CNN卷积神经网络整理

一、人工神经网络

1.1 人的视觉原理

最底层的特征基本上都是类似的（就是各种边缘）

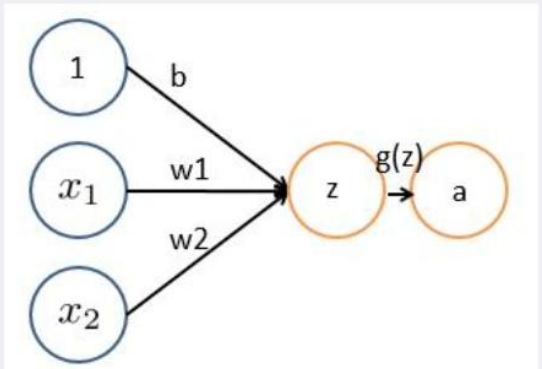
提取出该类物体的一些特征（车轮、眼睛、躯干）

不同的高级特征最终组合成相应的图像，从而让人类准确地区分不同的物体

1.2 神经网络

g(z) = g(w1 * x1+ w2 * x2 + b)，g表示激活函数，这里的b可以理解成 为更好达到目标而做调整的偏置项

神经网络的每个神经元如下

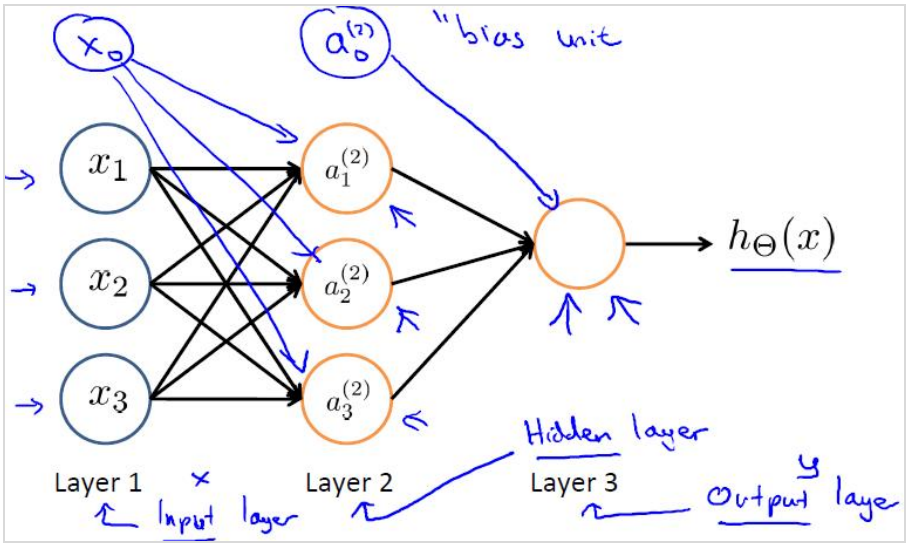


基本wx + b的形式，其中

- x_1、x_2表示输入向量
- w_1、w_2为权重，几个输入则意味着有几个权重，即每个输入都被赋予一个权重
- b为偏置bias
- g(z) 为激活函数
- a 为输出

如果每一层都可能由单个或多个神经元组成，每一层的输出将会作为下一层的输入数据，则

↓输入层和输出层中间夹着数层隐藏层，层和层之间是全连接的结构，同一层的神经元之间没有连接



最后

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$
$$h_\Theta(x) = a_1^{(3)} = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

二、卷积神经网络CNN简介

2.1定义

一类包含卷积计算且具有深度结构的**前馈神经网络**，是深度学习的代表算法之一，是**仿造生物的视知觉**机制构建的。具有**表征学习能力**，能够按其阶层结构对输入信息进行**平移不变分类**

2.2推荐文章

1. Deep Residual Learning for Image Recognition
2. Convolutional Neural Network （Hung-yi Lee）
3. Batch Normalization：Acclerating Deep Network Training by Reducing Internal Covariate Shift
4. ImageNet Classification with Deep Convolutional Neural Networks
5. Deep Inside Convolutional Networks：Visualising Image Classification Models and Saliency Maps
6. ResNeSt：Split-Attention Networks
7. YOLOv4:Optimal Speed and Accuracy of Object Detection

2.3CNN的三维体积神经元

width

height

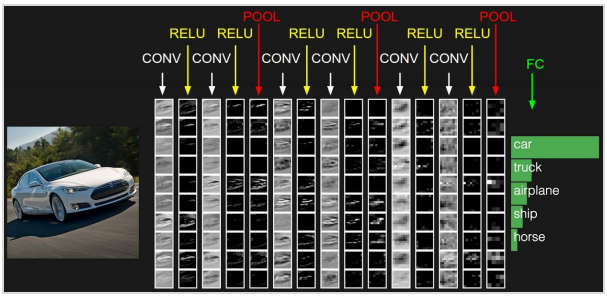
depth（神经元的维度）

> 例如一张图片我们描述为32×32×3（rgb），那么输入神经元也就具有32×32×3的维度

三、卷积神经网络的层次结构

3.1典型CNN模型

1. 卷积层（Convolutional Layer）：提取图像中的局部特征
2. 池化层（Pooling Layer）：大幅度降低参数量级，防止过拟合=降维，即取区域平均或最大
3. 全连接层（Fully-Connected Layer）：输出结果



上图中CNN要做的事情是：给定一张图片，是车还是马未知，是什么车也未知，现在需要模型判断这张图片里具体是一个什么东西，总之输出一个结果：如果是车 那是什么车

所以

- 最左边是数据输入层，对数据做一些处理，比如去均值（把输入数据各个维度都中心化为0，避免数据过多偏差，影响训练效果）、归一化（把所有的数据都归一到同样的范围）、PCA/白化等等。CNN只对训练集做“去均值”这一步。

中间是

- CONV：卷积计算层，线性乘积 求和。
- RELU：激励层，上文2.2节中有提到：ReLU是激活函数的一种。
- POOL：池化层，简言之，即取区域平均或最大。

最右边是

- FC：全连接层

3.2更常见的CNN

1.输入层（Input Layer）：输入数据

2.卷积层（Convolutional Layer）：使用**卷积核**进行特征提取和特征映射

3.线性整流层（Rectified Linear Units Layer，ReLU layer）：为CNN增加非线性映射，ReLU是激活函数的一种

4.池化层（Pooling Layer）：**下采样**，对特征图稀疏处理，减少数据运算量

5.全连接层（Fully-Connected Layer）：在CNN的尾部进行重新拟合，减少特征信息的损失

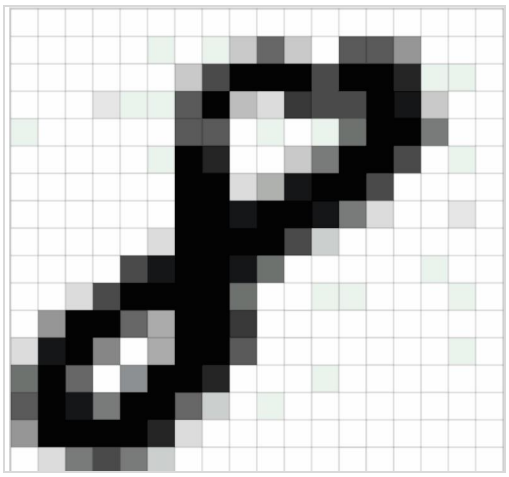
6.输出层（Output Layer）：输出结果

3.3其中经常使用的功能层

1. 归一化层（Batch Normalization）：对CNN中特征进行归一化
2. 切分层：对某些数据进行区域分割进行单独学习
3. 融合层：对独立进行的特征学习分支进行融合

输入层

输入图像，首先要将其转换为对应的二维矩阵，这个二维矩阵就是由图像每一个像素的像素值大小组成的，并将此二维矩阵存储，等待后面几层的操作。

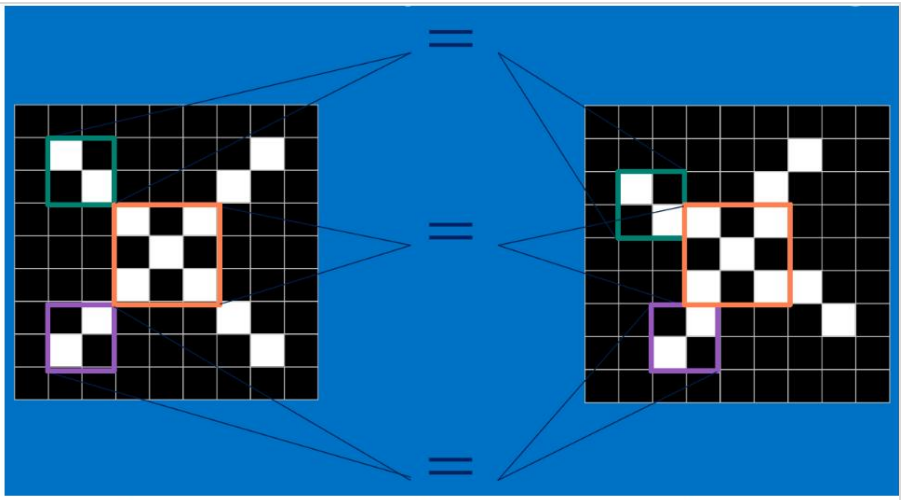


0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	12	0	11	39	137	37	0	152	147	84	0	0	0
0	0	1	0	0	0	41	160	250	255	235	162	255	238	206	11	13	0
0	0	0	16	9	9	150	251	45	21	184	159	154	255	233	40	0	0
10	0	0	0	0	0	145	146	3	10	0	11	124	253	255	107	0	0
0	0	3	0	4	15	236	216	0	0	38	109	247	240	169	0	11	0
1	0	2	0	0	0	253	253	23	62	224	241	255	164	0	5	0	0
6	0	0	4	0	3	252	250	228	255	255	234	112	28	0	2	17	0
0	2	1	4	0	21	255	253	251	255	172	31	8	0	1	0	0	0
0	0	4	0	163	225	251	255	229	120	0	0	0	0	0	11	0	0
0	0	21	162	255	255	254	255	126	6	0	10	14	6	0	0	9	0
3	79	242	255	141	66	255	245	189	7	8	0	0	5	0	0	0	0
26	221	237	98	0	67	251	255	144	0	8	0	0	7	0	0	11	0
25	255	141	0	87	244	255	208	3	0	0	13	0	1	0	1	0	0
45	248	228	116	235	255	141	34	0	11	0	1	0	0	0	1	3	0
85	237	253	246	255	210	21	1	0	1	0	0	6	2	4	0	0	0
6	23	112	157	114	32	0	0	0	0	2	0	8	0	7	0	0	0

卷积计算层

4.1 CNN怎么进行识别

计算机存储一张标准的“X”图案，然后把需要识别的未知图案跟标准"X"图案进行一个局部一个局部的比对

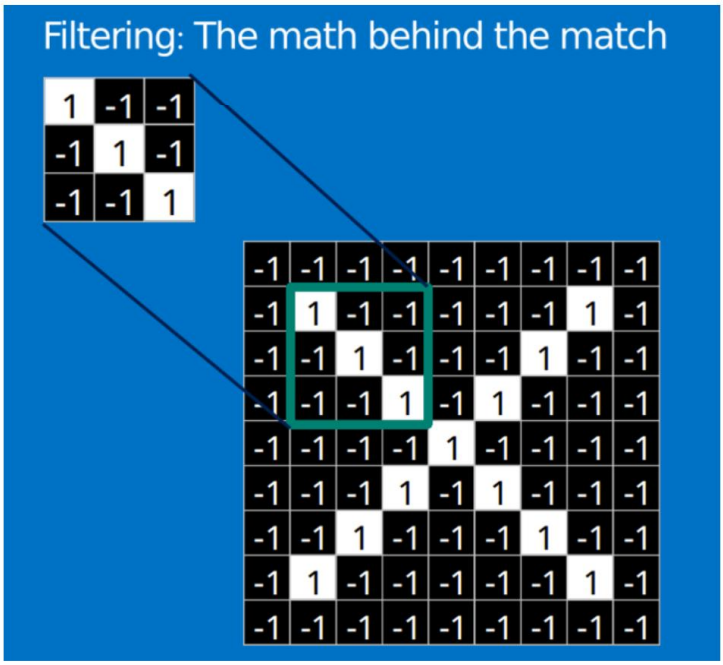


而未知图案的局部和标准X图案的局部一个一个比对时的计算过程，便是卷积操作。卷积计算结果为1表示匹配，否则不匹配。

对于CNN来说，它是一块一块地来进行比对。它拿来比对的这个“小块”我们称之为Features（特征）。在两幅图中大致相同的位置找到一些粗糙的特征进行匹配，CNN能够更好的看到两幅图的相似性，相比起传统的整幅图逐一比对的方法

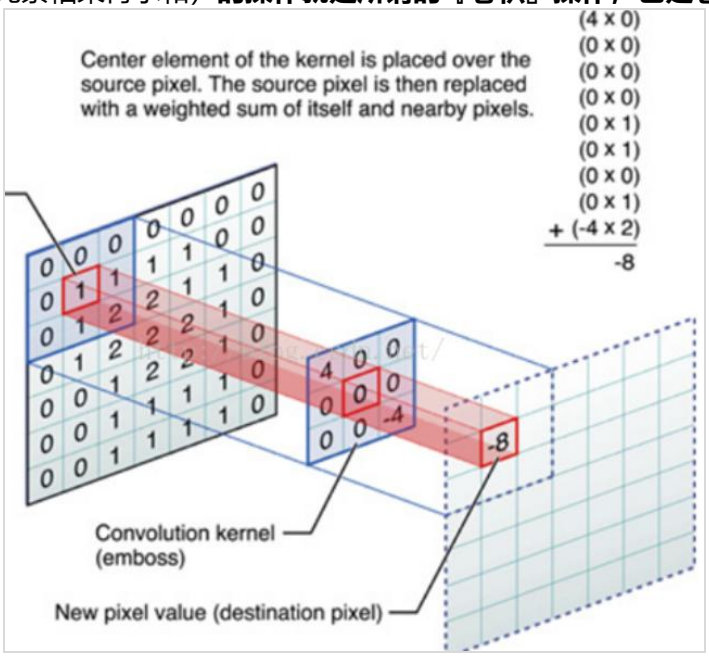
4.2 什么是卷积操作

看到这里是不是有了一点头目呢。但其实这只是第一步，你知道了这些Features是怎么在原图上面进行匹配的。但是你还不知道在这里面究竟进行的是怎样的数学计算，比如这个下面3*3的小块到底干了什么？



这里面的数学操作，就是我们常说的"卷积"操作。接下来，我们来了解下什么是卷积操作。

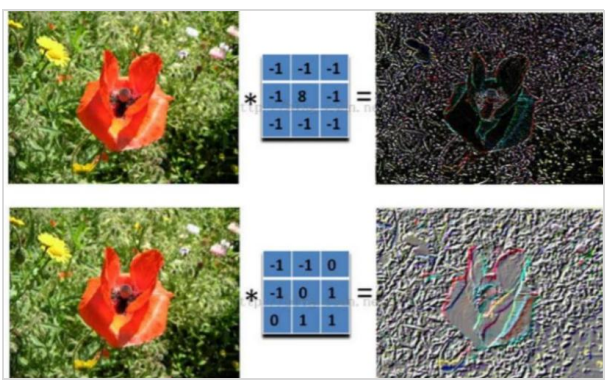
对图像（不同的数据窗口数据）**和滤波矩阵**（一组固定的权重：因为每个神经元的多个权重固定，所以又可以看做一个恒定的滤波器filter）**做内积**（逐个元素相乘再求和）**的操作就是所谓的『卷积』操作，也是卷积神经网络的名字来源**



多个滤波器叠加便成了卷积层

4.3 图像上的卷积

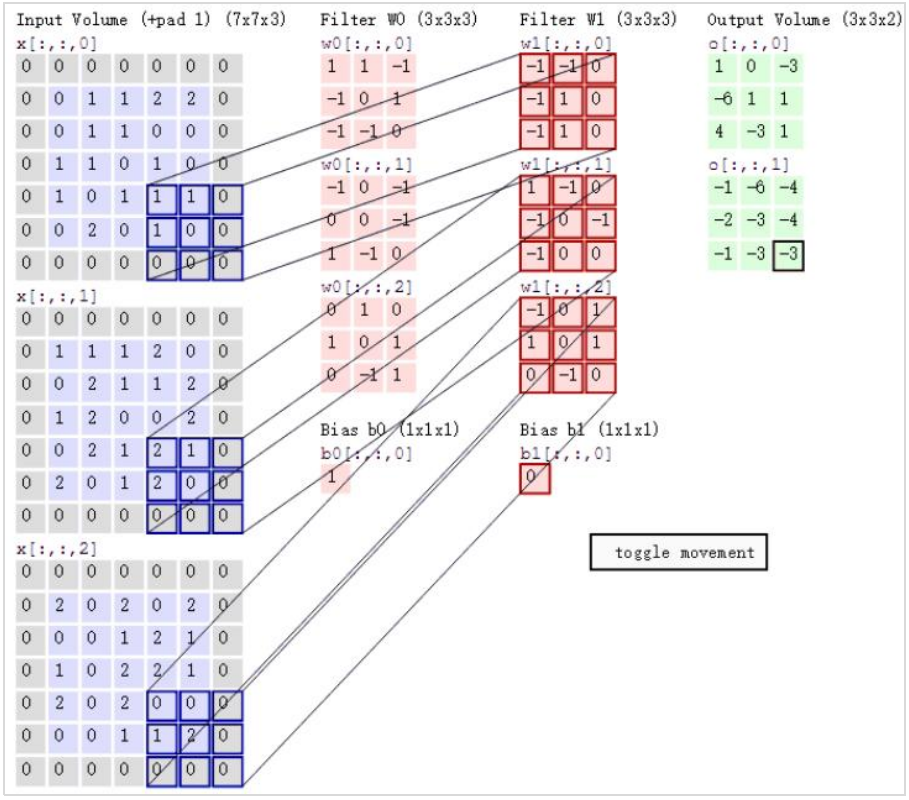
具体来说，左边是图像输入，中间部分就是滤波器filter（带着一组固定权重的神经元），不同的滤波器filter会得到不同的输出数据，比如颜色深浅、轮廓。相当于**如果想提取图像的不同特征，则用不同的滤波器filter**，提取想要的关于图像的特定信息：颜色深浅或轮廓

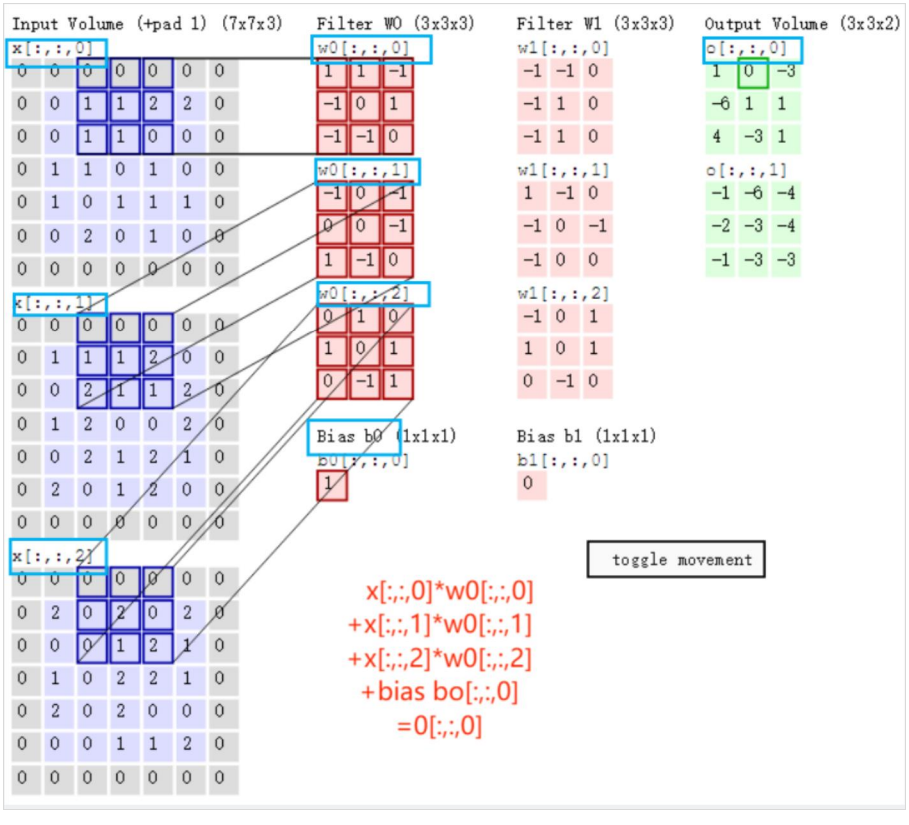


4.4 GIF动态卷积图

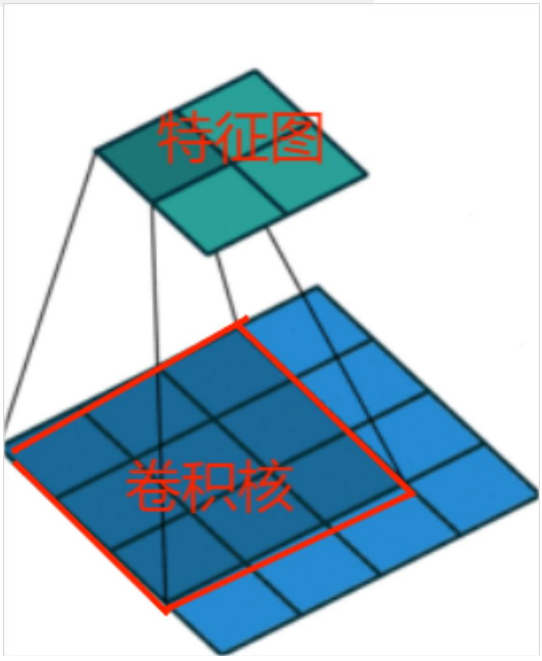
在CNN中，滤波器filter（带着一组固定权重的神经元）对局部输入数据进行卷积计算。每计算完一个数据窗口内的局部数据后，数据窗口不断平移滑动，直到计算完所有数据。这个过程中，有这么几个参数：

1. 深度depth：神经元个数，决定输出的depth厚度。同时代表滤波器个数。
2. 步长stride：决定滑动多少步可以到边缘。
3. 填充值zero-padding：在外围边缘补充若干圈0，方便从初始位置以步长为单位可以刚好滑倒末尾位置，通俗地讲就是为了总长能被步长整除。
   如果每次计算的时候，边缘只被计算一次，而中间被多次计算，那么得到的特征图也会丢失边缘特征，最终会导致特征提取不准确，那为了解决这个问题，我们可以在原始的输入图像的二维矩阵周围再拓展一圈或者几圈
4. 下图：depth=3，stride=2，zero-padding=1，展示的是深度为3的图像和2个卷积核进行运算，输出为2个特征图
5. 左边数据在变化，每次滤波器都是针对某一局部的数据窗口进行卷积，这就是所谓的CNN中的**局部感知**机制
6. 与此同时，数据窗口滑动，导致输入在变化，但中间滤波器Filter w0的权重（即每个神经元连接数据窗口的权重）是固定不变的，这个权重不变即所谓的CNN中的**参数（权重）共享**机制。





本来一个彩色图片会有三个二维矩阵，只有两个卷积核所以只有两个特征图



首先**卷积核也是一个二维矩阵**，这个二维矩阵要比输入图像的二维矩阵要小或相等，卷积核通过在输入图像的二维矩阵上不停的移动，每一次移动都进行一次乘积的求和，作为此位置的值。整个过程就是一个降维的过程，通过卷积核的不停移动计算，可以提取图像中最有用的特征。通常将卷积核计算得到的新的二维矩阵称为特征图，有几个卷积核就有几个特征图

CNN的激励层和池化层

5.1 激励层

常用的非线性激活函数有sigmoid、tanh、ReLU等等，前两者sigmoid/tanh比较常见于全连接层，后者relu常见于卷积层

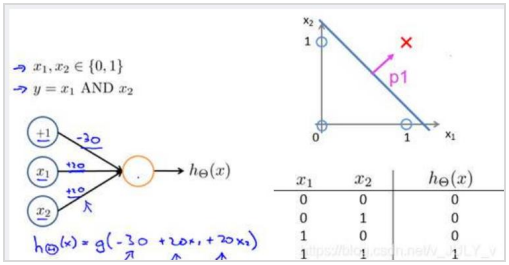
sigmoid函数

$$g(z) = \frac{1}{1+e^{-z}}$$

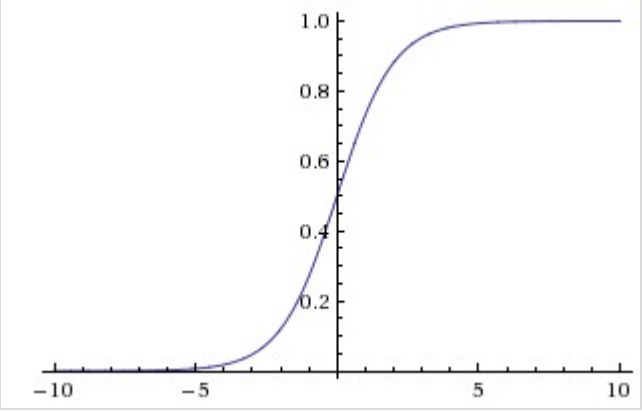
其中z是一个线性组合，比如z可以等于b + w1* x1 +w2 * x2

也就是说，**sigmoid函数的**功能是相当于把一个实数压缩至0到1之间。当z是非常大的正数时，**g(z)**会趋近于1，而z是非常小的负数时，则g(z)会趋近于0

压缩至0到1有何用处呢？用处是这样一来便可以把激活函数看作一种"分类的概率"，比如激活函数的输出为0.9的话便可以解释为90%的概率为正样本

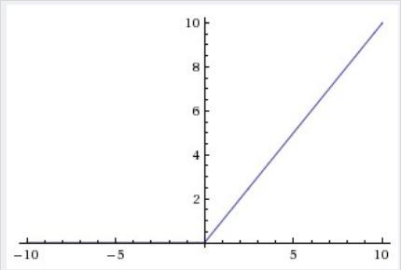


因此，sigmoid函数g(z)的图形表示如下（横轴表示定义域z，纵轴表示值域g(z)）



ReLU函数

收敛快，求梯度简单

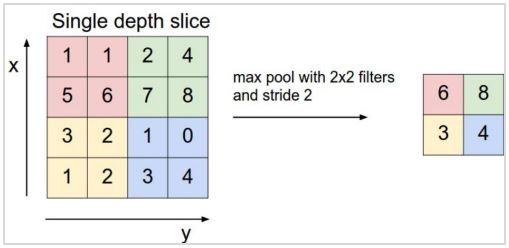


5.2 池化pool层=下采样

过多的特征并不需要，会带来过拟合、维度过高的问题

池化，简言之，即取区域平均或最大

取区域平均



参数说明：

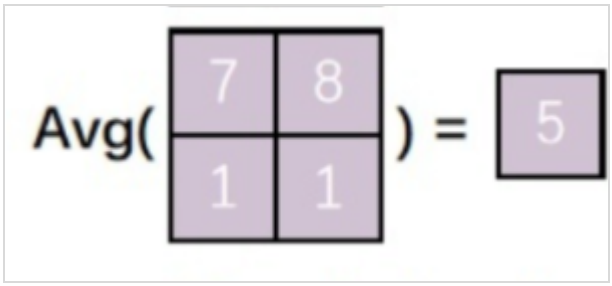
① kernel_size = 2：池化过程使用的正方形尺寸是2×2，如果是在卷积的过程中就说明卷积核的大小是2×2

② stride = 2：每次正方形移动两个位置（从左到右，从上到下），这个过程其实和卷积的操作过程一样

③ padding = 0：这个之前介绍过，如果此值为0，说明没有进行拓展

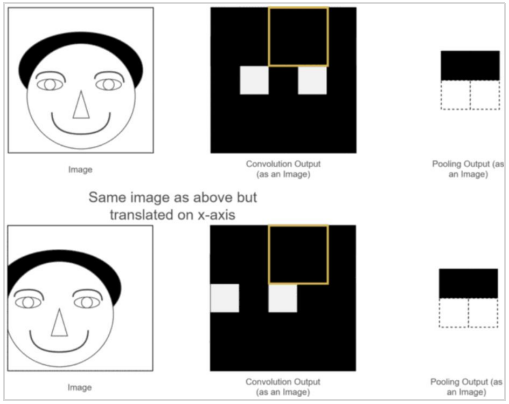
取区域最大

平均池化时采用向上取整



- 在减少参数量的同时，还保留了原图像的原始特征
- 有效防止过拟合

- 为卷积神经网络带来平移不变性，即平移图像不会造成认证失误



全连接层

一个线性特征映射的过程，将多维的特征输入映射为二维的特征输出，高维表示样本批次，低维常常对应任务目标(例如分类就对应每一个类别的概率)。全连接层主要对特征进行重新拟合，减少特征信息的丢失

g(z) = g(w1 * x1+ w2 * x2 + b)，g表示激活函数，这里的b可以理解成 为更好达到目标而做调整的偏置项