# 概述

## 名词解释

**TensorFlow《《 Caffe 《《 CNN 《《 深度学习《《 神经网络《《 感知器**

TensorFlow：谷歌发布的第二代机器学习系统

Caffe：Convolution Architecture for Feature Embedding (Extraction)，特征嵌入（提取）卷积架构，一种深度学习的框架

CNN：Convolutional Neural Network，卷积神经网络，深度学习的一种

深度学习：一种隐含层非常多的神经网络，神经元多，网络复杂

神经网络：一种模仿大脑结构和功能的数学计算模型

感知器：最简单的神经网络，一个感知器就是一个神经元

## 神经网络的分类

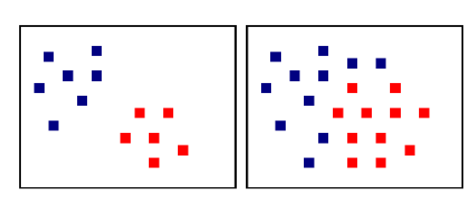


# 感知器

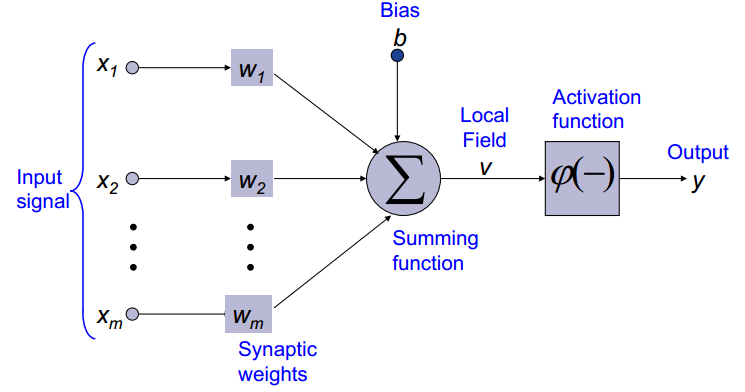
感知器是最简单的神经网络；分类器；一刀切（数据集线性可分）

神经网络中一个感知器就是一个神经元。

线性可分是指存在一个超平面将它们分开，例如下图左边线性可分，又被则不可分。已经被证明，如果两类模式是线性可分的，则算法一定收敛。



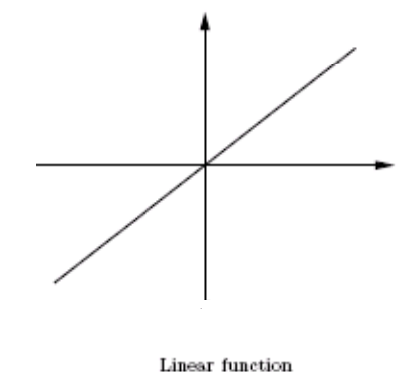
## 感知器的工作原理



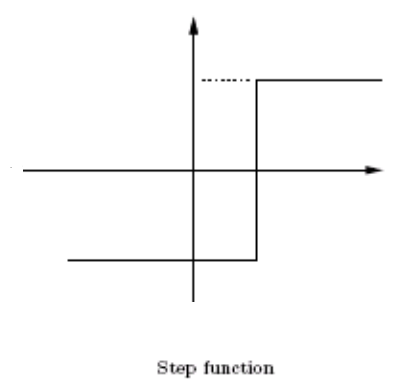
向量[]是一个输入数据，是各属性的权值。叫做激活函数。计算时，先将各属性加权求和，将结果输入激活函数，输出结果为类别。其实上图就是一个公式

## 激活函数

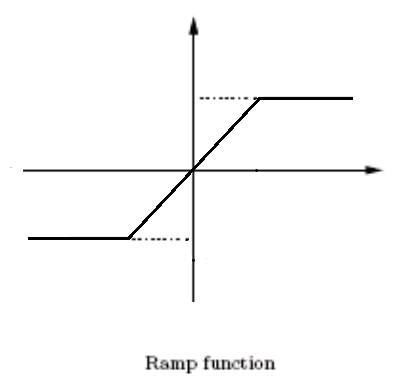
1. 线性函数 (Linear Function)



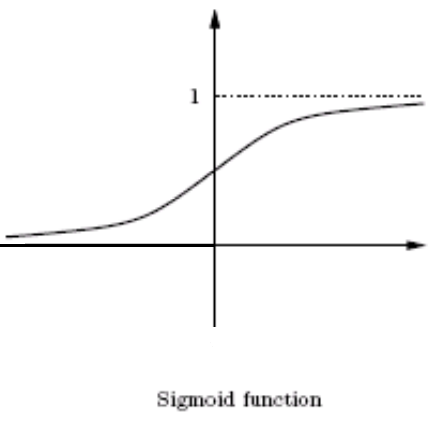
1. 阶梯函数 (Step Function)



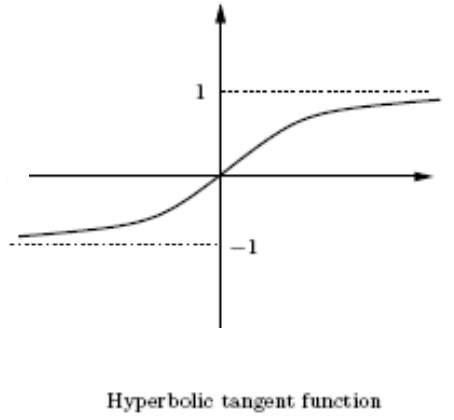
1. 斜升函数 (Ramp Function)



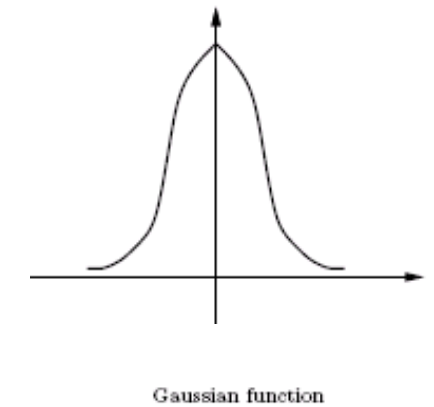
1. 逻辑函数 (Logistic Function)



1. 双曲正切 (Hyperbolic Tangent)

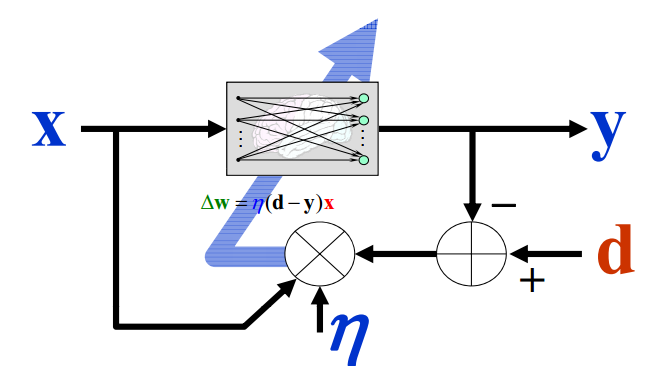


1. 高斯函数 (Gaussian Function)



**怎样学习的？**

机器学习是从数据中自动分析获得规律，并利用规律对未知数据进行预测的算法，其实是统计学习。这里的学习是基于反馈的，示意图如下：



； x：输入； y：实际输出； d：期望输出；

，不断调整w的过程就是学习的过程。

**梯度下降算法：**

在训练过程中修正权重使得网络误差最小。

在梯度下降算法中寻找的是极小值而不是最小值，收敛到局部最小就会停止。

**感知器参考学习资料：**

[1] 感知器： <http://www.cnblogs.com/dztgc/archive/2013/04/22/3036744.html> 比较基础，这些知识我第一次都讲过了，没有听懂的可以看这个复习一下。

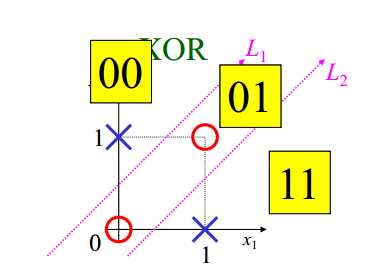
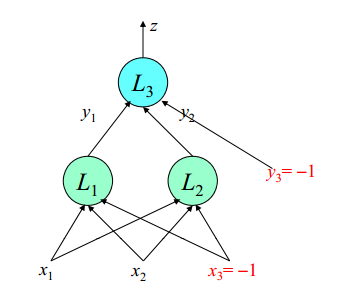
[2] 机器学习入门 - 感知器 (Perceptron)：<http://shaoxiongjiang.com/2013/03/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0%E5%85%A5%E9%97%A8-%E6%84%9F%E7%9F%A5%E5%99%A8-perceptron/> 对于理论上的经过、区别、原因等总结得很好。

[3] 《统计学习方法》读书笔记——感知机<http://www.cnblogs.com/OldPanda/archive/2013/04/12/3017100.html> 讲得相对较深，那些公式可以从本质上说明问题。如果看不懂欢迎随时问我。

**多层感知器：多个神经元多刀切**

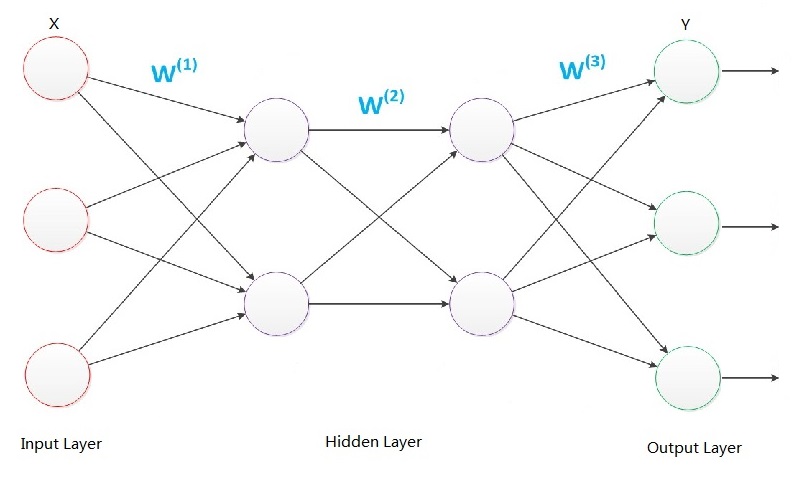
参考学习资料：系统学习机器学习之多层感知器

<http://blog.csdn.net/app_12062011/article/details/50379410> 只看第三部分多层感知器，看到反向传播之前就可以。其他部分讲得不好。

### 神经网络

其实上边那些都是神经网络。输出可以是单个值也可以是多个值。下图是一个一般的双层神经网络。



对堆叠多个层，也就是说这一层的输出作为下一层的输入，实现对输入信息的分级表达。

**BP（Back Propagation）神经网络的原理:**

对于多层的神经网络，在每层的各个神经元之间都有一个权重w，我们需要通过训练集训练得到所有的w值，这是我们的模型的目标。首先随机初始化所有的w值，在训练过程中不断修改，使其收敛接近正确的结果。

前向过程：

从输出层计算误差（输出层是有期望输出的，所以很容易计算误差）

使用误差来更新输出层权重

后向过程：

由输出层权重的更新前向传递得到隐含层的误差（隐含层没有期望输出，无法直接得到权重改变量，但是隐含层受输出层权重影响，可以扩散过来）

隐含层权重更新

**神经网络中的一些实用性经验**(无严格的理论证明，但在实际开发中已被认可)：

1. **激活函数**

激活函数需要满足的特征：

* 多个激活函数中必须存在非线性的函数，否则三层网络将不提供高于二层网络的计算能力
* 饱和性，即存在最大输出值和最小输出值，限定权值和激活函数的上下边界，使得有限训练次数内收敛。
* 连续性和光滑性，即和在整个变化范围内都有定义。该规则对反向传播学习是非常重要的。

S型函数是光滑、可微、非线性且饱和的，满足以上所有条件。

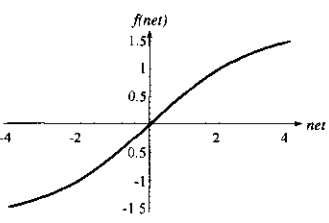
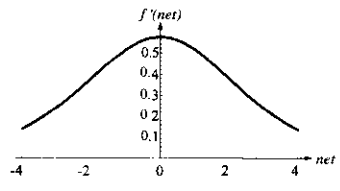
sigmoid函数是使用最广泛的激活函数，大多数情况下主要采用sigmoid函数。原因如下：

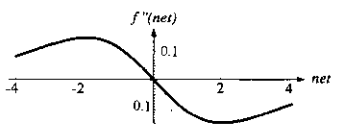
* sigmoid函数是一种S型函数。
* 对多项式分类器，采用梯度下降法训练会使得过程变得很敏感（输入的微小变化使得输出产生很大变化）。sigmoid可以避免这个问题。
* sigmoid可以提供输入层的分布式或全局性的表示。

1. **sigmoid函数的参数**

上式是sigmoid函数的一般表示形式。

调整上式中的a和b以达到特定的参数集之间的关系，c的作用是使图像上下偏移。例如a=1.716，b=2/3可以保证，二阶导数的极值大致在。如下图所示。



1. **输入信号规格化**

对输入数据先进行尺度变换，即数据归一化。一种最常用的归一化方式

1. **带噪声的训练方法**

当训练集很小时，可以构造一个虚拟的或者替代的模式来使用。可以加入d维高斯噪声使数据更具一般性。注意：**附加噪声的方差应该小于1，并且类别标记保持不变。**

1. **人造数据**

如果我们提前知道数据模式畸变的特征，可以人工制造数据来训练模型，使得模型可以包含更多的信息，在后续的模型使用中使模型应用更广泛。例如，文字识别中，文字可能倾斜或者旋转，可以在训练集中加入倾斜和旋转后的数据。

1. **隐含层神经元个数**

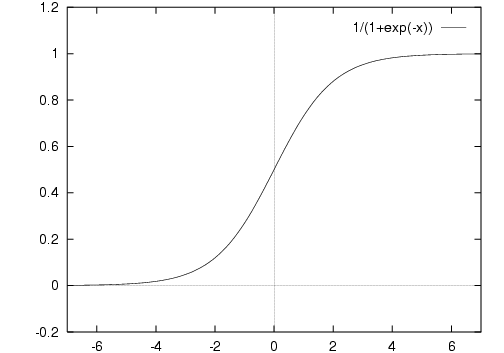
隐含层神经元个数决定了神经网络的表达能力，也决定了判定函数的复杂程度。如果模式容易分开，只需要较少的神经元，而如果模式从较高的复杂的函数中抽取，就需要更多的神经元。没有更多信息的情况下，没有简单的方式可以在训练之前设置隐含层神经元数目。

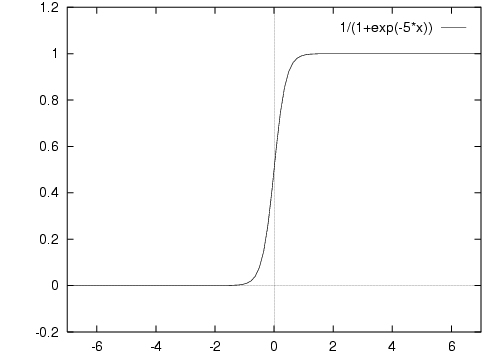
隐含层神经元个数决定了神经网络中的权值参数个数，显然，权值数不应该比训练样本的个数多。有一个简单的经验，**选取总的权值数为训练样本个数的1/10**。还有另一种方法，**根据训练数据调节网络的复杂程度**，先从大的网络、多的权值数开始，逐渐减少并保持网络的判别能力大致不变。

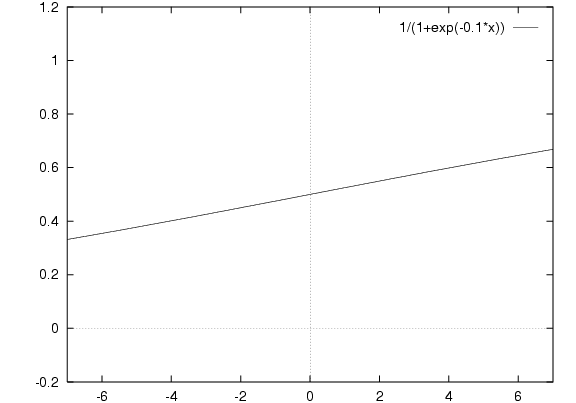
1. **权值初始化**

不能开始将权值初始化为0，否则学习过程不能开始。

我们期望设置权值使模型能够快速、均衡的学习。通常是先随机生成的，但是这个任意给定是有范围的。 ，因此，要选择若 太小，只有线性模型被实现；若 太大，可能在学习前就达到饱和。看下图



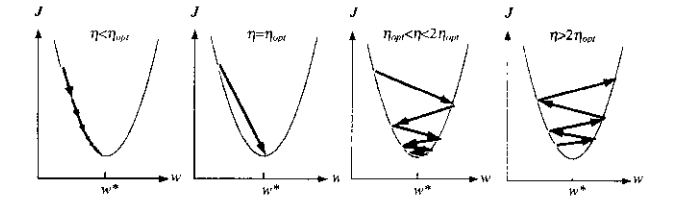




sigmoid函数的输入范围为因此我们需要设置来满足一个神经元上的激励在输入层到隐含层：如果一个神经元可以接收d个输入，应该选取隐含层到输出层：如果一个神经元可以接受的神经元数为*n*，应该选取

1. **学习率**

原则上，只要学习率足够小，就可以保证收敛。学习率的值决定收敛的速度。但实际上，学习率过小会导致收敛速度慢，而学习率过大则会产生振荡，甚至不会收敛。看下图



在实践中发现，sigmoid型网络中，可以首先将学习率设置为0.1，然后根据实际情况做相应调整即可。如果学习过程发散，则将学习率调小；如果学习速度太慢，将学习率调大。

最理想的方式当然是一步达到最优，如上左起第二张图所示，二次函数的最优学习率是可以计算出来的，即二次函数可以在理论上做到一步到达。

1. **权值衰减**

一种简化网络以及避免过拟合的方法是权值应当比较小。权值衰减最简单的方式是其中衰减过程中

1. **隐含层数**

一般的神经网络设计过程中，三层网络已经足够执行任意复杂的函数映射。除非特殊情况，不要使用多于三层的网络。

深度学习的网络都是多于三层的。那这样不是矛盾了么？传统的三层神经网络是全连接网络，而深度学习所用的卷积神经网络等都不是全连接的。神经网络层数不要太多是因为：1. 模型太复杂容易过拟合，2. 模型规模太大训练的时间和计算代价是不可接受的，3. 模型层数太深会产生梯度消失的问题，4.对于大多数问题，浅层网络足以解决。但是，实际中的问题如高分辨率图片的识别与分析，需要更加复杂的网络才能解决。

基于这些原因，现在的深度学习方法往往基于三个条件：高度复杂的模型、极其庞大的数据集、高性能GPU。数据量大了就不用担心过拟合的问题了，计算能力上去了就不用担心训练时间太久。这样就可以用更加复杂的模型来解决更加复杂的问题。

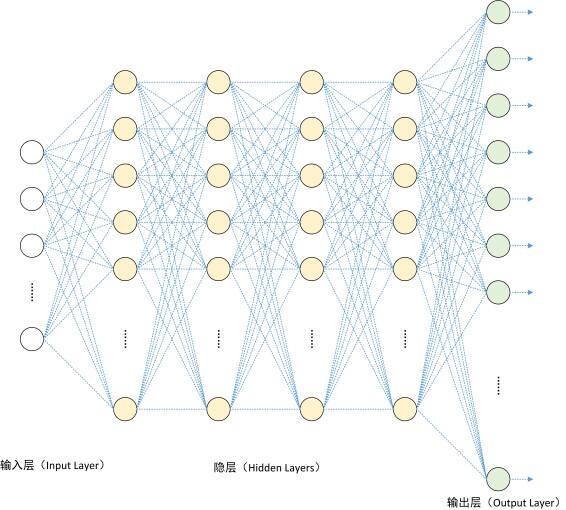
1. **交叉验证（K-CV）**

当数据量不足时，将原始数据分成K组（一般是均分），将每个子集数据分别做一次验证集，其余的K-1组子集数据作为训练集，这样会得到K个模型，用这K个模型最终的验证集的分类准确率的**平均数**作为此K-CV下分类器的性能指标。

K一般大于等于2，实际操作时一般从3开始取，只有在原始数据集合数据量小的时候才会尝试取2。K-CV可以有效的避免过学习以及欠学习状态的发生，最后得到的结果也比较具有说服性。

### 深度学习：

隐含层很多的神经网络。通过多层次的学习而得到对于原始数据的不同抽象层度的表示，进而提高分类和预测等任务的准确性。



深度学习本质上是一种表示学习，它跟传统的浅层神经网络的区别不只是在层数上。浅层神经网络需要人工提取特征再输入网络，深度学习把**特征提取**也归纳到学习内容中去，深度学习网络相当于一个黑盒，输入原始数据集，特征提取在训练过程中完成，然后输出结果。（这里需要解释一下什么是特征：特征就是用于描述事物的属性，最直观的就是数据库里边存的数据，比如学生，姓名、学号、年龄、成绩等这些就是特征。而对于一张图片，其实就是一个m x n x 3的矩阵，他的特征是什么？不明显，或者说不知道，这时候就需要深度学习来解决）

深度学习对于图像、语音、文本这种**特征不明显**（需要手工设计且很多没有直观的物理含义）的问题，深度模型能够在大规模训练数据上取得更好的效果。深度学习的一个主要优势在于可以利用海量训练数据（即大数据）。

**深度学习可以做什么？**

1. 语音识别

新模型基于神经网络而不是隐马尔可夫模型，把识别错误率降低了30%

1. 语义分割
2. 图像识别、图像分类、图片搜索

1000种不同分类，每一类有1000张图片，将图片分类，在神经网络使用之前，最好的错误记录是26%；使用神经网络后错误率下降到3.46%；人类的错误率也有5.1%。

街景图片中的文字识别。首先要找到文字部分，模型能够去有效预测像素中热点图，那些含有文字的像素点。训练的数据就是包含文字划分的多边形。要考虑大小字体，远近，不同颜色。训练的模型相对容易，就是卷积神经网络尝试去预测每个像素是否包括文字。



1. 图像着色



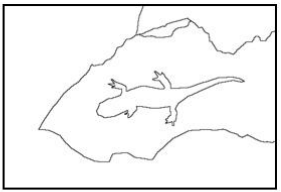
1. 深度信息预测



1. 图像恢复（从灰尘、雨水污染或者压缩过的图像）



1. 物体边缘检测

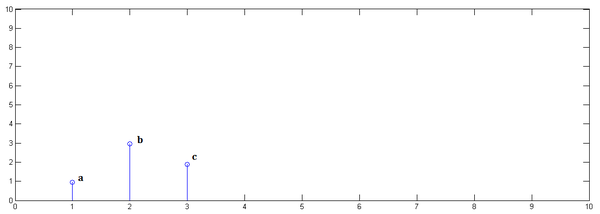
### CNN：

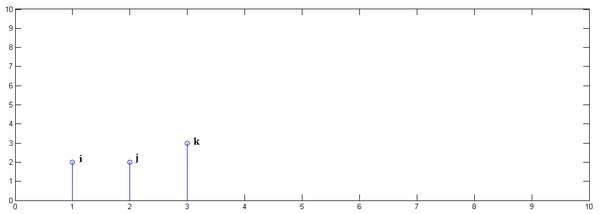
Convolutional Neural Network，卷积神经网络

卷积：

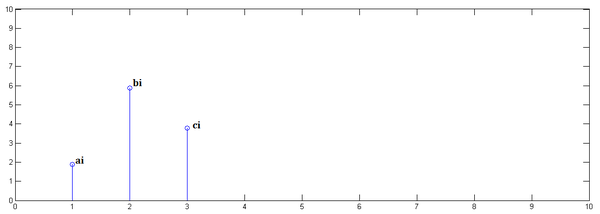
知乎上找了一个**卷积**的理解方式。

已知x[n] = {a, b, c}, y[n] = {i, j, k}如下图

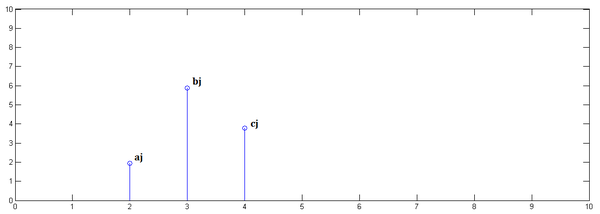




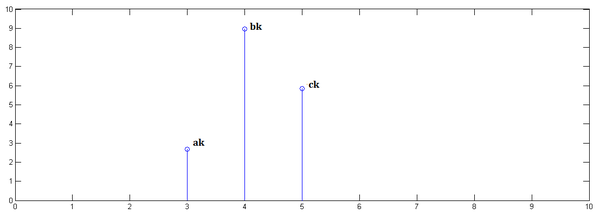
首先，x[n] 乘以y[1]并平移到位置1：



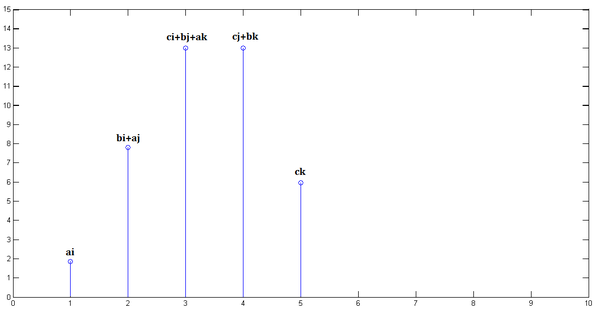
然后，x[n] 乘以y[2]并平移到位置2:



之后，x[n] 乘以y[3]并平移到位置3:



最后，把上面三个图叠加，就得到了x[n] \* y[n]：



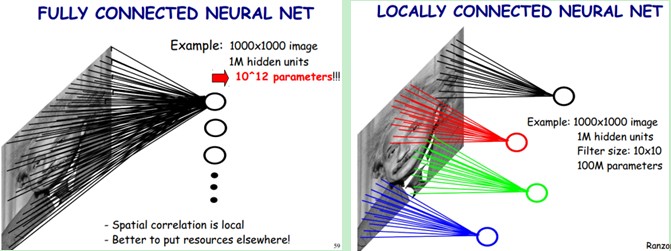
四个字概括：平移、叠加

卷积：一个函数(响应，y)在另一个函数（输入信号，x）的加权叠加。

**在输入信号的每个位置，叠加一个单位响应，就得到了输出信号。**

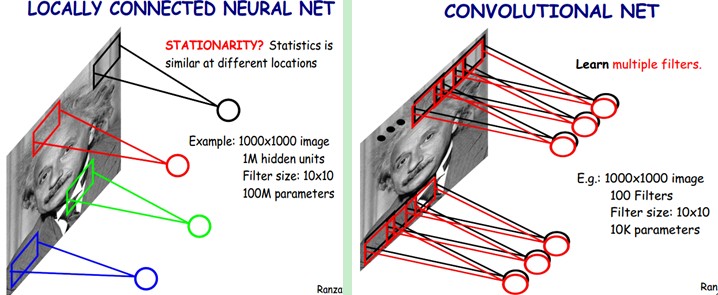
卷积神经网络：一个多层的神经网络；核心思想：局部感知、权值共享

局部感知：假设有1000x1000图像，有1百万个隐层神经元。如果全连接，会有1000x1000x1000000=10^12个连接，也就是10^12个权值参数。计算量太大。其实不需要每个神经元连接所有像素点，只需要局部连接，然后在高层将不同的局部信息综合起来就可以得到全局信息。假设将图像划分成10x10的块，每个神经元连接一个块，则会形成10x10x1000000=10^8个连接。由此可见，局部连接计算量会降低很多。

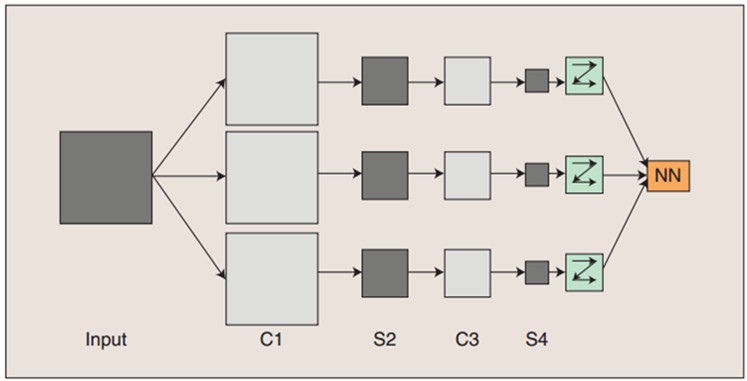


权值共享：隐含层的每一个神经元都连接10x10个图像区域，也就是说每一个神经元存在10x10=100个连接权值参数。假设每个神经元用的是同一个卷积核去卷积图像，也就是说每个神经元用的一样的100个权值参数，那么不管有多少个神经元，都只要100个参数就够了。

一个卷积核就是一种特征，如果想要提取多种特征，只需使用多种卷积核。所以假设我们加到100种滤波器，每种滤波器的参数不一样，表示它提出输入图像的不同特征，例如不同的边缘。这样每种滤波器去卷积图像就得到对图像的不同特征的放映，我们称之为Feature Map。100种卷积核x每种卷积核共享100个参数=100x100=1万个参数。

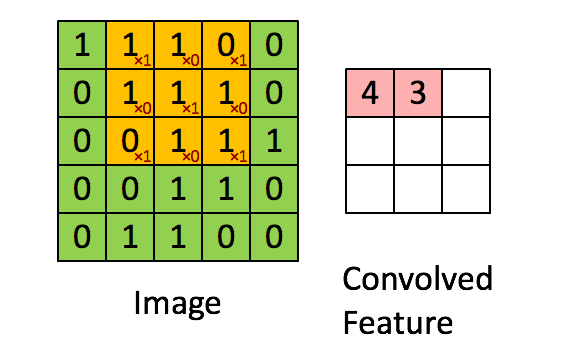
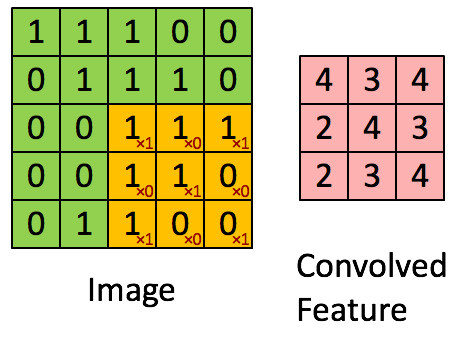


滤波过程：卷积后在C1层产生三个特征映射图，然后特征映射图中每组的四个像素再进行求和，加权值，加偏置，通过一个Sigmoid函数得到三个S2层的特征映射图。这些映射图再进过滤波得到C3层。这个层级结构再和S2一样产生S4。最终，这些像素值被光栅化，并连接成一个向量输入到传统的神经网络，得到输出。

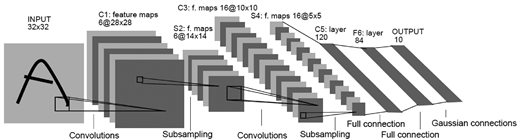


卷积过程：

****

**……**

卷积神经网络图：



信息逐层丢失。降低了网络模型的复杂度，减少了权值的数量。

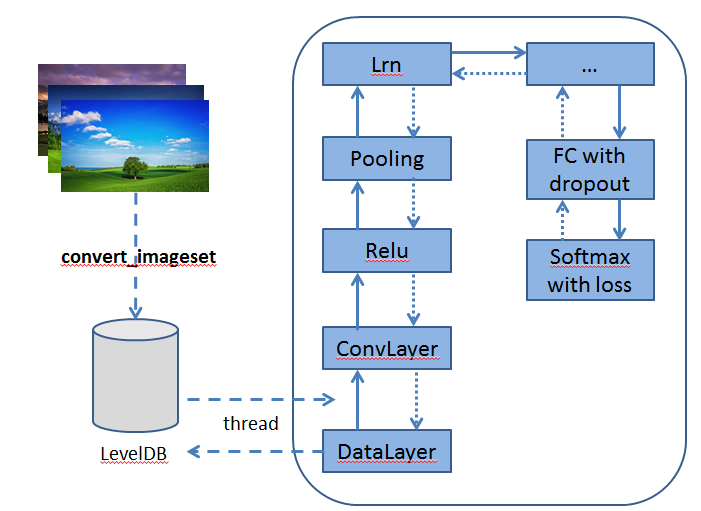
Caffe：Convolution Architecture for Feature Embedding (Extraction)

特征嵌入（提取）卷积架构

1. Caffe 是什么？

* CNN工具箱
* C++ 语言架构
* CPU 和GPU 无缝交换
* Python 和matlab的封装

1. Caffe 架构



1. Caffe原理

pocket

instapaper