

## Gradient Descent

这里我们再回顾一下gradient descent, 对于网络中的参数 $w_1, w_2, \dots, b_1, b_2, \dots$ , 通过求出相对应的梯度 $\Delta L(\theta)$ , 再根据梯度更新网络结构中的参数

$$\theta^i = \theta^{i-1} - \eta \Delta L(\theta^{i-1})$$

Network parameters  $\theta = \{w_1, w_2, \dots, b_1, b_2, \dots\}$

Starting Parameters  $\theta^0 \longrightarrow \theta^1 \longrightarrow \theta^2 \longrightarrow \dots$

$$\nabla L(\theta) = \begin{bmatrix} \partial L(\theta) / \partial w_1 \\ \partial L(\theta) / \partial w_2 \\ \vdots \\ \partial L(\theta) / \partial b_1 \\ \partial L(\theta) / \partial b_2 \\ \vdots \end{bmatrix}$$

Compute  $\nabla L(\theta^0)$        $\theta^1 = \theta^0 - \eta \nabla L(\theta^0)$   
Compute  $\nabla L(\theta^1)$        $\theta^2 = \theta^1 - \eta \nabla L(\theta^1)$   
Millions of parameters .....  
To compute the gradients efficiently, we use backpropagation.

## Chain Rule

backpropagation的核心思想就是链式法则

**Case 1**       $y = g(x) \quad z = h(y)$

$$\Delta x \rightarrow \Delta y \rightarrow \Delta z \quad \frac{dz}{dx} = \frac{dz}{dy} \frac{dy}{dx}$$

**Case 2**

$$x = g(s) \quad y = h(s) \quad z = k(x, y)$$

$$\begin{array}{ccc} & \Delta x & \\ \Delta s & \nearrow & \searrow \\ & \Delta y & \\ & \nearrow & \searrow \\ & \Delta z & \end{array} \quad \frac{dz}{ds} = \frac{\partial z}{\partial x} \frac{dx}{ds} + \frac{\partial z}{\partial y} \frac{dy}{ds}$$

## Backpropagation

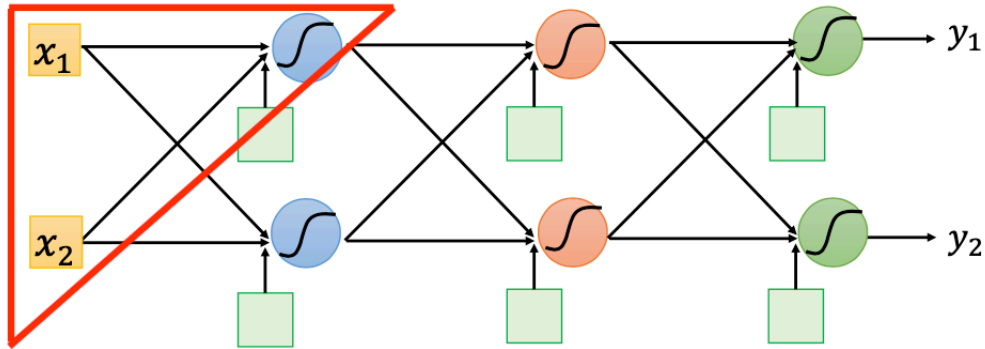
这里我们先定义了一个loss函数,  $l^n(\theta)$ 表示training data中 $y^n$ 和 $\hat{y}^n$ 之间的loss, 这个loss可以通过cross entropy或者MSE计算, 再将所有的loss进行求和, 得到 $L(\theta)$

$$L(\theta) = \sum_{n=1}^N l^n(\theta)$$

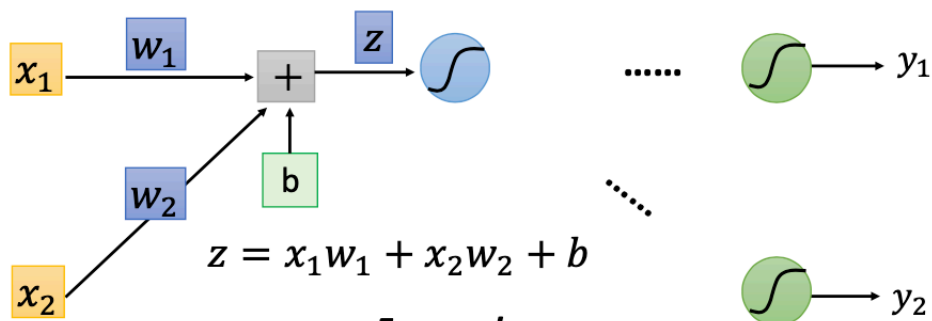
Backpropagation



$$L(\theta) = \sum_{n=1}^N l^n(\theta) \quad \rightarrow \quad \frac{\partial L(\theta)}{\partial w} = \sum_{n=1}^N \frac{\partial l^n(\theta)}{\partial w}$$



对 $w$ 求导, 可得  $\frac{\partial l^n}{\partial w} = \frac{\partial z}{\partial w} \frac{\partial l}{\partial z}$ , 这里我们截取了网络中的部分结构, 神经网络的forward过程计算  $\frac{\partial z}{\partial w}$ , backward过程计算  $\frac{\partial l}{\partial z}$



**Forward pass:**

Compute  $\partial z / \partial w$  for all parameters

$$\frac{\partial l}{\partial w} = ? \quad \frac{\partial z}{\partial w} \frac{\partial l}{\partial z}$$

(Chain rule)

**Backward pass:**

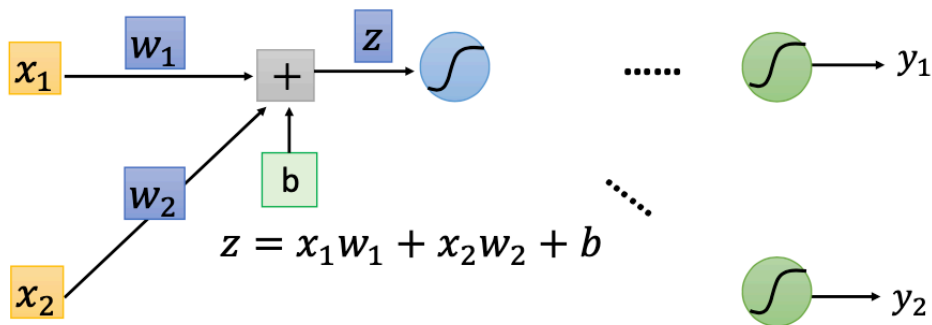
Compute  $\partial l / \partial z$  for all activation function inputs  $z$

## Forward pass

forward过程计算  $\frac{\partial z}{\partial w}$ , 即为权重所对应的上一层神经元的值( $x_1, x_2$ )

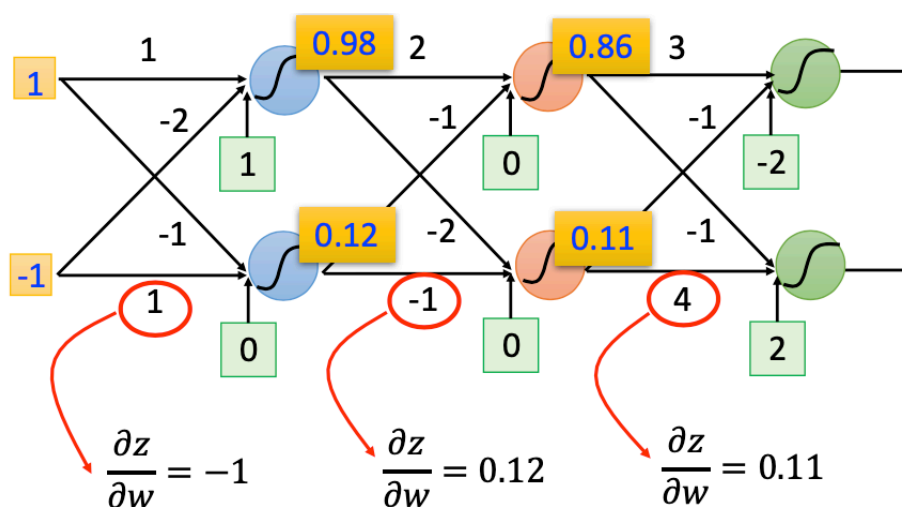
$$z = x_1 w_1 + x_2 w_2 + b$$

$$\frac{\partial z}{\partial w_1} = x_1, \quad \frac{\partial z}{\partial w_2} = x_2$$



$$\left. \begin{aligned} \frac{\partial z}{\partial w_1} &=? \quad x_1 \\ \frac{\partial z}{\partial w_2} &=? \quad x_2 \end{aligned} \right\} \text{The value of the input connected by the weight}$$

对于forward过程，计算出上层神经元的值后，才可以继续计算下一层神经元的梯度  $\frac{\partial z}{\partial w}$ ，



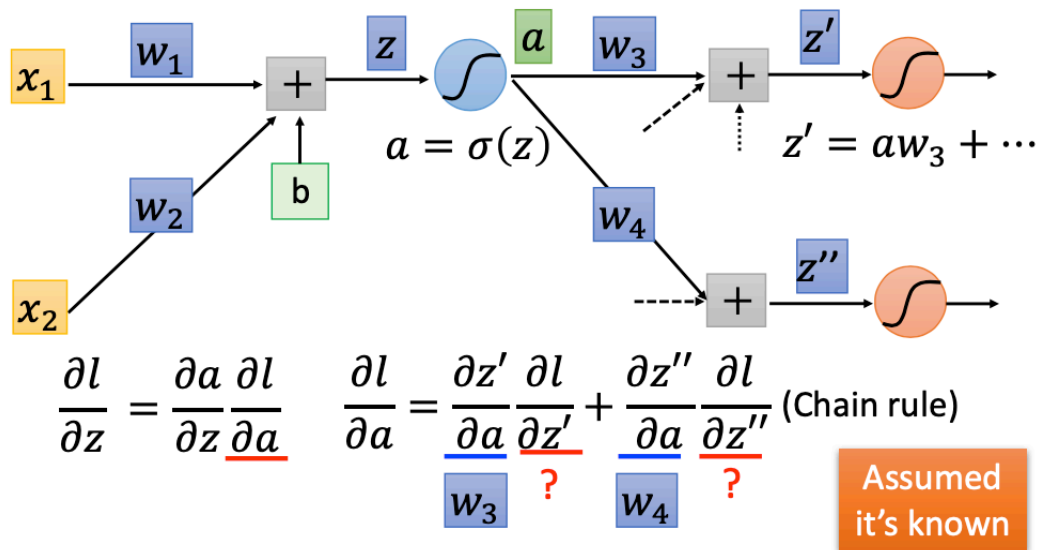
## Backward pass

### 公式推导

backward过程计算  $\frac{\partial l}{\partial z}$ ， $z$ 为激活函数 $\sigma(z)$ 的输入值。这里我们令 $a = \sigma(z)$ ，简化表达式的形式，根据chain rule

$$\frac{\partial l}{\partial z} = \frac{\partial a}{\partial z} \frac{\partial l}{\partial a}$$

其中  $\frac{\partial a}{\partial z} = \sigma'(z)$ ，对激活函数求一阶导数，可以很轻松地表示出来



接下来我们开始  $\frac{\partial l}{\partial a}$  的计算， $a$  会影响  $z'$ ,  $z''$ ，因此可以再次使用链式法则

$$\frac{\partial l}{\partial a} = \frac{\partial z'}{\partial a} \frac{\partial l}{\partial z'} + \frac{\partial z''}{\partial a} \frac{\partial l}{\partial z''}$$

由于  $z' = aw_3 + \dots$ ,  $z'' = aw_4 + \dots$ ，那么我们可以得出

$$\frac{\partial z'}{\partial a} = w_3, \quad \frac{\partial z''}{\partial a} = w_4$$

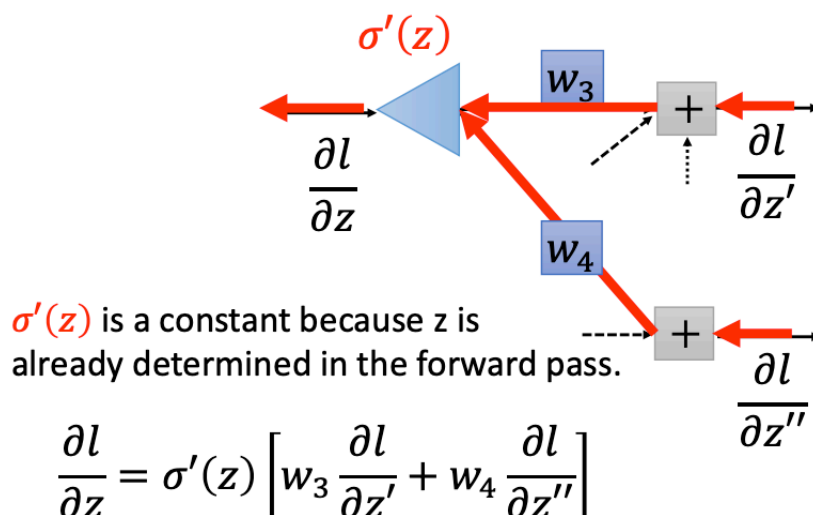
代入原式，可得

$$\frac{\partial l}{\partial z} = \frac{\partial a}{\partial z} \frac{\partial l}{\partial a} = \sigma'(z) \left[ w_3 \frac{\partial l}{\partial z'} + w_4 \frac{\partial l}{\partial z''} \right]$$

对于  $\frac{\partial l}{\partial z'}$ ,  $\frac{\partial l}{\partial z''}$ ，我们假设可以通过某种方式求得他们的值

另一个观点

我们可以从这个图中更加直观地了解backpropagation的过程，这里我们假设有一个神经元（图中的三角形），它不在原来的网络结构中，可以通过  $w_3 \frac{\partial l}{\partial z'} + w_4 \frac{\partial l}{\partial z''}$  来计算，前面再乘上一个放大系数  $\sigma'(z)$ ，这个放大系数的值是根据forward过程计算的，是一个常数，就可以得出  $\frac{\partial l}{\partial z}$  的值



## 两种情况

公式内的其他项都已经计算出来，还有  $\frac{\partial l}{\partial z'}$ ,  $\frac{\partial l}{\partial z''}$  没有得出具体的表达式，此步骤是为了求解  $\frac{\partial l}{\partial z'}$ ,  $\frac{\partial l}{\partial z''}$  的表达式

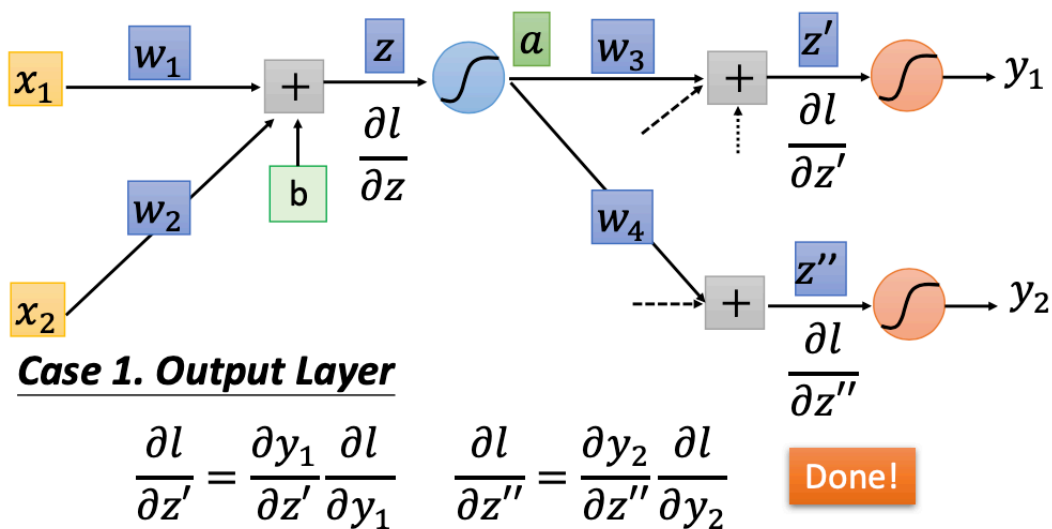
### Case1: Output Layer

为了求出  $\frac{\partial l}{\partial z'}$ ,  $\frac{\partial l}{\partial z''}$ ，这里再次使用了chain rule

$$\frac{\partial l}{\partial z'} = \frac{\partial y_1}{\partial z'} \frac{\partial l}{\partial y_1}$$

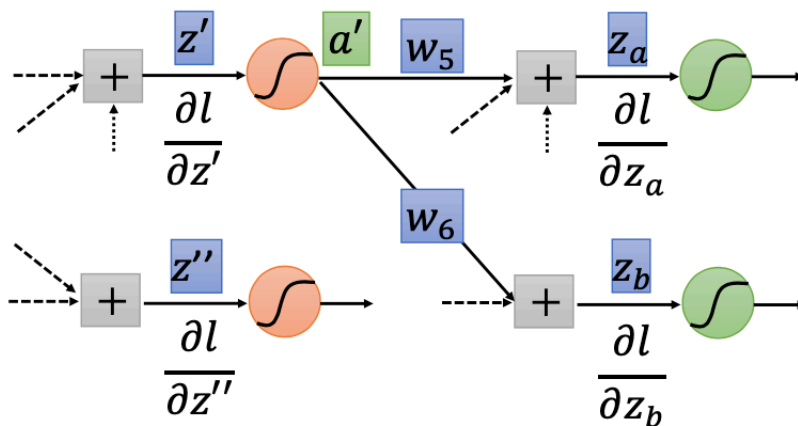
其中  $\frac{\partial y_1}{\partial z'}$  就是激活函数的输出值再对  $z'$  求导；

$\frac{\partial l}{\partial y_1}$  为相对应的loss函数对  $y_1$  求导，这个loss函数可以是cross entropy，也可以是MSE，对于不同的loss函数，计算出来的导数也不同



### Case2: Not output layer

如果现在假设后层不是output layer，而是hidden layer中的其中一层，计算方式就发生了一些小小的变化

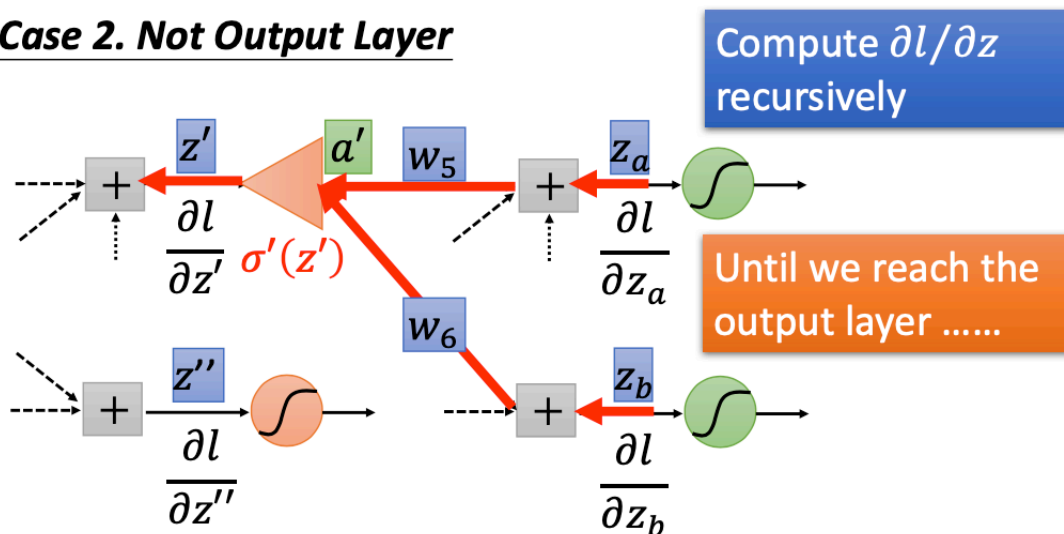


根据Case1的推导，知道了  $\frac{\partial l}{\partial z'}$ ,  $\frac{\partial l}{\partial z''}$  之后，就可以对  $\frac{\partial l}{\partial z'}$  进行求解

同理可得，知道了  $\frac{\partial l}{\partial z_a}$ ,  $\frac{\partial l}{\partial z_b}$ ，就可以对  $\frac{\partial l}{\partial z'}$  求解，如下图所示， $\frac{\partial l}{\partial z_a}$ ,  $\frac{\partial l}{\partial z_b}$  分别乘上对应的权重  $w_5, w_6$ ，前面再乘一个放大系数，就可得出  $\frac{\partial l}{\partial z'}$  的表达式

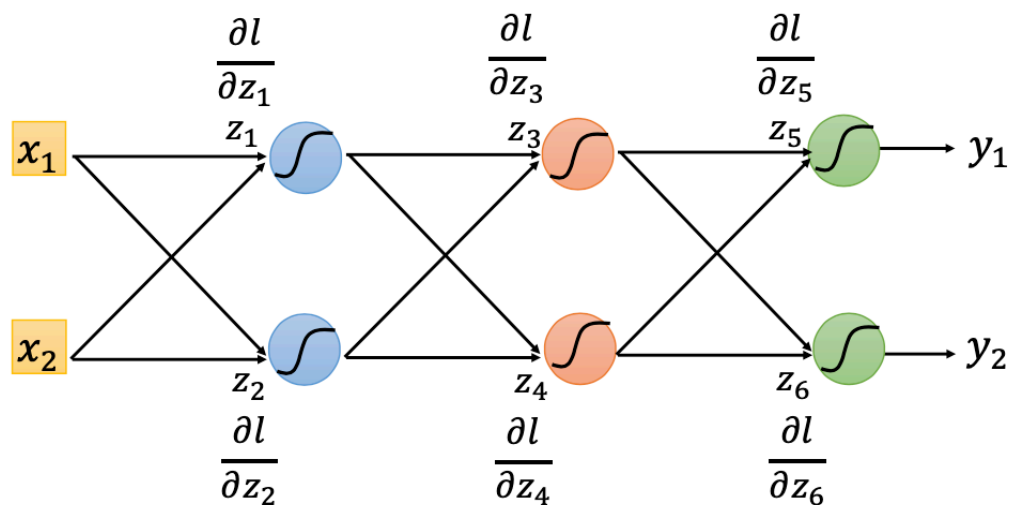
$$\frac{\partial l}{\partial z'} = \sigma'(z') \left[ w_5 \frac{\partial l}{\partial z_a} + w_6 \frac{\partial l}{\partial z_b} \right]$$

## Case 2. Not Output Layer

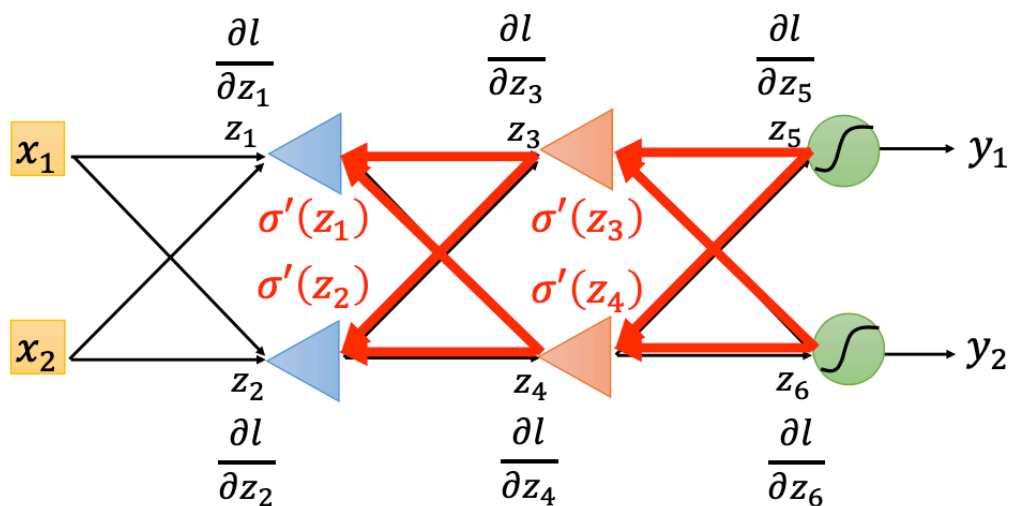


知道了 $z'$ ,  $z''$ 的值后, 就可以根据公式计算 $z$ 的值; 知道了 $z_a$ ,  $z_b$ 之后, 就可以计算 $z'$ 的值; .....一直循环这个步骤, 直到到达output layer为止

下图中有6个neural, 分别是 $z_1, z_2, z_3, z_4, z_5, z_6$ , 为激活函数的输入值, 现在要分别求得 $l$ 对这些函数的偏微分 $\frac{\partial l}{\partial z_i}$ 。按照我们之前的做法, 要求 $\frac{\partial l}{\partial z_1}$ , 就必须要求 $\frac{\partial l}{\partial z_3}, \frac{\partial l}{\partial z_4}$ , 而要求 $\frac{\partial l}{\partial z_3}, \frac{\partial l}{\partial z_4}$ 的值, 就必须要求分别求两次 $\frac{\partial l}{\partial z_5}, \frac{\partial l}{\partial z_6}$ 的值, 计算效率很低



如果我们先计算 $\frac{\partial l}{\partial z_5}, \frac{\partial l}{\partial z_6}$ , 就可以接着计算出 $\frac{\partial l}{\partial z_3}, \frac{\partial l}{\partial z_4}$ 的值, 再计算出 $\frac{\partial l}{\partial z_1}, \frac{\partial l}{\partial z_2}$ 的值, 计算效率可以提高很多



## Summary

神经网络的forward过程计算  $\frac{\partial z}{\partial w}$ ，backward过程计算  $\frac{\partial l}{\partial z}$ ，再相乘，就可以得出loss函数对每个hidden layer神经元的梯度，

$$\frac{\partial l}{\partial w} = \frac{\partial z}{\partial w} \frac{\partial l}{\partial z}$$

代入每次参数更新的公式，就可以得出每次的参数更新结果

$$w^i = w^{i-1} - \eta \Delta L(w^{i-1})$$

