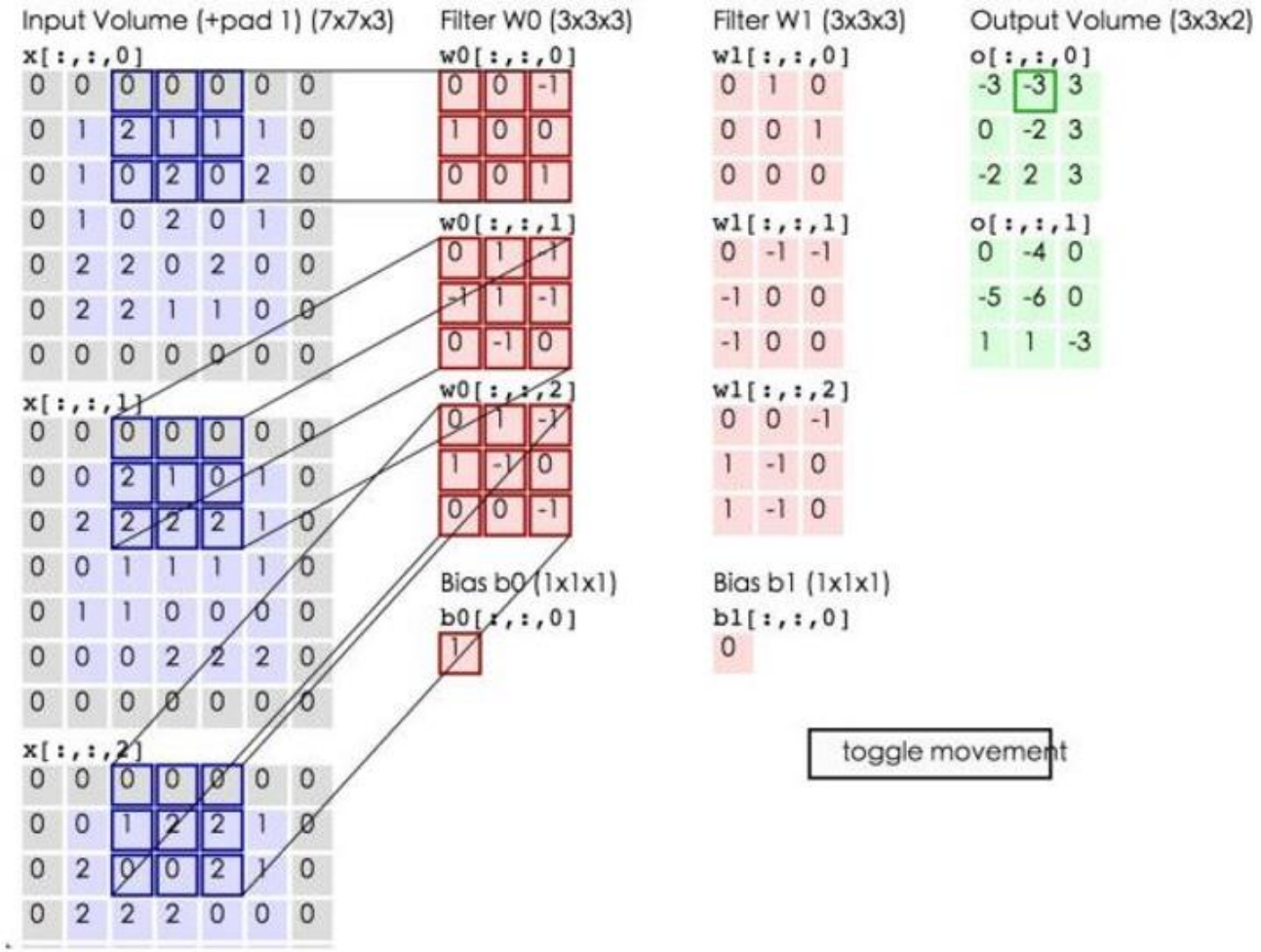
没有padding操作，称之为“Valid Padding”有效填充；

有padding操作，称之为“Same Padding”等大填充。

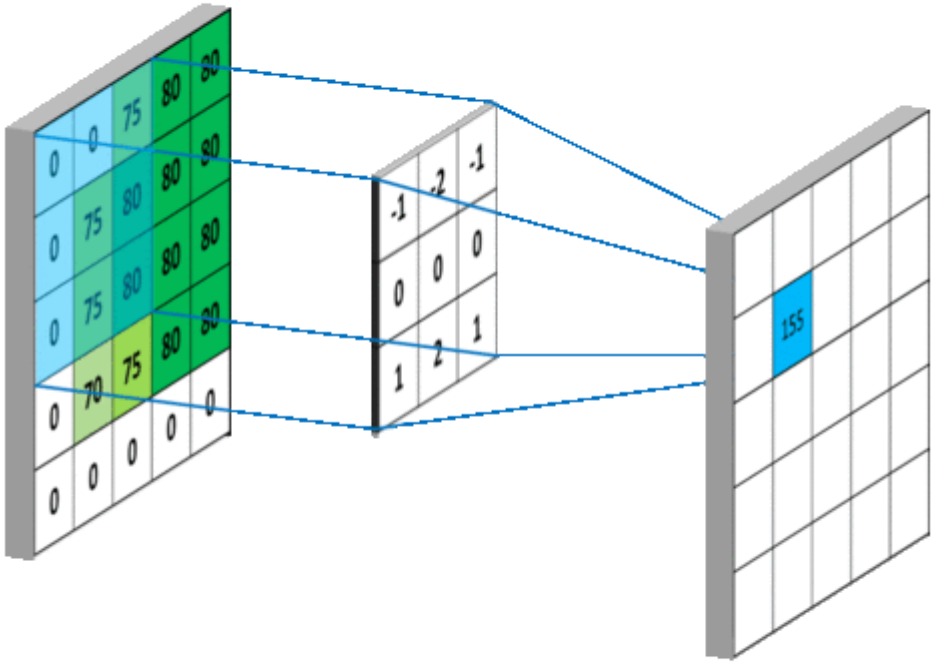
(1)卷积的计算

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

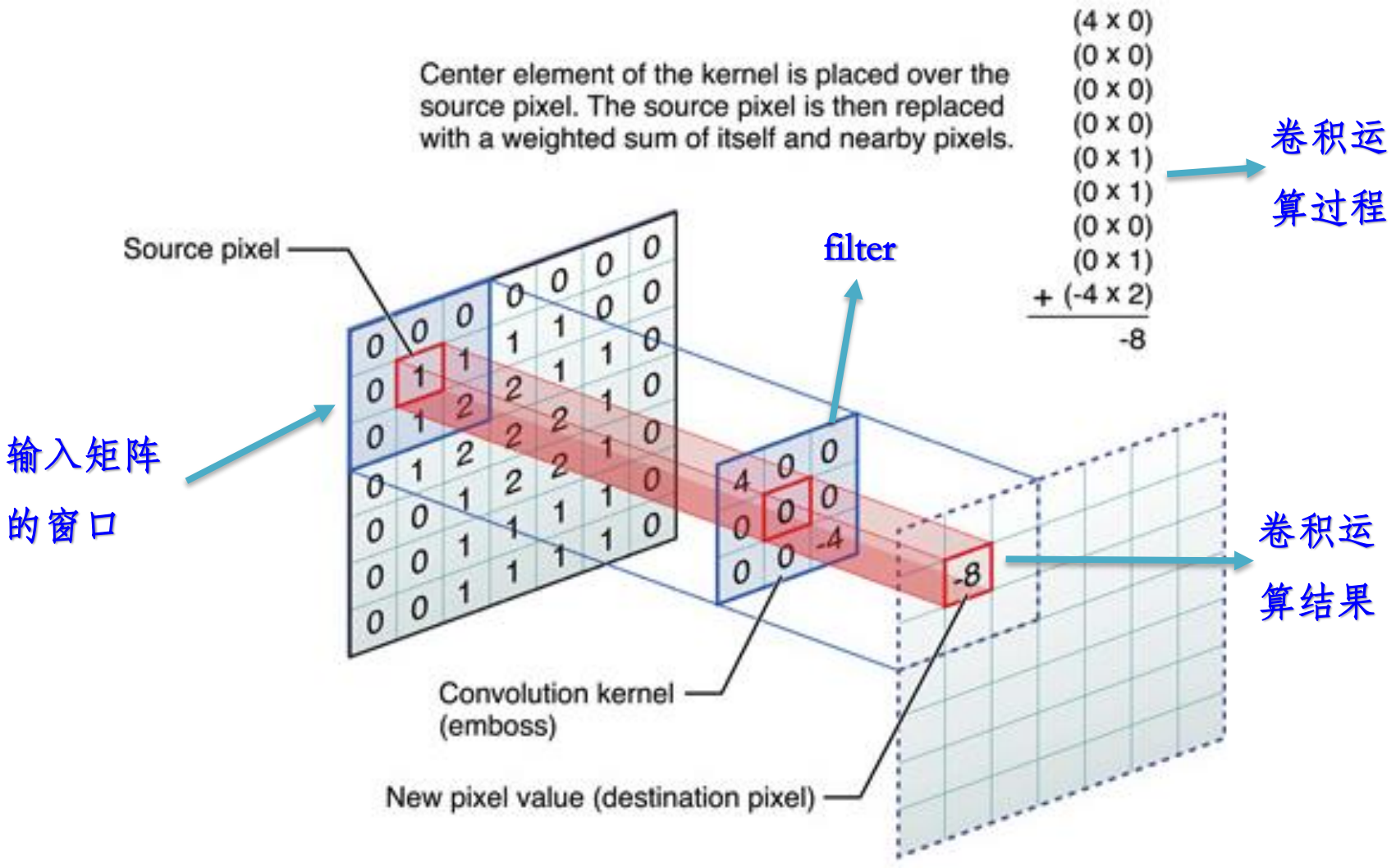
卷积计算过程



蓝色的矩阵(输入图像)对粉色的矩阵（filter）进行矩阵内积计算并将三个内积运算的结果与偏置值b相加（如：2+（-2+1-2）+（1-2-2）+1=2 - 3 - 3 + 1 = -3），计算后的值就是绿框矩阵的一个元素。

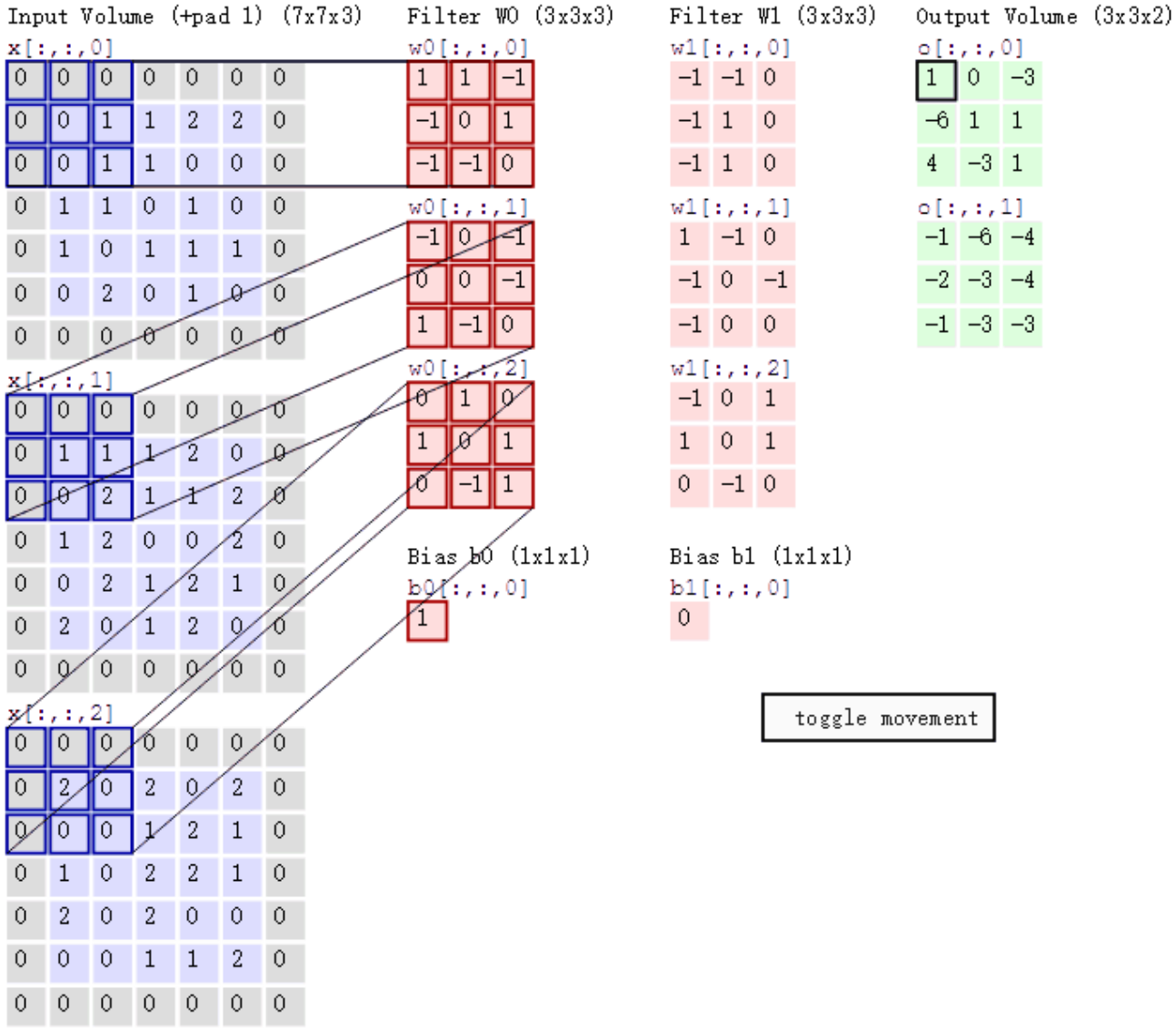


(1)卷积的计算


Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.
(4 x 0)
(0 x 0)
(0 x 0)
(0 x 0)
(0 x 1)
(0 x 1)
(0 x 0)
(0 x 1)
+ (-4 x 2)
-8
卷积运算过程
Source pixel
filter
输入矩阵的窗口
卷积运算结果
Convolution kernel (emboss)
New pixel value (destination pixel)

(1)卷积的计算

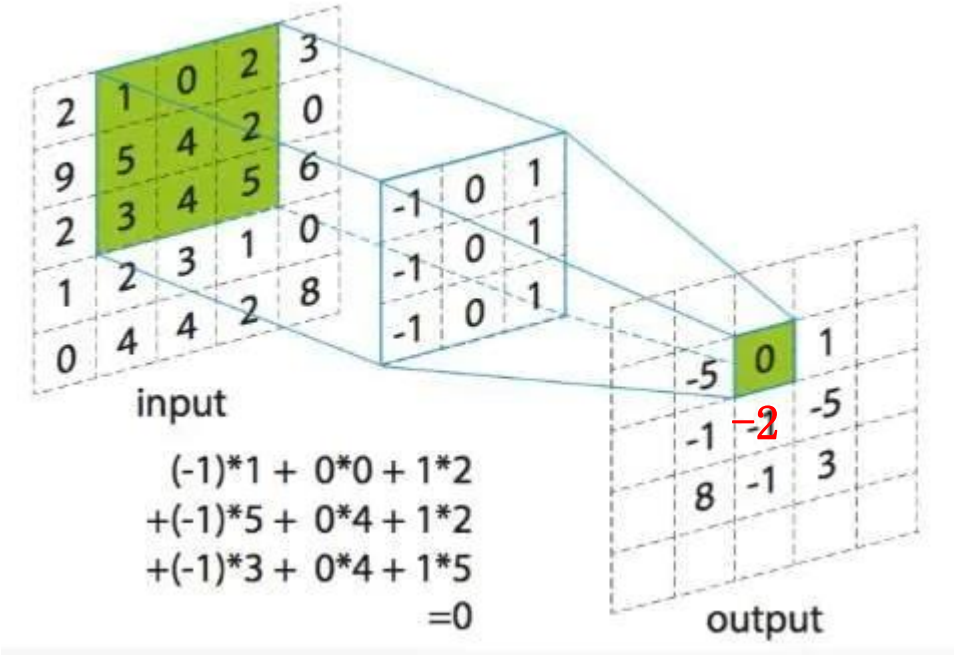
卷积计算动态过程



(1)卷积的计算

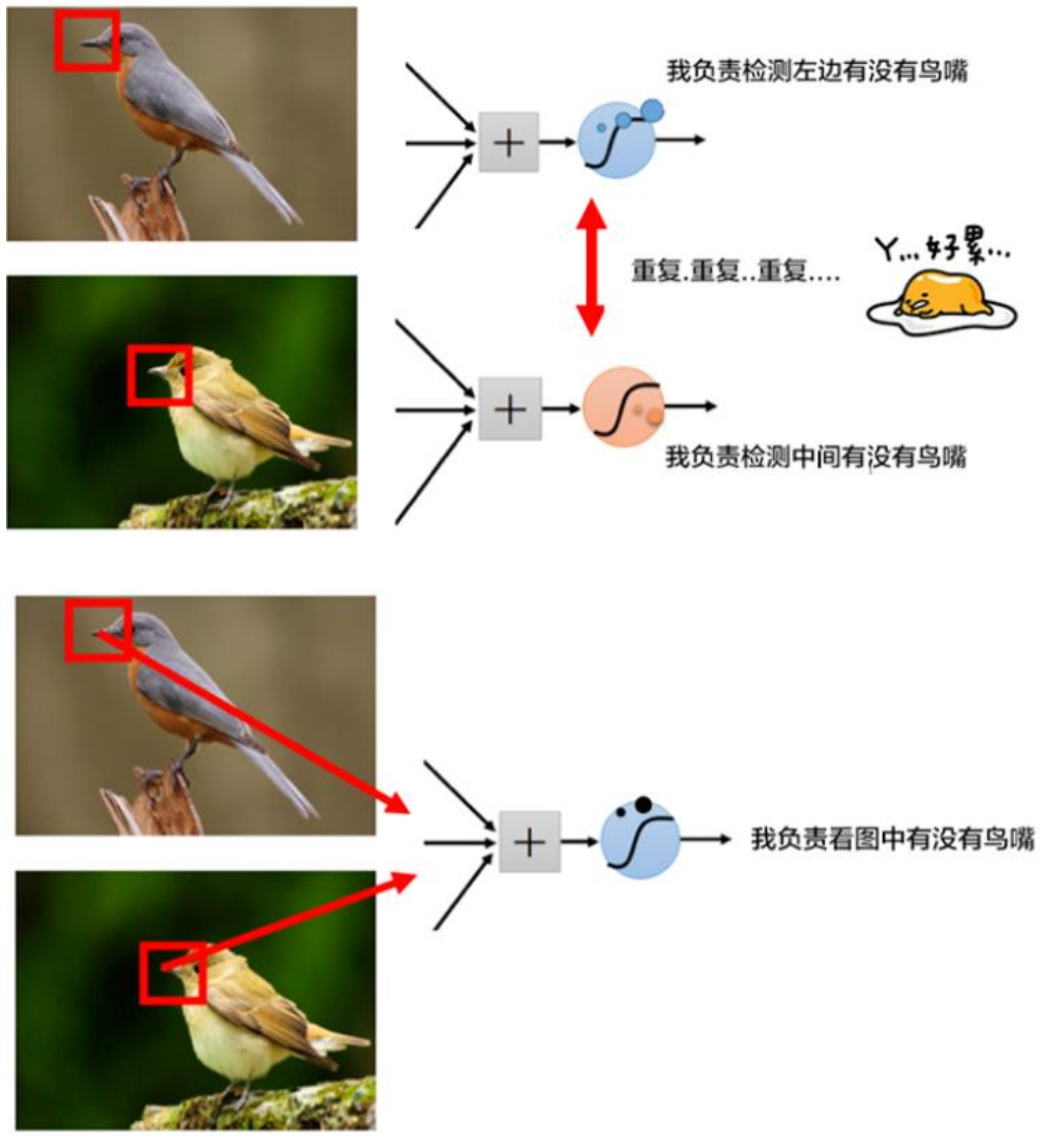
MATLAB实现卷积运算的函数为cov2(A,B,type)，其中type类型为full、same、valid。如果实现与神经网络卷积运算相同的功能，需要对卷积核旋转180度。

```
>> X = [2 1 0 2 3;9 5 4 2 0;2 3 4 5 6;1 2 3 1 0;0 4 4 2 8];
>> W = [-1 0 1;-1 0 1;-1 0 1];
>> W1 = rot90(W,2);
>> C1 = conv2(X,W,'valid')
C1 =
     5     0    -1
     1     2     5
    -8     1    -3
>> C2 = conv2(X,W1,'valid')
C2 =
    -5     0     1
    -1    -2    -5
     8    -1     3
```



(2)参数共享机制

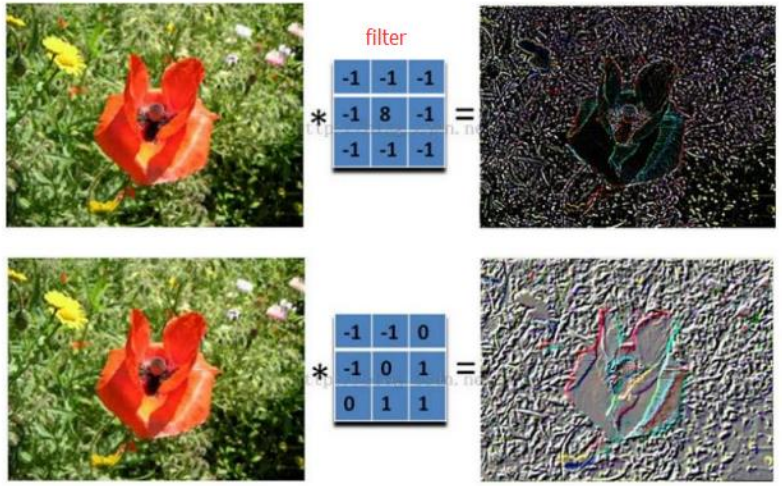
- 权值共享：不同的图像或者同一张图像共用一个卷积核，减少重复的卷积核。同一张图像当中可能会出现相同的特征，共享卷积核能够进一步减少权值参数。
- 如右图所示，为了找到鸟嘴，一个激活函数A需要检测图像左侧有没有鸟嘴，另外一个激活函数B需要检测另外一张图像中间有没有类似的鸟嘴。其实，鸟嘴都可能具有同样的特征，只需要一个激活函数C就可以，这个时候，就可以共享同样的权值参数（卷积核）。



(2)参数共享机制

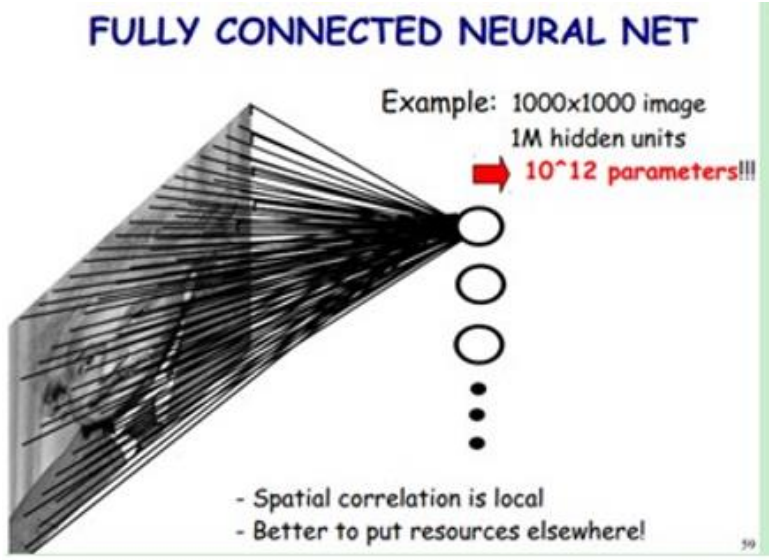
在卷积层中每个神经元连接数据窗的权重是固定的，每个神经元只关注一个特性。神经元就是图像处理中的滤波器，比如边缘检测专用的Sobel滤波器，即卷积层的每个滤波器都会有自己所关注一个图像特征，比如垂直边缘，水平边缘，颜色，纹理等等，这些所有神经元加起来就是整张图像的特征提取器集合。

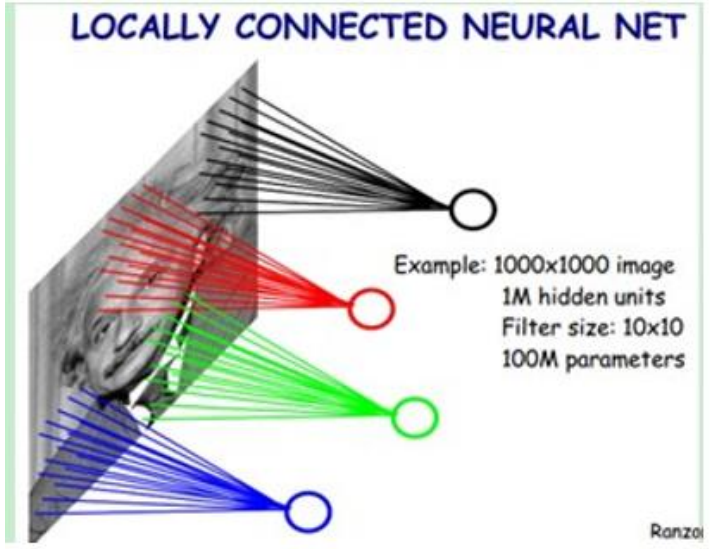
- 需要估算的权重个数减少：AlexNet 1亿 => 3.5w
- 一组固定的权重和不同窗口内数据做内积：卷积



(2)参数共享机制

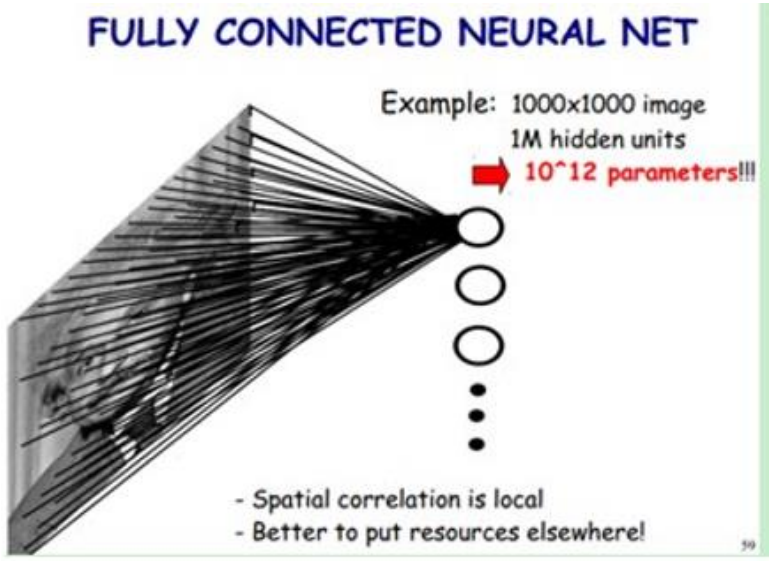
- 如果我们有1000*1000像素的图像，有1百万个隐层神经元，那么他们全连接的话（每个隐层神经元都连接图像的每一个像素点），就有$10^3 \times 10^3 \times 10^6 = 10^{12}$个连接，也就是$10^{12}$个权值参数。
- 然而图像的空间联系是局部的，就像人是通过一个局部的感受野去感受外界图像一样，每一个神经元都不需要对全局图像做感受，每个神经元只感受局部的图像区域，然后在更高层将这些感受不同局部的神经元综合起来就可以得到全局的信息了。这样就可以减少连接的数目，也就是减少神经网络需要训练的权值参数的个数了。

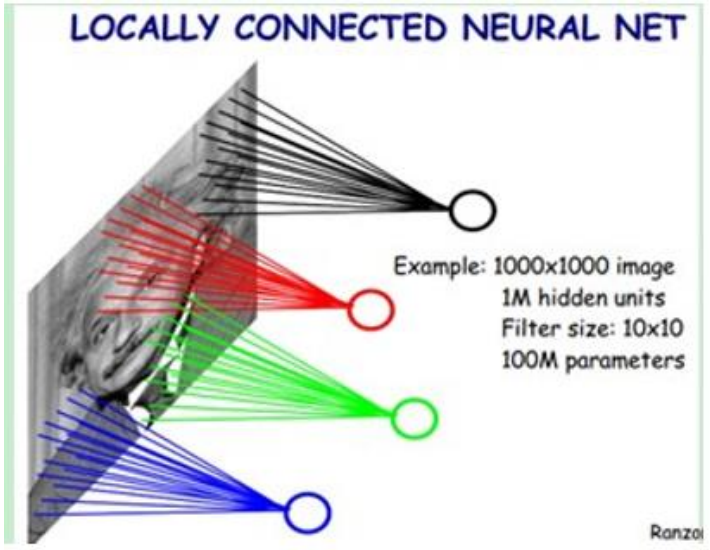




(2)参数共享机制

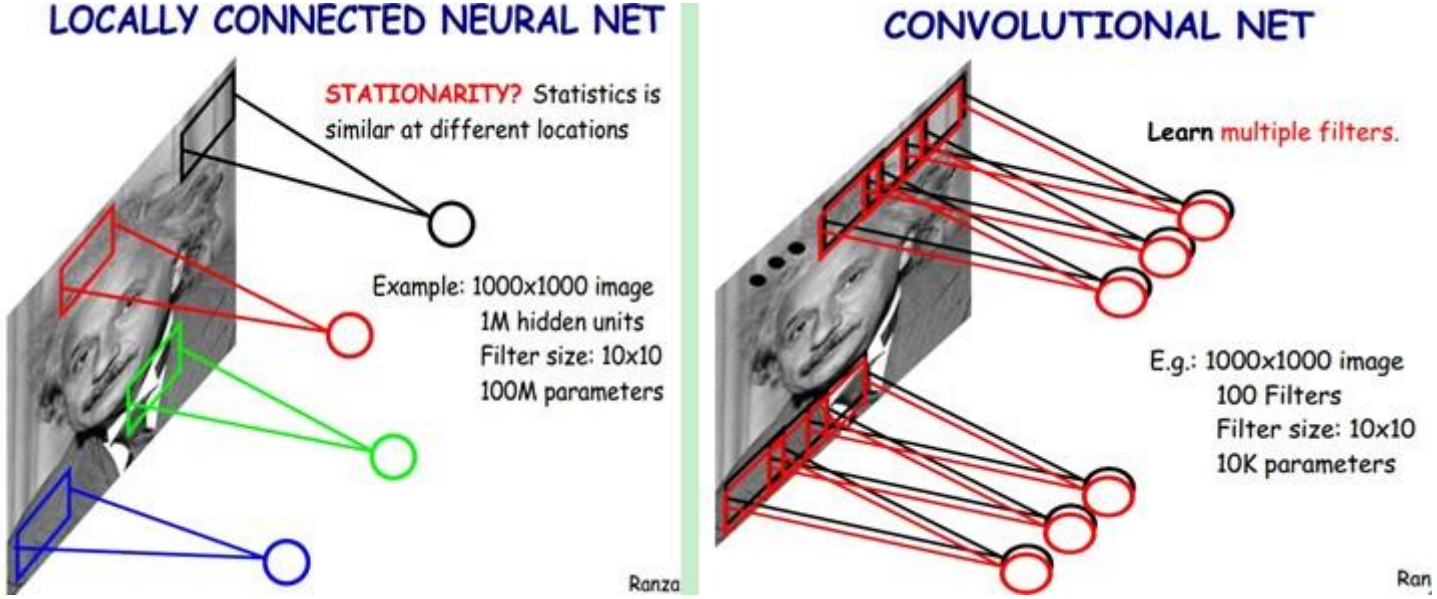
如下图右：假如局部感受野是10*10，隐层每个感受也只需要和这10*10的局部图像相连接，所以1百万个隐层神经元就只有一亿个连接，即10^8个参数。比原来减少了四个0（数量级），如果我们每个神经元这100个参数是相同的呢？也就是说每个神经元用的是同一个卷积核去卷积图像。这样只有100个参数，这就是权值共享！





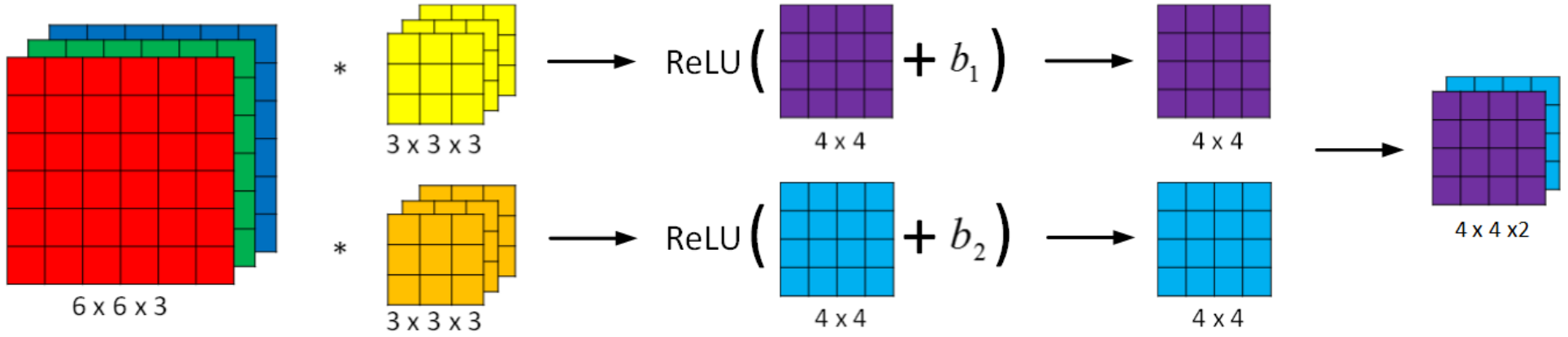
(2)参数共享机制

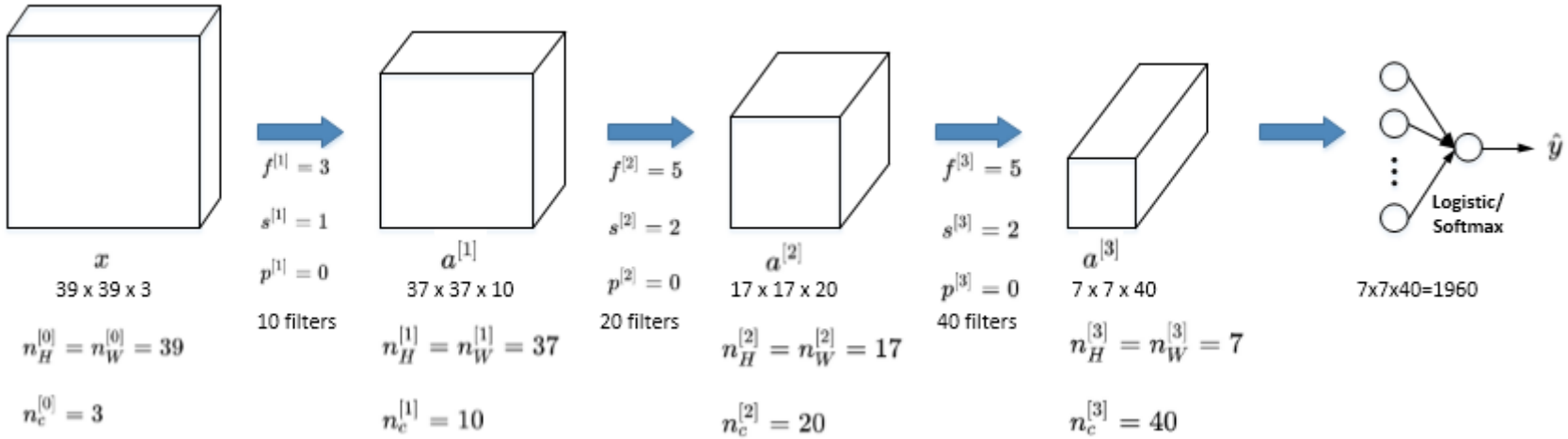
不能只提取一种特征，即需要提取不同的特征，否则特征提取不充分。假设100种滤波器，每种滤波器的参数不一样，表示它提取输入图像的不同特征。每种滤波器去卷积图像就得到对图像的不同特征的映射，称之为Feature Map。故100种卷积核就有100个Feature Map。这100个Feature Map就组成了一层神经元。100种卷积核*每种卷积核共享100个参数=100*100=10K，即1万个参数。见下图右：不同的颜色表达不同的滤波器。



(3)激励层

- 卷积神经网络的单层结构如下所示，整个过程与标准的神经网络单层结构非常类似。



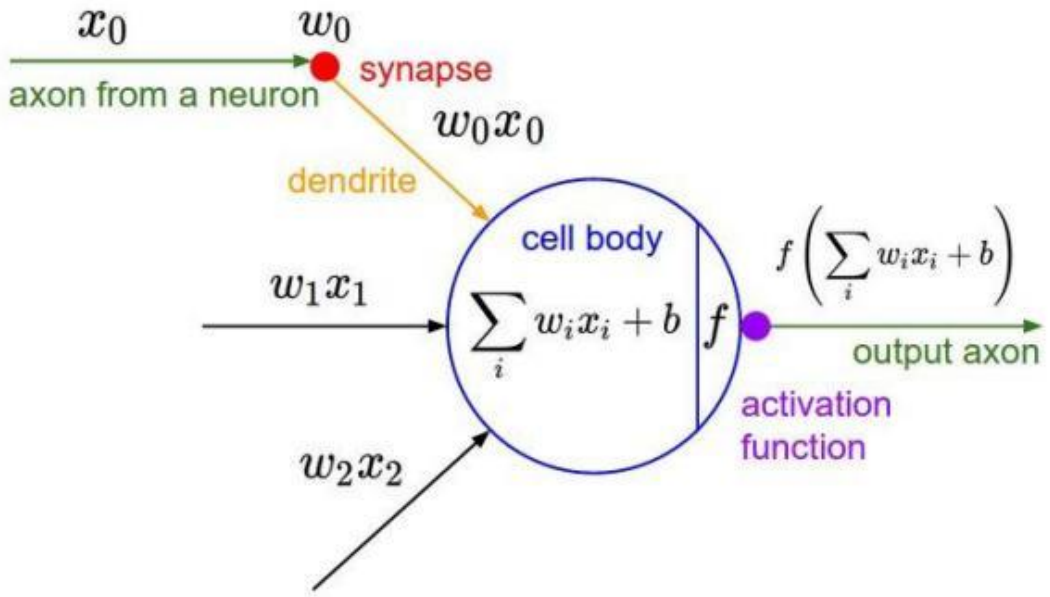


$$H_{out} = \left\lfloor \frac{H_{in} + 2H_{padding} - H_{kernel}}{H_{stride}} \right\rfloor + 1$$

$$W_{out} = \left\lfloor \frac{W_{in} + 2W_{padding} - W_{kernel}}{W_{stride}} \right\rfloor + 1$$

(3)激励层

把卷积层输出结果做非线性映射。



激励层的实践经验：

① 不要用sigmoid！不要用sigmoid！不要用sigmoid！

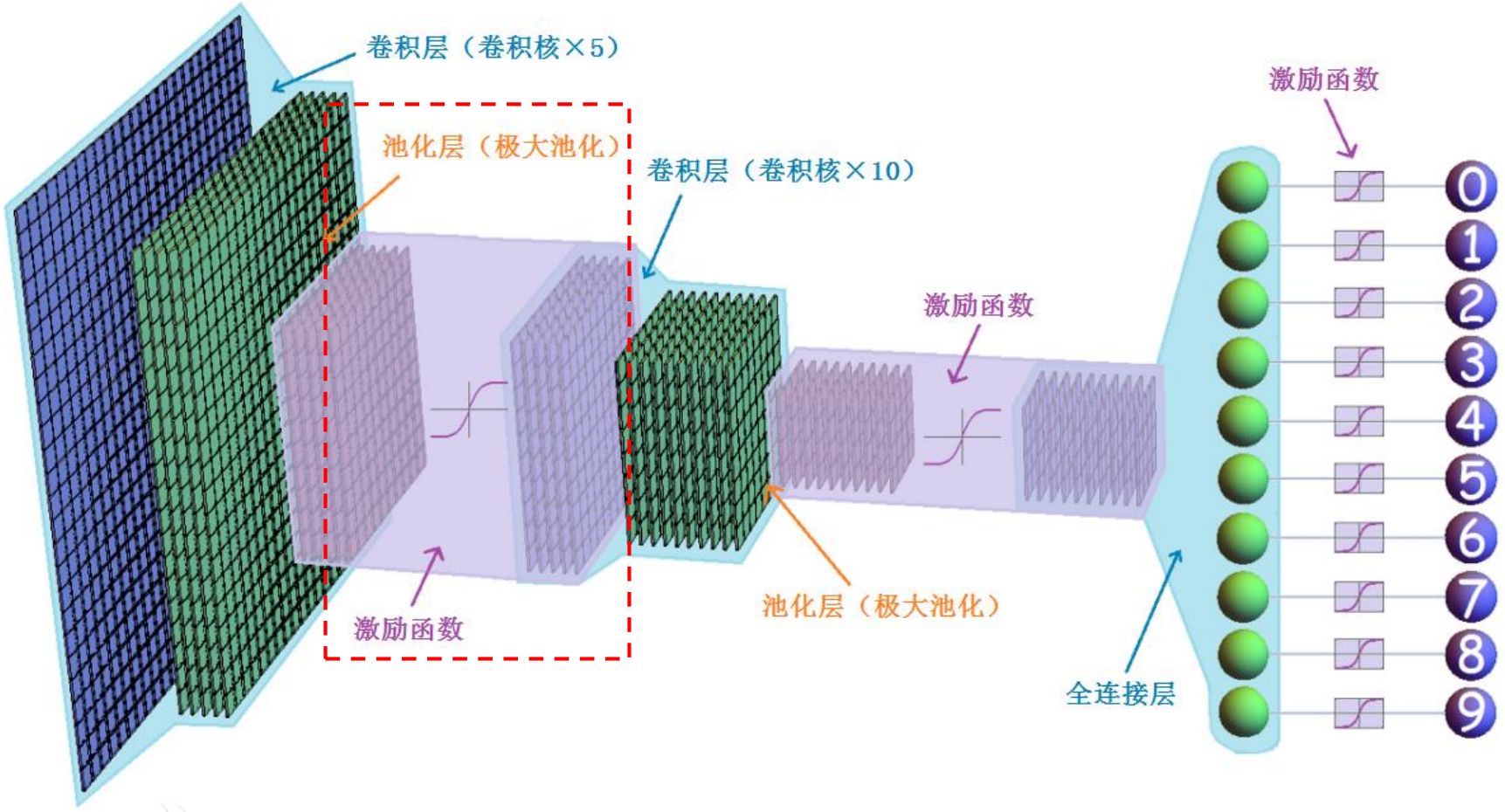
② 首先试RELU，因为快，但要小心点

③ 如果2失效，请用Leaky ReLU或者Maxout

④ 某些情况下tanh倒是有不错的结果，但是很少

4. 池化层

池化层夹在连续的卷积层之间， 用于压缩数据和参数的量，减小过拟合。简而言之，如果输入的是图像，那么池化层的最主要作用就是压缩图像。

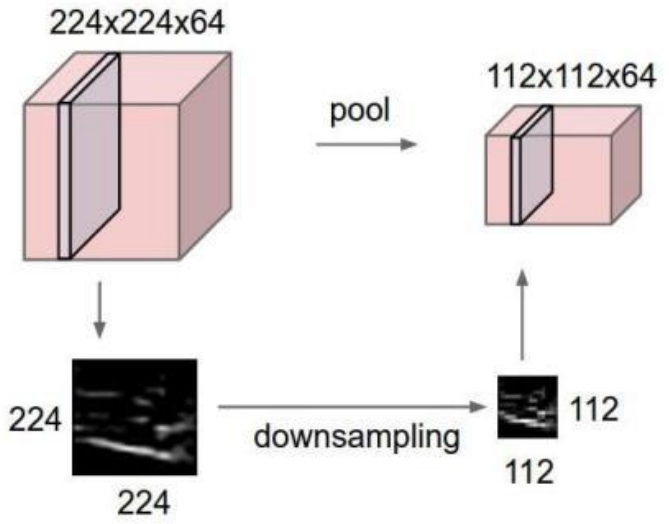


池化层的作用

1. 特征不变性：即特征尺度不变性，池化操作就是图像的resize。如一张狗的图像被缩小了一倍还能认出是一张狗的照片，说明图像中仍保留狗最重要的特征，图像压缩时去掉的信息并不重要，而留下的信息具有尺度不变性的特征，最能表达图像的特征。

2. 特征降维：一幅图像含有的信息很大且特征很多，但是有些信息对于图像任务没有太多用途或者有重复，可把这类冗余信息去除，把最重要的特征抽取出来。

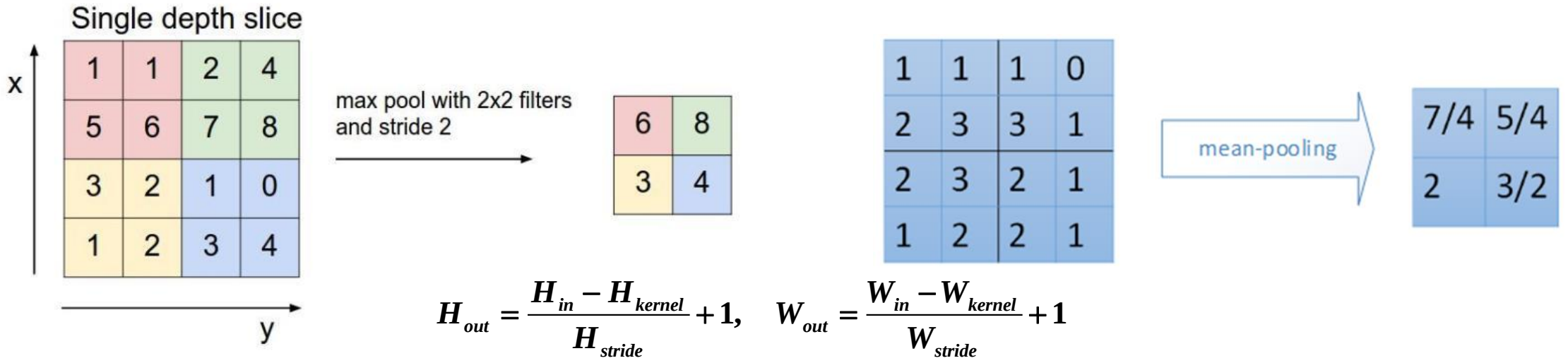
3. 在一定程度上防止过拟合，更方便优化。



池化层在某种程度上可以补偿图像中偏离中心和倾斜的对象。

池化层的计算

- 池化层用的方法有Max pooling 和 average pooling，而实际用的较多的是Max pooling。

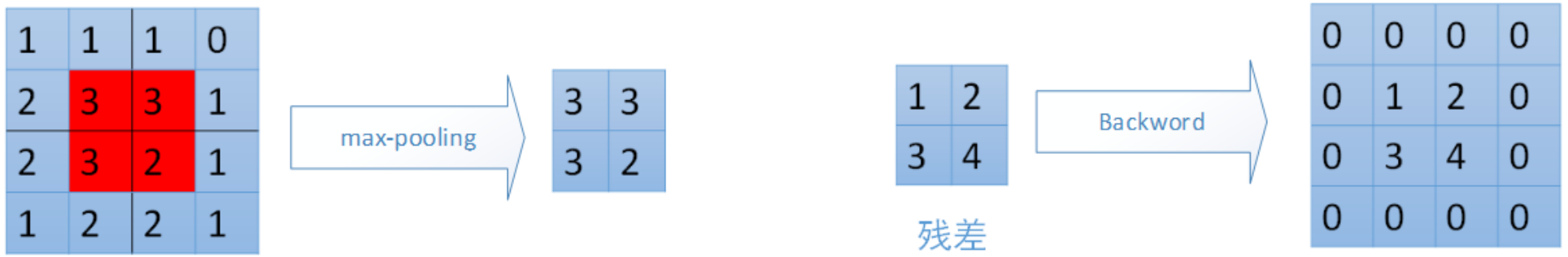


$$H_{out} = \frac{H_{in} - H_{kernel}}{H_{stride}} + 1, \quad W_{out} = \frac{W_{in} - W_{kernel}}{W_{stride}} + 1$$

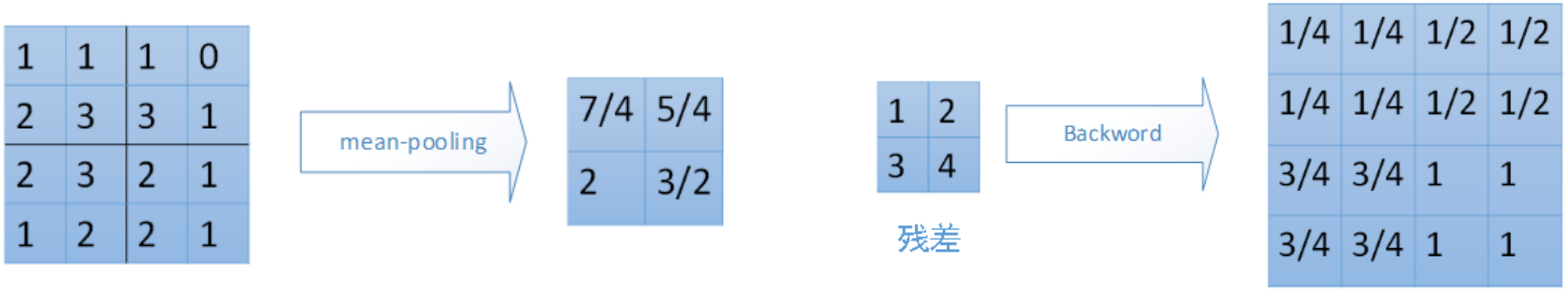
- Max pooling对于每个2*2的窗口选出最大的数作为输出矩阵的相应元素的值，比如输入矩阵第一个2*2窗口中最大的数是6，那么输出矩阵的第一个元素就是6，如此类推。
- average pooling对于每个2*2的窗口选出该窗口所有元素的均值数作为输出矩阵的相应元素的值，比如输入矩阵第一个2*2窗口中均值数是(1+1+2+3)/4 = 7/4，那么输出矩阵的第一个元素就是7/4 ，如此类推。

池化的反向传播

对于max-pooling，在前向计算时，是选取的每个2*2区域中的最大值，这里需要记录下最大值在每个小区域中的位置。在反向传播时，只有那个最大值对下一层有贡献，所以将残差传递到该最大值的位置，区域内其他2*2-1=3个位置置零。具体过程如下图，其中4*4矩阵中非零的位置即为前边计算出来的每个小区域的最大值的位置。



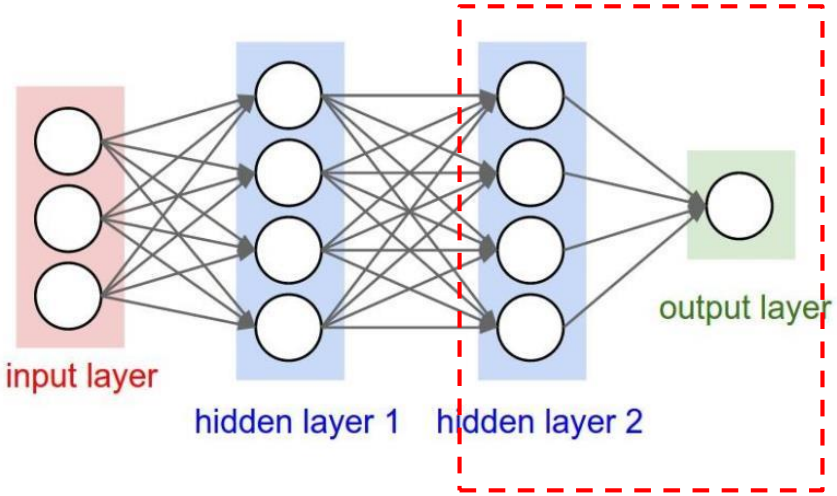
对于mean-pooling，需要把残差平均分成2*2=4份，传递到前边小区域的4个单元即可。具体过程如图：

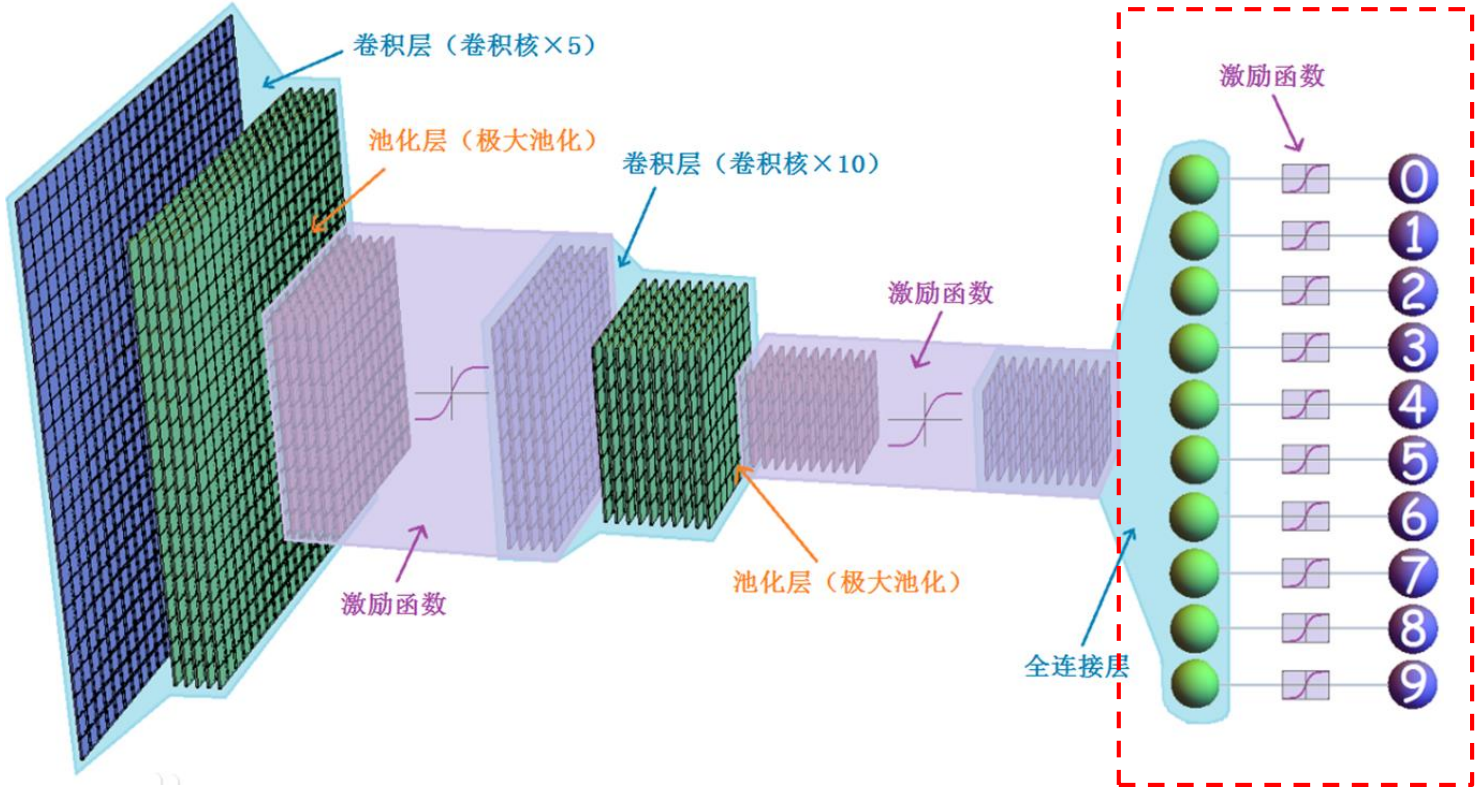


5. 全连接层

信阳师范学院
数学与统计学院

两层之间所有神经元都有权重连接，通常全连接层在卷积神经网络尾部，与传统的神经网络神经元的连接方式是一样的：





6. 卷积神经网络MNIST示例

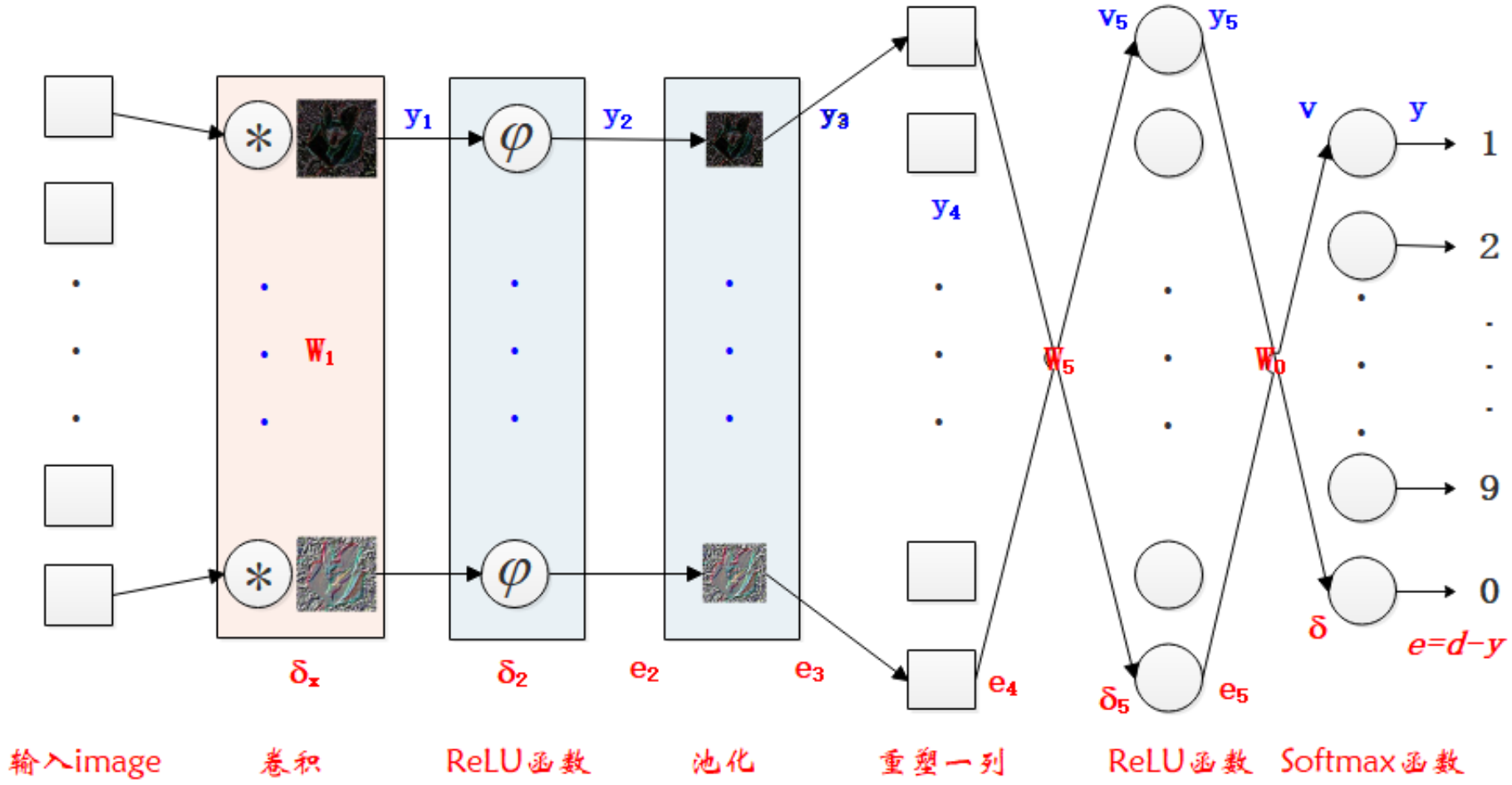
- 60000幅图像用于训练，10000幅图像用于测试，每幅都是28*28黑白图像。
- 网址：http://yann.lecun.com/exdb/mnist/
- 网络结果如下表所示：

层	备注	激活函数	层	备注	激活函数
输入层	28*28个节点		隐含层	100个节点	ReLU
卷积层	20个9*9卷积滤波器	ReLU	输出层	10个节点	Softmax
池化层	一个平均池化（2*2）				

识别MNIST手写数字的CNN

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

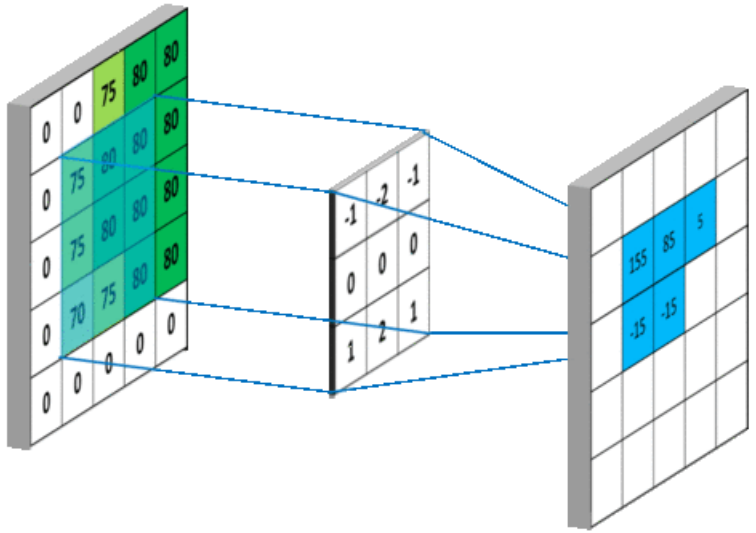
卷积神经网络算法设计的架构和变量的定义

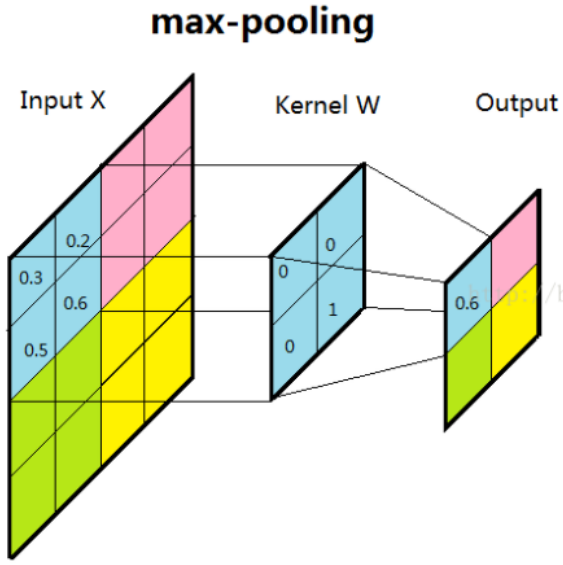


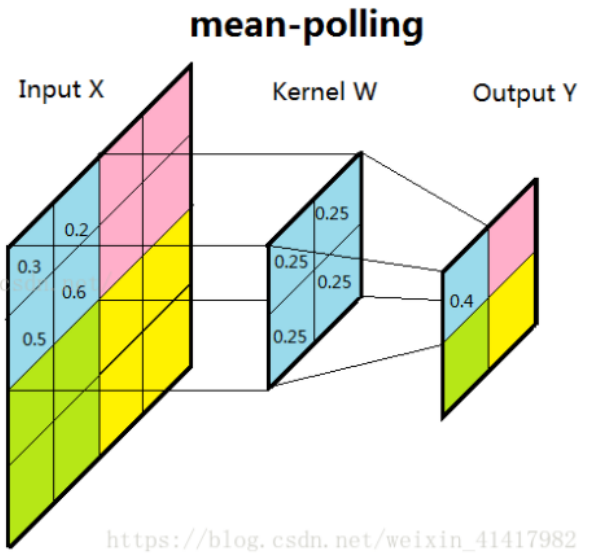
卷积运算函数Conv与池化运算函数Pool

```
function y = Conv(x, W)
%Conv函数实现卷积运算，输入W为输入层到卷积层权重，x是一幅图像，如28*28
    % W是三维矩阵，mrow和wcol代表卷积核的大小，numFilters代表滤波器的数量
    [wrow, wcol, numFilters] = size(W);
    [xrow, xcol, ~ ] = size(x);
    yrow = xrow - wrow + 1; %卷积运算后图像维度
    ycol = xcol - wcol + 1;
    y = zeros(yrow, ycol, numFilters); %卷积后结果
    for k = 1:numFilters %分别对每个卷积核进行卷积运算
        filter = W(:,:,k);
        %squeeze除去size为1的维度，旋转180°
        filter = rot90(squeeze(filter), 2);
        % valid返回在卷积过程中，未使用边缘补0部分进行计算的卷积结果部分
        y(:,:,k) = conv2(x, filter, 'valid');
    end
end
```

```
function y = Pool(x)
% Pool函数实现池化层功能，2x2 mean pooling
% x为池化层经过ReLU激活后的结果
    [xrow, xcol, numFilters] = size(x);
    y = zeros(xrow/2, xcol/2, numFilters); %池化后的图像维度
    for k = 1:numFilters
        filter = ones(2) / (2*2);   % for mean
        % valid返回在卷积过程中，未使用边缘补0部分进行计算的卷积结果部分
        image = conv2(x(:,:,k), filter, 'valid');
        y(:,:,k) = image(1:2:end, 1:2:end);
    end
end
```







6. 卷积神经网络MATLAB算法实现

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

MnistConv.m

```
function [W1,W5,Wo] = MnistConv(X,D,epochs,test,testlabel)
%卷积神经网络，反向传播训练，使用动量法更新权重，小批量随机梯度下降法
%卷积层权重W1，W5（池化层到隐藏层）和Wo（隐藏层到输出层）为分类神经网络的连接
%X是特征数据，三维矩阵，D为目标数据标签

    %% 权重初始化及变量初始化
    %乘0.01是因为要把W随机初始化到一个相对较小的值，使得加权和不至于过大
    W1 = 1e-2*randn([9 9 20]); %20个9*9卷积滤波器
    [mx,nx,~] = size(X);
    ms = ((mx - 9 + 1) - 2)/2 + 1; %卷积池化后的图像尺寸
    ns = ((nx - 9 + 1) - 2)/2 + 1;
    %Xavier初始化权重方法
    W5 = (2*rand(100,ms*ns*20)-1) * sqrt(6)/sqrt(100+ms*ns*20);
    Wo = (2*rand(10,100)-1) * sqrt(6)/ sqrt(10+100);
    alpha = 0.01; %学习率
    beta  = 0.95; %动量参数
    momentum1 = zeros(size(W1)); %动量法调整权重，对应卷积层
    momentum5 = zeros(size(W5)); %对应池化层到隐藏层
    momentumo = zeros(size(Wo)); %对应隐藏层到输出层
    N = length(D); %数据量

    %% 小批量算法
    %各批量数据bsize为100，权重在每一轮训练中都被调整N/100次。
    bsize = 100;
    %每一小批量数据点的第一个点的索引
    blist = 1:bsize:(N-bsize+1);

    for epoch = 1:epochs
        %% One epoch loop
        for batch = 1:length(blist)  %循环N/100次
            dW1 = zeros(size(W1));
            dW5 = zeros(size(W5));
            dWo = zeros(size(Wo));
            %% Mini-batch loop
            %对100个权重更新值求和并取平均值
            begin = blist(batch); %获得每批数据的起点
            for k = begin:begin+bsize-1 %对应每批数据的起点和终点索引
                %% 前向传播算法
                x  = X(:,:,k);   % Input：28x28的黑白图像
                y1 = Conv(x, W1); % Convolution，20个9*9卷积滤波器：20x20x20
                y2 = ReLU(y1);   %线性修正单元激活函数
                y3 = Pool(y2);   % Pooling，使用平均池化2*2：10x10x20
                y4 = reshape(y3,[],1);  % 2000*1列向量
                v5 = W5*y4;  % ReLU, W5为100*2000，得到100个隐藏层输出
                y5 = ReLU(v5); % 隐藏层使用ReLU
                v  = Wo*y5;    % Softmax, Wo为10*010，得到输出层10x1
                y  = Softmax(v); %输出层使用激活函数Softmax
                %% One-hot编码
                d = zeros(10,1);
                d(sub2ind(size(d),D(k),1)) = 1;
                %% Backpropagation，反向传播调整权重
                e = d - y;    % Output layer误差
                delta = e;
                e5 = Wo' * delta;   % Hidden(ReLU) layer
```

6. 卷积神经网络MATLAB算法实现

```
54 —                delta5 = (y5 > 0) .* e5; %ReLU一阶导数
55 —                e4 = W5' * delta5;  % Pooling layer
56 —                e3 = reshape(e4, size(y3));
57                  %池化层——ReLU——卷积层反向传播过程
58 —                e2 = zeros(size(y2));
59 —                W3 = ones(size(y2))/(2*2);
60 —                for c = 1:20
61 —                    e2(:,:,c) = kron(e3(:,:,c), ones([2 2])) .* W3(:,:,c);
62 —                end
63 —                delta2 = (y2 > 0) .* e2;  % ReLU layer, ReLU一阶导数
64 —                delta1_x = zeros(size(W1)); % Convolutional layer
65 —                for c = 1:20
66 —                    delta1_x(:,:,c) = conv2(x(:,:), ...
67                      rot90(delta2(:,:,c),2),'valid');
68 —                end
69 —                dW1 = dW1 + delta1_x;  %卷积层权重更新增量
70 —                dW5 = dW5 + delta5*y4'; %池化层到隐藏层权重更新增量
71 —                dWo = dWo + delta *y5'; %隐藏层到输出层权重更新增量
72 —            end
73              %% 动量法更新权重
74 —            dW1 = dW1 / bsize;
75 —            dW5 = dW5 / bsize;
76 —            dWo = dWo / bsize;
77 —            momentum1 = alpha * dW1 + beta * momentum1;
78 —            W1 = W1 + momentum1;
79 —            momentum5 = alpha * dW5 + beta * momentum5;
80 —            W5 = W5 + momentum5;
81 —            momentumo = alpha * dWo + beta * momentumo;
82 —            Wo = Wo + momentumo;
```

```
83 —            end
84 —            fprintf('第【%d】次训练中.....\n',epoch);
85 —        end
86
87          %% 测试
88 —        if nargin == 5
89 —            acc = 0;
90 —            N = length(testlabel);
91              %使用训练好的权重检测分类正确率
92 —            for k = 1:N
93 —                x = test(:,:,k);    % Input, 28x28
94 —                y1 = Conv(x, W1);  % Convolution,  20x20x20
95 —                y2 = ReLU(y1);
96 —                y3 = Pool(y2);  % Pool,10x10x20
97 —                y4 = reshape(y3, [], 1);    %2000
98 —                v5 = W5*y4; % ReLU,100
99 —                y5 = ReLU(v5);
100 —               v  = Wo*y5; % Softmax,10
101 —               y  = Softmax(v);
102 —               [~, i] = max(y);
103 —               if i == testlabel(k)
104 —                   acc = acc + 1;
105 —               end
106 —           end
107 —           acc = acc / N*100;
108 —           fprintf('Accuracy is %f\n', acc);
109 —       end
110 —   end
```

训练卷积神经网络

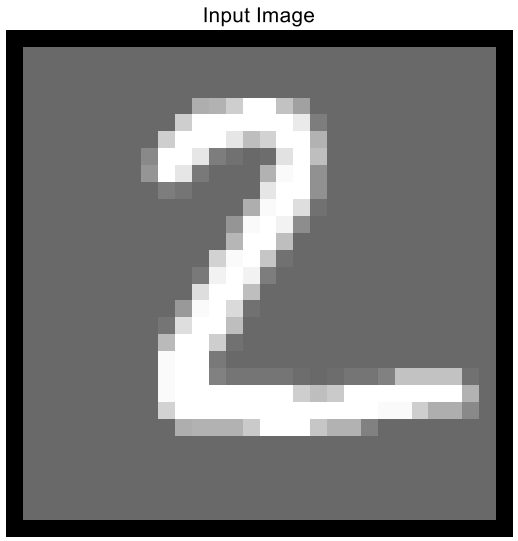
```
% 训练，60000组数据
trainImages = loadMNISTImages('MNIST\train-images.idx3-ubyte');
trainImages = reshape(trainImages, 28, 28, []);
trainLabels = loadMNISTLabels('MNIST\train-labels.idx1-ubyte');
trainLabels(trainLabels == 0) = 10;    % 0 --> 10
% 测试，10000组数据
testImages = loadMNISTImages('MNIST\t10k-images.idx3-ubyte');
testImages = reshape(testImages, 28, 28, []);
testLabels = loadMNISTLabels('MNIST\t10k-labels.idx1-ubyte');
testLabels(testLabels == 0) = 10;    % 0 --> 10
[W1, W5, Wo] = MnistConv(trainImages,trainLabels,5,testImages,testLabels);
```

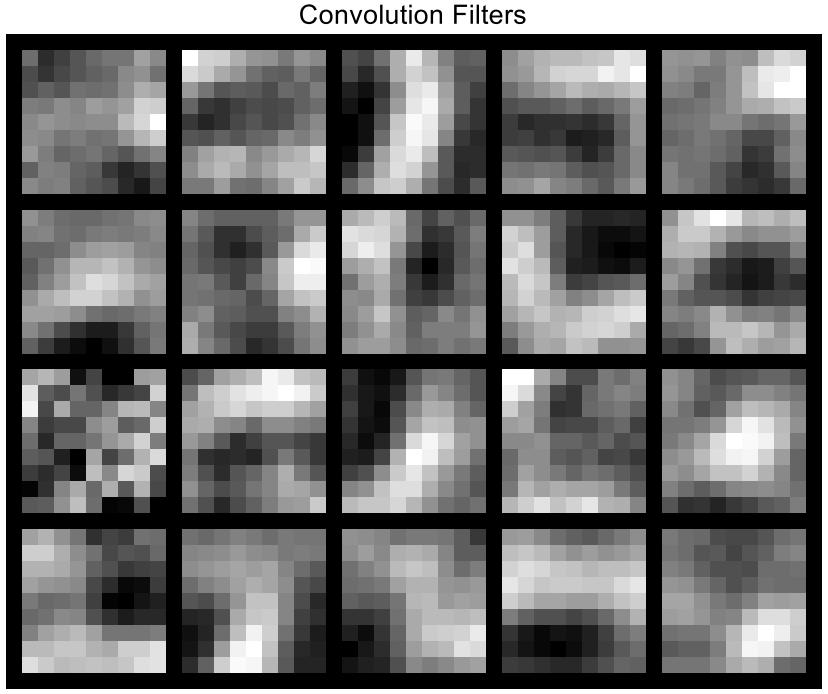
```
>> [W1, W5, Wo] = MnistConv(trainImages, trainLabels, 5, testImages, testLabels);
第【1】次训练中.....
第【2】次训练中.....
第【3】次训练中.....
第【4】次训练中.....
第【5】次训练中.....
Accuracy is 98.420000
```

```
% 再次训练测试
>> TestMnistConv
第【1】次训练中.....
第【2】次训练中.....
第【3】次训练中.....
第【4】次训练中.....
第【5】次训练中.....
Accuracy is 98.180000
```

卷积神经网络MNIST示例

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS





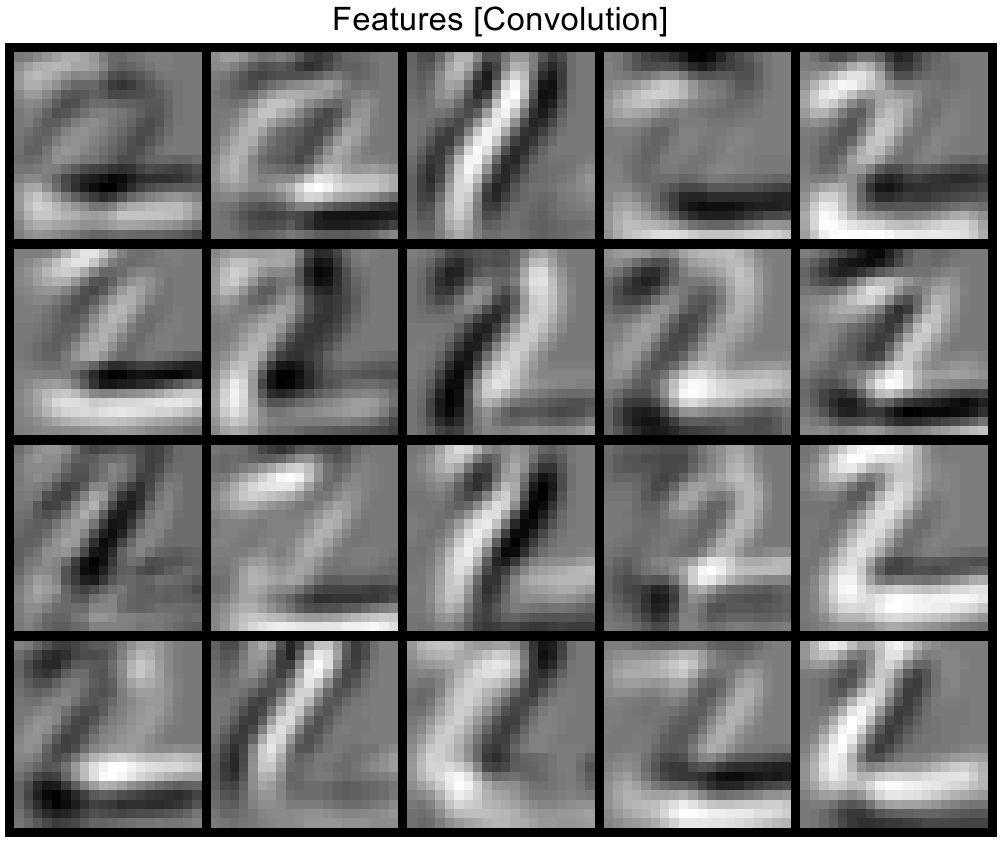
第一张图28*28的输入图像“2”；

第二张图包含20个训练后的卷积过滤器。每一个过滤器都是9*9的图像，并用灰度色调显示每个元素值。数值越大，色调越明亮。这些过滤器就是卷积神经网络想要从MNIST图像中提取的最好的特征。

卷积神经网络MNIST示例

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

卷积层对图像的处理结果(y1)，这个特征映射包括20个20*20的图像，从图像中可以明显看到因卷积过滤器而使输入图像产生的各种变化。



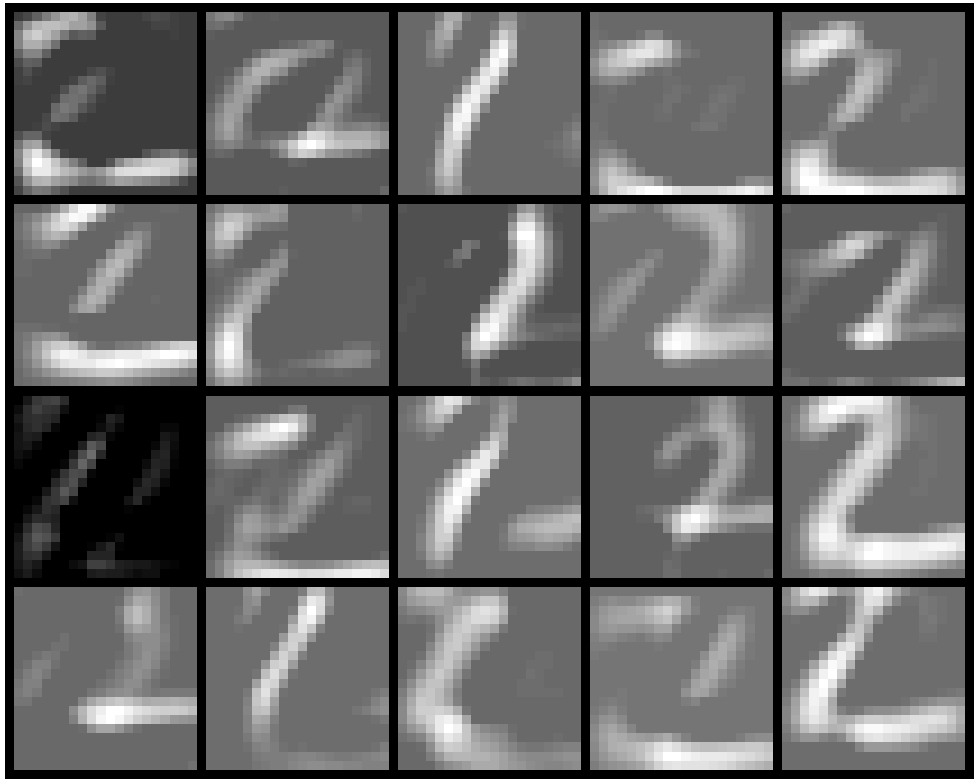
卷积神经网络MNIST示例

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

第四张图是ReLU函数对来自于卷积层的特征映射进行处理后的结果。从图中可以看到，其前一张图像中暗色像素被移除了，当前图像在文字上面大多是白色像素。

如第一行第一列和第三行第一列的图，图像“2”的特征不明显，需要更多的数据和训练来改善这个图像。

Features [Convolution + ReLU]

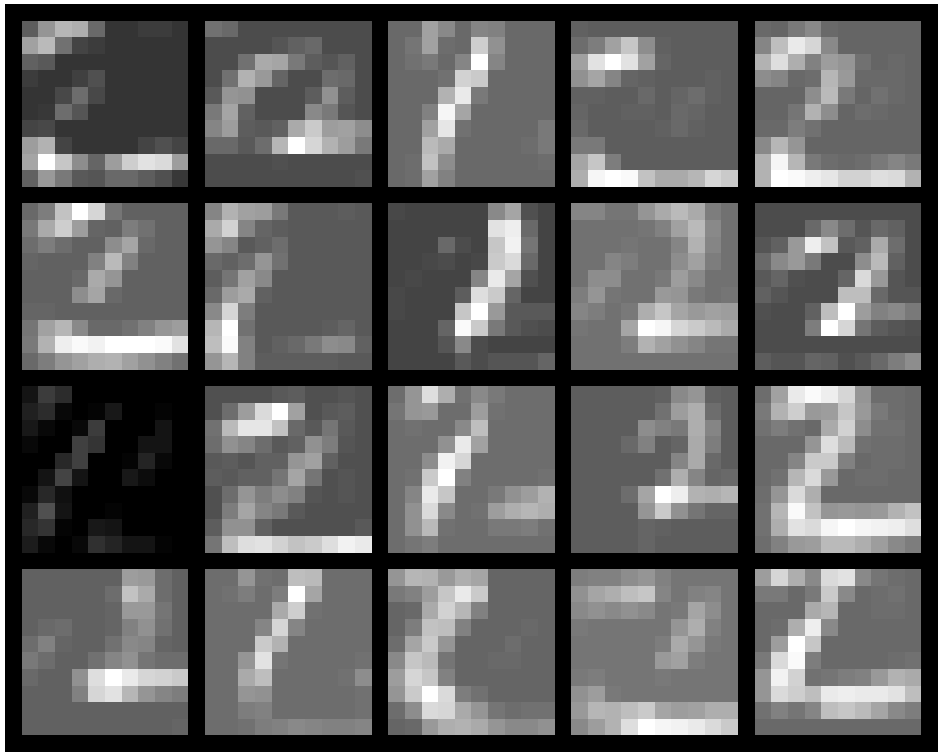


卷积神经网络MNIST示例

信阳师范学院
数学与统计学院
SCHOOL OF MATHEMATICS AND STATISTICS

第五张图提供了对ReLU层的输出进行平均池化处理后的图像。其中每一个图像都以10*10的空间继承了前一个图像的形状，并且尺寸是前一个图像的一半，这展示了池化层可以减少所需计算资源的程度。

Features [Convolution + ReLU + MeanPool]




01
IDEA
02
LEARNING
03
CREATIVE
INSPIRATION
EDUCATION
BUSINESS
PHOTO BOOK

感谢聆听