深度神经网络中的梯度下降法
相关公式及推导说明

李治

2018/03/08

整个深度神经网络的结构图，和运转过程如下图所示，截图来自于Ng的深度学习的课程。

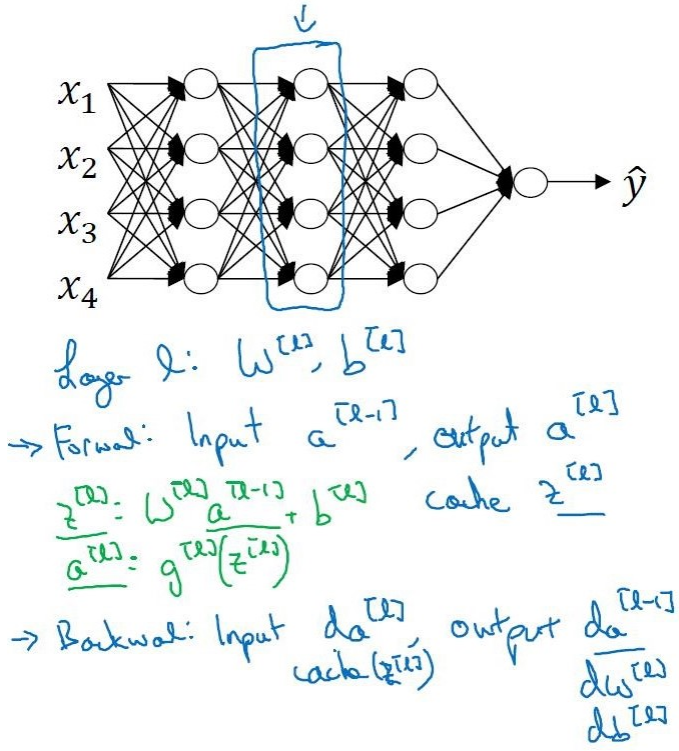

图 1: 深度神经网络结构运转示意图

1 梯度下降法

代价函数：$J(W,b)=\frac{1}{m}\sum_{i=1}^{m}(J_L(a^{[L](i)},y^i))$。其中m表示样本数，L表示神经网络的总层数，$a^{[L]}$表示第L层的输出值。$J_L$函数表示第L层的对于单个样本的代价函数。

如果最后一层，也就是第L层的激活函数是sigmoid函数，J_L函数应该如下所示。

$$J_L(a,y)=-[ylog(a)+(1-y)log(1-a)]$$

其中a表示最终预测为1的概率（比如图像识别标注中1表示是猫，0表示不是猫），y表示原本的真实值（二值变量，仅取0或者1）。

这个函数怎么来的，可以稍加解释下。

预测正确的概率p原先应该是：

$$\begin{cases} p=a, & \text{当}y=1\text{时;} \\ p=1-a, & \text{当}y=0\text{时;} \end{cases}$$

可以利用y仅取0和1，把它综合写成：

$$p=a^y(1-a)^{(1-y)}$$

若取样本量为m，根据最大似然估计（原理：一系列事件现在发生了，那我就默认这一系列事件发生的概率为最大，来求其中的未知参数），引入对数似然函数，去将连乘转变为连加。

对p取log，再加个负号就可以得到代价函数J_L函数。（这里p指预测正确的概率，肯定是越大越好，取log不影响越大越好，而我们的代价函数是求最小值，是越小越好，因此加上了一个负号）。

梯度下降法即找到下山最陡峭的方向（梯度方向），进而找到函数极小值点（非凸时，可能会陷入局部极小）。

梯度下降法的相关由来原理推荐阅读：

$$https://zhuanlan.zhihu.com/p/24913912$$

写得很通俗易懂。

下面对代价函数求导，

$$dW=\frac{\partial J}{\partial W},db=\frac{\partial J}{\partial b}$$

每一步的迭代公式(α为学习率)：

$$W = W - \alpha dW, b = b - \alpha db$$

2 前向传播过程

输入$A^{[l-1]}$,目的是输出$A^{[l]}$,同时缓存$Z^{[l]}$（因为反向传播中会用到）,l表示第l层神经网络。

$$Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]}; A^{[l]} = g^{[l]}(Z^{[l]})$$

其中$g^{[l]}$为第l层的激活函数。

Ng的课件截图如下：

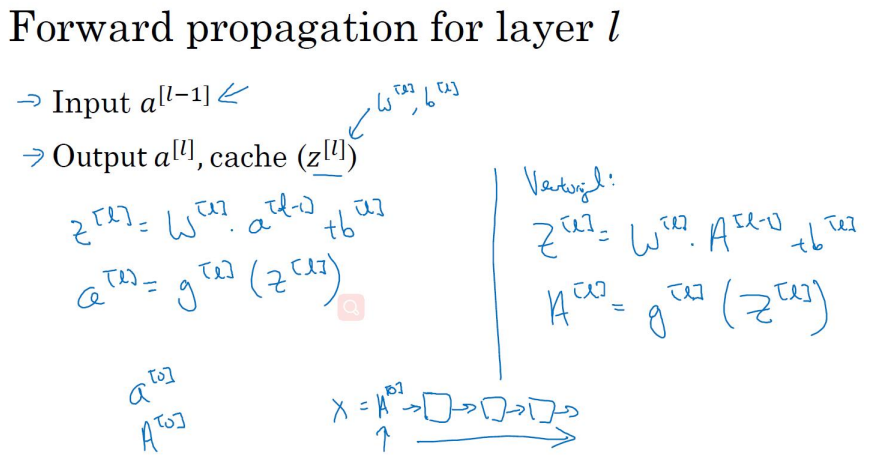

图 2: 深度神经网络前向传播截图

3 后向传播过程

输入$dA^{[l]}$（即$\frac{\partial J}{\partial A^{[l]}}$）,下面的计算过程全是根据上面写出的对应公式用链式法则得到。

$$dZ^{[l]} = \frac{\partial J}{\partial Z^{[l]}} = dA^{[l]} * g^{[l]'}(Z^{[l]})$$

$$dW^{[l]} = \frac{\partial J}{\partial W} = \frac{1}{m} dZ^{[l]} * A^{[l-1]T}$$

$$db^{[l]} = \frac{\partial J}{\partial b} = \frac{1}{m} \sum dZ^{[l]}$$

$$dA^{[l-1]} = \frac{\partial J}{\partial A^{[l-1]}} = W^{[l]T} dZ^{[l]}$$

Ng的课件截图如下：

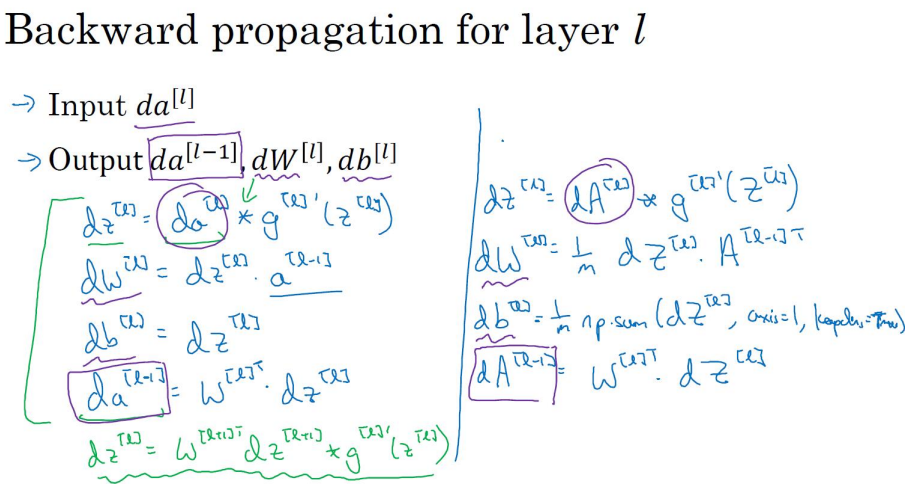

图 3: 深度神经网络后向传播截图