深度学习学习笔记

本笔记为Andrew和Udacity的深度学习课程所做，鉴于Andrew课程已有完整笔记，这里只做对笔者而言重要的笔记，可能对别的读者没有太大用处。 <https://github.com/Parker-Lyu/DeepL_Andrew_And_Uda>项目说明上有相关作业及笔记的链接。

本笔记的顺序按照Andrew的课程来做，Uda的课程作为项目补充。

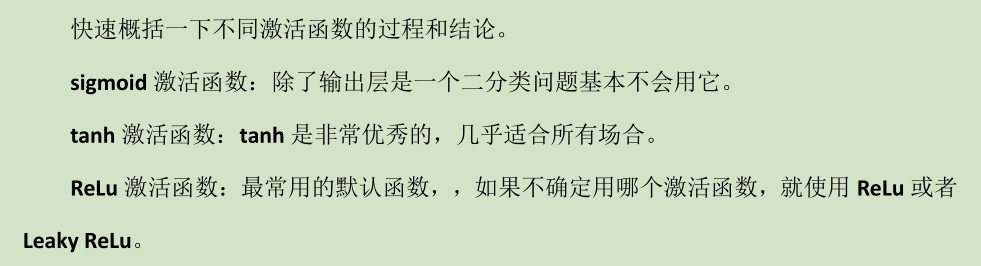
# 第一课 神经网络和深度学习

这部分主要讲了网络基础，关于梯度下降，自己写个网络，写出前向传播的公式，并且按照链式法则写出反向传播的公式，就可以用代码实现了。关于需要转置的问题，自己记录好矩阵纬度就ok了。

反向传播的代码实现见notebook。

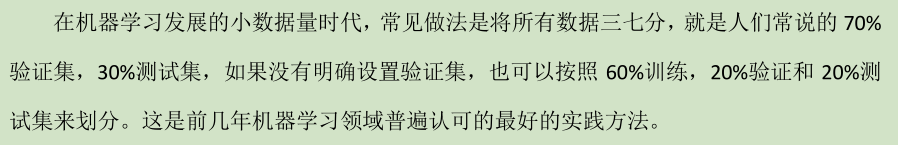
## 激活函数

为什么需要非线性激活函数？

如果没有非线性激活函数，深层神经网络可以看成线性函数的组合，计算一下整个网络就是一个单层的线性函数。

# 第二课 改善深层神经网络

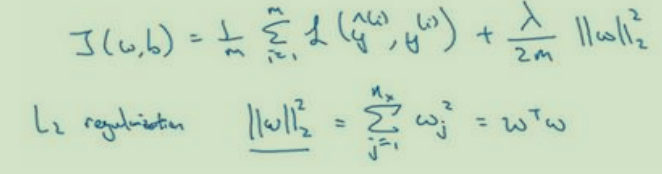
## 训练验证测试集



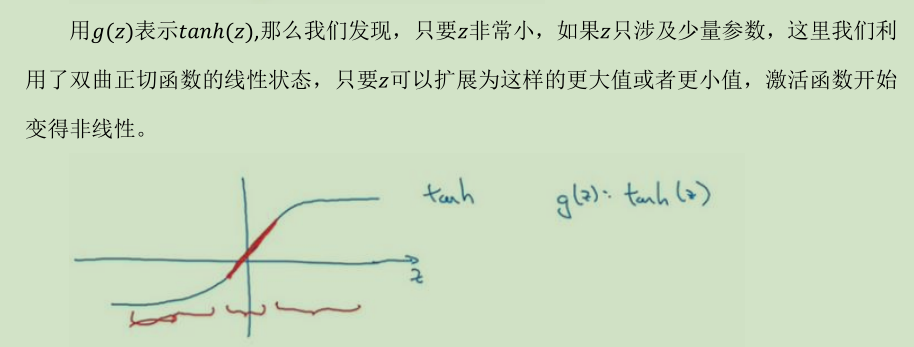
现在我们的数据量都很大，如果我有100W条数据，那么98w用来训练，1W作为验证，1W作为测试；

## 正则化

### L2范数/权重衰减



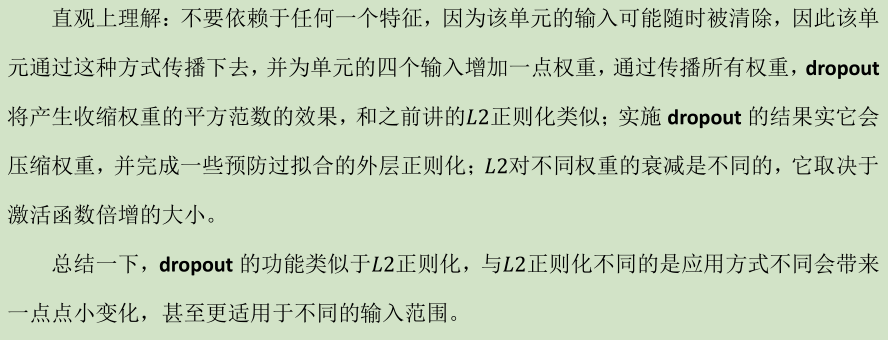
解释L2正则化，如果正则化参数lambda设置的足够大，所有w都接近于0，z也会接近于0；



这样，网络的激活函数会接近线性函数；之前讲过，线性激活函数的网络不过是个线性函数，这样，网络逐渐向线性回归靠拢，渐渐丧失过拟合的能力；

### dropout

drop用来解决过拟合，如果没有过拟合的现象，不要用。



### 其它解决过拟合的正则化方法

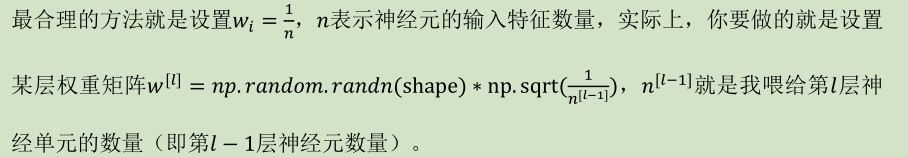
数据扩增：对称、翻转、裁切、加噪音

## 归一化输入特征有利于网络训练

## 神经网络权重初始化：

防止梯度消失/爆炸；

神经网络的计算实际上是这样的，为了防止z值过大或者过小，n越大，我们通常希望w越小，因为z是wx的和；



这个公示的意思实际上是，把标准正太分布除以的平方根，结果生成均值位0，方差为的矩阵；

其实，总结来讲，如果使用relu激活函数，设置方差为，

tanh作为激活函数，作为方差；

## 优化算法

mini-batch

优化方法：Momentum/RMSprop/Adam/Learning rate decay;

## 超参数调试：

两种调测参数的策略：

数据量超大而没有足够的计算资源时，在它实验时逐渐改良，不断调整。

有足够的计算资源时，快速实验；

## batch归一化

做数据预处理时，已经归一化了输入数据的特征，使训练更有效率，那训练过程中，归一化节点的值也会使训练更有效率。默认推荐归一化z值，即先归一化再激活。

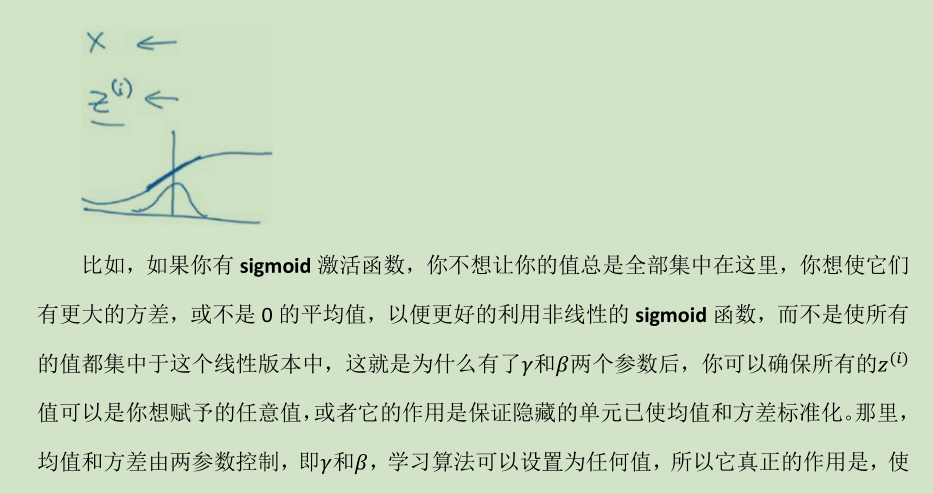
归一化不是把z固定为均值为0，方差为1，而是加入这个方程



其中

方程中的gama和bate是网络会自动更新学习的参数；

该方程的意义：



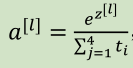
Andrew的课中讲到，因为batch-normalization操作，会重置z的均值，所以z = wx + b这一步计算中，b对z值的任何改变都会被normalization操作归一化，所以b是可以取消的。具体解释一下，batch-normalization这个思想来源于初始数据的归一化可以加快训练，也就是说，对于网络中的每个z，（每一层网络可能有n个z节点），只计算该批次数据中该z的std，mean，并进行归一化，而不是把该批次的该层所有z节点数据进行归一化。

batch-normalization有轻微的正则化效果，因为这个算法使基于mini-batch的，而每一个mini-batch的数据分布不会完全一样，这就使得每一批次的数据std，mean值不一样，就像dropout的操作一样，不使得网络过分依赖某一种数据分布；

训练时会使用mini-batch，但是实际使用网络时可能就是一次一条数据了，这时一条数据的std，mean就没有什么意义了，所以，而对于每一层的每一个z节点的std, mean，都可以用移动平均（指数加权平均）的方法来计算，这样即便在测试时，也可以使用batch-normalization。

## softmax

激活函数

为了使输出之和为1，

# 第三课 结构化机器学习项目