预备知识

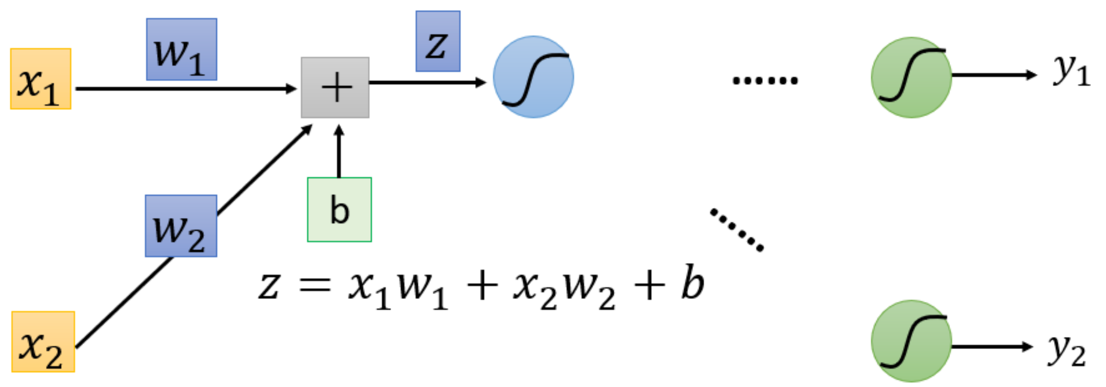
1. 前向神经网络（损失函数、sigmoid激活函数、前向传播的概念）
2. 梯度下降算法
3. 链式求导法则

反向传播最终我们求的是什么？$\frac{\partial l}{\partial w}$！

　　损失函数$Loss$关于w的偏导数！

　　其中损失函数是我们在网络模型输出层定义的以w为自变量的函数(当然也与输出值\hat{y}，真实值y有关），w是神经网络每个结点与结点之间的参数

怎么来求？我们举个例子



> 解释这张图片：输入向量是x_1, x_2,第一层第一个神经结点的参数是w_1, w_2, b，经过线性相加得到z，z经过激活函数sigmoid后得到值a（activation首字母）传入下一层，最后经过多个隐藏层得到输出y_1 和 y_2

$\frac{\partial l}{\partial w}$怎么算？先把它拆开:$\frac{\partial l}{\partial w} = \frac{\partial z}{\partial w}\frac{\partial l}{\partial z}$

问题分成了两部分，$\frac{\partial z}{\partial w}$，以及$\frac{\partial l}{\partial z}$

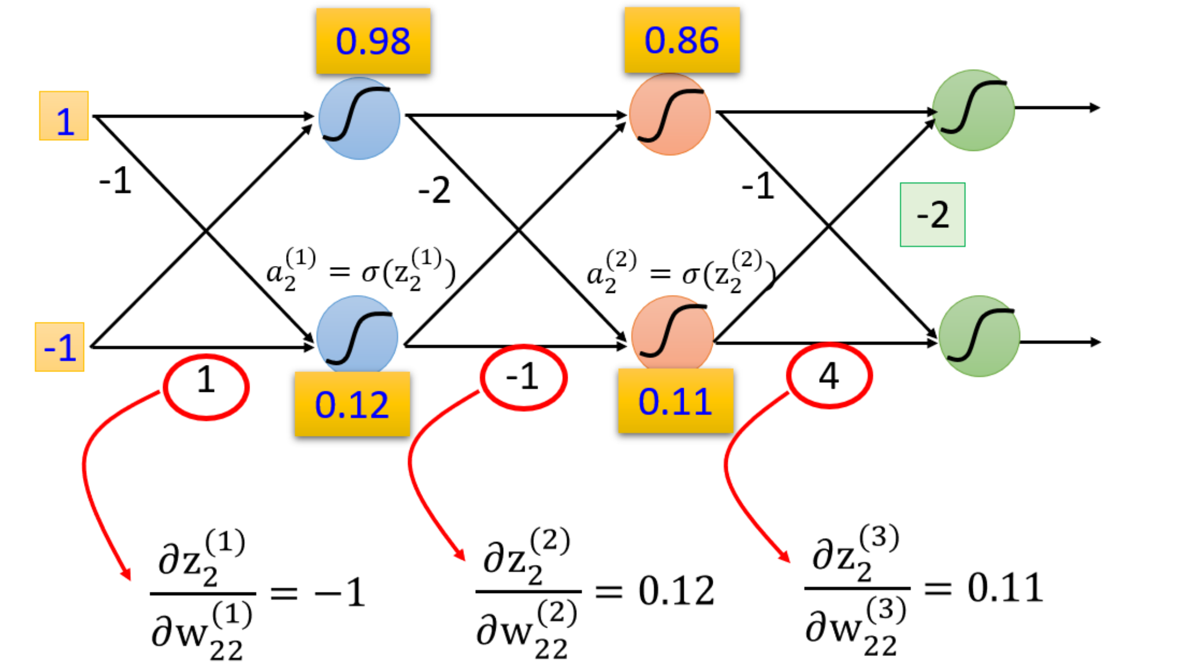
> 我们的z就是上图的z,w可以是上图的w_1或者w_2

$\frac{\partial z}{\partial w}$怎么算呐？

　基本的偏导数运算！在上图中我们已经有了$z = x_1w_1 + x_2w_2 + b$，那么显然

$$\frac{\partial z}{\partial w_1} = x_1, \frac{\partial z}{\partial w_1} = x_2$$

　在这里要延伸一下，如果要求的不在输入层，而在中间隐藏层，那$\frac{\partial z}{\partial w}$该怎么计算呢？答案：依然该结点的输入，即上一层结点的a值(activation)！！还是拿个图片举个例子



> 图片解释：输入了x_1, x_2，分别是1和-1，然后经过第一层传播，得到了$a_1^{(1)} = 0.98, a_2^{(1)} = 0.12$，$a_1^{(1)}$表示神经网络第一层第一个结点，$a_2^{(1)}$表示神经网络第一层第二个结点，这两个值是$z_1^{(1)}, z_2^{(1)}$经过激活函数sigmoid得到的，而$z_1^{(1)}, z_2^{(1)}$ 则是经过各自的线性函数得到的，即
>
> $$z_1^{(1)} = w_{11}^{(1)} x_1 + w_{12}^{(1)} x_2 + b_1^{(1)}, z_2^{(1)} = w_{21}^{(1)} x_1 + w_{22}^{(1)} x_2 + b_2^{(1)}$$
>
> 同理$z_2^{(2)}$是怎么得来的?
>
> $$z_2^{(2)} = w_{21}^{(2)} a_1^{(1)} + w_{22}^{(2)} a_2^{(1)} + b_2^{(2)}$$
>
> 同理$z_2^{(3)}$是怎么得来的?
>
> $$z_2^{(3)} = w_{21}^{(3)} a_1^{(2)} + w_{22}^{(3)} a_2^{(2)} + b_2^{(3)}$$

那么显然就有图中的几个等式：

$$\frac{\partial z_2^{(2)}}{\partial w_{22}^{(2)}} = a_2^{(1)} = 0.12, \frac{\partial z_2^{(3)}}{\partial w_{22}^{(3)}} = a_2^{(2)} = 0.11$$

休憩片刻

好的，到这里先停一下，回顾一下我们究竟在干什么?

我们为了求$\frac{\partial l}{\partial w}$，把它分解成了$\frac{\partial l}{\partial w} = \frac{\partial z}{\partial w}\frac{\partial l}{\partial z}$，同时我们会求了第一层的$\frac{\partial z}{\partial w_1} = x_1$,然后我们通过前向传播的公式，会求了任意层的z对该层的参数w求导，这时候也就意味着我们对任意一个参数w求导，其第一项我们已经成功解决，现在来搞定第二项！

$\frac{\partial l}{\partial z}$怎么算呐?再拆！ $\frac{\partial l}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial l}{\partial a}$

在此明确一下l是什么，z是什么,a又是什么。

l是损失函数，z是上一层的输入进入本神经元后线性向加得到的结果,a是这个z经过激活函数sigmoid后的结果！

现在问题又分成了两步?第一项和第二项！我们先来算第一项！

步骤1 $\frac{\partial a}{\partial z}$

由于

$$a = sigmoid(z) = \sigma(z) = \frac{1}{1+e^{-z}}$$

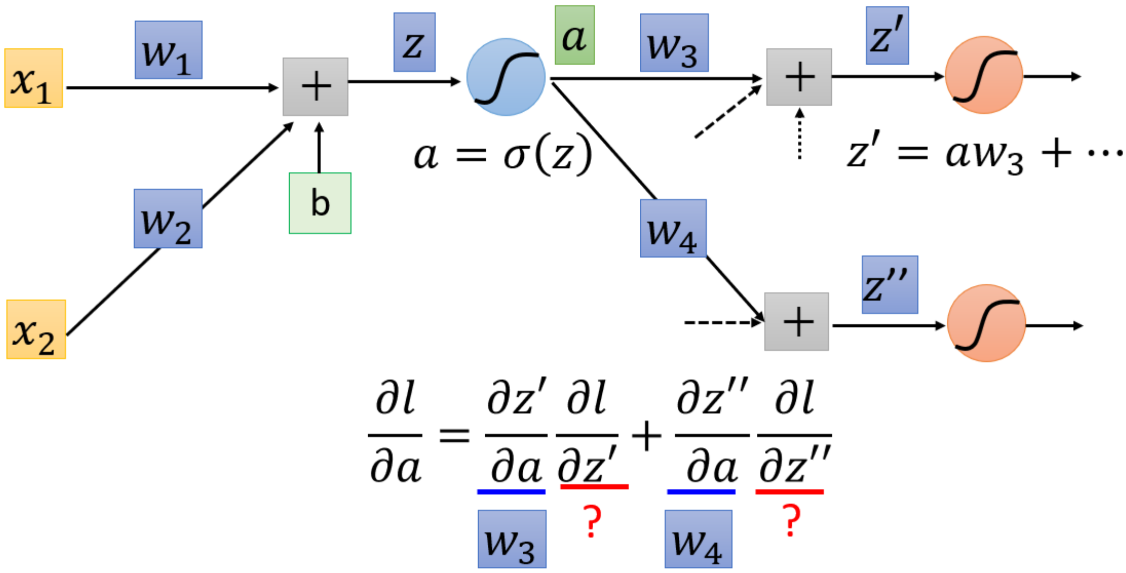
那么

$$\frac{\partial a}{\partial z} = \frac{e^{-z}}{(1+e^{-z})^2} = \frac{1+e^{-z}-1}{(1+e^{-z})^2} = \frac{1}{1+e^{-z}} - \frac{1}{(1+e^{-z})^2} = \frac{1}{1+e^{-z}} * (1 - \frac{1}{1+e^{-z}}) = \sigma(z)(1-\sigma(z))$$

第一项有了！

步骤2 $\frac{\partial l}{\partial a}$

怎么算？再拆！怎么拆？不好写，画个图举例



> 解释该图片：这里为了方便起见不再用上下标表示z和w在整个神经网络的具体位置，直接看图中的表示即可
>
> 我们使用的链式求到规则：
>
> $$\begin{tikzcd} & z' \arrow[dr] & \\ a \arrow[ur] \arrow[dr] & & l \\ & z'' \arrow[ur] & \end{tikzcd}$$
>
> 如果隐藏层单元更多，我们当然需要增加更多的$z''', z'''', z''''', z''''''$....作为中间变量,这里是表达的简洁形式
>
> 同时要了解a的下一层就是z', z'',而最终在得到l的输出层，可能距离z产生那一层中间还隔了许多个隐藏层！
>
> 因此我们通过链式求导得到了
>
> $$\frac{\partial a}{\partial z}\frac{\partial l}{\partial a} = a'(\frac{\partial z'}{\partial a}\frac{\partial l}{\partial z'} + \frac{\partial z''}{\partial a}\frac{\partial l}{\partial z''})$$
> $$其中\frac{\partial z'}{\partial a}即为参数w_3, \frac{\partial z''}{\partial a}即为参数w_4(线性函数求偏导)$$

因此现在我们有了

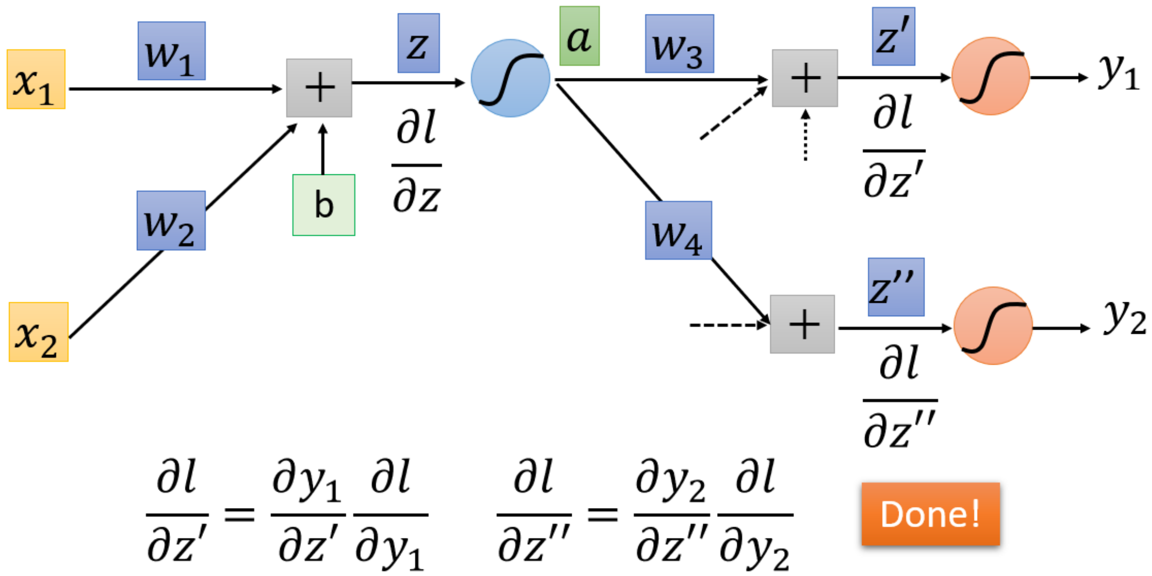
$$\frac{\partial l}{\partial z} = \frac{\partial a}{\partial z}\frac{\partial l}{\partial a} = a'(w_3\frac{\partial l}{\partial z'} + w_4\frac{\partial l}{\partial z''})$$

那么$\frac{\partial l}{\partial z'}$和$\frac{\partial l}{\partial z''}$可咋求？？

这还用说，你都会$\frac{\partial l}{\partial z}$了还不会$\frac{\partial l}{\partial z'}$和$\frac{\partial l}{\partial z''}$吗?

这时候可能又用同学上来问，哥这啥时候是个头啊？ 求到最后一层就是输出结束了！

我们最好还是来看看最后长啥样子！



> 我们把上个图变换一下，认为下一层就是输出结点，输出y_1, y_2
>
> $$\frac{\partial l}{\partial z} = a'(w_3\frac{\partial l}{\partial z'} + w_4\frac{\partial l}{\partial z''})$$
>
> 那么还是应用链式求导
>
> $$\frac{\partial l}{\partial z'} = \frac{\partial y_1}{\partial z'}\frac{\partial l}{\partial y_1}, \frac{\partial l}{\partial z''} = \frac{\partial y_2}{\partial z''}\frac{\partial l}{\partial y_2}$$

其中y是z'经过sigmoid函数得到的，而l则是损失函数，他直接与y相关，可以通过损失函数的定义求l关于y的偏导

举个例子，当损失函数定义为平方损失函数时

$$l = \sum_i^n (y_i - \hat{y}_i)^2$$

关于$\frac{\partial l}{\partial y_i}$的偏导 就是对$y_i$直接求导即可

总结

现在我们已经能求出损失函数l对任意的参数w的梯度了！

从新整理一下思路！

$$(1)\text{我们求}\frac{\partial l}{\partial w}\text{，咋么算呐?拆分！}\frac{\partial l}{\partial w}=\frac{\partial z}{\partial w}\frac{\partial l}{\partial z}$$

$$(2)\frac{\partial z}{\partial w}\text{就是上一层对应结点的}a\text{值，}\frac{\partial l}{\partial z}\text{怎么算呐?再拆！}\frac{\partial l}{\partial z}=\frac{\partial a}{\partial z}\frac{\partial l}{\partial a}\text{怎么算呐？再拆！}$$

$$(3)\frac{\partial a}{\partial z}\text{用}sigmoid\text{求偏导直接得到，}\frac{\partial l}{\partial a}\text{怎么算呐？再拆！}\left(\frac{\partial z'}{\partial a}\frac{\partial l}{\partial z'}+\frac{\partial z''}{\partial a}\frac{\partial l}{\partial z''}+\ldots\right)$$

$$(4)\text{其中}\frac{\partial z'}{\partial a},\frac{\partial z''}{\partial a},\frac{\partial z'''}{\partial a}\ldots\text{就是对应的参数}w\text{，而后一项}\frac{\partial l}{\partial z'}\text{，}\frac{\partial l}{\partial z''}\text{，}\frac{\partial l}{\partial z'''}\text{按照步骤}2\text{的拆法继续直到最后一层}$$

$$(5)\text{最后项}\frac{\partial y^1}{\partial z^1},\frac{\partial y^2}{\partial z^2}\ldots\text{用}sigmoid\text{求导直接得到，}\frac{\partial l}{\partial y^1},\frac{\partial l}{\partial y^2}\ldots\text{用损失函数对}y\text{求导直接得到}$$

升华

结束了吗？没有？哪里没有结束？

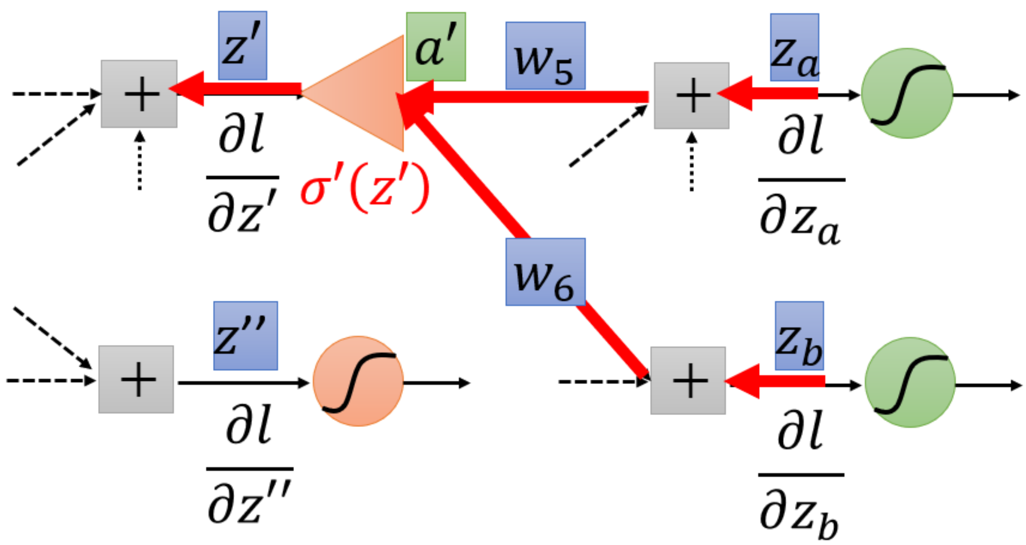
我们针对一个参数w求梯度，就要从它出现的这一层开始，一步步往回推导到，最终算出了梯度值，但是针对每一次神经网络的训练，我们要用For循环对每一个样本求梯度，再用for循环求每一个参数w的梯度，再在循环for循环里求参数w出现开始往后一步步的导数值，不现实，如何简化，如何向量化？这才是反向传播的意义所在！

> 什么是向量化？通俗地说是指能够使用矩阵代替循环操作进行运算，为什么要向量化？这样在实际运算时能够利用计算机并行处理的能力提高计算速度，前向神经网络的向量化简单来说就是 $Y=W^T*X+b$

怎么办？

从一个小的部分开始理解反向传播

举个例子：



> $$\text{求}\frac{\partial l}{\partial z'}$$
>
> $$\text{过程1：}\frac{\partial l}{\partial a'}=\left(\frac{\partial z_a}{\partial a'}\frac{\partial l}{\partial z_a}+\frac{\partial z_b}{\partial a'}\frac{\partial l}{\partial z_b}\right)=w_5\frac{\partial l}{\partial z_a}+w_6\frac{\partial l}{\partial z_b}$$
>
> $$\text{过程2：}\frac{\partial l}{\partial z'}=\frac{\partial a'}{\partial z'}\frac{\partial l}{\partial a'}=\sigma'(z')\frac{\partial l}{\partial a'}$$

现在只看图中红线，我们求解的方法把$\frac{\partial l}{\partial z_a}$和$\frac{\partial l}{\partial z_b}$作为输入，把$w_5$和$w_6$作为参数，进行前馈神经网络传播，然后在新的结点处进行激活，激活的方法是乘以$\sigma'(z')$，然后得到新的量$\frac{\partial l}{\partial z'}$继续向左传播！

整体理解反向传播

我们倒着来看上面的12345步！！！倒着来看神经网络！！！输出层就成了第一层！

$$（5）第一层（输出层）sigmoid求导得到\frac{\partial y^1}{\partial z^1}\frac{\partial y^2}{\partial z^2}...,损失函数对y求导直接得到\frac{\partial l}{\partial y^1}\frac{\partial l}{\partial y^2}...$$

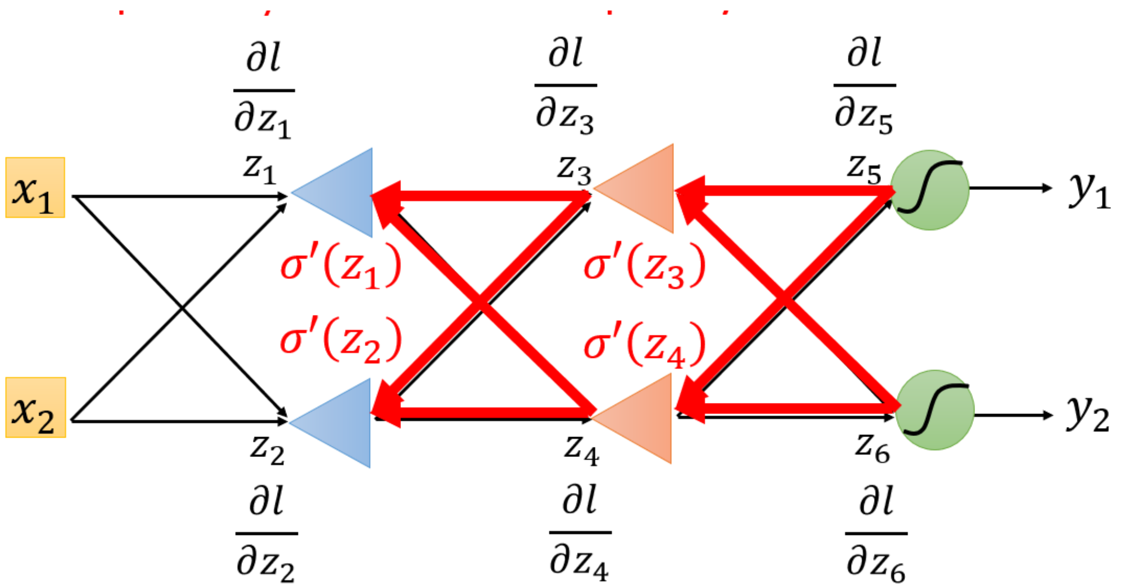
$$（4）\frac{\partial z^1}{\partial a^1},\frac{\partial z^2}{\partial a^2},..就是对应的参数w，我们用\frac{\partial z^1}{\partial a}\frac{\partial y^1}{\partial z^1}\frac{\partial l}{\partial y^1}=\frac{\partial l}{\partial a^1}，同理得到\frac{\partial l}{\partial a^2}....$$

$$(3)\frac{\partial a}{\partial z}用sigmoid求导直接得到直接求导得到,然后用\frac{\partial l}{\partial a}\frac{\partial a}{\partial z}得到\frac{\partial l}{\partial z}$$

$$(2)我们把\frac{\partial l}{\partial z}按照第4）步的方法继续乘以参数w，向上一层传播，在同3)一样乘以对应的\frac{\partial a}{\partial z},这等同于在反向传播$$

$$(1)传播至到参数w所在及结点后\frac{\partial z}{\partial w}就是上一层对应结点的a值，然后用\frac{\partial l}{\partial z}\frac{\partial z}{\partial w}得到最终的\frac{\partial l}{\partial w}即该参数的梯度$$

还是拿图来看看



我们只需要保存 输出层$\frac{\partial l}{\partial y}$, $\frac{\partial y}{\partial z}$ 保存每个神经元的 $\frac{\partial a}{\partial z}$，保存每个神经元的$a$值，然后从最后一层以 $\frac{\partial l}{\partial y}$为输入向量，乘对应的参数$w$，得到对应的$\frac{\partial l}{\partial z}$ 输入至下一结点，在下一个结点处乘以该结点的$\frac{\partial a}{\partial z}$即$\sigma'(z)$，作为新的输入向量传入下一结点以此类推直至要求的参数w结点，再乘以该结点的a值，得到梯度。

参考（基本是照搬）：

李宏毅深度学习机器学习课程