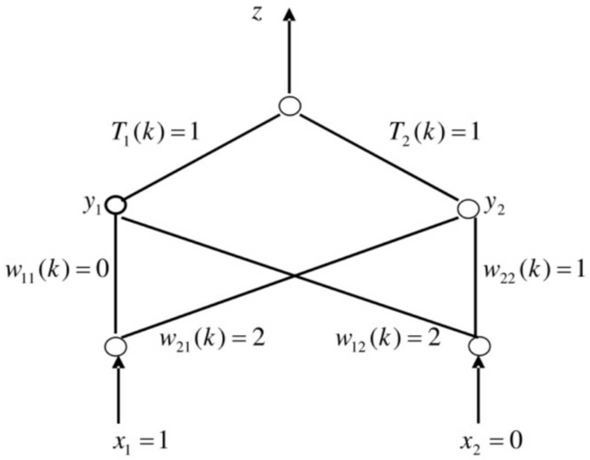
对如下的 BP 神经网络，学习系数$\eta=1$，各点的阈值$\theta=0$。作用函数为：

$$f(x)=\begin{cases}x & x\geq 1\\ 1 & x<1\end{cases}。$$

输入样本$x_1=1,x_2=0$，输出节点z的期望输出为 1，对于第k次学习得到的权值分别为$w_{11}(k)=0,w_{12}(k)=2,w_{21}(k)=2,w_{22}(k)=1,T_1(k)=1,T_2(k)=1$，求第$k$次和$k+1$次学习得到的输出节点值$z(k)$和$z(k+1)$（写出计算公式和计算过程）。



计算如下：

1． 第 k 次训练的正向过程如下：

$$y_i=f(\sum_j w_{ij}x_j-\theta_i)=f(net_i)$$

$$y_1=f(\sum_{j=1}^{2}w_{1j}x_j-\theta)=f(net_1)=f(0\times 1+2\times 0)=f(0)=1$$

$$y_2 = f(\sum_{j=1}^{2} w_{2j} x_j) = f(net_2) = f(2\times1+1\times0) = f(2) = 2$$

$$O_l = f(\sum_i T_{li} y_i - \theta_l) = f(net_l)$$

$$z = f(\sum_{i=1}^{2} T_i y_i) = f(net_l) = f(1\times1+1\times2) = f(3) = 3$$

$$E = \frac{1}{2}(1-3)^2 = 2$$

2．第 k 次训练的反向过程如下：

$$\delta_l = (z^{'} - z) f'(net_l) = (1-3)\times f^{'}(3) = -2\times1 = -2$$

$$\delta_i' = f'(net_i)\sum_l \delta_l T_{li}$$

$$\delta_1' = f'(net_1)\delta_l T_1 = f'(0)\times(-2)\times1 = 0\times(-2)\times1 = 0$$

$$\delta_2' = f'(net_2)\delta_l T_2 = f'(2)\times(-2)\times1 = 1\times(-2)\times1 = -2$$

$$T_1(k+1) = T_1(k) + \Delta T_1 = T_1(k) + \eta\delta_l y_1 = 1+1\times(-2)\times1 = -1$$

$$T_2(k+1) = T_2(k) + \Delta T_2 = T_2(k) + \eta\delta_l y_2 = 1+1\times(-2)\times2 = -3$$

$$W_{11}(k+1) = W_{11}(k) + \Delta W_{11}$$
$$= W_{11}(k) + \eta\delta_1' x_1 = 0+1\times0\times1 = 0$$

$$W_{12}(k+1)=W_{12}(k)+\Delta W_{12}$$
$$=W_{12}(k)+\eta\delta_1'x_2=2+1\times0\times0=2$$

$$W_{21}(k+1)=W_{21}(k)+\Delta W_{21}$$
$$=W_{21}(k)+\eta\delta_2'x_1=2+1\times(-2)\times1=0$$

$$W_{22}(k+1)=W_{22}(k)+\Delta W_{22}$$
$$=W_{22}(k)+\eta\delta_2'x_2=1+1\times(-2)\times0=1$$

3．第 $k+1$ 次学习的正向过程如下：

$$y_i=f(\sum_j w_{ij}x_j-\theta_i)=f(net_i)$$

$$y_1=f(\sum_{j=1}^{2}w_{1j}x_j)=f(0\times1+2\times0)=f(0)=1$$

$$y_2=f(\sum_{j=1}^{2}w_{2j}x_j)=f(0\times1+1\times0)=f(0)=1$$

$$O_l=f(\sum_i T_{li}y_i-\theta_l)=f(net_l)$$
$$z=f(\sum_{i=1}^{2}T_i y_i)=f(1\times(-1)+1\times(-3))=f(-4)=1$$

$$E=\frac{1}{2}(1-1)^2=0$$

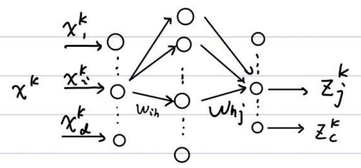
5. 给定 d 维空间中的 n 个样本 $x_i, i = 1, \ldots, n$，已知它们分别属于 c 个不同的类别。现在拟利用这些样本来训练一个三层前向神经网络（即包含一个输入层，一个隐含层和一个输出层）。假定采用如下平方损失函数作为该网络的目标函数：

$$E(w) = \sum_{k=1}^{n} \sum_{j=1}^{c} (t_j^k - z_j^k)^2$$

这里，t_j^k表示样本x_k在输出层第 j 个结点的期望输出值（即该值已知，由样本x_k的类别标签来决定），z_j^k表示样本x_k在输出层第 j 个结点的实际输出值（即通过网络计算所得的输出值），w 记录所有待学习的网络参数，包含输入层至隐含层的各个权重w_{ih}以及隐层至输出层的各个权重w_{hj}。请结合上述三层前向神经网络，分别写出w_{ih}和w_{hj}的更新公式。（学习率为 1，此网络不包含激活函数）（15'）

5. 思路：题目太抽象，可以画个图辅助理解。另外要注意辨别 n 和 d 分别都是什么。

$$x^1 = [x_1^1, x_2^1, \cdots x_d^1]^T$$
$$\vdots$$
$$x^i = [x_1^i, x_2^i, \cdots x_d^i]^T$$
$$\vdots$$
$$x^n = [x_1^n, x_2^n, \cdots x_d^n]^T$$



设隐层共 H 个神经元

$$\frac{\partial E}{\partial w_{hj}} = \frac{\partial \sum_{k=1}^{n}\sum_{j=1}^{c}(t_j^k - z_j^k)^2}{\partial w_{hj}} = \sum_{k=1}^{n}\sum_{j=1}^{c}\left[-2(t_j^k - z_j^k)\right] \cdot \frac{\partial z_j^k}{\partial w_{hj}}$$

$$= \sum_{k=1}^{n}\sum_{j=1}^{c}\left[-2(t_j^k - z_j^k)\right] \frac{\partial \sum_{h=1}^{H} w_{hj}\left(\sum_{i=1}^{d} x_i^k w_{ih}\right)}{\partial w_{hj}}$$

$$= \sum_{k=1}^{n}\sum_{j=1}^{c}\left[-2(t_j^k - z_j^k)\right] \frac{\partial\left(w_{1j}\sum_{i=1}^{d} x_i^k w_{i1} + w_{2j}\sum_{i=1}^{d} x_i^k w_{i2} + \cdots + w_{hj}\sum_{i=1}^{d} x_i^k w_{ih} + \cdots\right)}{\partial w_{hj}}$$

$$= -2\sum_{k=1}^{n}\sum_{j=1}^{c}(t_j^k - z_j^k) \cdot \sum_{i=1}^{d} x_i^k w_{ih}$$

$$\frac{\partial E}{\partial w_{ih}} = \sum_{k=1}^{n}\sum_{j=1}^{c}\left[-2(t_j^k - z_j^k)\right] \frac{\partial \sum_{h=1}^{H} w_{hj}\left(\sum_{i=1}^{d} x_i^k w_{ih}\right)}{\partial w_{ih}}$$

$$= \sum_{k=1}^{n}\sum_{j=1}^{c}\left[-2(t_j^k - z_j^k)\right] \frac{\partial\left(w_{1j}\sum_{i=1}^{d} x_i^k w_{i1} + w_{2j}\sum_{i=1}^{d} x_i^k w_{i2} + \cdots + w_{hj}\sum_{i=1}^{d} x_i^k w_{ih} + \cdots\right)}{\partial w_{ih}}$$

$$= \sum_{k=1}^{n}\sum_{j=1}^{c}\left[-2(t_j^k - z_j^k)\right] \frac{\partial w_{hj}\sum_{i=1}^{d} x_i^k w_{ih}}{\partial w_{ih}}$$

$$= \sum_{k=1}^{n}\sum_{j=1}^{c}\left\{-2(t_j^k - z_j^k)\right\} \cdot \frac{\partial w_{hj}\left(x_1^k w_{1h} + x_2^k w_{2h} + \cdots + x_i^k w_{ih} + \cdots\right)}{\partial w_{ih}}$$

$$= \sum_{k=1}^{n}\sum_{j=1}^{c}\left\{-2(t_j^k - z_j^k)\right\} \cdot w_{hj} \cdot x_i^k$$

$$\begin{cases} w_{hj} = w_{hj} - \dfrac{\partial E}{\partial w_{hj}} \\ w_{ih} = w_{ih} - \dfrac{\partial E}{\partial w_{ih}} \end{cases}$$

6. 课本 P75

深度前馈网络

推导

sigmoid函数 $f'(z)=f(z)(1-f(z))$

tanh函数 $f'(z)=1-(f(z))^2$

假设神经网络(NN)总共有 L 层

当第 $L-1$ 层时，权重求导

$$\frac{\partial J}{\partial W_{ij}^{L-1}}=\frac{\partial J}{\partial z_i^L}\frac{\partial z_i^L}{\partial W_{ij}^{L-1}}=\delta_i^L a_j^{L-1} \qquad \delta_i^L=\frac{\partial J}{\partial z_i^L}=\frac{\partial}{\partial z_i^L}\sum_{i=1}^{s_L}\frac{1}{2}||y_i-f(z_i^L)||^2=-(y_i-f(z_i^L))f'(z_i^L)$$

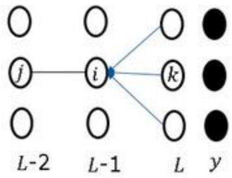
当第 $L-2$ 层时，权重求导

$$\frac{\partial J}{\partial W_{ij}^{L-2}}=\frac{\partial J}{\partial z_i^{L-1}}\frac{\partial z_i^{L-1}}{\partial W_{ij}^{L-2}}=\delta_i^{L-1}a_j^{L-2}$$

$$\delta_i^{L-1}=\frac{\partial J}{\partial z_i^{L-1}}=\frac{\partial}{\partial z_i^{L-1}}\sum_{k=1}^{s_L}\frac{1}{2}||y_k-f(z_k^L)||^2=\sum_{k=1}^{s_L}-\left(y_k-f(z_k^L)\right)f'(z_k^L)\frac{\partial z_k^L}{\partial z_i^{L-1}}$$

$$=\sum_{k=1}^{s_L}\delta_k^L \bullet w_{ki}^{L-1}f'(z_i^{L-1})$$

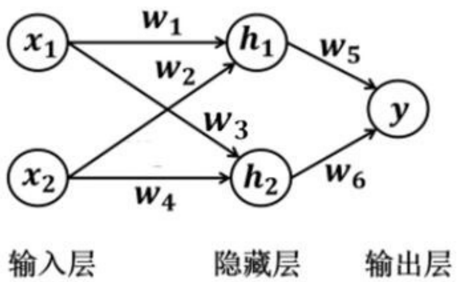
$$=(\sum_{k=1}^{s_L}\delta_k^L w_{ki}^{L-1})f'(z_i^{L-1})$$



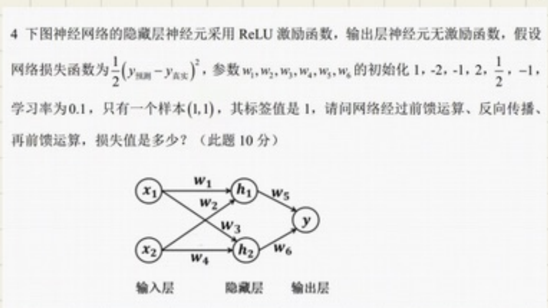
$$z_k^L=W_{ki}^{L-1}a_i^{L-1}+b_k^{L-1}$$

$$a_i^{L-1}=f(z_i^{L-1})$$

4 下图神经网络的隐藏层神经元采用 ReLU 激励函数，输出层神经元无激励函数，假设网络损失函数为 $\frac{1}{2}\left(y_{预测}-y_{真实}\right)^2$，参数 w_1,w_2,w_3,w_4,w_5,w_6 的初始化 1，-2，-1，2，$\frac{1}{2}$，-1，学习率为0.1，只有一个样本(1,1)，其标签值是 1，请问网络经过前馈运算、反向传播、再前馈运算，损失值是多少？（此题 10 分）



手动计算BP神经网络



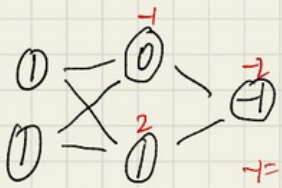
4 下图神经网络的隐藏层神经元采用 ReLU 激励函数，输出层神经元无激励函数，假设网络损失函数为 $\frac{1}{2}\left(y_{预测}-y_{真实}\right)^2$，参数 w_1,w_2,w_3,w_4,w_5,w_6 的初始化 1，-2，-1，2，$\frac{1}{2}$，-1，学习率为0.1，只有一个样本(1,1)，其标签值是 1，请问网络经过前馈运算、反向传播、再前馈运算，损失值是多少？（此题 10 分）

首先进行前馈传播:

$h_1=relu(x_1w_1+x_2w_2)=relu(1-2)=0$

$h_2=relu(x_1w_3+x_2w_4)=1$

$J_1=\frac{1}{2}(\hat{y}_1-y)^2=\frac{1}{2}(-1-1)^2=2$



$-1=W_5\times(-2)$

$2=W_6\times(-2)$

黑字为输出值

红字为残差

接下来进行反馈传播，此处的 ΔW_i 为后层残差 × 前层输出值 $\times f'(z_i)$

更新 W_i

$W_i'=W_i-\text{learning\_rate}\times\Delta W_i$

$\Delta W_1=(-1)\times1\times0=0$

$\Delta W_2=(-1)\times1\times0=0$

$\Delta W_3=2\times1\times1=2$

$\Delta W_4=2-0.2=1.8$

$\Delta W_5=(-2)\times0\times1=0$ (output无激励)

$W_6=(-2)\times1\times1=(-2)$

$W_1'=1 \quad W_2'=-2 \quad W_3'=-1.2$

$W_4'=1.8 \quad W_5'=0.5 \quad W_6'=-0.8$

再次前向传播

$h_1'=relu(x_1w_1'+x_2w_2')=relu(-1)=0$

$h_2'=relu(x_1w_3'+x_2w_4')=relu(-1.2+1.8)=relu(0.6)=0.6$

$\hat{y}_2=h_1'w_5'+h_2'w_6'=0+0.6\times(-0.8)=-0.48$

$\Rightarrow J_2=\frac{1}{2}(\hat{y}_2-y)^2=\frac{1369}{1250}=1.0952$

4.2 如采用 tanh()作为多层感知器中隐节点的激活函数，试推导 BP 算法，并讨论为什么多层感知器一般不常用 tanh()作为激活函数。

参考答案:

（1）证明：（延续课件表述）

$$for\ node\ j,\quad net_j = \sum_i w_{ij}O_i,\quad O_j = f(net_j)$$

这里目标函数设为平方误差

$$E = \frac{1}{2}\sum_j (y_j - \hat{y}_j)^2$$

误差梯度

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial net_j}\frac{\partial net_j}{\partial w_{ij}} = \delta_j O_i,\quad \delta_j = \frac{\partial E}{\partial net_j}$$

对于输出节点

$$O_j = \hat{y}_j$$

$$\delta_j = \frac{\partial E}{\partial \hat{y}_j}\frac{\partial \hat{y}_j}{\partial net_j} = -(y_j - \hat{y}_j)f'(net_j)$$

又因为这里

$$f(x) = \tanh x$$

$$f'(x) == \frac{4e^{-2x}}{(1+e^{-2x})^2} = \frac{1}{(\frac{e^x+e^{-x}}{2})^2} = \frac{1}{\cosh^2 x}$$

则

$$\delta_j = \frac{\partial E}{\partial \hat{y}_j}\frac{\partial \hat{y}_j}{\partial net_j} = -(y_j - \hat{y}_j)f'(net_j) = -(y_j - \hat{y}_j)\frac{1}{\cosh^2(net_j)}$$

对于隐层节点

$$\delta_j = \frac{\partial E}{\partial net_j} = \sum_k \frac{\partial E}{\partial net_k}\frac{\partial net_k}{\partial O_j}\frac{\partial O_j}{\partial net_j} = \sum_k \delta_k w_{jk} f'(net_j) = \sum_k \delta_k w_{jk}\frac{1}{\cosh^2(net_j)}$$

权值学习

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$$
$$\Delta w_{ij}(t) = -\eta\delta_j(t)O_i(t)$$

其中，η为学习步长

（2）tanh()作为激活函数的缺点

多层感知器一般不常用tanh()作为激活函数主要有两个原因，第一可能出现梯度消失的问题：由于激活函数在饱和区导数接近 0，导致在后向传递求导的链式法则下，小数相乘结果接近于 0，故不适合深层网络；第二个原因是tanh()涉及到幂运算，速度较慢。