**颜色判断：**

数字图片在计算机中的表达形式：

点阵( h ╳ w )： 一般我们所说的多少多少像素，就是长\*宽的结果。

彩色图像：点阵图中每个点是彩色，常使用RGB方式来表示彩色图片，每个点会用一个三维向量表示，分别代表红、绿、蓝这三种颜色的数值，这三项也叫做该点颜色的通道。此时整个图片的数据会是一个三维数组，前两维代表行列，最后一维代表通道，为3。

灰度图像：每个点是黑白的，但是黑色有深浅之分。每个点的数值一般用[0,255]范围内的一个数字来代表该点黑色的深浅程度；如果也用三维数组表示，它的通道数为1。

二值图：(0：黑色 ；1：白色)

**卷积神经网络(CNN)原理：**

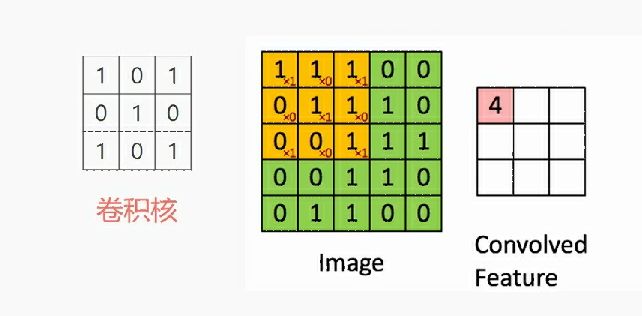
卷积神经网络不再是对图片的一个个像素点进行处理，而是使用卷积核对一块像素区域进行扫描，这种方法是为了加强图像信息的连续性，加深神经网络对图像的理解。CNN 包含以下层次：输入层、卷积层、激励层、池化层、全连接层。

进行图像分类时，首先要设定好类别数量。例如在车牌颜色判断时可以设定，黄、蓝、绿、白、黑五个类别。我们将图像输入到计算机，计算机最终输出的是图像主要内容分别属于每种类别的概率。实际上，计算机会输出一个包含类别与相应概率的列向量，如 [ 黄 0.01, 蓝 0.96, 绿 0.01, 白 0.01，黑 0.01] 识别与分类的任务将由卷积神经网络完成。

**特征图**：进行图像分类时，输入图像与训练集中的图像中的像素值通常难以一一配对，对此可以通过匹配局部区域的方法完成识别。相同的局部叫做特征（representation），或被称为特征图（feature map）。较多的 feature map 通常能带来更好的识别结果，但过多的 feature map 会引发参数冗余、过拟合等问题。

**卷积：**卷积层能完成对图像的特征采集工作，通常为RGB图像。由特征图（feature map）作为滤波器(简称卷积核)的子集，核是一个包含若干权重的神经元，表现为一个由权重组成的矩阵，其覆盖范围不大于图像。核将平移过每张特征图，在每一个位置都对元素和权重进行点积，并记录在输出矩阵的相应位置，这一操作被称为卷积滤波。对每张特征图进行卷积滤波后可得到一个完整的输出矩阵。

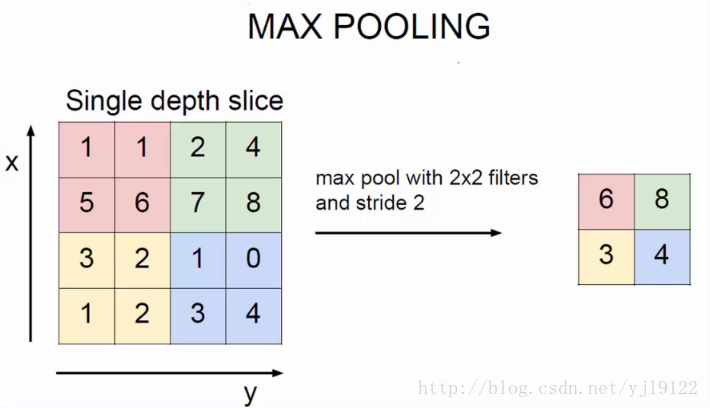
**卷积的功效**：如果不采用将图像卷积而直接把原图像作为输出图像用，会产生大量参数，增加计算量。除了减少参数，卷积还能防止过拟合。



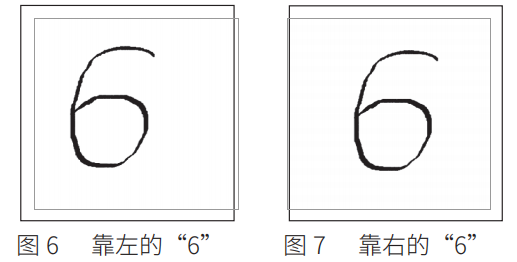
**激活函数：**卷积是一种线性运算而现实生活中得到的数据都是非线性的。因此，我们要给 特征图(feature map) 中的像素的加入非线性。在激励层中，运用非线性函数进行数据处理能够为线性函数增加非线性，增强网络的表达能力。

relu函数是卷积神经网络中很常见的非线性函数。由 图 像 可 知， 经 过ReLU 操作，大于等于零的像素值保持不变，小于零的像素值被改为 0。这使得许多无关参数被直接忽略，从而减小了计算量。在后面的神经网络设计中，笔者采用的也是 ReLU 函数。

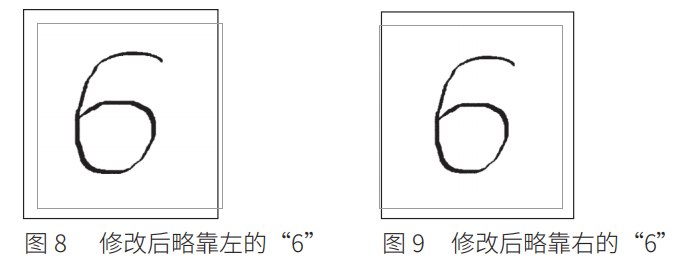
**池化（pool）：**经过卷积和 ReLU 操作，神经网络已从图像中提取出多个 feature map，并对其进行初步筛选，将其非线性化。但此时 feature map 依然过于庞杂，因此还需要通过池化来减小规模。池化有多种类型，如 Max, Average, Sum 等。大多数时候 Max（最大池化）的使用效果更好。此处以 Max 为例进行说明。图 5 中展示的是对一个经过卷积和 ReLU 操作后的 feature map 进行池化处理的流程。定义一个 2×2 的矩阵，取其中最大的像素值，记录到输出矩阵中。矩阵可以平移，直至所有的区域内最大值都被记录下来。由此，输入feature map 的维度被降低，即把一张高分辨率的图片转化为一张低分辨率的图片，但其中最关键的信息得到完整保留，同时减少了冗余记忆，有利于防止过拟合。



池化的另一个作用是：使图像小的平移与变形趋于稳定。为了解释这一作用，可以举一个例子：



很多时候，人们关注的是图像某个局部在全图中的相对位置。例如以上两张图片，除了“6”的位置不一致，并没有其他差别。在人看来，这几乎是两个一模一样的“6”而计算机认为这是两张截然不同的图片。池化能减少这种较小的平移或形变对图像分类的影响，由最大池化的原理可知，feature map 的维度被降低后，图像的该部分区域就会变小。对每一部分都进行该操作，整体视觉效果方面，如果使两张图重叠，透过上层的图像看，两个“6”的相对距离也缩小了。经过多次的池化操作，两个“6”的相对距离会越来越小，逐渐趋近于零。这样，计算机就都能识别出两张图中的“6”了。



卷积层、激励层、池化层的叠用：

卷积层、激励层、池化层可以视为一个基本模块，这个基本模块将在整个图像分类过程中多次出现。前一个模块的输出即作为下一个模块的输入。

**随机丢弃层(Dropout):**

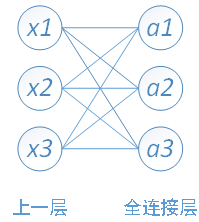
在每个训练批次中，随机丢弃层采用对上一层输出的结果按一定概率随机丢弃，即忽略一部分该层的特征检测器(让一部分隐藏层节点值为0)，可以提高神经网络的容错性和健壮性，明显减少**过拟合现象**。

**过拟合现象：**指神经网络训练的结果对训练数据过度依赖而在实际预测计算时反而不够准确。在训练数据搜集的范围不足够全、准确度不足够高的情况下，即使将神经网络的误差训练到极小，预测时如果输入数据与训练数据差异较大，也可能造成神经网络预测结果偏移很大。

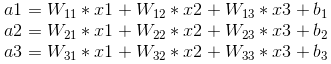
**全连接层概述：**

全连接层的作用是对特征进行高度精简化。

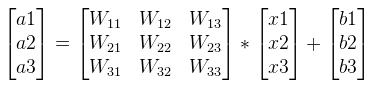
简单介绍：



其中，x1,x2,x3为全连接的输入，a1,a2,a3为输出，则有：



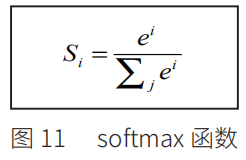
写成矩阵形式：



**softmax 函数：**

一般网络为多分类问题，一般会在全连接层后面接一个softmax层，得到1\*n的向量，n为输出类别，此向量的每个值表示这个样本属于每个类的概率。避免上溢和下溢，方便后面的损失函数做运算。

softmax 函数表达式如图 11 所示：



**损失函数(loss):** 用来表示**真实值(y)**与**预测值(y\_)**的差距。训练神经网络是通过不断改变神经网络中所有参数，使损失函数不断减小，从而训练出更高的准确率的神经网络模型。常用的损失函数有均方误差、自定义、交叉熵等。

**均方误差(mes)：**指真实值(y\_)与预测值(y)之差的平方和，再求平均值。

计算公式：

,m表示有m个样本

例如：二分类对错问题：真实值为1，预测值为0.2时，在做计算做计算需要将结果转化为二维向量，真实值为[1,0] ；预测值为[0.2，0.8]，向量的第一个数代表“对”的概率，第二个数代表“错”的概率。那么均方误差就是，

如果真实值为[1,0] ；预测值为[0.8，0.2],则均方误差为：

，可以看出二分类问题，均方误差其实一般会把误差缩小，但仍在[0,1]范围之内。

**交叉熵：**表示两个概率分布之间的距离。交叉熵越大，两个概率分布之间的距离越远，两个概率分布越相异；交叉熵越小，两个概率分布越近，两个概率越相似。

计算公式：



例：

真实值y\_=(1,0)；

第一个神经网络预测结果为y1=(0.6,0.4)；第二个神经网络预测结果为y1=(0.8,0.2) ，判断哪个神经网络模型更加接近标准答案。

根据公式可得：





由于0.222>0.097,所以预测结果y2与真实值y\_更接近，预测得更准。

**前向传播算法：**

当我们使用前馈神经网络接收输入x并产生输出y时，信息通过网络向前流动。输入x提供初始信息，然后传播到每一层的隐藏单元，最终产生输出y,这称之为**前向传播**。在训练中，前向传播可以持续向前直到它产生一个标量代价函数。

**反向传播算法：**

允许来自代价函数的信息通过网络向后流动，以便计算梯度。反向传播这个术语经常被误解为用于多层神经网络的整个学习算法。实际上，反向传播仅指用于计算梯度的方法，而随机梯度下降，则使用该梯度来进行学习。我们最常需要的梯度是代价函数关于参数的梯度，即IMG_256

**优化器：**

它的主要作用是根据反向传播算法中计算损失函数(loss)的梯度来计算，并调节可变参数。

它的核心思想是：要计算出最小的函数损失值，必须先计算出损失函数的梯度，然后按照梯度的方向使函数损失值逐渐减少，通过对权值的不断更新调整，使得函数损失值达到最小，从而获得最优解。

**SGD(随机梯度下降算法)**：是一个基于梯度下降的改进算法,SGD每次随机选择一个样本来迭代更新一次，而不是针对所有的样本。因此该算法明显的降低了计算量。

IMG_256

上式中，IMG_256表示学习率，IMG_257表示随机选中的样本，IMG_256损失函数J(θ)的方向导数，即梯度。

卷积神经网络的训练是通过前向传播和反向传播过程来实现的。前向传播是通过输入计算出神经元的激活值，是自下向上的过程；反向传播根据损失计算梯度，更新网络权值，是自上向下的过程。