**CNN卷积神经网络教程**

实战项目——基于pytorch的深度学习花朵种类识别项目完整教程（内涵完整文件和代码）

[大白话pytorch基本知识点及语法+项目实战](https://blog.csdn.net/zhaohaobingniu/article/details/119923064)

**文章目录**

* [超详细——CNN卷积神经网络教程（零基础到实战）](#CNN_0)
* [CNN](#CNN_7)
* [知识点：](#_9)
* [K近邻算法：（有缺点，神经网络弥补了）](#K_35)
* [传统神经网络层级结构（nn，二维）：](#nn_62)
* [数据预处理：](#_77)
* [前向传播：](#_93)
* [神经网络基础线性函数：（计算得分值）](#_95)
* [损失函数L：](#L_106)
* [计算损失值：](#_108)
* [softmax分类器->得各类概率值：](#softmax_133)
* [反向传播：](#_140)
* [卷积神经网络（cnn，三维）：](#cnn_154)
* [卷积层：](#_169)
* [池化层：](#_198)

**CNN**

**知识点：**

较好数据集：CIFAR-10 数据集

机器学习流程：数据获取、特征提取、建立模型、评估与应用

[深度](https://so.csdn.net/so/search?q=深度&spm=1001.2101.3001.7020)学习特征：数据特征决定了模型的上限、算法与参数选择决定了

神经网络作用：自动进行各种特征提取

神经网络目标：更新w权值矩阵

神经网络矩阵计算：都是内积计算，两矩阵对应位置相乘然后相加

神经网络包括两部分：前向传播和[反向传播](https://so.csdn.net/so/search?q=反向传播&spm=1001.2101.3001.7020)

回归任务：得分值->损失值

概率任务：概率值->损失值

[pytorch](https://so.csdn.net/so/search?q=pytorch&spm=1001.2101.3001.7020)是实现cnn的一种方法

如何逼近这个上限、预处理和特征提取是最核心的

**K近邻算法：（有缺点，神经网络弥补了）**

计算流程：

1.计算已知类别数据集中的点与当前点的距离

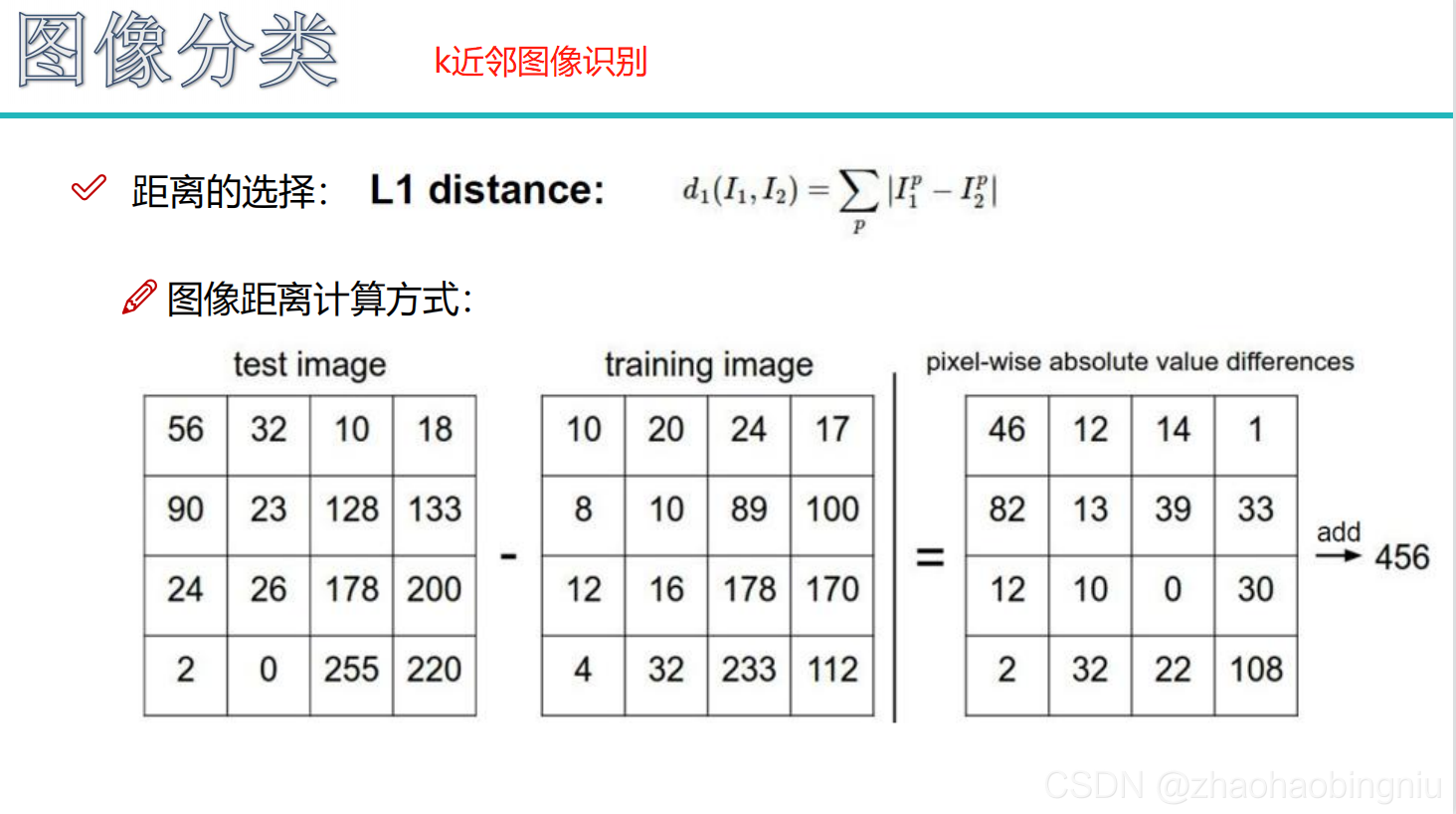
2.按照距离依次排序

3.选取与当前点距离最小的K个点

4.确定前K个点所在类别的出现概率

5.返回前K个点出现频率最高的类别作为当前点预测分类

k近邻进行的图像识别，用测试照片和众多训练照片各个对应像素点相减，得到的差值矩阵各个数据的总和最小的，则测试图片和该训练图片一种类别。



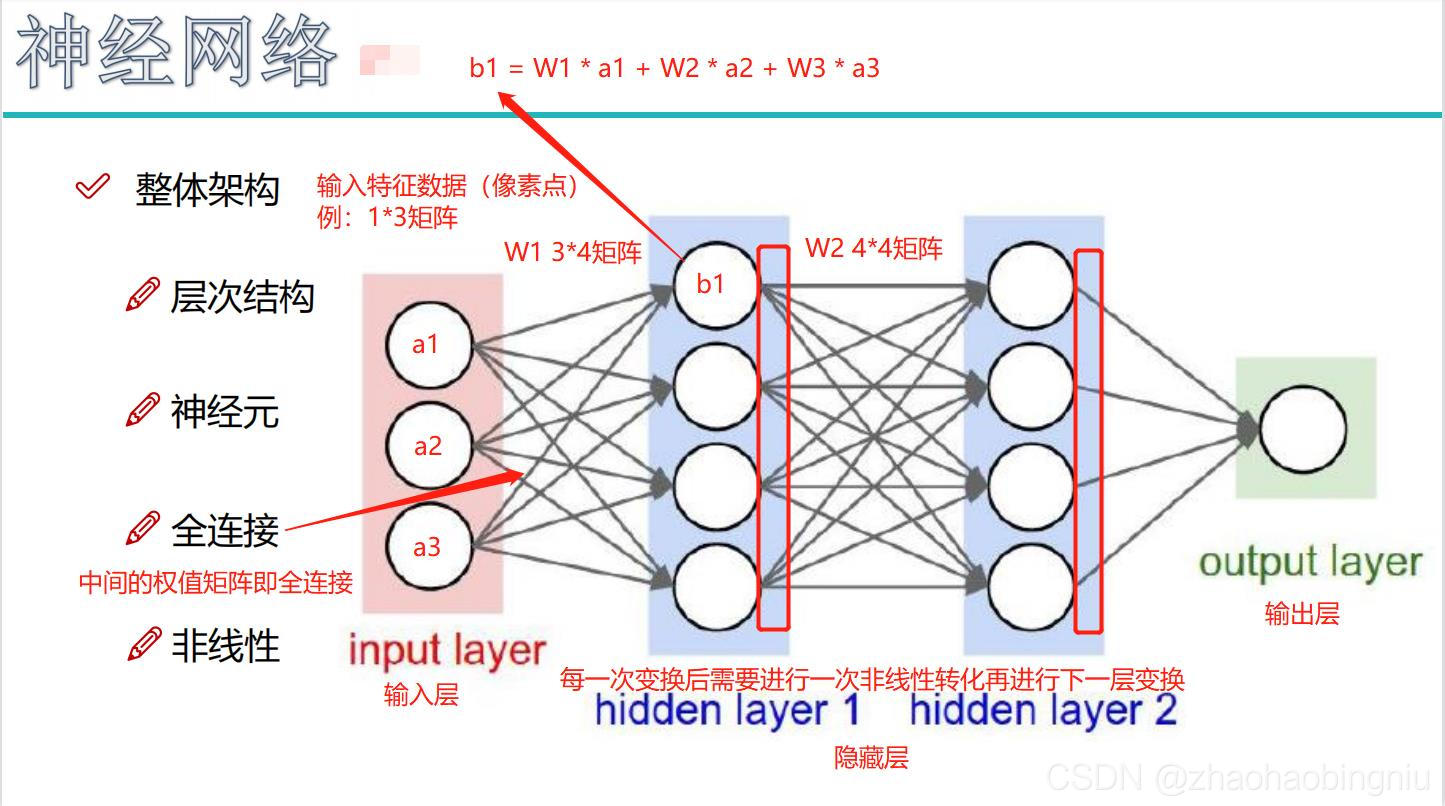
缺点：不能分清哪个是主体，哪个是背景，干扰太大

**传统神经网络层级结构（nn，二维）：**

隐层神经元数量越多效果越好，但太多容易出现过拟合

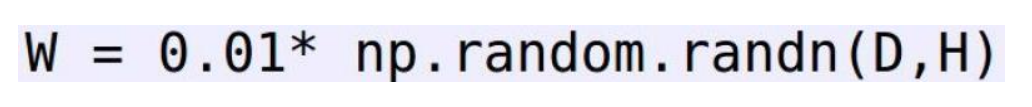
非[线性](https://so.csdn.net/so/search?q=线性&spm=1001.2101.3001.7020)变换用的激活函数

全连接层不能搞三维

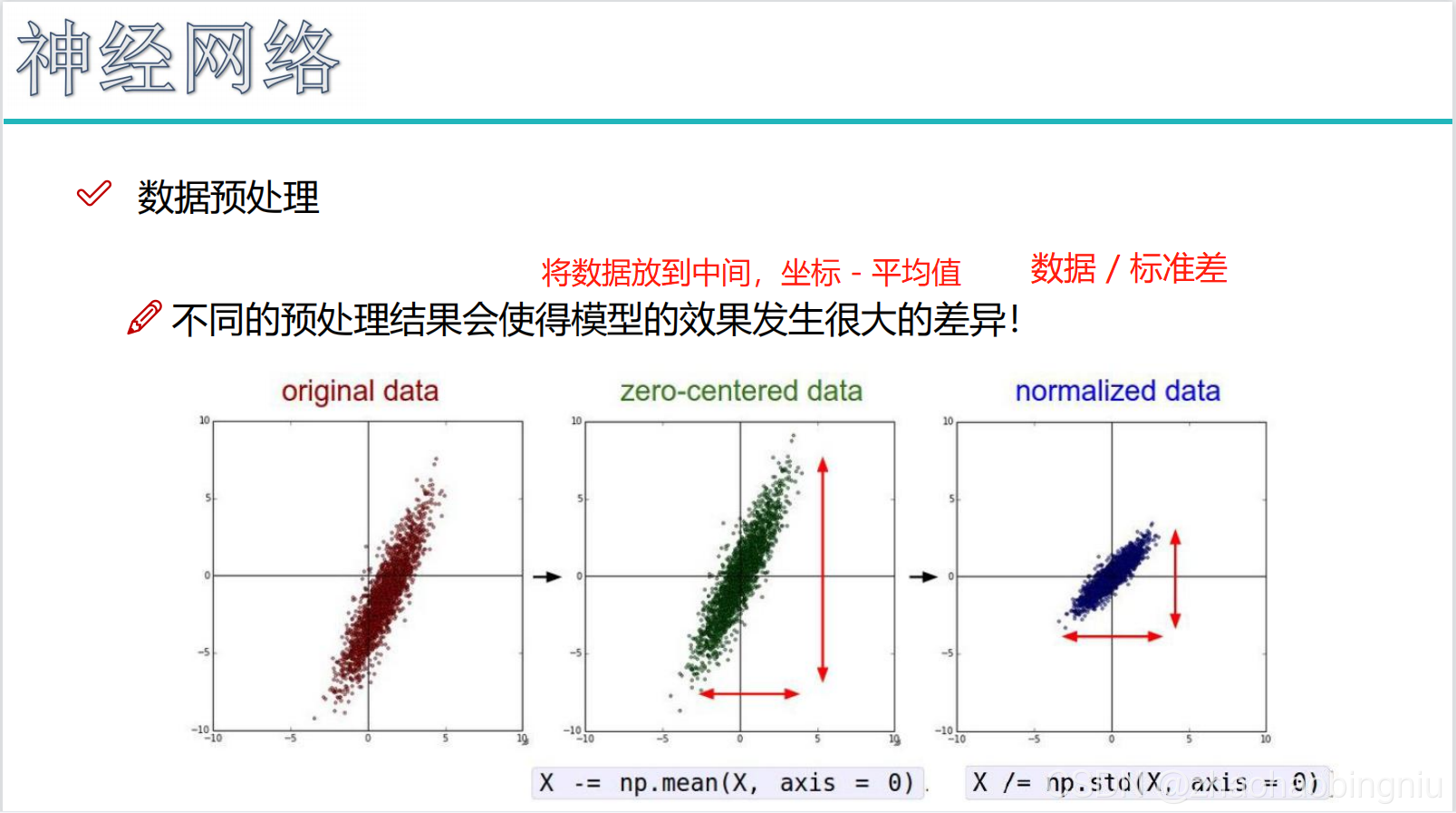


[**数据预处理**](https://so.csdn.net/so/search?q=数据预处理&spm=1001.2101.3001.7020)**：**

参数初始化：初始化权值矩阵（随机生成）



数据标准化：

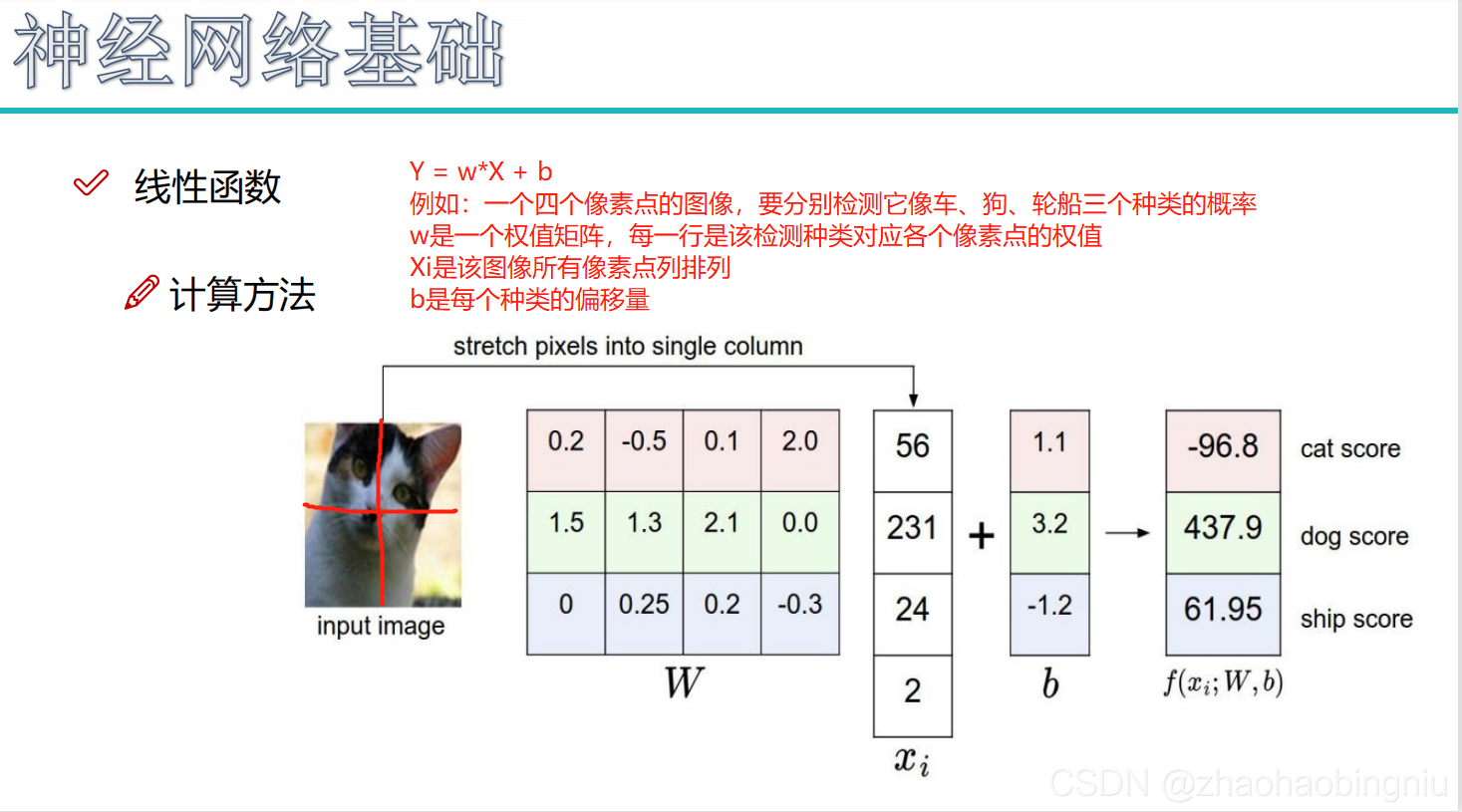


**前向传播：**

**神经网络基础线性函数：（计算得分值）**

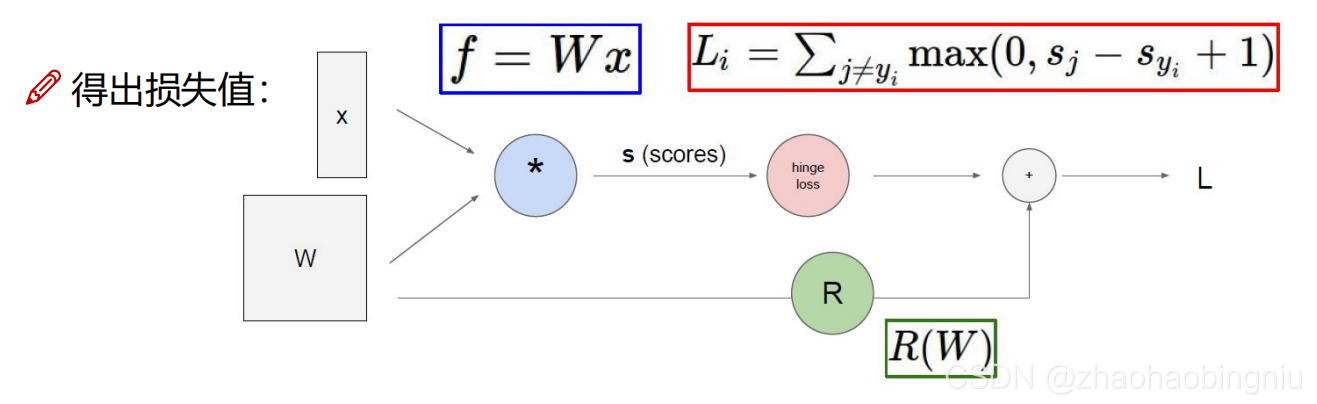
F（Xi，w，b）= w\*xi + b

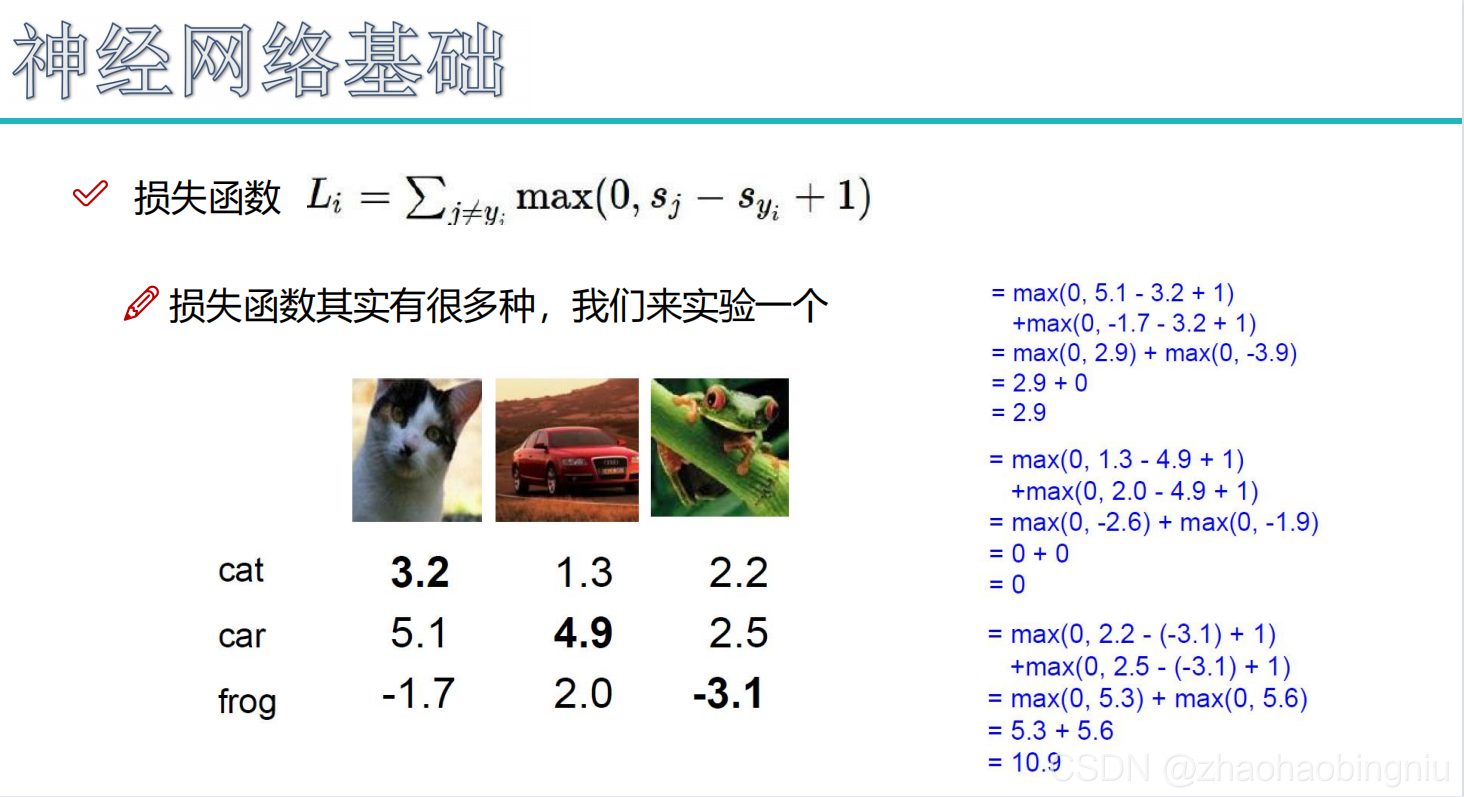
w 权值矩阵是不断优化出来的，初始化一个随机的

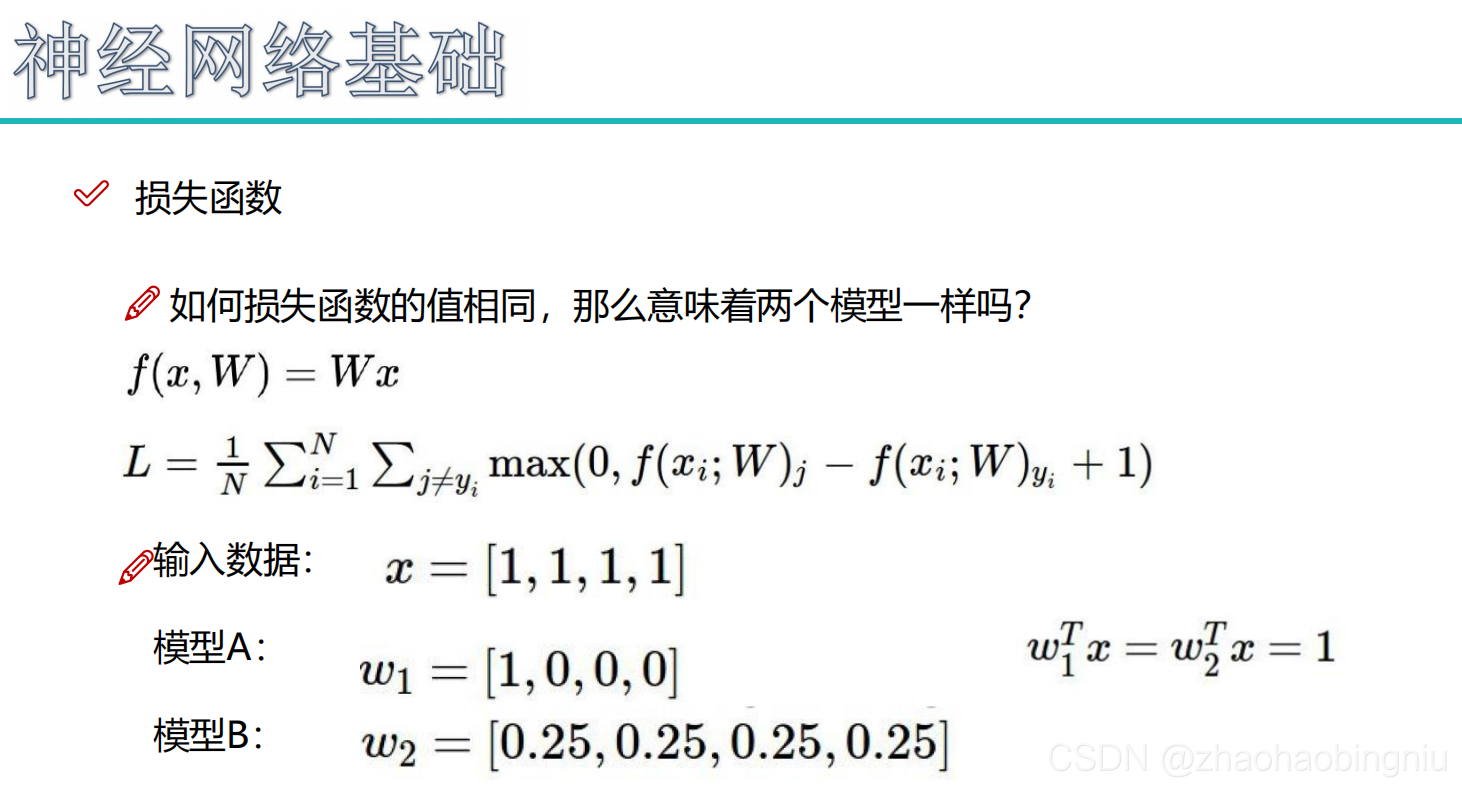


[**损失函数**](https://so.csdn.net/so/search?q=损失函数&spm=1001.2101.3001.7020)**L：**

**计算损失值：**



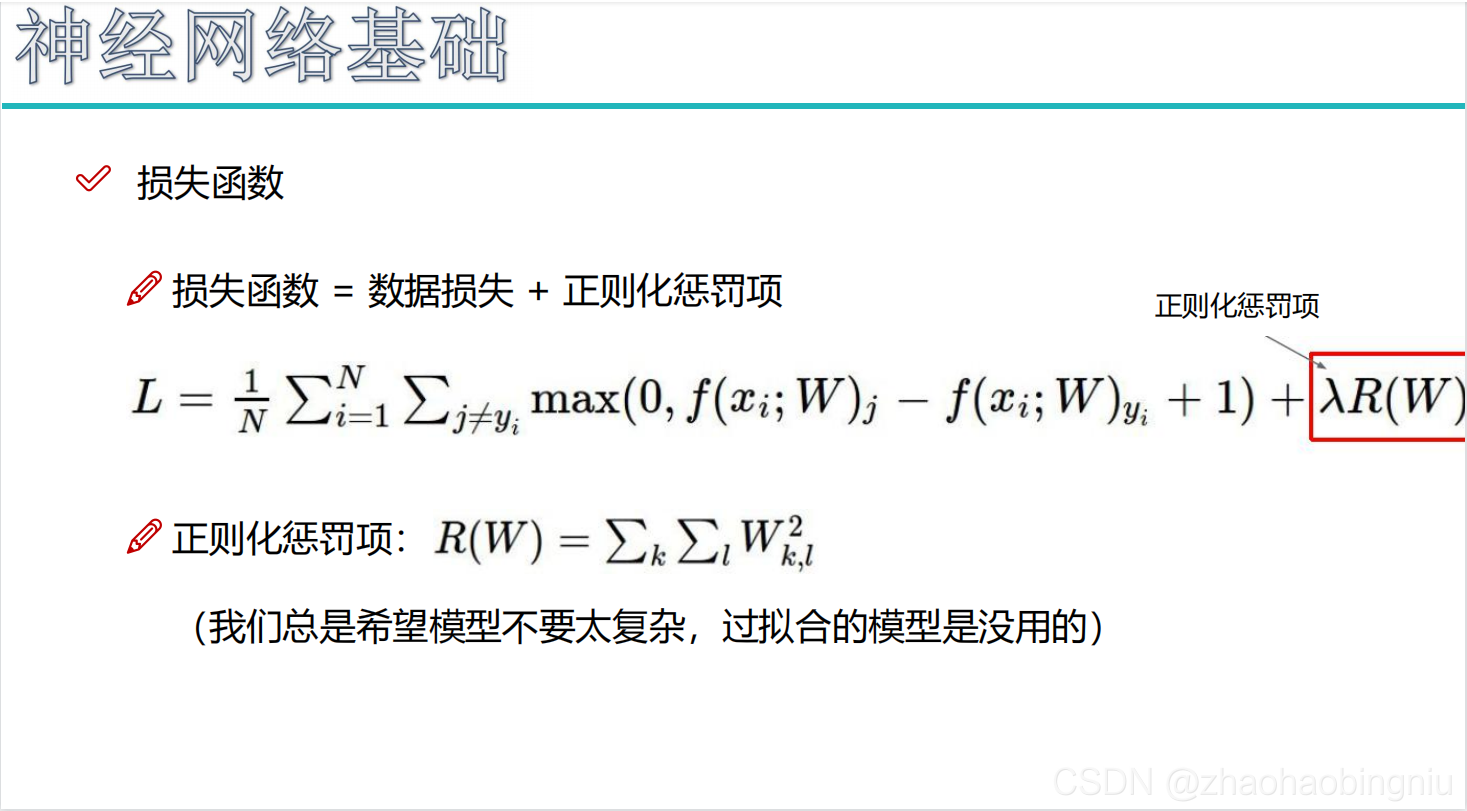




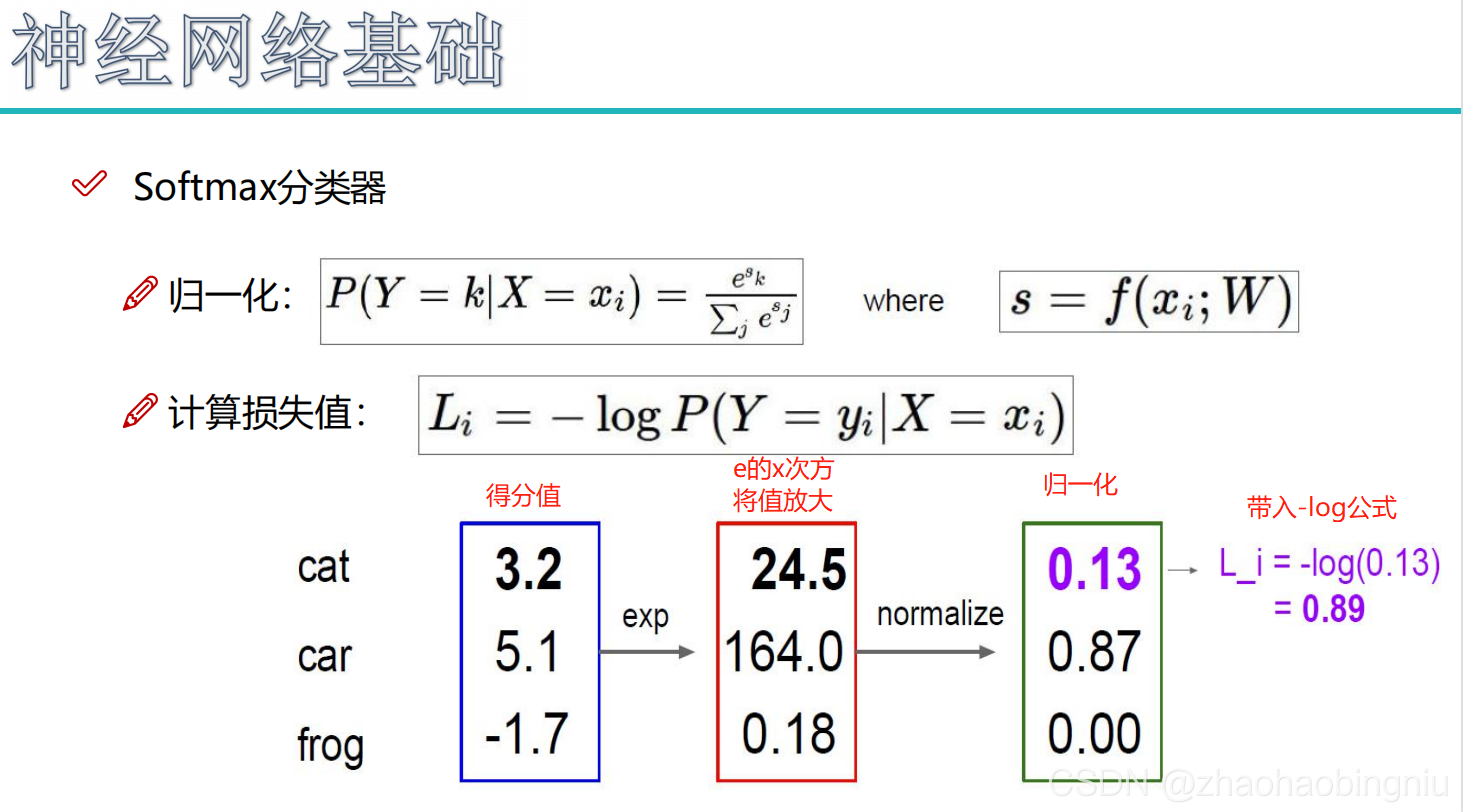
max（0，（该预测的各个类别的概率的最大值 - 正确种类的预测值 + 1 ））

0和该项取一个大值，+1 是容错值，即正确类别比其他类别的最大值要大1才行

R（w）是防止过拟合，即（1，0，0，0）和（0.25，0.25，0.25，0.25），前者可能恰好这张图拟合的很好，但肯定不行，加入平方项让其变成大值被去掉



[softmax](https://so.csdn.net/so/search?q=softmax&spm=1001.2101.3001.7020)**分类器->得各类概率值：**



**反向传播：**

用结果对相关参数分别做偏导，可以得到该参数对结果的影响

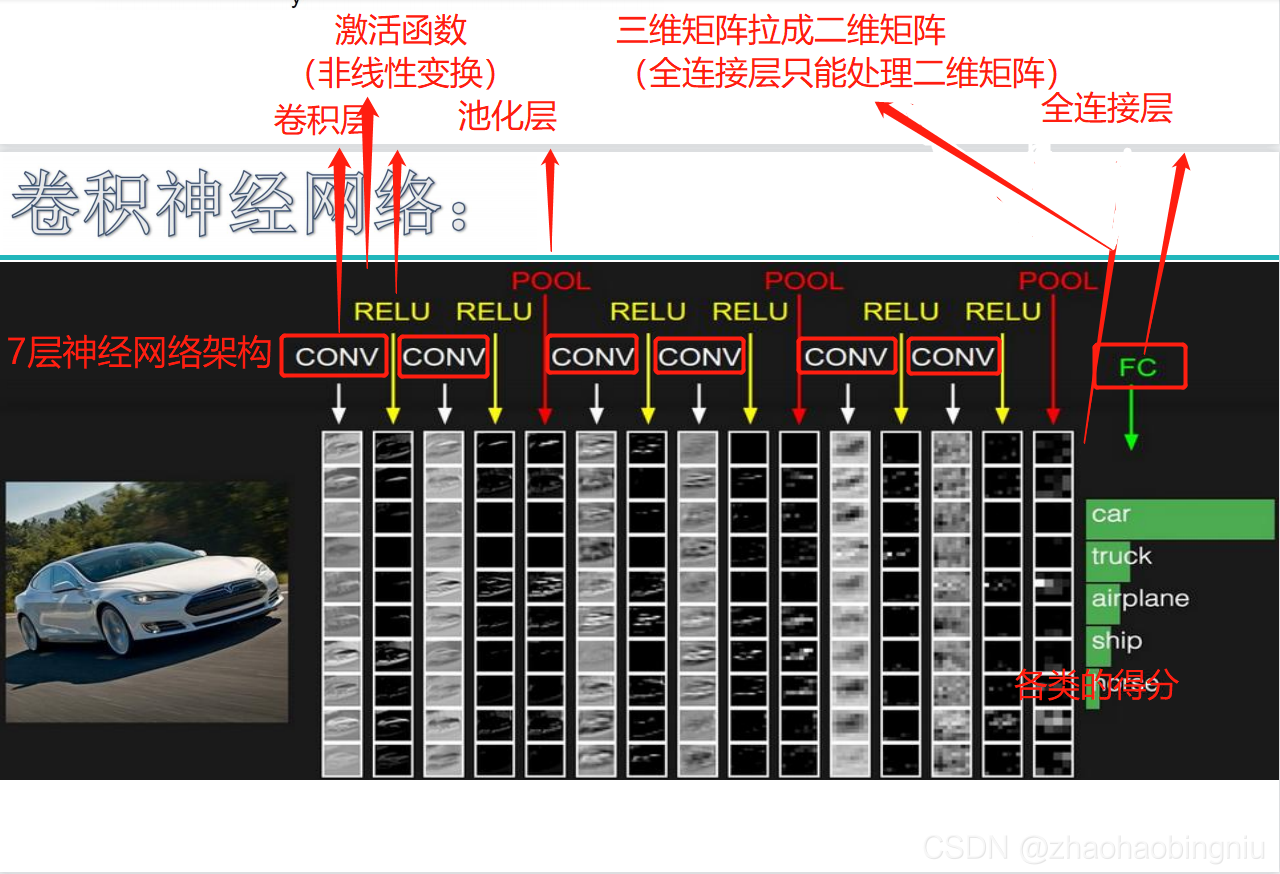


[**卷积神经网络**](https://so.csdn.net/so/search?q=卷积神经网络&spm=1001.2101.3001.7020)**（cnn，三维）：**

得到几个类别的概率值

输入层->卷积层（提取特征）->激活函数（非线性处理，卷积操作后必须有这一步）->（几次）池化层（压缩特征）->全连接层（加上权重，输入层和隐层的矩阵运算）

卷积神经网络架构：



**卷积层：**

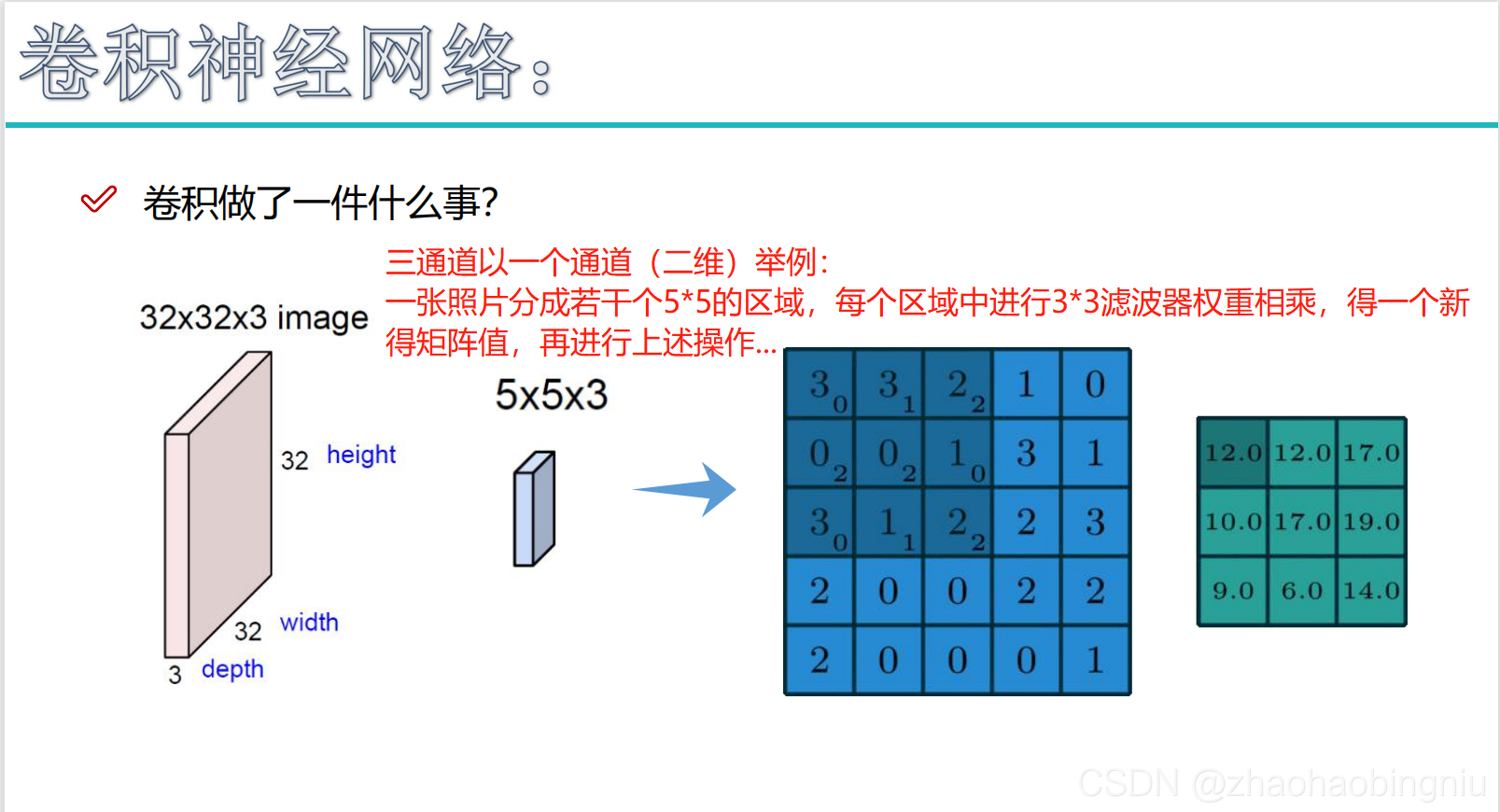
一般卷积是3\*3核，步长为1

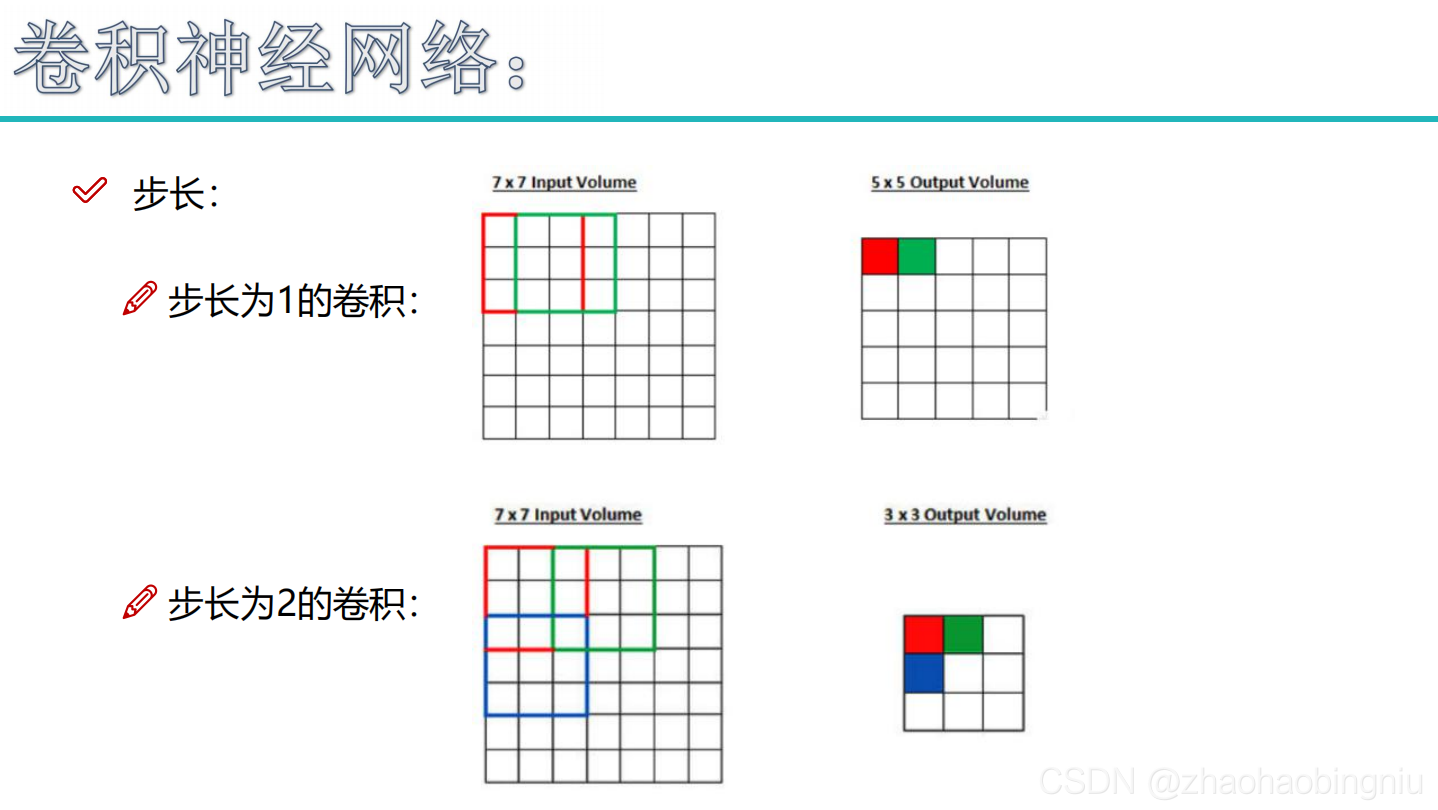
要得到几个特征值就选几个卷积核

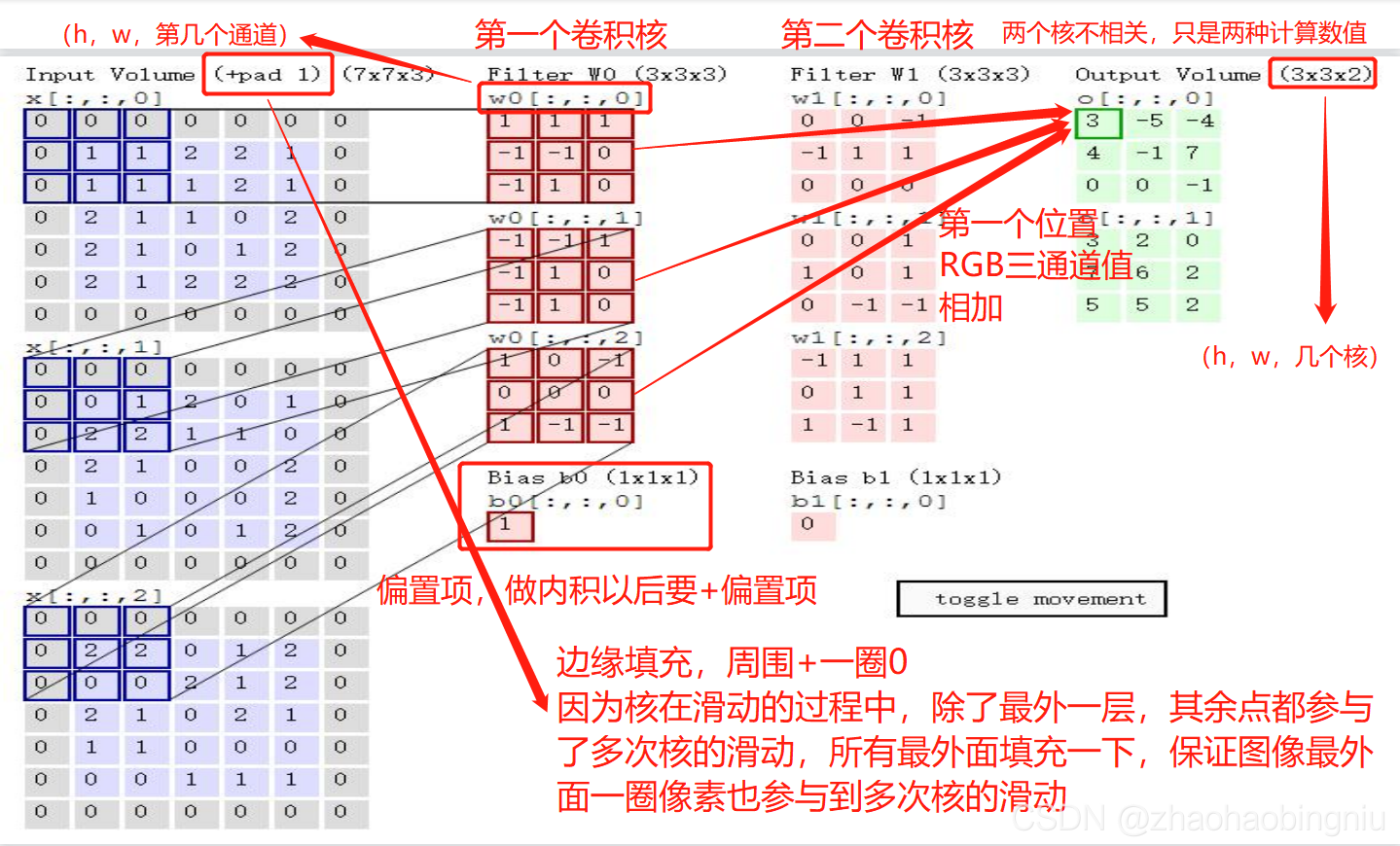
三个通道分别计算然后加在一起

相乘（做内积）后要 + b（偏置项）

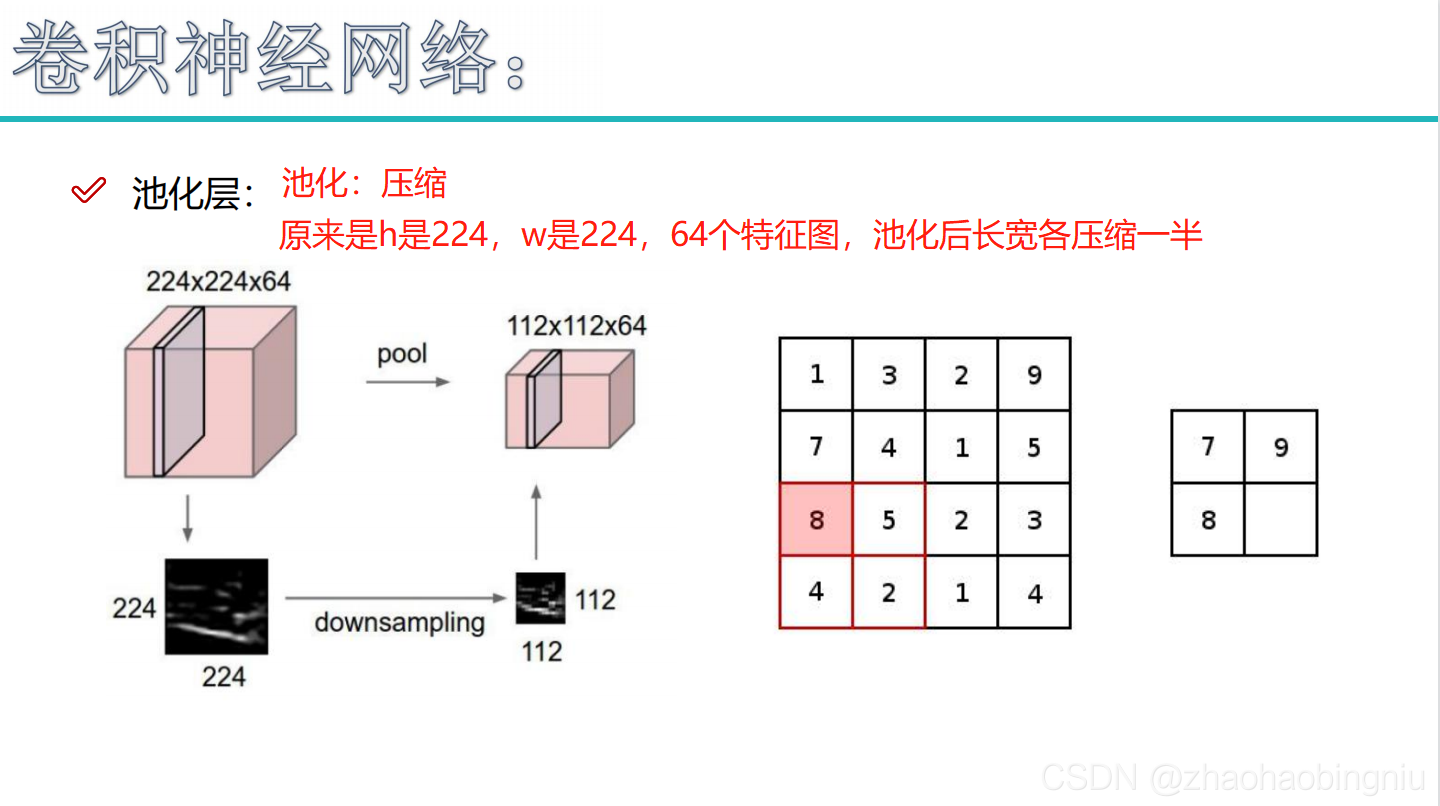
绿色的为一次卷积得到的特征图（filter）







**池化层：**



推荐B站看一下唐宇迪讲解，很通俗易懂

[4天教会你深度学习|Opencv+PyTorch+CNN+Python入门到实战课程](https://www.bilibili.com/video/BV1KP4y147Qx?p=29)