神经网络第二章的三个问题

张翼鹏　杨思敏

October 16, 2019

目录

1 代价函数的多种形式

代价函数是对实际输出和期望输出之间的距离关系的一种度量。

1.1 $L2$-范数代价函数

$$C(\theta)=\frac{1}{2n}\sum_{x}\|y(x)-a\|_2$$

特点：通过平方计算放大了估计值和真实值的距离，因此对偏离真实值的输出给予很大的惩罚；该函数为平滑函数，有利于误差梯度的计算。

1.2 $L1$-范数代价函数

$$C(\theta)=\frac{1}{n}\sum_{x}\|y(x)-a\|_1$$

特点：对偏离真实值的输出相对不敏感，因此在存在异常值时有利于保持模型的稳定。

1.3 交叉熵代价函数

$$C(\theta)=-\frac{1}{n}\sum_{x}\sum_{j}[y_j\ln a_j+(1-y_j)\ln(1-a_j)]$$

特点：计算代价对输出层权重和偏置的偏导时，消去了$\sigma'(z)$项，避免了输出层神经元饱和时梯度下降法训练速度下降的现象；且符合学习规律，即输出值与期望值差距越大，训练速度越快。

2 随机梯度下降算法何时收敛

画曲线图，横坐标为**迭代次数**，纵坐标为**训练集代价**和**模型在测试集上的分类准确率**。

如果分类准确率不再整体呈上升趋势，而是接近横向波动，那么迭代就可以停止了。我们可以规定，如果连续10个迭代期分类准确率没有提升，那么就停止迭代。

对于随机梯度下降算法，最后结果会在最小值附近徘徊，若想使其更好地收敛，可以随迭代次数的增加逐渐减小学习速率。

3 反向传播算法的时间复杂度

时间复杂度为：

$$O(V+E)$$

V为节点数，也即偏置数；E为连接边数，也即权重数。

根据反向传播算法的方程2：

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$$

计算每一层的误差δ时，下一层的每个权重参与了一次乘法运算，本层的每个带权输入z参与了一次运算。所以反向传播过程的时间复杂度为$O(V+E)$；而前向传播同理，根据：

$$a^l = \sigma(w^l a^{l-1} + b^l)$$

每个权重和偏置都参与一次运算，时间复杂度也是$O(V+E)$。所以对于单个样本，一次完整的反向传播算法的时间复杂度为$O(V+E)$。

另外还有一个问题，前向传播也可以实现链式法则求偏导，而且不用储存每一层的激活值向量，看似节省了空间复杂度，为什么不用前向传播求偏导。首先要说明，前向传播和反向传播求偏导的区别是先求出第一层偏导然后依次向前求，还是先求出最后一层偏导然后依次向后求。

因为反向传播仅需要完整地遍历一次神经网络，就可以求出代价对所有权重和偏置的偏导；但是前向传播，从某个权重（或偏置）出发，进行一次前向遍历，只能算出代价对这个权重（或偏置）的偏导（即路线重复，计算有冗余）。换个角度分析，反向传播求偏导时，每两个连结的神经元之间的偏导数只参与一次计算；但前向传播求偏导时，同样的量会用到多次。而且，前向传播求偏导时，对于单个权重（或偏置），前向遍历时，随着层数的增大，路径会变得越来越多，遍历每一层后算出的中间量也会越来越多，我们需要一直记录这些中间量，会消耗很多的存储空间，只有走到输出层时，才能将这些量求和合并；而反向传播求偏导时，由于每走过一层我们就会得到代价对一些权重（或偏置）的偏导，相当于一边走一边合并，计算代价对某层某个权重（或偏置）时的路径分叉量，仅仅等于上一层的结点数。