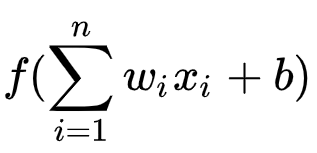
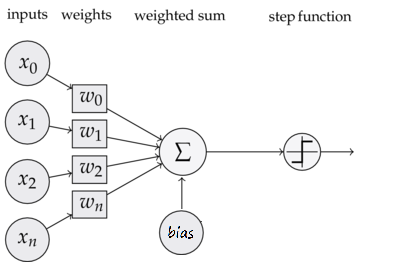
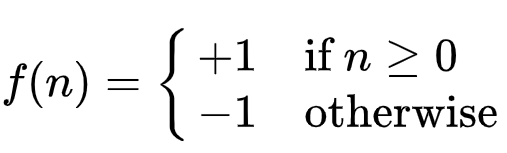
℗人工神经网络[Artificial Neural Network, ANN]，简称神经网络[NN]，是一种模仿生物神经网络（也就是大脑）的结构和功能的模型，用于对函数进行估计或近似计算。

℗受到生物（人或其他动物）神经网络功能的运作启发，人工神经网络由大量的运算节点，也就是人工神经元[neurons]，构成。大多数情况下人工神经网络能在外界信息的基础上改变内部结构，是一种自适应系统，通俗的讲就是具备学习功能。

℗神经网络是一种非线性统计性数据建模工具。它通常是通过一个基于统计学的学习方法进行训练，从而得到优化。通过统计学方法，我们既能够得到大量的可以用函数来表达的局部结构，也可以让神经网络获得简单的决定能力和简单的判断能力。同人类的猜想一样，这种方法比起严谨逻辑学推理演算来说，更快更简洁，因而在复杂问题的近似处理上更具有优势。

℗感知器[Perceptron]是康奈尔航空实验室的Frank Rosenblatt于1957年发明的一种人工神经网络。它是一种极简单的神经网络，是生物神经元的简单抽象。大家在高中生物中已经学到，神经元大致可分为胞体、树突、轴突。单个神经元可被视为一种只有两种状态的机器——兴奋时为{是}，静息时为{否}，神经元是否兴奋态取决于它的自身性质和收到的来自其他神经元的输入信号。当促进兴奋的信号量总和超过了某个阈值时，神经元就会兴奋，产生电脉冲，并影响后续神经元。为了模拟神经细胞行为，与之对应的感知机的基础概念被提出，如℗{ on PPT }。

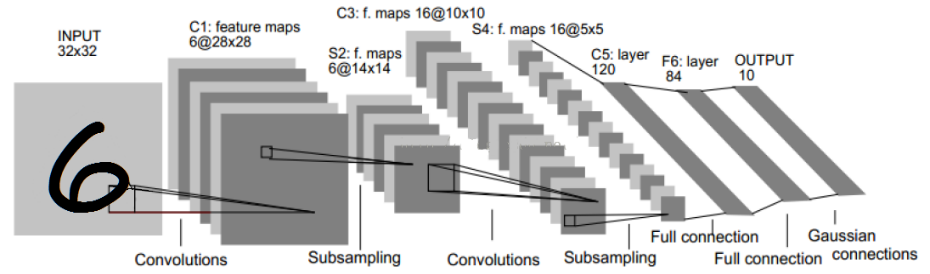
{讲解，图与公式如上}事实上，感知器可以拟合任何的线性函数，任何线性分类问题都可以用感知器来解决。{AND}/{OR}运算可以看作是二分类问题。然而，感知器却不能实现{XOR}运算，异或运算不是线性的，所以你无法用一条直线把分类0和分类1分开。

℗多层感知器[Multilayer Perceptron, MLP]是感知器的推广，克服了感知器不能对线性不可分数据进行识别的弱点。MLP由多个节点层组成，每一层内神经元都全连接到下一层每个神经元（也称为全连接）；除了输入节点，每个节点都带有非线性激活函数。MLP可以将一组输入向量映射到一组输出向量。MLP本身可以使用任何形式的激活函数，但因为MLP往往需要解决连续数据的问题，并且要使用反向传播算法进行学习，激活函数必须可导，常用的函数有sigmoid函数{y=1/(1+e^(-x))}，双曲正切函数及softplus函数{y=log(1 + e^x)}。

℗深度学习是通过组合低层特征，形成更加抽象的高层的属性、类别或特征，以发现数据的分布特征。 深度学习的概念由Hinton等人于2006年提出，而若干年后出现的卷积神经网络才是第一个真正的多层结构学习算法。深度学习的观测值（例如一幅图像）可以使用多种方式来表示，如像素值向量，或者更抽象的一系列边、特定形状的区域等。而深度学习利用观测值进行学习（例如，人脸识别或面部表情识别），深度学习就可以利用特征学习和分层特征提取高替代手工获取特征。这正巧与人工神经网络的功能吻合。℗深度学习的主要应用方向是：{on PPT}

℗要谈卷积神经网络，当然要从卷积[Convolution]说起。{进Demo}

℗再就是卷积神经网络了。卷积神经网络[Convolutional Neural Networks, CNN]是一类包含卷积或相关计算且具有深度结构的神经网络，是深度学习（deep learning）的代表算法之一。对卷积神经网络的研究始于二十世纪80至90年代，时间延迟网络和LeNet-5是最早被证实有效的卷积神经网络算法；在二十一世纪后，随着数值计算设备的改进，卷积神经网络得到了快速发展，并被大量应用于计算机视觉、自然语言处理等领域。卷积神经网络模仿了生物的视觉知觉，其隐含层内的卷积核参数共享和层间连接的稀疏性使得卷积神经网络能够以较小的计算量对格点化（grid-like topology）特征（例如像素和音频）进行学习，并且有稳定的效果。

℗接下来我们以最经典的卷积神经网络LeNet-5为例大致讲解一下卷积神经网络的结构。输入32\*32的手写字体图片，这些手写字体包含0~9数字，也就是相当于10个类别的图片。输出分类结果，0~9之间的数。因此我们可以知道，这是一个多分类问题，总共有十个类，因此神经网络的最后输出层必然有10个。

输入层[Input layer] => 32\*32的图片，也就是相当于1024个神经元。

C1卷积层[Convolutional layer] =>选择6个特征卷积核，然后卷积核大小选择5\*5，这样我们可以得到6个特征图，然后每个特征图的大小为32-5+1=28，也就是神经元的个数为6\*28\*28=784。

S2池化层[Pooling layer] => 这就是下采样层，也就是使用最大池化进行下采样，池化的size，选择(2,2)，也就是相当于对C1层28\*28的图片，进行分块，每个块的大小为2\*2，这样我们可以得到14\*14个块，然后我们统计每个块中，最大的值作为下采样的新像素，因此我们可以得到S1结果为：14\*14大小的图片，共有6个这样的图片。

C3卷积层 => 这一层我们选择卷积核的大小依旧为5\*5，据此我们可以得到新的图片大小为14-5+1=10，然后我们希望可以得到16张特征图。这一层是最难理解的，我们知道S2包含：6张14\*14大小的图片，我们希望这一层得到的结果是：16张10\*10的图片。这16张图片的每一张，是通过S2的6张图片进行加权组合得到的，具体是怎么组合的呢？为了解释这个问题，我们先从简单的开始，我现在假设输入6特征图的大小是5\*5的，分别用6个5\*5的卷积核进行卷积，得到6个卷积结果图片大小为1\*1。我们假设输入第i个特征图的各个像素值为x1i，x2i……x25i，因为每个特征图有25个像素。因此第i个特征图经过5\*5的图片卷积后，得到的卷积结果图片的像素值Pi可以表示成：

https://img-blog.csdn.net/20150903204947566?watermark/2/text/aHR0cDovL2Jsb2cuY3Nkbi5uZXQv/font/5a6L5L2T/fontsize/400/fill/I0JBQkFCMA==/dissolve/70/gravity/Center

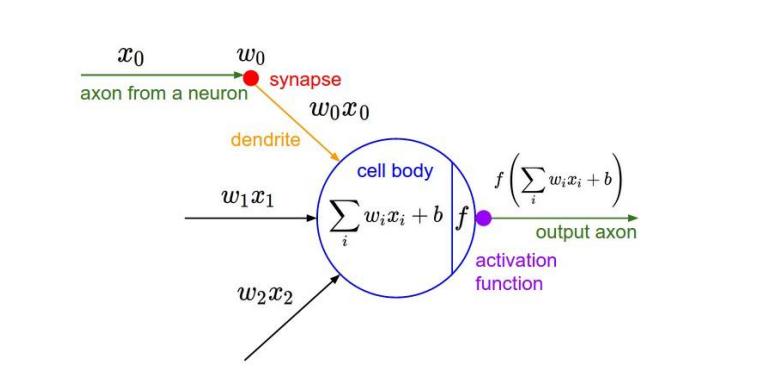
这个是卷积公式。因此对于上面的P1~P6的计算方法，直接根据公式就行。然后我们把P1~P6相加起来，也就是：P=P1+P2+…+P6。因此我们的输出特征图就是：Out=f(P+b)。这个就是从S2到C3的计算方法，其中b表示偏置项，f为激活函数。我们回归到原来的问题，根据上面的过程，也就是其实我们用5\*5的卷积核去卷积每一张输入的特征图，卷积后，我们得到6张10\*10新图片，这个时候，我们把这6张图片相加在一起，然后加一个偏置项b，然后用激活函数进行映射，就可以得到一张10\*10的输出特征图。通过调整卷积核的参数我们可以得到剩下的10\*10输出图。总之，C3层每个图片是通过S2图片进行卷积后，然后相加，并且加上偏置b，最后在进行激活函数映射得到的结果。

S4池化层 => 采样层，对C3的16张10\*10的图片进行最大池化，池化块的大小为2\*2。因此最后S4层为16张大小为5\*5的图片。至此我们的神经元个数已经减少为：16\*5\*5=400。

C5卷积层 => 我们继续用5\*5的卷积核进行卷积，然后我们希望得到120个特征图。这样C5层图片的大小为5-5+1=1，也就是相当于1个神经元，120个特征图，因此最后只剩下120个神经元了。

F6全连接层[Fully connected layer] =>这个时候，神经元的个数已经够少的了，后面我们就可以直接利用全连接神经网络，也就是前面讲到的MLP，进行这120个神经元的后续处理。

输出层[Output layer] =>在CNN中，输出层是损失函数层[Loss layer]，也就是通过某种分类算法，收集所有上一层的输入并且得到答案（输入是何种分类）。而称之为损失函数层，是因为在训练时如果答案与参考值不同时，该层会惩罚上层的对应的神经元，也就是一层层调整卷积核的参数。各种不同的损失函数适用于不同类型的任务，例如，Softmax交叉熵损失函数常常被用于在多个类别中选出一个， Sigmoid交叉熵损失函数常常用于多个独立的二分类问题，欧几里德损失函数常常用于结果取值范围为任意实数的问题。



除了LeNet-5中这些之外，最常见的还有激励层：把卷积层输出结果做非线性映射，往往紧接在卷积层之后。CNN采用的激励函数一般为ReLU(Rectified Linear Unit)，它的特点是收敛快，求梯度简单，但较脆弱。

这大概就是一般的卷积神经网络的结构。℗接下来就是卷积神经网络的运算方法了。

℗首先是输入数据的处理。为了除去异常数据和不同数据集内数据均值不同的影响，我们需要对数据经行简单处理，常用的有两套方法——去均值、标准化；去相关、白化。{进 Demo}

℗接下来就是CNN内部的运算了。℗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好比就是整张图像的特征提取器集合。