**<https://zhuanlan.zhihu.com/p/77471866>**

**【填充（Padding）】**

前面可以发现，输入图像与卷积核进行卷积后的结果中损失了部分值，输入图像的边缘被“修剪”掉了（边缘处只检测了部分像素点，丢失了图片边界处的众多信息）。这是因为边缘上的像素永远不会位于卷积核中心，而卷积核也没法扩展到边缘区域以外。

这个结果我们是不能接受的，有时我们还希望输入和输出的大小应该保持一致。为解决这个问题，可以在进行卷积操作前，对原矩阵进行边界**填充（Padding）**，也就是在矩阵的边界上填充一些值，以增加矩阵的大小，通常都用“0”来进行填充的。

**【步长(Stride)】**

滑动卷积核时，我们会先从输入的左上角开始，每次往左滑动一列或者往下滑动一行逐一计算输出，我们将每次滑动的行数和列数称为Stride。

**某一层滤波器的通道数 = 上一层特征图的通道数。**如上图所示，我们输入一张 6x6x3 的RGB图片，那么滤波器（ 3x3x3 ）也要有三个通道。

**某一层输出特征图的通道数 = 当前层滤波器的个数。**如上图所示，当只有一个filter时，输出特征图（ 4x4 ）的通道数为1；当有2个filter时，输出特征图（4x4x2 ）的通道数为2。

<https://zhuanlan.zhihu.com/p/29119239>

Text

Description automatically generated

\*以下计算演示均省略掉了 Bias ，严格来说其实每个卷积核都还有一个 Bias 参数。

Table

Description automatically generated

<https://blog.csdn.net/yjl9122/article/details/70198357>

Graphical user interface, text

Description automatically generated

<https://www.jianshu.com/p/cbe93f5152cb>

Table

Description automatically generated

CNN计算量

<https://blog.csdn.net/liuweiyuxiang/article/details/99637649>

Text

Description automatically generated

<https://fengziyjun.github.io/content/conv_questions.html>

Text

Description automatically generated

<https://zhuanlan.zhihu.com/p/38224147>

剪枝

剪枝是决策树类算法防止过拟合的方法。如果决策树的结构过于复杂，可能会导致过拟合问题，此时需要对树进行剪枝，消掉某些节点让它变得更简单。剪枝的关键问题是确定减掉哪些树节点以及减掉它们之后如何进行节点合并。决策树的剪枝算法可以分为两类，分别称为预剪枝和后剪枝。前者在树的训练过程中通过停止分裂对树的规模进行限制；后者先构造出一棵完整的树，然后通过某种规则消除掉部分节点，用叶子节点替代。

数据增广

数据增广是解决过拟合中思想比较朴素的方法。训练集越多，过拟合的概率越小，数据增广是一个比较方便有效屡试不爽的方法，但各类领域的增广方法都不同。

在计算机视觉领域中，增广的方式是对图像旋转，缩放，剪切，添加噪声等。

在自然语言处理领域中，可以做同义词替换扩充数据集。

语音识别中可以对样本数据添加随机的噪声。

Dropout

Dropout是神经网络中防止过拟合的方法。dropout的做法是在训练时随机的选择一部分神经元进行正向传播和反向传播，另外一些神经元的参数值保持不变，以减轻过拟合。dropout机制使得每个神经元在训练时只用了样本集中的部分样本，这相当于对样本集进行采样，即bagging的做法。最终得到的是多个神经网络的组合。

Early Stopping

提前停止的策略是在验证集误差出现增大之后，提前结束训练；而不是一直等待验证集 误差达到最小。提前停止策略十分简单,执行效率高，但需要额外的空间备份参数。

集成学习

集成学习算法也可以有效的减轻过拟合。Bagging通过平均多个模型的结果，来降低模型的方差。Boosting不仅能够减小偏差，还能减小方差。

<https://bitjoy.net/2019/03/24/neural-networks-and-deep-learning%EF%BC%88%E4%B8%89%C2%B7%E4%BA%8C%EF%BC%89%E8%BF%87%E6%8B%9F%E5%90%88%E4%B8%8E%E6%AD%A3%E5%88%99%E5%8C%96/>

过拟合的含义就是网络在训练集上性能很好，但是在验证集（或者测试集）上的性能较差，这说明网络在训练集上训练过头了

判断网络是否过拟合的方法就是观察网络在训练集和测试集上的accuracy和loss的变化曲线。对于accuracy，如果训练集的accuracy很高接近100%且收敛了，但测试集上的accuracy和训练集上的accuracy相差较大也收敛了（如下图收敛到82%左右），说明网络过拟合了。对于loss，如果训练集的loss一直在下降，但测试集的loss先下降后又上升，也说明网络过拟合了。这两种现象，虽然指标不同，但含义是一样的，即网络在训练集上的性能一直在提高甚至到完美水平，但在测试集上的性能提高到一定水平后不再变化甚至下降了。

应对过拟合最好的方法就是增加训练数据，如果能把所有可能的数据都收集到，对所有数据产生过拟合，那相当于对所有数据都能预测得很好，那问题本质上已经解决了。

但是，在实际应用场景中，不可能收集到所有数据，而且数据往往是严重不足的，此时，应对过拟合主要有三种方法：正则化、Dropout和数据增强，

正则化的思路就是修改损失函数，使损失函数考虑模型复杂度。考虑正则化的损失函数的通用公式如下： A picture containing text

Description automatically generated

其中C0为原始的没有正则化项的损失函数，比如MSE或者交叉熵损失等，Ω(w)表示正则化项，即用来惩罚模型复杂度的，λ表示正则化参数，用来平衡C0和Ω(w)的重要性。

正则化又分为L2正则和L1正则

正则化不仅能解决过拟合问题，还能使结果更加稳定。我们都知道，神经网络的参数是随机初始化的，很有可能不同的随机初始值收敛到的网络权重不一样，导致最终模型性能有差异。加入正则化后，对所有权重都有正则化约束，使得梯度下降在探索不同w的方向时，都能得到一定程度的更新，而不会说某个方向比较好就一直在那个方向探索。

那么正则化为什么能解决过拟合问题呢？从公式(1)可知，加入正则化之后，除了要优化网络对训练数据预测的准确度之外，还需要使得网络的权重w尽量小，权重越小则网络越简单，极端情况下如果权重等于0，则相当于少了一个参数，网络肯定更简单了。所以加入正则化之后，网络的复杂度降低了，网络的泛化能力就更强了，也就更不容易过拟合。

**Dropout**

Dropout的做法和L1正则、L2正则很不一样，它不对损失函数进行修改，而是对网络结构进行修改。Dropout会在每个mini-batch训练时，随机删掉网络中一半的神经元（并不是真正的删除，而是暂时把对应的w设置为0）。它的效果相当于每次训练只用了一半的神经元，那么这次mini-batch和下次mini-batch相当于训练了不同的网络，最后预测时恢复所有的神经元。这样得到的预测结果相当于多个神经网络进行了average或者voting，使得预测结果更加鲁棒。比如有5个模型，其中3个模型预测对了，2个模型预测错了，做一下voting的话，就能得到正确的结果，而如果只是一个模型的话，如果刚好预测错了，那就错了。

Dropout的另一个解释时，因为网络在训练时会随机删掉一半的节点，那么节点间的依赖关系就减弱了，迫使神经元依赖较少的信息也要得到比较好的预测结果，所以网络会更加鲁棒。Dropout在深层神经网络中，特别有用，能有效防止过拟合。不过也正因为此，dropout会增加训练收敛的时间，这是可以理解的

<https://cloud.tencent.com/developer/news/246466>

在图像问题上，大家最开始用的就是非常基础的数据增强——加噪音、裁剪、翻转和镜像。

**翻转和镜像**：把原图沿x轴或者y轴翻转来形成新的样本

比如，对于手写字体，“b”翻转之后就成了“d”，标签不一样了，就不能用这种方法生成新的样本；

<https://dorianzi.github.io/2019/05/01/Digit-Recognizer-By-CNN/>

对于手写体 可以试试旋转小度数，缩放，随机水平/上下移动

<https://www.pianshen.com/article/5314724977/>

Graphical user interface, text, application

Description automatically generated

A picture containing waterfall chart

Description automatically generated

<https://www.yuque.com/zy19970/vo3yim/vmfeh5?language=zh-cn>

Graphical user interface, text, application, email

Description automatically generated

<https://jichilen.github.io/html/2019/03/09/paper_reading/>

IoU是一个很常见的评估度量，然而我们在做检测的时候预测的往往不是直接的IoU，而是去预测bbox的坐标

然而IoU并不是那么好直接作为预测目标，对于匹配的anchor，IoU是可以计算的，但是对于与gt不相交的bbox，IoU为0，优化无法继续

<https://www.geek-share.com/detail/2789863120.html>

如果∣ A ∩ B ∣ = 0 |A \cap B|=0∣A∩B∣=0则IoU无法反映出A和B到底是比较靠近还是相距比较远。

IoU不能反映出到底两个框是如何相交的。

<https://blog.csdn.net/lmxt520/article/details/98473024>

计算：

<https://frank909.blog.csdn.net/article/details/91366128?utm_medium=distribute.pc_relevant_t0.none-task-blog-BlogCommendFromMachineLearnPai2-1.channel_param&depth_1-utm_source=distribute.pc_relevant_t0.none-task-blog-BlogCommendFromMachineLearnPai2-1.channel_param>

<https://blog.csdn.net/lmxt520/article/details/98473024?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-4.channel_param&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-4.channel_param>

Text

Description automatically generated

Chart, waterfall chart

Description automatically generated