**CNN**

1. **概念及特点**

CNN-卷积神经网络。特点：局部感受、参数共享（这两种技术可以降低模型复杂度，减少参数数量）、多核卷积（可以提取多种特征）。具有局部不变性：平移、倾斜、比例缩放等。

<http://blog.csdn.net/u013007900/article/details/51381084>

1. **模块**
2. **卷积层**
3. 卷积运算

卷积：

互相关函数：

许多机器学习库中使用互相关函数但叫卷积（《Deep Learning1,P279》）

1. 参数与输出特征图尺寸

o=⌊（i+2p−k）/ s⌋+1 (o-输出，i-输入，p-padding, k-核，s-stride)

* SAME padding:

out = ceil(float(in) / float(strides))

* VALID padding:

out = ceil(float(in - filter + 1) / float(strides))

1. 参数与感受野

感受野：卷积神经网络每一层特征图上的像素点在原始图像上映射的区域大小。

感受野计算时有下面的几个情况需要说明：

* 第一层卷积层的特征图像素的感受野的大小等于滤波器的大小
* 深层卷积层的感受野大小和它之前所有层的滤波器大小和步长有关系（padding不影响感受野，stride只影响下一层featuremap的感受野，ksize影响的是该层的感受野）

这里的每一个卷积层还有一个strides的概念，这个strides是之前所有层stride的乘积。

　　即**strides（i） = stride(1) \* stride(2) \* ...\* stride(i-1)**

关于感受野大小的计算采用top to down的方式， 即先计算最深层在前一层上的感受野，然后逐渐传递到第一层，使用的公式可以表示如下：

**RF = 1 #待计算的feature map上的感受野大小**  
**for layer in （top layer To down layer）:**  
**RF = ((RF -1)\* stride) + fsize**

stride 表示卷积的步长； fsize表示卷积层滤波器的大小　。

Alexnet网络pool5输出的特征图上的像素在输入图像上有很大的感受野（have very large receptive fields (195 × 195 pixels)）和步长（strides (32×32 pixels) ）。

(pool5->conv5->conv4->conv3->pool2->conv2->pool1->conv1：3->5->7->9->19->23->47->195)

<http://www.cnblogs.com/objectDetect/p/5947169.html>

1. 其他卷积层

反卷积（名称：Deconvolution/ Transposed Convolution/Fractional Strided Convolution）

应用场景：神经网络可视化、场景分割、生成模型等。

<https://buptldy.github.io/2016/10/29/2016-10-29-deconv/>

<https://www.zhihu.com/question/43609045?sort=created>

带孔卷积（dilation conv a.k.a atrous con，在Segmentation类问题中经常用到的）：

1. **激活层**

激活函数通常有如下一些**性质**：

**非线性**： 如果激活函数是恒等激活函数的时候（即f(x)=x），就不满足这个性质了，而且如果MLP使用的是恒等激活函数，那么其实整个网络跟单层神经网络是等价的。

**可微性**： 当优化方法是基于梯度的时候，这个性质是必须的。

**单调性**： 当激活函数是单调的时候，单层网络能够保证是凸函数。

**f(x)≈x：** 满足这个性质的时候，如果参数的初始化是random的很小的值，那么神经网络的训练将会很高效；如果不满足这个性质，那么就需要很用心的去设置初始值。

**输出值的范围**： 激活函数输出有限时，梯度的优化方法会更稳定，因为特征表示受有限权值的影响更显著；当激活函数输出无限时，模型训练会更高效，一般需更小的learning rate.

1. Sigmiod：

**优点**:输出映射在(0,1)之间,单调连续,输出范围有限,优化稳定,可用作输出层; 求导容易。

**缺点**：由于软饱和性，易梯度消失; 输出不以0为中心。

1. Tanh：

**优点**：比Sigmoid函数收敛速度更快；相比Sigmoid函数，其输出以0为中心。

**缺点：**未改变Sigmoid函数的最大问题——由于饱和性产生的梯度消失。

1. ReLU

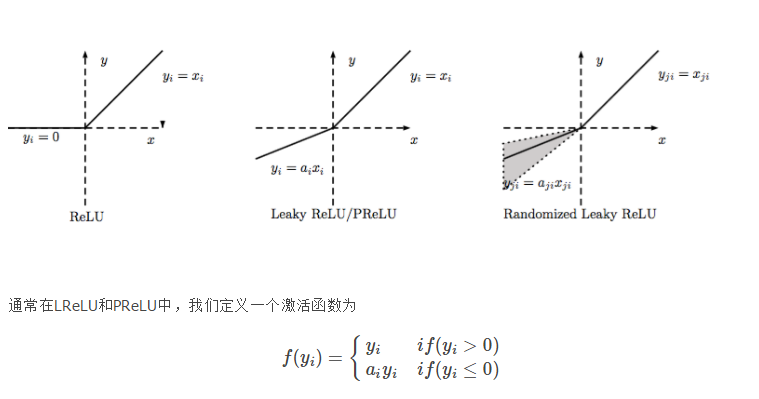
**优点**：

1.比起Sigmoid和tanh，ReLU在SGD中能快速收敛，因为线性、非饱和的形式。  
2.Sigmoid和tanh涉及很多很expensive的操作（比如指数），ReLU可以更加简单的实现。  
3.有效缓解了梯度消失的问题。  
4.在没有无监督预训练的时候也能有较好的表现。

5.提供了神经网络的稀疏表达能力。

**缺点：**  
随着训练的进行，可能出现神经元死亡，权重无法更新的情况。如果发生这种情况，那么流经神经元的梯度从这一点开始将永远是0。ReLU神经元在训练中不可逆地死亡了。

1. Prelu/LReLU/RReLU



**-LReLU**  
ai较小且固定时，为LReLU。最初为了避免梯度消失。但发现LReLU对准确率并没有太大的影响。应用LReLU时，要小心地重复训练，选出合适ai，LReLU的表现结果才比ReLU好。因此提出自适应地从数据中学习参数的PReLU。

**-PReLU**

可以自适应地从数据中学习参数。具有收敛速度快、错误率低的特点。

[**http://www.cnblogs.com/rgvb178/p/6055213.html**](http://www.cnblogs.com/rgvb178/p/6055213.html)

1. **maxout:**

如果激发函数采用sigmoid函数，隐含层节点的输出表达式为：

http://images.cnitblog.com/blog/381513/201311/18100501-844f6fbd0d254eef9b8169d81300a034.png

如果是maxout激发函数，则其隐含层节点的输出表达式为：

http://images.cnitblog.com/blog/381513/201311/18100510-a14ba98a40a543c099f00d1aa104d2ff.pnghttp://images.cnitblog.com/blog/381513/201311/18100518-7c567fbd32f64745979203a4785e1ab3.png

所以，Maxout 具有 ReLU 的优点（如：计算简单，不会 饱和），同时又没有 ReLU 的一些缺点 （如：容易 [**Go**](http://lib.csdn.net/base/go) die）。还是有一些缺点：就是把参数double了。

1. **池化层**

作用：下采样，减少网络参数，防止过拟合

方法：平均池化mean pooling和最大池化max pooling

操作：

* 一般池化：kernel\_size=stride，区域不重叠
* 重叠池化：kernel\_size > stride。（ALexNet中，作者使用了重叠池化，其他的设置都不变的情况下， top-1和top-5 的错误率分别减少了0.4% 和0.3%。）
* 空间金字塔池化：不懂，不整理，网址如下。

<http://blog.csdn.net/mao_kun/article/details/50507376>

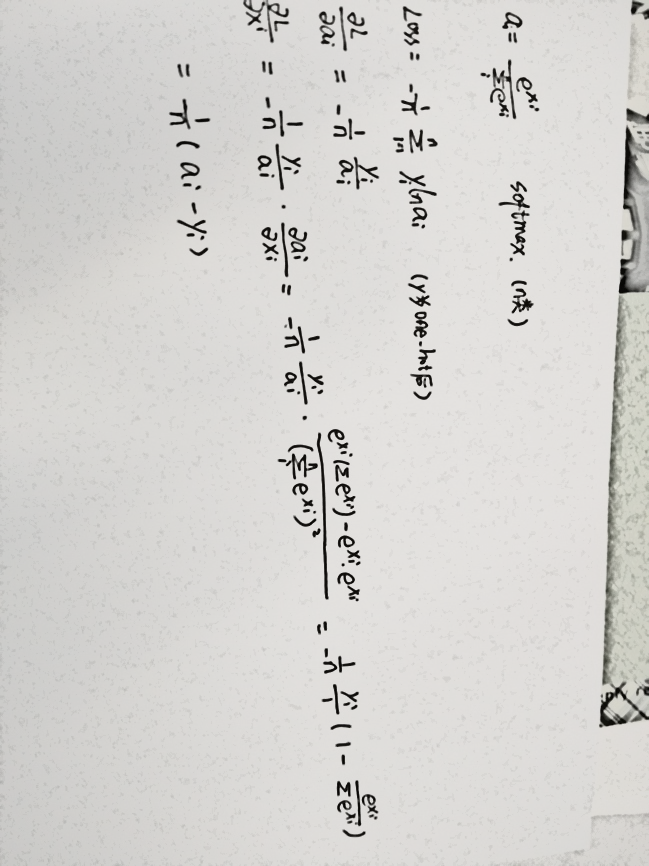
**4、损失函数**

1）均方误差MSE

 y为期望输出，a为神经元输出

2）交叉熵损失函数（cross-entropy）



xi为最后一层的输出。不一定对

3）K-L散度

4）其他机器学习常见损失函数

<http://blog.csdn.net/google19890102/article/details/50522945>

1. **常用网络模型及特点**
2. LeNet

三个卷积层，两池化层，激活函数为sigmoid，两全连层

1. AlexNet

五卷积层、三池化层、三全连层、dropout(防止过拟合)，激活函数ReLU替换sigmiod、重叠池化

1. **GoogleNet**

**Inception**原因：

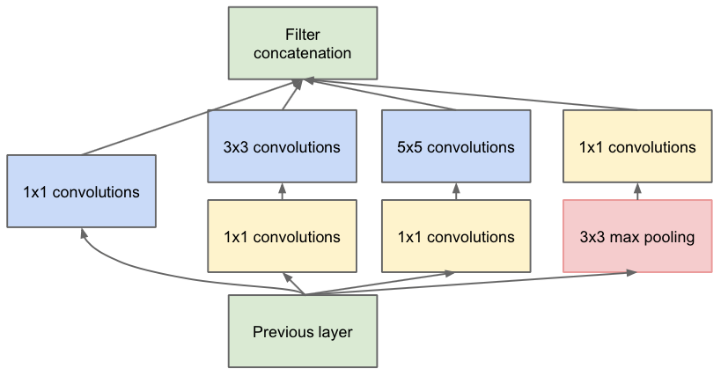
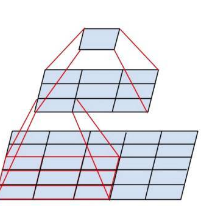
增大网络的缺点——过拟合和计算量大。解决方法是增加网络深度和宽度的同时减少参数。为减少参数，全连接就需变成稀疏连接，但在实现上，全连接变稀疏后实际计算量并不会有质的提升，因为大部分硬件是针对密集矩阵计算优化的，稀疏矩阵虽然数据量少，但是所耗的时间却是很难缺少。

所以需要一种方法，既能达到稀疏的减少参数的效果，又能利用硬件中密集矩阵优化的东风。Inception就是在这样的情况下应运而生。

<http://blog.csdn.net/stdcoutzyx/article/details/51052847>

具体**Inception v1-v4:**

Inception v1： 1x1，3x3，5x5的conv和3x3的pooling堆叠。可增加网络宽度和对尺度适应性。5×5的卷积核所需计算量太大，造成特征图厚度大。为避免这一现象，在3x3前，5x5前，max pooling后分别加上了1x1的卷积核起到了降低特征图厚度的作用

**V1 V2**

Inception v2：加入Batch Normalization层；5\*5卷积核由两个3\*3卷积核代替。

v3：分解， 7x7分解成1x7,7x1，3x3分解成（1x3,3x1）。可加速计算，使网络加深，增加网络非线性。非对称的卷积结构拆分，比对称拆为相同的小卷积核效果更明显，可增加特征多样性。在 **Inception Module**中使用分支，还在分支中使用了分支。

v4：比v3，结合了残差网（ResNet）。



<http://blog.csdn.net/u014114990/article/details/52583912>

<http://blog.csdn.net/diamonjoy_zone/article/details/70576775>

1. VGG
2. ResNet
3. **参数初始化**
4. **偏置初始化**

**偏置（biases）初始化。**通常将偏置初始化为0，因为随机小数值权重矩阵已经打破了对称性。对于ReLU，有研究人员喜欢用如0.01这样的小数值常量作为所有偏置初始值，认为这样能让所有的ReLU单元一开始就激活，能保存并传播梯度。然而，这样做是不是总能提高算法性能并不清楚（有时实验结果反而性能更差），所以通常还是使用0来初始化偏置参数。

1. **权重初始化**

* **高斯分布初始化：**直接对参数进行高斯分布初始化，通常均值为0，标准差为0.01。

### Xavier 初始化：方差为下图左所示的均匀分布，nodein, nodeout为输入输出节点个数。

### 或

### （原理是为了保证输入输出数据方差一致）

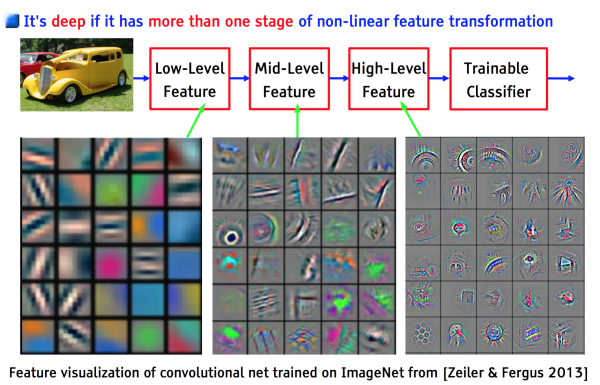
### MSRA 权重初始化：Var（W）=4./(nodein+nodeout)（Xavier 推导时假设激活函数线性，显然常用的ReLU 并不满足。）

**补充：Batch Normalization**是在进行非线性激活函数前，对输入值做 normalize

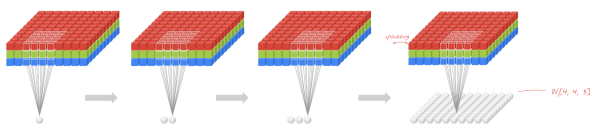
<http://zozoz.github.io/2017/03/18/%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C%E6%9D%83%E9%87%8D%E5%88%9D%E5%A7%8B%E5%8C%96/>

1. **问题及答案**
2. **CNN最成功的应用是在CV，那为什么NLP和Speech的很多问题也可以用CNN解出来？为什么AlphaGo里也用了CNN？这几个不相关的问题的相似性在哪里？CNN通过什么手段抓住了这个共性？**

相关性在于，存在**局部与整体的关系**，由低层次的特征经过组合组成高层次的特征，且得到不同特征之间的空间相关性。如下图：低层次的直线／曲线等特征，组合成为不同的形状，最后得到汽车的表示。



**CNN抓住此共性的手段主要有四个：局部连接／权值共享／池化操作／多层次结构。**

局部连接使网络可以提取数据的局部特征；权值共享大大降低了网络的训练难度，一个Filter只提取一个特征，在整个图片（或者语音／文本） 中进行卷积；池化操作与多层次结构一起，实现了数据的降维，将低层次的局部特征组合成为较高层次的特征，从而对整个图片进行表示。如下图：

上图中，**如果每一个点的处理使用相同的Filter，则为全卷积，如果使用不同的Filter，则为Local-Conv。**

1. **当在卷积神经网络中添加池化层(pooling layer)时，变换的不变性保持稳定，这样的理解是否正确？**

**是。**pooling可以保持某种局部不变性（旋转、平移、伸缩等）

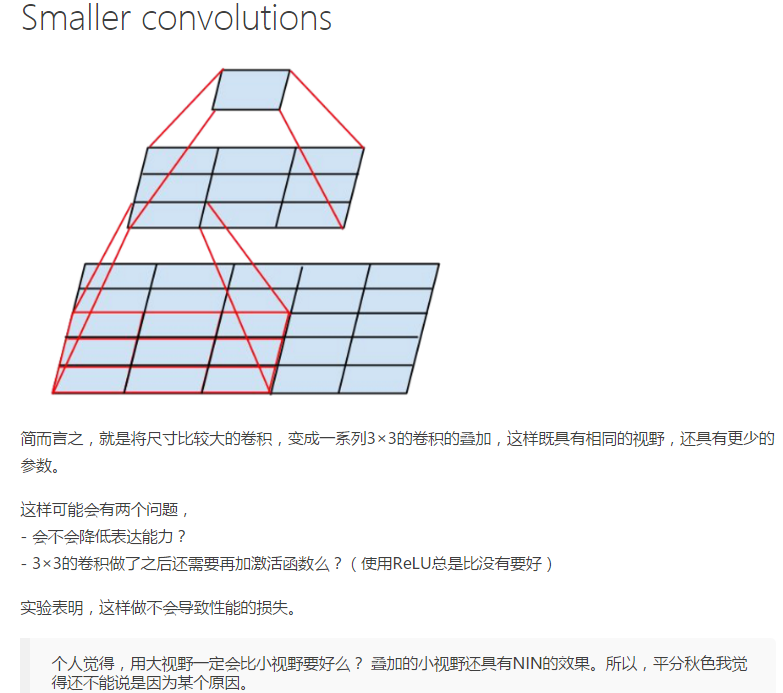
1. **解释什么是CNN**

见上

1. **增大卷积核的大小是否会提高卷积神经网络的性能？**

从googleNet的inception模块看，应该是不会提高性能。（没找到）

卷积核与视野有关，在保证相同视野的前提下，用几个小核代替大核可以减少参数。



1. **为什么很多做人脸的Paper会最后加入一个Local Connected Conv？**

DeepFace 先进行了两次全卷积＋一次池化，提取了低层次的边缘／纹理等特征。

后接了3个Local-Conv层，这里是用Local-Conv的原因是，人脸在不同的区域存在不同的特征（眼睛／鼻子／嘴的分布位置相对固定），当不存在全局的局部特征分布时，Local-Conv更适合特征的提取。

**Local connected conv局部参数不共享。**

1. **如何pre-train model呀?**

没看懂题意。

解决问题时，不用从零开始训练一个新模型，可以从在类似问题中训练过的模型(预训练模型)入手。然后fine-tuning。

1. **relu是什么，梯度消失如何解决？**

非饱和非线性激活函数。优缺点见上。求导简单，且右端求导不会饱和，梯度始终为1。

1. **inception v1-v4有什么特点**

见上

1. **面detection跟我讲receptive field很重要， 但我连dilation conv都不知道。Transposed Convolution, Fractionally Strided Convolution，sub-pixel conv，dilation conv 这几个讲清楚，这些名字的含义是怎么来的，说明白也是一个号的面试题。**

见上

1. **什麽样的资料集不适合用深度学习?**

数据集太小，数据样本不足时，深度学习相对其它机器学习算法，没有明显优势。

数据集没有局部相关特性，目前深度学习表现比较好的领域主要是图像／语音／自然语言处理等领域，这些领域的一个共性是局部相关性。图像中像素组成物体，语音信号中音位组合成单词，文本数据中单词组合成句子，这些特征元素的组合一旦被打乱，表示的含义同时也被改变。对于没有这样的局部相关性的数据集，不适于使用深度学习算法进行处理。举个例子：预测一个人的健康状况，相关的参数会有年龄、职业、收入、家庭状况等各种元素，将这些元素打乱，并不会影响相关的结果。

1. **Weights Initialization. 不同的方式，造成的后果。为什么会造成这样的结果。?**
2. **Activation Function. 选用什么，有什么好处，为什么会有这样的好处。**

ReLU等，解释见上。

1. **为什么网络够深(Neurons 足够多)的时候，总是可以避开较差Local Optima？**

[The Loss Surfaces of Multilayer Networks](https://zhuanlan.zhihu.com/)

1. **Loss. 有哪些定义方式（基于什么？）， 有哪些优化方式，怎么优化，各自的好处，以及解释。**

Cross-Entropy / MSE / K-L散度