P31-2021-CNN

在Image Classification中，我们将模型的类别进行one-hot编码，然后我们将图片丢到模型中进行分类输出一个one-hot向量长度一样的预测概率，向量长度表示类别的数量，然后用交叉熵计算损失。图片一般是3D-tensor，由RGB三个channel，其余两维是像素值，我们尝试将他们拉直想变成向量便于我们Network训练,使用全连接层但是发现会出现参数过多，易过拟合。为了减少参数的数量我们继续观察图片的特性，如果我们直接对图片观察局部信息，也许我们不需要全部图片的特征，而只需要图片的局部特征来进行分类预测，为此我们选择设置一个感受野(Receptive field)只管心在我们在这个局部区域里面的值，通过它来计算局部特征。感受野直接可以重叠，可以采用不同大小的卷积，或者考虑不同的channel。特别的设置卷积核大小(kernel size)，根据论文和经验中一般采用3\*3。每个channel都设置为相同的卷积核。还需要设置步长(stride)，一般需要边缘重合，采用步长扫描整张图片。不同图片的相同特征一定会在扫描整图时候找到这个特征，让不同位置的卷积实现参数共享(parameter sharing)可以得到相同的结果。全连接层->感受野->参数共享->卷积层，这个变化参数在变少，模型的复杂度在变小，当然弹性也在变低，虽然卷积层有大的模型偏差，但是适合于影像的处理。我们采用不同filter来抓取图片中特征。，类似就是图像处理中的滤波器，不同参数的卷积核是用来特定的特征。多层的卷积，可以认为前层卷积输出也是一张图片，到下一层卷积时，其实是上上一层9\*9的范围的特征。如果越深可以检测到大范围的特征。还存在池化(pooling)，将图片变小，例如MAX或者mean，不改变图片的channel，实践上一般是卷积后进行池化，减少计算量，但是如果计算资源够用，我们不需要进行pool。在网络中经过卷积和池化后最后还是需要全连接层拉直进行预测。

下围棋可以用CNN来解决问题，基于当前的棋盘上已经落子的位置来预测下一次的落子落在哪里更好。认为棋盘是一个19\*19的图片，然后channel设置为48，Alphago设置5\*5的filter，围棋和图片存在某种特性，因为在局部区域可能出现相同的摆阵，但是对于池化就不同了，如果对围棋使用池化，是否是对围棋结构的破坏，在论文中页表示没有使用pooling，表明CNN不一定使用池化层。

P32-2022-使用验证集，模型仍然是过拟合

采用前面的使用到的能分类数码宝贝和宝可梦的最好的线段长度H，通过三个不同的模型使用相同的数据来求解最佳的H，将三个H用在测试集上，然后拿loss最低的H来用于全部的数据，但是其实可以认在上测试H时也是一种训练，在p=，H够大时拟合了，但是不一定拟合实际的。

P33-2022 鱼和熊掌我全都要

根据前面的，我们如何使用少量的H获得小的loss？

复习前面的直线拟合实际曲线，认为是一堆线段和Relu增加非线性拟合得到的，一个隐层就可以是实现用来拟合的直线。模型越来越深，我们的loss越来越低，但是模型越大，就需要大资料。如果我们选择横向发展来代替深度，每增加一个神经元，参数是非线性增长，计算量几何增长。和深层相比，深层更高效而且参数少。参数少表示只需要少量资料而且不易overfitting。采用逻辑电路来表明深层和前层区别，对于宽度学习来说d个输入需要个们，例如4个输入采用深度学习只需要3个门，减少了参数但实现了相同的功能。还是复习前面的拟合函数例子，表明多隐层就是函数的嵌套，由设置多层，每层都有两个神经元，每个神经元都只有一个W，一个B，和一个RelU，我们发现每次的输出都是一个X->Y对应的V，在经过一层变为W，是一个通过K层实现了变化，但是宽度学习虽然只需要一层就需要个参数，参数过多就会过拟合。事实和想象相反，其实深度学习相比宽度学习不需要大量资料，只要超过二次深度学习表现优于宽度学习。

# P34-spatial transformer layer

CNN并不能真正做到scaling和rotation,人类容易分辨一张图片中处在不同地方的数字，但是对于机器来说两张是不同的图片，我们提出spatial transformer layer，来实现对于图片的旋转缩放，让机器可以识别提高正确率，是对于图片的相关操作。本质还是一个神经层和图像处理里面一样通过矩阵操作实现旋转。还是通过调整权重来完成将图片的操作，

这个层叫NN层，但是需要前层对应合适的权重，如果需要实现缩小放大，我们只需要乘以正定矩阵，大于1 实现放大，小于1缩小。但是对于旋转，我们需要用到左乘矩阵加上偏执，对于像素点来说，我们只需要6个参数即可，,如果我们输入都是整数，可以得到很好的解，不会有小数点的索引，但是我们进行计算不应该该对其四舍五入，这种情况是无法使用梯度下降来求解的，为此我们选择了插值法，对于求出来的小数实际上是在四个点的区间内,我们不单纯的参考它跟距离最近的那个点，而是四个点的数值都参考.这种情况下NN参数有些微的变化的时候Output也会有些微的变化，就可以利用梯度下降来优化求解了。