

模式识别

第9讲 神经网络模式识别

2018~2019学年



内容安排

一、绪论、数学基础（第1讲）

二、聚类分析（第2讲）

三、判别函数分类法（几何分类法）（第3、4讲）

四、统计决策分类法（概率分类法）（第5、6讲）

五、特征提取与选择（第7讲）

六、模糊模式识别（第8讲）

七、神经网络模式识别（第9讲）

期末考试（平时作业：40%，期末考试：60%）

七、神经网络模式识别

- 7.1 人工神经网络发展概况
- 7.2 神经网络基本概念
- 7.3 前馈神经网络
- 7.4 反馈网络模型Hopfield网络

7.1 人工神经网络发展概况

人工神经网络(Artificial Neural Networks, ANN):

简称神经网络。

模拟人脑神经细胞的工作特点:

- * 单元间的广泛连接;
- * 并行分布式的信息存贮与处理;
- * 自适应的学习能力等。

与目前按串行安排程序指令的计算机结构截然不同。

优点:

- (1) 较强的容错性;
- (2) 很强的自适应学习能力;
- (3) 可将识别和若干预处理融为一体进行;
- (4) 并行工作方式;
- (5) 对信息采用分布式记忆, 具有鲁棒性。

五个发展阶段：

第一阶段：启蒙期，始于1890年（Williams James）。

1943年：形式神经元的数学模型M-P模型提出；

1957年：感知器算法的此处（Frank Rosenblatt）

第二阶段：低潮期，始于1969年。

《感知器》(Perceptrons)一书出版，指出局限性。

第三阶段：复兴期，从1982年到1986年。

Hopfield的两篇论文提出新的神经网络模型；

《并行分布处理》出版，提出反向传播算法。

第四阶段：新兴期，1987年到2006年。

回顾性综述文章“神经网络与人工智能”。

第五阶段：深度学习热潮，2006年至今。

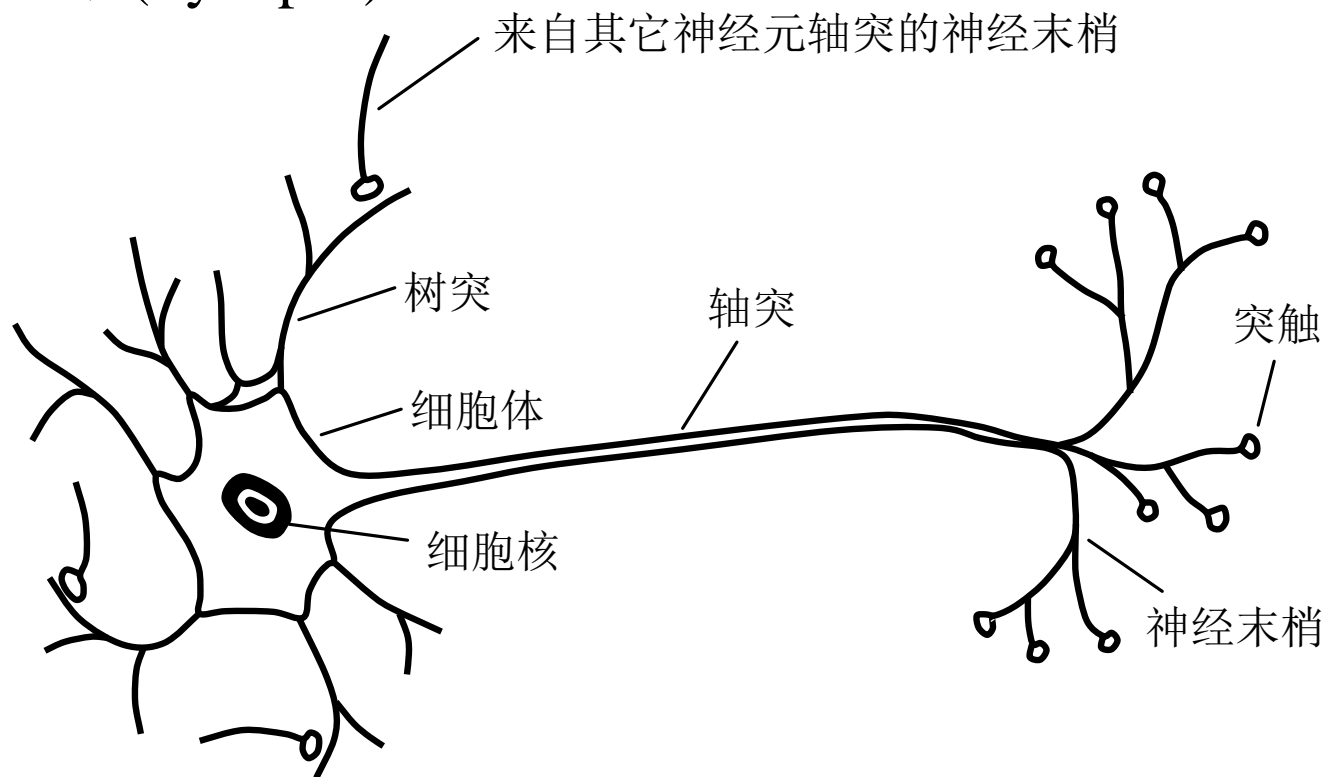
G. E. Hinton提出有效训练多层网络的随机NN算法。

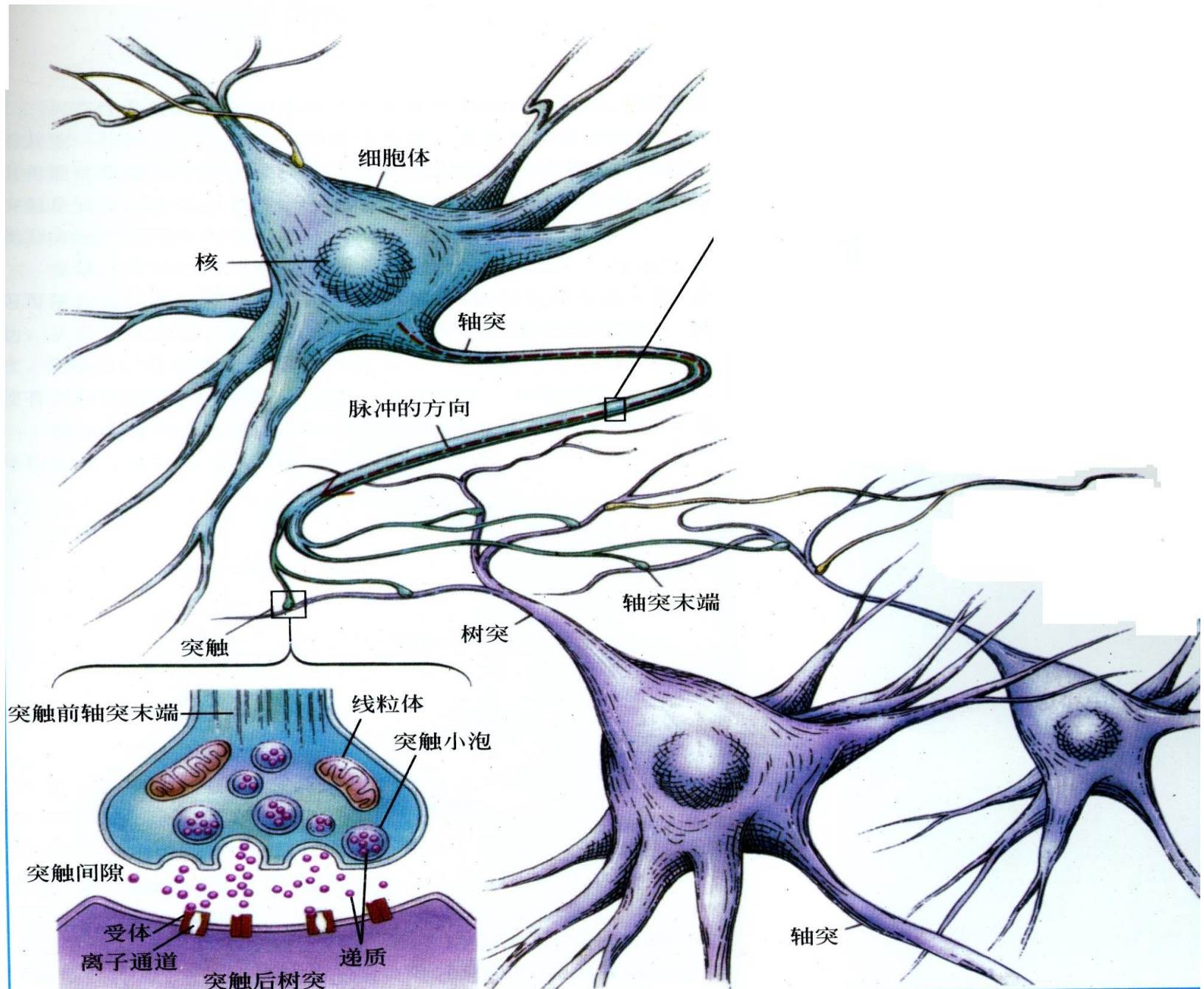
7.2 神经网络基本概念

7.2.1 生物神经元

1. 生物神经元的结构

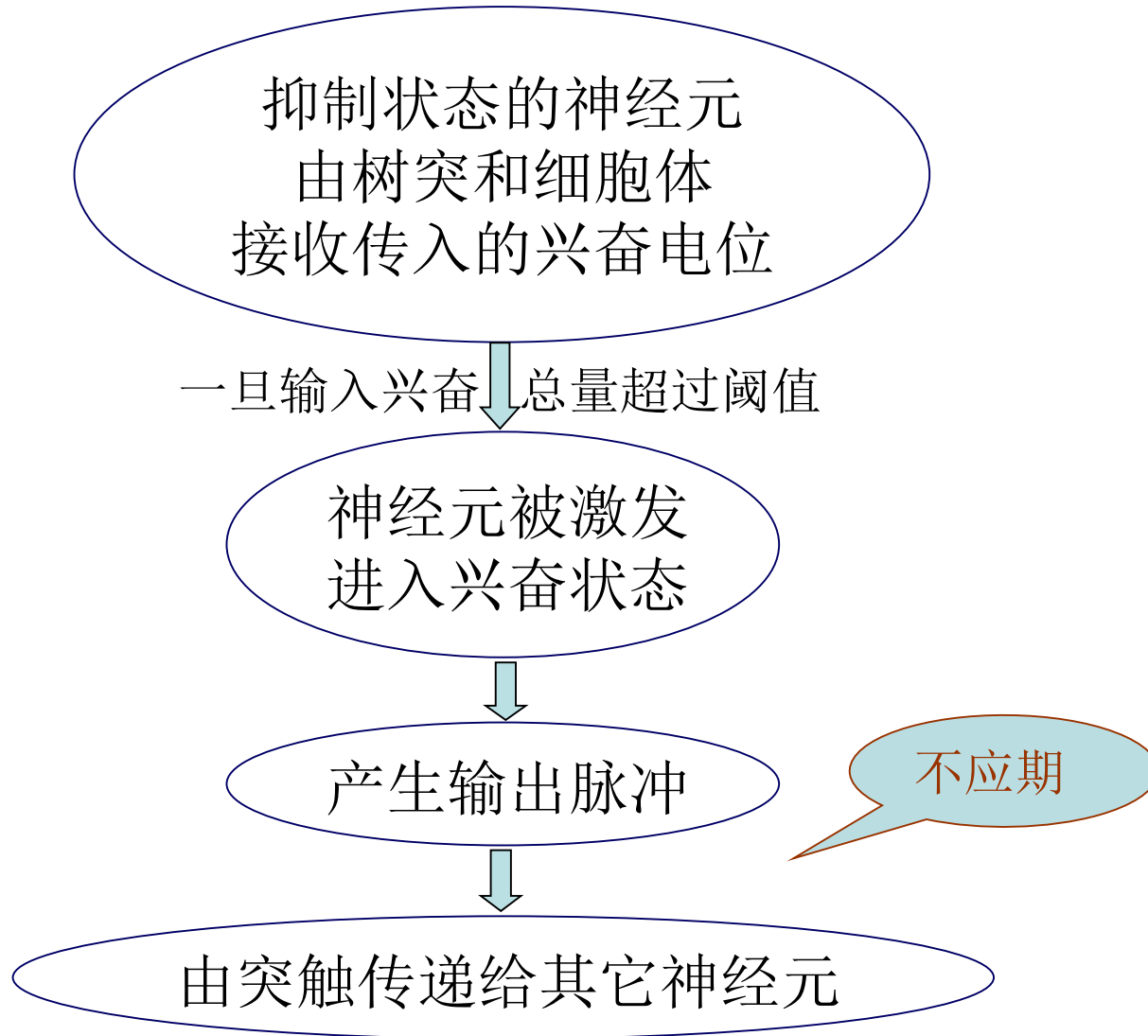
细胞体(Cell Body)、树突(Dendrite)、轴突(Axon)和突触(Synapse)。



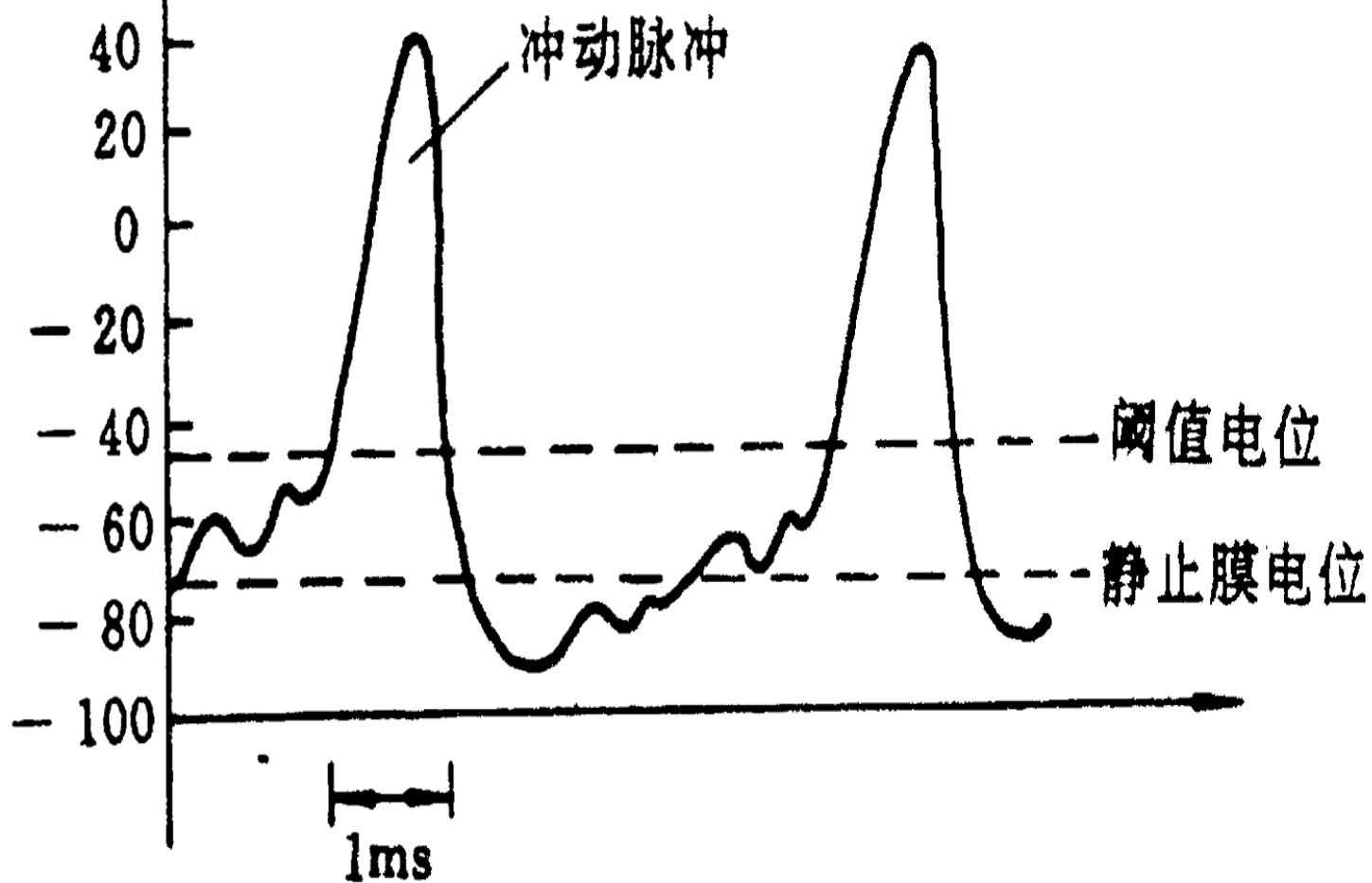


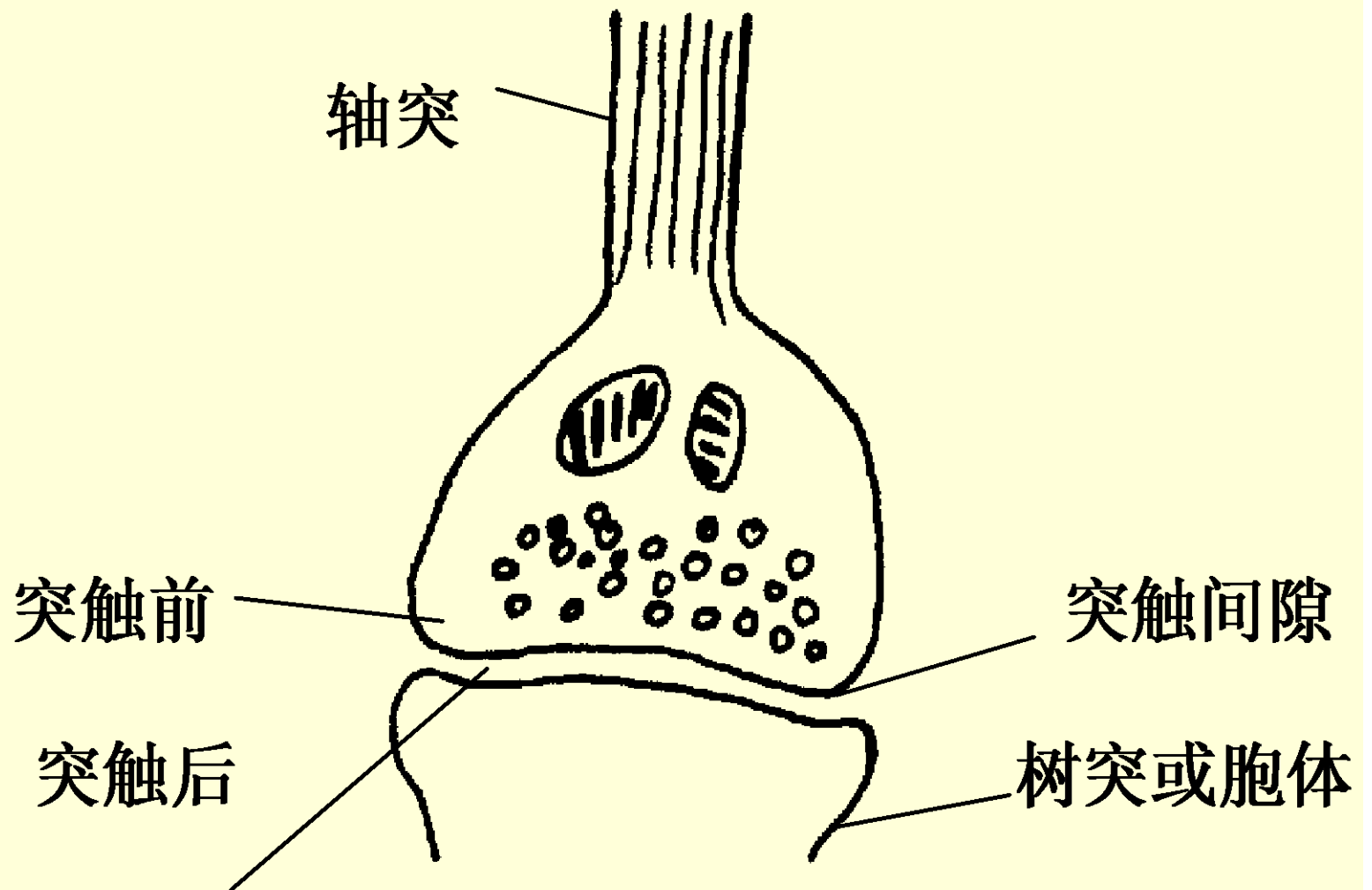
2. 生物神经元的工作机制

兴奋和抑制两种状态。



膜电位 (mV)





神经元的整合

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。

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。

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人工神经元（节点）。

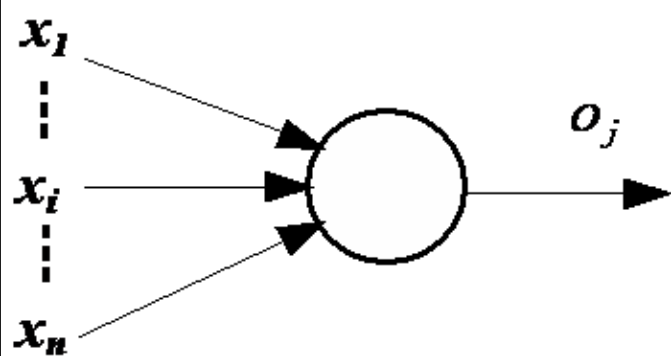
从三个方面进行模拟：

- 节点的信息处理能力（神经元模型）
- 节点与节点之间连接（网络拓扑结构）
- 节点相互连接的强度（通过学习调整）

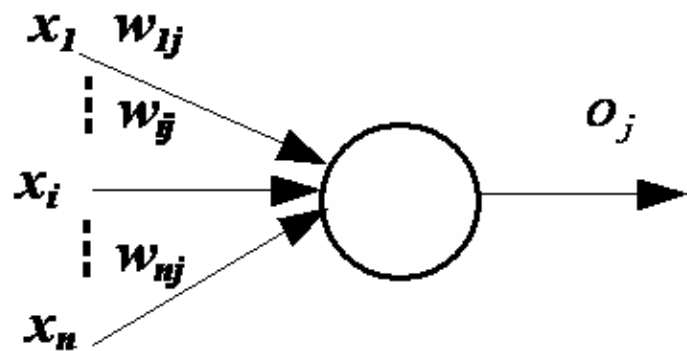
神经元模型的六点假设：

- (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激活特性；
- (4) 神经元输入与输出间有固定时滞, 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。

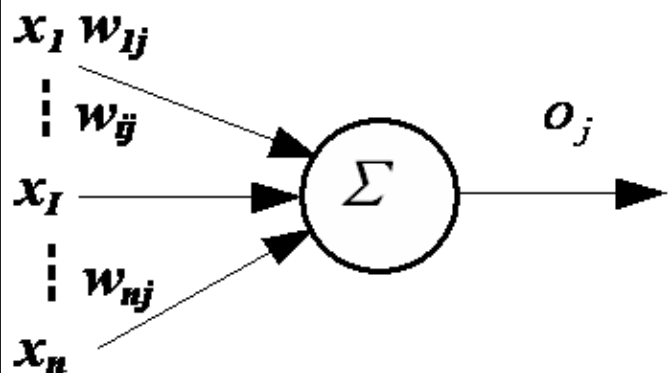
神经元模型示意图



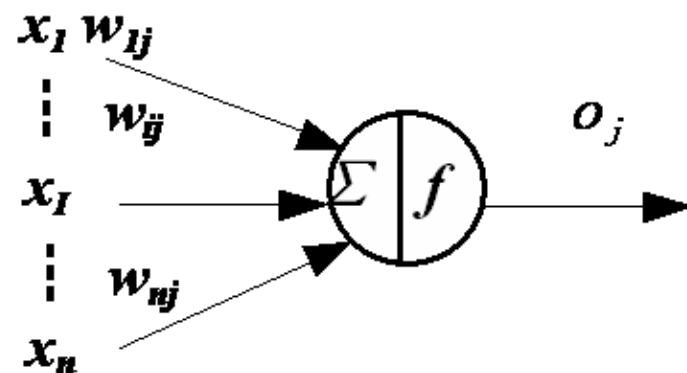
(a)多输入单输出



(b)输入加权



(c)输入加权求和



(d)输入-输出函数

7.2.2 人工神经元及神经网络

人工神经元：生物神经元的简化模拟。

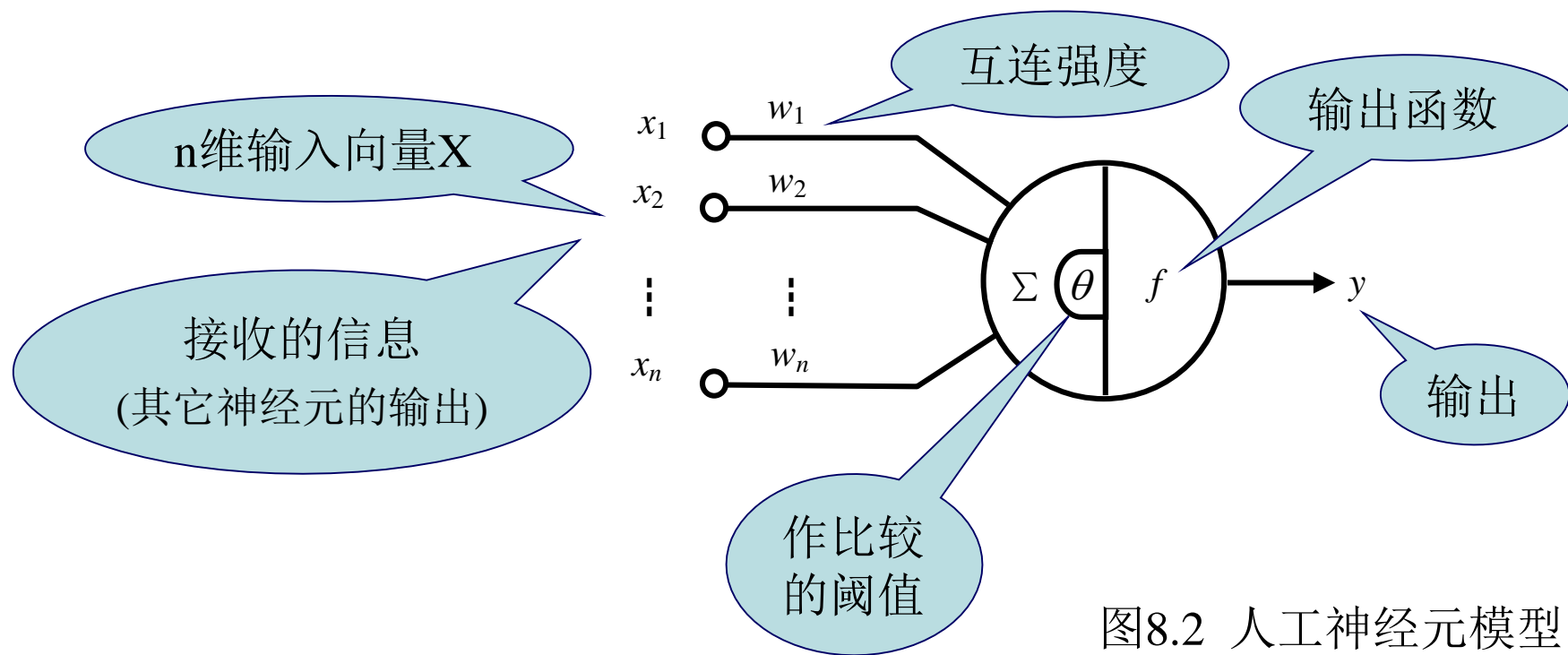


图8.2 人工神经元模型

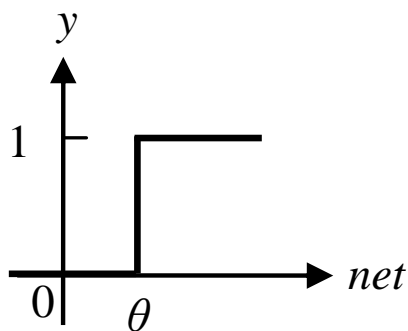
神经元互连：信息传递路径轴突-突触-树突的简化；

连接的权值：互连神经元间突触性质与强度的模拟。

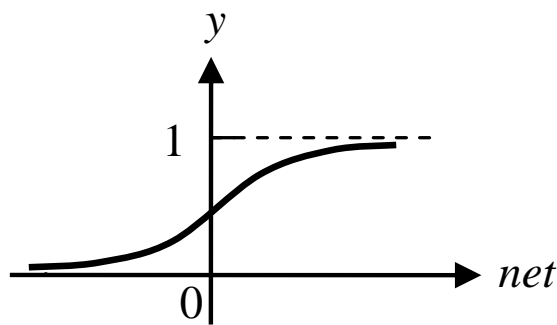
神经元动作模型: $net = \sum_{i=1}^n w_i x_i \quad (x_i, w_i \in R)$

$$y = f(net)$$

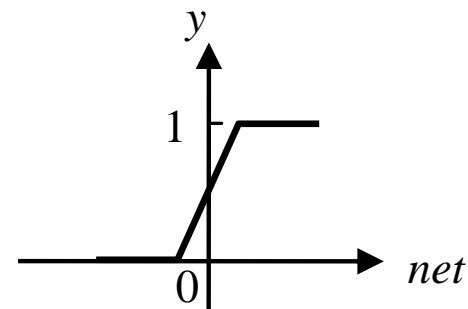
输出函数 f : 也称作用函数、激活函数, 一般是非线性。



(a)
阈值型



(b)
S型



(c)
伪线性型

f 为阈值型函数时: $y = \text{sgn}\left(\sum_{i=1}^n w_i x_i - \theta\right)$

设 $w_{n+1} = -\theta$, 点积形式: $y = \text{sgn}(\mathbf{W}^T \mathbf{X})$

式中, $\mathbf{W} = [w_1, \Lambda, w_n, w_{n+1}]^T$ $\mathbf{X} = [x_1, \Lambda, x_n, 1]^T$

7.2.3 神经网络的学习

学习：

神经网络的最重要特征之一。

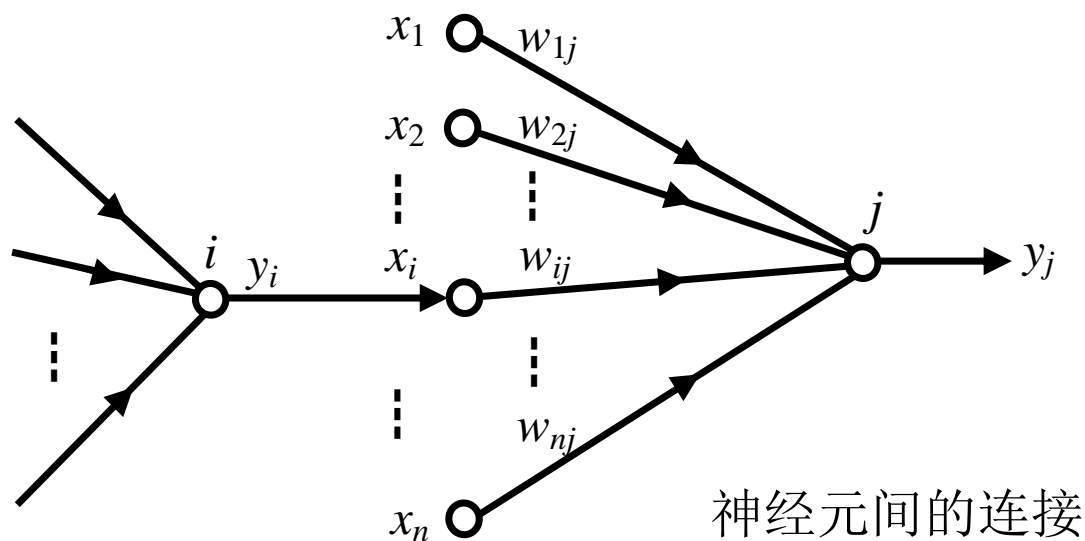
实质：

同一类训练集的样本输入输出模式反复作用于网络，网络按照一定的训练规则自动调节神经元之间的连接强度或拓扑结构，使实际输出满足期望的要求或者趋于稳定。

典型的权值修正方法： Hebb学习规则、误差修正学习(**δ 学习**)

1. Hebb学习规则

如果神经网络中某一神经元与另一直接与其相连的神经元同时处于兴奋状态，那么这两个神经元之间的连接强度应该加强。



$$w_{ij}(t+1) = w_{ij}(t) + \eta[y_j(t)y_i(t)]$$

$w_{ij}(t+1)$: 当前时刻 t 修正后的权值;

η : 学习因子, 表示学习速率的比例常数;

$y_j(t), y_i(t)$: 分别表示 t 时刻第 j 个和第 i 个神经元的状态 (输出)。

由于 $x_i(t)=y_i(t)$ 有:

$$w_{ij}(t+1) = w_{ij}(t) + \eta[y_j(t)x_i(t)]$$

2. δ 学习规则

- (1) 选择一组初始权值 $w_{ij}(1)$;
- (2) 计算某一输入模式对应的实际输出与期望输出的误差;
- (3) 更新权值, 阈值可视为输入恒为 (-1) 的一个权值;

$$w_{ij}(t+1) = w_{ij}(t) + \eta[d_j - y_j(t)]x_i(t)$$

式中, η : 学习因子;

$d_j, y_j(t)$: 第 j 个神经元的期望输出与实际输出;

$x_i(t)$: 第 j 个神经元的第 i 个输入。

- (4) 返回 (2), 直到对所有训练模式网络输出均能满足要求。

神经网络的学习体现在:

权值变化;

网络结构变化。

7.2.4 神经网络的结构分类

分层结构

有明显层次，信息流向由输入层到输出层。

—— 前馈网络

互连结构

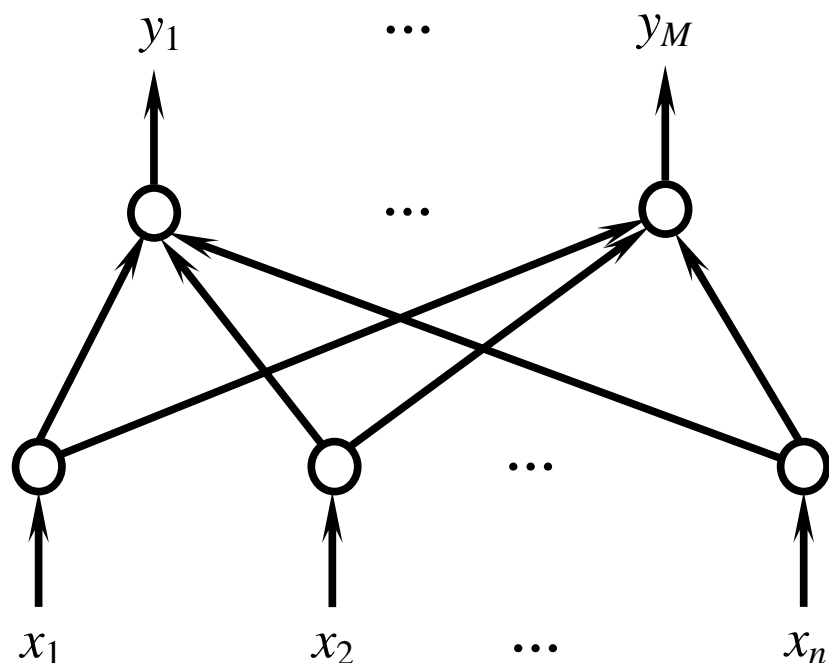
没有明显层次，任意两个神经元之间可连接，具有输出单元到隐层单元或输入单元的反馈连接。

—— 反馈网络

7.3 前馈神经网络

7.3.1 感知器

感知器（Perceptron）：Frank Rosenblatt于1957年提出。



感知器结构示意图

结构特点：

- * 双层（输入层、输出层）；
- * 两层单元之间为全互连；
- * 连接权值可调。
- * 输出层神经元个数等于类别数（两类问题时输出层为一个神经元）。

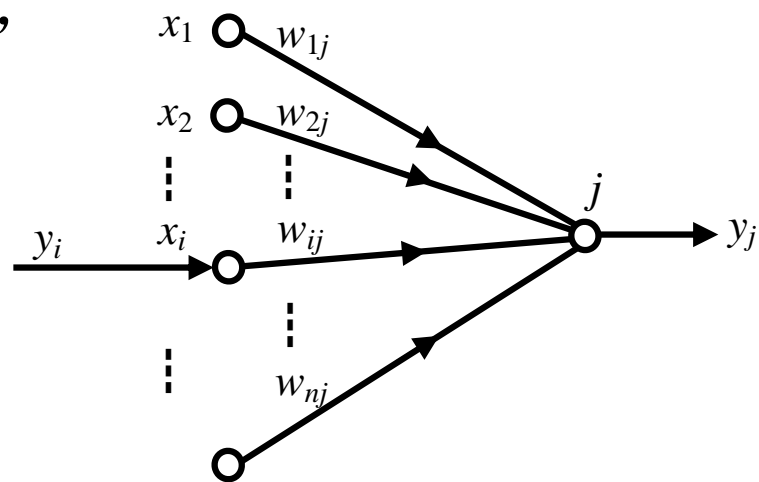
设输入模式， $\mathbf{X} = [x_1, x_2, \Lambda, x_n]^T$ ，共 M 类(对应 M 个输出节点)。

输出层第 j 个神经元对应第 j 个模式类，

$$\text{输出为 } y_j = f\left(\sum_{i=1}^n w_{ij} x_i - \theta_j\right)$$

θ_j : 第 j 个神经元的动作阈值；

w_{ij} : 输入模式第 i 个分量与
输出层第 j 个神经元间的连接权。



输出单元对所有输入数值加权求和，经阈值型输出函数产生一组输出模式。

令 $w_{(n+1)j} \equiv -\theta_j$ 。取

$$\mathbf{W}_j = [w_{1j}, w_{2j}, \Lambda, w_{(n+1)j}]^T \quad \mathbf{X} = [x_1, x_2, \Lambda, x_n, 1]^T$$

有

$$y_j = f\left(\sum_{i=1}^{n+1} w_{ij} x_i\right) = f(\mathbf{W}_j^T \mathbf{X})$$

M 类问题判决规则(神经元的输出函数) 为

$$y_j = f(\mathbf{W}_j^T \mathbf{X}) = \begin{cases} +1, & \text{若 } \mathbf{X} \in \omega_j \\ -1, & \text{若 } \mathbf{X} \notin \omega_j \end{cases} \quad 1 \leq j \leq M$$

* 正确判决的关键:

输出层每个神经元必须有一组合适的权值。

* 感知器采用监督学习算法得到权值;

* 权值更新方法: δ 学习规则。

算法描述

第一步: 设置初始权值 $w_{ij}(1)$, $w_{(n+1)j}(1)$ 为第 j 个神经元的阈值。

第二步: 输入新的模式向量。

第三步: 计算神经元的实际输出。

设第 k 次输入的模式向量为 \mathbf{X}_k ，与第 j 个神经元相连的权向量为

$$\mathbf{W}_j(k) = [w_{1j}, w_{2j}, \Lambda, w_{(n+1)j}]^T$$

第 j 个神经元的实际输出为

$$y_j(k) = f[\mathbf{W}_j^T(k) \mathbf{X}_k] \quad 1 \leq j \leq M$$

第四步：修正权值。

$$\mathbf{W}_j(k+1) = \mathbf{W}_j(k) + \eta[d_j - y_j(k)]\mathbf{X}_k$$

d_j : 第 j 个神经元的期望输出。

$$d_j = \begin{cases} +1, & \mathbf{X}_k \in \omega_j \\ -1, & \mathbf{X}_k \notin \omega_j \end{cases} \quad 1 \leq j \leq M$$

第五步：转到第二步。

当全部学习样本都能正确分类时，学习过程结束。

经验证明，当 η 随 k 的增加而减小时，算法一定收敛。

7.3.2 BP网络（前馈网络的反向传播算法）

BP网络：采用BP算法（Back-Propagation Training Algorithm）
的多层感知器。

误差反向传播算法

认识最清楚、应用最广泛。

性能优势：识别分类

1. 多层感知器

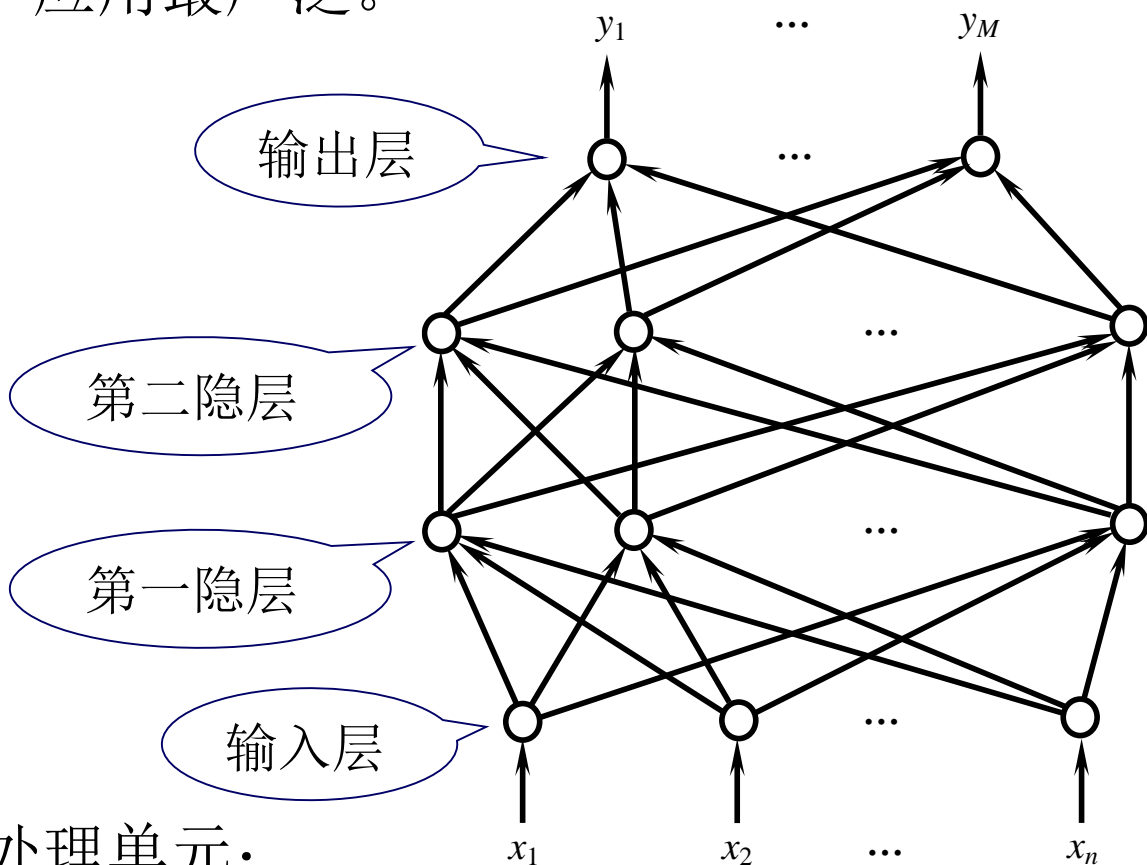
针对感知器学习
算法的局限性：模式
类必须线性可分。

结构：

前馈网络；

中间层为一层或多层处理单元；

只允许一层连接权可调。



2. BP算法

两个阶段 { **正向传播阶段**: 模式信息前向传播, 逐层更新状态
反向传播阶段: 误差信息反向传播, 逐层校正权值

BP算法的学习过程

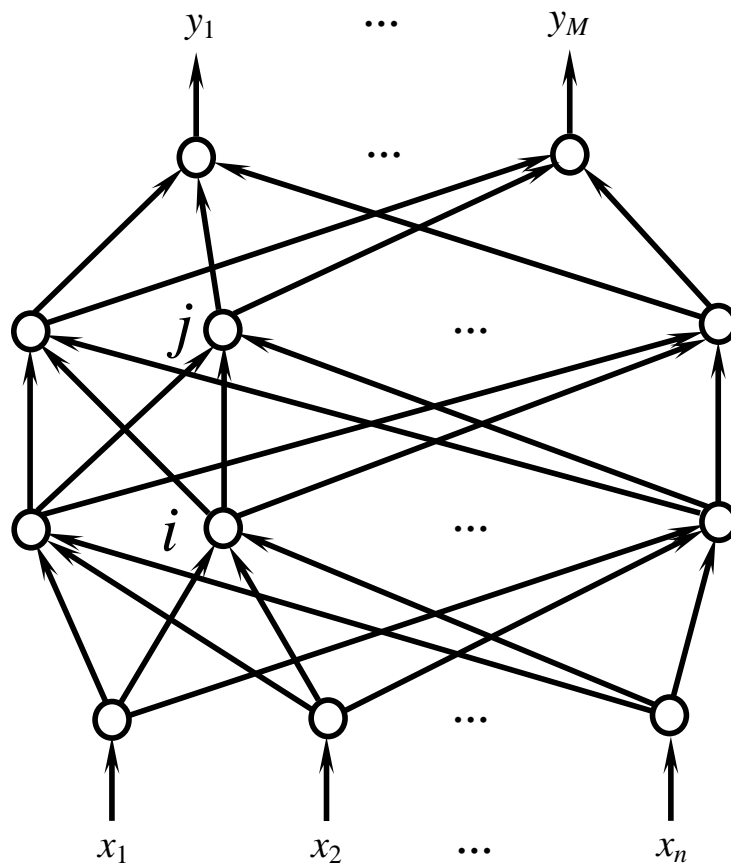
设: 某层任一神经元 j 的
净输入为 net_j , 输出为 y_j ;
相邻低一层中任一
神经元 i 的输出为 y_i 。

$$net_j = \sum_i w_{ij} y_i$$

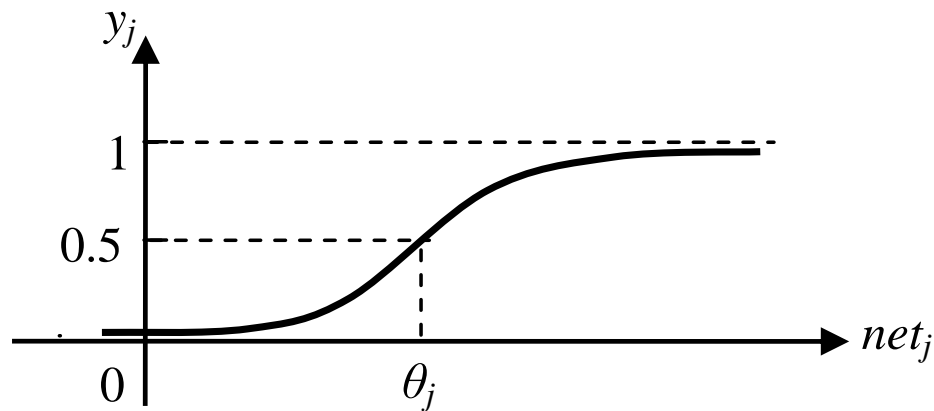
$$y_j = f(net_j)$$

w_{ij} : 前一层神经元 i 与
当前层神经元 j 之间的连接权;

$f(\cdot)$: 神经元的输出函数。



Sigmoid型（S型）输出函数：



$$y_j = f(net_j) = \frac{1}{1 + e^{-(net_j + \theta_j)/h_0}}$$

θ_j : 神经元动作阈值；

h_0 : 输出函数的形状参数（值越小，越陡峭）。

设：输出层中第 k 个神经元的实际输出为 y_k ，输入为 net_k ；
与输出层相邻的隐含层中任一神经元 j 的输出为 y_j 。

$$net_k = \sum_j w_{jk} y_j$$

$$y_k = f(net_k)$$

(1) 对输入模式 \mathbf{X}_p ，若输出层中第 k 个神经元的期望输出为 d_{pk} ，实际输出为 y_{pk} 。输出层的输出平方误差：

$$E_p = \frac{1}{2} \sum_k (d_{pk} - y_{pk})^2$$

若输入 N 个模式，网络的系统均方误差为：

$$E @ \frac{1}{N} \sum_p E_p = \frac{1}{2N} \sum_p \sum_k (d_{pk} - y_{pk})^2$$

当输入 \mathbf{X}_p 时， w_{jk} 的修正增量： $\Delta_p w_{jk} = -\eta \frac{\partial E_p}{\partial w_{jk}}$

其中， $-\frac{\partial E_p}{\partial w_{jk}} = -\frac{\partial E_p}{\partial net_k} \cdot \frac{\partial net_k}{\partial w_{jk}}$

由 $net_k = \sum_j w_{jk} y_j$ 式得到： $\frac{\partial net_k}{\partial w_{jk}} = \frac{\partial}{\partial w_{jk}} \sum_j w_{jk} y_{pj} = y_{pj}$

令 $\delta_{pk} @ -\partial E_p / \partial net_k$ 可得（请自行验算）

输出单元的误差： $\delta_{pk} = (d_{pk} - y_{pk})y_{pk}(1 - y_{pk})$

输出单元的修正增量： $\Delta_p \omega_{jk} = \eta \delta_{pk} y_{pj}$

（2）对于与输出层相邻的隐层中的神经元 j 和该隐含层前低一层中的神经元 i ：

$$\delta_{pj} = y_{pj}(1 - y_{pj}) \sum_k \delta_{pk} w_{jk}$$

$$\Delta_p w_{ij} = \eta \delta_{pj} y_{pi}$$

输出层中神经元输出的误差反向传播到前面各层，对各层之间的权值进行修正。

BP算法步骤:

第一步：对权值和神经元阈值初始化：(0, 1)上分布的随机数。

第二步：输入样本，指定输出层各神经元的希望输出值。

$$d_j = \begin{cases} +1, & \mathbf{X} \in \omega_j \\ -1, & \mathbf{X} \notin \omega_j \end{cases} \quad j = 1, 2, \Lambda, M$$

第三步：依次计算每层神经元的实际输出，直到输出层。

第四步：从输出层开始修正每个权值，直到第一隐含层。

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j y_i \quad 0 < \eta < 1$$

1. 若 j 是输出层神经元，则： $\delta_j = y_j(1 - y_j)(d_j - y_j)$

2. 若 j 是隐含层神经元，则： $\delta_j = y_j(1 - y_j) \sum_k \delta_k w_{jk}$

第五步：转到第二步（输入下一个样本），直至权值稳定。

改进的权值修正：

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j y_i + \alpha [w_{ij}(t) - w_{ij}(t-1)]$$

α ：平滑因子， $0 < \alpha < 1$ 。

—— 收敛快、权值平滑变化

BP算法存在问题：

- * 存在局部极小值问题；
- * 算法收敛速度慢；
- * 隐层单元数目的选取无一般指导原则；
- * 新加入的学习样本影响已学完样本的学习结果。

7.3.3 竞争学习神经网络

1. 竞争学习

典型的非监督学习策略。

结构特点：

与二层前馈网络类似；

输出层具有侧抑制。

竞争层：
竞争学习
网络的核心

侧抑制：

* 输出层各单元之间相互用较大的负权值输入对方，构成正反馈。

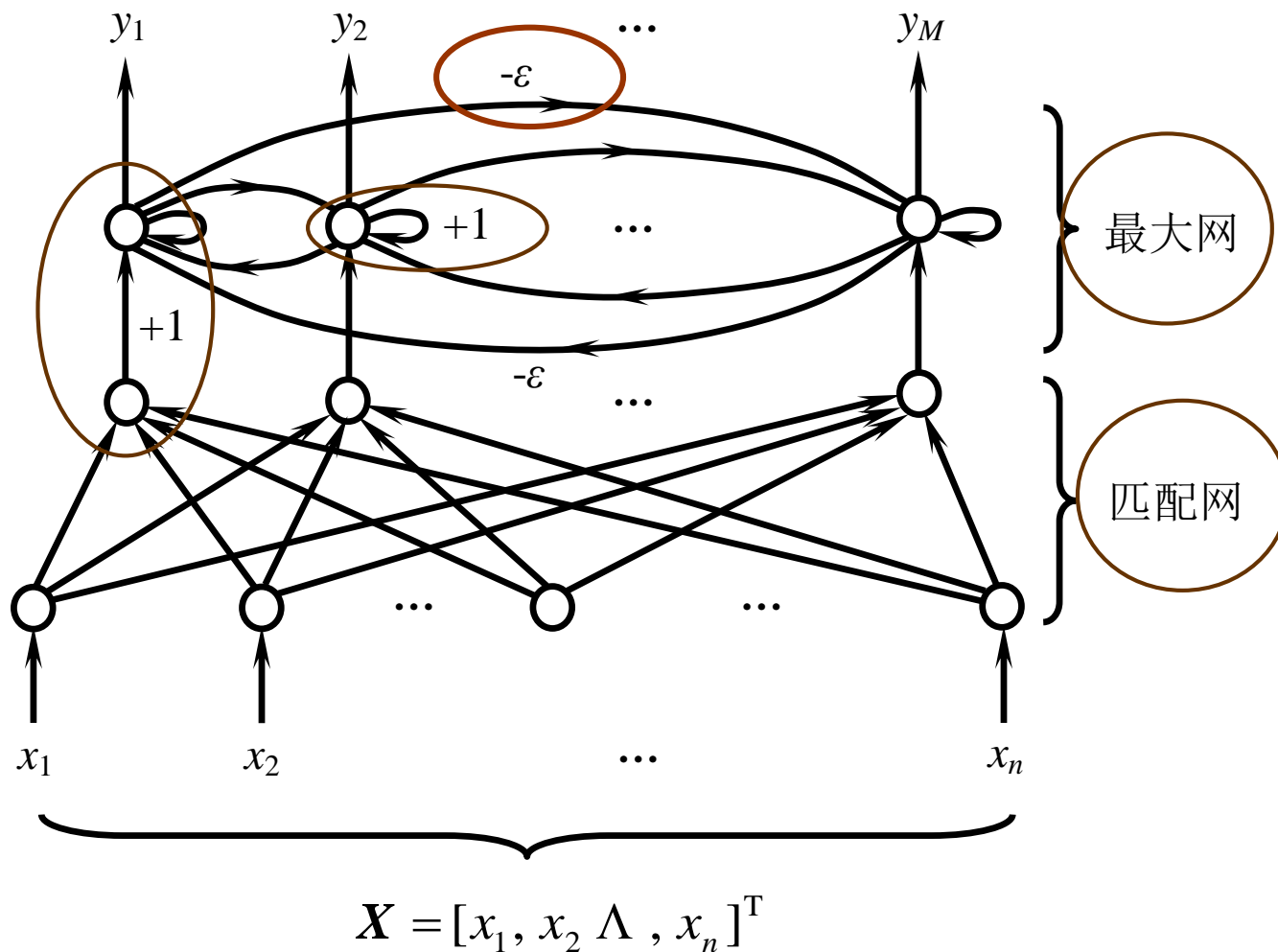
加强自身

* 具有最高输入总和的单元的输出状态为1，其他单元为0。

2. 汉明（Hamming）网分类器

仿效生物神经网络“中心激励，侧向抑制”的功能。

结构：



分类准则：样本间汉明距离最小。

工作原理：

二值模式向量
(分量：+1, -1)

- * 每个模式类由一个典型样本代表；
- * 匹配网计算输入样本与各类典型样本的匹配度，由匹配度决定匹配网的输出；
- * 由最大网给出输入样本所在类别号（分类）。

匹配度 = n - 输入样本与典型样本之间的汉明距离（**请验证**）

$$= n - \frac{1}{2} \left(n - \sum_{i=1}^n x_i x_i^j \right) = \sum_{i=1}^n \frac{x_i^j}{2} x_i + \frac{n}{2}$$

x_i ：输入样本的第 i 个分量；

x_i^j ：第 j 类典型样本的第 i 个分量；

n ：样本向量的维数。

输入样本与典型样本越相似：
汉明距离越小，
匹配度越大。

匹配网上层每个神经元的输出：

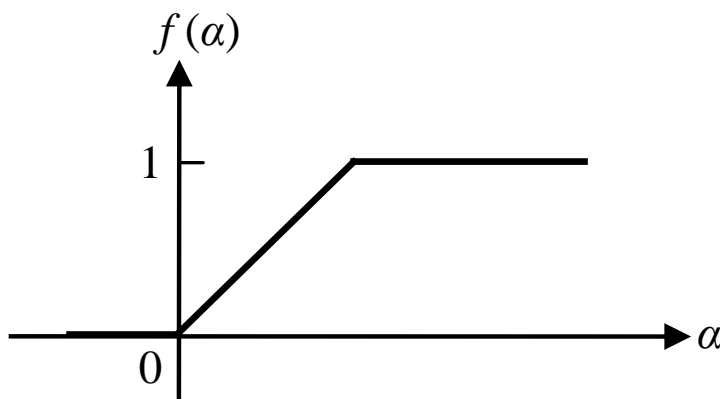
$$s_j = f\left(\sum_{i=1}^n w_{ij} x_i + \theta_j\right)$$

$w_{ij} = x_i^j / 2$ ：输入样本第*i*个分量与匹配网上层第*j*个神经元的连接权；

$\theta_j = n/2$ ：第*j*个神经元的阈值。

w_{ij} 由第*j*类典型样本的各分量确定。

匹配网输出函数 $f(\cdot)$ ：



汉明网算法步骤:

第一步: 设置权值和神经元阈值 (由已知类别的代表样本)。

$$w_{ij} = \frac{x_i^j}{2} \quad \theta_j = \frac{n}{2} \quad 1 \leq i \leq n \quad 1 \leq j \leq M$$

x_i^j : 第 j 类典型样本的第 i 个分量;

w_{ij} : 匹配网上层神经元 j 和输入样本第 i 个分量的连接权;

θ_j : 神经元 j 的阈值。

$$\omega_{lk} = \begin{cases} 1, & k = l \\ -\varepsilon, & k \neq l \end{cases} \quad \varepsilon = \frac{1}{M} \quad 1 \leq k \leq M \quad 1 \leq l \leq M$$

ω_{lk} : 最大网中第 l 个神经元和第 k 个神经元的连接权;

最大网中神经元的阈值为零。

第二步：输入未知样本，计算匹配网上层各神经元的输出 s_j ，
设置最大网中神经元输出的初始值。

设最大网中第 j 个神经元在 t 时刻的输出为 $y(t)$ ，则

$$s_j = f\left(\sum_{i=1}^n w_{ij} x_i + \theta_j\right) \quad 1 \leq j \leq M$$

$$y_j(0) = s_j \quad 1 \leq j \leq M$$

第三步，进行迭代运算直到收敛。判别：最大输出值的节点号即为未知样本的类别号。

$$y_j(t+1) = f\left[y_j(t) - \varepsilon \sum_{k=1}^M y_k(t)\right] \quad 1 \leq j \leq M$$

第四步：转到第二步，对下一个 ^{$k \neq j$} 未知样本进行分类识别。

3. 自组织特征映射神经网络（SOM网络）

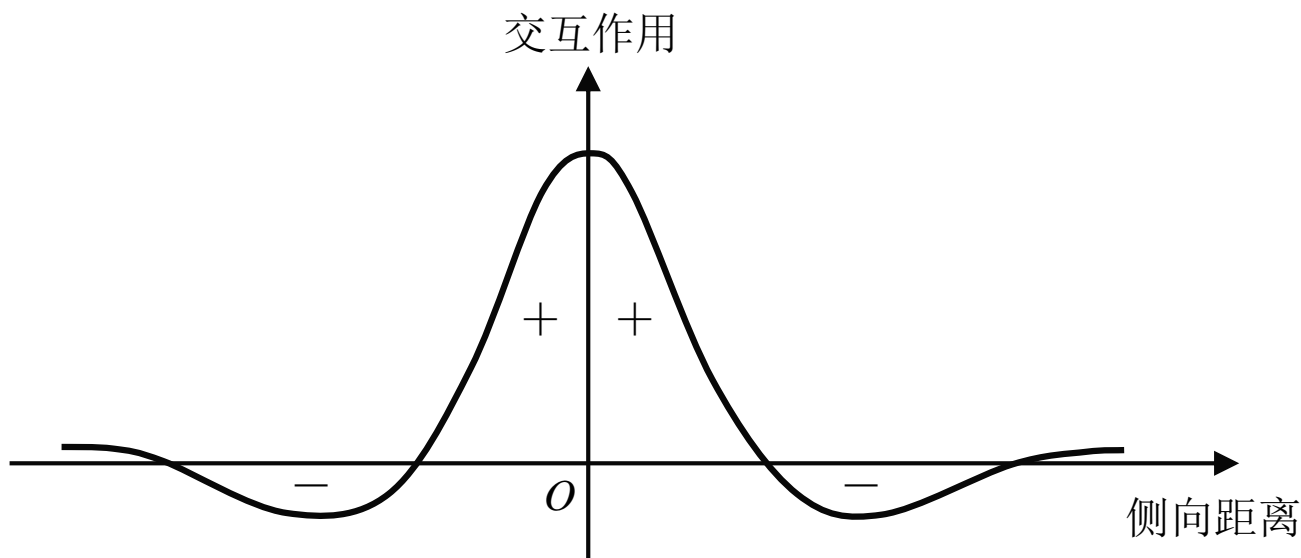
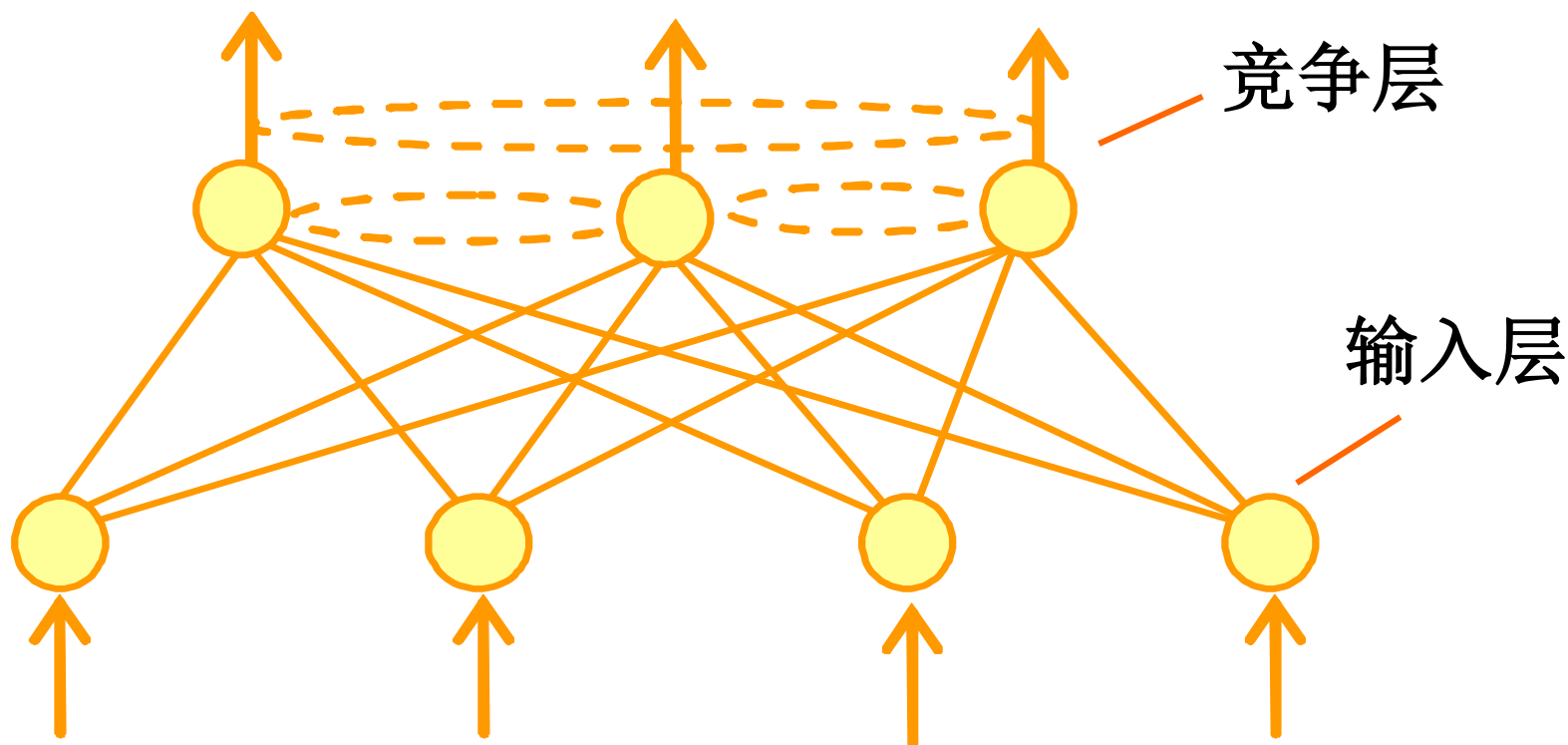


图8.10 神经元之间相互作用与距离的关系

T. Kohonen关于自组织特征映射的含义：

神经网络中邻近的各神经元通过侧向交互作用彼此相互竞争，自适应地发展成检测不同信号的特殊检测器。



自组织神经网络的典型结构

自组织学习 (Self-Organized Learning) :

通过自动寻找样本中的内在规律和本质属性，自组织、自适应地改变网络参数与结构。

自组织网络的自组织功能是通过竞争学习 (competitive learning) 实现的。

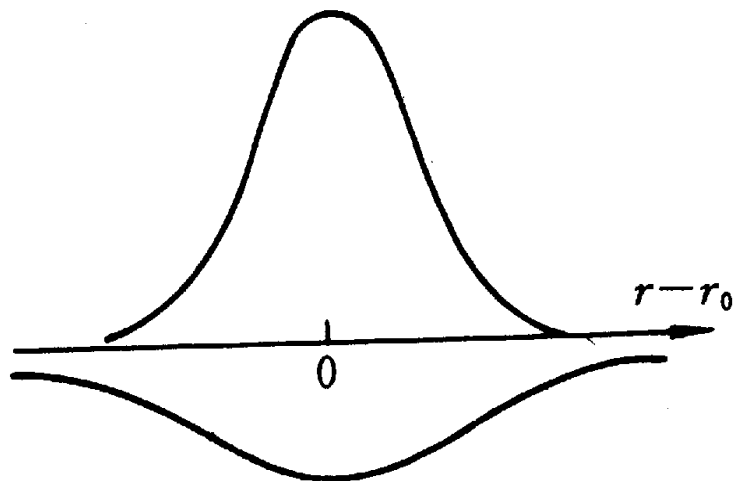
竞争学习规则——赢家通吃(Winner-Take-All)

网络的输出神经元之间相互竞争以求被激活，结果在每一时刻只有一个输出神经元被激活。这个被激活的神经元称为竞争获胜神经元，而其它神经元的状态被抑制，故称为Winner Take All。

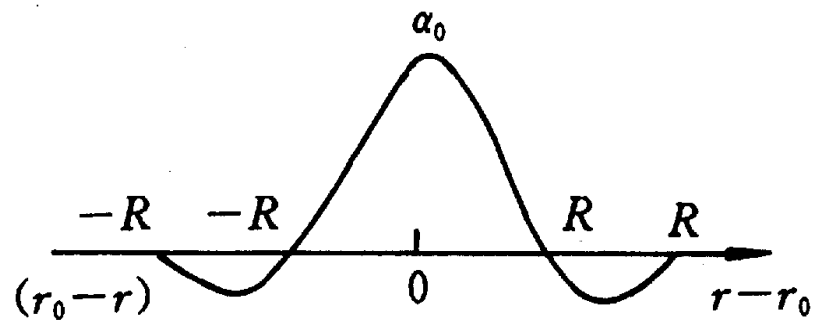
- 1981年芬兰Helsinki大学的 Kohonen 教授提出一种自组织特征映射网，简称SOM网。又被称为Kohonen网。
- Kohonen认为：一个神经网络接受外界输入模式时，将会分为不同的对应区域，各区域对输入模式具有不同的响应特征，而且这个过程是自动完成的。自组织特征映射正是根据这一看法提出来的，其特点与人脑的自组织特性相类似。

SOM网的权值调整域

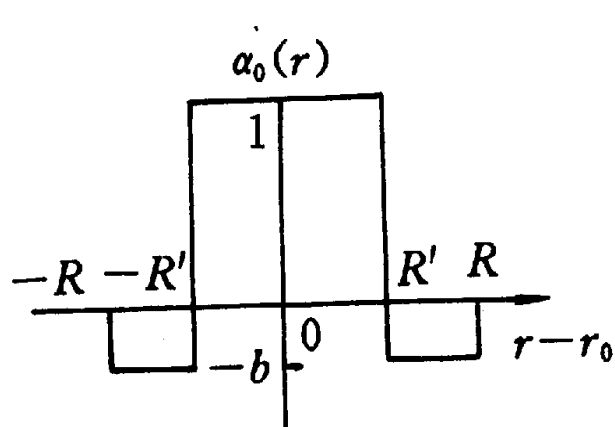
SOM网的获胜神经元对其邻近神经元的影响是由近及远，由兴奋逐渐转变为抑制，因此其学习算法中不仅获胜神经元本身要调整权向量，它周围的神经元在其影响下也要程度不同地调整权向量。这种调整可用三种函数表示：



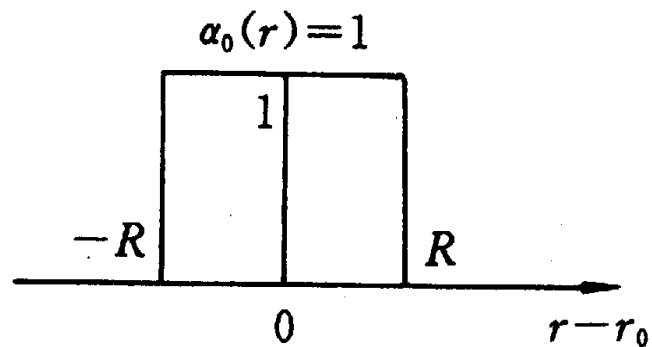
(a)



(b)



(c)



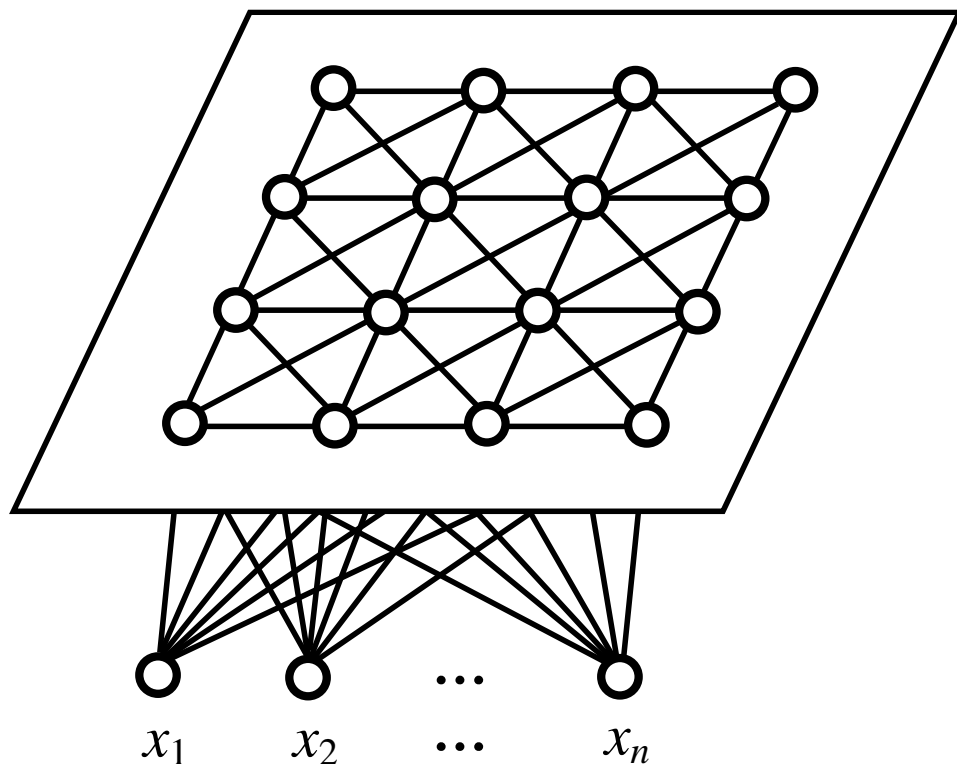
(d)

SOM网的权值调整域

以获胜神经元为中心设定一个邻域半径，该半径圈定的范围称为**优胜邻域**。在SOM网学习算法中，优胜邻域内的所有神经元均按其离开获胜神经元的距离远近不同程度地调整权值。

优胜邻域开始定得很大，但其大小随着训练次数的增加不断收缩，最终收缩到半径为零。

SOM网络结构:



输入层：每个神经元与输出层所有神经元连接。

输入连续值模式向量。

输出层：广泛连接，格阵形式。

竞争学习算法：由交互作用函数取代简单的侧抑制。

自组织特征映射算法步骤:

第一步: 设置初始权值, 定义输出层神经元的邻域。

第二步: 输入新的模式向量 $\mathbf{X} = [x_1, x_2, \dots, x_n]^T$ 。

第三步: 计算输入模式到每个输出层神经元 j 的距离 d_j 。

$$d_j = \sum_{i=1}^n [x_i(t) - w_{ij}(t)]^2 \quad 1 \leq j \leq M$$

$w_{ij}(t)$: t 时刻输入层神经元 i 到输出层神经元 j 之间的连接权。

第四步: 选择与输入模式距离最小的输出层神经元 j^* 。

第五步: 修改与 j^* 及其邻域中神经元连接的权值。

设 t 时刻神经元 j^* 的邻域用 $NE_{j^*}(t)$ 表示, 权值修改为:

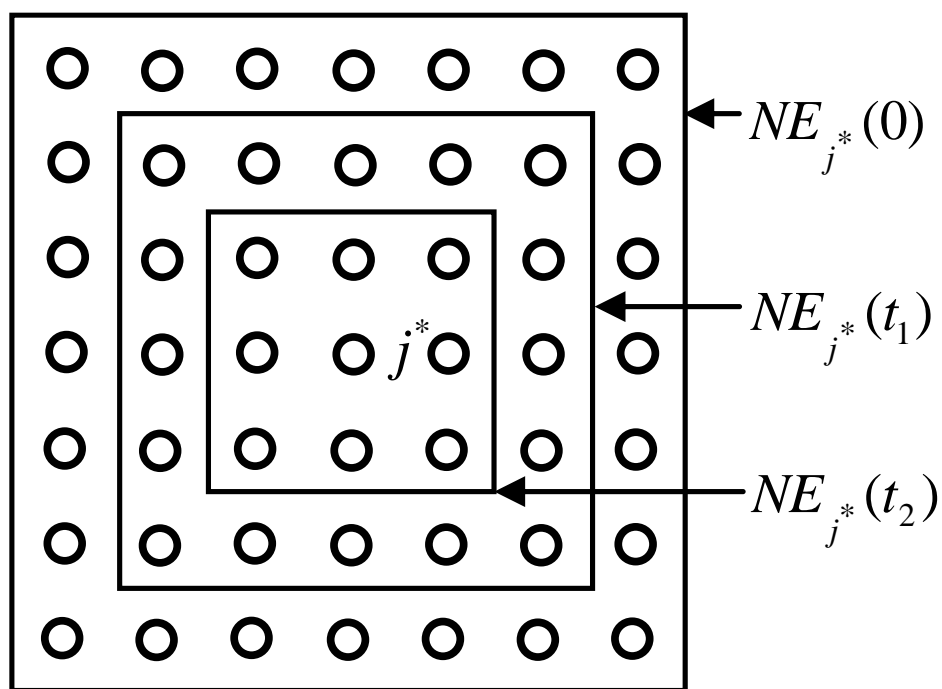
$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)[x_i(t) - w_{ij}(t)] \quad j \in NE_{j^*}(t) \quad 1 \leq i \leq n$$

$\eta(t)$: 修正参数, $0 < \eta(t) < 1$, 随 t 的增加而减小。

第六步: 转到第二步。

聚类中心：存储在与神经元 j^* 连接的权值上。

输出层神经元优胜邻域的选择：



初始邻域选择大些，随算法的进行逐步收缩。

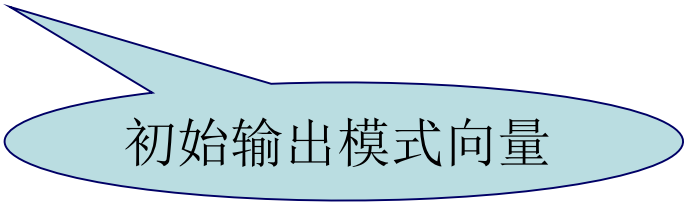
7.4 反馈网络模型Hopfield网络

模拟人脑联想记忆功能的神经网络模型。

寻找记忆：

网络由初始状态向稳定状态演化的过程。

结构：



初始输出模式向量

单层全互连、权值对称的神经网络。

Hopfield网络
(HNN)

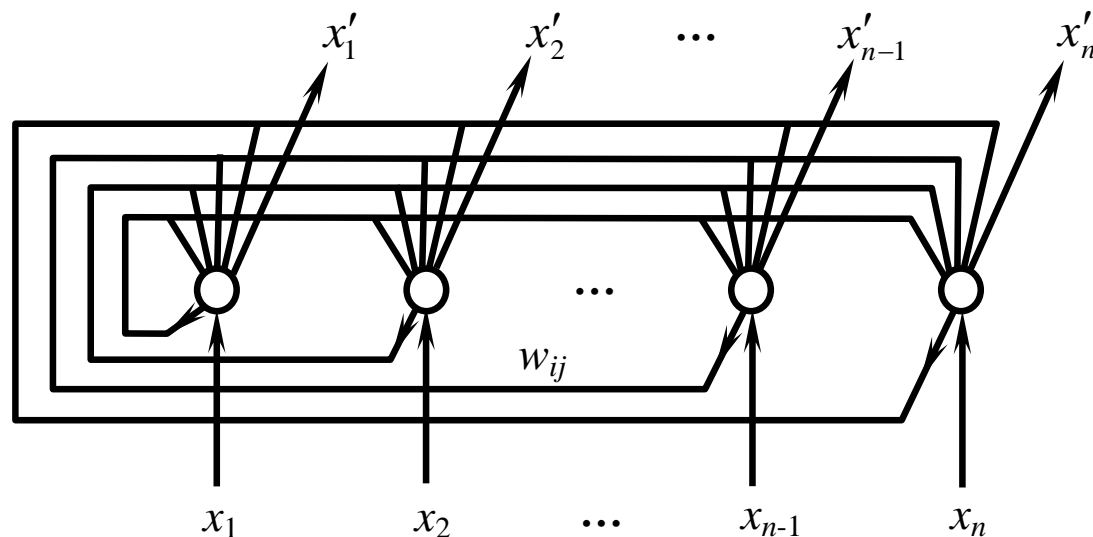
离散型HNN(DHNN):

M-P模型二值神经元

连续型HNN(CHNN):

神经元为连续时间输出。

DHNN:



- * 每个神经元的输出通过加权与其余神经元的输入端连接;
- * 输入模式向量的各分量及神经元的输出值取(+1)或(-1);
- * 神经元的个数与输入模式向量的维数相同;
- * 记忆样本记忆在神经元之间的连接权上。
- * 每个模式类有一个记忆样本，是网络的一个稳定输出状态。

设有 M 类模式，则有 M 个记忆样本，分别是网络的 M 个稳定输出状态。

设 $\mathbf{X}^s = [x_1^s, x_2^s, \Lambda, x_n^s]^T$ 是第 s 类的记忆样本。为了存储 M 个记忆样本，神经元 i 和神经元 j 之间的权值 w_{ij} 为

$$w_{ij} = \begin{cases} \sum_{s=1}^M x_i^s x_j^s, & i \neq j \\ 0, & i = j \end{cases} \quad i, j = 1, 2, \Lambda, n$$

若神经元 i 的输入为 u_i ，输出为 x'_i ，则

$$u_i = \sum_{j=1}^n w_{ij} x'_j$$

$$x'_i = f(u_i) = f\left(\sum_{j=1}^n w_{ij} x'_j\right)$$

$$\text{式中, } f(u_i) = \begin{cases} +1, & u_i > 0 \\ -1, & u_i < 0 \end{cases}$$

联想：输入一个未知类别的模式 \mathbf{X} ，则网络初始状态由 \mathbf{X} 决定，根据上述算法，网络从初始状态开始逐步演化，最终趋向于一个稳定状态，即输出一个与未知类别模式相似的记忆样本。

说明：

定义网络的能量函数

$$E = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} x'_j x'_i$$

由某一神经元的状态的变化量 $\Delta x'_i$ 引起的 E 变化量为

$$\Delta E = -\frac{1}{2} \left(\sum_{j=1}^n w_{ij} x'_j \right) \Delta x'_i$$

式中， $w_{ij} = w_{ji}$ ， $w_{ii} = 0$ 。

$\Delta E < 0$ ， E 有界，网络最终可达到一个不随时间变化的稳定状态。

算法步骤:

第一步：给神经元的连接权赋值，即存贮记忆样本。

$$w_{ij} = \begin{cases} \sum_{s=1}^M x_i^s x_j^s, & i \neq j \\ 0, & i = j \end{cases} \quad i, j = 1, 2, \Lambda, n$$

第二步：用输入的未知类别的模式 $\mathbf{X} = [x_1, x_2, \Lambda, x_n]^T$ 设置网络的初始状态。

若 $x'_i(t)$ 表示神经元 i 在 t 时刻的输出状态，则初始值：

$$x'_i(0) = x_i \quad i = 1, 2, \Lambda, n$$

第三步：迭代计算 $x'_i(t+1)$ 至算法收敛（每个 i 进行多次 t -迭代）。

$$x'_i(t+1) = f \left[\sum_{j=1}^n w_{ij} x'_j(t) \right] \quad i = 1, 2, \Lambda, n$$

神经元输出与未知模式匹配最好的记忆样本。

第四步：转到第二步，输入新模式。

Hopfield神经网络的局限性:

- * 网络能记忆和正确回忆的样本数相当有限;

已证明：当记忆不同模式类的样本数小于网络中神经元个数（或模式向量的维数）的0.15倍时，收敛于伪样本的情况才不会发生。

- * 如果记忆中的某一样本的某些分量与别的记忆样本的对应分量相同，这个记忆样本可能是一个不稳定的平衡点。

可以利用正交算法消除。

End of This Part