CNN

CNN,卷积神经网络，大量用于计算机视觉|图像处理，在图像识别、目标检测、图像分割等领域的运用可以取到传统计算机视觉难以达到的精度和效果。

一、 引入

全连接网络当输入层特征维度较高时，由于中间层神经元全连接，需要训练的参数（权值、偏移等）较多，需要进行大量运算，网络训练速度很慢。因此，引入 CNN，卷积层的神经元与前层仅部分连接|非全连接，且同一层中某些神经元的连接权重和偏移共享（即相同），大量减少需要训练的参数~~~~神奇~~~

CNN 包含的层有：

输入层：数据输入；

卷积层：用卷积核进行特征提取和映射；

激励层：非线性映射（卷积是线性运算，只有非线性化了，才能表示更复杂的函数，网络效果才能更好）；

池化层：下采样，特征稀疏，减少数据运算量；

全连接：CNN 尾部，重新拟合，减少特征信息损失；

输出层：输出结果。

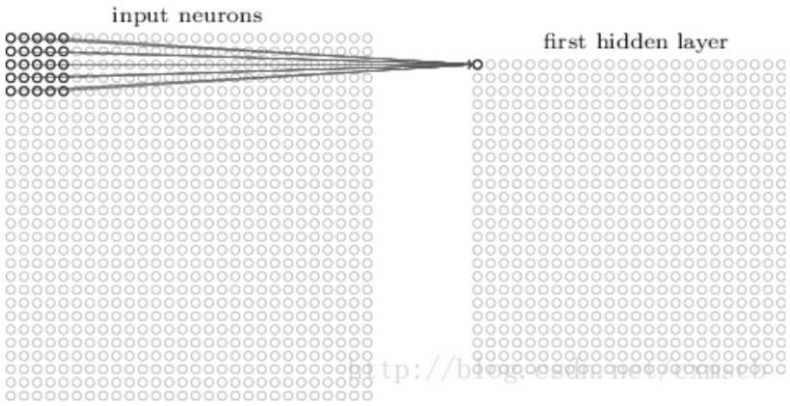
其它层：归一化层（特征归一化）、切分层（分区域单独学习）、融合层（将分区域单独学习的分枝进行融合）；

输入层

CNN 与全连接的输入层（输入格式为：一维向量）不同，其输入保留了图片本身的结构。

卷积层

几个重要概念：local receptive fields（感受视野）、shared weights（共享权值）；



上图那一小块儿就是感受视野，全连接的感受视野很大，覆盖整个输入层，而 CNN 就是一小块儿。每条线上参数就是权值，构成的权值矩阵就是卷积核，一个卷积核从左到右扫描生成一个 feature map(特征映射图)，没错，这一个 map 对应一个卷积核，用的都是这同一个权值矩阵，这就是 shared weights（权值共享）。可以看到，原本那么多要训练的权值，大大减少，因为共享了。卷积核权值矩阵初值自己随机设定，往往加一个偏移，即经过卷积层就是做“卷积+偏移”运算。（这个很重要，因为我们所谓的模型训练，其实就是在优化卷积核的权值矩阵和偏移，优化到 loss 足够小，就是训练好了）

输入图片若是 RGB，就是 3 层，那么卷积核、感受视野就是 3 层。每层卷积核有 2 个，那么特征映射图就有 2 个。就是说，输入层深度决定卷积核、感受视野深度；卷积核个数，决定 feature map(特征映射图)个数。

激励层

对卷积层输出进行非线性映射，非线性化后才可表示更复杂的函数，网络效果更好。常用的激励函数是：ReLu 函数（全连接里的 softmax）。卷积层和激励层通常合称为“卷积层”。

池化层

对卷积层输出进行特征稀疏、降维、减少运算量。

常用方法：max pooling(最大池化)和 average pooling（平均池化）。类似的，可以称为池化视野，分别取其中的最大、平均值作为视野的特征值，从左到右，然后向下移动步长大小，继续从左到右。

归一化层（归一化，防止“梯度弥散”，加速训练）

1、Batch Normalization(批量归一化)（样本间归一化）

在网络中间进行预处理，对网络上一层输出进行归一化，然后再输入到下一层，可以有效防止“梯度弥散”，加速网络训练。

具体就是每次训练取 batch_size 大小的样本，BN 层位于卷积层和激励层之间，对数据进行求均值、方差，然后归一化再输出至激励层，可达到调整激励函数偏导的效果，防止“梯度弥散”（反向传播时梯度过小，导致网络前几层的参数|权值和偏移不能有效学习、更新、优化）（batch normalization 的训练和测试有所区别，因为训练时一次输入一堆数据，有均值和方差。但是测试时一次输入一个数据，其均值方差为 0，要想单个数据测试的话，可以使用训练时计算的均值和方差的均值来做归一化，不然测试会出错）

2、Local Response Normalization（近邻归一化）

不同相邻卷积核间，即经过激励后的不同 feature map 间。

两者区别：BN 是在激励层前，同一 map 不同数据间作用；LRN 是在激励层后，不同 map 间作用。

切分层

有时对数据图片要进行多分辨率（多个大小的卷积核）训练学习，就要进行切分，单独对某一块儿进行力度更大的学习。

融合层

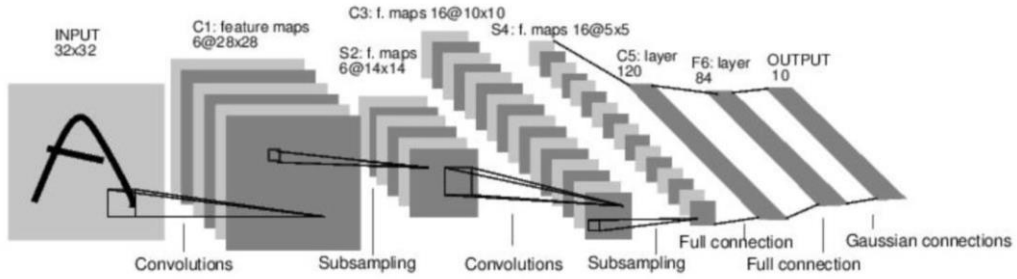
可以对切分层进行融合，也可以对不同卷积核提取的特征进行融合。如：GoogleNet 就是用多分辨率进行目标特征学习，然后通过 padding 使每一个 feature map 长宽一致，再将多个 map 在深度上拼接在一起。（融合的方法有几种，一种是特征矩阵之间的拼接级联，另一种是在特征矩阵上进行运算）

全连接和输出层

全连接是在 CNN 尾部进行重新拟合，减少特征信息的丢失；输出层准备目标结果输出。（最后全连接会把多维图像信息（此时需要的连接已大幅度减少）转换为一维，最后经输出层输出）

典型 CNN

LeNet-5（第一个成功用于 MNIST 的 CNN）

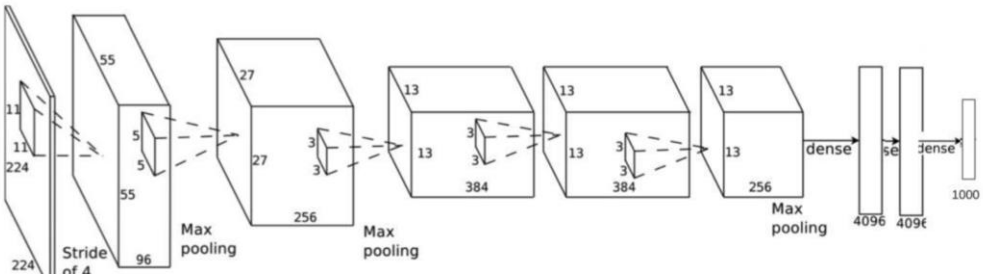


Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2

输入层—卷积层 1—池化层 1—卷积层 2—池化层 2—全连接层 1—全连接层 2（输出）

（卷积核都是 5*5，步长为 1；池化核（最大池化）都是 2*2，步长为 2）

AlexNet

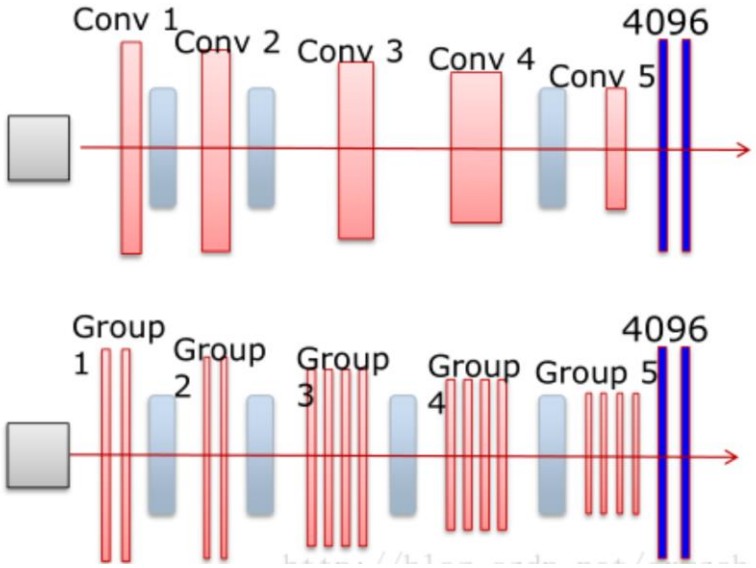


可以看出，经典 CNN 结构通常为：输入层➞（卷积层+➞池化层？）+➞全连接层+➞输出层

（卷积核有 11*11、5*5、3*3，池化核都是 3*3）

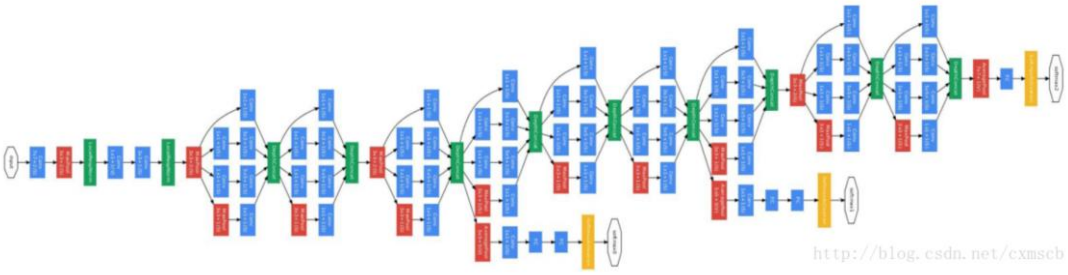
VGGNet

VGGNet（下）和 AlexNet（上）相差不大，只是在卷积层部分增加了一些层。

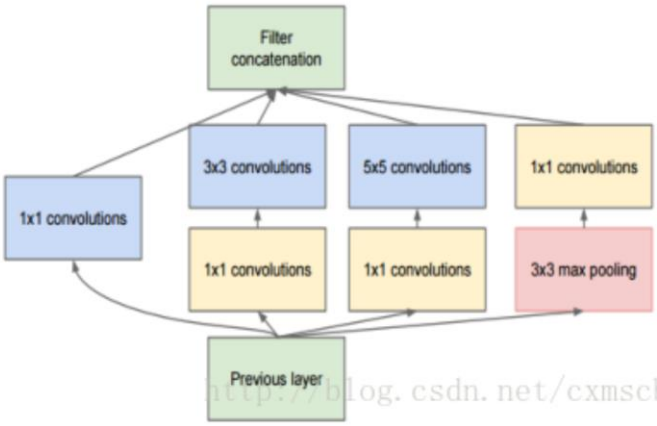


GoogleNet

使用多个不同分辨率的卷积核进行特征提取，最后再对其 feature map 按深度 padding 在一起，提高训练效率（和精度）



网络里用得最多的模块化结构（Inception module）

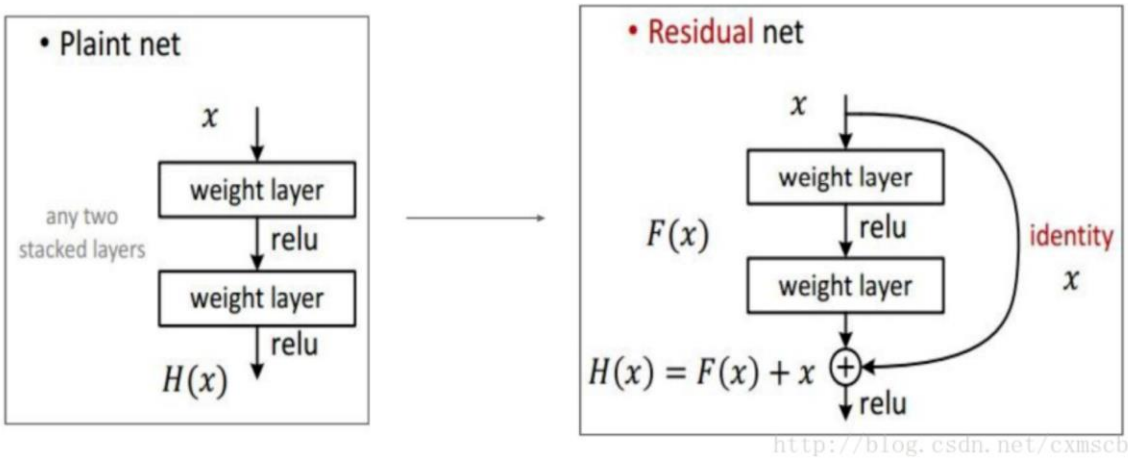


在 Inception module 中使用到了很多 1×1 的卷积核，使用 1×1 的卷积核，步长为 1 时，输入的 feature map 和输出的 feature map 长宽不会发生改变，但可以通过改变 1×1 的卷积核的数目，来达到减小 feature map 的厚度的效果，从而做到一些训练参数的减少。

GoogleNe 另一个特点就是它是全卷积结构（FCN）的，网络最后没有使用全连接层。一方面这样可以减少参数数目，不容易过拟合；一方面也带来了一些空间信息的丢失。代替全连接层的是全局平均池化（Global Average Pooling，GAP）的方法，思想是：为每一个类别输出一个 feature map ，再取每一个 feature map 上的平均值，作为最后的 softmax 层的输入。

ResNet

残差网络，这是一个很牛 X 的网络，和之前那些不一样（都是将输入一层一层传递下去），它将低层的学习特征和高层的学习特征进行融合（加法运算），这样反向传递就不会因为层次较深而难以训练，导数传递更快，减少“梯度弥散”。



http://blog.csdn.net/cxmscb

注：

1、 CNN 应用确实很广，一般的 CV 像人脸识别、行人|车辆检测等，采用 CNN（的经典网络）后精度会大幅度提升；
2、 这些网络都是经典网络，都是大神研究出来并经过大量测试的，直接拿来用就会收到很好的检测效果。具体咋用，比如用 tensorflow 就完全按照网络结构一层一层搭（一句一句写）就行。网络构建好后，输入格式处理过的数据，前向+反向传播（训练）—不断优化(梯度下降、跌迭代法等)网络参数（权值、偏移等）直到 loss(交叉熵、MSE 等)足够小，然后保存模型并调用模型进行前向传播（推理），输出处理结果，比如识别出人脸、行人、车辆等；
3、 选取典型网络（如 ResNet）、典型应用(如人脸识别)来复现其功能，理解源码，改进源码，迁移运用……（都是这套路）

关于 CNN，先就写这么多，以后耍熟了，再补充~~~