第4章 经典人工神经网络

一、生物神经元与生物神经网络

人工神经网络是对人脑的模拟

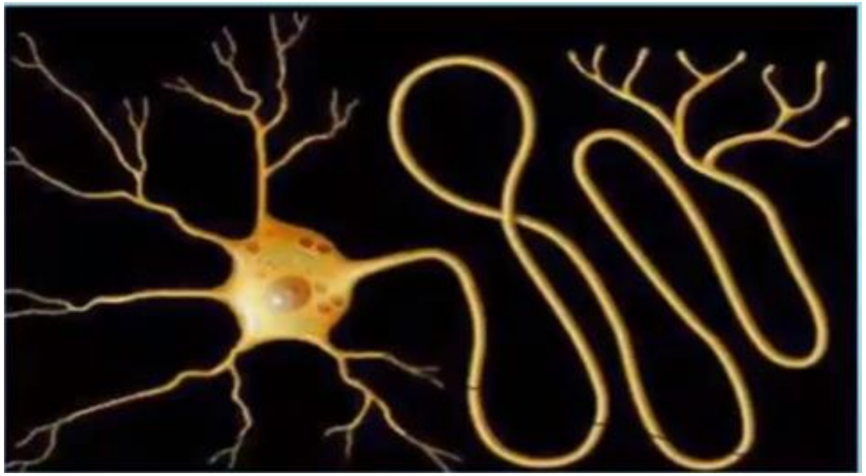
人工神经元 模拟 生物神经元

人工神经网络 模拟 生物神经网络

一、生物神经元与生物神经网络

人类大脑大约包含有1.4×10^{11}个神经元，每个神经元与大约$10^3\sim10^5$个其它神经元相连接，构成一个极为庞大而复杂的网络，即生物神经网络。

神经生理学和神经解剖学的研究结果表明，神经元是脑组织的基本单元，是神经系统结构与功能的单位。



一、生物神经元与生物神经网络

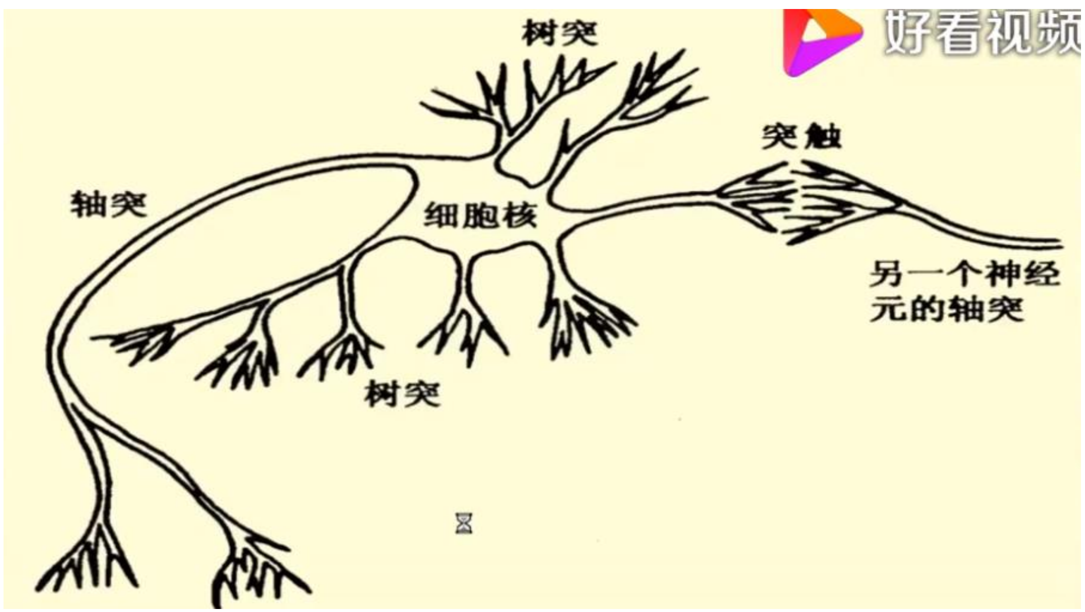
1、生物神经元的结构

生物神经元在结构上由：

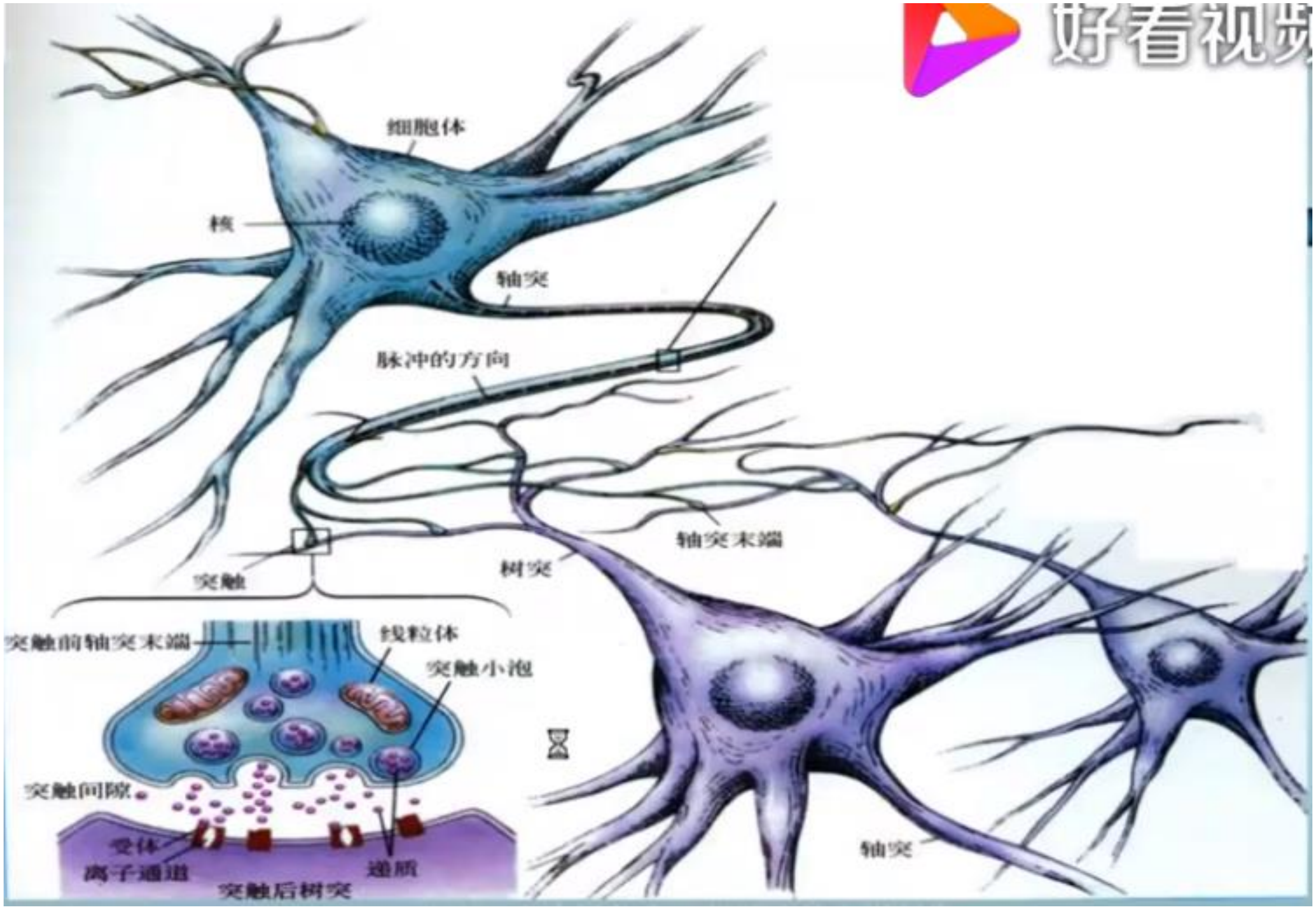
细胞体(Cell body)　　树突(Dendrite)

轴突(Axon)　　突触(Synapse)

四部分组成。用来完成神经元间信息的接收、传递和处理。

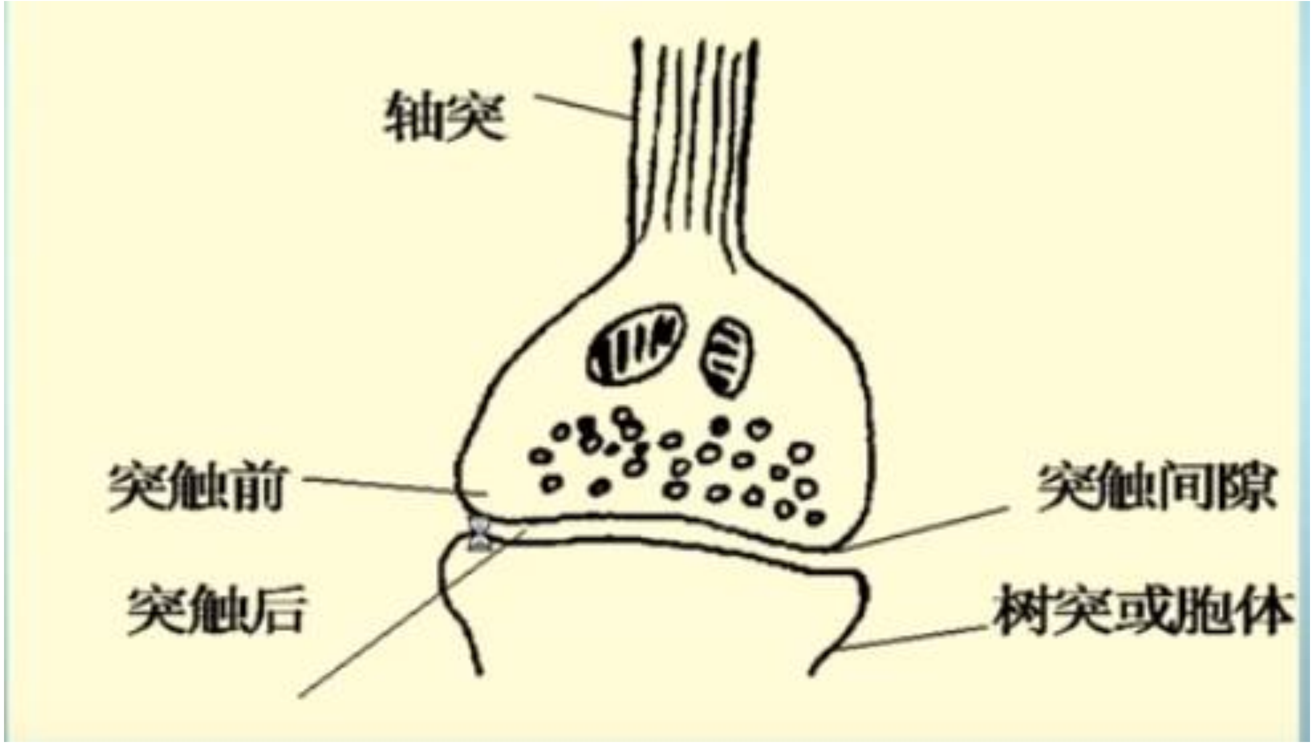


一、生物神经元与生物神经网络



一、生物神经元与生物神经网络

2、生物神经元信息的传递与接收



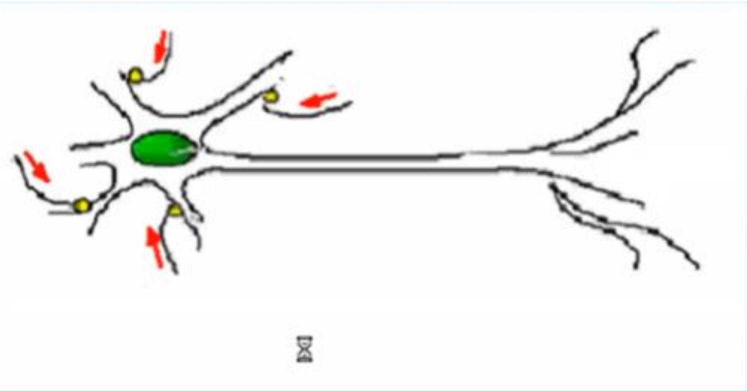
一、生物神经元与生物神经网络

3、生物神经元信息的整合

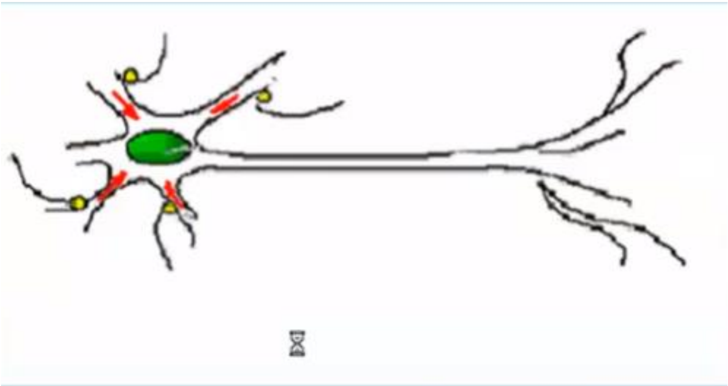
空间整合： 同一时刻产生的刺激所引起的膜电位变化，大致等于各单独刺激引起的膜电位变化的代数和。

时间整合： 各输入脉冲抵达神经元的时间先后不一样。总的突触后膜电位为一段时间内的累积。

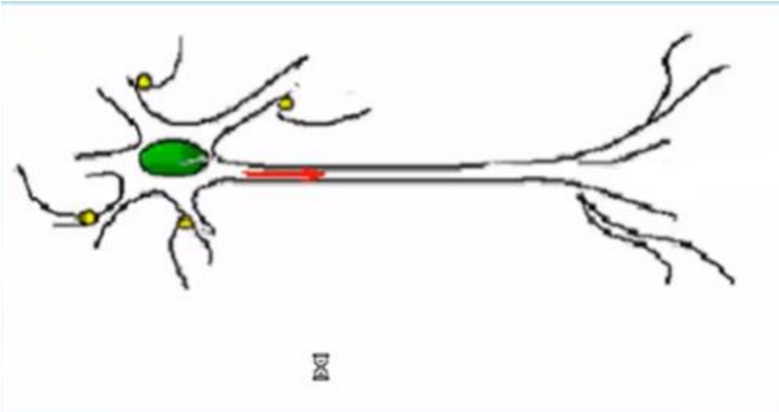
一、生物神经元与生物神经网络



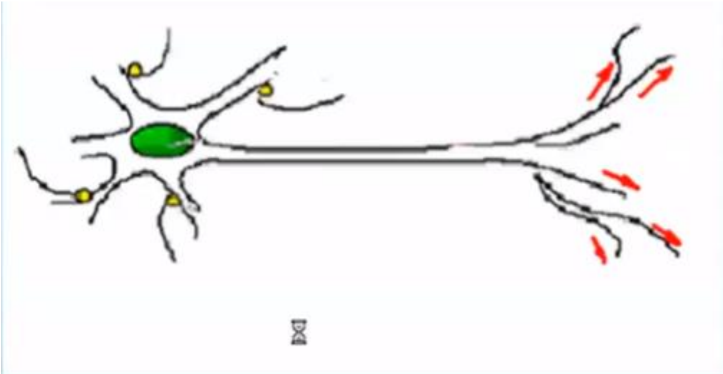
信息输入



信息传播与处理



信息传播与处理（整合）

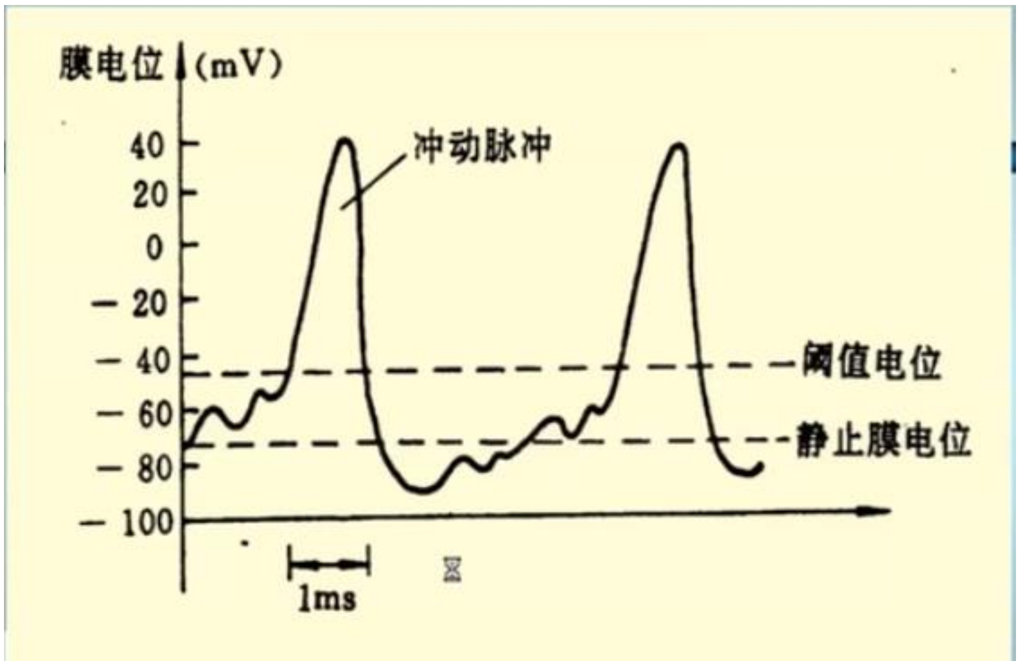


信息输出

一、生物神经元与生物神经网络

神经元间信息的产生、传递和处理是一种电化学活动。

神经元状态：	膜电位：
静息	极　化
兴奋	去极化
抑制	超极化



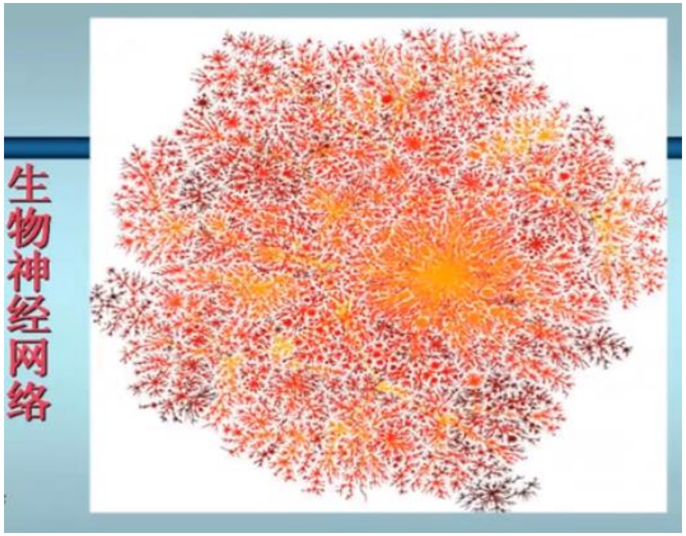
一、生物神经元与生物神经网络

4、生物神经网络

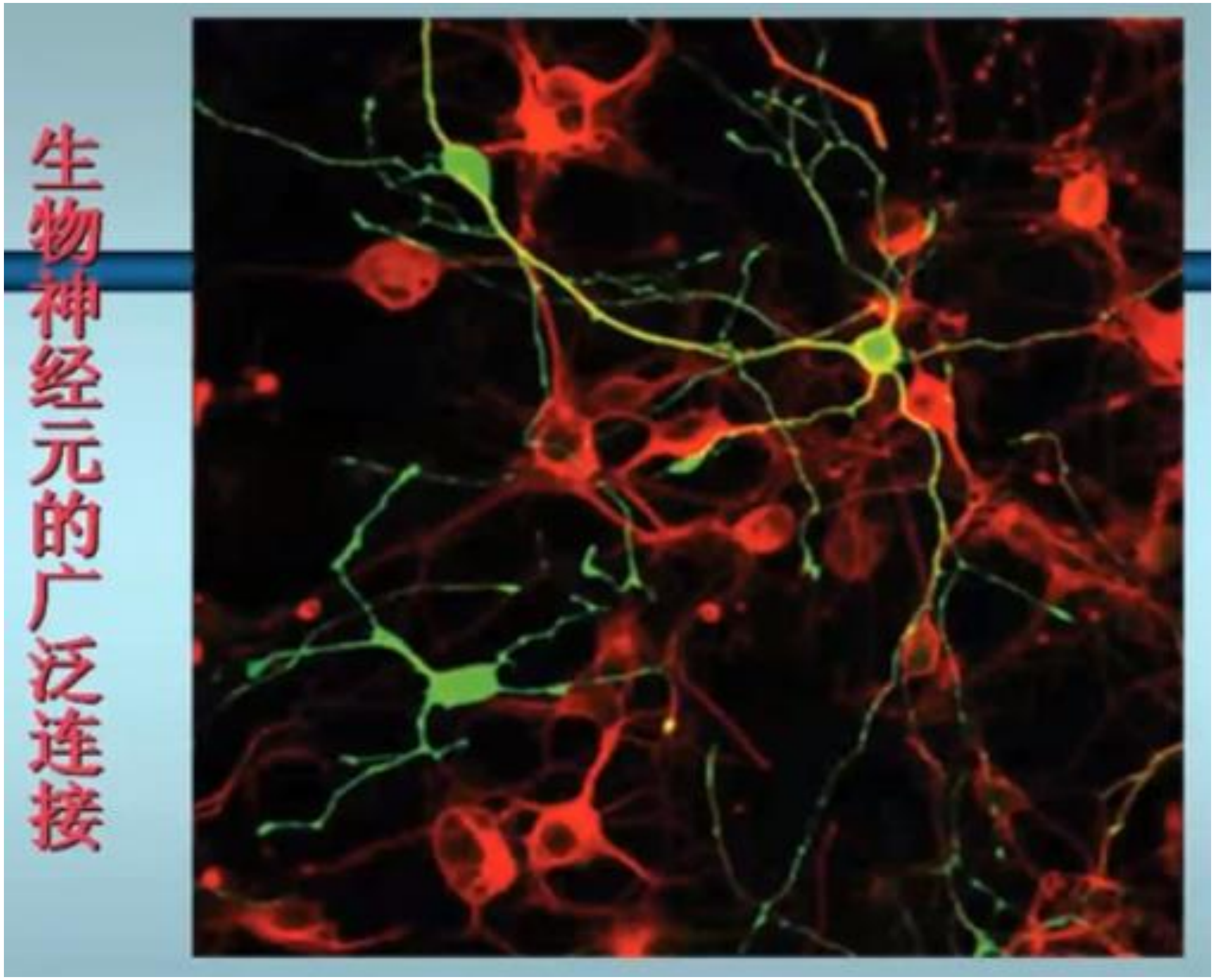
由多个生物神经元以确定方式和拓扑结构 相互连接即形成生物神经网络。

生物神经网络的功能不是单个神经元信息处理功能的简单叠加。

神经元之间的突触连接方式和连接强度不同并且具有可塑性，这使神经网络在宏观呈现出千变万化的复杂的信息处理能力。



一、生物神经元与生物神经网络



一、生物神经元与生物神经网络

二、人工神经元

神经元及其突触是神经网络的基本器件。因此，模拟生物神经网络应首先模拟生物神经元。在人工神经网络中，神经元常被称为“处理单元”。有时从网络的观点出发常把它称为“节点”。人工神经元是对生物神经元的一种形式化描述。

二、人工神经元

1、语言描述（六点假设）

(1) 每个神经元都是一个多输入单输出的信息处理单元；

(2) 神经元输入分兴奋性输入和抑制性输入两种类型；

(3) 神经元具有空间整合特性和阈值特性；

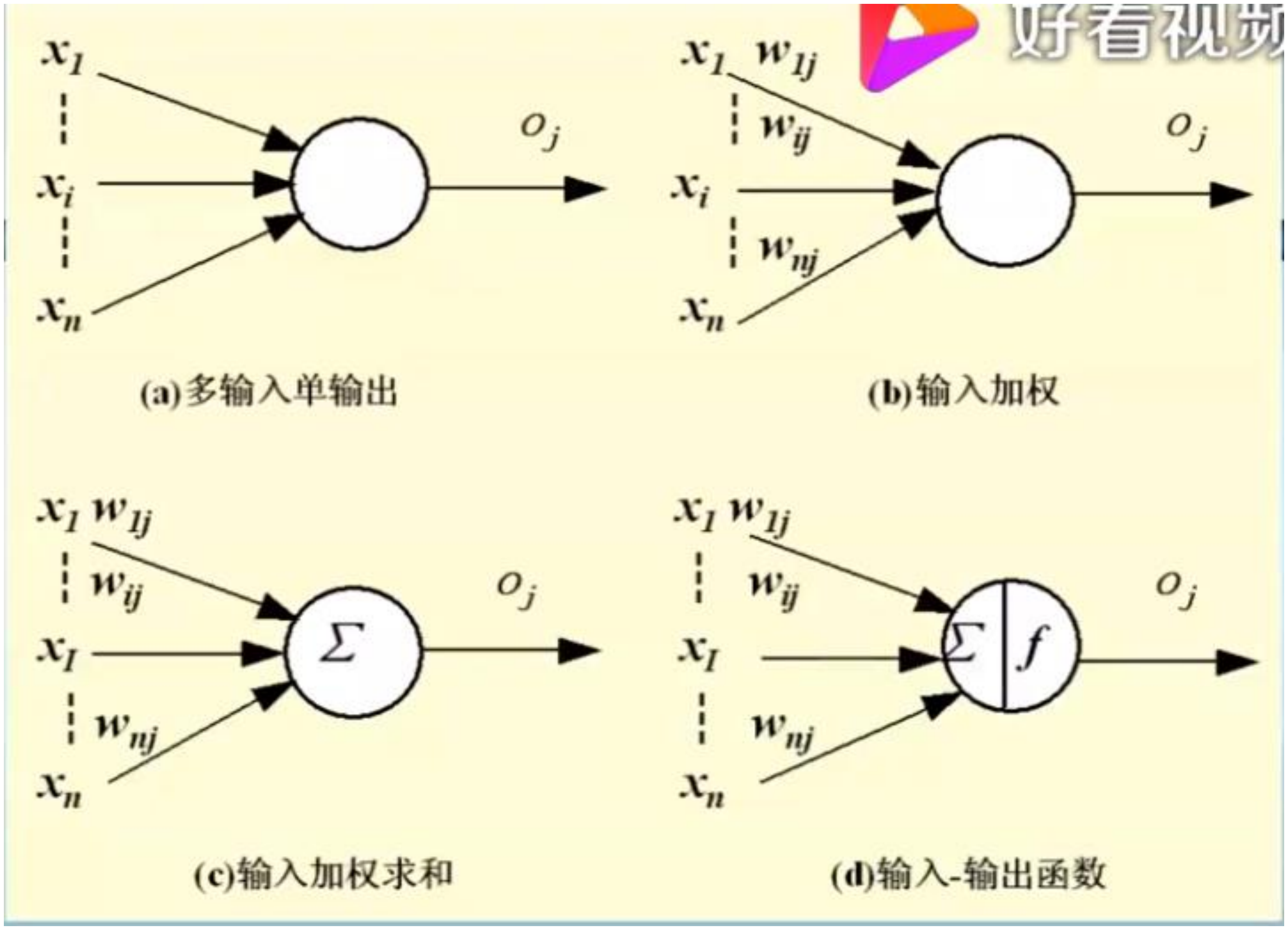
(4) 神经元输入与输出间有固定的时滞，主要取决于突触延搁；

(5) 忽略时间整合作用和不应期；

(6) 神经元本身是非时变的，即其突触时延和突触强度均为常数。

二、人工神经元

2、符号描述



二、人工神经元

3、数学描述

$$o_j(t) = f\{[\sum_{i=1}^{n} w_{ij} x_i(t - \tau_{ij})] - T_j\}$$

τ_{ij}—— 输入输出间的突触时延；

T_j—— 神经元j的阈值；

w_{ij}—— 神经元i到j的突触连接系数或称权重值；

$f()$——神经元转移函数。

$$o_j(t+1) = f\{[\sum_{i=1}^{n} w_{ij} x_i(t)] - T_j\}$$

二、人工神经元

$$\mathrm{net}'_j(t) = \sum_{i=1}^{n} w_{ij} x_i(t)$$

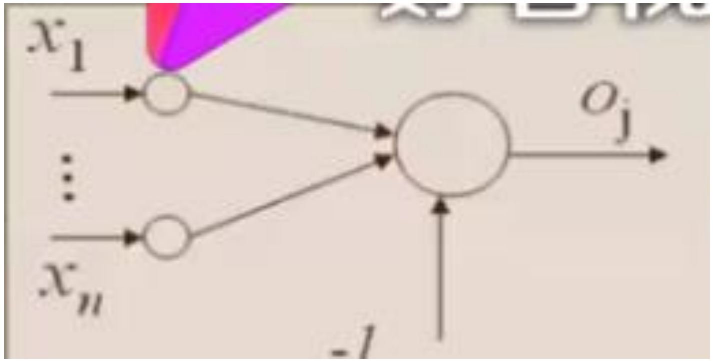
$$W_j = (w_1 \quad w_2 \quad \cdots \quad w_n)^T$$

$$X = (x_1 \quad x_2 \quad \cdots \quad x_n)^T$$

$$\mathrm{net}'_j = W_j^T X$$

令 $x_0 = -1$，$w_0 = T_j$

则有 $-T_j = x_0 w_0$

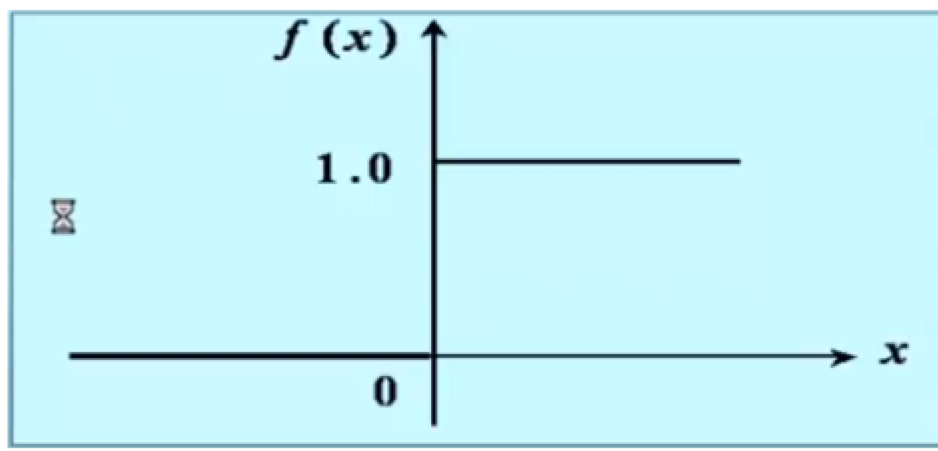


二、人工神经元

$$\mathrm{net}'_j - T_j = net_j = \sum_{i=0}^{n} w_{ij} x_i = \mathbf{W}_j^{\mathrm{T}} \mathbf{X}$$

$$o_j = f(net_j) = f(W_j^{\mathrm{T}} X)$$

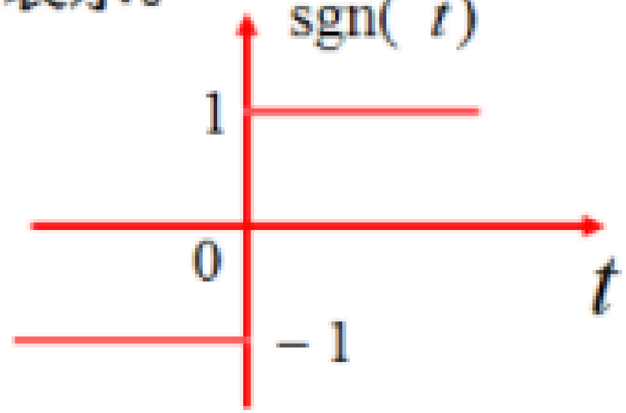
$$f(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$



二、人工神经元

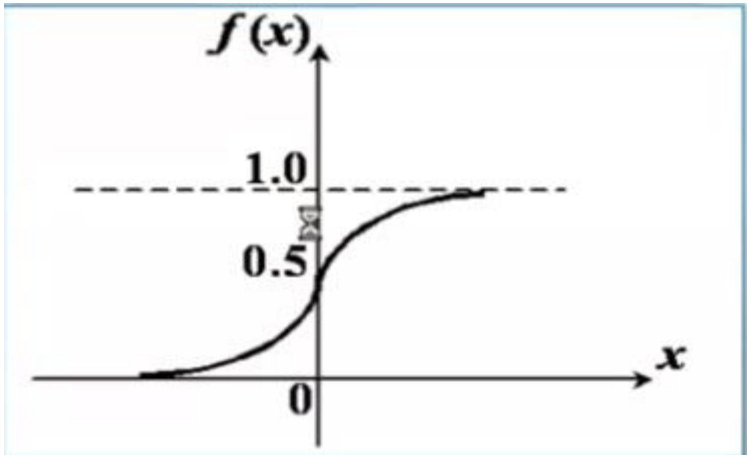
$$\mathrm{sgn}(t)=\begin{cases}1 & t>0\\ -1 & t<0\end{cases}$$

表示。

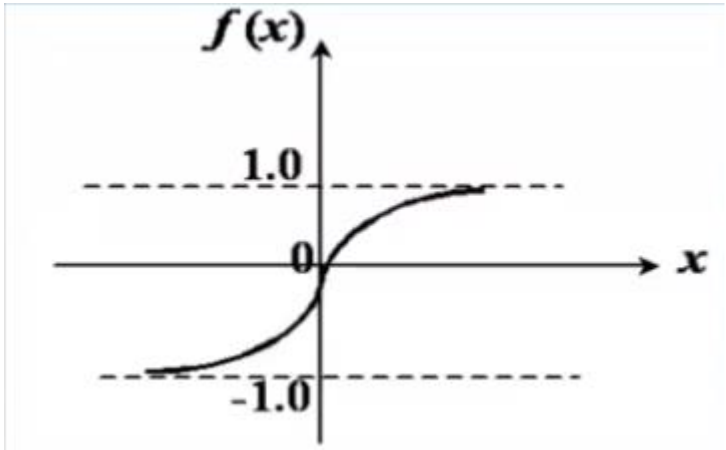


二、人工神经元

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

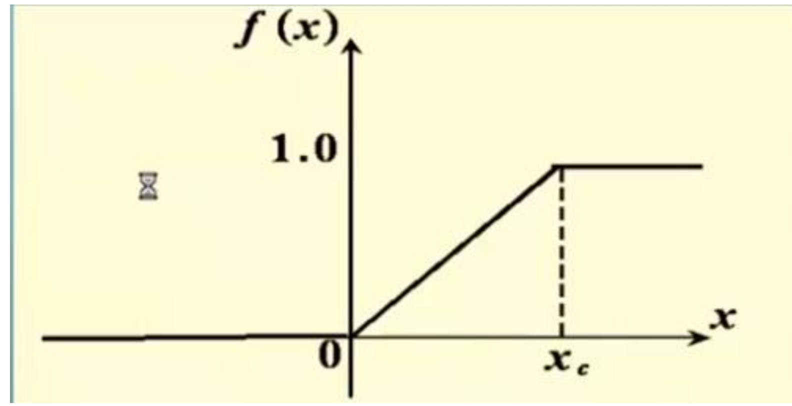


$$f(x)=\frac{2}{1+e^{-x}}-1=\frac{1-e^{-x}}{1+e^{-x}}$$

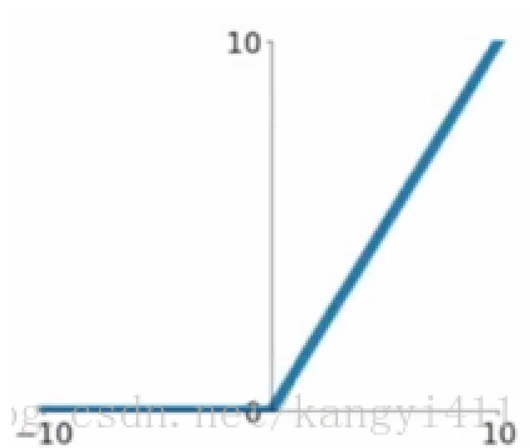


二、人工神经元

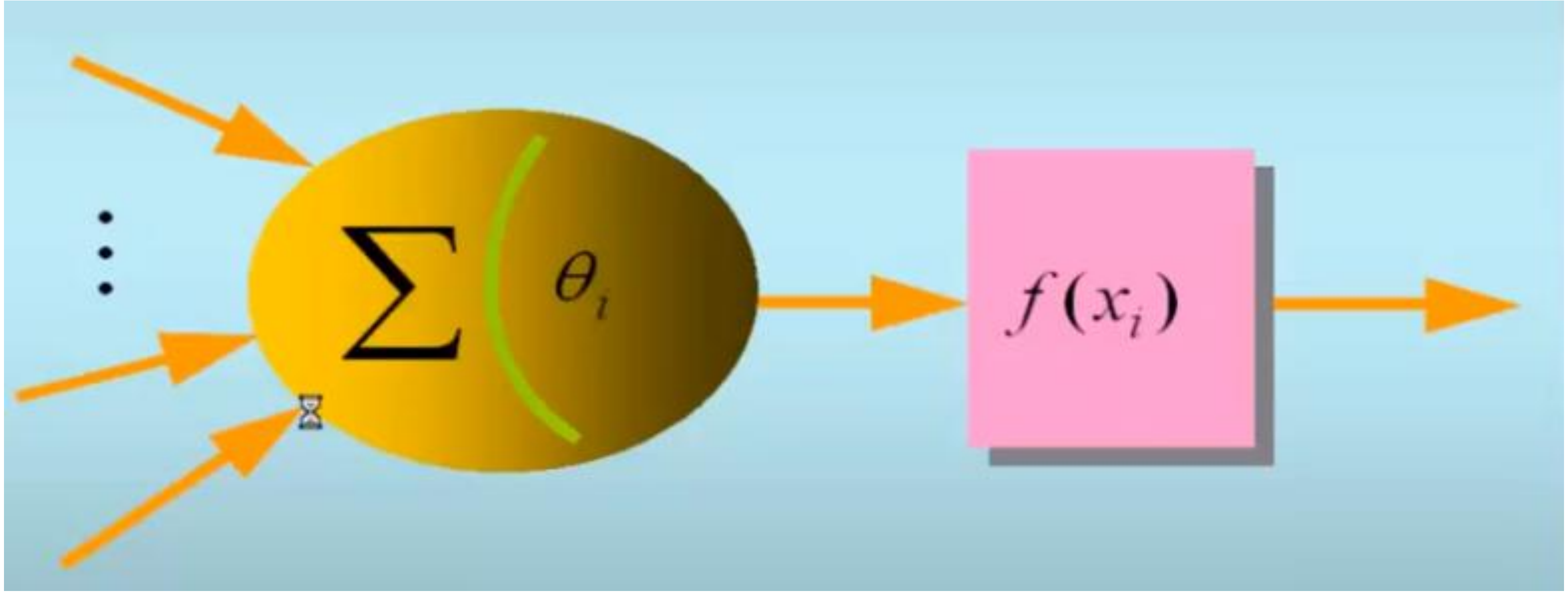
$$f(x)=\begin{cases} 0 & x\leq 0 \\ cx & 0<x\leq x_c \\ 1 & x_c<x \end{cases}$$



$$f(x) = max(0, x)$$

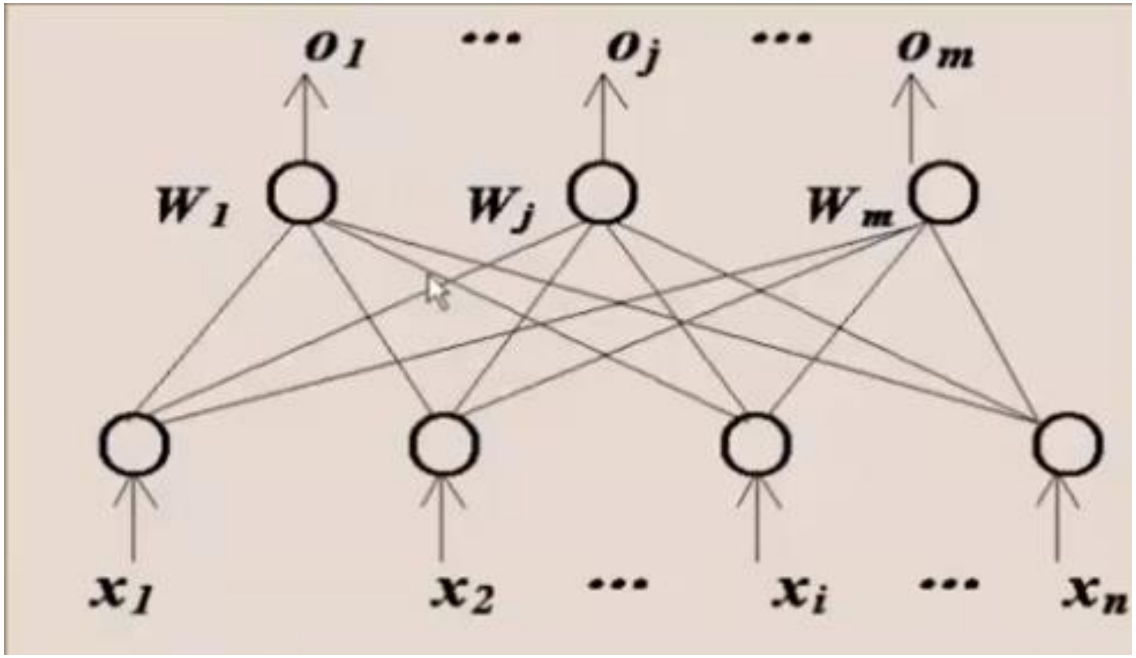


三、单层感知器



三、单层感知器

1、单层感知器符号表达（拓扑结构）



三、单层感知器

2、单层感知器数学表达

$$\mathbf{X} = (x_1, x_2, \ldots x_i, \ldots, x_n)^T$$

$$\mathbf{O} = (o_1, o_2, \ldots o_i, \ldots, o_m)^T$$

$$\mathbf{W}_j = (w_{1j}, w_{2j}, \ldots w_{ij}, \ldots, w_{nj})^T$$

$$j=1,2,\ldots,m$$

净输入：$net_j = \sum_{i=1}^{n} w_{ij} x_i \qquad j = 1,2,\ldots,m$

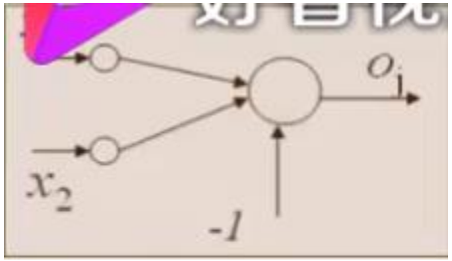
输出：$o_j = sgn(net_j - T_j) = sgn(\sum_{i=0}^{n} w_{ij} x_i) = sgn(W_j^T X)$

$$j = 1,2,\ldots,m$$

三、单层感知器

3、单层感知器的功能

(1)设输入向量$X=(x_1, x_2)^T$



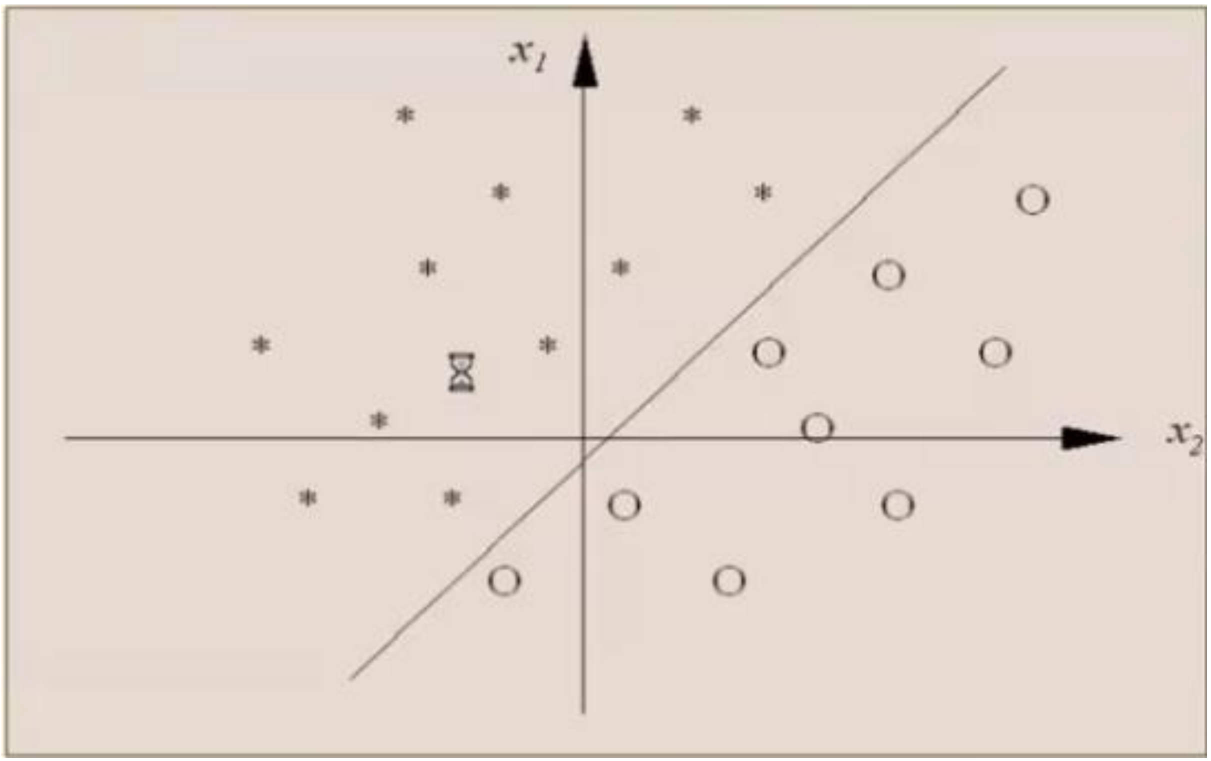
单计算节点感知器

$$\text{输出}: o_j = \begin{cases} 1 & w_{1j}x_1 + w_{2j}x_2 - T_j > 0 \\ -1 & w_{1j}x_1 + w_{2j}x_2 - T_j < 0 \end{cases}$$

则由方程 $w_{1j}x_1+w_{2j}x_2-T_j=0$

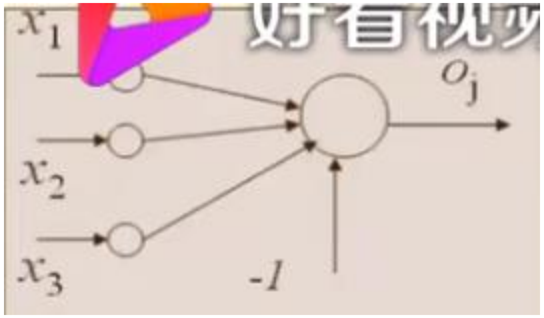
确定了二维平面上的一条分界线。

三、单层感知器



三、单层感知器

(2)设输入向量$X=(x_1,x_2,x_3)^T$

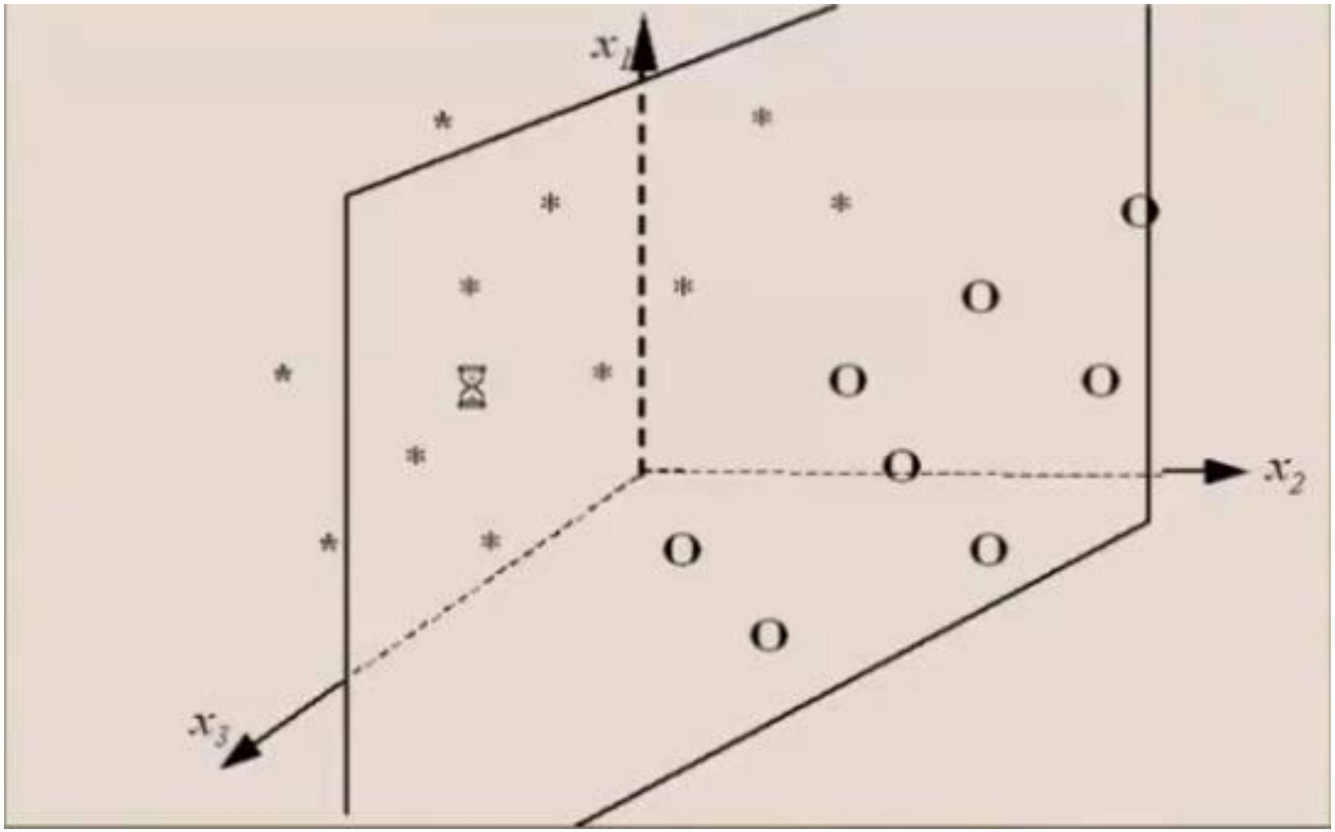


$$
\text{输出：}\quad o_j=\begin{cases}1 & w_{1j}x_1+w_{2j}x_2+w_{3j}x_3-T_j>0\\ -1 & w_{1j}x_1+w_{2j}x_2+w_{3j}x_3-T_j<0\end{cases}
$$

则由方程 $w_{1j}x_1+w_{2j}x_2+w_{3j}x_3-T_j=0$

确定了三维空间上的一个分界平面。

三、单层感知器



三、单层感知器

(3) 设输入向量$X=(x_1,\ x_2,\dots,\ x_n)^T$

输出：$\operatorname{sgn}(w_{1j}x_1+w_{2j}x_2+\dots+w_{nj}xn-T_j)$

则由方程 $w_{1j}x_1+w_{2j}x_2+\dots+w_{nj}x_{\mathrm{n}}-T_j=0$

确定了n维空间上的一个分界平面。

一个最简单的单计算节点感知器具有分类功能。其分类原理是将分类知识存储于感知器的权向量（包含了阈值）中，由权向量确定的分类判决界面将输入模式分为两类。

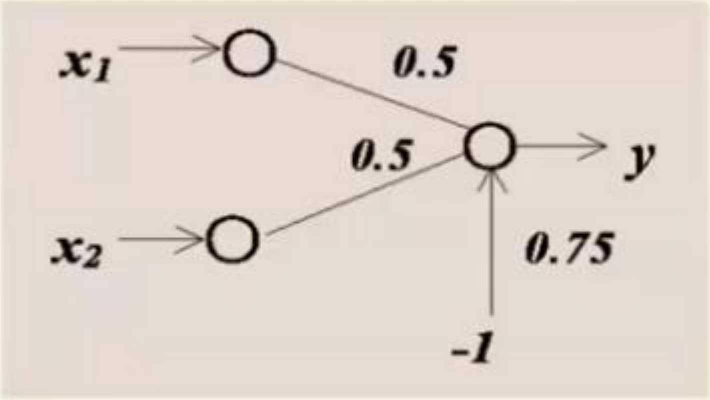
三、单层感知器

4、单层感知器解决“与”、“或”问题的实例

逻辑“与”真值表

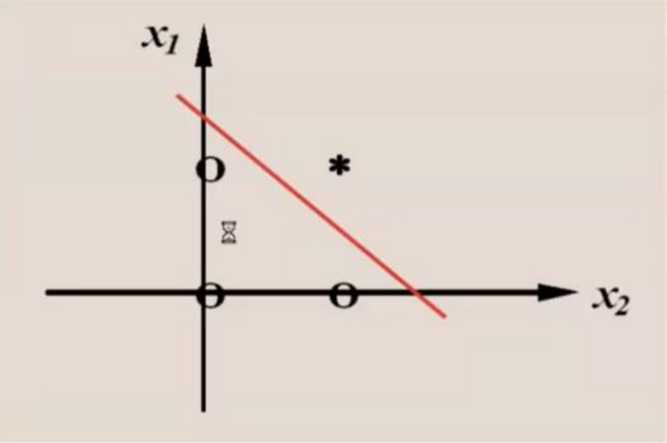
x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

感知器结构



$$w_1x_1+w_2x_2-T=0$$

$$0.5x_1+0.5x_2-0.75=0$$

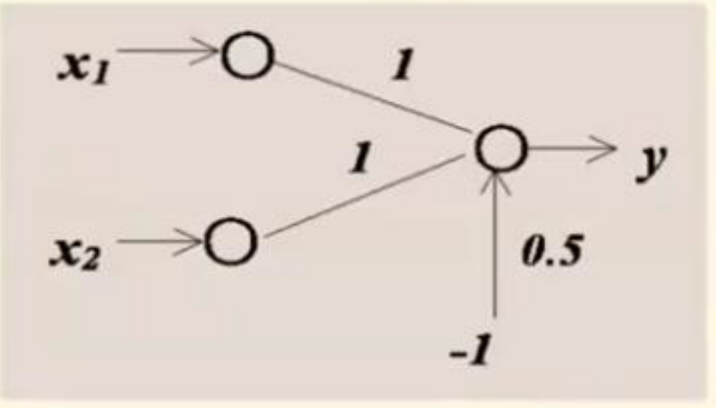


三、单层感知器

逻辑“或”真值表

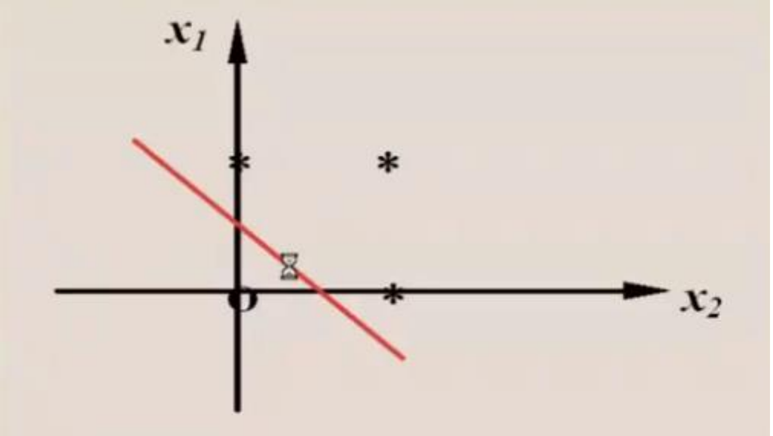
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

感知器结构



$$w_1x_1+w_2x_2-T=0$$

$$x_1+x_2-0.5=0$$



三、单层感知器

一个最简单的单计算节点感知器具有分类功能。其分类原理是将分类知识存储于感知器的权向量（包含了阈值）中，由权向量确定的分类判决界面将输入模式分为两类。

三、单层感知器

5、单层感知器的训练算法

(1) 对各权值$w_{0j}(0), w_{1j}(0), \cdots, w_{nj}(0)$，$j=1, 2, \cdots, m$（$m$为计算层的节点数）赋予较小的非零随机数；

(2) 输入样本对$\{X^p, d^p\}$，其中$X^p=(-1, x_1^p, x_2^p, \cdots, x_n^p)$，$d^p$为期望的输出向量（教师信号），上标$p$代表样本对的模式序号，设样本集中的样本总数为$P$，则$p=1,2,\cdots, P$；

(3)计算各节点的实际输出 $o_j^p(t)=\mathrm{sgn}[W_j^T(t)X^p]$，$j=1,2,\ldots,m$；

(4)调整各节点对应的权值，$W_j(t+1)= W_j(t)+ \eta[d_j^p-o_j^p(t)]X^p$，$j=1, 2, \cdots, m$，其中为学习率，用于控制调整速度，太大会影响训练的稳定性，太小则使训练的收敛速度变慢，一般取$0< \eta \leq 1$；

(5)返回到步骤(2)输入下一对样本，周而复始直到对所有样本，感知器的实际输出与期望输出相等。

三、单层感知器

6、实例

$X^1=(-1,\ 1,\ -2,\ 0)^T \qquad d^1=-1$

$X^2=(-1,\ 0,\ 1.5,\ -0.5)^T \qquad d^2=-1$

$X^3=(-1,\ -1,\ 1,\ 0.5)^T \qquad d^3=1$

设初始权向量$W(0)=(0.5,1,-1,0)^T$，$\eta=0.1$。注意，输入向量中第一个分量x_0恒等于-1，权向量中第一个分量为阈值，试根据以上学习规则训练该感知器。

第一步 输入X^1，得

$$W^T(0)X^1=(0.5,1,-1,0)(-1,1,-2,0)^T=2.5$$

$$o^1(0)=sgn(2.5)=1$$

$$W(1)=W(0)+\eta[d^1-o^1(0)]X^1$$
$$=(0.5,1,-1,0)^T+0.1(-1-1)(-1,1,-2,0)^T$$
$$=(0.7,0.8,-0.6,0)^T$$

三、单层感知器

第二步 输入X^2，得

$$W^T(1)X^2=(0.7,0.8,-0.6,0)(-1,0,1.5,-0.5)^T=-1.6$$

$$o^2(1)=sgn(-1.6)=-1$$

$$W(2)=W(1)+\eta[d^2-o^2(1)]X^2$$

$$=(0.7,0.8,-0.6,0)^T+0.1[-1-(-1)](-1,0,1.5,-0.5)^T$$

$$=(0.7,0.8,-0.6,0)^T$$

第三步 输入X^3，得

$$W^T(2)X^3=(0.7,0.8,-0.6,0)(-1,-1,1,0.5)^T=-2.1$$

$$O^3(2)=sgn(-2.1)=-1$$

$$W(3)=W(2)+\eta[d^3-o^3(2)]X^3$$

$$=(0.7,0.8,-0.6,0)^T+0.1[1-(-1)](-1,-1,1,0.5)^T$$

$$=(0.5,0.6,-0.4,0.1)^T$$

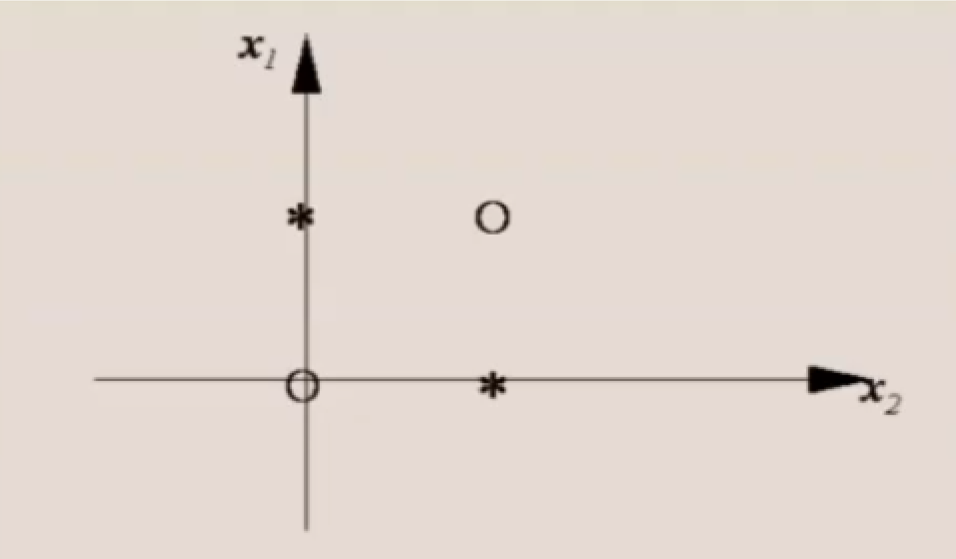
三、单层感知器

7、单层感知器的局限性

问题：能否用感知器实现“异或”功能？

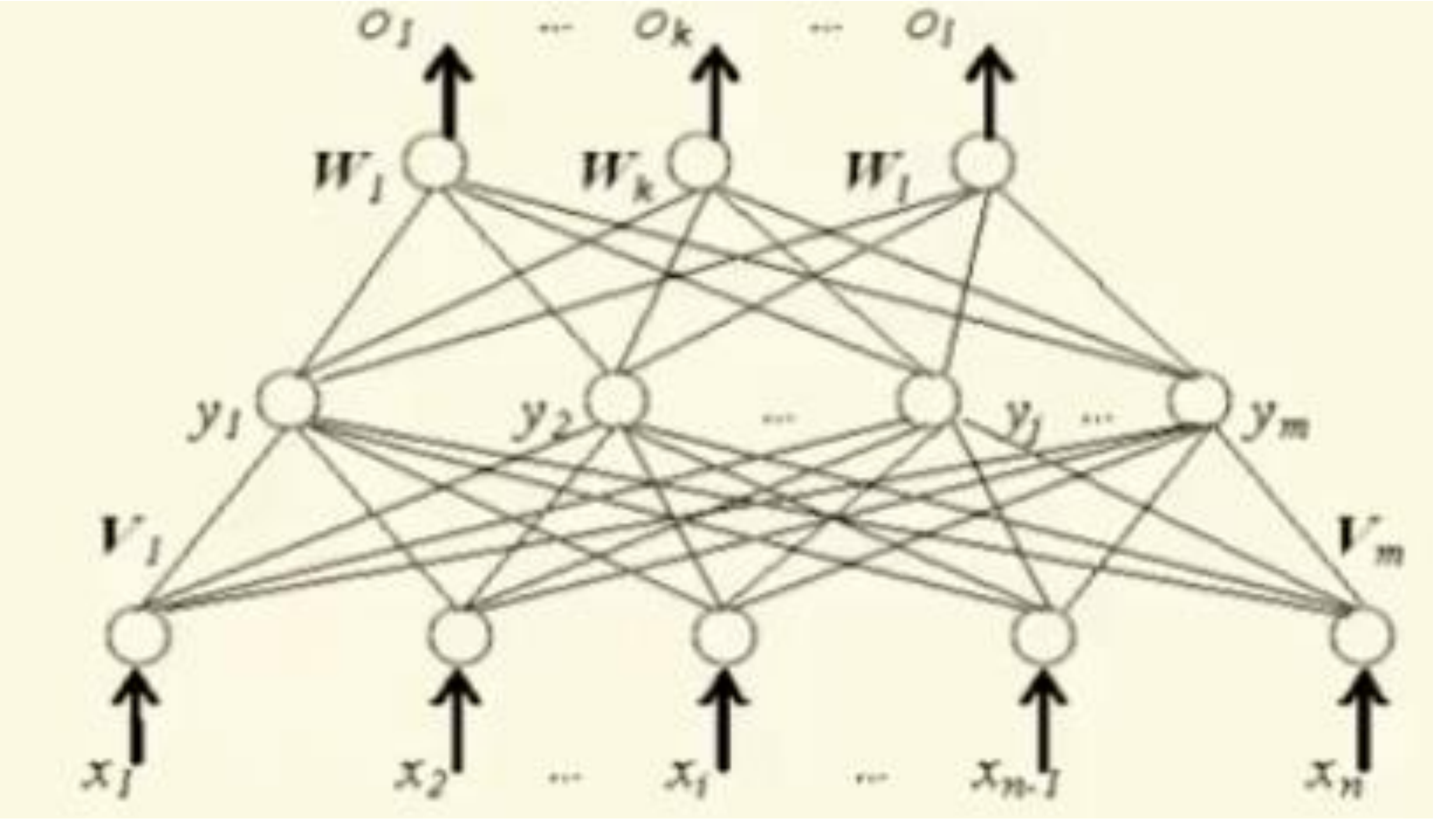
“异或”的真值表

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

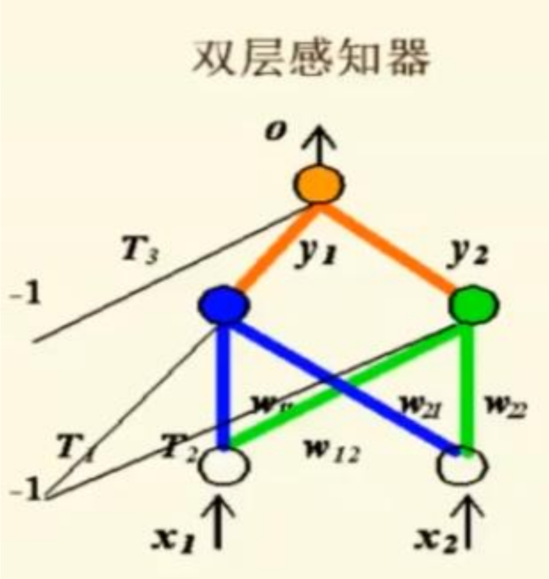


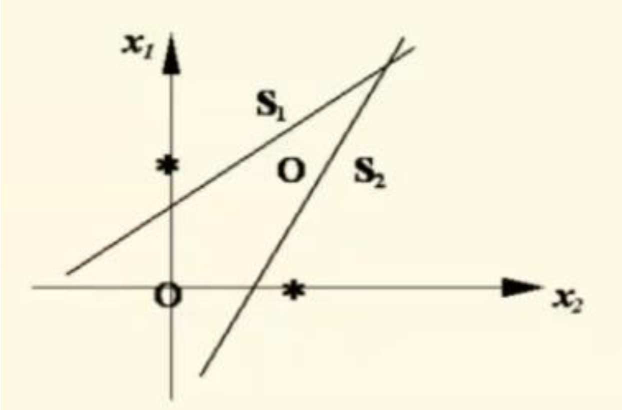
四、多层感知器

1、多层感知器功能



四、多层感知器





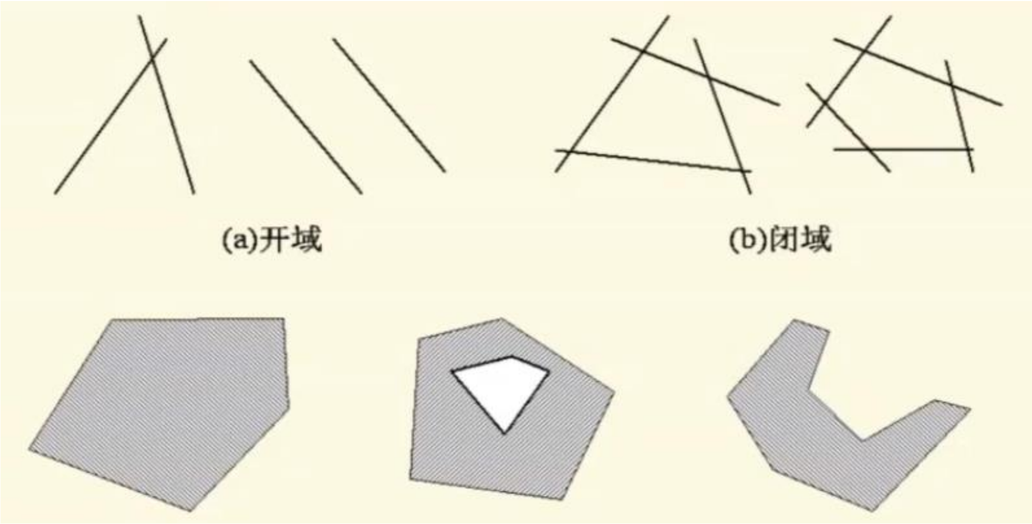
x1	x2	y1	y2	o
0	0	1		
0	1	1		
1	0	0		
1	1	1		

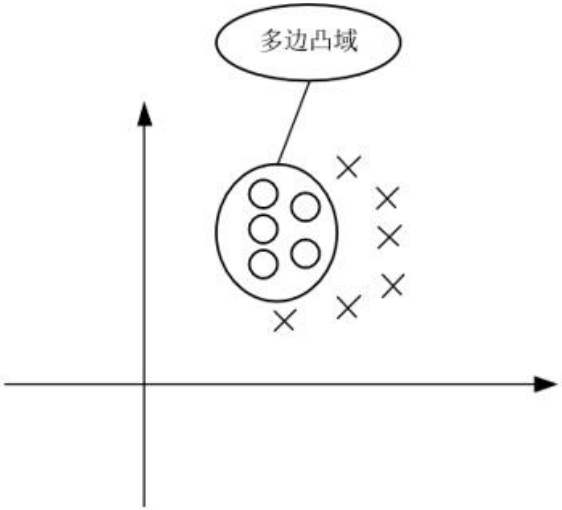
x1	x2	y1	y2	o
0	0		1	
0	1		0	
1	0		1	
1	1		1	

x1	x2	y1	y2	o
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	1	1	0

x1	x2		o
0	0		0
0	1		1
1	0		1
1	1		0

四、多层感知器





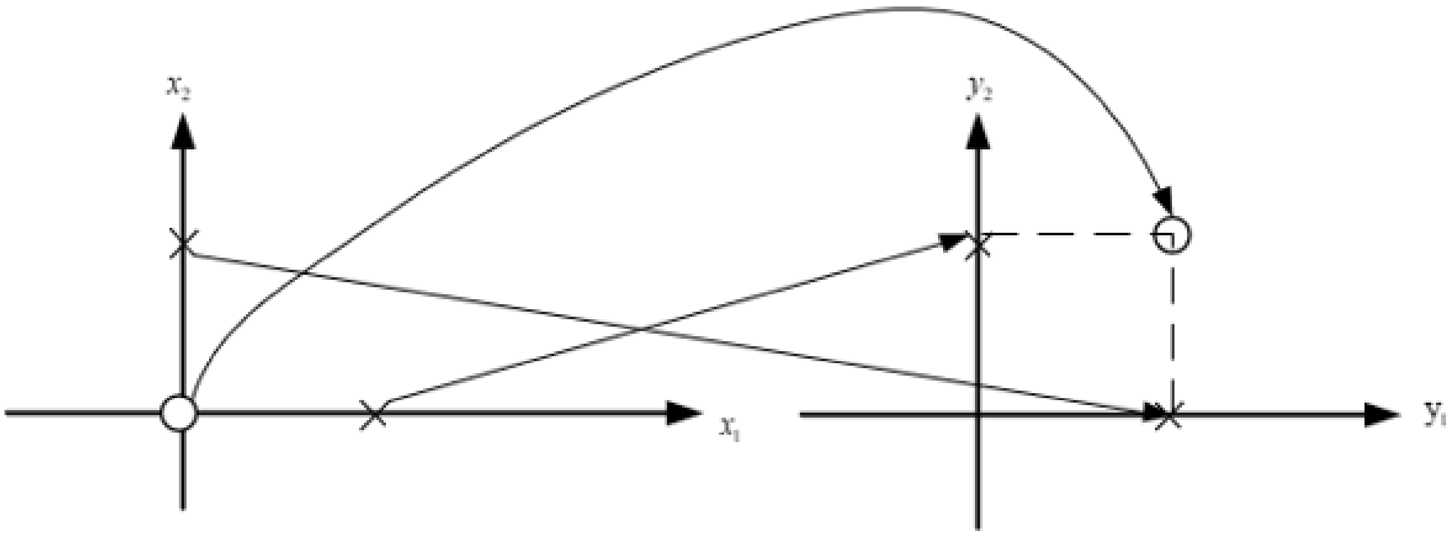
四、多层感知器

与非问题

x_1	x_2	y_1	y_2	0
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	1	1	0

隐藏

四、多层感知器



非线性变换的结果

四、多层感知器

2、多层感知器数学表达

输入向量：$X=(x_1,x_2,\dots,x_i,\dots,x_n)^T$

隐层输出向量：$Y=(y_1,y_2,\dots,y_j,\dots,y_m)^T$

输出层输出向量：$O=(o_1,o_2,\dots,o_k,\dots,o_l)^T$

期望输出向量：$d=(d_1, d_2,\dots,d_k,\dots,d_l)^T$

输入层到隐层之间的权值矩阵：$V=(V_1,V_2,\dots,V_j,\dots,V_m)$

隐层到输出层之间的权值矩阵：$W=(W_1,W_2,\dots,W_k,\dots,W_l)$

对于输出层：$o_k = f(net_k) \quad k=1,2,\dots,l$

$$net_k = \sum_{j=0}^{m} w_{jk} y_j \quad k=1,2,\dots,l$$

对于隐层：$y_j = f(net_j) \quad j=1,2,\dots,m$

$$net_j = \sum_{i=0}^{n} v_{ij} x_i \quad j=1,2,\dots,m$$

四、多层感知器

单极性Sigmoid函数：

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

Sigmoid函数的导数：

$$f'(x)=f(x)[1-f(x)]$$

双极性Sigmoid函数

$$f(x)=\frac{2}{1+e^{-x}}-1 \qquad f'(x)=\frac{1}{2}\left[1-f^2(x)\right]$$

四、多层感知器

3、网络误差与权值调整思路

输出误差E定义：$E=\frac{1}{2}(d-O)^2=\frac{1}{2}\sum_{k=1}^{l}(d_k-o_k)^2$

将以上误差定义式展开至隐层：

$$E=\frac{1}{2}\sum_{k=1}^{l}[d_k-f(net_k)]^2=\frac{1}{2}\sum_{k=1}^{l}[d_k-f(\sum_{j=0}^{m}w_{jk}y_j)]^2$$

进一步展开至输入层：

$$E=\frac{1}{2}\sum_{k=1}^{l}\{d_k-f[\sum_{j=0}^{m}w_{jk}f(net_j)]\}^2$$

$$=\frac{1}{2}\sum_{k=1}^{l}\{d_k-f[\sum_{j=0}^{m}\underline{w_{jk}}f(\sum_{i=0}^{n}\underline{v_{ij}}x_i)]\}^2$$

四、多层感知器

$$\Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}} \qquad j=0,1,2,\ldots,m;\ \ k=1,2,\ldots,l$$

$$\Delta v_{ij} = -\eta \frac{\partial E}{\partial v_{ij}} \qquad i=0,1,2,\ldots,n;\ \ j=1,2,\ldots,m$$

式中负号表示梯度下降，常数 $\eta \in (0,1)$表示比例系数。

在全部推导过程中，对输出层有$j=0,1,2,\ldots,m$； $k=1,2,\ldots,l$

对隐层有 $i=0,1,2,\ldots,n$； $j=1,2,\ldots,m$

四、多层感知器

4、BP算法推导

对于输出层，

$$\Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}} = -\eta \frac{\partial E}{\partial net_k} \frac{\partial net_k}{\partial w_{jk}}$$

对隐层，

$$\Delta v_{ij} = -\eta \frac{\partial E}{\partial v_{ij}} = -\eta \frac{\partial E}{\partial net_j} \frac{\partial net_j}{\partial v_{ij}}$$

对输出层和隐层各定义一个误差信号，令

$$\delta_k^o = -\frac{\partial E}{\partial net_k} \qquad \delta_j^y = -\frac{\partial E}{\partial net_j}$$

四、多层感知器

$$\Delta w_{jk} = \eta \delta_k^o y_j$$

$$\Delta v_{ij} = \eta \delta_j^y x_i$$

$$\delta_k^o = -\frac{\partial E}{\partial net_k} = -\frac{\partial E}{\partial o_k}\frac{\partial o_k}{\partial net_k} = -\frac{\partial E}{\partial o_k} f'(net_k)$$

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

对于输出层

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

$$\frac{\partial E}{\partial o_k} = -(d_k - o_k)$$

四、多层感知器

对于隐层

$$E=\frac{1}{2}\sum_{k=1}^{l}[d_k-f(\sum_{j=0}^{m}w_{jk}y_j)]^2$$

$$\frac{\partial E}{\partial y_j}=-\sum_{k=1}^{l}(d_k-o_k)f'(net_k)w_{jk}$$

$$\delta_k^o=\frac{\partial E}{\partial o_k}f'(net_k)$$

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

$$\delta_k^o=(d_k-o_k)o_k(1-o_k)$$

四、多层感知器

$$\delta_j^y = [\sum_{k=1}^{l}(d_k - o_k) f'(net_k) w_{jk}] f'(net_j)$$

$$= (\sum_{k=1}^{l} \delta_k^o w_{jk}) y_j (1 - y_j)$$