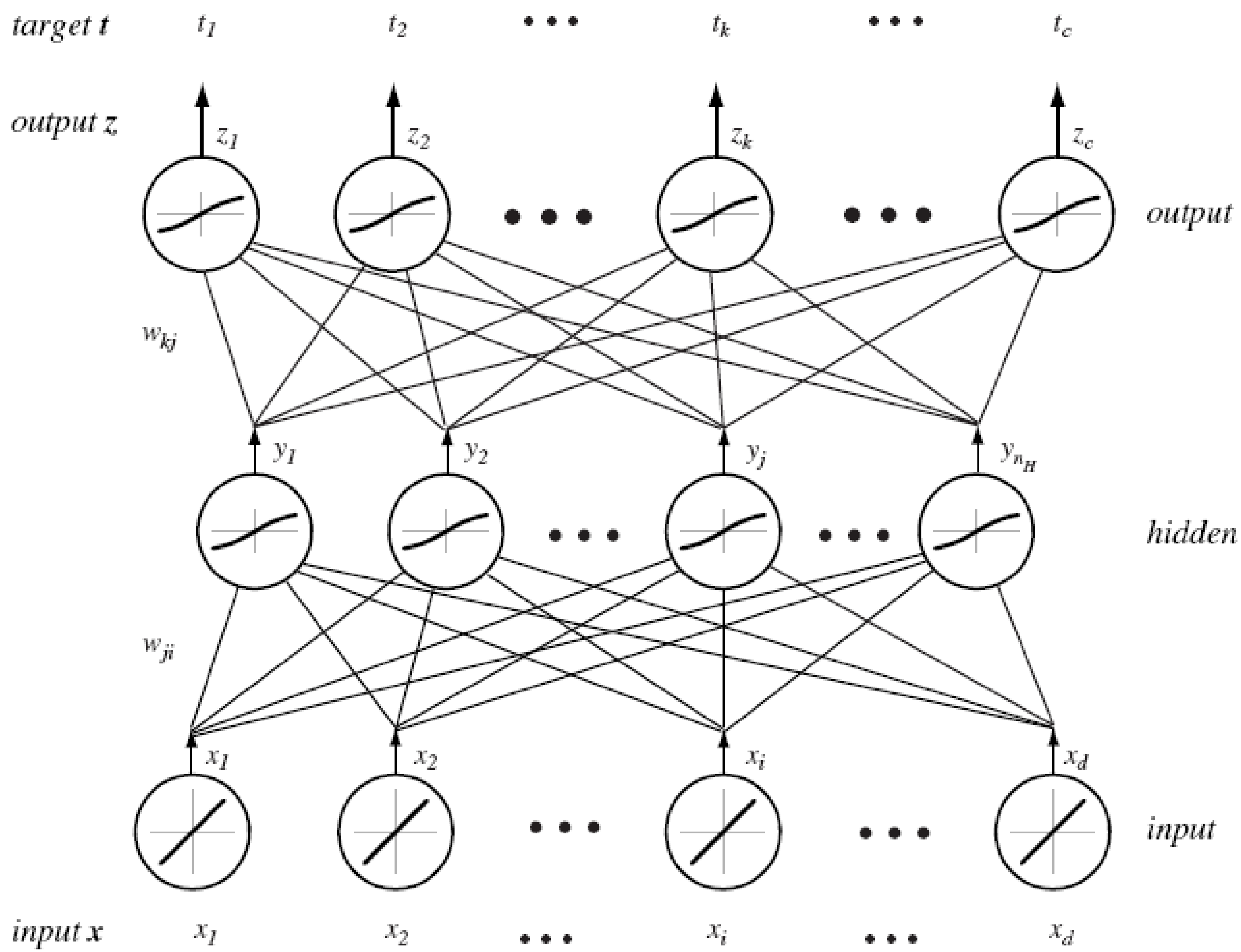
模式分类：杜达著

$$net_j = \sum_{i=1}^{d} \omega_{ji} x_i + \omega_{j0} = \omega_j^t x \qquad net_k = \sum_{j=1}^{n_H} \omega_{kj} y_j + \omega_{k0} = \omega_k^t y$$

$$x = (x_0, x_1, \ldots, x_d)^t, x_0 = 1 \qquad \omega_j = (\omega_{j0}, \omega_{j1}, \ldots, \omega_{jd})^t$$



$$x = (x_1, \ldots, x_d)^T$$

$$t = (t_1, \ldots, t_c)^T$$

$$J = \frac{1}{2} \sum_{k=1}^{c} (t_k - z_k)^2$$

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

$$y_j = f(net_j)$$

$$z_k = f(net_k)$$

$$net_j = \sum_{i=1}^{d} \omega_{ji} x_i + \omega_{j0}$$

$$net_k = \sum_{j=1}^{n_H} \omega_{kj} y_j + \omega_{k0}$$

训练样本:(x,t)

输入层：单元i的输入：x_i　单元数量：d

单元i的输出：x_i

单元i的激活函数：线性函数

隐　层：单元j的输入：net_j　单元数量：n_H

$$net_j = \sum_{i=1}^{d} \omega_{ji} x_i + \omega_{j0} = \omega_j^t x$$

$$x = (x_0, x_1, \ldots, x_d)^t, x_0 = 1$$

$$\omega_j = (\omega_{j0}, \omega_{j1}, \ldots, \omega_{jd})^t$$

单元j的输出：$y_j = f(net_j)$

单元j的激活函数：非线性函数

输出层：单元k的输入：net_k　单元数量：c

$$net_k = \sum_{j=1}^{n_H} \omega_{kj} y_j + \omega_{k0} = \omega_k^t y$$

$$y = (y_0, y_1, \ldots, y_{n_H})^t, y_0 = 1$$

$$\omega_k = (\omega_{k0}, \omega_{k1}, \ldots, \omega_{kn_H})^t$$

单元k的输出：$z_k = f(net_k)$

单元k的激活函数：非线性函数

两层神经网络分类器

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

（1）分类问题：C类分类

（2）特征空间维数：d

（3）已知条件：训练样本集 $D=\{(x,t)\}$

其中 $x=(x_1,\ldots,x_d)^t$为特征向量， t 为c维目标向量，用以表示x的类别：

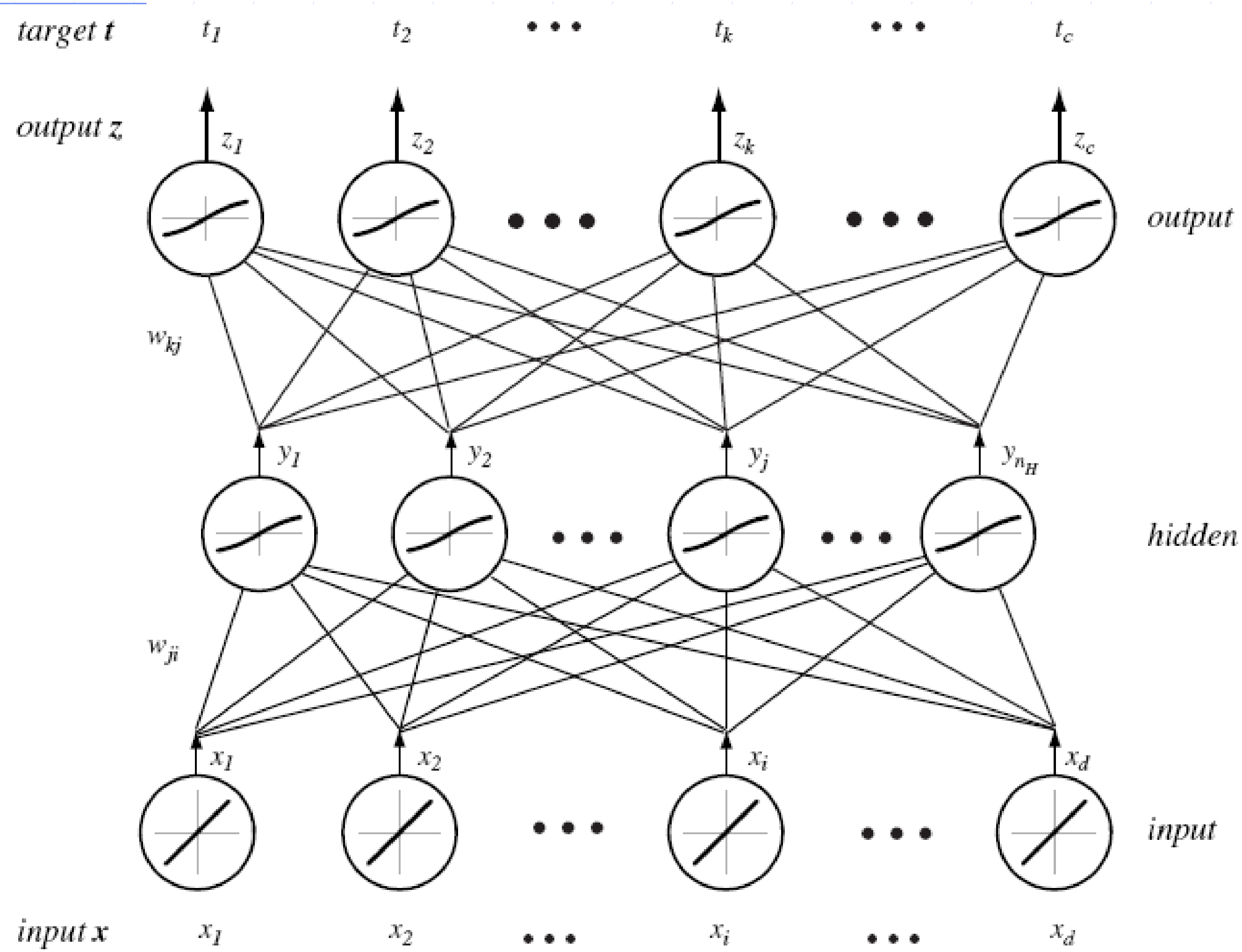
$$t=(1,0,0,\ldots,0)^t \quad 如果x\in\omega_1$$

$$t=(0,1,0,\ldots,0)^t \quad 如果x\in\omega_2$$

$$t=(0,0,0,\ldots,1)^t \quad 如果x\in\omega_c$$

（4）神经网络结构： $d\times n_H\times c$

（5）激活函数：可微的非线性函数



target t
t_1
t_2
t_k
t_c
output z
z_1
z_2
z_k
z_c
output
w_{kj}
y_1
y_2
y_j
y_{n_H}
hidden
w_{ji}
x_1
x_2
x_i
x_d
input
input x
x_1
x_2
x_i
x_d

（6）神经网络的训练目标：调整权系数 ω

即所有的 ω_{kj} 及 ω_{ji}，使得对于训练集中的每一个训练样本 (x,t)，网络的输出尽可能满足：

$$z(x)=\begin{pmatrix} z_1(x) \\ \dots \\ z_c(x) \end{pmatrix}=\begin{pmatrix} t_1 \\ \dots \\ t_c \end{pmatrix}=t$$

（7）优化准则：对于样本集D，使下述误差函数取得最小值：

$$J(\omega)=\sum_{x\in D} J_x(\omega)$$

$$J_x(\omega)=\frac{1}{2}\sum_{k=1}^{c}\left(t_k-z_k(x)\right)^2$$

实际使用中神经网络对新样本的分类：哪一个输出层神经元的输出值最大,就判该样本属于哪一类.

$$z(x)=\begin{pmatrix} z_1(x) \\ \dots \\ z_c(x) \end{pmatrix} \approx \begin{pmatrix} t_1 \\ \dots \\ t_c \end{pmatrix} = t$$

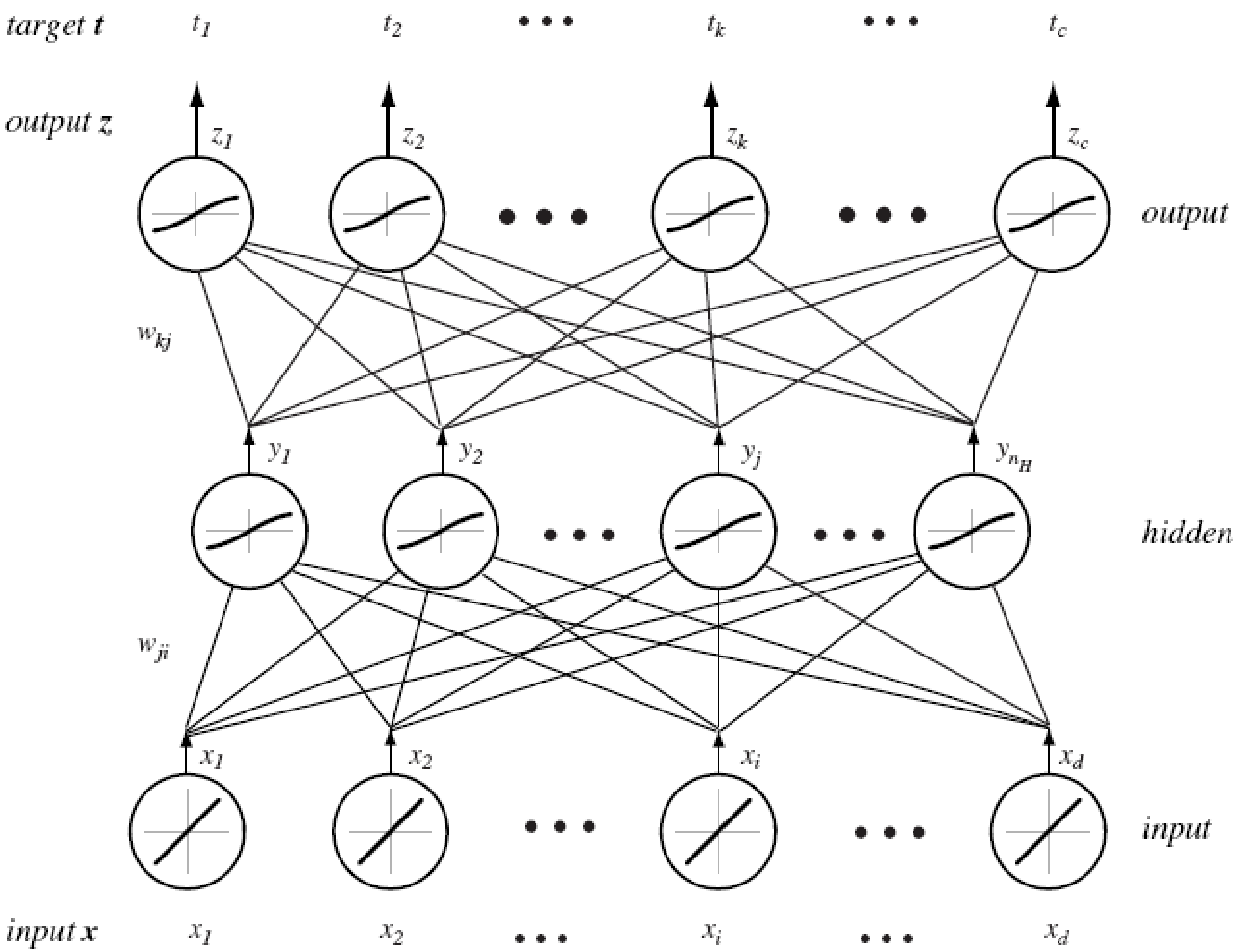
神经网络学习

1）问题的提出

输入一个样本 (x,t)，计算网络的输出z，根据z与t的差距调整所有的权系数 ω ，使z与t尽可能接近，即使 $J_x(\omega)$ 近可能小。应该如何调整权系数？

$$J_x(\omega)=\frac{1}{2}\sum_{k=1}^{c}\left(t_k - z_k\right)^2$$

$$J_x(\omega) = \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2 \qquad net_k = \sum_{j=1}^{n_H}\omega_{kj}y_j + \omega_{k0} \qquad net_j = \sum_{i=1}^{d}\omega_{ji}x_i + \omega_{j0}$$



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

$$z_k = f(net_k)$$
$$y_j = f(net_j)$$

2）权系数的调整方法

（1）误差函数：

$$J = J_x(\omega) = \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2$$

（2）计算 $\partial J / \partial \omega_{kj}$ ， $\partial J / \partial \omega_{ji}$

（3）调整权系数(Gradient Descent Procedure)：

$$\omega_{kj} \leftarrow \omega_{kj} - \eta \frac{\partial J}{\partial \omega_{kj}} \qquad \omega_{ji} \leftarrow \omega_{ji} - \eta \frac{\partial J}{\partial \omega_{ji}}$$

（3）对输出层权系数的微分

$$J=\frac{1}{2}\sum_{k=1}^{c}(t_k-z_k)^2$$

$$net_k=\sum_{j=1}^{n_H}\omega_{kj}y_j+\omega_{k0}$$

$$z_k=f(net_k)$$

$$\frac{\partial J}{\partial \omega_{kj}}=\frac{\partial J}{\partial net_k}\frac{\partial net_k}{\partial \omega_{kj}}$$

$$\frac{\partial J}{\partial net_k}=\frac{\partial J}{\partial z_k}\frac{\partial z_k}{\partial net_k}=-(t_k-z_k)f'(net_k) \qquad \frac{\partial net_k}{\partial \omega_{kj}}=y_j$$

令 $\frac{\partial J}{\partial net_k}=\delta_k$ 可得 $\frac{\partial J}{\partial \omega_{kj}}=\delta_k y_j$

$$net_j = \sum_{i=1}^{d} \omega_{ji} x_i + \omega_{j0}$$

$$y_j = f(net_j)$$

（4）对隐层权系数的微分

$$\frac{\partial J}{\partial \omega_{ji}} = \frac{\partial J}{\partial net_j} \frac{\partial net_j}{\partial \omega_{ji}}$$

$$\frac{\partial J}{\partial net_j} = \frac{\partial J}{\partial y_j} \frac{\partial y_j}{\partial net_j} \qquad \frac{\partial J}{\partial y_j} = \sum_{k=1}^{c} \delta_k \omega_{kj} \qquad \frac{\partial y_j}{\partial net_j} = f'(net_j)$$

$$\frac{\partial J}{\partial net_j} = f'(net_j) \sum_{k=1}^{c} \delta_k \omega_{kj} \qquad \frac{\partial net_j}{\partial \omega_{ji}} = x_i$$

令 $\delta_j = \dfrac{\partial J}{\partial net_j}$ 可得 $\dfrac{\partial J}{\partial \omega_{ji}} = \delta_j x_i$

$$\frac{\partial J}{\partial y_j} = \left\{ \sum_{k=1}^{c} \frac{\partial J}{\partial z_k} \frac{\partial z_k}{\partial y_j} \right\}$$

$$= -\sum_{k=1}^{c} (t_k - z_k) \frac{\partial z_k}{\partial y_j}$$

$$= -\sum_{k=1}^{c} (t_k - z_k) \frac{\partial z_k}{\partial net_k} \frac{\partial net_k}{\partial y_j}$$

$$= -\sum_{k=1}^{c} (t_k - z_k) f'(net_k) \omega_{kj} \quad = \sum_{k=1}^{c} \delta_k \omega_{kj}$$

$$J = \frac{1}{2} \sum_{k=1}^{c} (t_k - z_k)^2$$

$$z_k = f(net_k)$$

$$net_k = \sum_{j=1}^{n_H} \omega_{kj} y_j$$

（5）权系数的调整：

$$\omega_{kj} \leftarrow \omega_{kj} - \eta \frac{\partial J}{\partial \omega_{kj}} \qquad \frac{\partial J}{\partial \omega_{kj}} = \delta_k y_j \qquad \delta_k = -\left(t_k - z_k\right) f^{'}(net_k)$$

$$\omega_{ji} \leftarrow \omega_{ji} - \eta \frac{\partial J}{\partial \omega_{ji}} \qquad \frac{\partial J}{\partial \omega_{ji}} = \delta_j x_i \qquad \delta_j = f'(net_j) \sum_{k=1}^{c} \delta_k \omega_{kj}$$

反向传播算法（Back Propagation）

（1）对于给定的样本集 $D=\{(x,t)\}$ ，初始化网络结构 $d\times n_H\times c$ 。初始化权系数 ω ，学习速率 η ，阈值 θ ，变量 $k=1$。

（2）从D中取出第k个样本 (x,t)，根据该样本更新权系数 ω：

$$\omega_{kj}\leftarrow\omega_{kj}-\eta\partial J/\partial\omega_{kj}\qquad \omega_{ji}\leftarrow\omega_{ji}-\eta\partial J/\partial\omega_{ji}$$

（3）k=K+1，如果 $k>n$ ，令k=1。转第2步继续进行循环。退出条件：在给定样本集上的平均误差足够小。

$$J=J_x(\omega)=\frac{1}{2}\sum_{k=1}^{c}(t_k-z_k)^2$$