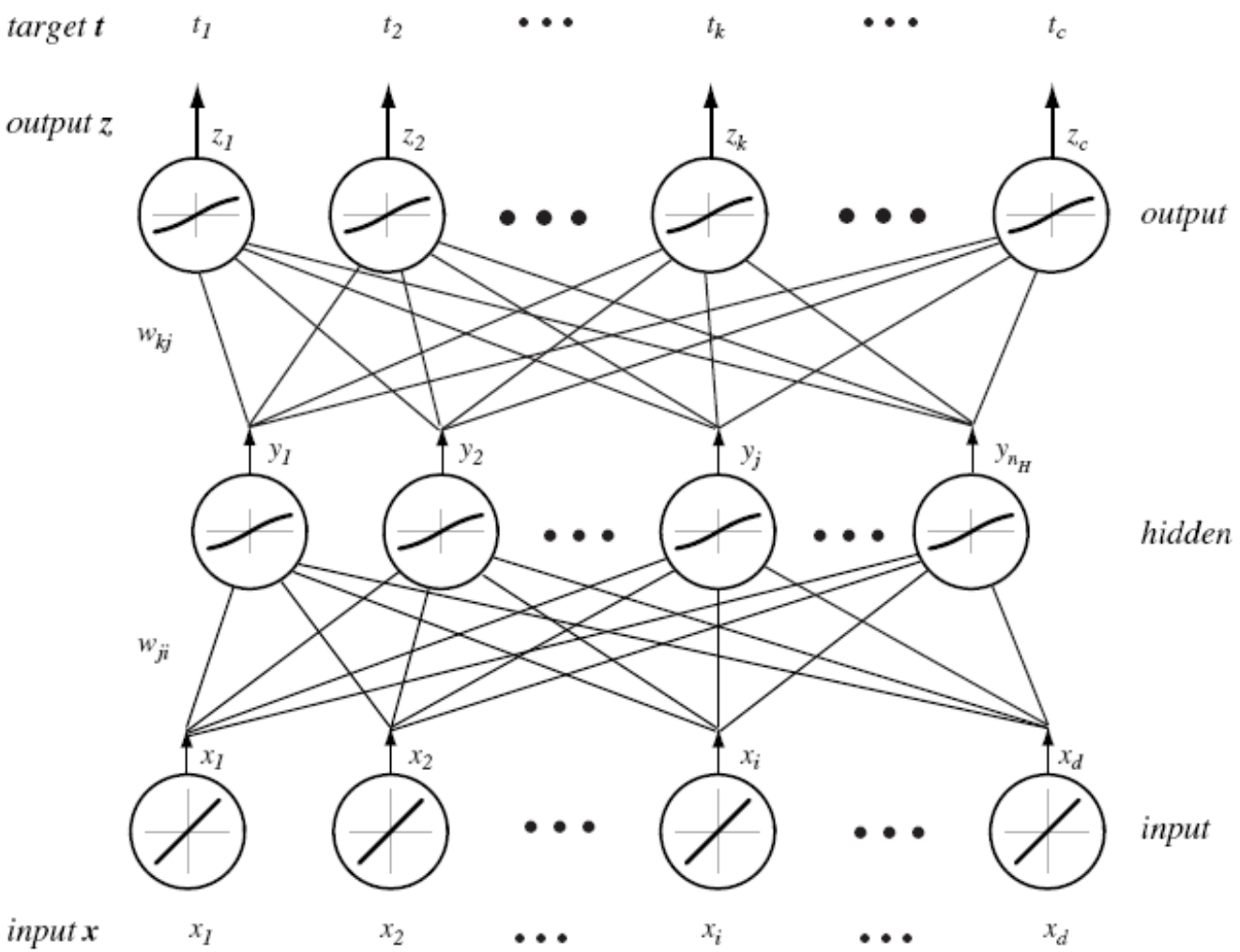

target t
t1
t2
tk
tc
output z
z1
z2
zk
zc
output
wkj
y1
y2
yj
yn_H
hidden
wji
x1
x2
xi
xd
input
input x
x1
x2
xi
xd

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：

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

优化准则：对于样本集D，使下述误差函数取得最小值：

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

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

权系数的调整方法

（1）误差函数：

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

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

（3）调整权系数(Gradient Descent Procedure)：

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

权系数的调整：

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

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

3）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，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：训练误差足够小。

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

随机化反向传播算法

（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、学习率 η 。

（2）随机从D中取出一个样本 (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）如果训练误差足够小则退出循环，否则转第2步继续进行循环。

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

注意：上述算法可以保证$J(\omega)$趋于极小。

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

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

批处理反向传播算法

（1）误差函数

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

（2）误差函数对权系数偏微分的计算

$$\frac{\partial J(\omega)}{\partial \omega_{kj}}=\sum_{x\in D}\frac{\partial J_x(\omega)}{\partial \omega_{kj}} \qquad \frac{\partial J(\omega)}{\partial \omega_{ji}}=\sum_{x\in D}\frac{\partial J_x(\omega)}{\partial \omega_{ji}}$$

（3）权系数的更新

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

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

算法的具体描述：

（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，学习率 η，阈值 θ。

（2）根据样本集D更新权系数 ω：

$$\omega_{kj}\leftarrow\omega_{kj}-\eta\partial J(\omega)/\partial\omega_{kj}$$

$$\omega_{ji}\leftarrow\omega_{ji}-\eta\partial J(\omega)/\partial\omega_{ji}$$

（3）计算 $\Delta J=J(\omega_{pre})-J(\omega)$，如果 $\Delta J<\theta$ 结束训练，并认为此时的 ω 为最优。否则转第2步继续进行循环。

反向传播算法中的若干关键问题

1）激活函数（Activation Function）

激活函数f应满足如下基本条件:

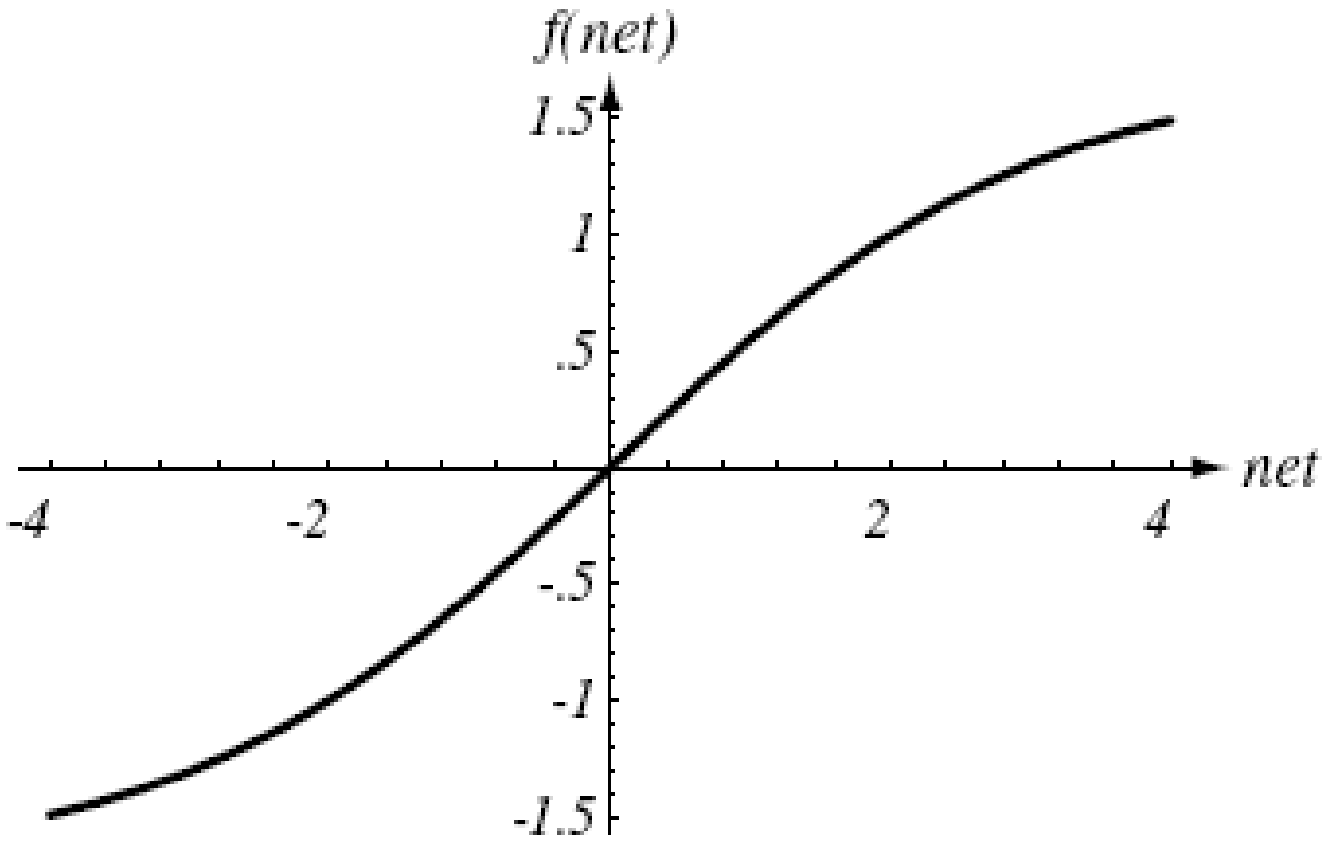
非线性、有界、连续、可微

常用的激活函数：Sigmoid函数

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

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

$$a=1.716 \quad b=2/3$$


f(net)
1.5
1
.5
-.5
-1
-1.5
net
-4
-2
2
4

2）目标向量：

目标向量应该根据激活函数的值域进行选择。设训练样本为 (x,t)，且该样本属于 ω_k。则根据激活函数的不同，可分别采用如下方法：

$$t=\begin{pmatrix} t_1 \\ \vdots \\ t_k \\ \vdots \\ t_c \end{pmatrix}=\begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix}$$

$$t_k=1, t_i=0, 1\le i\le c, i\ne k$$

$$t=\begin{pmatrix} t_1 \\ \vdots \\ t_k \\ \vdots \\ t_c \end{pmatrix}=\begin{pmatrix} -1 \\ \vdots \\ 1 \\ \vdots \\ -1 \end{pmatrix}$$

$$t_k=1, t_i=-1, 1\le i\le c, i\ne k$$

3）权系数的初始化：

在训练过程开始前，权系数通常采用随机化方法进行随机赋值，且初始权值不宜过大。典型的方法是：

对于每一个权系数ω_{ji}，按照均匀分布在区间 $[-1/\sqrt{d},1/\sqrt{d}]$ 内随机选择一个数值进行赋值。

对于每一个权系数 ω_{kj}，按照均匀分布在区间$[-1/\sqrt{n_H},1/\sqrt{n_H}]$内随机选择一个数值进行赋值。

4）学习速率

学习速率 η 直接影响权系数调整时的步长。学习速率过小，导致算法收敛速度缓慢。学习速率过大，导致算法不收敛。学习速率的典型取值 $\eta \approx 0.1$ 。另外学习速率可变。

5）误差函数的局部极小值

调整权系数的目标是使误差函数取得最小值。但是，采用梯度下降法(Gradient Descent Procedure)不能保证获得最小值，而只能保证得到一个极小值。

如果训练过程无法使误差函数降低到预期的程度，一种常用的方法是：再一次对权系数进行随机初始化，并重新训练网络。

6）学习曲线

样本集D的划分：一般情况下，可把已知的样本集D划分为三个子集，即训练集、确认集、测试集。

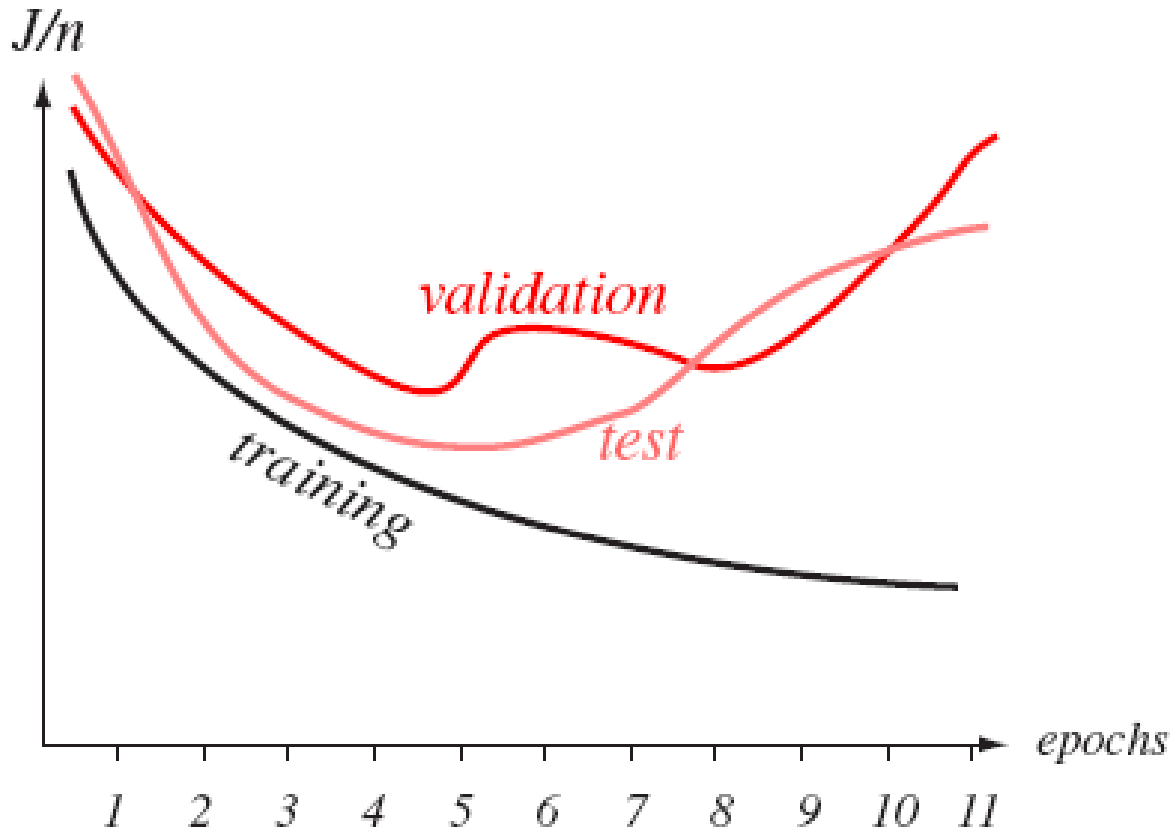
训练集(Training Set)：用来调整权系数，使误差函数尽可能变小。

确认集(Validation Set)：用来初步验证神经网络对未来新样本的分类能力，并据此确定最佳的权系数。神经网络的训练过程需要采用训练集及确认集共同完成。

测试集(Test Set)：在训练过程最终结束后，再采用测试集对网络的分类性能进行最后测试，用以评估神经网络在实际应用中的实际性能。

Epoch: 在训练集D中的每一个训练样本均参与了一次对权系数的更新之后，称为完成了一个Epoch。

学习曲线



7）结束训练的准则：

过度拟合：在经过长时间的训练以后，神经网络对训练样本的分类能力很好，但是对测试样本的分类能力却很差，这一现象称为过度拟合。

结束训练的准则：通过学习曲线可以避免过度拟合的发生。在确认集上的平均误差函数取得最小值的时候，就可以停止训练过程，并认为此时的权系数为最优的权系数。

8）隐层单元数量的确定

隐层单元的数量直接决定了神经网络的复杂度。隐层单元数量过少，无法实现复杂的分类面，因此无法对复杂的数据分布进行有效分类。隐层单元数量过多，容易产生过度拟合，因此影响对未来新样本的分类能力。

经验法则：隐层单元数量决定了网络权系数的数量。合适的n_H应该保证网络权系数的数量近似等于N/10（N为训练集中样本的数量）。

阅读材料

杜达著：模式分类