一个例子弄清神经网络

1. 前言

当遇到棘手问题的时候，逃避它往往很容易。对我而言，人工神经网络中的反向传播算法就是属于这样一个棘手的问题之一。而Tensorflow给我的逃避提供了很好的条件。当日子越久，觉得自己好像什么也不会。今天我要正视这个问题。我们来一起彻底弄清楚反向传播是怎么一回事。勇士们，我们一起看下去吗。这篇文章提供反向传播理论支持，理论结束之后，我们从头编写一个神经网络，实现手写字符的识别。没有Tensorflow, 没有PyTorch。对我而言，公式推导的过程中最让人疑惑的可能不是内容本身，而是看起来吓人的公式。原本公式是为了简化内容的啊。我多希望看到公式不懂的时候，有个人在旁边告诉我他们是什么意思。这篇文章我想尽一起办法对初学者友好。如果你觉得有些太过简单，可自行跳过，专注于你关注的部分。

2. 网络学习的目标

在神经网络中，我们用损失函数(cost function/loss function)来表示神经网络在给定数据集上面的表现。通常来说，我们希望损失函数越小越好， 而这个损失函数越变越小的过程就是学习(learning)的过程。在手写字符识别这例子里，在单个训练样本(**图像-标签对**)的损失函数, 即预测输出和标签之间的差距,我们可以这样定义:

$$C(\mathbf{w},\mathbf{b}) = \frac{1}{2}\|\mathbf{y}-\mathbf{o}\|^2 = \frac{1}{2}\sum_{i=0}^{9}(\mathbf{y}_i-\mathbf{o}_i)^2 \tag{1}$$

一张图片大小为28x28，我们将其展开为一个一维的向量 $\mathbf{x}$, 那么这个向量有784个分量。这张图片对应的标签同样用一个一维的向量$\mathbf{y}$来表示， 由于我们最终分了10类，因此标签向量有10个分量。同理神经网络的输出也可以表示成一个有10个分量的向量$\mathbf{o}$。神经网络当中所有的权重可以构成一个向量 $\mathbf{w}$, 同样地所有偏置也可以构成一个向量 $\mathbf{b}$。输入一张图片$\mathbf{x}$， 经过由$\mathbf{w}$和$\mathbf{b}$所确定的神经网络的前向传播之后，我们可以得到对应输出$\mathbf{o}_{\mathbf{w},\mathbf{b}}(\mathbf{x})$(简写为$\mathbf{o}$)。 多个训练样本的损失函数就是单个损失函数(记为C^i)的叠加:

$$L(\mathbf{w},\mathbf{b}) = \sum_{i=1}^{N} C^n(\mathbf{w},\mathbf{b}) \tag{2}$$

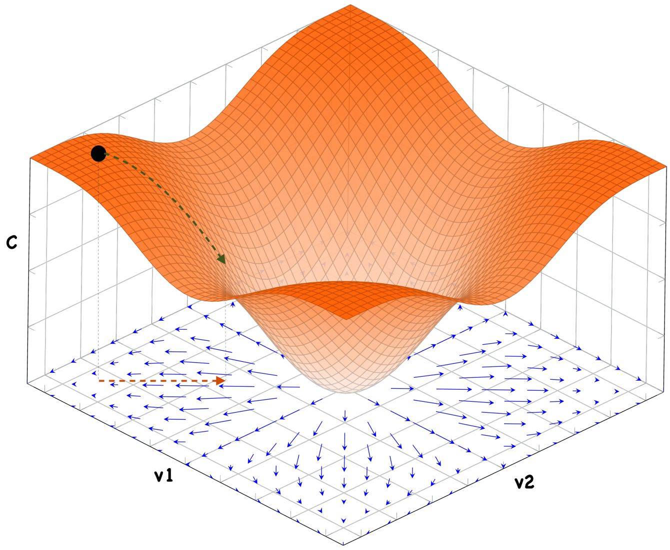
神经网络**学习**的过程就是不断调整$\mathbf{w}$, $\mathbf{b}$ 让 $L(\mathbf{w},\mathbf{b})$逐渐变小的过程。

3. 参数更新的策略

我们现在来考虑，如何才能够使得单个训练样本的损失函数C越来越小。我们将这个时刻的损失函数记为C, 下个时刻的损失函数记为C', 那么损失的变化量可以表示成:

$$\Delta C = C' - C \tag{3}$$

要让损失函数函数越来越小，我们只要想办法让ΔC每次都为负就可以了。接下来的问题就变成了: **如何才能使ΔC每次都为负?** 为了说清楚，我们现在考虑一个简单的情况，假设损失函数$C(\mathbf{v})$只包含两个参数$v1, v2$, 即$C(\mathbf{v}) = C(v1, v2)$。



根据为微积分的知识，损失函数的变化量可以表示成:

$$\Delta C \approx \frac{\partial C}{\partial v_1}\Delta v_1 + \frac{\partial C}{\partial v_2}\Delta v_2 = \left[\frac{\partial C}{\partial v_1}, \frac{\partial C}{\partial v_2}\right]^{\mathsf{T}} \cdot [\Delta v_1, \Delta v_1] \tag{4}$$

其中符号 · 表示的是向量的内积。为了方便记 $\nabla C = \left[\frac{\partial C}{\partial v_1}, \frac{\partial C}{\partial v_2}\right]^{\mathsf{T}}$, $\Delta v = [\Delta v_1, \Delta v_1]^{\mathsf{T}}$, 于是上式可以写成:

$$\Delta C \approx \nabla C \cdot \Delta v \tag{5}$$

回到我们的问题: 如何才能使ΔC每次都为负? 于是有聪明人想出了这样一个办法。让参数v按照下面的方法更新, 新的参数

$$v' = v - \eta \nabla C \tag{6}$$

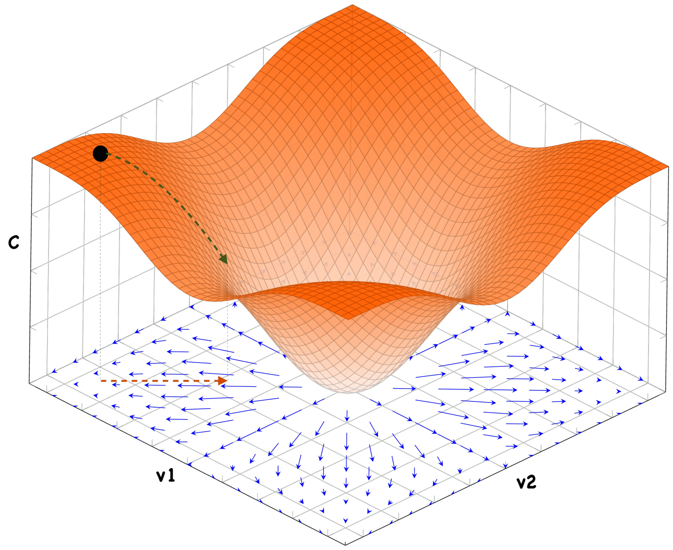
其中η是一个正数。你应该会问为什么要这样更新呢? 把等式右边的v移到左边就有

$$\Delta v = v' - v = -\eta \nabla C \tag{7}$$

将这个式子带入到损失函数的改变量中就得到

$$\Delta C \approx \nabla C \cdot (-\eta \nabla C) = -\eta(\nabla C)^2 \tag{8}$$

由于 我们规定$\eta > 0$, 且$(\nabla C)^2 \geq 0$, 因此就有$\Delta C < 0$。于是我们就找到一种参数更新策略使得每次更新都会使得损失函数减小。



我们把$\nabla C = \left[\frac{\partial C}{\partial v_1}, \frac{\partial C}{\partial v_2}\right]^{\mathsf{T}}$叫做损失函数$C$在$(v_1, v_2)$处的**梯度**。上图中用$v_1$-$v_2$平面上的蓝色箭头表示梯度的大小和方向。由于在上面的策略当中，参数v的变化方向(红色箭头)和梯度的方向正好相反，即使得梯度减小。因此上面的参数更新策略被叫做**梯度下降**方法。

具体来说，对于上述例子中的每个参数的更新策略，可以由$v' = v - \eta\nabla C$的到:

$$\begin{aligned} v_1 \rightarrow v_1' &= v_1 - \eta\frac{\partial C}{\partial v_1} \\ v_2 \rightarrow v_2' &= v_2 - \eta\frac{\partial C}{\partial v_2} \end{aligned} \tag{9}$$

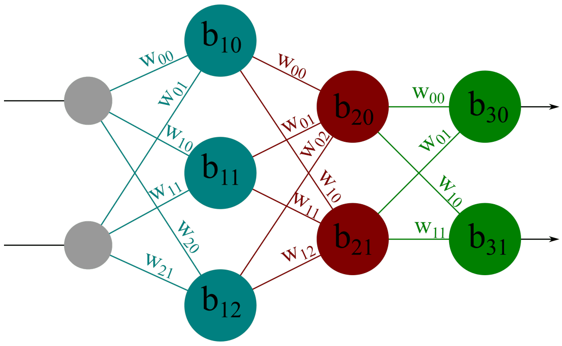
于是从二元函数推广，可以得到神经网络中任意一个权重和偏置的更新策略，

$$\begin{aligned} w_k \rightarrow w_k' &= w_k - \eta\frac{\partial C}{\partial w_k} \\ b_l \rightarrow b_l' &= b_l - \eta\frac{\partial C}{\partial b_l} \end{aligned} \tag{10}$$

到这里我们就找到了一种神经网络参数更新的策略，每一次网络参数的更新都可以让损失函数的值减小，从而使得网络的预测的值越来越接近训练数据的标签。要实现这个策略，我们只需要计算损失函数对所有的权重和偏置的偏微分，即梯度的分量就可以了。

4. 计算梯度分量的方法

接下来要做的就是找到一个方法来计算损失函数C对任意一个权重和偏置的偏微分。为了说清楚，这里我们使用下面这个简单的神经网络来做演示。这个神经网络有4层，每层的神经元个数分别是[2, 3, 2, 2]。由于输入层的神经元没有`加权求和过激活`的操作，我这里把输入层叫做第0层。随后的1, 2, 3层分别用不同颜色加以区分。



对于非输入层上的任意一个神经元的输出值a都可以表示成:

$$a = \sigma(z) \tag{11}$$

其中 z 表示这个神经元加权求和的结果，σ 表示函数Sigmoid。具体来说，对于第1层第0个神经元的输出值为

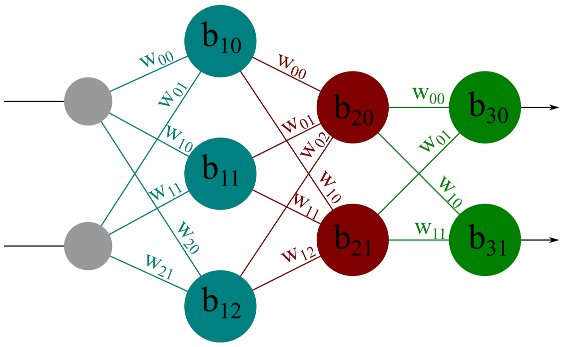
$$a_{10} = \sigma(z_{10}) \tag{12}$$

其中$z_{10} = w_{00}a_{00} + w_{01}a_{01} + b_{10}$。可见任意一个神经元的输出$a$是中间值$z$的函数，而$z$又是$w$和$b$的函数。

同理，我们认为损失函数C是是权重w和偏置b的函数，我们同样也可以认为C是中间变量z的函数。损失函数对第1层的其中一个权重w_{00}的偏导数，据链式求导法则可以得到:

$$\frac{\partial C}{\partial w_{00}} = \frac{\partial C}{\partial z_{10}} \frac{\partial z_{10}}{\partial w_{00}} \tag{13}$$

计算 $\frac{\partial z}{\partial w}$



由于激活$a_{10} = \sigma(z_{10})$ 的中间变量 $z_{10} = w_{00}a_{00} + w_{01}a_{01} + b_{10}$， 因此有

$$\frac{\partial z}{\partial w_{00}} = a_{00} \tag{14}$$

即为与该神经元相连的上一层神经元的输出值。于是可以得到这一层中连接到该神经元的权重w对应的$\frac{\partial z}{\partial w}$ 分别为a_{00}, a_{01}；那么整个第1层所有权重(我们用一个矩阵表示)

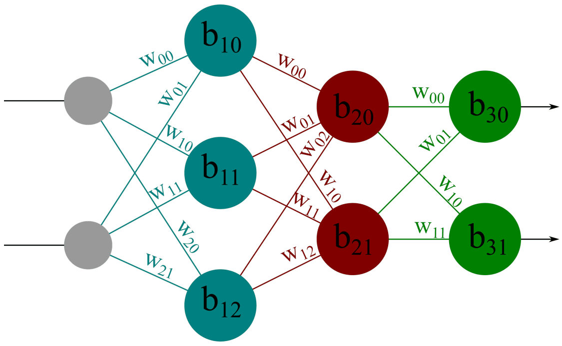
$$\begin{bmatrix} w_{00} & w_{01} \\ w_{10} & w_{11} \\ w_{20} & w_{21} \end{bmatrix} \tag{15}$$

对应的$\frac{\partial z}{\partial w}$ 为

$$\begin{bmatrix} a_{00} & a_{01} \\ a_{00} & a_{01} \\ a_{00} & a_{01} \end{bmatrix} \tag{16}$$

注意到， 每一行对应的值是**上一层**神经元的输出。

计算 $\frac{\partial C}{\partial z}$



$$\frac{\partial C}{\partial z_{10}} = \frac{\partial C}{\partial a_{10}} \frac{\partial a_{10}}{\partial z_{10}} \tag{17}$$

因为$a_{10} = \sigma(z_{10})$, 因此对上式中第二个乘积项可以写成

$$\frac{\partial a_{10}}{\partial z_{10}} = \sigma'(z_{10}) \tag{18}$$

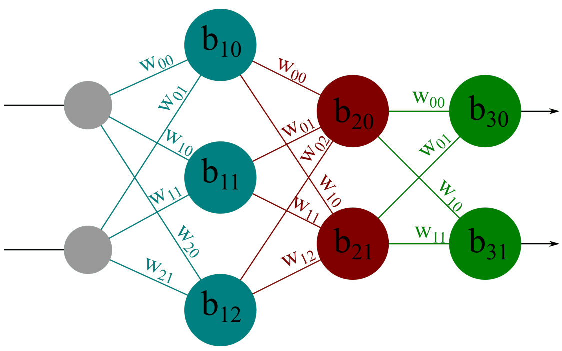
那么这一层所有节点对应的 $\frac{\partial a}{\partial z}$ 构成向量为

$$[\sigma'(z_{10}), \sigma'(z_{11}), \sigma'(z_{12})]^{\top} \tag{19}$$

z 在前向传播的时候可以容易地计算出来，而Sigmoid的导数 $\sigma'(z)$ 也是知道的。

于是还剩下一项 $\frac{\partial C}{\partial a_{10}}$, (Todo, 添加一段解释)

$$\frac{\partial C}{\partial a_{10}} = \frac{\partial C}{\partial z_{20}} \frac{\partial z_{20}}{\partial a_{10}} + \frac{\partial C}{\partial z_{21}} \frac{\partial z_{21}}{\partial a_{10}} \tag{20}$$



由于 $z_{20} = w_{00}a_{10} + w_{01}a_{11} + w_{02}a_{12}$, 因此 $\frac{\partial z_{20}}{\partial a_{10}} = w_{00}$, 同理 $\frac{\partial z_{21}}{\partial a_{10}} = w_{10}$, 于是上式变成:

$$\frac{\partial C}{\partial a_{10}} = \frac{\partial C}{\partial z_{20}} w_{00} + \frac{\partial C}{\partial z_{21}} w_{10} \tag{21}$$

同理，这一层其他神经元对应有

$$
\begin{aligned}
\frac{\partial C}{\partial a_{11}} &= \frac{\partial C}{\partial z_{20}} w_{01} + \frac{\partial C}{\partial z_{21}} w_{11} \\
\frac{\partial C}{\partial a_{12}} &= \frac{\partial C}{\partial z_{20}} w_{02} + \frac{\partial C}{\partial z_{21}} w_{12}
\end{aligned} \tag{22}
$$

写成矩阵的形式是这样的

$$
\begin{bmatrix} \frac{\partial C}{\partial a_{10}} \\ \frac{\partial C}{\partial a_{11}} \\ \frac{\partial C}{\partial a_{12}} \end{bmatrix} = \begin{bmatrix} w_{00} & w_{10} \\ w_{01} & w_{11} \\ w_{02} & w_{12} \end{bmatrix} \begin{bmatrix} \frac{\partial C}{\partial z_{20}} \\ \frac{\partial c}{\partial z_{21}} \end{bmatrix} = \begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \frac{\partial C}{\partial z_{20}} \\ \frac{\partial c}{\partial z_{21}} \end{bmatrix} \tag{23}
$$

于是有

$$
\begin{aligned}
\frac{\partial C}{\partial z_{10}} &= \frac{\partial C}{\partial a_{10}} \frac{\partial a_{10}}{\partial z_{10}} \\
&= \sigma'(z_{10}) \frac{\partial C}{\partial a_{10}} \\
&= \sigma'(z_{10}) \left(w_{00} \frac{\partial C}{\partial z_{20}} + w_{10} \frac{\partial C}{\partial z_{21}} \right)
\end{aligned} \tag{24}
$$

再回到

$$
\begin{aligned}
\frac{\partial C}{\partial w_{00}} &= \frac{\partial C}{\partial z_{10}} \frac{\partial z_{10}}{\partial w_{00}} \\
&= a_{00} \frac{\partial C}{\partial z_{10}} \\
&= a_{00} \sigma'(z_{10}) \left(w_{00} \frac{\partial C}{\partial z_{20}} + w_{10} \frac{\partial C}{\partial z_{21}} \right)
\end{aligned} \tag{25}
$$

$$
\frac{\partial C}{\partial w_{00}^{i}} = a_{00} \sigma'(z_{10}) \left(w_{00}^{j} \frac{\partial C}{\partial z_{20}} + w_{10}^{j} \frac{\partial C}{\partial z_{21}} \right) \tag{26}
$$

汇总一下:

$$
\nabla \mathbf{w}^{i} = \begin{bmatrix} a_{00} & a_{01} \\ a_{00} & a_{01} \\ a_{00} & a_{01} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial C}{\partial z_{10}} \\ \frac{\partial C}{\partial z_{11}} \\ \frac{\partial C}{\partial z_{12}} \end{bmatrix} \tag{27}
$$

其中,

$$
\begin{aligned}
\begin{bmatrix} \frac{\partial C}{\partial z_{10}} \\ \frac{\partial C}{\partial z_{11}} \\ \frac{\partial C}{\partial z_{12}} \end{bmatrix} &= \begin{bmatrix} \sigma'(z_{10}) \\ \sigma'(z_{11}) \\ \sigma'(z_{12}) \end{bmatrix} \odot \begin{bmatrix} \frac{\partial C}{\partial a_{10}} \\ \frac{\partial C}{\partial a_{11}} \\ \frac{\partial C}{\partial a_{12}} \end{bmatrix} \\
&= \begin{bmatrix} \sigma'(z_{10}) \\ \sigma'(z_{11}) \\ \sigma'(z_{12}) \end{bmatrix} \odot \left(\begin{bmatrix} w_{00} & w_{10} \\ w_{01} & w_{11} \\ w_{02} & w_{12} \end{bmatrix} \begin{bmatrix} \frac{\partial C}{\partial z_{20}} \\ \frac{\partial c}{\partial z_{21}} \end{bmatrix} \right) \\
&= \begin{bmatrix} \sigma'(z_{10}) \\ \sigma'(z_{11}) \\ \sigma'(z_{12}) \end{bmatrix} \odot \left(\begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \end{bmatrix}^{\top} \begin{bmatrix} \frac{\partial C}{\partial z_{20}} \\ \frac{\partial c}{\partial z_{21}} \end{bmatrix} \right) \\
&= \begin{bmatrix} \sigma'(z_{10}) \\ \sigma'(z_{11}) \\ \sigma'(z_{12}) \end{bmatrix} \odot \left(\mathbf{w}^{j\top} \begin{bmatrix} \frac{\partial C}{\partial z_{20}} \\ \frac{\partial c}{\partial z_{21}} \end{bmatrix} \right)
\end{aligned} \tag{28}
$$

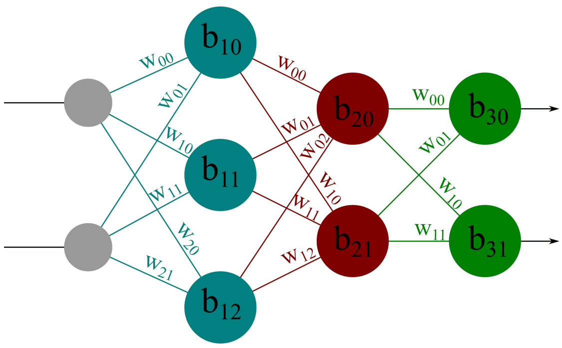
即

$$
\frac{\partial C}{\partial \mathbf{z_1}} = \sigma'(\mathbf{z_1}) \odot \left(\mathbf{w}^{2\top} \frac{\partial C}{\partial \mathbf{z_2}} \right) \tag{29}
$$

于是:

$$
\nabla \mathbf{w}^1 = \begin{bmatrix} a_{00} & a_{01} \\ a_{00} & a_{01} \\ a_{00} & a_{01} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial C}{\partial z_{10}} \\ \frac{\partial C}{\partial z_{11}} \\ \frac{\partial C}{\partial z_{12}} \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ a_{00} & a_{01} \\ a_{00} & a_{01} \end{bmatrix} \odot \left\{ \begin{bmatrix} \sigma'(z_{10}) \\ \sigma'(z_{11}) \\ \sigma'(z_{12}) \end{bmatrix} \odot \left(\mathbf{w}^{2\top} \begin{bmatrix} \frac{\partial C}{\partial z_{20}} \\ \frac{\partial C}{\partial z_{21}} \end{bmatrix} \right) \right\} \tag{30}
$$

即只要算出每个神经元的 $\frac{\partial c}{\partial z}$ 即可算出梯度的改变量， 先把最后一层的 $\frac{\partial c}{\partial z}$ 算出，一层层就可往前算出了。



输出层，

$$
\mathrm{C}(\mathbf{w}, \mathbf{b}) = \frac{1}{2} \|\mathbf{y} - \mathbf{o}\|^2 = \frac{1}{2} \sum_{i=0}^{1} (\mathbf{y}_i - \mathbf{o}_i)^2 = \frac{1}{2} \left(y_0^2 - 2y_0 o_0 + o_0^2 + y_1^2 - 2y_1 o_1 + o_1^2 \right) \tag{31}
$$

$$
o_0 = \sigma(z_{30}) \tag{32}
$$

$$
\frac{\partial C}{\partial z_{30}} = \frac{\partial C}{\partial o_0} \frac{\partial o_0}{\partial z_{30}} = (o_0 - y_0)\sigma'(z_{30}) \tag{33}
$$

同理

$$
\frac{\partial C}{\partial z_{31}} = \frac{\partial C}{\partial o_1} \frac{\partial o_1}{\partial z_{31}} = (o_1 - y_1)\sigma'(z_{31}) \tag{34}
$$

$$\frac{\partial C}{\partial \mathbf{z_3}} = (\mathbf{o} - \mathbf{y})\sigma'(\mathbf{z_3}) \tag{35}$$

总结

相邻两层i, j关键量的计算关系:

$$\frac{\partial C}{\partial \mathbf{z_i}} = \sigma'(\mathbf{z_i}) \odot \left(\mathbf{w^j}^\intercal \frac{\partial C}{\partial \mathbf{z_j}} \right) \tag{36}$$

最后一层若j=o, 即第j层是输出层:

$$\frac{\partial C}{\partial \mathbf{z_o}} = (\mathbf{o} - \mathbf{y})\sigma'(\mathbf{z_o}) \tag{37}$$

第i层w的改变量:

$$\nabla \mathbf{w}^i = [\mathbf{a}_h, \mathbf{a}_h, \mathbf{a}_h]^\intercal \odot \frac{\partial C}{\partial \mathbf{z}_i} \tag{38}$$

对于偏置

仿照类似的方法可以得到:

$$\frac{\partial C}{\partial b_{10}} = \frac{\partial C}{\partial z_{10}} \frac{\partial z_{10}}{\partial b_{10}} \tag{39}$$

前向传播计算 $\frac{\partial z}{\partial b}$

$a_{10} = \sigma(z_{10})$；$z_{10} = w_{00}a_{00} + w_{01}a_{01} + b_{10}$；因此, $\frac{\partial z_{10}}{\partial b_{10}} = 1$

那么,

$$\frac{\partial \mathbf{z}_1}{\partial \mathbf{b}_1} = [1, 1, 1]^\intercal \tag{40}$$

反向传播计算 $\frac{\partial C}{\partial z}$

$$\nabla \mathbf{b}^i = \frac{\partial C}{\partial \mathbf{z}_i} \tag{41}$$

对每一个训练样本得到的梯度分量求和，即可得到总的梯度分量:

$$\begin{aligned} \frac{\partial \mathrm{L}(\mathbf{w}, \mathbf{b})}{\partial w} &= \sum_{n=1}^{N} \frac{\partial \mathrm{C}^n(\mathbf{w}, \mathbf{b})}{\partial w} \\ \frac{\partial \mathrm{L}(\mathbf{w}, \mathbf{b})}{\partial b} &= \sum_{n=1}^{N} \frac{\partial \mathrm{C}^n(\mathbf{w}, \mathbf{b})}{\partial b} \end{aligned} \tag{42}$$