BP 算法

- **(1)** 网络的构成

神经元的网络输入：

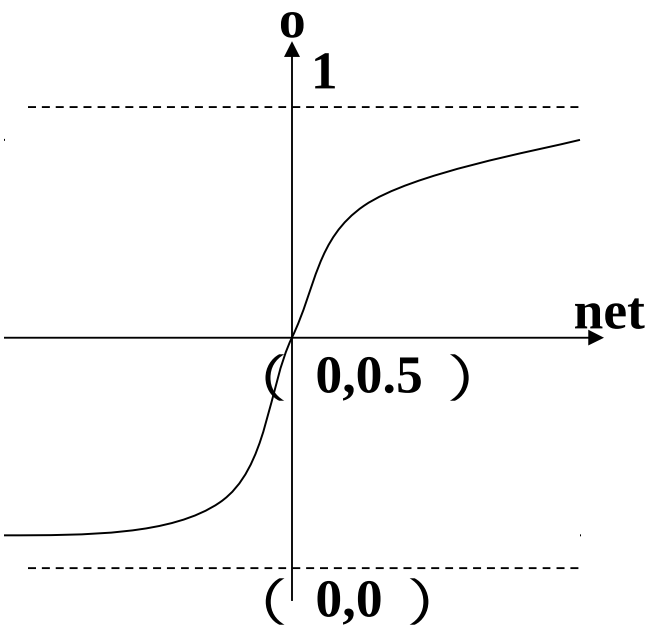
$$\mathbf{net}_i=\mathbf{x}_1\mathbf{w}_{1i}+\mathbf{x}_2\mathbf{w}_{2i}+\ldots+\mathbf{x}_n\mathbf{w}_{ni}$$

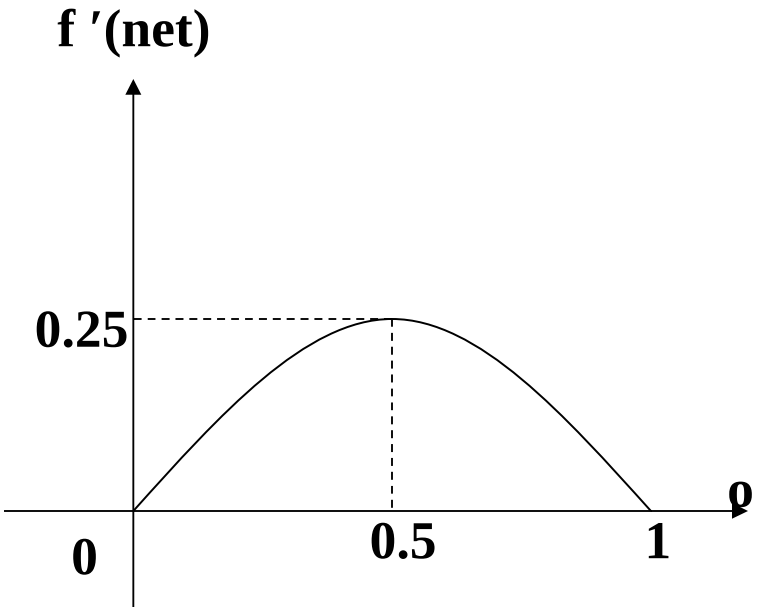
神经元的输出：$o=f(net)=\dfrac{1}{1+e^{-net}}$

$$f'(net)=-\frac{1}{(1+e^{-net})^2}(-e^{-net})=o-o^2=o(1-o)$$

输出函数分析

$$o = \frac{1}{1 + e^{-net}}$$

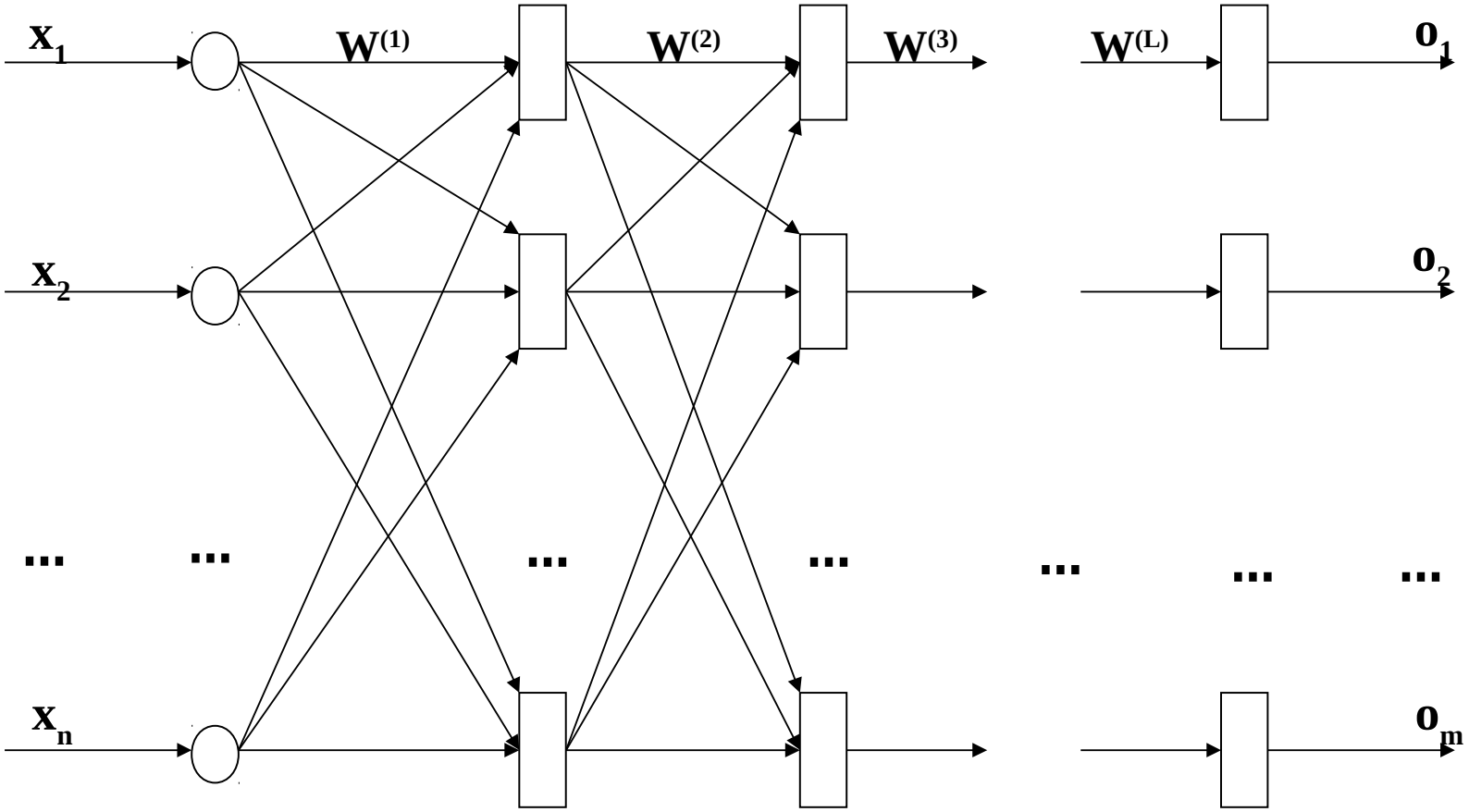




- 应该将 **net** 的值尽量控制在收敛比较快的范围内
- 可以用其它的函数作为激活函数，只要该函数是处处可导的

BP 网络中是基于最小平方误差准则和梯度下降优化方法来确定权值调整法则。

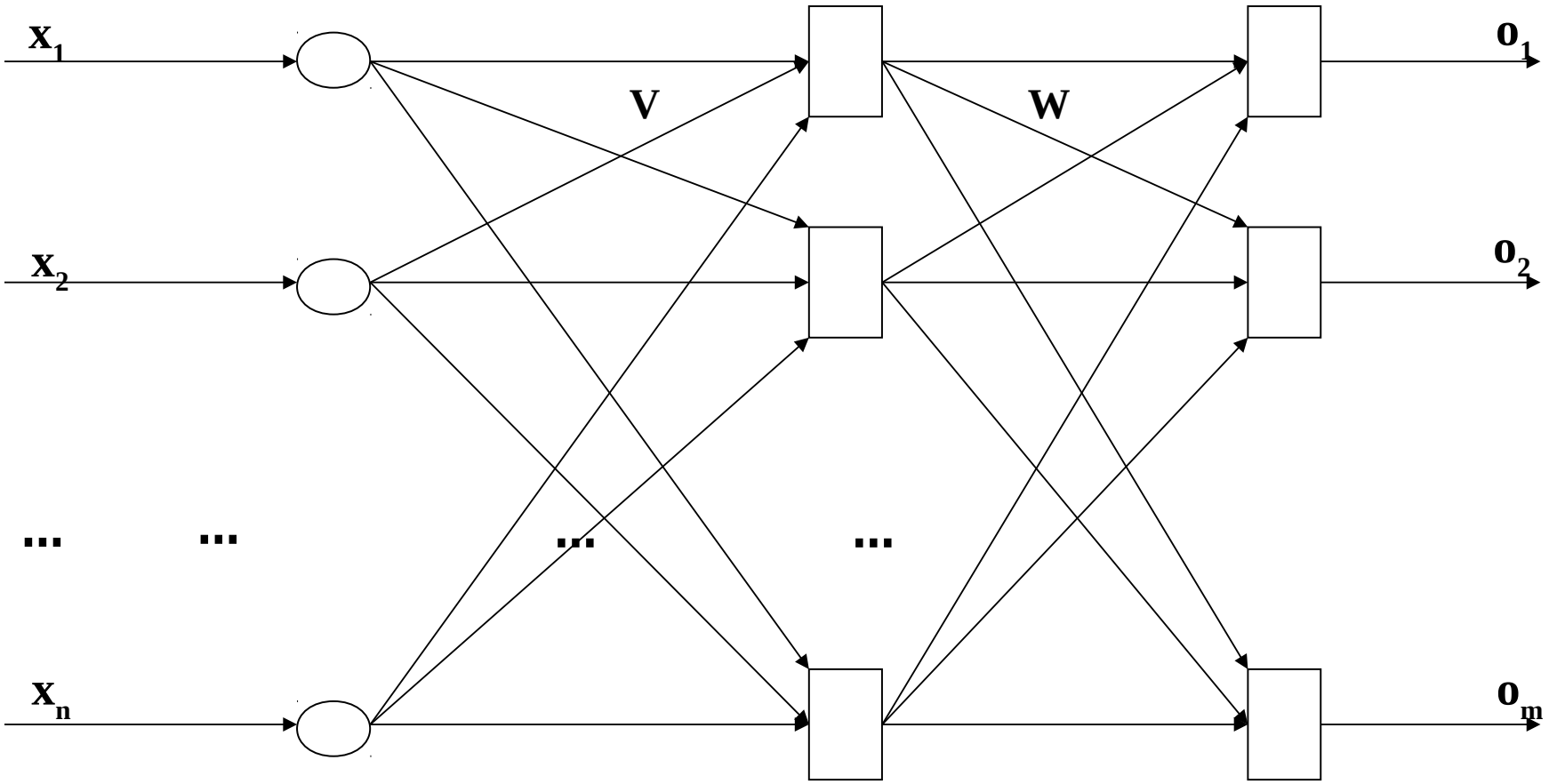
网络的拓扑结构



网络的拓扑结构

1. BP 网的结构
2. 输入向量、输出向量的维数、网络隐藏层的层数和各个隐藏层神经元的个数的决定
3. 实验：增加隐藏层的层数和隐藏层神经元个数不一定总能够提高网络精度和表达能力。
4. BP 网一般都选用二级网络。

三层（两级）神经网络模型



3 算法的理论基础

- 基本假设
  - 网络含有 L 层
  - 联接矩阵： $W^{(1)}$ ， $W^{(2)}$ ， …， $W^{(L)}$
  - 第 k 层的神经元： H_k 个
  - 自变量数： $n*H_1+H_1*H_2+H_2*H_3+\ldots+H_L*m$
  - 样本集： $S=\{(X_1,Y_1),(X_2,Y_2),\ldots,(X_s,Y_s)\}$

用 **E** 代表 E_P，用（ **X**， **Y** ）代表（ X_P， Y_P ）

$X=(x_1, x_2, \cdots, x_n)$

$Y=(y_1, y_2, \cdots, y_m)$

该样本对应的实际输出为

$O=(o_1, o_2, \cdots, o_m)$

误差测度 $E=\sum_{p=1}^{s}E_p$

- 用理想输出与实际输出的方差作为相应的误差测度

$$E=\frac{1}{2}\sum_{k=1}^{m}(y_k-o_k)^2$$

理想输出（也就是训练数据中的输出值）

神经网络的实际输出

Oi= 前层神经元的输出

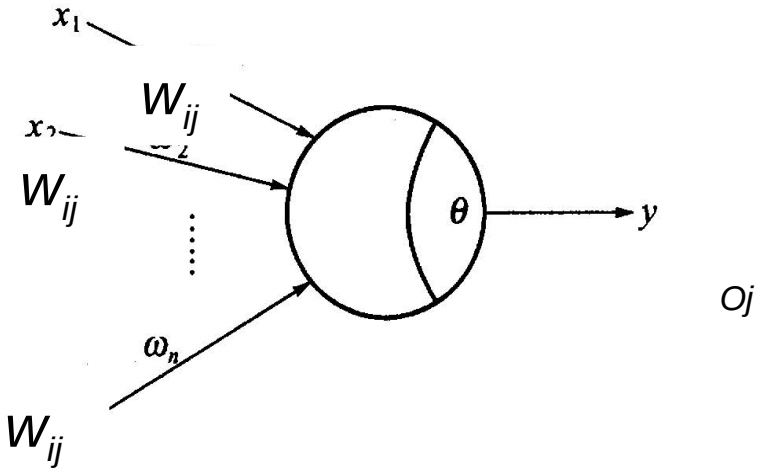


图 4－2　M－P 神经元模型

第 K 层 (有 H_k 个神经元) 隐含层和输出层神经元的操作

输入整合：

$$net_j = \sum_{i=0}^{H_k} w_{ij} o_i$$

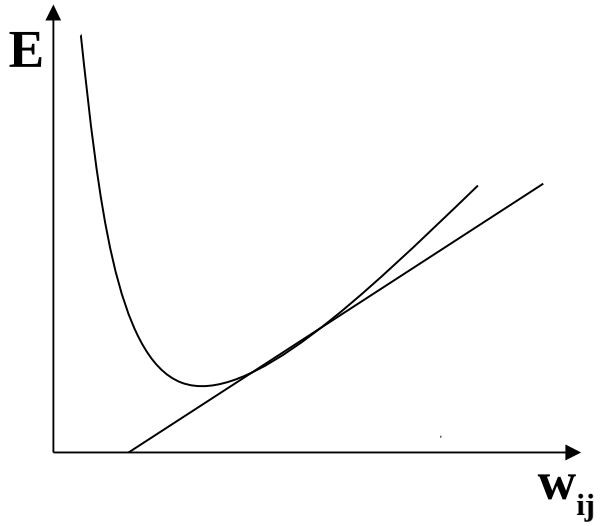
神经元 j 的输出 (激活)

$$o_j = f(net_j) = \frac{1}{1+e^{-net_j}}$$

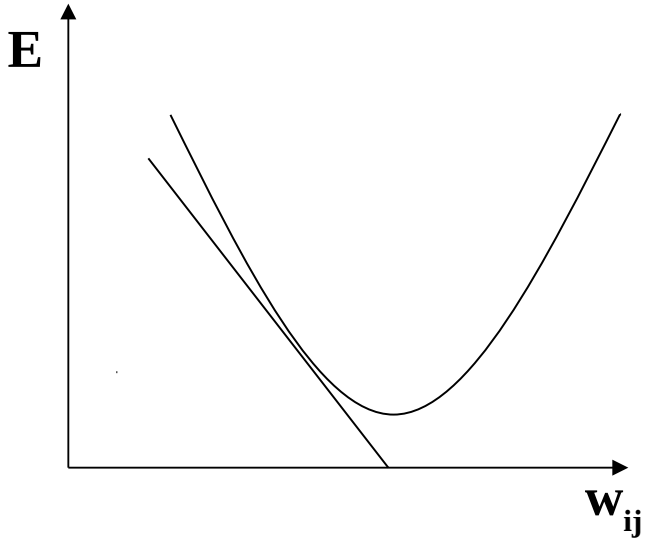
$$f'(net_j) = -\frac{1}{(1+e^{-net_j})^2}(-e^{-net_j}) = o_j - o_j^{\,2} = o_j(1-o_j)$$

最速下降法，要求 E 的极小点

取 $\Delta w_{ij} \propto -\frac{\partial E}{\partial w_{ij}}$



$\frac{\partial E}{\partial w_{ij}}>0$，此时 $\Delta w_{ij}<0$



$\frac{\partial E}{\partial w_{ij}}<0$, 此时 $\Delta w_{ij}>0$

最速下降法，要求 E 的极小点

$$-\frac{\partial E}{\partial w_{ij}} = -\frac{\partial E}{\partial net_j} \cdot \frac{\partial net_j}{\partial w_{ij}}$$

W_{ij}= 表示第 k-1 层神经元 i 与第 k 层第 j 个神经元 j 之间的连接权重。

net_j 表示第 k 层神经元 j 的总输入

O_i 表示第 k-1 层神经元的输出

而其中的

$$net_j = \sum_{i=0}^{H_k} w_{ij} o_i$$

所以，

$$\frac{\partial net_j}{\partial w_{ij}} = \frac{\partial\left(\sum_i w_{ij} o_i\right)}{\partial w_{ij}} = o_i$$

最速下降法，要求 E 的极小点

$$-\frac{\partial E}{\partial w_{ij}} = -\frac{\partial E}{\partial net_j}\cdot\frac{\partial net_j}{\partial w_{ij}}$$

$$= -\frac{\partial E}{\partial net_j}\cdot\frac{\partial\left(\sum_i w_{ij}o_i\right)}{\partial w_{ij}}$$

$$= -\frac{\partial E}{\partial net_j}\cdot o_i$$

每次训练时，为已知

令 $\delta_j = -\frac{\partial E}{\partial net_j}$

要确定 ΔW_{ij}

关键是确定此值

所以 $\Delta w_{ij}=\alpha\delta_j o_i$

α 为学习率

问题转化成如何求得网络中各神经元 j 的δ_j

$$\delta_j = -\frac{\partial E}{\partial net_j}$$

情况 1： AN_j 为输出层神经元

$o_j=f(net_j)$

容易得到

$$\frac{\partial o_j}{\partial net_j}=f'(net_j)$$

从而 $$\delta_j=-\frac{\partial E}{\partial net_j}$$

$$=-\frac{\partial E}{\partial o_j}\cdot\frac{\partial o_j}{\partial net_j}$$

$$=-\frac{\partial E}{\partial o_j}\cdot f'(net_j)$$

$\delta_j=-\frac{\partial E}{\partial net_j}$ 的问题转化成确定 $-\frac{\partial E}{\partial o_j}$

AN_j 为输出层神经元

$$-\frac{\partial E}{\partial o_j}=-\frac{\partial\left(\frac{1}{2}\sum_{k=1}^{m}\left(y_k-o_k\right)^2\right)}{\partial o_j}$$

$$=-(\frac{1}{2}\frac{\partial\left(y_j-o_j\right)^2}{\partial o_j})$$

$$=-(-\frac{2}{2}(y_j-o_j))$$

$$=(y_j-o_j)$$

AN_j 为输出层神经元

所以，$\delta_j = (y_j - o_j) f'(net_j)$

故，当 AN_j 为输出层的神经元时，它对应的联接权 w_{ij} 应该按照下列公式进行调整：

$$w_{ij} = w_{ij} + \alpha \delta_j o_i$$

$$= w_{ij} + \alpha f'(net_j)(y_j - o_j) o_i$$

$$= w_{ij} + \alpha \cdot O_j \cdot (1 - O_j) \cdot (y_j - O_j) \cdot O_i$$

情况 2： AN_j 为隐藏层神经元

$$\delta_j = -\frac{\partial E}{\partial net_j}$$

$$= -\frac{\partial E}{\partial o_j} \cdot \frac{\partial o_j}{\partial net_j}$$

$$\frac{\partial o_j}{\partial net_j} = f'(net_j)$$

$$\delta_j = -\frac{\partial E}{\partial o_j} \cdot f'(net_j)$$

而

$$E = \frac{1}{2}\sum_{k=1}^{m}(y_k - o_k)^2$$

函数

其中，O_k 是 O_j（隐含层神经元 j 的输出）的函数。

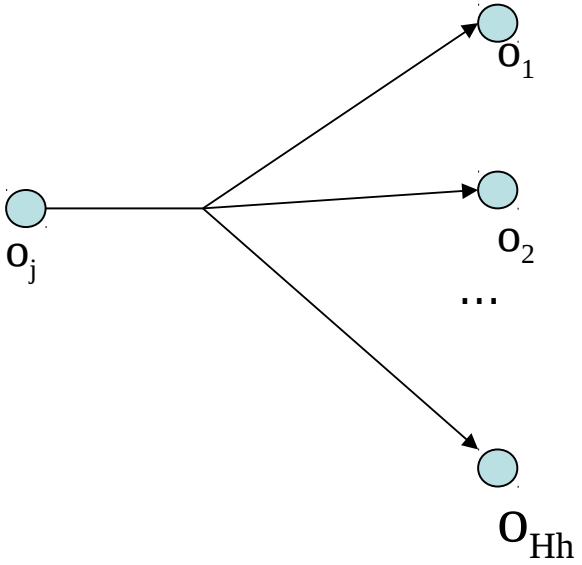
AN_j 为隐藏层神经元

j 神经元的输出为 K 层的输入，即 K 为 j 神经元所在层的下一层。

$\mathbf{net_k}$ 是 $\mathbf{o_j}$ 下一级的神经元的网络输入

$$\mathbf{net_k} = \sum_{j=1}^{H_h} w_{jk} o_j$$

$$\frac{\partial E}{\partial o_j} = \sum_{k=1}^{H_k} \left(\frac{\partial E}{\partial net_k} \cdot \frac{\partial net_k}{\partial o_j} \right)$$



$$\frac{\partial net_k}{\partial o_j} = \frac{\partial \left(\sum_{j=1}^{H_h} w_{jk} o_j \right)}{\partial o_j} = w_{jk}$$

AN_j 为隐藏层神经元

于是，
$$\frac{\partial E}{\partial o_j}=\sum_{k=1}^{H_k}\left(\frac{\partial E}{\partial net_k}\cdot\frac{\partial net_k}{\partial o_j}\right)=\sum_{k=1}^{H_k}\left(\frac{\partial E}{\partial net_k}\cdot w_{jk}\right)$$

而
$$\delta_k=-\frac{\partial E}{\partial net_k}$$

所以，
$$\frac{\partial E}{\partial o_j}=-\sum_{k=1}^{H_k}\delta_k w_{jk}$$

AN_j 为隐藏层神经元

$$\delta_j = -\frac{\partial E}{\partial o_j} \cdot f'(net_j) = -\left(-\sum_{k=1}^{H_k} \delta_k w_{jk}\right) \cdot f'(net_j)$$

或

$$\delta_j = \left(\sum_{k=1}^{H_k} \delta_k w_{jk}\right) \cdot f'(net_j)$$

AN_j 为隐藏层神经元

$$\Delta w_{ij} = \alpha \left(\sum_{k=1}^{H_k} \delta_k w_{jk} \right) \cdot f'(net_j) \cdot o_i$$

于是

$$w_{ij} = w_{ij} + \alpha \left(\sum_{k=1}^{H_k} \delta_k w_{jk} \right) \cdot f'(net_j) \cdot o_i$$

$$= w_{ij} + \alpha \cdot \left(\sum_{k=1}^{H_k} \delta_k w_{jk} \right) \cdot O_j \cdot (1 - O_j) \cdot O_i$$

也就是说，对于隐含层 H 的神经元 j 来说，将所有与之相连的输出神经元（或者是下一层的隐含层的神经元）k 输出端的误差乘以对应的权值 w_{jk}，并求和，作为隐含层神经元 j 的输出误差。所以这个过程也成为“误差反向传播”。