



信阳师范学院  
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

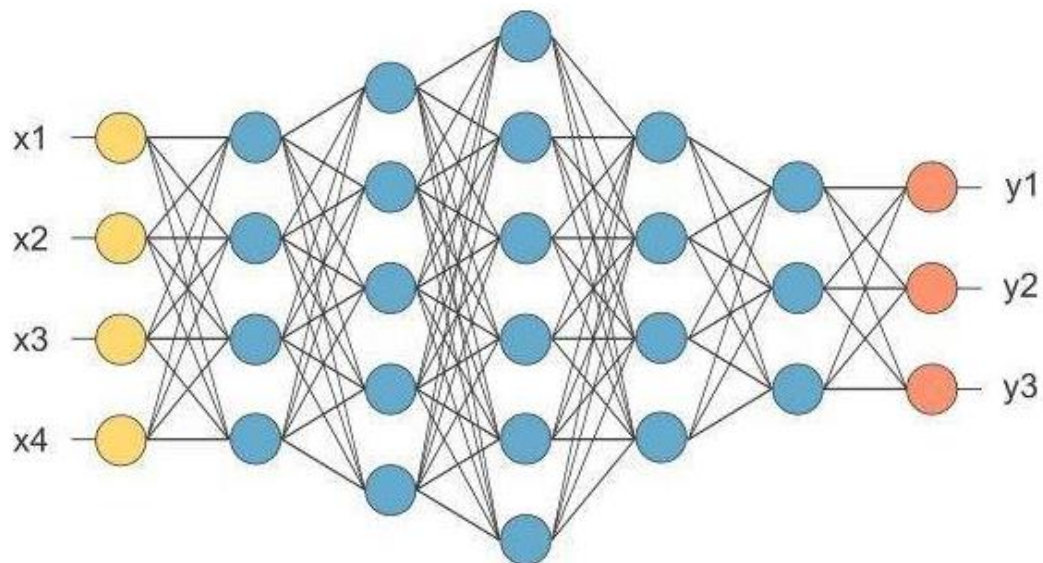
# 目录

## CONTENTS

- A 机器学习简介
- B 单层神经网络
- C 多层神经网络
- D 神经网络及其分类
- E 深度学习
- F 卷积神经网络

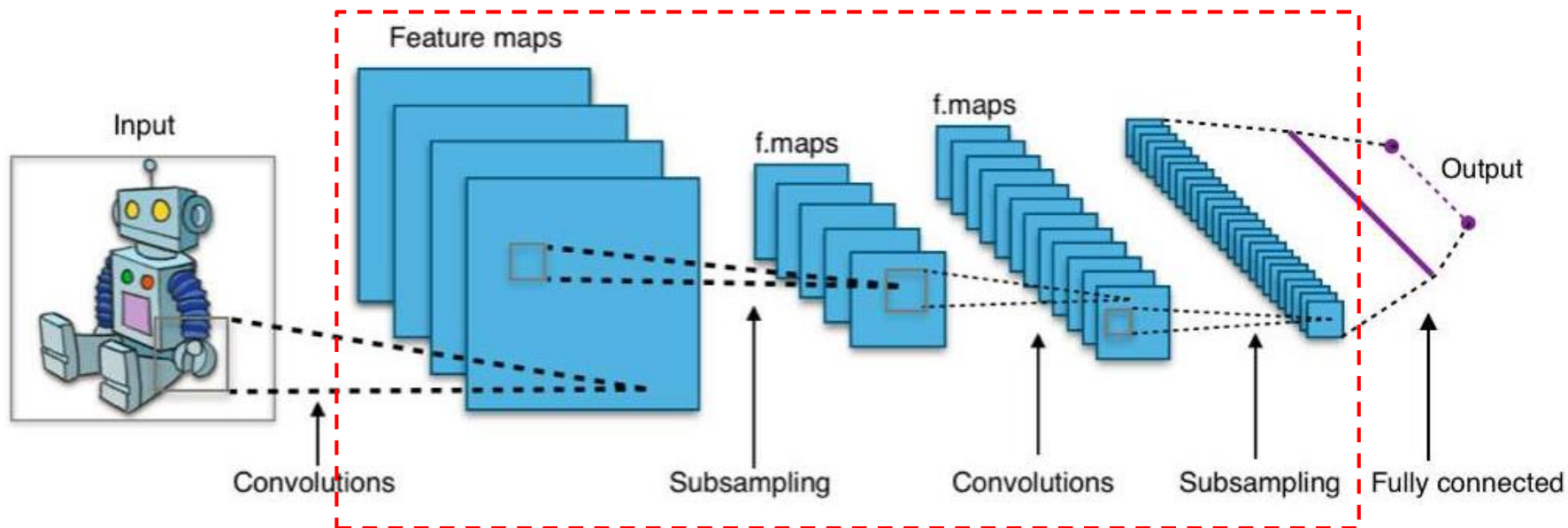


# 1. 卷积神经网络概述



普通深度神经网络，隐藏层层数增多而已。

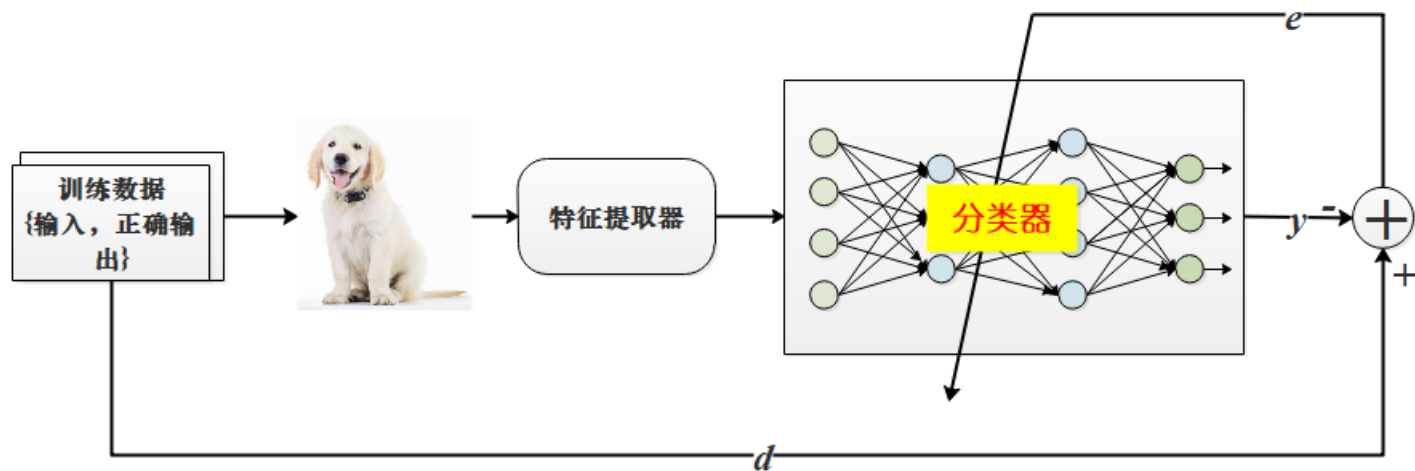
卷积神经网络：输入层、卷积层、激活函数、池化层、全连接层。



# 1. 卷积神经网络概述

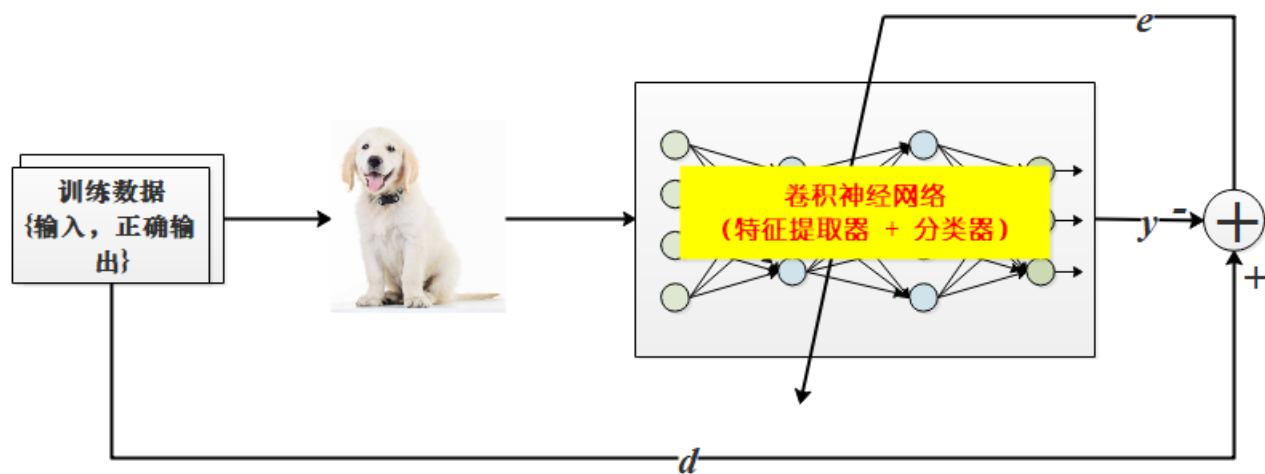
- 深度神经网络的重要性在于它打开了通向知识分层处理的复杂非线性模型和系统性方法的大门。卷积神经网络(ConvNet)是专门做图像识别的深度神经网络。从2012年开始，它引领着大多数的计算机视觉领域。
- 卷积神经网络不只是有很多隐藏层的深度神经网络，它也是一种模仿大脑视觉皮质进行图像处理和识别图像的深度网络。
- 图像识别基本上算是一种分类问题，卷积神经网络输出层通常采用多分类神经网络。然而，直接将原始图像用于图像识别而不考虑识别方法将会导致很差的结果。为了对比图像特征，需要提取处理图像。如第4节、第5节所使用的MNIST手写识别数字原始图像，分类结果较为有效，这是因为它们只是简单的黑白色块。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

# 1. 卷积神经网络概述



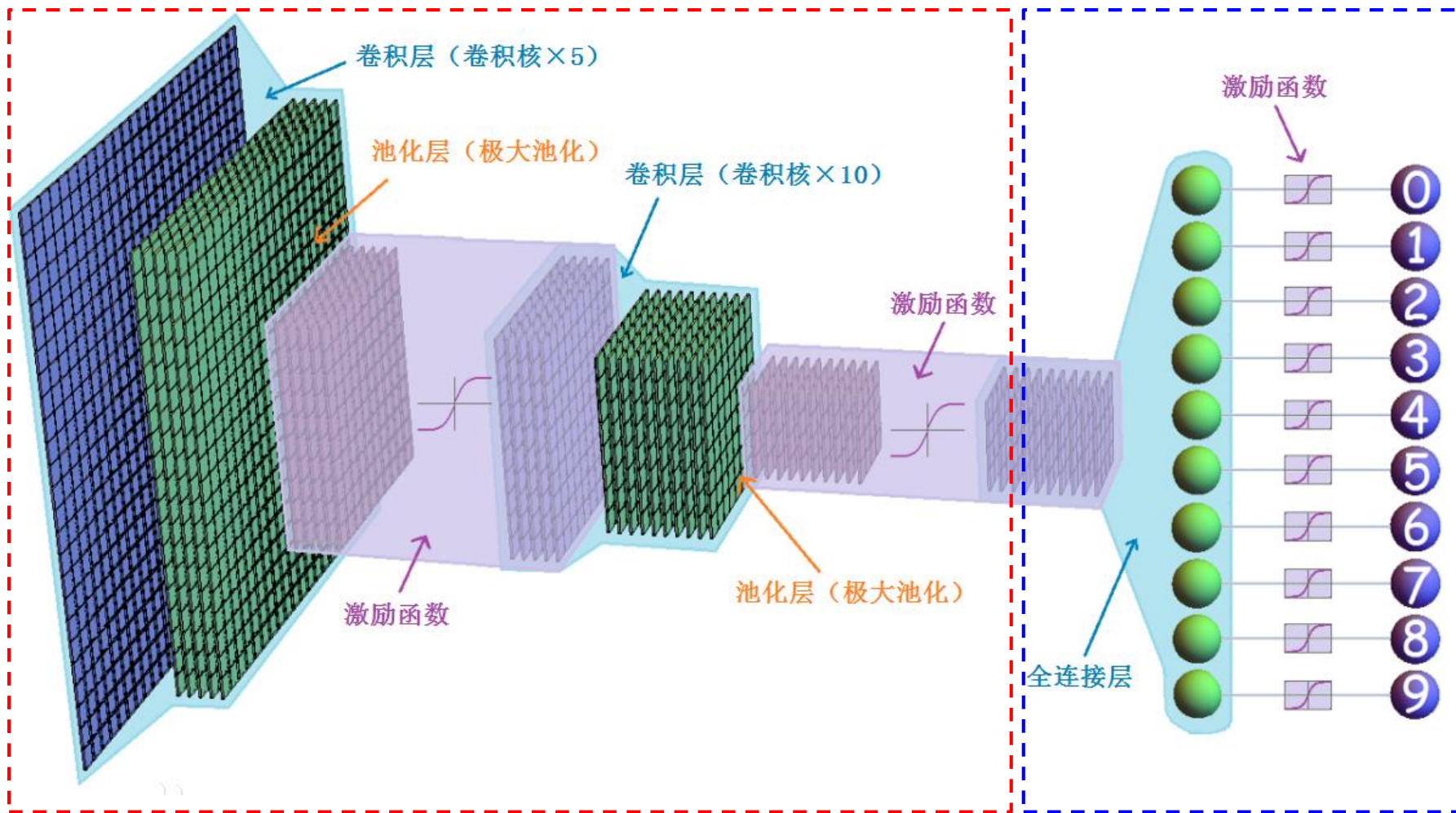
在神经网络之前，特征提取器根据使用领域不同，被设计成具有不同功能。它可能产生不同水平的学习性能。但是，这些特征提取器与机器学习是相互独立的。

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

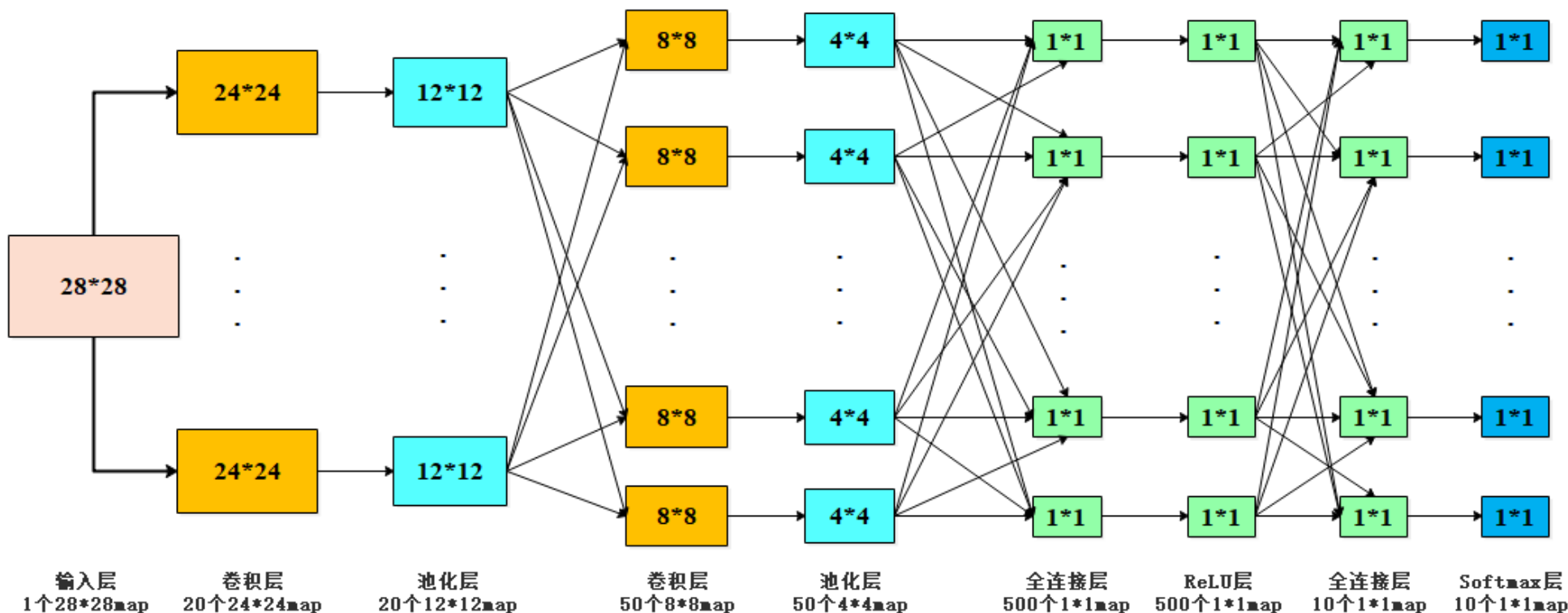
CNN包含提取输入图像特征的神经网络和另外一个进行图像分类的神经网络。

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

$$h_{\text{记忆}}(t) = f_{\text{认知}}(t) * g_{\text{遗忘}}(t)$$

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

$$= \int_0^{+\infty} f_{\text{认知}}(\tau)g_{\text{遗忘}}(t-\tau)d\tau$$

- 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

```
conv_test.m x +
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



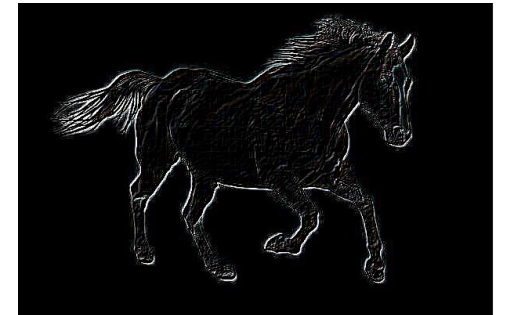
水平边缘



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



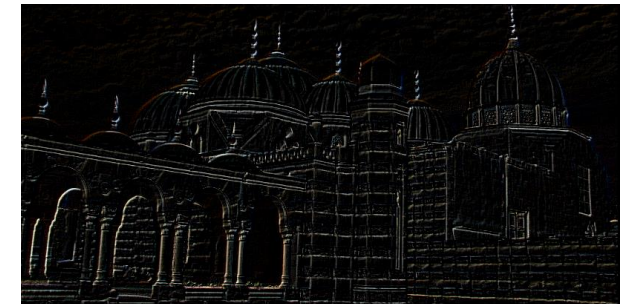
45度的浮雕滤波器



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

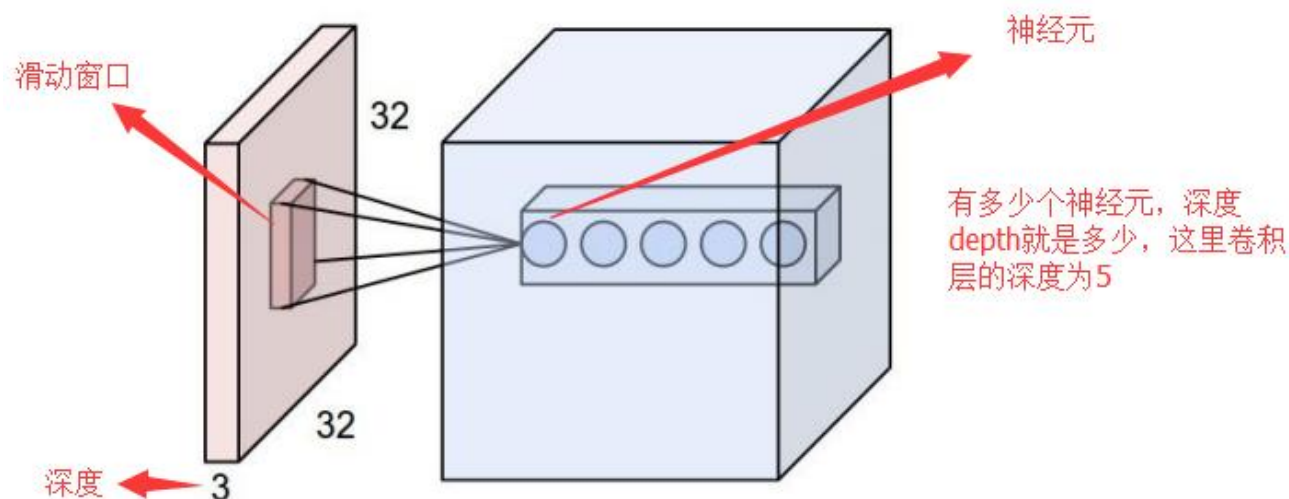
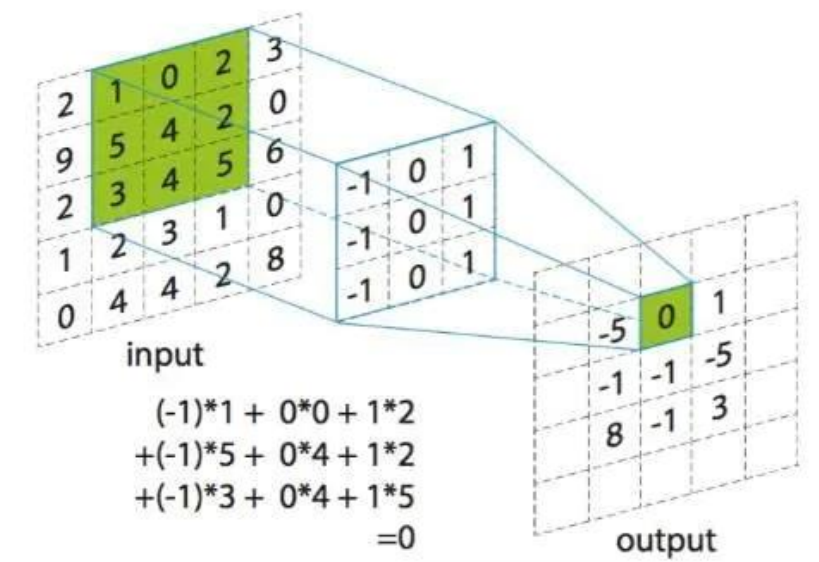


均值模糊



### 3. 卷积层

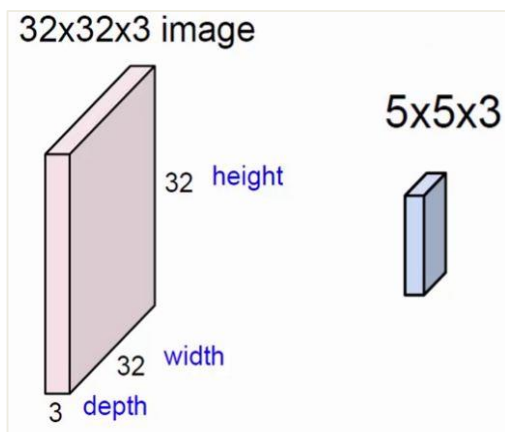
- 卷积层两个关键操作：
  - 局部关联**：每个神经元可看做一个滤波器(filter)，第 $i$ 层的每一个神经元都只和第 $i - 1$ 层的一个局部窗口内的神经元相连，构成一个局部连接网络。
  - 窗口(receptive field)滑动**，filter对局部数据计算。
- 卷积层相关概念：
  - 深度/depth；步长/stride（窗口一次滑动的长度）；填充值/zero-padding



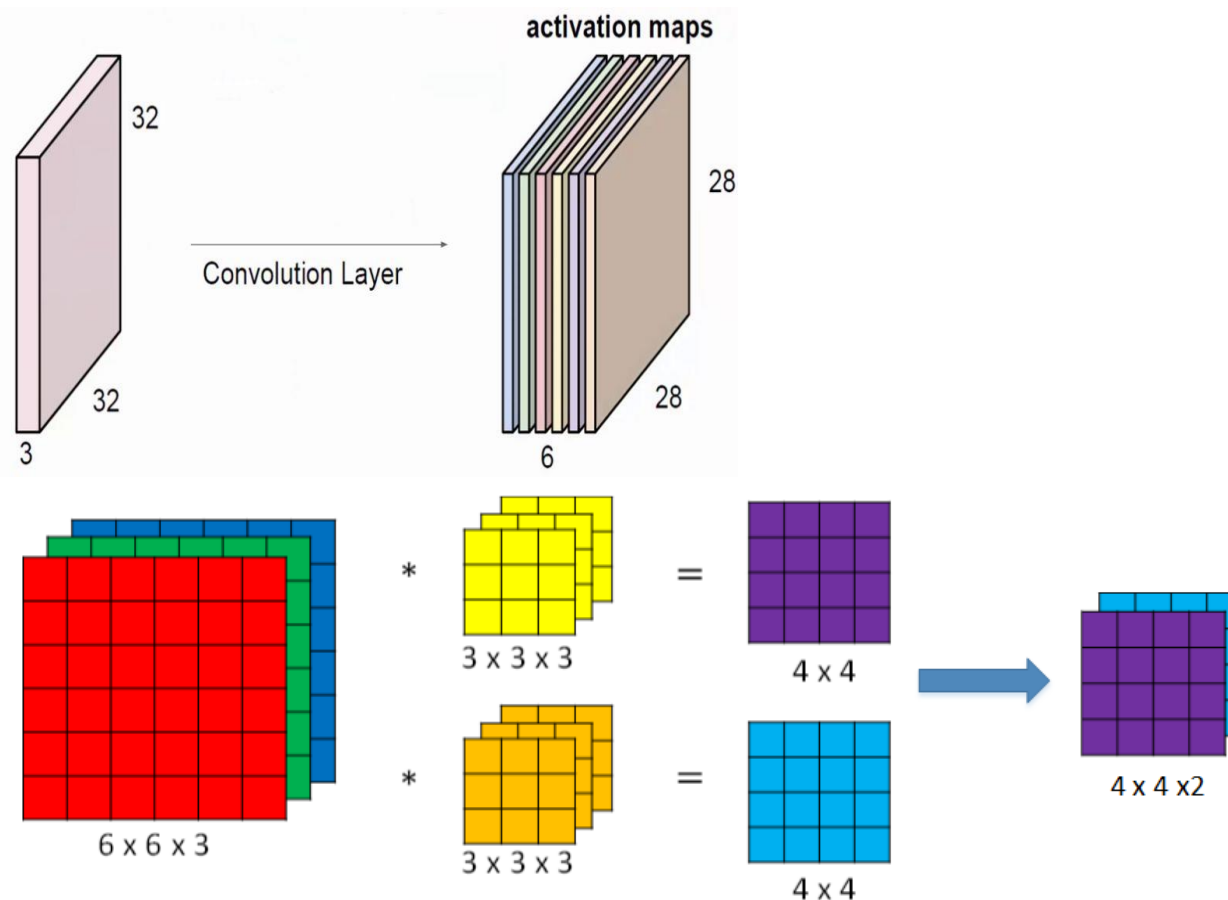
### 3. 卷积层



左图左方块是输入层，尺寸为 $32 \times 32$ 的3通道图像，小方块是filter（卷积核），尺寸为 $5 \times 5$ ，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。



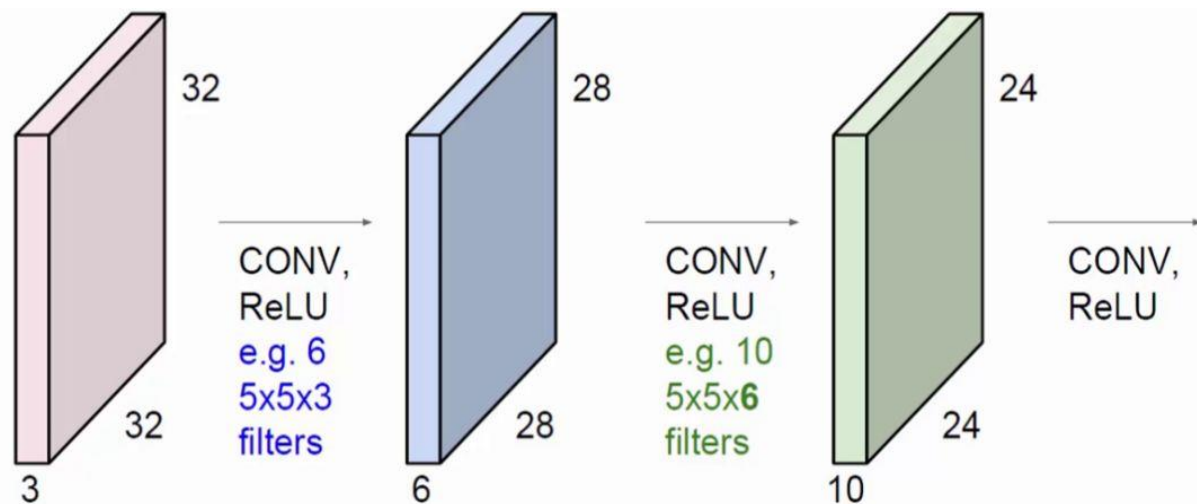
如右下图：为了进行多个卷积运算，实现更多边缘检测，可以增加更多的滤波器组。例如设置第一个滤波器组实现垂直边缘检测，第二个滤波器组实现水平边缘检测。不同滤波器组卷积得到不同的输出，个数由滤波器组决定。



### 3. 卷积层



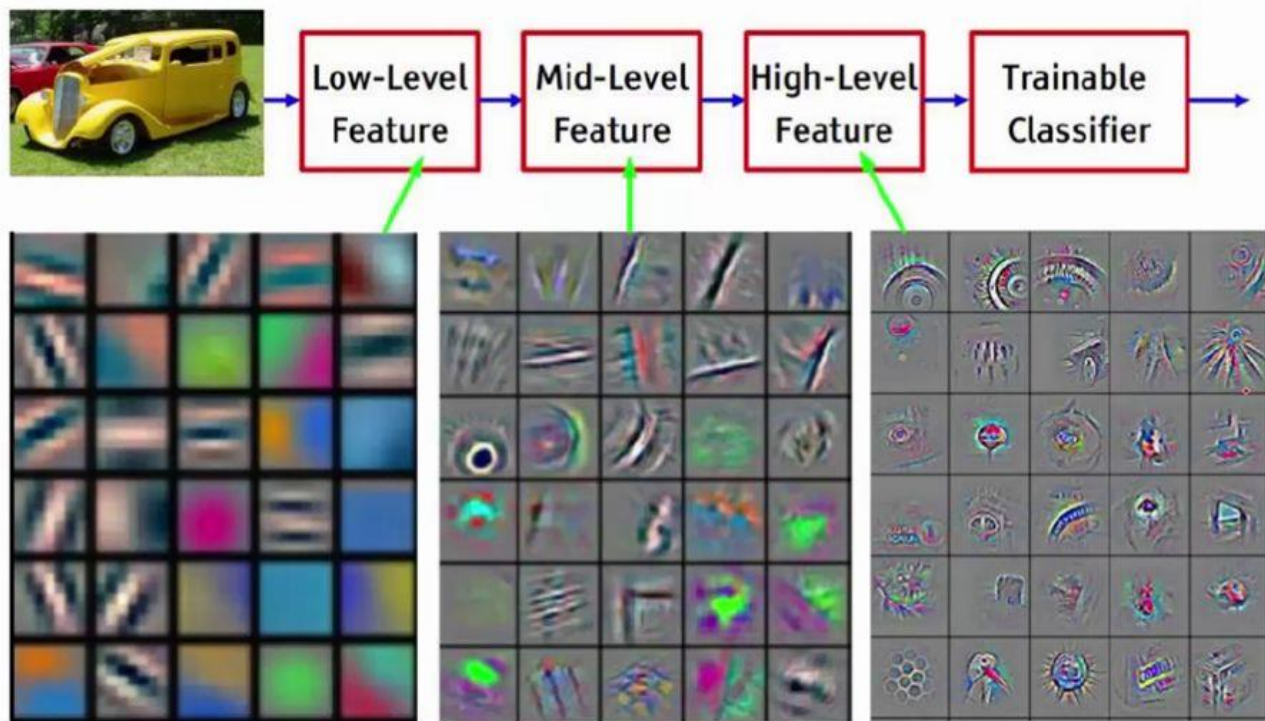
- 卷积网络的核心思想是将：局部感受野、权值共享以及时间或空间亚采样(池化)这三种结构思想结合起来获得了某种程度的位移、尺度、形变不变性。
- 卷积不仅限于对原始输入的卷积。蓝色方块是在原始输入上进行卷积操作，使用了6个filter得到了6个提取特征图。绿色方块还能对蓝色方块进行卷积操作，使用了10个filter得到了10个特征图。每一个filter的深度必须与上一层输入的深度相等。





### 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。

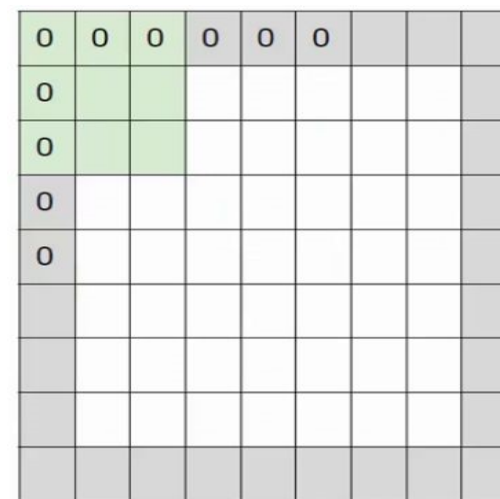
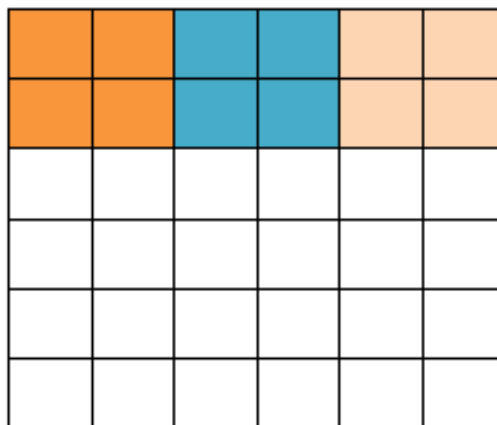
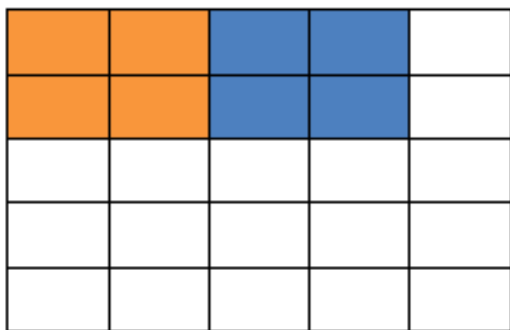




### 3. 卷积层



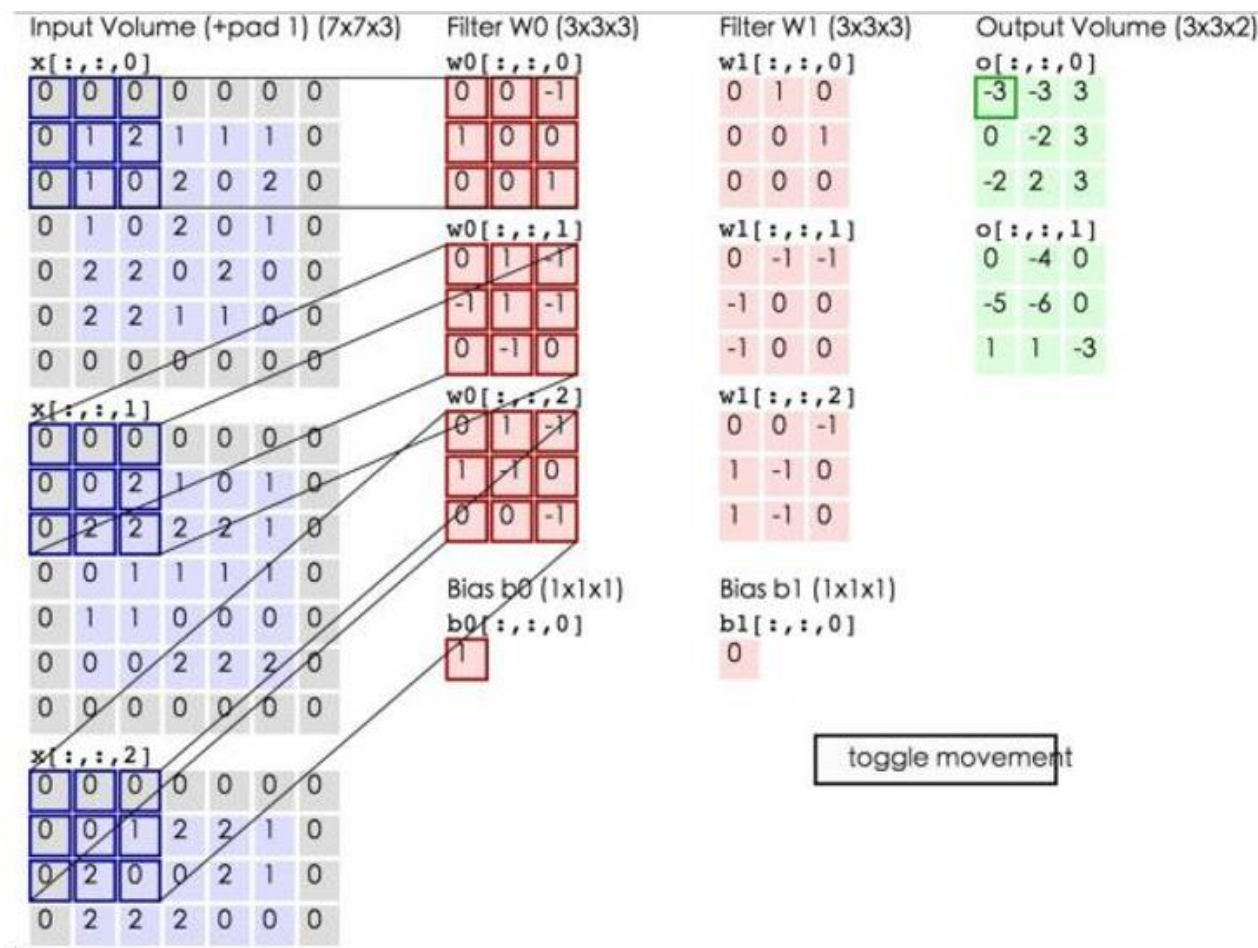
- **填充值:** 如下左图，一个 $5 \times 5$ 的图片（一个格子一个像素），滑动窗口取 $2 \times 2$ ，步长取2，那么还剩下1个像素没法滑完，怎么办？
- 在原先的矩阵加了一层填充值，使得变成 $6 \times 6$ 的矩阵，那么窗口就可以刚好把所有像素遍历完。这就是填充值的作用。



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

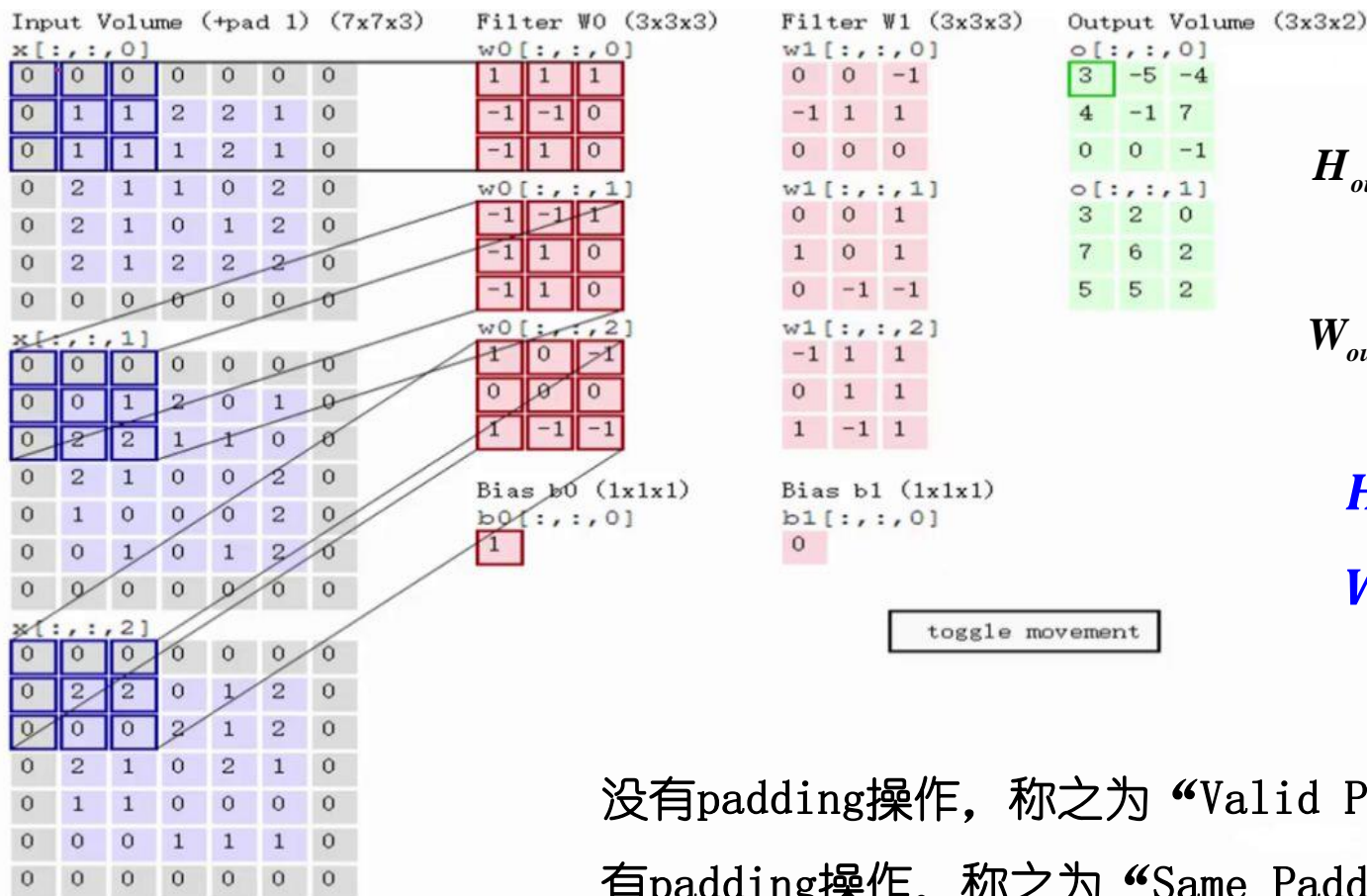
蓝色矩阵周围有一圈灰色的框，即填充值。左区域的三个大矩阵是原式图像的输入，RGB三个通道用三个矩阵表示，大小为 $7 \times 7 \times 3$ 。



- 粉色矩阵就是卷积层的神经元，这里表示有两个神经元( $w_0, w_1$ )；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 $3 \times 3$ ，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  $b_0$ 是Filter W0的偏置项，Bias  $b_1$ 是Filter W1的偏置项。

# (1)卷积的计算

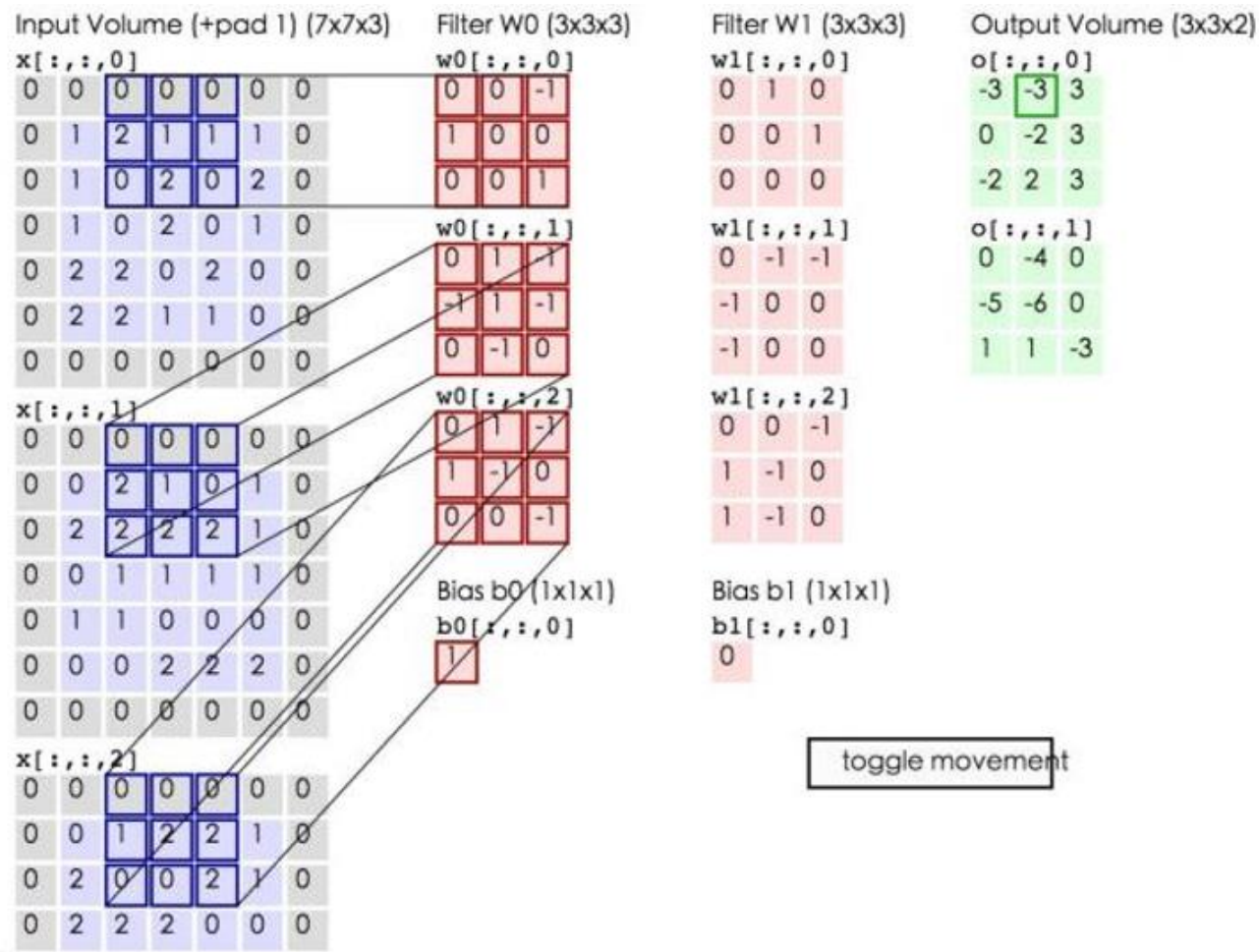
## 卷积结果输出维度的计算



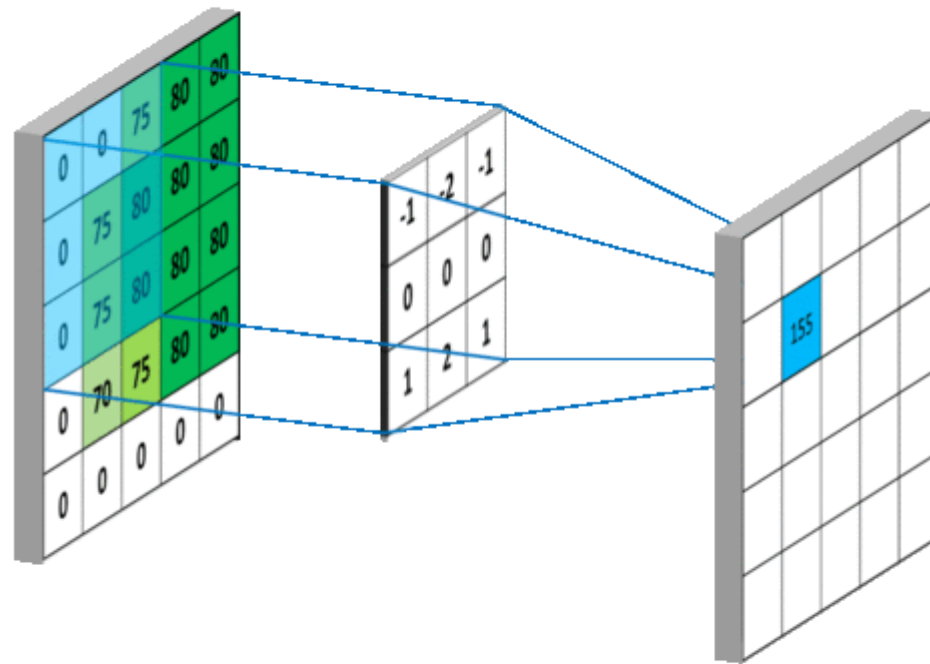
没有padding操作，称之为“Valid Padding”有效填充；  
有padding操作，称之为“Same Padding”等大填充。

# (1)卷积的计算

## 卷积计算过程

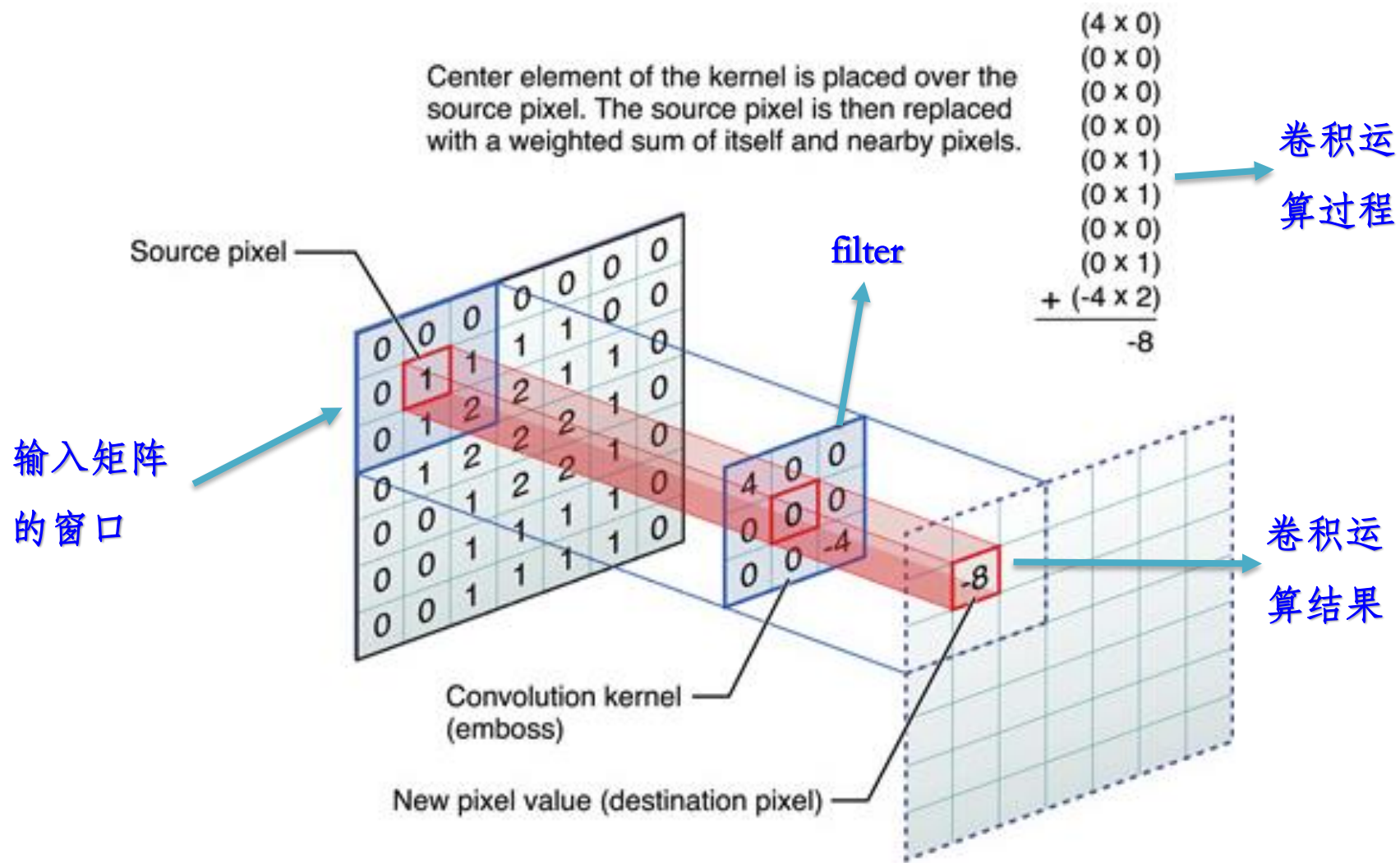


蓝色的矩阵(输入图像)对粉色的矩阵 (filter) 进行矩阵内积计算并将三个内积运算的结果与偏置值b相加 (如:  $2 + (-2 + 1 - 2) + (1 - 2 - 2) + 1 = 2 - 3 - 3 + 1 = -3$ ) , 计算后的值就是绿框矩阵的一个元素。





# (1)卷积的计算

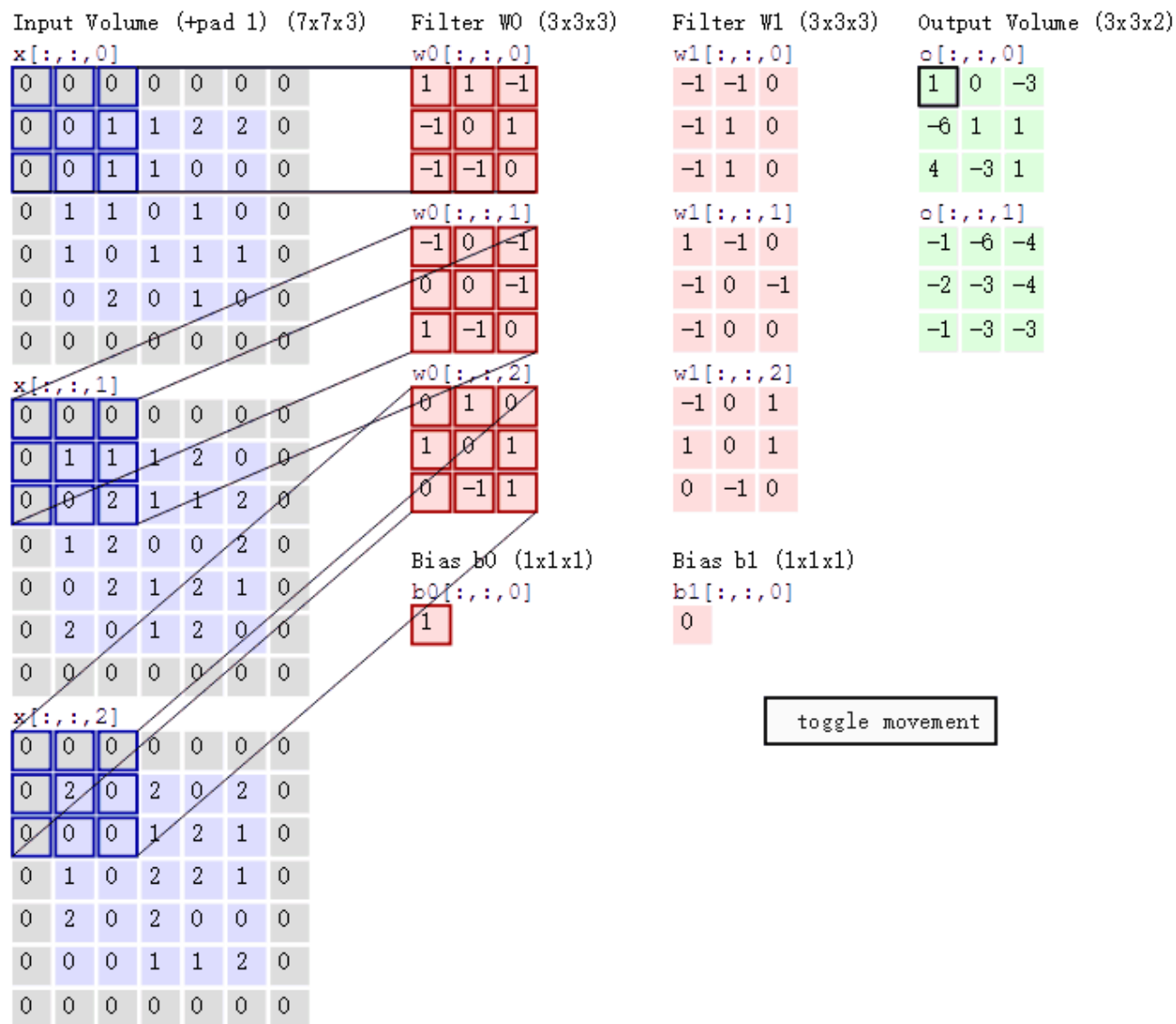




# (1)卷积的计算



## 卷积计算动态过程

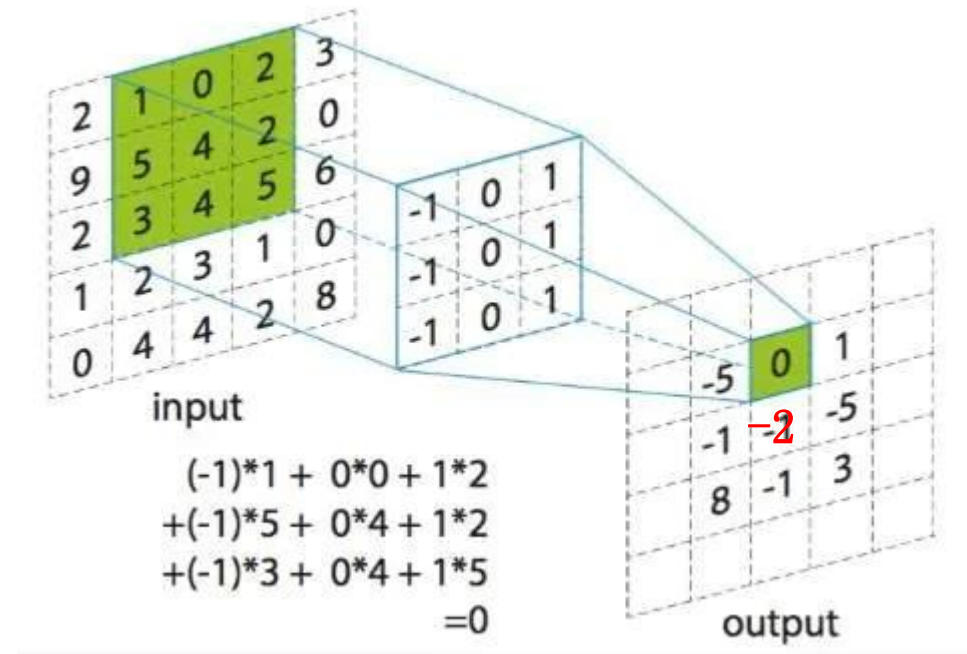


# (1)卷积的计算

MATLAB实现卷积运算的函数为`conv2(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。

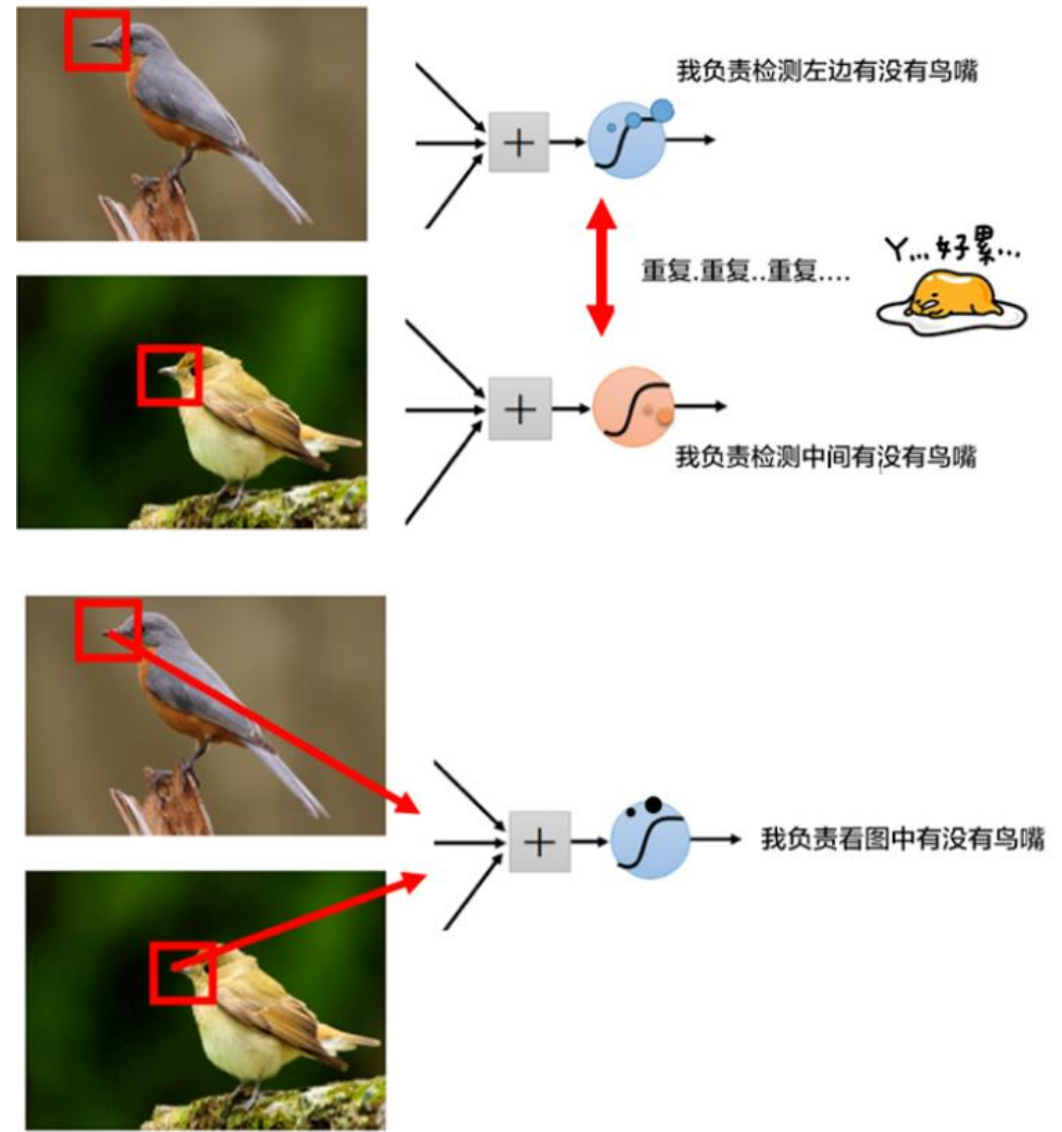
```
>> X = [2 1 0 2 3;9 5 4 2 0;2 3 4 5 6;1 2 3 1 0;0 4 4 2 8];
>> W = [-1 0 1;-1 0 1;-1 0 1];
>> W1 = rot90(W,2);
>> C1 = conv2(X,W,'valid')
C1 =
     5     0    -1
     1     2     5
    -8     1    -3

>> C2 = conv2(X,W1,'valid')
C2 =
    -5     0     1
    -1    -2    -5
     8    -1     3
```



## (2)参数共享机制

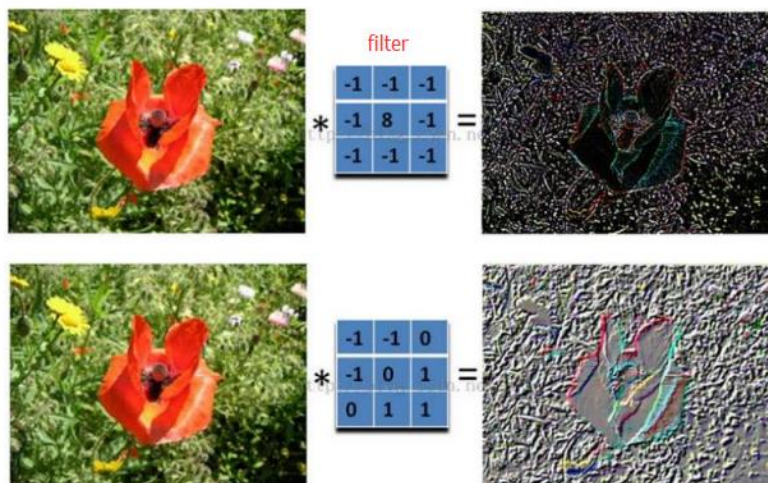
- 权值共享：不同的图像或者同一张图像共用一个卷积核，减少重复的卷积核。同一张图像当中可能会出现相同的特征，共享卷积核能够进一步减少权值参数。
- 如右图所示，为了找到鸟嘴，一个激活函数A需要检测图像左侧有没有鸟嘴，另外一个激活函数B需要检测另外一张图像中间有没有类似的鸟嘴。其实，鸟嘴都可能具有同样的特征，只需要一个激活函数C就可以，这个时候，就可以共享同样的权值参数（卷积核）。



## (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。

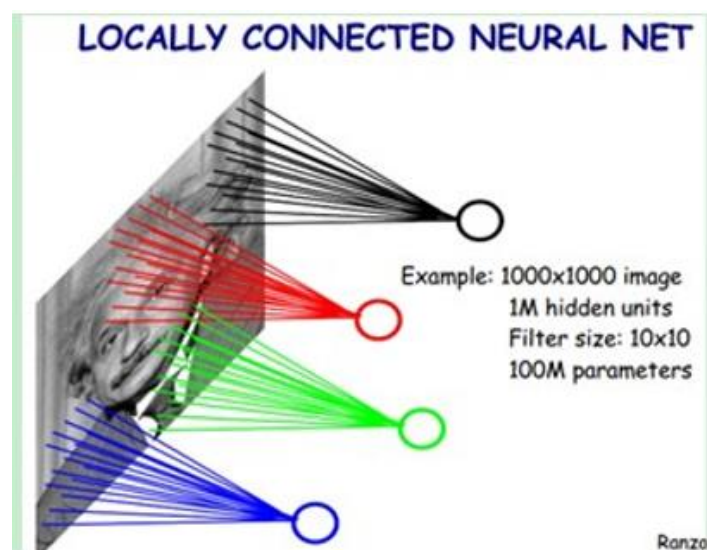
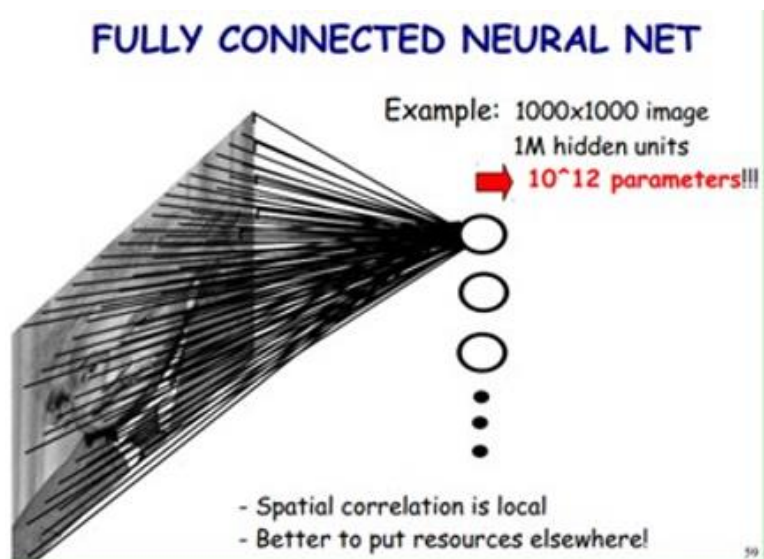
- 需要估算的权重个数减少：AlexNet 1亿  $\Rightarrow$  3.5w
- 一组固定的权重和不同窗口内数据做内积：卷积



## (2)参数共享机制



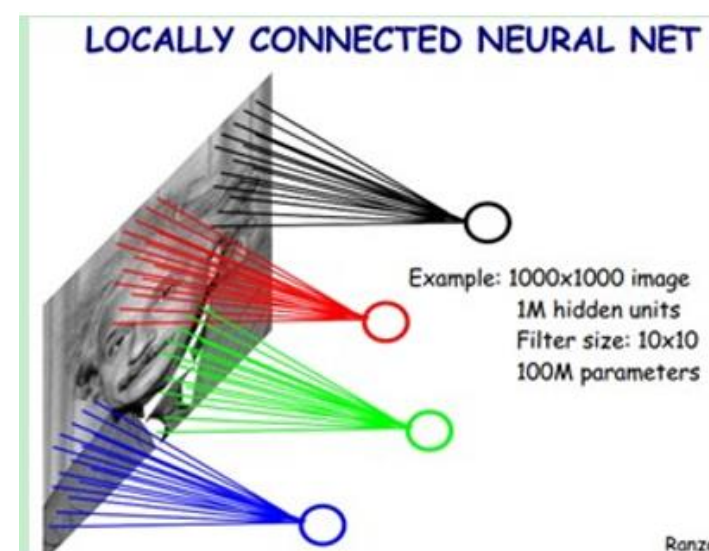
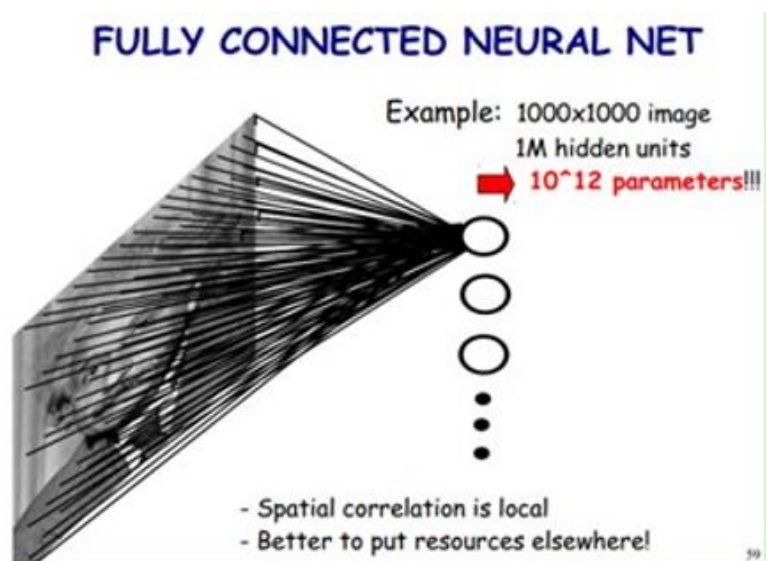
- 如果我们有1000\*1000像素的图像，有1百万个隐层神经元，那么他们全连接的话（每个隐层神经元都连接图像的每一个像素点），就有 $10^3 \times 10^3 \times 10^6 = 10^{12}$ 个连接，也就是 $10^{12}$ 个权值参数。
- 然而图像的空间联系是局部的，就像人是通过一个局部的感受野去感受外界图像一样，每一个神经元都不需要对全局图像做感受，每个神经元只感受局部的图像区域，然后在更高层将这些感受不同局部的神经元综合起来就可以得到全局的信息了。这样就可以减少连接的数目，也就是减少神经网络需要训练的权值参数的个数了。





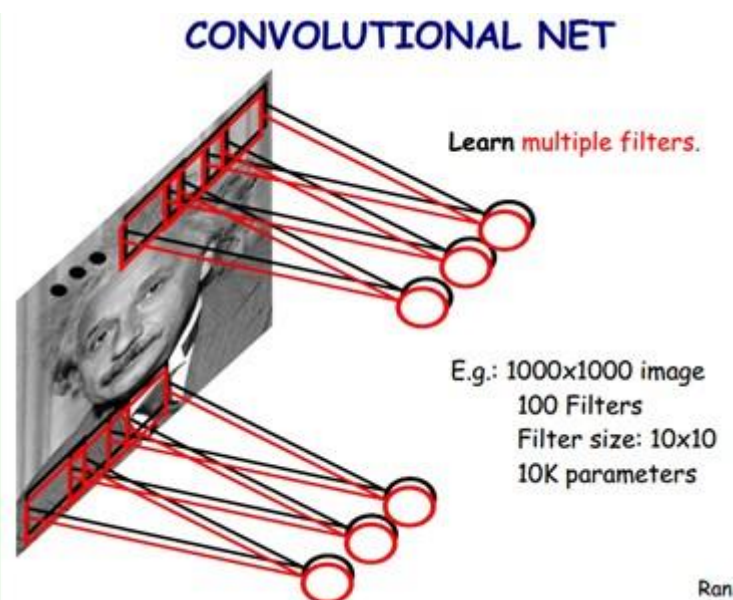
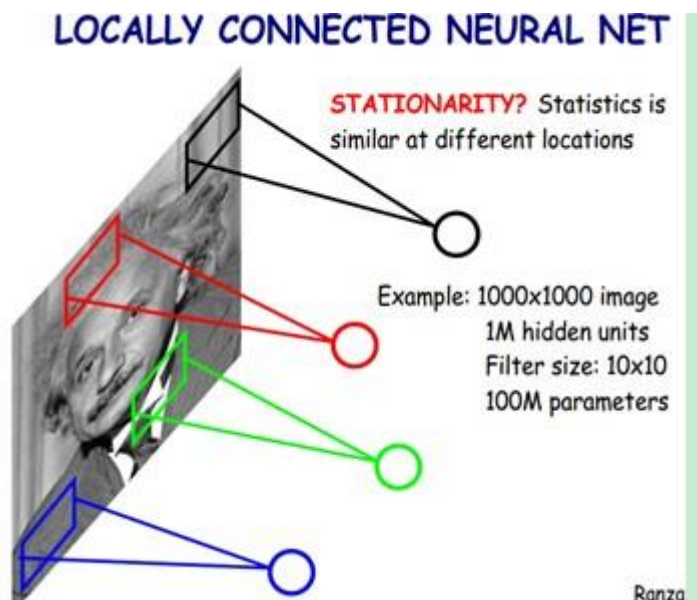
## (2) 参数共享机制

如下图右：假如局部感受野是 $10 \times 10$ ，隐层每个感受也只需要和这 $10 \times 10$ 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**！



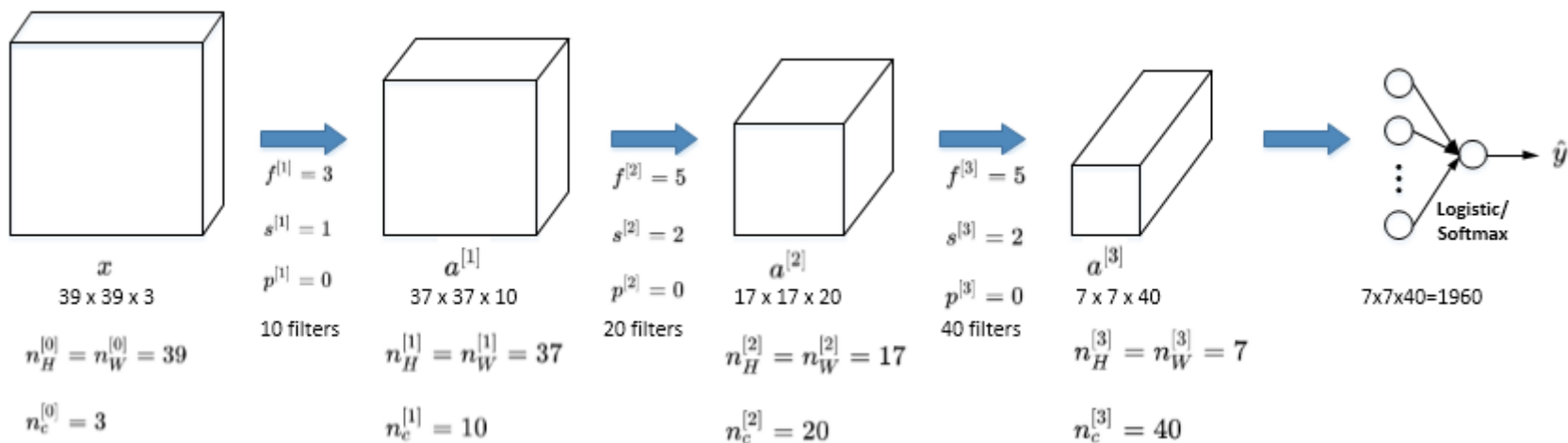
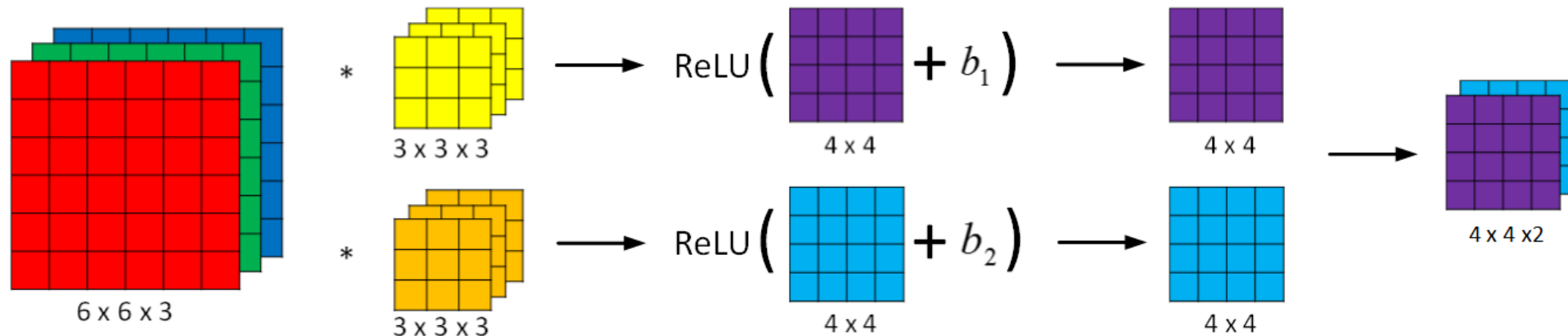
## (2)参数共享机制

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。

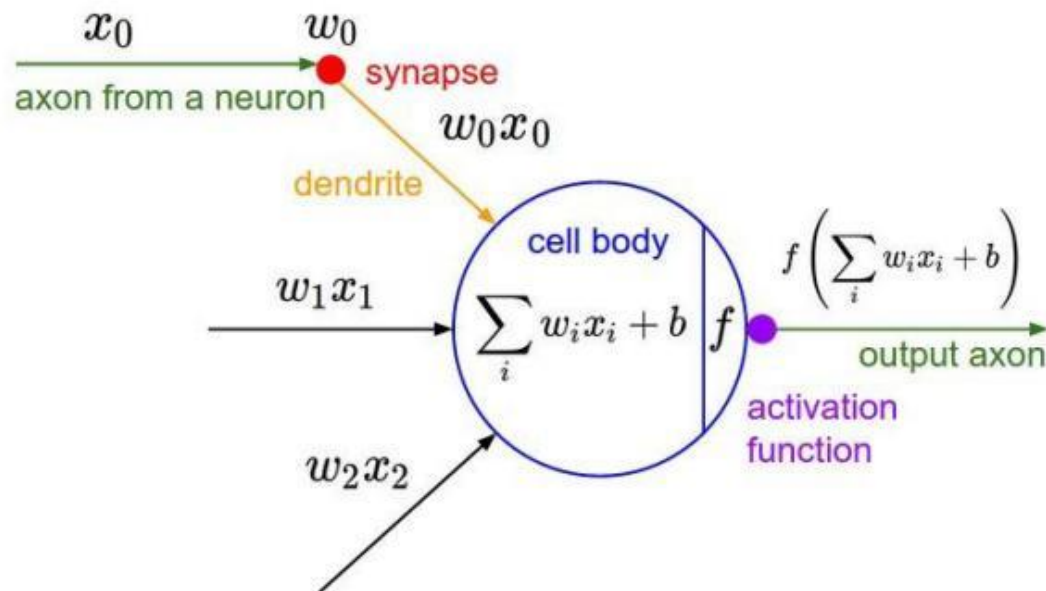


$$H_{out} = \left\lfloor \frac{H_{in} + 2H_{padding} - H_{kernel}}{H_{stride}} \right\rfloor + 1$$

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

### (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：

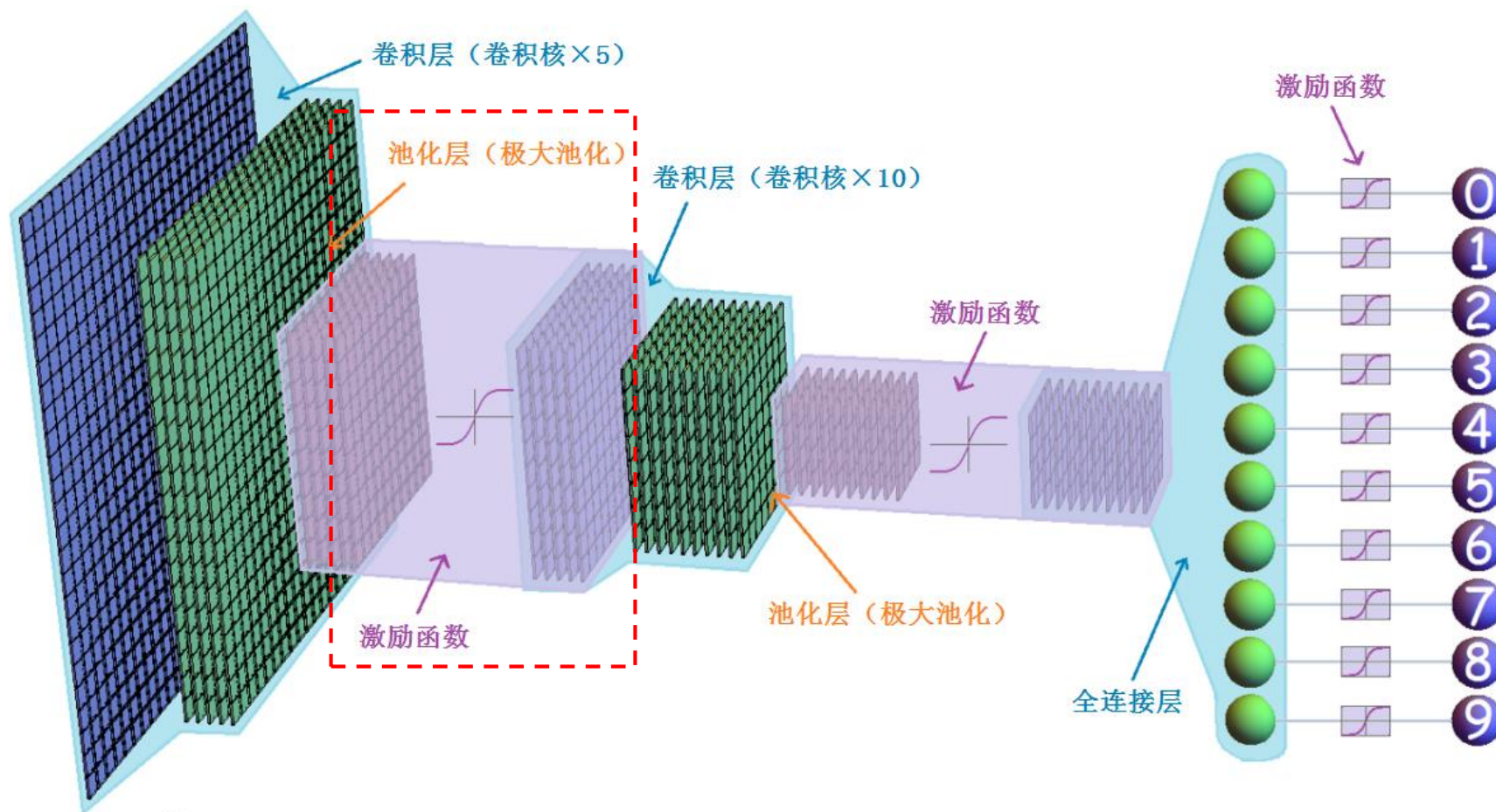
- ① 不要用sigmoid! 不要用sigmoid! 不要用sigmoid!
- ② 首先试RELU, 因为快, 但要小心点
- ③ 如果2失效, 请用Leaky ReLU或者Maxout
- ④ 某些情况下tanh倒是有不错的结果, 但是很少



## 4. 池化层

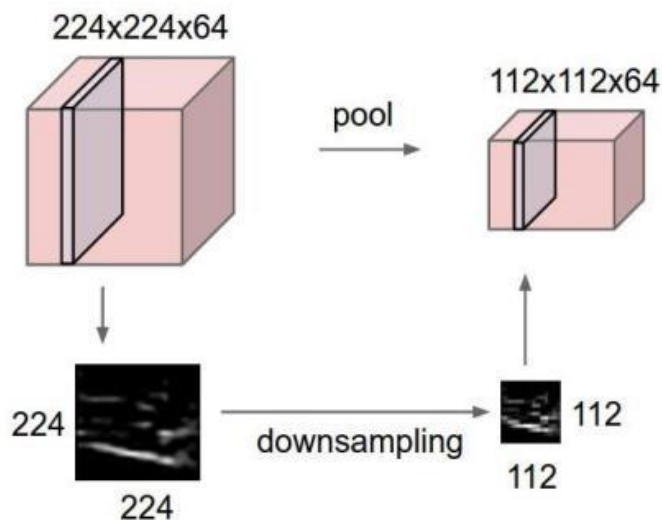


池化层夹在连续的卷积层之间，用于压缩数据和参数的量，减小过拟合。简而言之，如果输入的是图像，那么池化层的最主要作用就是压缩图像。



# 池化层的作用

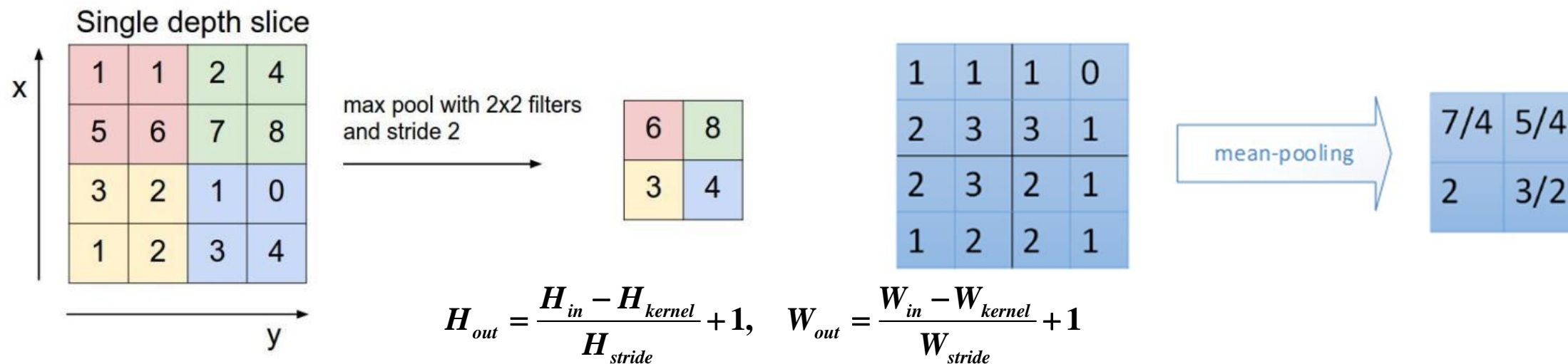
1. 特征不变性：即特征尺度不变性，池化操作就是图像的resize。如一张狗的图像被缩小了一倍还能认出是一张狗的照片，说明图像中仍保留狗最重要的特征，图像压缩时去掉的信息并不重要，而留下的信息具有尺度不变性的特征，最能表达图像的特征。
2. 特征降维：一幅图像含有的信息很大且特征很多，但是有些信息对于图像任务没有太多用途或者有重复，可把这类冗余信息去除，把最重要的特征抽取出来。
3. 在一定程度上防止过拟合，更方便优化。



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

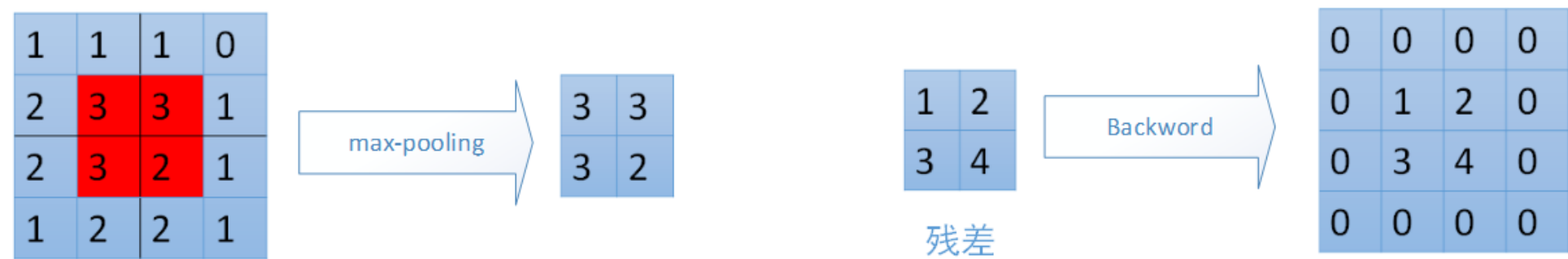
- 池化层用的方法有[Max pooling](#)和 [average pooling](#)，而实际用的较多的是Max pooling。



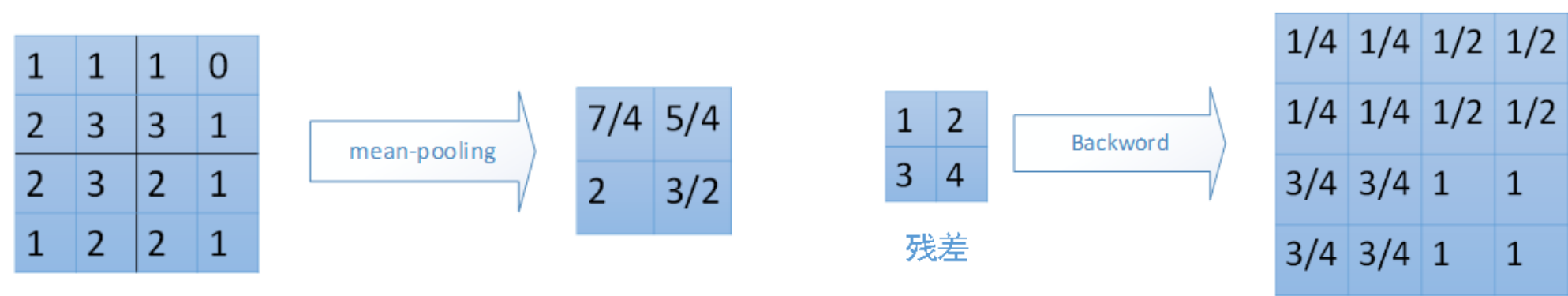
- [Max pooling](#)对于每个2\*2的窗口选出最大的数作为输出矩阵的相应元素的值，比如输入矩阵第一个2\*2窗口中最大的数是6，那么输出矩阵的第一个元素就是6，如此类推。
- [average pooling](#)对于每个2\*2的窗口选出该窗口所有元素的均值数作为输出矩阵的相应元素的值，比如输入矩阵第一个2\*2窗口中均值数是  $(1+1+2+3)/4 = 7/4$ ，那么输出矩阵的第一个元素就是7/4，如此类推。

# 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：

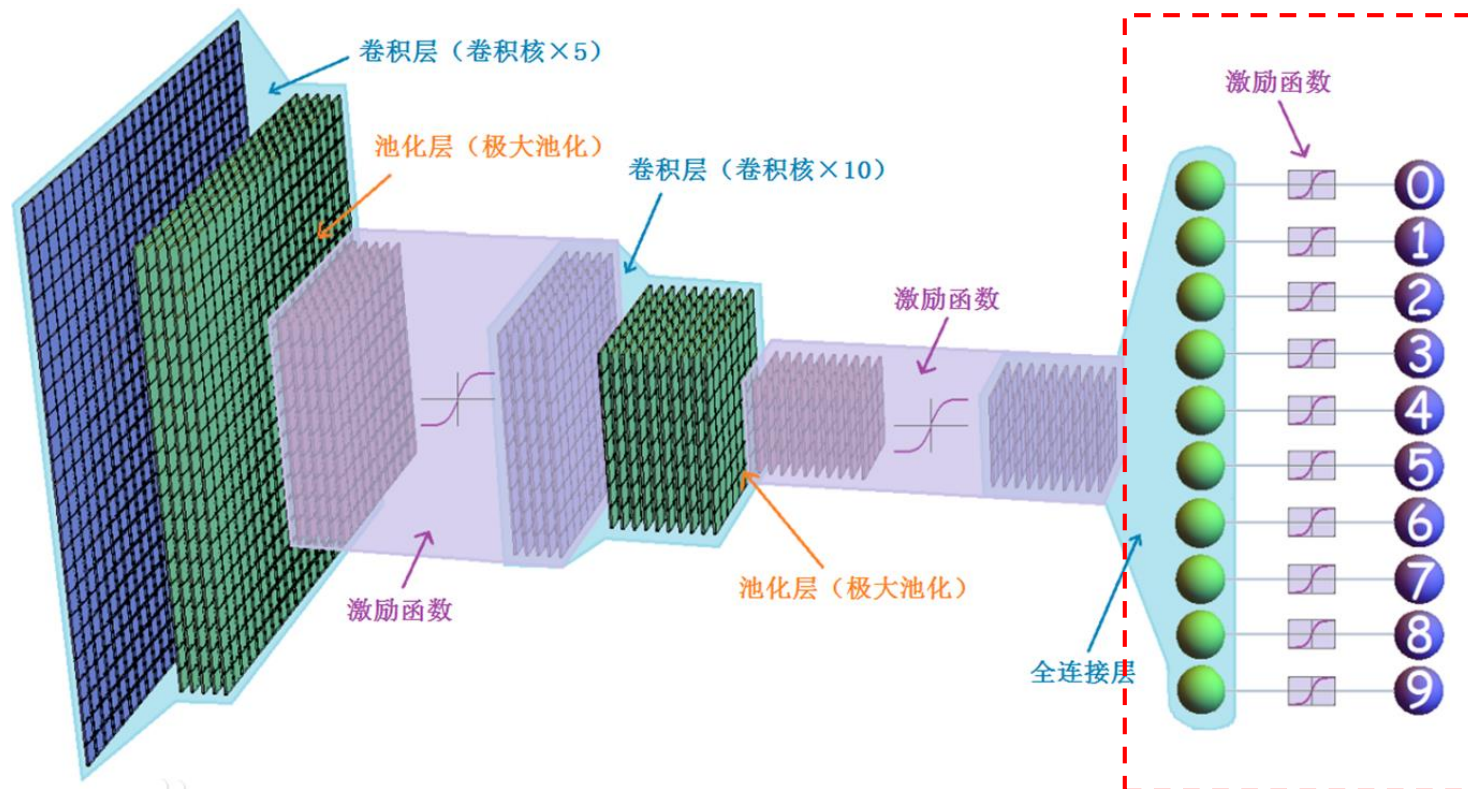
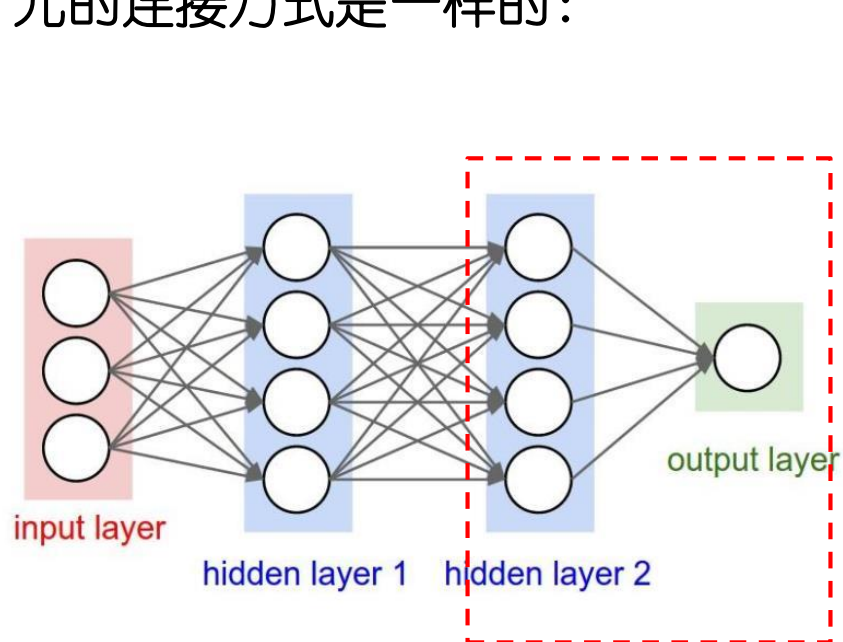




## 5. 全连接层



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：



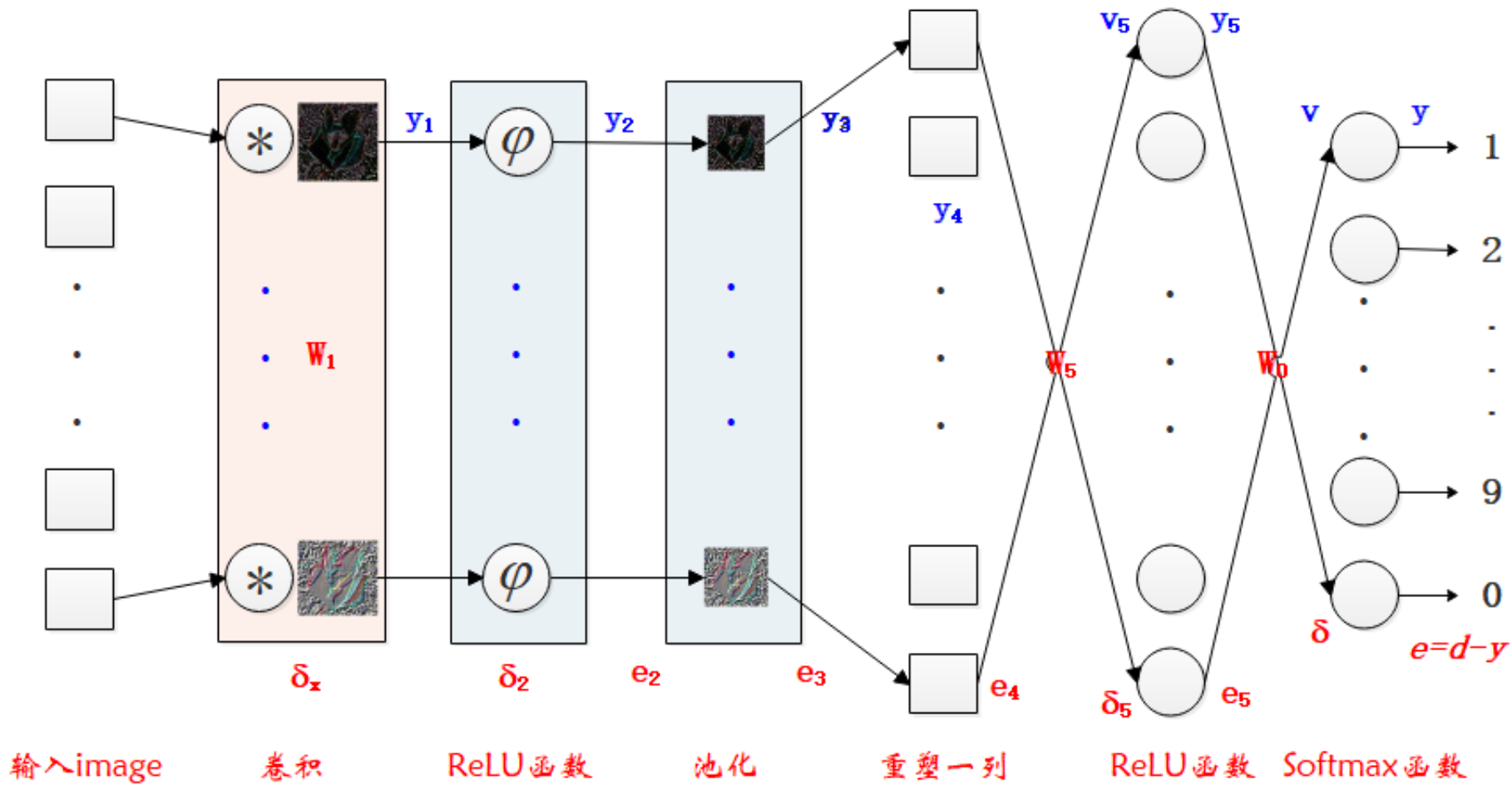
## 6. 卷积神经网络MNIST示例

- 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

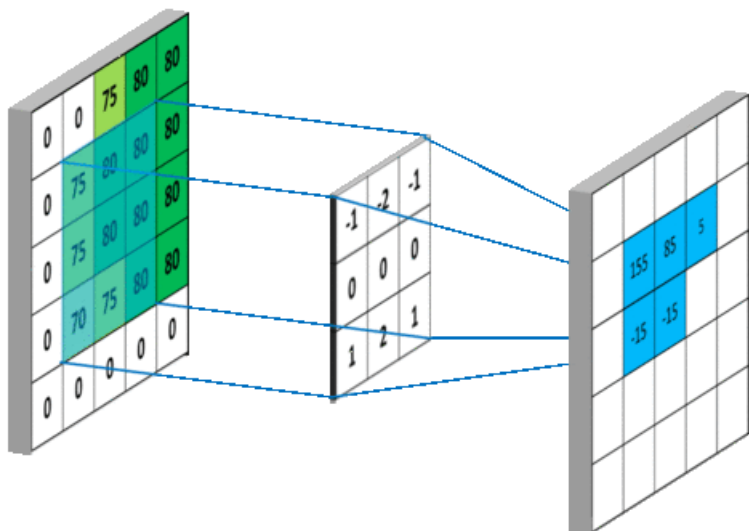
## 卷积神经网络算法设计的架构和变量的定义



# 卷积运算函数Conv与池化运算函数Pool

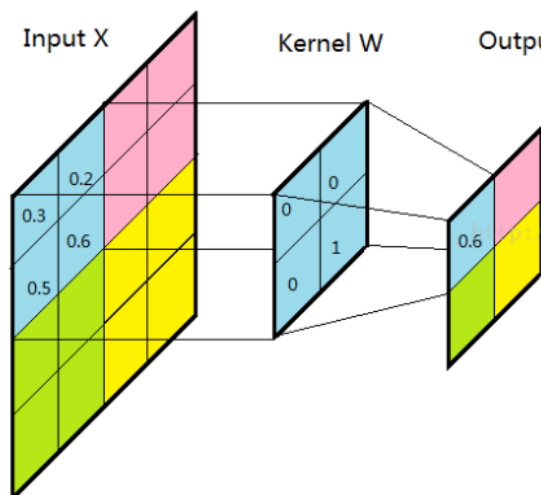


```
1 function y = Conv(x, W)
2 %Conv函数实现卷积运算, 输入W为输入层到卷积层权重, x是一幅图像, 如28*28
3 % W是三维矩阵, mrow和wcol代表卷积核的大小, numFilters代表滤波器的数量
4 [wrow, wcol, numFilters] = size(W);
5 [xrow, xcol, ~] = size(x);
6 yrow = xrow - wrow + 1; %卷积运算后图像维度
7 ycol = xcol - wcol + 1;
8 y = zeros(yrow, ycol, numFilters); %卷积后结果
9 for k = 1:numFilters %分别对每个卷积核进行卷积运算
10     filter = W(:, :, k);
11     %squeeze除去size为1的维度, 旋转180°
12     filter = rot90(squeeze(filter), 2);
13     % 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
14     y(:, :, k) = conv2(x, filter, 'valid');
15 end
16 end
```

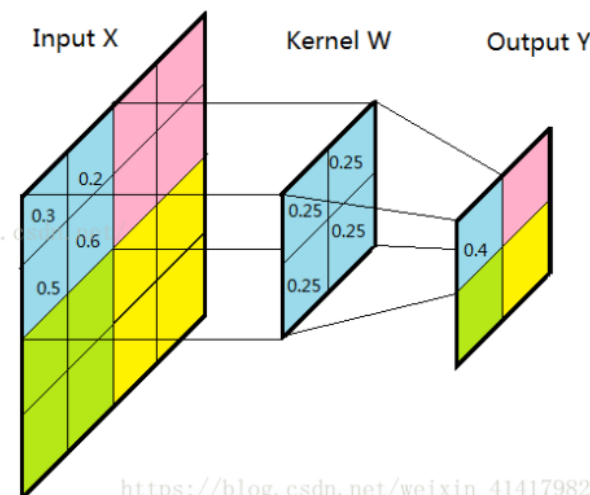


```
1 function y = Pool(x)
2 % Pool函数实现池化层功能, 2x2 mean pooling
3 % x为池化层经过ReLU激活后的结果
4 [xrow, xcol, numFilters] = size(x);
5 y = zeros(xrow/2, xcol/2, numFilters); %池化后的图像维度
6 for k = 1:numFilters
7     filter = ones(2) / (2*2); % for mean
8     % valid返回在卷积过程中, 未使用边缘补0部分进行计算的卷积结果部分
9     image = conv2(x(:, :, k), filter, 'valid');
10     y(:, :, k) = image(1:2:end, 1:2:end);
11 end
12 end
```

max-pooling



mean-polling





## 6. 卷积神经网络MATLAB算法实现



```
MnistConv.m  +
1  function [W1, W5, Wo] = MnistConv(X, D, epochs, test, testlabel)
2  %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
3  %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权重
4  %X是特征数据，三维矩阵，D为目标数据标签
5
6  %% 权重初始化及变量初始化
7  %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
8  W1 = 1e-2*randn([9 9 20]); %20个9*9卷积滤波器
9  [mx, nx, ~] = size(X);
10 ms = ((mx - 9 + 1) - 2)/2 + 1; %卷积池化后的图像尺寸
11 ns = ((nx - 9 + 1) - 2)/2 + 1;
12 %Xavier初始化权重方法
13 W5 = (2*rand(100, ms*ns*20)-1) * sqrt(6)/sqrt(100+ms*ns*20);
14 Wo = (2*rand(10, 100)-1) * sqrt(6)/sqrt(10+100);
15 alpha = 0.01; %学习率
16 beta = 0.95; %动量参数
17 momentum1 = zeros(size(W1)); %动量法调整权重，对应卷积层
18 momentum5 = zeros(size(W5)); %对应池化层到隐藏层
19 momentum0 = zeros(size(Wo)); %对应隐藏层到输出层
20 N = length(D); %数据量
21
22 %% 小批量算法
23 %各批量数据bsize为100，权重在每一轮训练中都被调整N/100次。
24 bsize = 100;
25 %每一小批量数据点的第一个点的索引
26 blist = 1:bsize:(N-bsize+1);
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
25
```

## 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(:, :, c), ...
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 — momentum0 = alpha * dWo + beta * momentum0;
82 — Wo = Wo + momentum0;
```

```
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.....
```

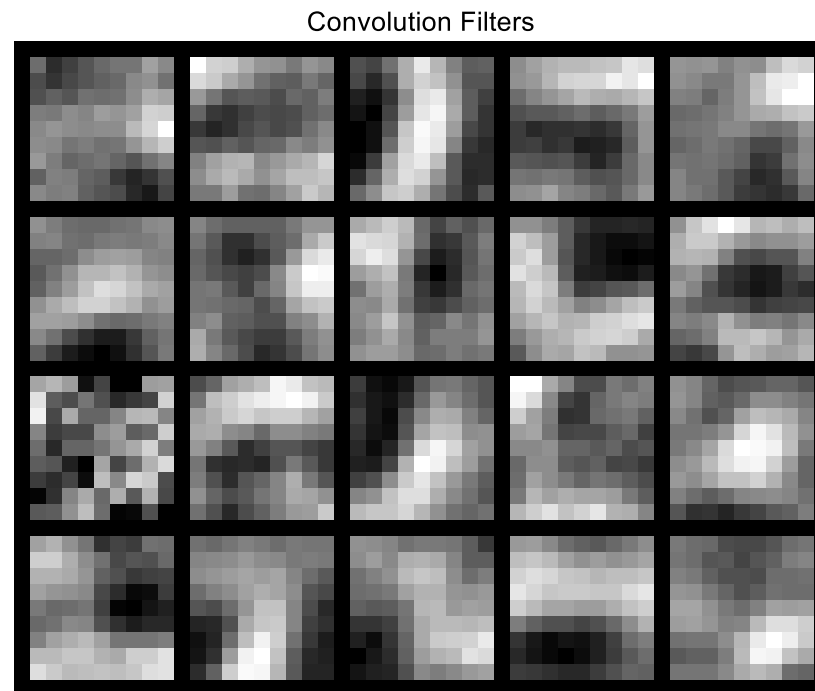
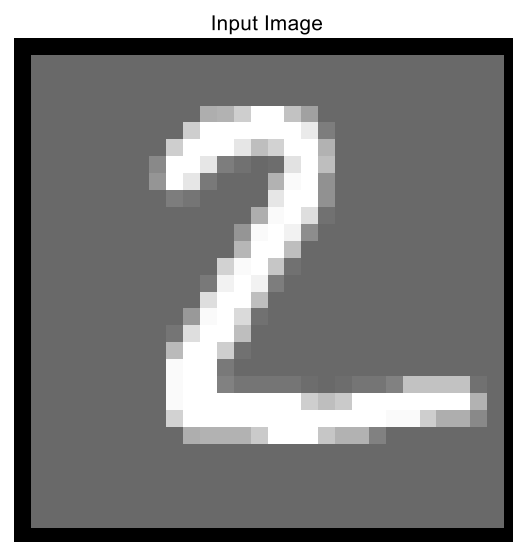
```
第【3】次训练中.....
```

```
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

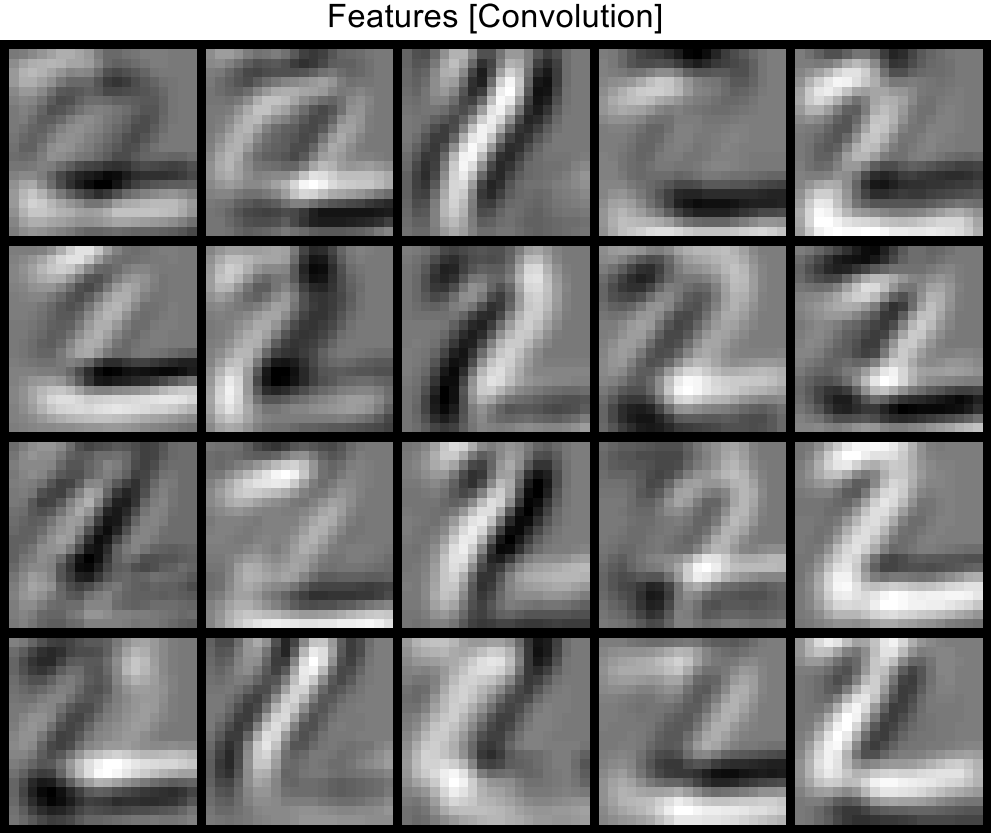


第一张图28\*28的输入图像“2”；

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。

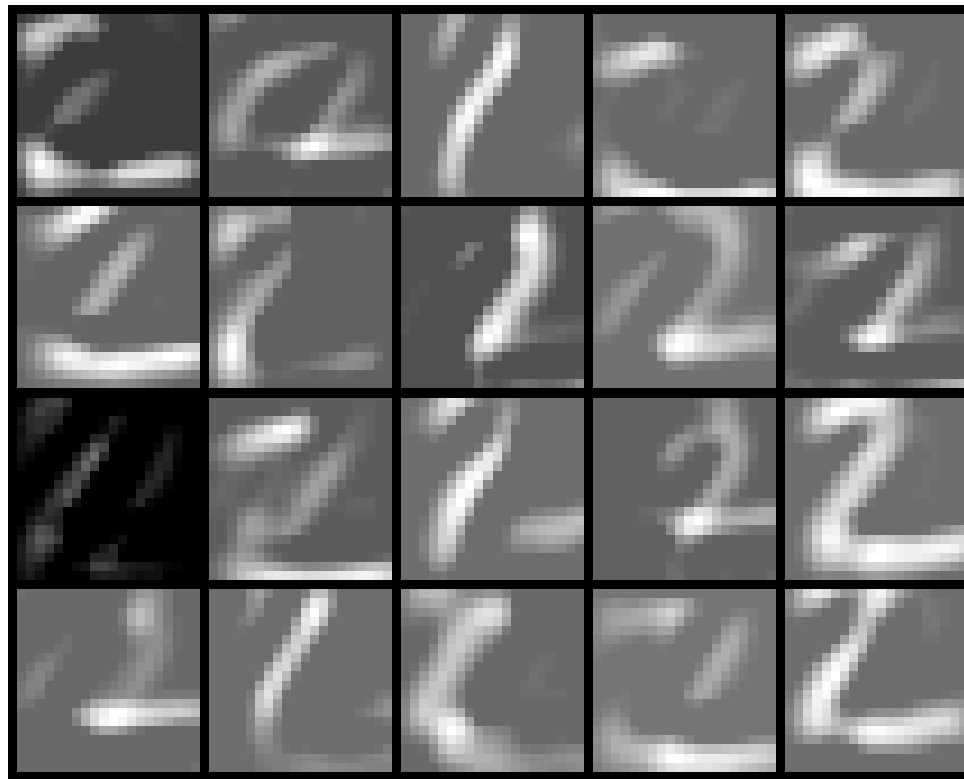


# 卷积神经网络MNIST示例

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]



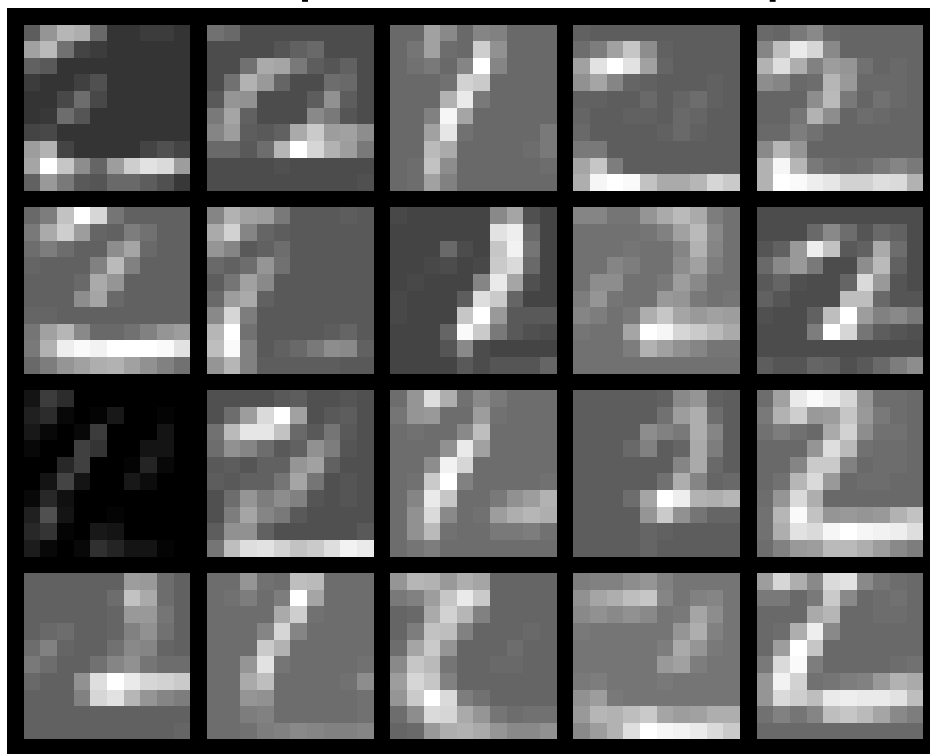
# 卷积神经网络MNIST示例



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 $10 \times 10$ 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]





---

# 感谢聆听

---