softmax 函数以及相关求导过程+交叉熵（Cross entropy）

原创 ytusdc 已于 2023-01-14 15:42:08 修改 阅读量6.8k 收藏 29 点赞数 11
分类专栏： AI之路 - Face 文章标签： 神经网络 深度学习

目录

参考文章（**需要看**）：

简单的交叉熵 损失函数，你真的懂了吗？：简单的交叉熵损失函数，你真的懂了吗？_红色石头的专栏-CSDN博客_交叉熵损失函数

python， numpy 代码实现：softmax交叉熵的两种形式 + numpy 实现

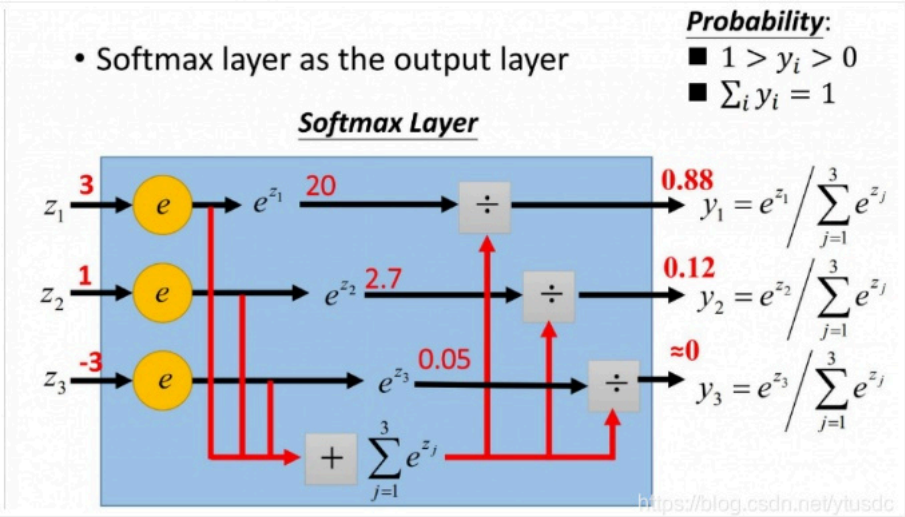
一、 softmax函数

softmax(柔性最大值)函数，一般在神经网络 中， softmax**用于多分类过程中**。它将多个神经元的输出，映射到（0,1）区间内，可以看成概率来理解进行多分类。softmax函数的公式形式：

$$S_i = \frac{e^{z_i}}{\sum_k e^{z_k}}$$

S_i代表的是第i个神经元的输出。

更形象的如下图表示：



softmax直白来说就是将原来输出是3,1,-3通过softmax函数一作用，就映射成为(0,1)的值，而这些值的累和为1（满足概率的性质），那么我们可理解成概率，在最后选取输出结点的时候，我们就可以选取概率最大（也就是值对应最大的）结点，作为我们的预测目标！

为什么 Softmax 只用在神经网络的最后一层？

现在进入重要部分，**Softmax 仅用于最后一层以对值进行归一化，而其他激活函数（relu、leaky relu、sigmoid 和其他各种）用于内层。**

如果我们看到其他激活函数，如 relu、leaky relu 和 sigmoid，它们都使用唯一的单个值来带来非线性。他们看不到其他值是什么。

但是在 Softmax 函数中，在分母中，它取所有指数值的总和来归一化所有类的值。**它考虑了范围内所有类的值，这就是我们在最后一层使用它的原因。我们分析所有的值来知道Input属于哪个类。**

对 Softmax 的误解

关于 Softmax 的第一个也是最大的误解是，它通过归一化值的输出是每个类的概率值，这完全错误。这种误解是因为这些值的总和为 1，但它们只是不是类的概率.

在最后一层并不是单独使用 Sotmax，我们更喜欢使用 Log Softmax，它只是对来自 Softmax 函数的归一化值进行对数。

Log Softmax 在数值稳定性、更便宜的模型训练成本和 Penalizes Large error（误差越大惩罚越大）方面优于 Softmax。

这就是在神经网络中用作激活函数的 Softmax 函数。相信读完本文后你对它已经有了一个清楚的了解。

具体分析参考：

log_softmax与softmax的区别 - Genpock - 博客园

二、交叉熵（Cross entropy）

首先理解一下熵的概念，假设 p 和 q 是关于样本集的两个分布，其中 **p 是样本集的真实分布，q 是样本集的估计分布**，那么按照真实分布 p 来衡样本所需要编码长度的期望（平均编码长度），即信息熵：

$$H(p) = \sum_{i}^{n} p_i log\frac{1}{p_i} = \sum_{i}^{n} -p_i log p_i$$

如果用估计分布 q 来表示真实分布 p 的平均编码长度（信息量），即交叉熵：

$$H(p,q) = \sum_{i=1}^{n} p_i log\frac{1}{q_i} = \sum_{i=1}^{n} -p_i log q_i$$

信息熵，反应的是香农信息量的期望。信息熵代表的是随机变量或整个系统的不确定性，熵越大，随机变量或系统的不确定性就越大。

交叉熵本质上可以看成，用一个猜测的分布的编码方式去编码其真实的分布，得到的平均编码长度或者信息量。交叉熵可在神经网络(机器学习)中函数，p 表示真实标记的分布，q 则为训练后的模型的预测标记分布，交叉熵损失函数可以衡量 p 与 q 的相似性。交叉熵作为损失函数还有一个好处sigmoid函数在梯度下降时能避免均方误差损失函数学习速率降低的问题，因为学习速率可以被输出的误差所控制。

交叉熵越低，这个策略就越好，最低的交叉熵也就是使用了真实分布所计算出来的信息熵，因为此时 p_k = q_k ，交叉熵 = 信息熵。这也是为学习中的分类算法中，我们总是最小化交叉熵，因为**交叉熵越低，就证明由算法所产生的策略最接近最优策略，也间接证明我们算法所算出的非真实真实分布。**

详细参考下面的两个链接：

链接1：如何通俗的解释交叉熵与相对熵？ - 知乎

链接2：该回答已被删除 - 知乎

三、softmax loss

在神经网络后面添加Softmax，真实的标签（或者是类别）就相当于真实的分布，经过Softmax得出的值就是预测的结果，因此可以使用交叉熵函失函数。有了交叉熵的概念，我们就可以得出，Softmax的损失函数：

$$Loss = -\sum_{i} y_i ln a_i$$

其中y代表我们的真实值，a代表我们softmax求出的值。i 代表的是输出结点的标号。ai 表示这个样本属于第 i 个类别的概率。

y是一个one-hot的向量表示。y是一个1*T的向量（T是softmax输出的总类别个数），里面的T个值，而且只有1个值是1，其他T-1个值都是0。那么的值是1呢？答案是真实标签对应的位置的那个值是1，其他都是0。所以这个公式其实有一个更简单的形式：

$$Loss = -log a_i$$

当然此时要限定 i 是指向当前样本的真实标签，此时的 yi=1 。因此得到上面的结果。

举个栗子：假设一个5分类问题，然后一个样本 i 的标签y=[0,0,0,1,0]，也就是说样本 i 的真实标签是4，假设模型预测的结果概率（softmax的输出[0.1,0.15,0.05,0.6,0.1]，可以看出这个预测是对的，那么对应的损失 L=-log(0.6)，也就是当这个样本经过这样的网络参数产生这样的预测p时，它的log(0.6)。那么假设p=[0.15,0.2,0.4,0.1,0.15]，这个预测结果就很离谱了，因为真实标签是4，而你觉得这个样本是4的概率只有0.1（远不如其他概率高在测试阶段，那么模型就会预测该样本属于类别3），对应损失L=-log(0.1)。那么假设p=[0.05,0.15,0.4,0.3,0.1]，这个预测结果虽然也错了，但是没有么离谱，对应的损失L=-log(0.3)。我们知道log函数在输入小于1的时候是个负数，而且log函数是递增函数，所以-log(0.6) < -log(0.3) < -log(0.1)。简单预测错比预测对的损失要大，预测错得离谱比预测错得轻微的损失要大。

3.1、交叉熵的优缺点

使用逻辑函数得到概率，并结合交叉熵当损失函数时，当模型效果差的时，学习速度较快，模型效果好时，学习速度会变慢。

采用了类间竞争机制，比较擅长于学习类间的比较散。

四、softmax 相关求导

先复习一下求导公式：

基本初等函数求导公式

(1) $(C)' = 0$

(2) $(x^{\mu})' = \mu x^{\mu-1}$

(3) $(\sin x)' = \cos x$

(4) $(\cos x)' = -\sin x$

(5) $(\tan x)' = \sec^2 x$

(6) $(\cot x)' = -\csc^2 x$

(7) $(\sec x)' = \sec x \tan x$

(8) $(\csc x)' = -\csc x \cot x$

(9) $(a^x)' = a^x \ln a$

(10) $(e^x)' = e^x$

(11) $(\log_a x)' = \dfrac{1}{x \ln a}$

(12) $(\ln x)' = \dfrac{1}{x}$,

(13) $(\arcsin x)' = \dfrac{1}{\sqrt{1-x^2}}$

(14) $(\arccos x)' = -\dfrac{1}{\sqrt{1-x^2}}$

(15) $(\arctan x)' = \dfrac{1}{1+x^2}$

(16) $(\operatorname{arc\,cot} x)' = -\dfrac{1}{1+x^2}$

函数的和、差、积、商的求导法则

设 $u = u(x)$，$v = v(x)$ 都可导，则

（1） $(u \pm v)' = u' \pm v'$

（2） $(Cu)' = Cu'$（C 是常数）

（3） $(uv)' = u'v + uv'$

（4） $\left(\dfrac{u}{v}\right)' = \dfrac{u'v - uv'}{v^2}$

我们要求的是我们的loss对于神经元输出zi的梯度，即：

$$\frac{\partial C}{\partial z_i}$$

根据复合函数求导法则：

$$\frac{\partial C}{\partial z_i} = \sum_j \left(\frac{\partial C_j}{\partial a_j}\frac{\partial a_j}{\partial z_i}\right)$$

这里为什么是 aj 而不是 ai，这里要看一下softmax的公式了，因为softmax公式的特性，它的分母包含了所有神经元的输出，所以，对于不等于i的其他面，也包含着zi，所有的 a 都要纳入到计算范围中，并且后面的计算可以看到需要分为i=j 和 i≠j 两种情况求导。
下面我们一个一个推，首先：

$$\frac{\partial C_j}{\partial a_j} = \frac{\partial(-y_j \ln a_j)}{\partial a_j} = -y_j \frac{1}{a_j}$$

第二个稍微复杂一点，我们先把它分为两种情况：

①如果$i=j$:

$$\frac{\partial a_i}{\partial z_i}=\frac{\partial(\frac{e^{z_i}}{\sum_k e^{z_k}})}{\partial z_i}=\frac{\sum_k e^{z_k}e^{z_i}-(e^{z_i})^2}{(\sum_k e^{z_k})^2}=(\frac{e^{z_i}}{\sum_k e^{z_k}})(1-\frac{e^{z_i}}{\sum_k e^{z_k}})=a_i(1-a_i)$$

②如果$i\neq j$:

$$\frac{\partial a_j}{\partial z_i}=\frac{\partial(\frac{e^{z_j}}{\sum_k e^{z_k}})}{\partial z_i}=-e^{z_j}(\frac{1}{\sum_k e^{z_k}})^2e^{z_i}=-a_ia_j$$

接下来我们只需要把上面的组合起来：

$$\frac{\partial C}{\partial z_i}=\sum_j(\frac{\partial C_j}{\partial a_j}\frac{\partial a_j}{\partial z_i})=\sum_{j\neq i}(\frac{\partial C_j}{\partial a_j}\frac{\partial a_j}{\partial z_i})+\sum_{i=j}(\frac{\partial C_j}{\partial a_j}\frac{\partial a_j}{\partial z_i})$$

$$=\sum_{j\neq i}-y_j\frac{1}{a_j}(-a_ia_j)+(-y_i\frac{1}{a_i})(a_i(1-a_i))$$

$$=\sum_{j\neq i}a_iy_j+(-y_i(1-a_i))$$

$$=\sum_{j\neq i}a_iy_j+a_iy_i-y_i$$

$$=a_i\sum_j y_j-y_i$$

最后的结果看起来简单了很多，最后，针对分类问题，我们给定的结果 yi 最终只会有一个类别是1，其他类别都是0，因此，对于分类问题，这个梯

$$\frac{\partial C}{\partial z_i}=a_i-y_i$$

知乎上求导加示意图也很清晰，两者结合看一下：详解softmax函数以及相关求导过程 - 知乎

求导写详细步骤： softmax回归推导 - JohnRed - 博客园