

智能控制理论

(Intelligent Control Theories)

第六章 神经网络基本理论

主讲教师：段朝霞（能电院自动化系）

办公室：勤学楼1212

邮 箱：duanzx1989@163.com

模糊控制在处理数值数据、自学习能力等方面还远没有达到人脑的境界。人工神经网络从另一个角度出发，即从人脑的生理学和心理学着手，通过人工模拟人脑的工作机理来实现机器的部分智能行为。

人脑是一部不寻常的智能机，它能以惊人的高速度解释感觉器官传来的含糊不清的信息。它能觉察到喧闹房间内的窃窃私语，能够识别出光线暗淡的胡同中的一张面孔，更能通过不断地学习而产生伟大的创造力。古今中外，许许多多科学家为了揭开大脑机能的奥秘，从不同的角度进行着长期的不懈努力和探索，逐渐形成了一个多学科交叉的前沿技术领域——神经网络(**Neural Network**)。

人工神经网络（简称神经网络，**Neural Network**）是模拟人脑思维方式的数学模型。

神经网络是在现代生物学研究人脑组织成果的基础上提出的，用来模拟人类大脑神经网络的结构和行为。神经网络反映了人脑功能的基本特征，如并行信息处理、学习、联想、模式分类、记忆等。

6.1 神经网络发展历史

神经网络的发展历程经过4个阶段。

1 启蒙期（1890-1969年）

- 1890年，W. James 发表专著《心理学》，讨论了脑的结构和功能。
- 1943年，心理学家 W.S. McCulloch 和数学家 W. Pitts 提出了描述脑神经细胞动作的数学模型，即 M-P 模型（第一个神经网络模型）。
- 1949年，心理学家 Hebb 实现了对脑细胞之间相互影响的数学描述，从心理学的角度提出了至今仍对神经网络理论有着重要影响的 Hebb 学习法则。
- 1958年，E. Rosenblatt 提出了描述信息在人脑中贮存和记忆的数学模型，即著名的感知机模型（Perceptron）。
- 1962年，Widrow 和 Hoff 提出了自适应线性神经网络，即Adaline网络，并提出了网络学习新知识的方法，即 Widrow 和 Hoff 学习规则（即 δ 学习规则），并用电路进行了硬件设计。

6.1 神经网络发展历史

2 低潮期（1969-1982）

受当时神经网络理论研究水平的限制及冯·诺依曼式计算机发展的冲击等因素的影响，神经网络的研究陷入低谷。在美、日等国有少数学者继续着神经网络模型和学习算法的研究，提出了许多有意义的理论和方法。

- 1969年，S. Grossberg 和 A. Carpenter 提出了至今为止最复杂的 ART 网络，该网络可以对任意复杂的二维模式进行自组织、自稳定和大规模并行处理。
- 1972年，Kohonen 提出了自组织映射的 SOM 模型

6.1 神经网络发展历史

3 复兴期（1982-1986）

- 1982年，物理学家 Hopfield 提出了 Hopfield 神经网络模型，该模型通过引入能量函数，实现了问题优化求解，1984年他用此模型成功地解决了旅行商路径优化问题(TSP)。
- 1986年，在Rumelhart和McClland等出版《Parallel Distributed Processing》一书，提出了一种著名的多层神经网络模型，即BP网络。该网络是迄今为止应用最普遍的神经网络。

6.1 神经网络发展历史

4 新连接机制时期（1986-现在）

- 神经网络从理论走向应用领域，出现了神经网络芯片和神经计算机。
- 神经网络主要应用领域有：模式识别与图象处理（语音、指纹、故障检测和图象压缩等）、控制与优化、预测与管理（市场预测、风险分析）、通信等。

6.2 神经网络的基本概念

1. 生物神经元的结构与功能特点

2. 人工神经元的模型

3. 神经网络的结构

4. 神经网络的工作方式

5. 神经网络的学习

6. 神经网络的分类

6.2 神经网络的基本概念

6.2.1 生物神经元的结构与功能特点

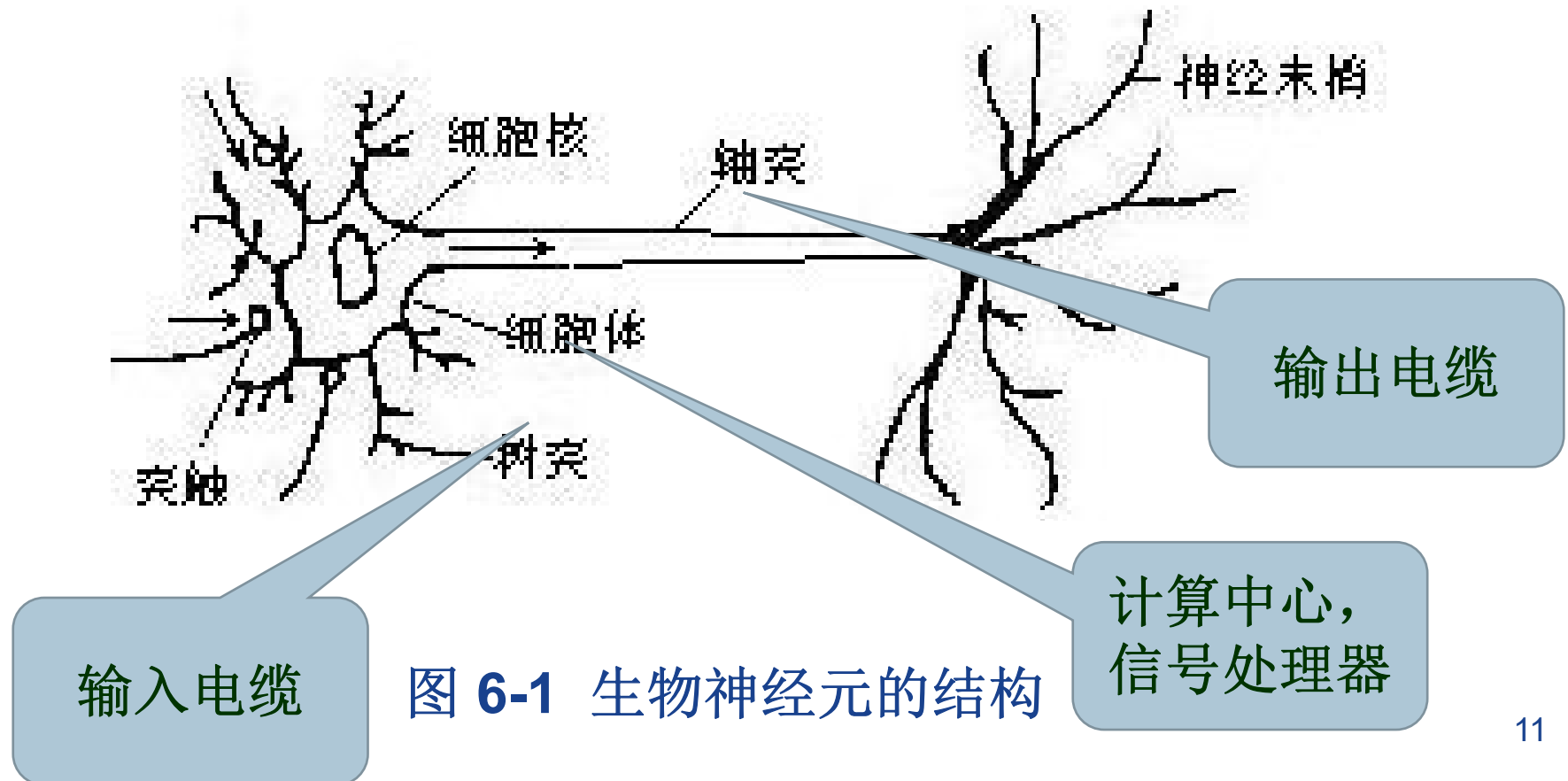
神经生理学和神经解剖学证明了人的思维是由人脑完成的。神经元是组成人脑的最基本单元，它能够接受并处理信息，人脑大约由 $10^{11} \sim 10^{12}$ 个神经元组成，其中每个神经元约与 $10^4 \sim 10^5$ 个神经元通过突触连接，因此，人脑是一个复杂的信息并行加工处理巨系统。

探索脑组织的结构、工作原理及信息处理的机制，是整个人类面临的一项挑战，也是整个自然科学的前沿领域。

6.2 神经网络的基本概念

1. 生物神经元的结构

生物神经元（以下简称神经元），也称神经细胞，是构成神经系统的基本单元。神经元主要由细胞体、树突和轴突构成。



6.2 神经网络的基本概念

2. 生物神经元的功能特点

从生物控制论的观点来看，作为控制和信息处理基本单元的神经元，具有以下功能特点。

(1) 时空整合功能

神经元对于不同时间通过同一突触传入的信息，具有时间整合功能；对于同一时间通过不同突触传入的信息，具有空间整合功能。两种功能相互结合，是使生物神经元具有时空整合的输入信息处理功能。

6.2 神经网络的基本概念

(2) 动态极化性

在每一种神经元中，信息都是以预知的确定方向流动的，即从神经元的接收信息部分（细胞体、树突）传到轴突的起始部分，再传到轴突终端的突触，最后再传给另一神经元。尽管不同的神经元在形状及功能上都有明显的不同，但大多数神经元都是按这一方向进行信息流动的。

(3) 兴奋与抑制状态

所谓兴奋状态是指神经元对输入信息经整合后使细胞膜电位升高，且超过了动作电位的阈值，此时产生神经冲动并由轴突输出。

抑制状态是指对输入信息整合后，细胞膜电位值下降到低于动作电位的阈值，从而导致无神经冲动输出。

6.2 神经网络的基本概念

(4) 结构的可塑性

由于突触传递信息的特性是可变的，也就是它随着神经冲动传递方式的变化，传递作用强弱不同，形成了神经元之间连接的柔性，这种特性又称为神经元结构的可塑性。

(5) 脉冲与电位信号的转换

突触界面具有脉冲与电位信号的转换功能。沿轴突传递的电脉冲是等幅的、离散的脉冲信号，而细胞膜电位变化为连续的电位信号，这两种信号是在突触接口进行变换的。

6.2 神经网络的基本概念

(6) 突触延期和不应期

突触对信息的传递具有时延和不应期，在相邻的两次输入之间需要一定的时间间隔，在此期间，无激励，不传递信息，这称为不应期。

(7) 学习、遗忘和疲劳

由于神经元结构的可塑性，突触的传递作用有增强、减弱和饱和的情况。所以，神经细胞也具有相应的学习、遗忘和疲劳效应（饱和效应）。

6.2 神经网络的基本概念

6.2.2 人工神经元模型

生物神经元经抽象化后，可得到如图6-2所示的一种人工神经元模型，它有三个基本要素。

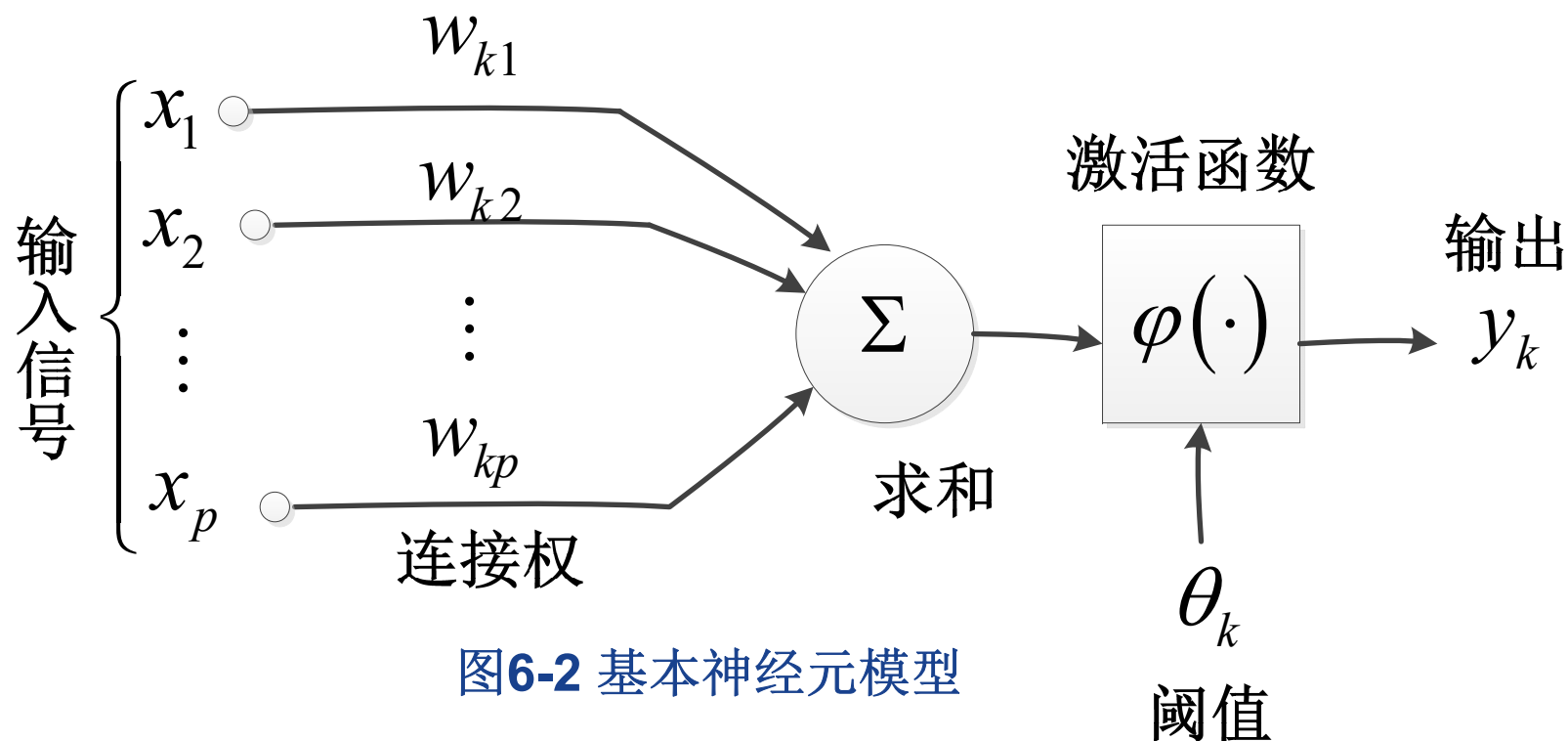


图6-2 基本神经元模型

6.2 神经网络的基本概念

1. 连接权

连接权对应于生物神经元的突触，各个神经元之间的连接强度由连接权的权值表示，权值为正表示激活，为负表示抑制。

2. 求和单元

用于求取各输入信号的加权和（线性组合）。

3. 激活函数

激活函数起非线性映射作用，并将神经元输出幅度限制在一定范围内，一般限制在 $(0,1)$ 或 $(-1,1)$ 之间。激活函数也称传输函数。

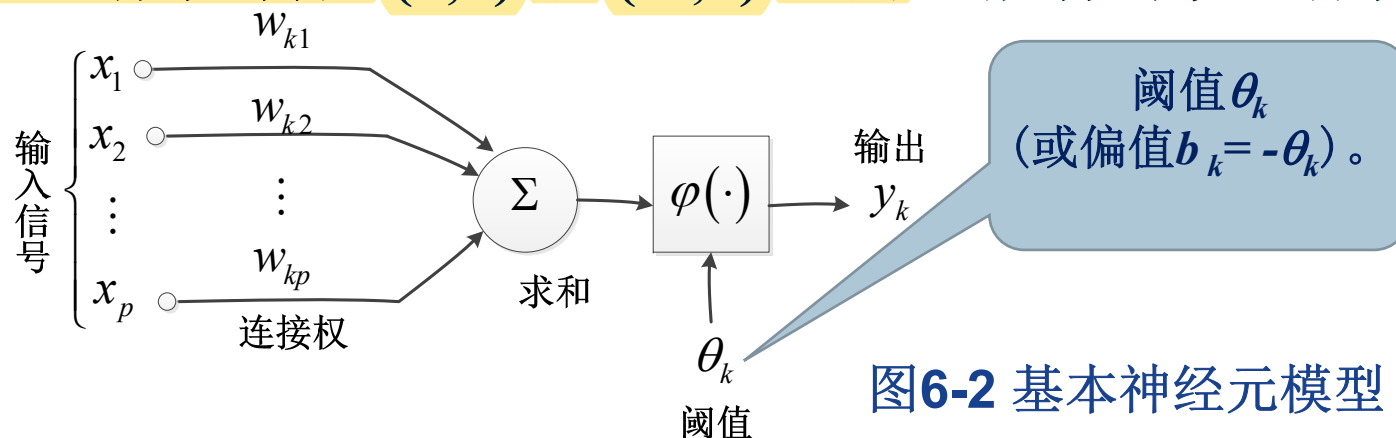


图6-2 基本神经元模型

6.2 神经网络的基本概念

以上作用可分别以数学式表达出来:

$$u_k = \sum_{j=1}^p w_{kj} x_j, y_k = \varphi(u_k - \theta_k)$$

x_1, x_2, \dots, x_p ——输入信号,

$w_{k1}, w_{k2}, \dots, w_{kp}$ ——神经元 k 的权值,

u_k —— 线性组合结果,

θ_k ——为阈值,

$\varphi(\cdot)$ ——激活函数,

y_k ——神经元 k 的输出,

6.2 神经网络的基本概念

若把输入的维数增加一维，则可将阈值 θ_k 包括进去。

即

$$u_k = \sum_{j=0}^p w_{kj} x_j, y_k = \varphi(u_k)$$

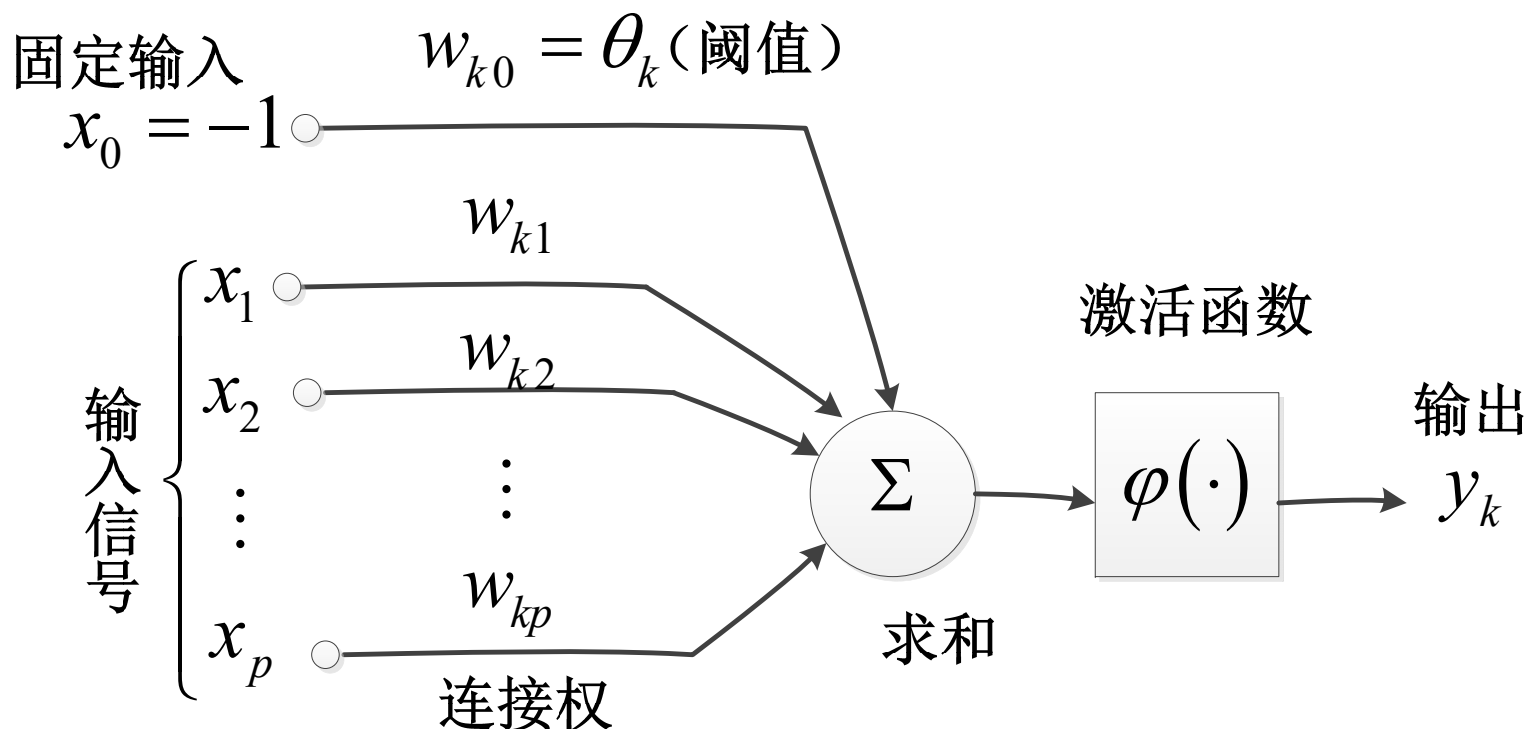


图6-3a 输入扩维后的神经元模型（包括阈值）

6.2 神经网络的基本概念

若把偏值 b_k 包括进去。即

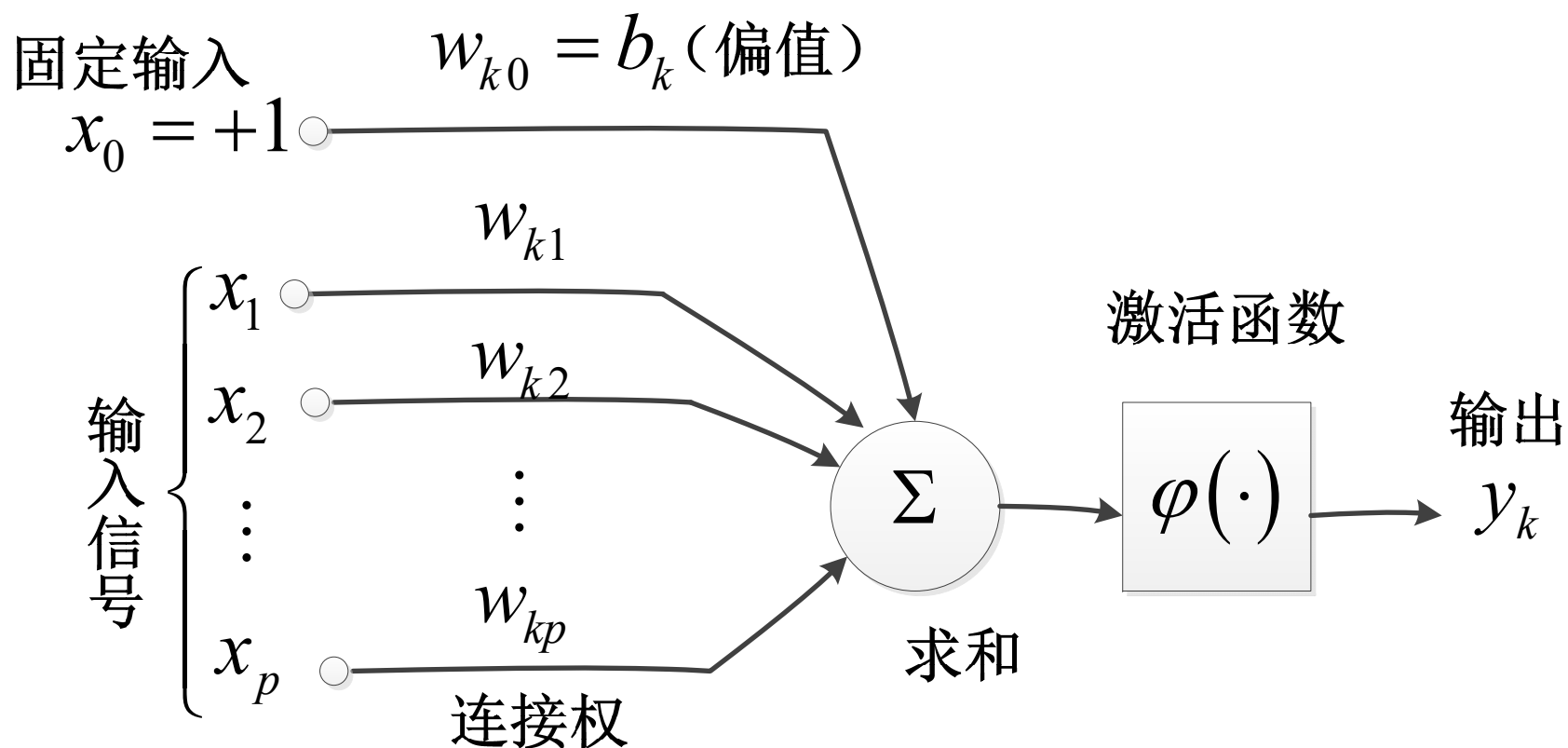


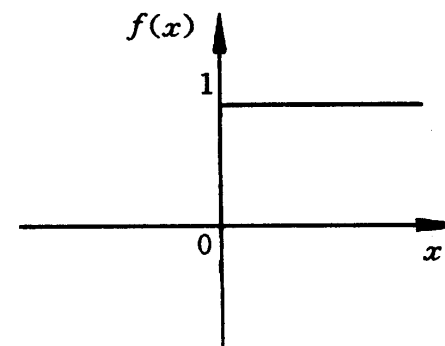
图6-3b 输入扩维后的神经元模型（包括偏值）

6.2 神经网络的基本概念

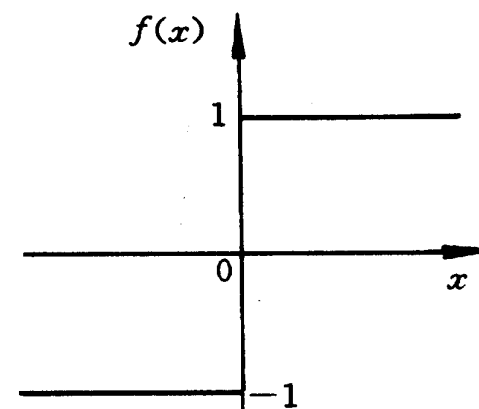
常用的激活函数 $\varphi(\cdot)$ 的种类：

1) 阈值型函数(阶跃函数)

$$y = \varphi(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$$



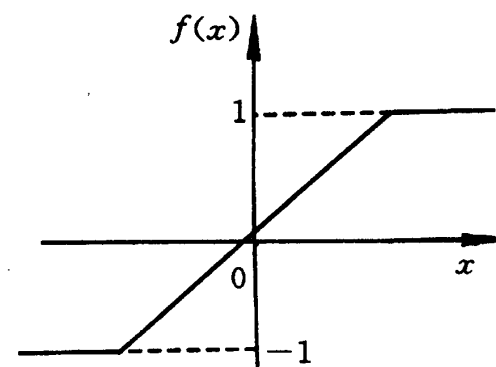
$$y = \varphi(x) = \begin{cases} 1, x \geq 0 \\ -1, x < 0 \end{cases}$$



6.2 神经网络的基本概念

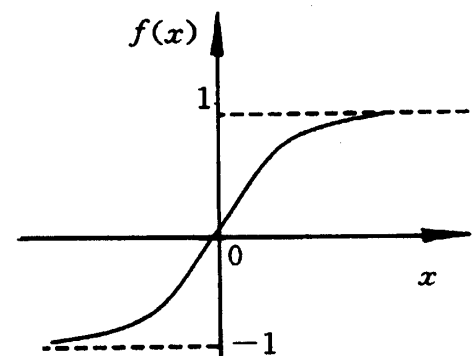
2) 饱和型函数（分段线性函数）

$$y = \varphi(x) = \begin{cases} 1, & x \geq \frac{1}{k} \\ kx, & -\frac{1}{k} < x < \frac{1}{k} \\ -1, & x < -\frac{1}{k} \end{cases}$$



3) 双曲函数

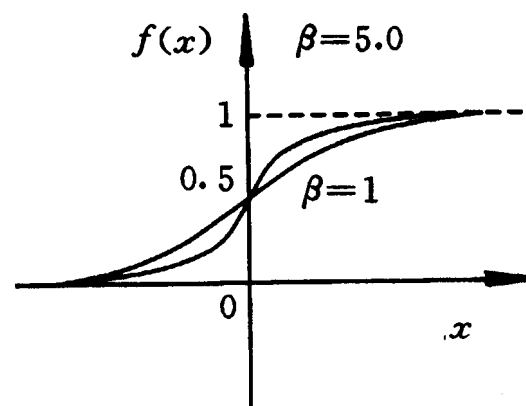
$$y = \varphi(x) = \arctan(x)$$



6.2 神经网络的基本概念

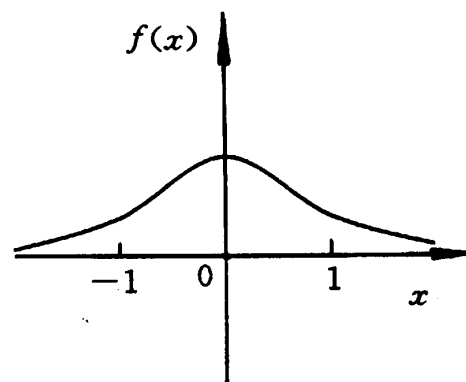
4) S型函数

$$y = \varphi(x) = \frac{1}{1 + e^{-\beta x}}, \beta > 0$$



5) 高斯函数

$$y = \varphi(x) = e^{\left(-\frac{x^2}{b^2}\right)}$$



6.2 神经网络的基本概念

6.2.3 神经网络的结构

人工神经网络 (Artificial Neural Networks, ANN) 是由大量人工神经元经广泛互连而组成的，它可用来模拟脑神经系统的结构和功能。

人工神经网络可以看成是以人工神经元为节点，用有向加权弧连接起来的有向图。

人工神经元就是对生物神经元的模拟，而有向加权弧则是轴突—突触—树突对的模拟。有向弧的权值表示相互连接的两个神经元间相互作用的强弱。

6.2 神经网络的基本概念

人工神经网络是生物神经网络的一种模拟和近似。它主要从两个方面进行模拟。

- ❖ 一种是从生理结构和实现机理方面进行模拟，它涉及到生物学、生理学、心理学、物理及化学等许多基础科学。由于生物神经网络的结构和机理相当复杂，现在距离完全认识它们还相差甚远；
- ❖ 另外一种是从功能上加以模拟，即尽量使得人工神经网络具有生物神经网络的某些功能特性，如学习、识别、控制等功能。

根据连接方式，人工神经网络主要分为两种：

1. 前馈型网络

神经元分层排列，组成输入层、隐含层和输出层。每一层的神经元只接受前一层神经元的输入。输入模式经过各层的顺次变换后，由输出层输出。在各神经元之间不存在反馈。感知器和误差反向传播网络采用前向网络形式。

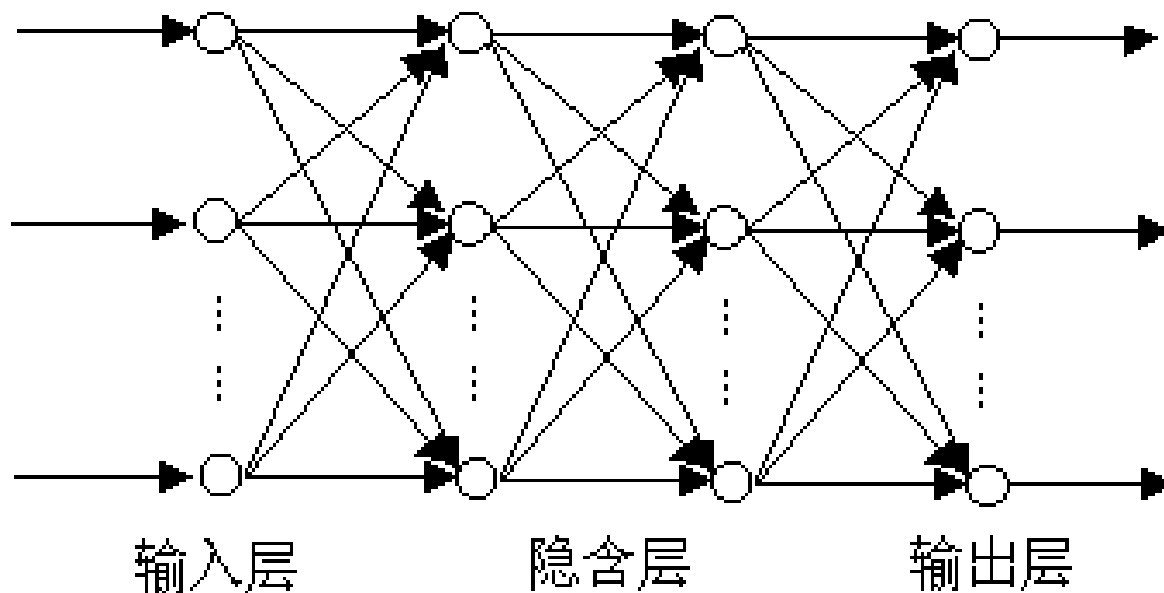


图 6-4 前馈型神经网络

6.2 神经网络的基本概念

2. 反馈型网络（递归网络/回归网络）

该网络结构在输出层到输入层存在反馈，即每一个输入节点都有可能接受来自外部的输入和来自输出神经元的反馈。这种神经网络是一种反馈动力学系统，它需要工作一段时间才能达到稳定。（Hopfield 网络）

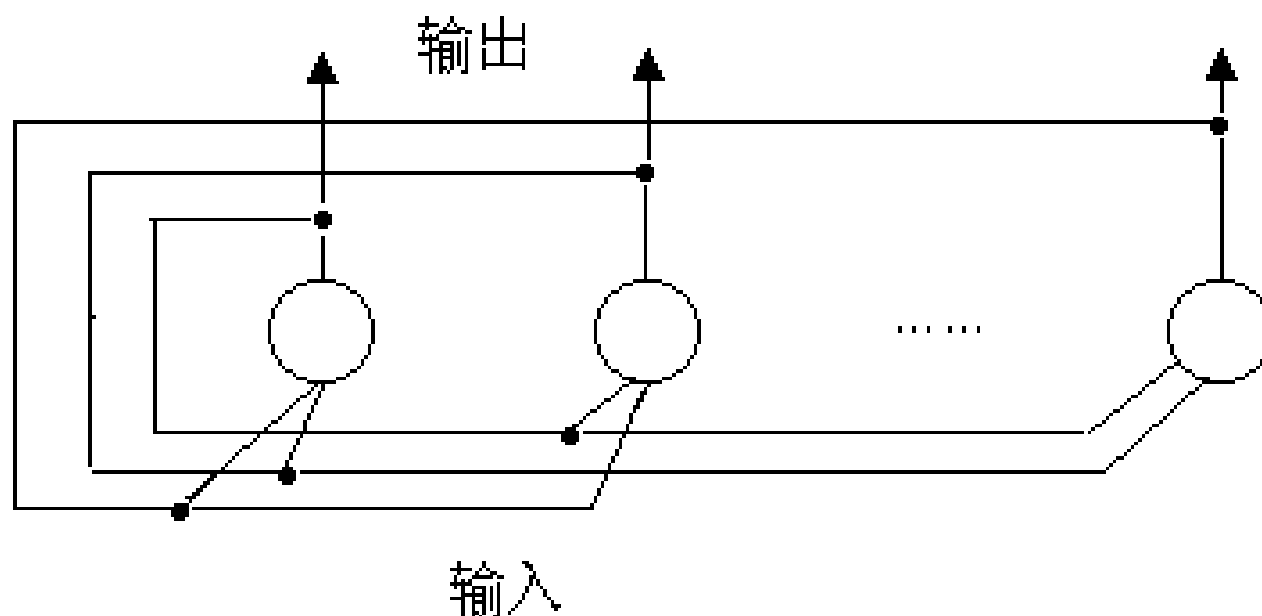


图 6-5 反馈型神经网络

6.2 神经网络的基本概念

6.2.4 神经网络的工作方式

神经网络的工作过程主要分为两个阶段：

- 第一阶段：学习期，此时各计算单元状态不变，各连接权上的权值可通过学习来修改；
- 第二阶段：工作期，此时各连接权固定，计算单元变化，以达到某种稳定状态。

6.2 神经网络的基本概念

从作用效果看，前馈网络主要是函数映射，可用于模式识别和函数逼近。

反馈网络按对能量函数的极小点的利用来分类有两种：

- 能量函数的所有极小点都起作用，这一类主要用作各种联想存储器；
- 只利用全局极小点，它主要用于求解最优化问题。

6.2 神经网络的基本概念

6.2.5 神经网络的学习

1. 学习方式

通过向环境学习获取知识并改进自身性能是神经网络的一个重要特点，在一般情况下，性能的改善是按某种预定的度量调节自身参数（如权值）随时间逐步达到的，学习方式（按环境所供信息的多少分）有以下三种。

- 1) 有监督学习（有教师学习）
- 2) 无监督学习（无教师学习）
- 3) 强化学习（再励学习）

6.2 神经网络的基本概念

1) 有监督学习（有教师学习）

这种学习方式需要外界存在一个“教师”，他可对一组给定输入提供应有的输出结果（正确答案），这组已知的输入——输出数据称为训练样本集。学习系统可根据已知输出与实际输出之间的差值（误差信号）来调节系统参数。

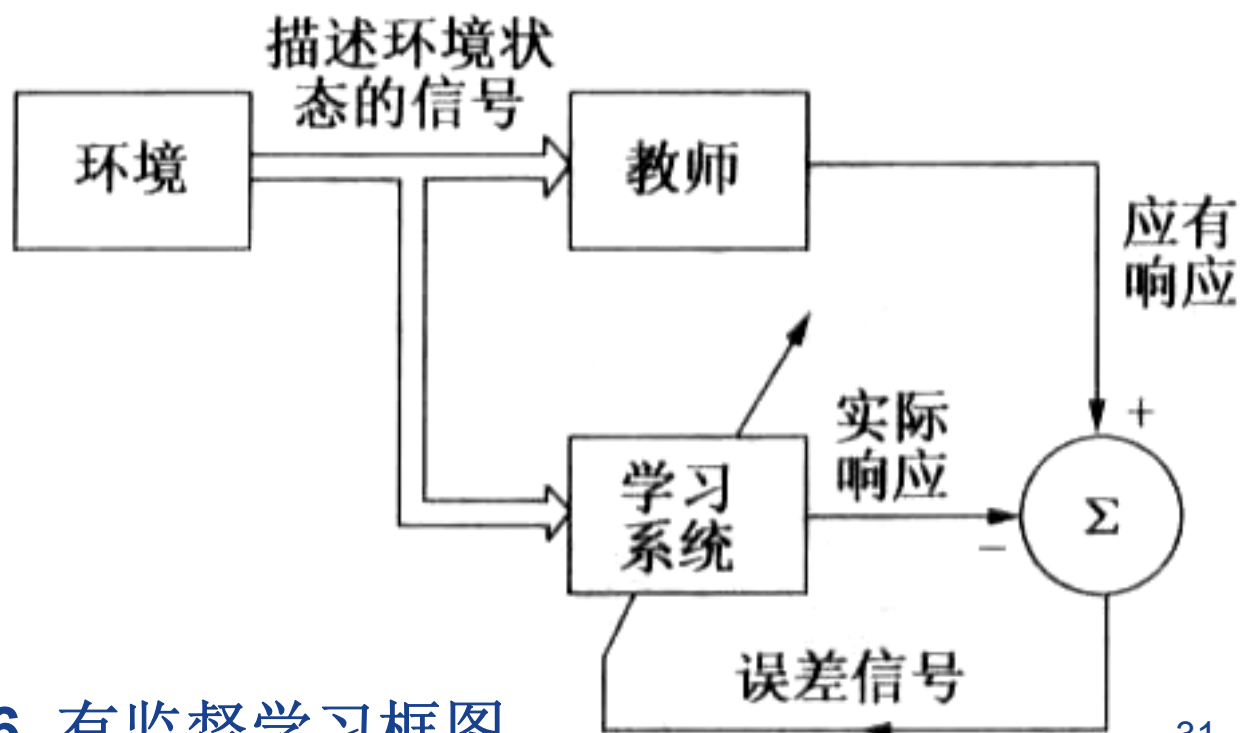


图6-6 有监督学习框图

6.2 神经网络的基本概念

2) 无监督学习（无教师学习）

无监督学习不存在外部教师，学习系统完全按照环境所提供数据的某些统计规律来调节自身参数或结构（这是一种自组织过程），以表示外部输入的某种固有特性（如聚类，或某种统计上的分布特征）。

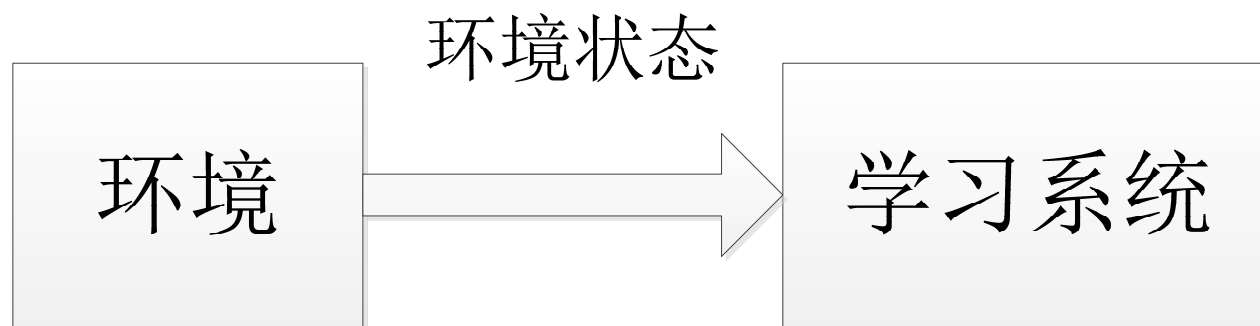


图6-7 无监督学习框图

6.2 神经网络的基本概念

3) 强化学习（再励学习）

介于上述两种学习之间，外部环境对系统输出结果只给出评价（奖或罚）而不是给出正确答案，学习系统通过强化那些受奖励的动作来改善自身性能。

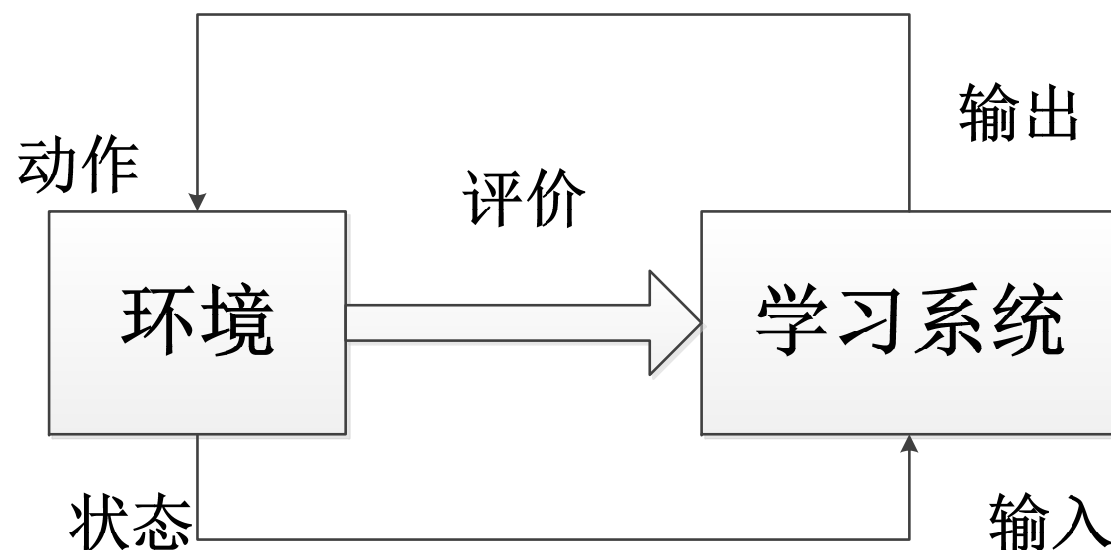


图6-8 强化学习框图

6.2 神经网络的基本概念

2. 学习算法

- ◆ δ 学习规则（误差修正规则）
- ◆ Hebb 学习规则
- ◆ 竞争（competitive）学习规则

6.2 神经网络的基本概念

(1) δ 学习规则（误差修正规则）

令 $y_i(k)$ 为输入 $x(k)$ 时神经元 i 在 k 时刻的实际输出， $t_i(k)$ 表示相应的期望输出，则误差信号可写为：

$$e_i(k) = t_i(k) - y_i(k)$$

误差纠正学习的最终目的是使某一基于 $e_i(k)$ 的目标函数达最小，以使网络中每一个输出单元的实际输出在某种统计意义上最逼近于期望输出。

6.2 神经网络的基本概念

最常用的目标函数是均方误差判据，定义为：

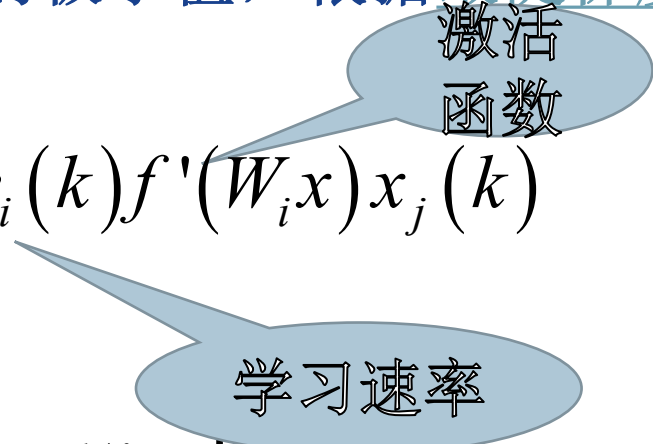
$$J = E \left\{ \frac{1}{2} \sum_{i=1}^L (t_i - y_i)^2 \right\}$$

其中 E 是统计期望算子， L 为样本的数量。上式的前提是被学习的过程是宽平稳的，具体方法可用最陡梯度下降法。用 J 在时刻 k 瞬时值 $J(k)$ 代替 J

$$J(k) = \frac{1}{2} \sum_{i=1}^L (t_i(k) - y_i(k))^2 = \frac{1}{2} \sum_{i=1}^L e_i^2(k)$$

6.2 神经网络的基本概念

问题变为求 $J(k)$ 对权值 w_{ij} 的极小值，根据最陡梯度下降法可得：

$$\Delta w_{ij}(k) = \eta \sum_{i=1}^L e_i(k) f'(W_i x) x_j(k)$$


$$W_i = [w_{i0}, w_{i1}, w_{i2}, \dots, w_{iM}] ,$$

$$x = [x_0, x_1, x_2, \dots, x_M]^T , \quad \delta_i(k) = \sum_{i=1}^L e_i(k) f'(W_i x)$$

$$\theta(k) = W_i^T x$$

$$y_i(k) = f(\theta(k)) = f(W_i x)$$

梯度下降法(补充)

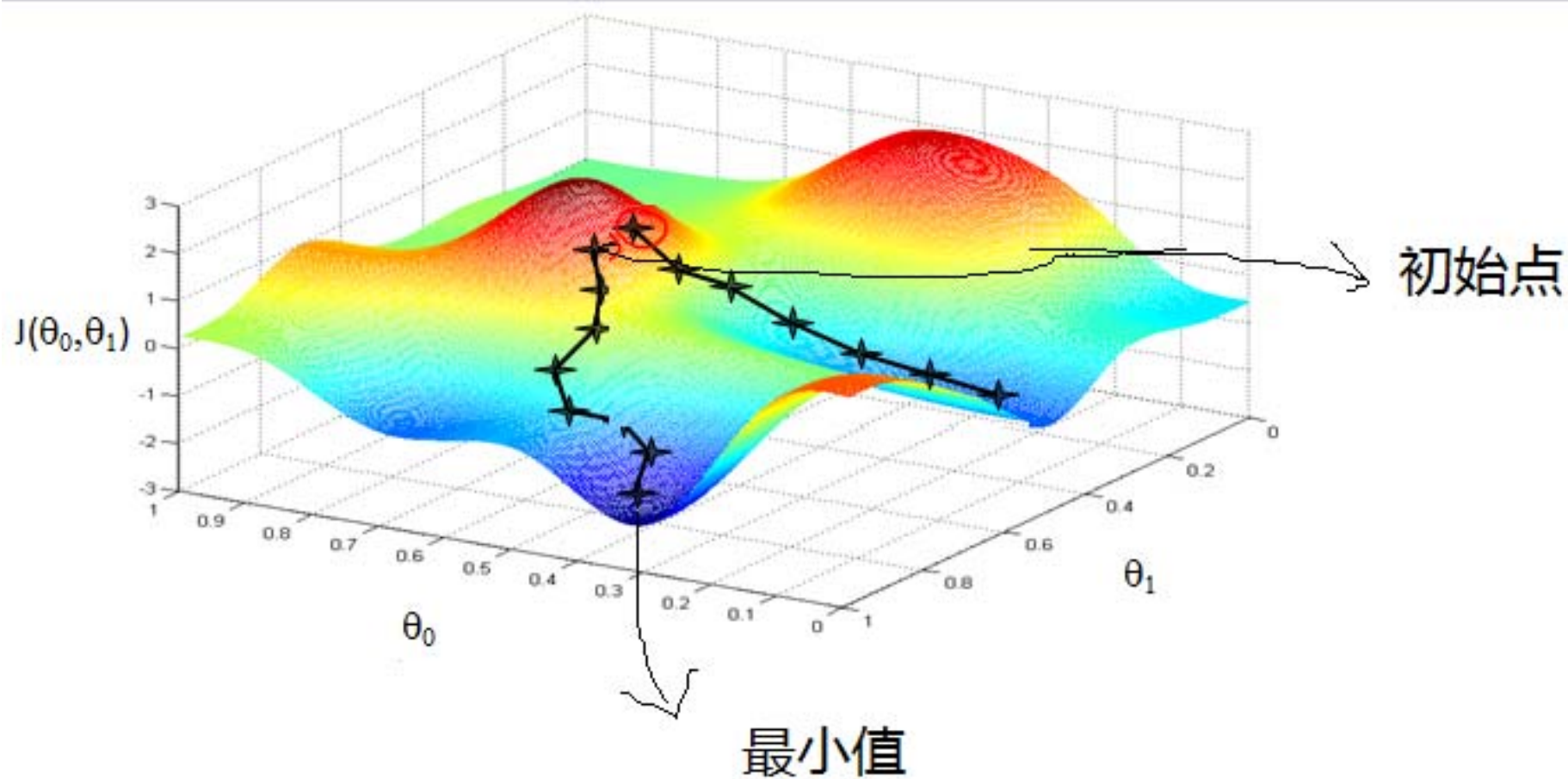
❖ 梯度

$$\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)^T$$

$$\nabla f(x_0, y_0) = \left(\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial y_0} \right)^T$$

❖ 梯度向量的意义：函数变化增加最快的方向，对 $f(x, y)$ 在点 (x_0, y_0) 沿着梯度的方向是 $f(x, y)$ 增加最快的方向。或者说沿着梯度向量的方向，更加容易找到函数的极大值。那么，反过来沿着梯度向量相反的方向，也就是 $-\left(\frac{\partial f}{\partial x_0}, \frac{\partial f}{\partial y_0} \right)^T$ 的方向，更加容易找到函数的最小值。

梯度下降法(补充)



6.2 神经网络的基本概念

(2) Hebb学习

“当某一突触（连接）两端的神经元的激活同步（同为激活或同为抑制）时，该连接的强度应增加，反之则应减弱”，用数学方式可描述为：

$$\Delta w_{ij}(k) = F_i(y_i(k), x_j(k))$$

式中 $y_i(k)$, $x_j(k)$ 分别为 w_{ij} 两端神经元的状态，其中最常用的一种情况为：

$$\Delta w_{ij}(k) = \eta \cdot y_i(k) \cdot x_j(k)$$

式中 η 为学习速率。

由于 $w_{ij}(k)$ 与 $y_i(k)$, $x_j(k)$ 的相关成比例，有时称之为**相关学习规则**。

6.2 神经网络的基本概念

原始的Hebb学习规则对权值矩阵的取值未做任何限制，因而学习后权值可取任意值。为了克服这一缺点，在Hebb学习规则的基础上增加一个衰减项，即

$$\Delta w_{ij}(k) = \eta \cdot y_i(k) \cdot x_j(k) - dr \cdot w_{ij}(k)$$

衰减项的加入能够增加网络学习的“记忆”功能，并且能够有效地对权值的取值加以限制。衰减系数 dr 的取值在 $[0,1]$ 之间。当取0时，就变成原始的Hebb学习规则。

6.2 神经网络的基本概念

另外，Hebb规则还可以采用有监督的学习，对于有监督学习的Hebb规则而言，将目标输出代替实际输出。由此，算法被告知的就是网络应该做什么，而不是网络当前正在做什么，可描述为：

$$\Delta w_{ij}(k) = \eta \cdot t_i(k) \cdot x_j(k)$$

6.2 神经网络的基本概念

(3) 竞争 (Competitive) 学习

在竞争学习时网络各输出单元互相竞争，最后达到只有一个最强者激活。众多输出单元中如有某一单元较强，则它将获胜并抑制其他单元，最后只有比较强者处于激活状态

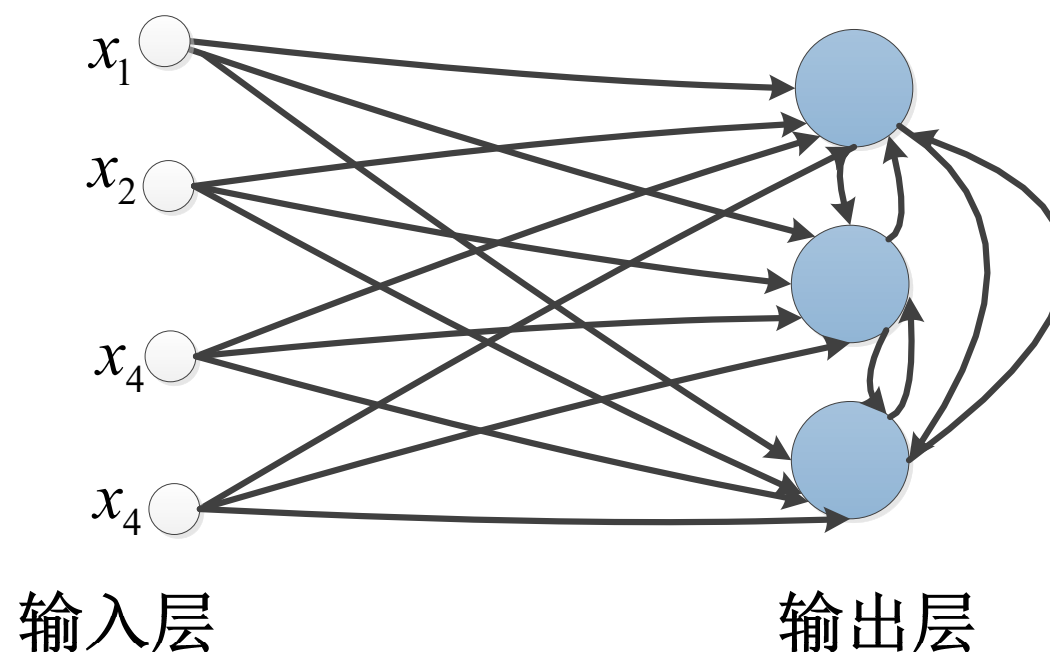


图6-9 竞争学习网络

6.2 神经网络的基本概念

最常用的竞争学习规则有以下三种：

Kohonen规则：
$$\Delta w_{ij}(k) = \begin{cases} \eta(x_j - w_{ij}), & \text{若神经元}j\text{竞争成功} \\ 0, & \text{若神经元}j\text{竞争失败} \end{cases}$$

Instar规则：
$$\Delta w_{ij}(k) = \begin{cases} \eta y_i(x_j - w_{ij}), & \text{若神经元}j\text{竞争成功} \\ 0, & \text{若神经元}j\text{竞争失败} \end{cases}$$

Outstar规则：
$$\Delta w_{ij}(k) = \begin{cases} \eta (y_i - w_{ij})/x_j, & \text{若神经元}j\text{竞争成功} \\ 0, & \text{若神经元}j\text{竞争失败} \end{cases}$$

6.2 神经网络的基本概念

3. 学习与自适应

当学习系统所处环境平稳时（统计特征不随时间变化），从理论上说通过监督学习可以学到环境的统计特征，这些统计特征可被学习系统（神经网络）作为经验记住。

如果环境是非平稳的（统计特征随时间变化），通常的监督学习没有能力跟踪这种变化，为解决此问题需要网络有一定的自适应能力，此时对每一个不同输入都作为一个新的例子对待。

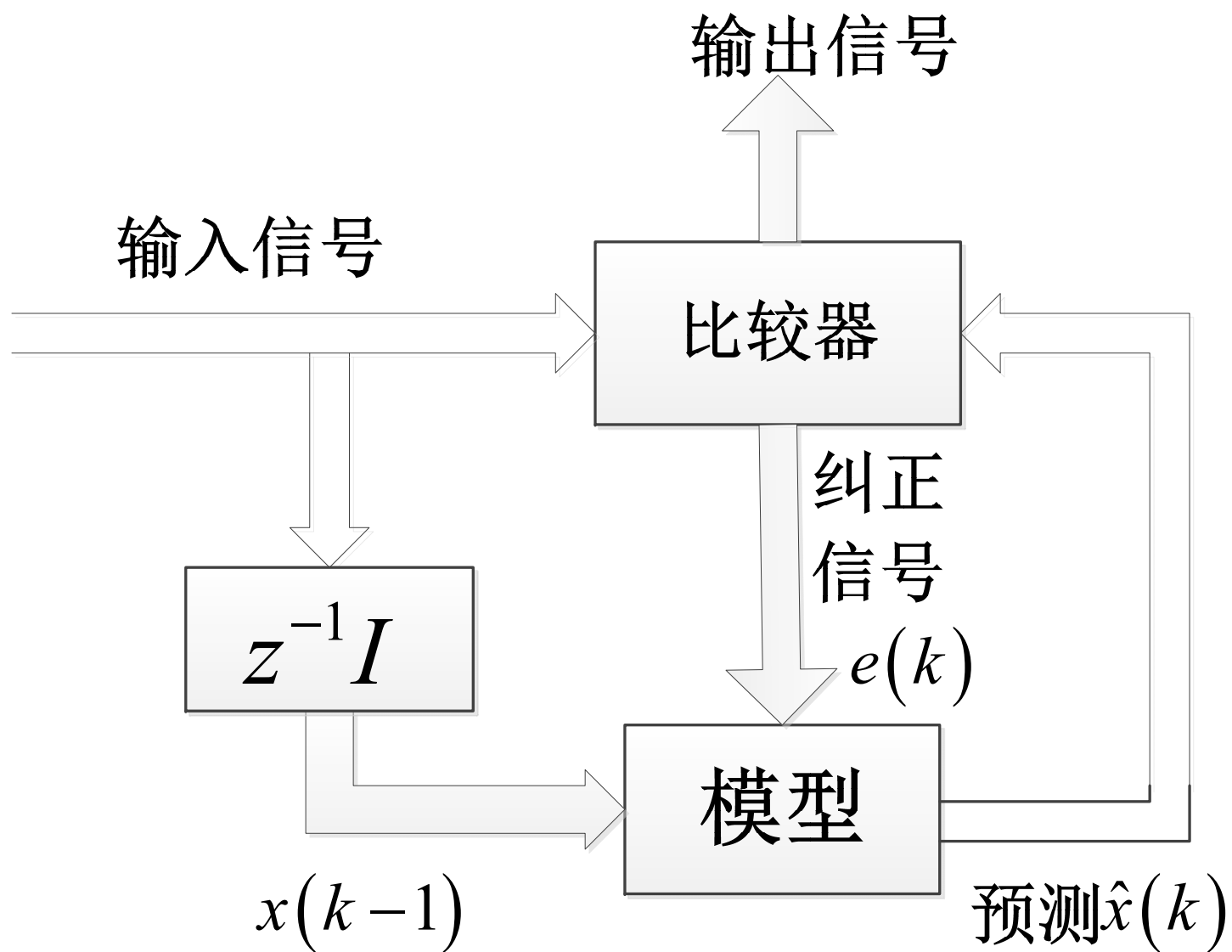


图6-10 自适应学习框图

6.2 神经网络的基本概念

6.2.6 神经网络的分类

神经网络根据不同的情况，可按以下几方面进行分类：

- (1) **按功能分**：连续型与离散型、确定型与随机型、静态与动态神经网络；
- (2) **按连接方式分**：前馈（或称前向）型与反馈型神经网络；
- (3) **按逼近特性分**：全局逼近型与局部逼近型神经网络；
- (4) **按学习方式分**：有监督学习、无监督学习和强化学习神经网络。

6.3 典型神经网络模型

1. MP (McCulloch-Pitts) 模型

2. 感知机神经网络

3. 自适应线性网络 (**Adaline, Adaptive Linear Element**)

4. **BP (Back Propagation)** 网络

5. 径向基(RBF, Radial Basis Function)网络

6. 自组织竞争网络, 自组织特征映射网络 (**SOM**)

7. **Hopfield**网络

6.3 典型神经网络模型I——MP模型

6.3.1 M P 模型

MP模型最初是由美国心理学家 **McCulloch** 和数学家 **Pitts** 在1943年共同提出的，它是有固定的结构和权组成的，它的权分为**兴奋性突触权**和**抑制性突触权**两类，如抑制性突触权被激活，则神经元被抑制，输出为零。

而兴奋性突触权的数目比较多，兴奋性突触权能否激活，则要看它的累加值是否大于一个阈值，大于该阈值神经元即兴奋。

6.3 典型神经网络模型I——MP模型

MP模型的结构

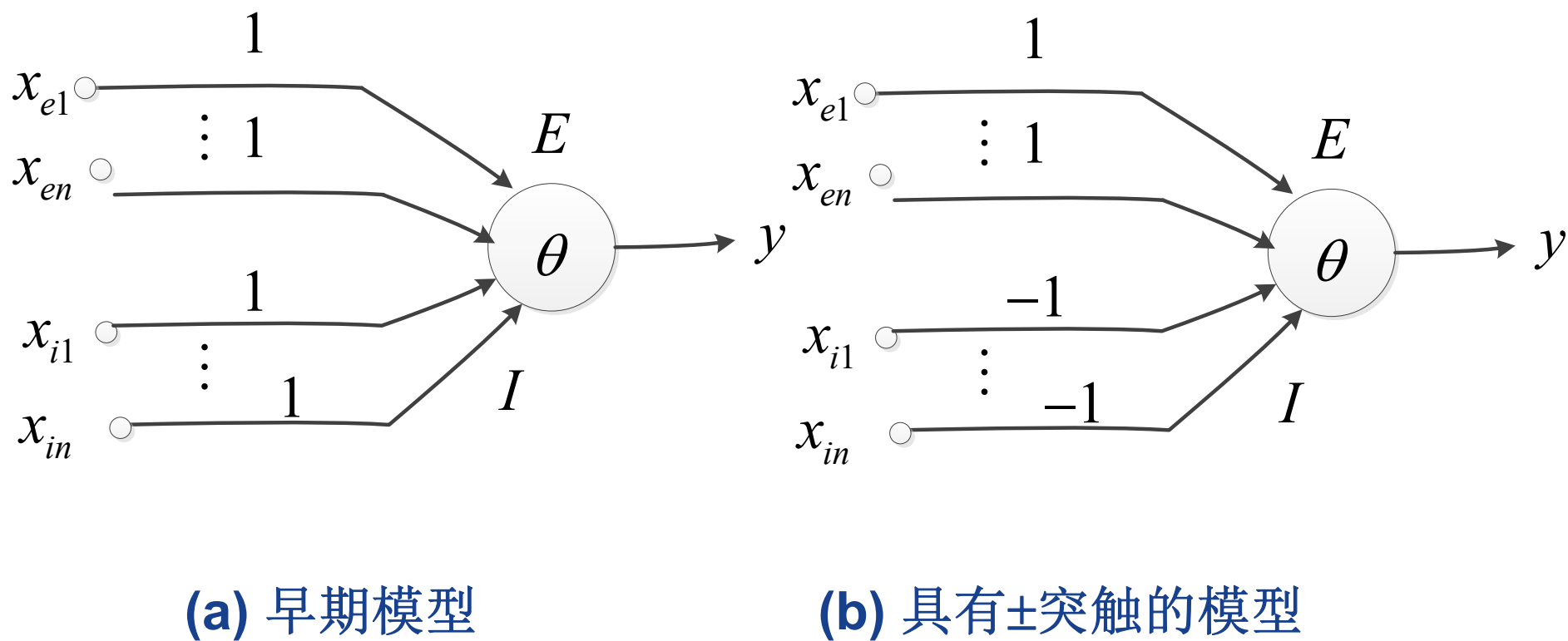


图6-11 MP模型中单个神经元示意图

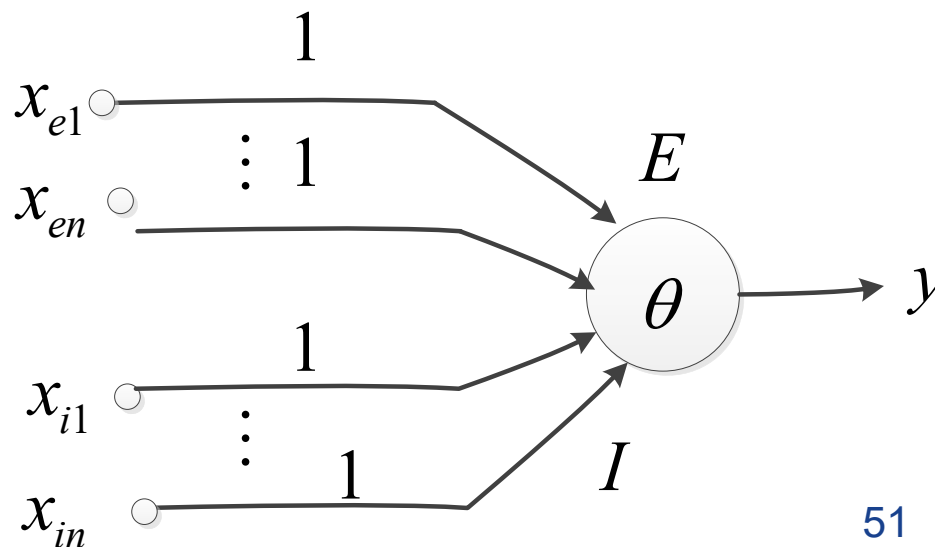
6.3 典型神经网络模型I——MP模型

变换关系为

$$E = \sum_{j=1}^n x_{ej}, I = \sum_{k=1}^n x_{ik}$$

式中 x_{ej} ($j=1,2,\dots,n$) 为兴奋性突触的输入, x_{ik} ($k=1,2,\dots,n$) 为抑制性突触的输入, 则输入与输出的转换关系为

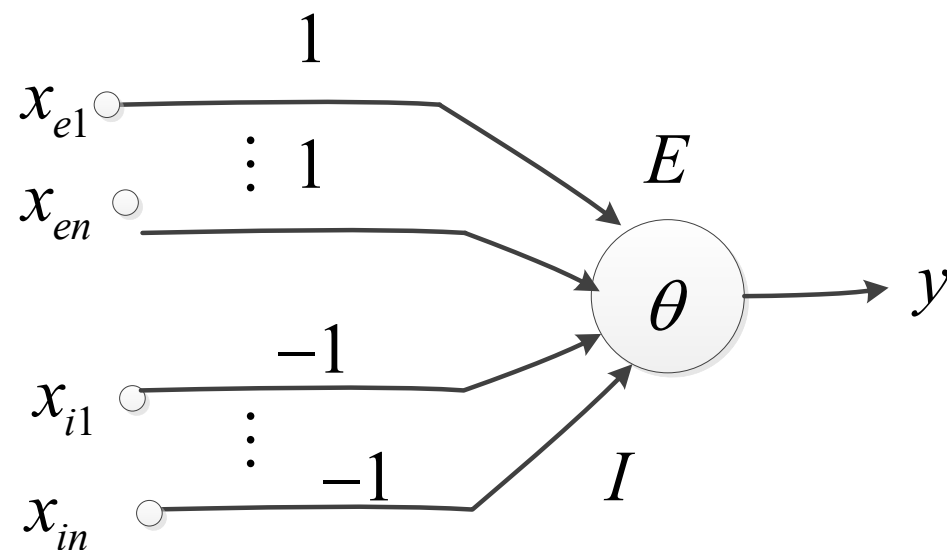
$$y = \begin{cases} 1, & I = 0, E \geq \theta \\ 0, & I = 0, E < \theta \\ 0, & I > 0 \end{cases}$$



6.3 典型神经网络模型I——MP模型

如果兴奋与抑制突触用权 ± 1 表示，而总的作用用加权的办法实现，兴奋为1，抑制为-1，则有

$$y = \begin{cases} 1 & \sum_{j=1}^n x_{ej} - \sum_{k=1}^n x_{ik} \geq \theta \\ 0 & \sum_{j=1}^n x_{ej} - \sum_{k=1}^n x_{ik} < \theta \end{cases}$$



6.3 典型神经网络模型I——MP模型

利用MP模型分别表示的或、与、非逻辑关系式：

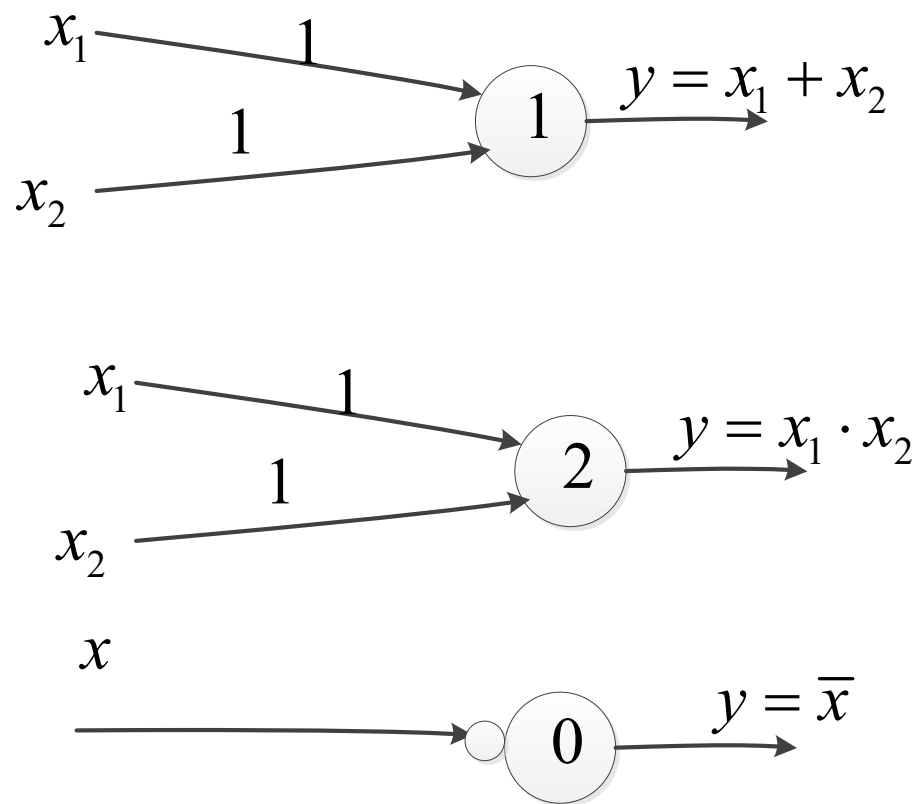


图6-12 用MP模型实现的布尔逻辑

6.3 典型神经网络模型——感知机神经模型

6.3.2 感知机神经网络

1. 感知机的网络结构

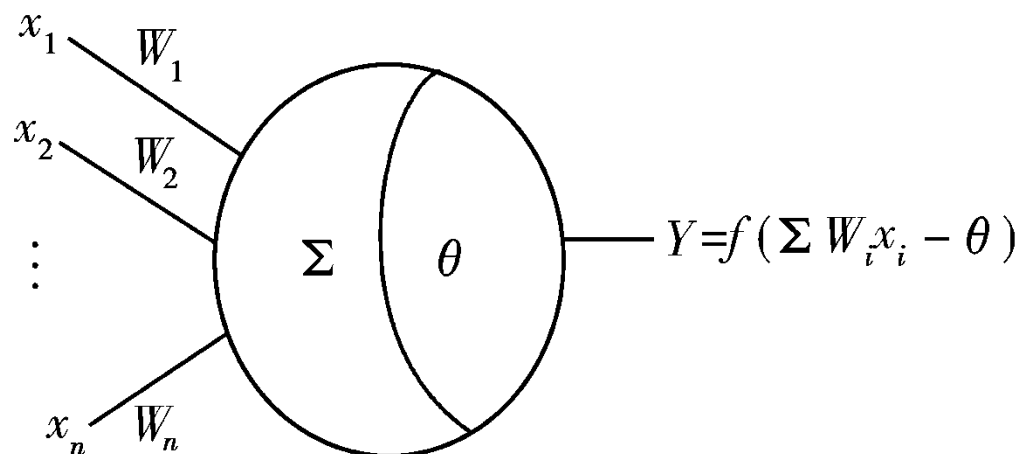
1957年心理学家Frank Rosenblatt及其合作者为了研究大脑的存储、学习和认知过程而提出的一类神经网络模型，并称其为感知机（Perceptron）。

感知机较MP模型又进一步，它的输入可以是非离散量，它的权不仅是非离散量，而且可以通过调整学习而得到。

6.3 典型神经网络模型——感知机神经模型

感知机(Perceptron) 是一个具有单层计算单元的神经网络，由线性阈值元件组成。

■感知机的结构



- 激发函数为阈值型函数，当其输入的加权和大于或等于阈值时，输出为**1**，否则为**0**或**-1**。
- 它的权系数 **W** 可变，这样它就可以学习。

6.3 典型神经网络模型——感知机神经模型

■感知机的学习算法

为方便起见，将阈值 θ (它也同样需要学习) 并入 W 中，令 $W_{n+1} = -\theta$ ， X 向量也相应地增加一个分量 $x_{n+1}=1$ ，则

$$y = f\left(\sum_{i=1}^{n+1} W_i x_i\right)$$

学习算法：

① 给定初始值：赋给 $W_i(0)$ 各一个较小的随机非零值，这里 $W_i(t)$ 为 t 时刻第 i 个输入的权 ($1 \leq i \leq n$)， $W_{n+1}(t)$ 为 t 时刻的阈值；

② 输入一样本 $X=(x_1, \dots, x_n, 1)$ 和 它的期望输出 d ；

③ 计算实际输出 $Y(t) = f\left(\sum_{i=1}^{n+1} W_i(t) x_i\right)$

④ 修正权 W ： $W_i(t+1) = W_i(t) + \eta[d - Y(t)]x_i$ ， $i=1, 2, \dots, n+1$

⑤ 转到②直到 W 对一切样本均稳定不变为止。

6.3 典型神经网络模型——感知机神经模型

由于感知机的权值可以通过学习调整而得到，因此它被认为是最早提出的一种神经网络模型。

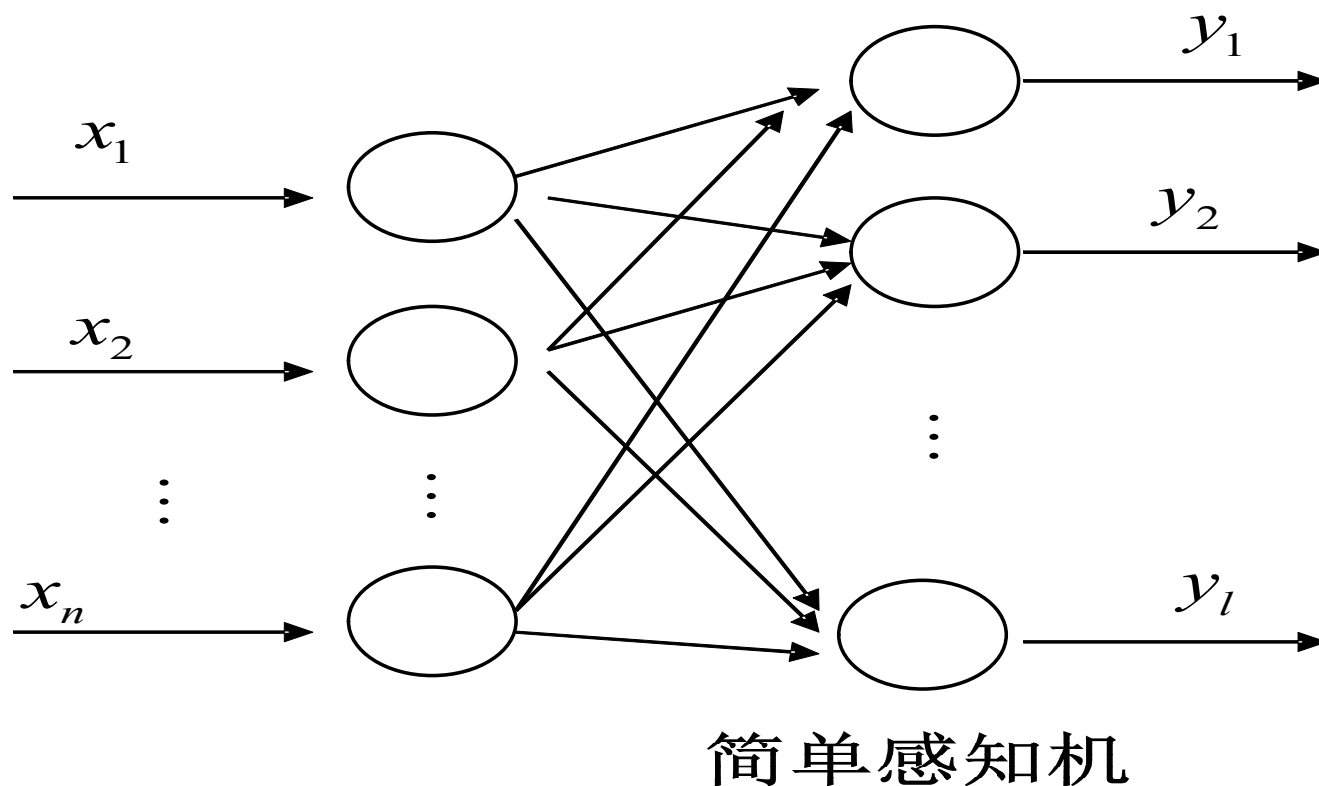


图6-13 感知机的结构（简单感知机）

6.3 典型神经网络模型——感知机神经模型

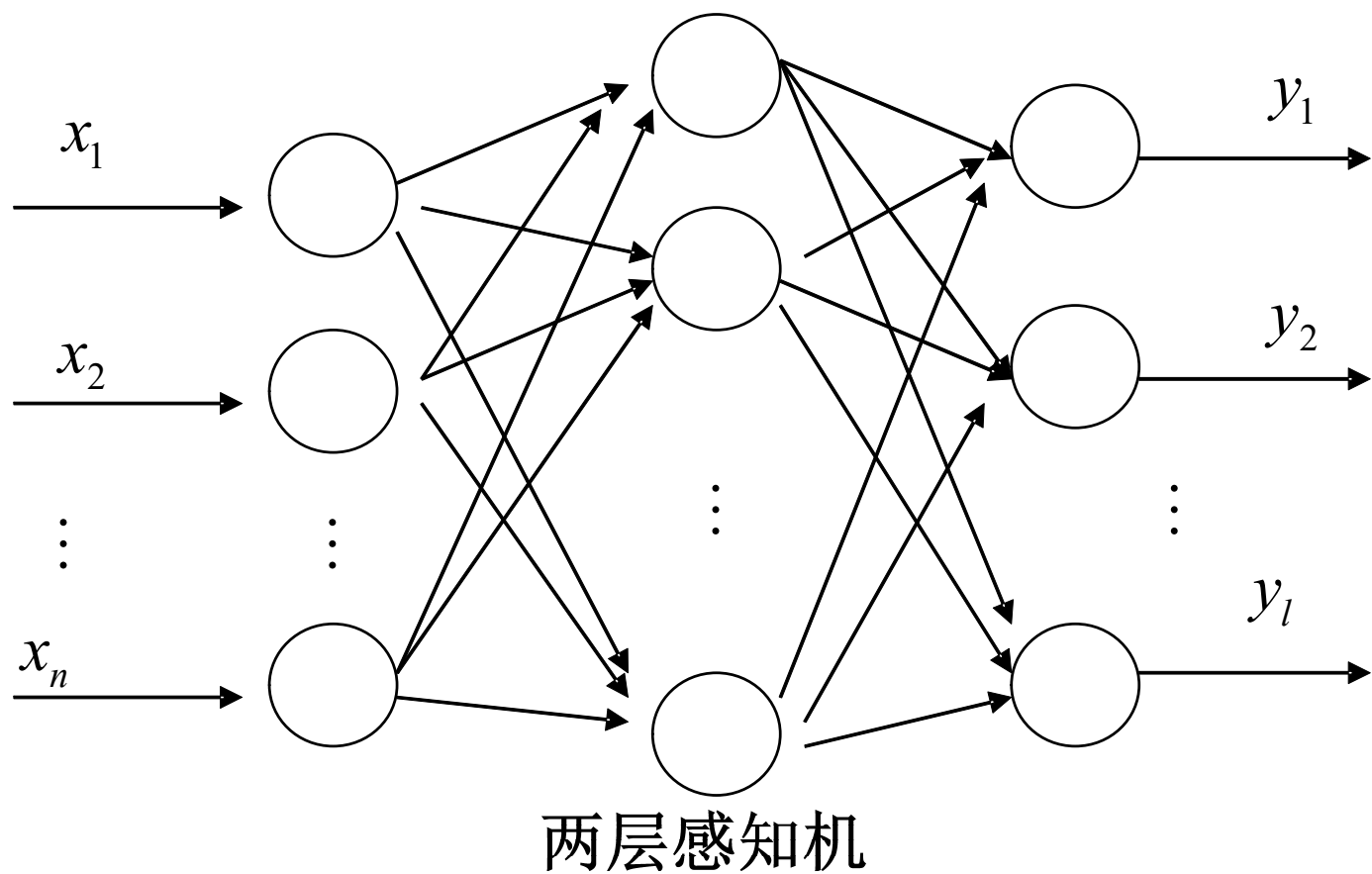
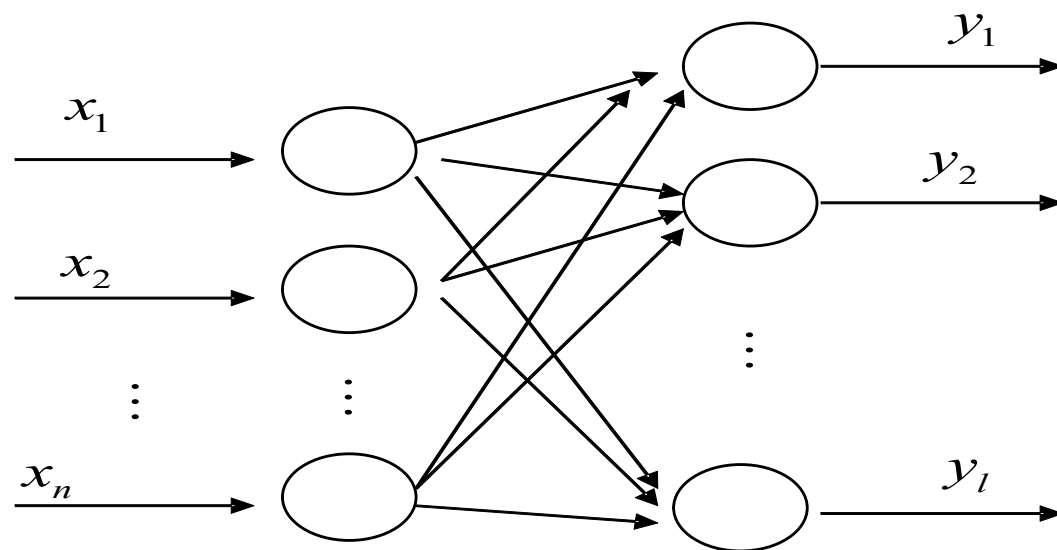


图6-13 感知机的结构（两层感知机）

6.3 典型神经网络模型——感知机神经模型

在这种模型中，输入模式 $x = [x_1, x_2, \dots, x_n]^T$ 通过各输入端点分配给下一层的各节点，下一层就是中间层，中间层可以是一层也可以是多层，最后通过输出层节点得到输出模式 $y = [y_1, y_2, \dots, y_l]^T$ 。

在这类前馈网络中没有层内连接，也没有隔层的前馈连接。每一节点只能前馈联接接到其下一层的所有节点。



简单感知机

6.3 典型神经网络模型——感知机神经模型

一个两层感知机结构（包括输入层、一个隐含层和一个输出层），有两层连接权，其中输入层和隐含层单元间的连接权值是随机设定的固定值，不可调节；输出层与隐含层单元间的连接权值是可调的。

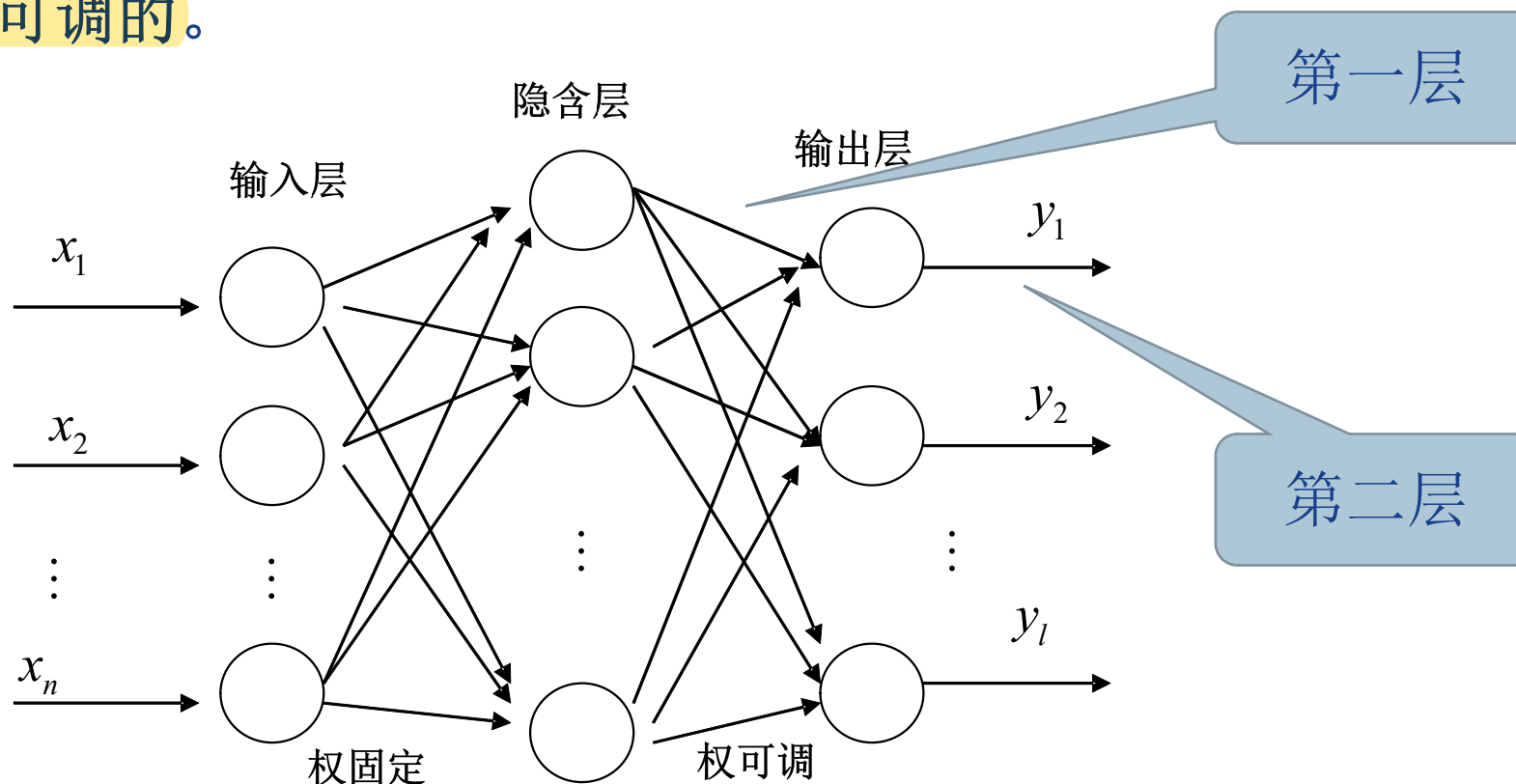


图6-14 两层感知机的结构

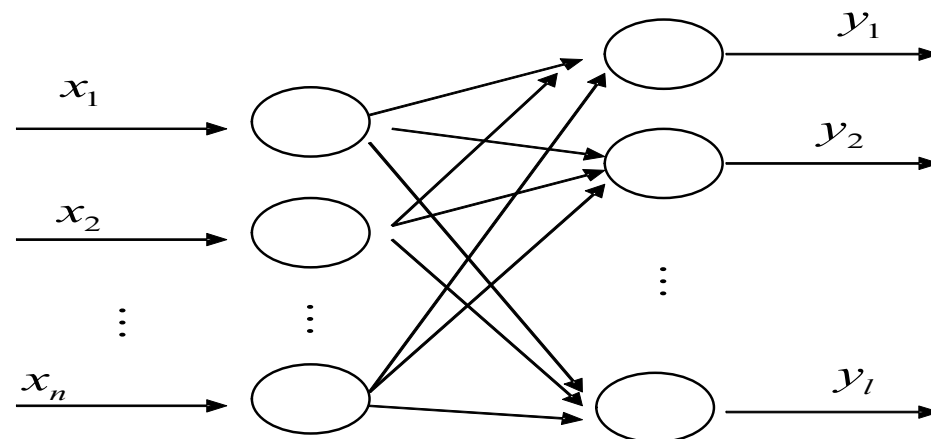
6.3 典型神经网络模型——感知机神经模型

输出层的第 i 个神经元的输入总和和输出分别为

$$net_i = \sum_{j=1}^n w_{ij} x_j - \theta_i, y_i = f(net_i), i = 1, 2, \dots, l \quad (6-1)$$

式中 θ_i 为输出层神经元 i 的阈值， n 为输入层的节点数，即输入的个数。 $f(\cdot)$ 为激活函数。感知机中的激活函数使用了阶跃限幅函数，因此感知机能够将输入向量分为两个区域，即有

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



简单感知机

6.3 典型神经网络模型——感知机神经模型

2. 感知机学习

- ◆ 感知机的学习是典型的有教师学习，可以通过样本训练达到学习的目的。
- ◆ 训练的条件有两个：训练集和训练规则。
- ◆ 感知机的训练集就是由若干个输入——输出模式对构成的一个集合，所谓输入输出模式对是指一个输入模式及其期望输出模式所组成的向量对。它包括二进制值输入模式及其期望输出模式，每个输出对应一个分类。
- ◆ **F. Rosenblatt** 已证明，如果两类模式是线性可分的（指存在一个超平面将它们分开），则算法一定收敛。

6.3 典型神经网络模型——感知机神经模型

2. 感知机学习算法

设有 N 个训练样本，在感知机训练期间，不断用训练集中的每个模式对训练网络。当给定某一个样本 p 的输入输出模式对时，感知机输出单元会产生一个实际输出向量，用期望输出与实际的输出之差来修正网络连接权值。

权值的修正采用简单的 δ 学习规则，它是一个有教师的学习过程，其基本思想是利用某个神经单元的期望输出与实际的输出之间的差来调整该神经单元与上一层中相应神经单元的连接权值，最终减小这种偏差。也就是说，**神经单元之间连接权的变化正比于输出单元期望输出与实际的输出之差。**

6.3 典型神经网络模型——感知机神经模型

简单感知机输出层的任意神经元 i 的连接权值 w_{ij} 和阈值 θ_i 修正公式为

$$\Delta w_{ij} = \eta(t_i^p - y_i^p) \cdot x_j^p = \eta e_i^p \cdot x_j^p \quad (6-2)$$

$$\Delta \theta_i = \eta(t_i^p - y_i^p) \cdot 1 = \eta e_i^p \quad (6-3)$$

t_i^p ——在样本 p 作用下的第 i 个神经元的期望输出;

y_i^p ——在样本 p 作用下的第 i 个神经元的实际输出;

η ——为学习速率 ($0 < \eta \leq 1$)，用于控制权值调整速度。

6.3 典型神经网络模型——感知机神经模型

期望输出与实际输出之差为

$$e_i^p = t_i^p - y_i^p(k) = \begin{cases} 1 & (t_i^p = 1, y_i^p(k) = 0) \\ 0 & (t_i^p = y_i^p(k)) \\ -1 & (t_i^p = 0, y_i^p(k) = 1) \end{cases} \quad (6-4)$$

由此可见，权值变化量与两个量有关：输入状态 x_j 和输出误差。当且仅当输出单元 i 有输出误差，且相连输入状态 x_j 为1时，修正权值或增加一个量或减少一个量。

$$\Delta w_{ij} = \eta(t_i^p - y_i^p) \cdot x_j^p = \eta e_i^p \cdot x_j^p$$

6.3 典型神经网络模型——感知机神经模型

3. 感知机的线性可分性

感知机可以对线性可分性输入模式进行分类，例如，两维输入 x_1, x_2 。其分界线为 $n-1$ 维（ $2-1=1$ ）直线，则

$$w_1x_1 + w_2x_2 - \theta = 0$$

- ◆ 当 $w_1x_1 + w_2x_2 \geq \theta$ 时， $y=1$ ，此时把输入模式划分为“1”类，用“●”代表输出为1的输入模式。
- ◆ 当 $w_1x_1 + w_2x_2 < \theta$ 时， $y=0$ ，此时把输入模式划分为“0”类，用“○”代表输出为0的输入模式。

$$net_i = \sum_{j=1}^n w_{ij}x_j - \theta_i, y_i = f(net_i), i = 1, 2, \dots, l$$

6.3 典型神经网络模型——感知机神经模型

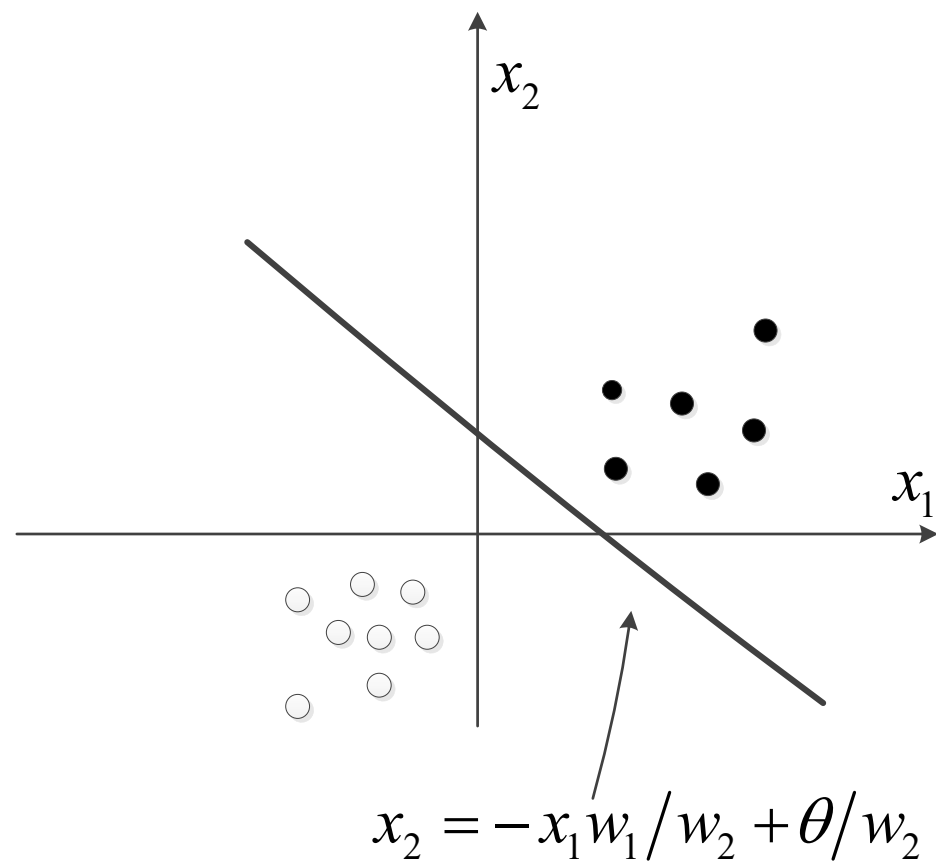


图6-15 线性分割图

6.3 典型神经网络模型——感知机神经模型

例6-1： 利用简单感知机对“与”、“或”和“异或”问题进行分类。

解：逻辑“与”、“或”和“异或”的真值表如表6-1所示。

表6-1 真值表

X_1	X_2	Y		
		AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

6.3 典型神经网络模型——感知机神经模型

- ◆ 对于逻辑“与”和逻辑“或”可将输入模式按照其输出分成两类，线性可分。
- ◆ 对于逻辑“异或”将输入模式按照其输出分成两类，即这四个输入模式分布在二维空间中，线性不可分。

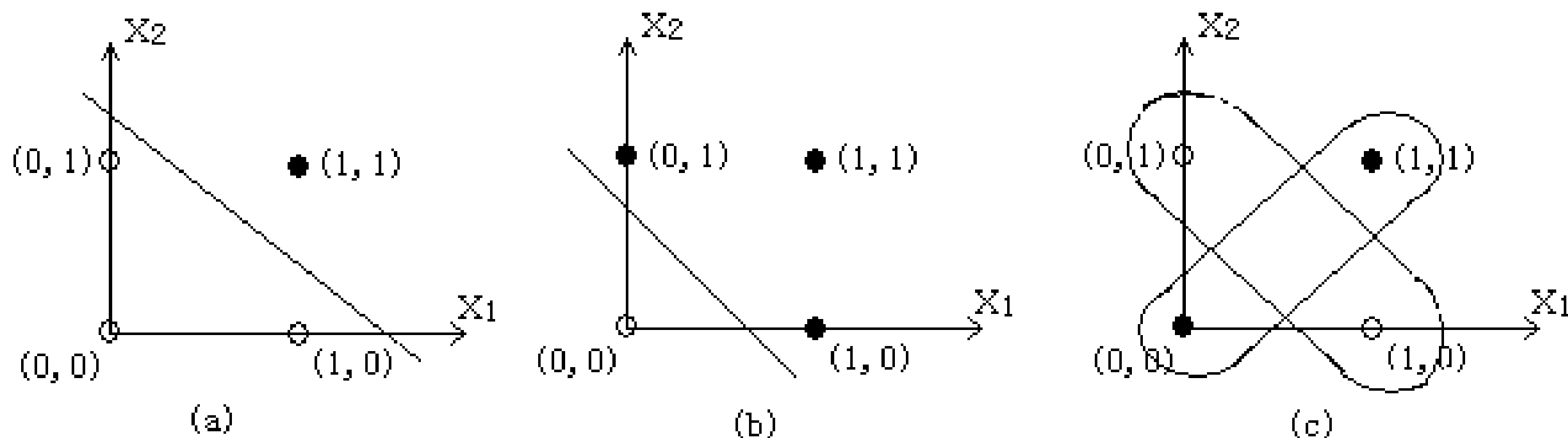


图6-16 AND,OR和XOR输入模式的空间分布

6.3 典型神经网络模型——感知机神经模型

感知机为解决逻辑异或问题，可以设计一个多层的网络，即含有输入层，隐含层和输出层的结构。

可以证明，只要隐含层单元数足够多，用多层感知机网络可实现任何模型分类。但是，隐单元的状态不受外界直接控制，这给多层网络的学习带来极大困难。

6.3 典型神经网络模型——感知机神经模型

4. 感知机收敛性定理

定理 1.1 如果样本输入函数是线性可分的，那么感知机学习算法经过有限次迭代后可收敛到正确的权值或权向量。

定理 1.2 假定隐含层单元可以根据需要自由设置，那么用双隐含层感知机可以实现任意的二值逻辑函数。

6.3 典型神经网络模型——感知机神经模型

5. 感知机网络学习算法的计算步骤

- (1) 初始化：置所有的加权系数为最小的随机数；
- (2) 提供训练集：给出顺序赋值的输入向量 x_1, x_2, \dots, x_n 和期望的输出向量（训练集） t_1, t_2, \dots, t_l ；
- (3) 计算实际输出：按式(6-1)计算输出层各神经元的输出；

$$net_i = \sum_{j=1}^n w_{ij} x_j - \theta_i, y_i = f(net_i), i = 1, 2, \dots, l \quad (6-1)$$

6.3 典型神经网络模型——感知机神经模型

(4) 按式(6-4)计算期望值与实际输出的误差;

$$e_i^p = t_i^p - y_i^p(k) = \begin{cases} 1 & (t_i^p = 1, y_i^p(k) = 0) \\ 0 & (t_i^p = y_i^p(k)) \\ -1 & (t_i^p = 0, y_i^p(k) = 1) \end{cases} \quad (6-4)$$

(5) 按式(6-2)和式(6-3)调整输出层的的加权系数 w_{ij} 和阈值 θ_i 。

$$\Delta w_{ij} = \eta(t_i^p - y_i^p) \cdot x_j^p = \eta e_i^p \cdot x_j^p \quad (6-2)$$

$$\Delta \theta_i = \eta(t_i^p - y_i^p) \cdot 1 = \eta e_i^p \quad (6-3)$$

返回计算(3)步, 直到误差满足要求为止。

6.3 典型神经网络模型——感知机神经模型

例6-2 建立一个感知机网络，使其能够完成“与”的功能。

解 为了完成“与”函数，建立一个两输入、单输出的一个感知机网络。根据表6-1中“与”函数的真值表，可得训练样本的输入向量为： $X=[0\ 0\ 1\ 1; 0\ 1\ 0\ 1]$ ，

目标向量为： $T=[0\ 0\ 0\ 1]$ 。

根据感知机网络学习算法的计算步骤，利用MATLAB语言编写程序。

ex6_2

6.3 典型神经网络模型——感知机神经模型

结果显示:

$$W_{ij} = \begin{matrix} 2.6462 & 2.3252 \end{matrix}$$

$$b = 4.6169$$

$$y = \begin{matrix} 0 & 0 & 0 & 1 \end{matrix}$$

结果显示:

$$W_{ij} = \begin{matrix} 1.6992 & 2.7595 \end{matrix}$$

$$b = 3.6557$$

$$y = \begin{matrix} 0 & 0 & 0 & 1 \end{matrix}$$

结果显示:

$$W_{ij} = \begin{matrix} 0.6787 & 0.7577 \end{matrix}$$

$$b = 0.7431$$

$$y = \begin{matrix} 0 & 0 & 0 & 1 \end{matrix}$$

6.3 典型神经网络模型——感知机神经模型

使用新的训练样本——或（OR）

输入向量为： $X=[0 \ 0 \ 1 \ 1; 0 \ 1 \ 0 \ 1]$,

目标向量为： $T=[0 \ 1 \ 1 \ 1]$ 。

结果显示：

$$W_{ij} = \begin{matrix} & 0 & 1 & 1 & 1 \\ 0 & 0.7655 & 0.7952 & & \end{matrix}$$

$$b = 0.1869$$

$$y = \begin{matrix} 0 & 1 & 1 & 1 \end{matrix}$$

$$W_{ij} = \begin{matrix} & 0 & 1 & 1 & 1 \\ 0 & 0.7094 & 0.7547 & & \end{matrix}$$

$$b = 0.2760$$

$$y = \begin{matrix} 0 & 1 & 1 & 1 \end{matrix}$$

6.3 典型神经网络模型——感知机神经模型

使用新的训练样本——异或（OR）

输入向量为： $X=[0\ 0\ 1\ 1; 0\ 1\ 0\ 1]$,

目标向量为： $T=[0\ 1\ 1\ 0]$ 。

输出结果：

epoch = 10000 %训练的最大次数

Wij = 1.0e+03 * [7.3596 7.3599]

b = 1.4721e+04

y = 0 0 0 0

6.3 典型神经网络模型——自适应线性神经网络

6.3.3 自适应线性神经网络

线性神经网络是一种简单的神经元网络，它可以由一个或多个线性神经元构成。1962年由美国斯坦福大学教授Berhard Widrow提出的自适应线性元件网络（**Adaline**）是线性神经网络最早的典型代表，它是一个由输入层和输出层构成的**单层前馈型**网络。它与感知机神经网络的不同之处在于其每个神经元的传输函数为**线性函数**，因此自适应线性神经网络的输出可以取任意值，而感知机神经网络的输出只能是**1或0**。

6.3 典型神经网络模型——自适应线性神经网络

线性神经网络采用由 Berhard Widrow 和 Marcian Hoff 共同提出的一种新的学习规则，也称为 **Widrow-Hoff** 学习规则，或者 **LMS (Least Mean Square)** 算法来调整网络的权值和阈值。自适应线性神经网络的学习算法比感知机网络的学习算法的收敛速度和精度都有较大的提高。

6.3 典型神经网络模型——自适应线性神经网络

自适应线性神经网络主要用于函数逼近、信号预测、系统辨识、模式识别和控制等领域。

1. 线性神经网络结构

对于具有 M 个输入、 L 个输出的线性神经网络。输出层的第 i 个神经元的输入总和（即激活函数）和输出分别为

$$net_i = \sum_{j=1}^M w_{ij} x_j - \theta_i, y_i = f(net_i), i = 1, 2, \dots, L \quad (6-5)$$

式中 θ_i 为输出层神经元 i 的阈值， M 为输入层的节点数，即输入的个数。 $f(.)$ 为激活函数，它为线性函数的传输函数。

6.3 典型神经网络模型——自适应线性神经网络

2. 自适应线性神经网络的学习

在训练网络的学习阶段，设有 N 个训练样本，先假定用其中的某一个样本 p 的输入输出模式对 X_p 和 $\{T_p\}$ 对网络进行训练，输出层的第 i 个神经元在样本 p 作用下的输入为：

$$net_i = \sum_{j=1}^M w_{ij} x_j^p - \theta_i, \quad i = 1, 2, \dots, L$$

式中 θ_i 为输出层神经元 i 的阈值， M 为输入层的节点数，即输入的个数。

6.3 典型神经网络模型——自适应线性神经网络

输出层第 i 个神经元的输出为:

$$y_i^p = f\left(net_i^p\right), i = 1, 2, \cdots, L$$

式中 $f(.)$ 为线性激活函数。

它将网络的输入原封不动地输出，因此有

$$y_i^p = net_i^p = \sum_{j=1}^M w_{ij} x_j^p - \theta_i, \quad i = 1, 2, \cdots, L \quad (6-6)$$

对于每一样本 p 的输入模式对的二次型误差函数为

$$J_p = \frac{1}{2} \sum_{i=1}^L \left(t_i^p - y_i^p\right)^2 = \frac{1}{2} \sum_{i=1}^L e_i^2 \quad (6-7)$$

6.3 典型神经网络模型——自适应线性神经网络

则系统对所有 N 个训练样本的总误差函数为

$$J = \sum_{p=1}^N J_p = \frac{1}{2} \sum_{p=1}^N \sum_{i=1}^L (t_i^p - y_i^p)^2 = \frac{1}{2} \sum_{p=1}^N \sum_{i=1}^L e_i^2 \quad (6-8)$$

式中 N 为模式样本对数； L 为网络输出节点数； t_i^p 表示在样本 p 作用下的第 i 个神经元的期望输出， y_i^p 表示在样本 p 作用下的第 i 个神经元的实际输出。

6.3 典型神经网络模型——自适应线性神经网络

线性神经网络加权系数修正是采用 **Widrow-Hoff** 学习规则，又称为最小均方误差算法（**LMS**）。它的实质是利用梯度最速下降法，使权值沿误差函数的负梯度方向改变。**Widrow-Hoff** 学习规则的权值变化量正比于网络的输出误差及网络的输入矢量。根据梯度法，可得输出层的任意神经元 i 的加权系数修正公式为

$$\Delta w_{ij} = -\eta \frac{\partial J_p}{\partial w_{ij}} \quad (i=1,2,\dots,L; j=1,2,\dots,M)$$

式中学习速率 η 为常值，当 $\eta = \frac{\alpha}{\|X_p\|_2^2}$, α 为常值，当 $0 < \alpha < 2$ 时，可使算法收敛。 η 随着输入样本 X_p 自适应地调整。

6.3 典型神经网络模型——自适应线性神经网络

因为
$$\frac{\partial J_p}{\partial w_{ij}} = \frac{\partial J_p}{\partial net_i^p} \cdot \frac{\partial net_i^p}{\partial w_{ij}} = \frac{\partial J_p}{\partial net_i^p} \cdot x_j^p$$

定义
$$\delta_i^p = -\frac{\partial J_p}{\partial net_i^p} = -\frac{\partial J_p}{\partial y_i^p} \cdot \frac{\partial y_i^p}{\partial net_i^p} = (t_i^p - y_i^p) \cdot f'(net_i^p) = e_i \cdot f'(net_i^p)$$

则
$$\Delta w_{ij} = \eta \cdot \delta_i^p \cdot x_j^p$$

由于激活函数 $f(.)$ 为线性函数，故 $f'(net_i^p) = 1$

所以

$$\frac{\partial J_p}{\partial w_{ij}} = -e_i \cdot x_j^p$$

6.3 典型神经网络模型——自适应线性神经网络

可得输出层的任意神经元 i 的加权系数修正公式为

$$\Delta w_{ij} = \eta \cdot (t_i^p - y_i^p) \cdot x_j^p = \eta \cdot e_i \cdot x_j^p \quad (6-9)$$

同理，阈值 θ_i 的修正公式为

$$\Delta \theta_i = \eta(t_i - y_i) = \eta e_i \quad (6-10)$$

以上两式构成了最小均方误差算法（**LMS**），或 **Widrow-Hoff** 学习算法，它实际上也是 δ 学习规则的一种特例。

6.3 典型神经网络模型——自适应线性神经网络

3. 线性神经网络学习算法的计算步骤

- (1) 初始化：置所有的加权系数为最小的随机数；
- (2) 提供训练集：给出顺序赋值的输入向量 x_1, x_2, \dots, x_M 和期望的输出向量（训练集） t_1, t_2, \dots, t_L ；
- (3) 计算实际输出：按式(6-5)和式(6-6)计算输出层各神经元的输出。

$$net_i = \sum_{j=1}^M w_{ij} x_j - \theta_i, y_i = f(net_i), i = 1, 2, \dots, L \quad (6-5)$$

$$y_i^p = net_i^p = \sum_{j=1}^M w_{ij} x_j^p - \theta_i, \quad i = 1, 2, \dots, L \quad (6-6)$$

6.3 典型神经网络模型——自适应线性神经网络

(4) 按式(6-7)或式(6-8)计算期望值与实际输出的误差。

$$J_p = \frac{1}{2} \sum_{i=1}^L (t_i^p - y_i^p)^2 = \frac{1}{2} \sum_{i=1}^L e_i^2 \quad (6-7)$$

$$J = \sum_{p=1}^N J_p = \frac{1}{2} \sum_{p=1}^N \sum_{i=1}^L (t_i^p - y_i^p)^2 = \frac{1}{2} \sum_{p=1}^N \sum_{i=1}^L e_i^2 \quad (6-8)$$

(5) 按式(6-9)和式(6-10)调整输出层的的加权系数 w_{ij} 和阈值 θ_i 。

$$\Delta w_{ij} = \eta \cdot (t_i^p - y_i^p) \cdot x_j^p = \eta \cdot e_i \cdot x_j^p \quad (6-9)$$

$$\Delta \theta_i = \eta(t_i - y_i) = \eta e_i \quad (6-10)$$

返回计算(3)步，直到误差满足要求为止。

6.3 典型神经网络模型IV——BP神经网络

6.3.4 BP神经网络

1986年 D.E. Rumelhart 和 J.L. McClelland 提出了一种利用误差反向传播训练算法的神经网络，简称 BP (Back Propagation) 网络，是一种有隐含层的多层前馈网络，系统地解决了多层网络中隐含单元连接权的学习问题。

如果网络的输入节点数为 M 、输出节点数为 L ，则此神经网络可看成是从 M 维欧氏空间到 L 维欧氏空间的映射。

6.3 典型神经网络模型IV——BP神经网络

主要作用：

模式识别与分类：用于语言、文字、图像的识别，医学特征的分类和诊断等。

函数逼近：用于非线性控制系统的建模、机器人的轨迹控制及其他工业控制等。

数据压缩：编码压缩和恢复，图像数据的压缩和存储，以及图像特征的抽取等。

6.3 典型神经网络模型IV——BP神经网络

1. BP算法原理

BP学习算法的基本原理是梯度最速下降法，它的中心思想是调整权值使网络总误差最小。也就是采用梯度搜索技术，以期使网络的实际输出值与期望输出值的误差均方值为最小。网络学习过程是一种误差边向后传播边修正权系数的过程。

6.3 典型神经网络模型IV——BP神经网络

多层网络运行BP学习算法时，实际上包含了正向和反向传播两个阶段。

在正向传播过程中，输入信息从输入层经隐含层逐层处理，并传向输出层，每一层神经元的状态只影响下一层神经元的状态。

如果在输出层不能得到期望输出，则转入反向传播，将误差信号沿原来的连接通道返回，通过修改各层神经元的权值，使误差信号最小。

6.3 典型神经网络模型IV——BP神经网络

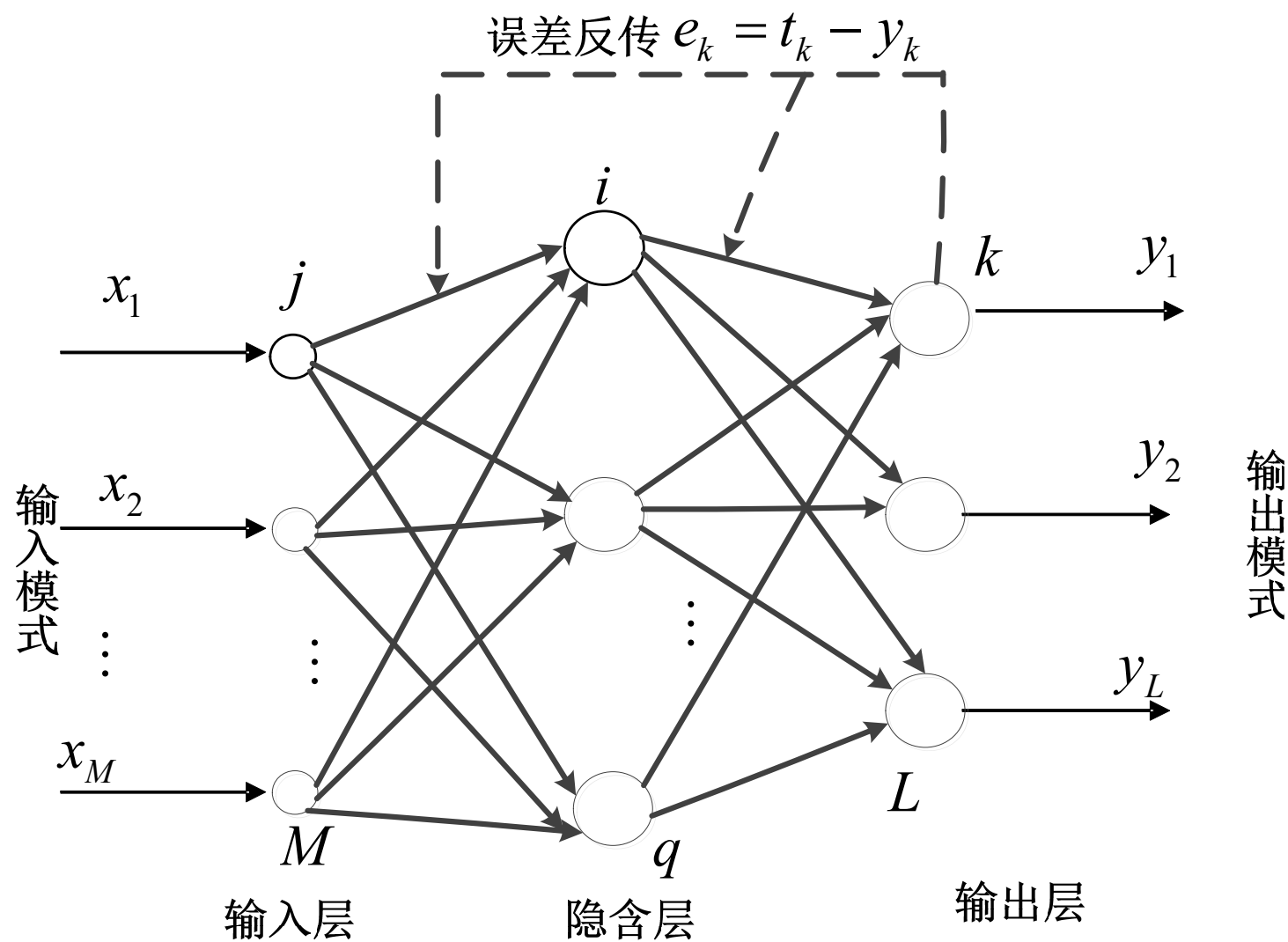


图 6-17 BP网络的结构图

6.3 典型神经网络模型IV——BP神经网络

BP网络结构与多层感知机的差别：

- ❖ 多层感知机结构中只有一层权值可调，其他各层权值是固定的、不可学习的；**BP**网络的每一层连接权值都可通过学习来调节。
- ❖ 感知机结构中的处理单元为线性输入输出关系，单元为二进制的0或1；而**BP**网络的基本处理单元（输入层除外）为非线性的输入输出关系，一般选用**S（sigmoid）**型函数。

6.3 典型神经网络模型IV——BP神经网络

2. BP网络的前馈计算

在训练网络的学习阶段，设有N个训练样本，先假定用其中的某一个样本 p 的输入输出模式对 X_p 和 T_p 对网络进行训练，隐含层的第 i 个神经元在样本 p 作用下的输入为：

$$net_i^p = \sum_{j=1}^M w_{ij} o_j^p - \theta_i = \sum_{j=1}^M w_{ij} x_j^p - \theta_i \quad (i=1,2,\dots, q) \quad (6-11)$$

式中， x_j^p 和 o_j^p 分别为输入节点 j 在样本 p 作用时的输入和输出，对输入节点而言两者相当； w_{ij} 为输入层神经元 j 与隐含层神经元 i 之间的连接权值； θ_i 为隐含层神经元 i 的阈值， M 为输入层的节点数，即输入的个数。

6.3 典型神经网络模型IV——BP神经网络

隐含层第*i*个神经元的输出为

$$o_i^p = g(net_i^p) \quad (i=1, 2, \dots, q) \quad (6-12)$$

式中 $g(.)$ 为激活函数。对于Sigmoid型激活函数

$$g(x) = \frac{1}{1 + \exp[-(x + \theta_1)/\theta_0]}$$

式中，参数 θ_1 表示偏值，正的 θ_1 使激活函数水平向左移动。 θ_0 的作用是调节Sigmoid函数形状的，较小的 θ_0 使 Sigmoid 函数逼近一个阶跃限幅函数，而较大的 θ_0 将使 Sigmoid 函数变的较为平坦，如图6-18所示。

6.3 典型神经网络模型IV——BP神经网络

$$g(x) = \frac{1}{1 + \exp[-(x + \theta_1)/\theta_0]}$$

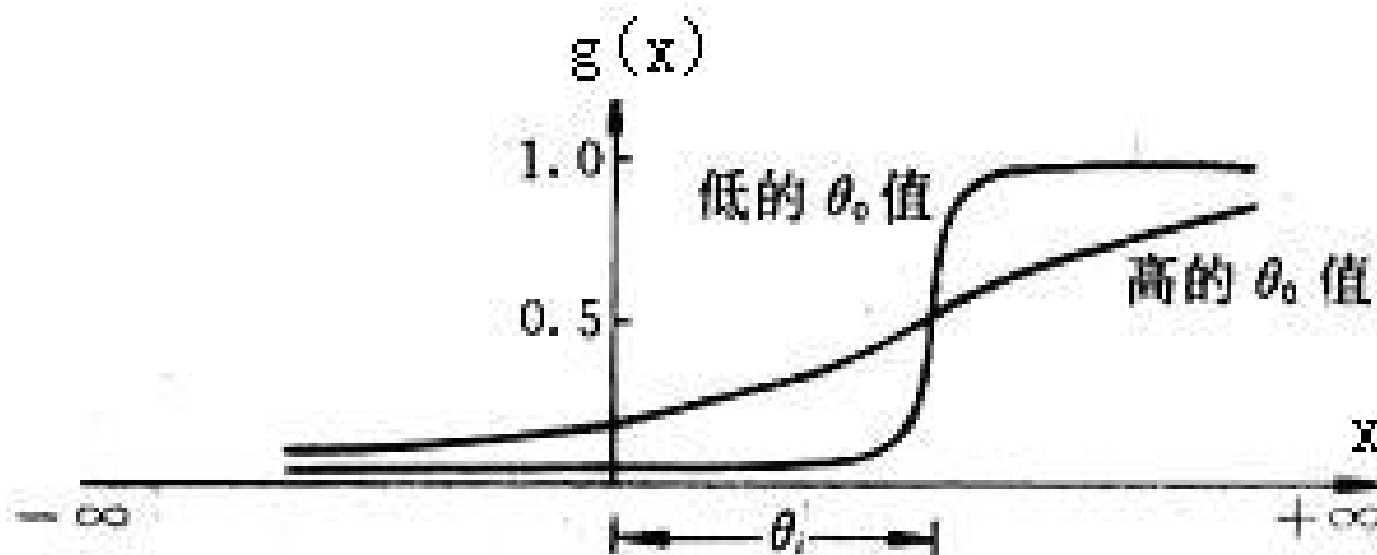


图6-18 具有偏值的和形状调节的Sigmoid函数

6.3 典型神经网络模型IV——BP神经网络

隐含层激活函数 $g(net_i^p)$ 的微分函数为

$$g'(net_i^p) = g(net_i^p)[1 - g(net_i^p)] = o_i^p(1 - o_i^p) \\ (i=1,2,\dots, q) \quad (6-13)$$

隐含层第 i 个神经元的输出 o_i^p 通过权系数向前传播到输出层第 k 个神经元，作为它的输入之一，而输出层第 k 个神经元的总输入为

$$net_k^p = \sum_{i=1}^q w_{ki} o_i^p - \theta_k \quad (k=1,2,\dots, L) \quad (6-14)$$

式中 w_{ki} 为隐含层神经元 i 与输出层神经元 k 之间的连接权值； θ_k 为输出层神经元 k 的阈值， q 为隐含层的节点数。

6.3 典型神经网络模型IV——BP神经网络

输出层的第 k 个神经元的实际输出为

$$o_k^p = g(net_k^p) \quad (k=1, 2, \dots, L) \quad (6-15)$$

输出层激活函数 $g(net_k^p)$ 的微分函数为

$$g'(net_k^p) = g(net_k^p)[1 - g(net_k^p)] = o_k^p(1 - o_k^p) \quad (k=1, 2, \dots, L) \quad (6-16)$$

6.3 典型神经网络模型IV——BP神经网络

若其输出与给定模式对的期望输出 t_k^p 不一致，则将其误差信号从输出端反向传播回来，并在传播过程中对加权系数不断修正，使在输出层神经元上得到所需要的期望输出值 t_k^p 为止。对样本 p 完成网络权系数的调整后，再送入另一样本模式对进行类似学习，直到完成 N 个样本的训练学习为止。

6.3 典型神经网络模型IV——BP神经网络

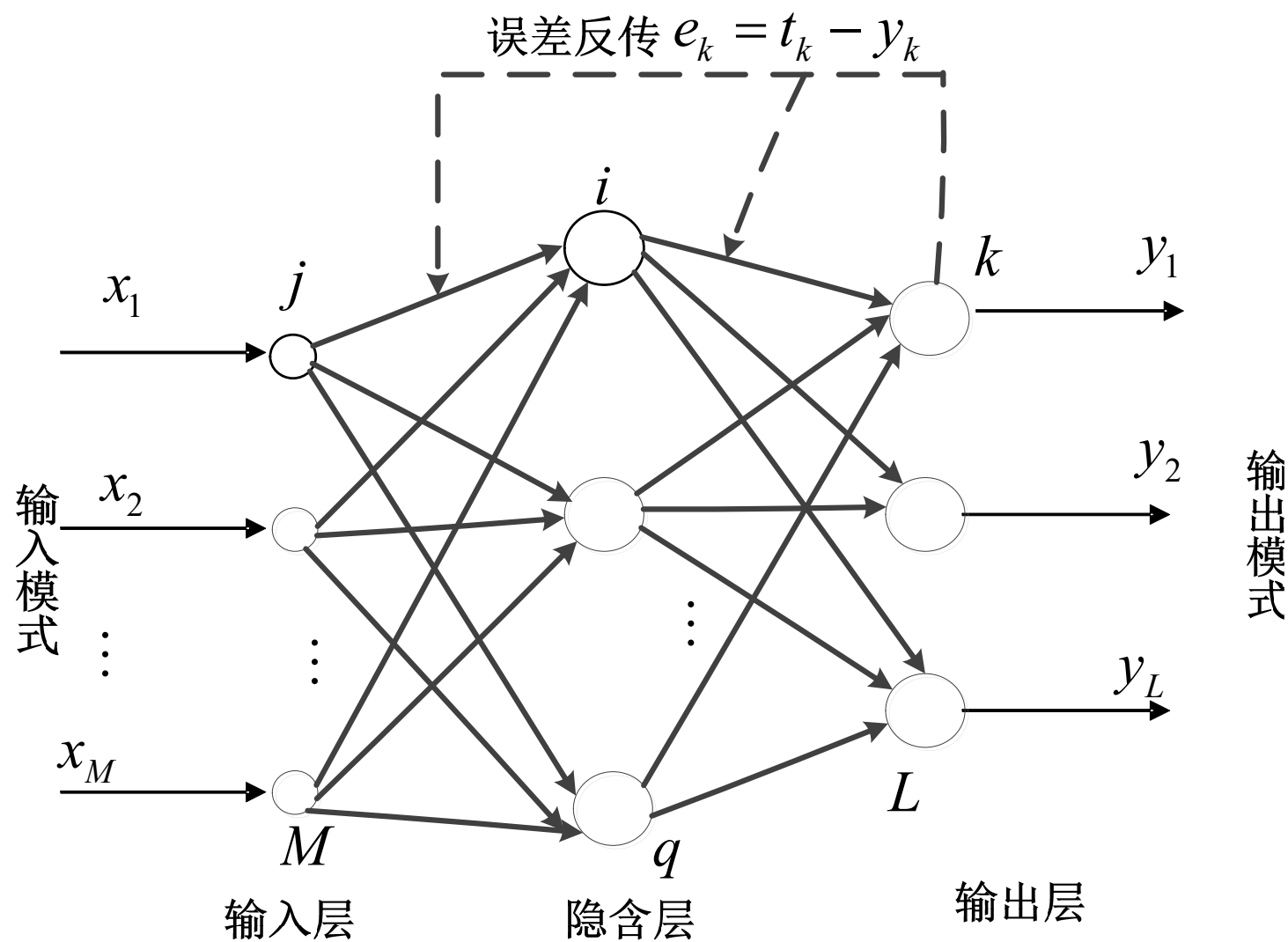


图 6-17 BP网络的结构图

6.3 典型神经网络模型IV——BP神经网络

3. BP网络权系数的调整规则

对于每一样本 p 的输入模式对的二次型误差函数为

$$J_p = \frac{1}{2} \sum_{k=1}^L (t_k^p - o_k^p)^2 \quad (6-17)$$

则系统对所有 N 个训练样本的总误差函数为

$$J = \sum_{p=1}^N J_p = \frac{1}{2} \sum_{p=1}^N \sum_{k=1}^L (t_k^p - o_k^p)^2 \quad (6-18)$$

式中 N 为模式样本对数； L 为网络输出节点数。

6.3 典型神经网络模型IV——BP神经网络

在线学习与离线学习

在线学习 (基于 J_p) : 对训练集内每个模式对逐一更新网络权值的一种学习方式, 其特点是学习过程中需要较少的存储单元, 但有时会增加网络的整体输出误差。

离线学习 (批处理学习, 基于 J) : 用训练集内所有模式依次训练网络, 累加各权值修正量并统一修正网络权值的一种学习方式, 它能使权值变化沿最快速下降方向进行。

6.3 典型神经网络模型IV——BP神经网络

在线学习(基于 J_p):

(1) 输出层权系数的调整

权系数应按 J_p 函数梯度变化的反方向调整,使网络逐渐收敛。根据梯度法,可得输出层每个神经元权系数的修整公式为

$$\begin{aligned}\Delta w_{ki} &= -\eta \frac{\partial J_p}{\partial w_{ki}} = -\eta \frac{\partial J_p}{\partial net_k^p} \cdot \frac{\partial net_k^p}{\partial w_{ki}} \\ &= -\eta \frac{\partial J_p}{\partial net_k^p} \cdot \frac{\partial}{\partial w_{ki}} \left(\sum_{i=1}^q w_{ki} o_i^p - \theta_k \right) \\ &= -\eta \frac{\partial J_p}{\partial net_k^p} o_i^p\end{aligned}\tag{6-19}$$

式中 η 为学习速率, $\eta>0$ 。

6.3 典型神经网络模型IV——BP神经网络

定义

$$\begin{aligned}\delta_k^p &= -\frac{\partial J_p}{\partial net_k^p} = -\frac{\partial J_p}{\partial o_k^p} \cdot \frac{\partial o_k^p}{\partial net_k^p} \\ &= (t_k^p - o_k^p) g'(net_k^p) \\ &= (t_k^p - o_k^p) o_k^p (1 - o_k^p)\end{aligned}$$

因此输出层的任意神经元k的加权系数修正公式为

$$\Delta w_{ki} = \eta \delta_k^p o_i^p = \eta o_k^p (1 - o_k^p) (t_k^p - o_k^p) o_i^p \quad (6-20)$$

式中 o_k^p 输出节点 k 在样本 p 作用时的输出； o_i^p 隐含节点 i 在样本 p 作用时的输出； t_k^p 在样本 p 输入输出对作用时输出节点 k 的目标值。

6.3 典型神经网络模型IV——BP神经网络

(2) 隐含层权系数的调整

根据梯度法，可得隐含层每个神经元权系数的修正公式为

$$\begin{aligned}\Delta w_{ij} &= -\eta \frac{\partial J_p}{\partial w_{ij}} = -\eta \frac{\partial J_p}{\partial net_i^p} \cdot \frac{\partial net_i^p}{\partial w_{ij}} = -\eta \frac{\partial J_p}{\partial net_i^p} \cdot \frac{\partial}{\partial w_{ij}} \left(\sum_{j=1}^M w_{ij} o_j^p - \theta_i \right) \\ &= -\eta \frac{\partial J_p}{\partial net_i^p} \cdot o_j^p = \eta \delta_i^p o_j^p\end{aligned}$$

式中 η 为学习速率， $\eta > 0$ 。

$$\delta_i^p = -\frac{\partial J_p}{\partial net_i^p} = -\frac{\partial J_p}{\partial o_i^p} \cdot \frac{\partial o_i^p}{\partial net_i^p} = -\frac{\partial J_p}{\partial o_i^p} \cdot g'(net_i^p) = -\frac{\partial J_p}{\partial o_i^p} \cdot o_i^p (1 - o_i^p)$$

6.3 典型神经网络模型IV——BP神经网络

由于隐含层一个单元输出的改变会影响与该单元相连接的所有输出单元的输入，即

$$\begin{aligned}-\frac{\partial J_p}{\partial o_i^p} &= -\sum_{k=1}^L \frac{\partial J_p}{\partial net_k^p} \cdot \frac{\partial net_k^p}{\partial o_i^p} = -\sum_{k=1}^L \frac{\partial J_p}{\partial net_k^p} \cdot \frac{\partial}{\partial o_i^p} \left(\sum_{i=1}^q w_{ki} o_i^p - \theta_k \right) \\ &= \sum_{k=1}^L \left(-\frac{\partial J_p}{\partial net_k^p} \right) \cdot w_{ki} = \sum_{k=1}^L \delta_k^p \cdot w_{ki}\end{aligned}$$

因此，隐含层的任意神经元 i 的加权系数修正公式为

$$\Delta w_{ij} = \eta \delta_i^p o_j^p = \eta o_i^p (1 - o_i^p) \left(\sum_{k=1}^L \delta_k^p \cdot w_{ki} \right) o_j^p \quad (6-21)$$

式中 o_i^p 是隐含节点 i 在样本 p 作用时的输出； o_j^p 输入节点 j 在样本 p 作用时的输出，即输入节点 j 的输入（因对输入节点两者相当）。

6.3 典型神经网络模型IV——BP神经网络

输出层的任意神经元 k 在样本 p 作用时的加权系数增量公式为：

$$w_{ki}(k+1) = w_{ki}(k) + \eta \delta_k^p o_i^p \quad (6-22)$$

隐含层的任意神经元 i 在样本 p 作用时的加权系数增量公式为：

$$w_{ij}(k+1) = w_{ij}(k) + \eta \delta_i^p o_j^p \quad (6-23)$$

6.3 典型神经网络模型IV——BP神经网络

离线学习——基于 J 的加权系数空间的梯度搜索

输出层和隐含层的任意神经元 k 和 i 在所有样本作用时的加权系数增量公式为：

$$w_{ki}(k+1) = w_{ki}(k) + \eta \sum_{p=1}^N \delta_k^p o_i^p \quad (6-24)$$

$$w_{ij}(k+1) = w_{ij}(k) + \eta \sum_{p=1}^N \delta_i^p o_j^p \quad (6-25)$$

式中 δ_k^p 和 δ_i^p 的计算方法同上，即

$$\delta_k^p = o_k^p (1 - o_k^p) (t_k^p - o_k^p) \quad (6-26)$$

$$\delta_i^p = o_i^p (1 - o_i^p) \left(\sum_{k=1}^L \delta_k^p \cdot w_{ki} \right) \quad (6-27)$$

6.3 典型神经网络模型IV——BP神经网络

在线学习与离线学习

在线学习(基于 J_p)：对训练集内每个模式对逐一更新网络权值的一种学习方式，其特点是学习过程中需要较少的存储单元，但有时会增加网络的整体输出误差。使用在线学习时一般使学习因子足够小，以保证用训练集内每一个模式训练一次之后，权值的总体变化充分接近于最快速下降。

离线学习（批处理学习, 基于 J ）：用训练集内所有模式依次训练网络，累加各权值修正量并统一修正网络权值的一种学习方式，它能使权值变化沿最快速下降方向进行。其缺点是学习过程中需要较多的存储单元，好处是学习速度快。

6.3 典型神经网络模型IV——BP神经网络

4. BP网络学习算法的计算步骤

- (1) 初始化：置所有的加权系数为最小的随机数；
- (2) 提供训练集：给出顺序赋值的输入向量 $x^{(1)}, x^{(2)}, \dots, x^{(N)}$ 和期望的输出向量（训练集） $t^{(1)}, t^{(2)}, \dots, t^{(N)}$ ；
- (3) 计算实际输出：按式 (6-11)-(6-16) 计算隐含层、输出层各神经元的输出。
- (4) 按式 (6-17) 或式 (6-18) 计算期望值与实际输出的误差
- (5) 按式 (6-22) 或式 (6-24) 调整输出层的的加权系数 w_{ki}
- (6) 按式 (6-23) 或式 (6-25) 调整隐含层的的加权系数 w_{ij}
- (7) 返回计算 (3) 步，直到误差满足要求为止。

6.3 典型神经网络模型IV——BP神经网络

5. 使用BP算法时应注意的几个问题

(1) 学习速率 η 的选择非常重要。在学习初始阶段 η 选得大些可使学习速度加快，但当临近最佳点时， η 必须相当小，否则加权系数将产生反复振荡而不能收敛。采用变学习速率方案，令学习速率 η 随着学习的进展而逐步减少，可收到较好的效果。引入惯性系数 α 的办法，也可使收敛速度加快。

(2) 采用S形激活函数式时，由于输出层各神经元的理想输出值只能接近于1或0，而不能达到1或0。这样在设置各训练样本的期望输出分量 t_k^p 时，不能设置为1或0，以设置为0.9或0.1较为适宜。

6.3 典型神经网络模型IV——BP神经网络

(3) S型非线性激活函数 $g(x)$ 随着 $|x|$ 的增大梯度下降, 即 $|g'(x)|$ 减小并趋于0, 不利于权值的调整, 因此希望 $|x|$ 工作在较小的区域, 故应考虑网络的输入。若实际问题给以网络的输入量较大, 需做归一化处理, 网络的输出也要进行相应的处理。对于具体问题, 需经调试而定, 且需经验知识的积累。

(4) 学习开始时如何设置加权系数 w_{ij} 和 w_{ki} 的初值非常重要, 如将所有初值设置为相等值, 则由式(6-23)可知; 所有隐含层加权系数的调整量相同, 从而使这些加权系数总相等, 因此, 各加权系数的初值以设置为随机数为宜。

$$w_{ij}(k+1) = w_{ij}(k) + \eta \delta_i^p o_j^p \quad (6-23)$$

6.3 典型神经网络模型IV——BP神经网络

(5) **BP**网络的学习过程中系统可能陷入某些局部最小值，或某些静态点，或在这些点之间振荡。在这种情况下，不管进行多少次迭代，系统都存在很大误差。因此，在学习过程中，应尽量避免落入某些局部最小值点上，引入惯性项有可能使网络避免落入某一局部最小值。

6.3 典型神经网络模型IV——BP神经网络

6. BP网络学习算法的改进

BP网络的主要优点:

- 只要有足够多的隐层和隐节点，BP网络可以逼近任意的非线性映射关系；
- BP网络的学习算法属于全局逼近的方法，因而它具有较好的泛化能力。

BP网络的主要缺点:

- 收敛速度慢；
- 局部极值；
- 难以确定隐层和隐节点的个数。

6.3 典型神经网络模型IV——BP神经网络

改进方法

(I) 引入惯性项

有时为了使收敛速度快些，可在加权系数修正公式中增加一个惯性项，使加权系数变化更平稳些。

输出层的任意神经元 k 在样本 p 作用时的加权系数增量公式为：

$$w_{ki}(k+1) = w_{ki}(k) + \eta \delta_k^p o_i^p + \alpha[w_{ki}(k) - w_{ki}(k-1)] \quad (6-29)$$

隐含层的任意神经元 i 在样本 p 作用时的加权系数增量公式为：

$$w_{ij}(k+1) = w_{ij}(k) + \eta \delta_i^p o_j^p + \alpha[w_{ij}(k) - w_{ij}(k-1)] \quad (6-30)$$

式中 α 为惯性系数， $0 < \alpha < 1$ 。

6.3 典型神经网络模型IV——BP神经网络

(2) 引入动量项

$$w(k+1) = w(k) + \eta[(1-\alpha)D(k) + \alpha D(k-1)] \quad (6-31)$$

式中， $w(k)$ 既可表示单个的连接权系数，也可以表示连接权向量（其元素为连接权系数）； $D(k) = -\partial J / \partial w(k)$ 为 k 时刻的负梯度； $D(k-1)$ 是 $k-1$ 时刻的负梯度； η 为学习速率， $\eta > 0$ ； α 为动量因子， $0 \leq \alpha < 1$ 。

6.3 典型神经网络模型IV——BP神经网络

(3) 变尺度法

标准的BP学习算法所采用的是一阶梯度法，因而收敛较慢。

若采用二阶梯度法，则可以大大改善收敛性。二阶梯度法的算法为

$$w(k+1) = w(k) - \eta [\nabla^2 J(k)]^{-1} \nabla J(k)$$

其中

$$\nabla J(k) = \frac{\partial J}{\partial w(k)}, \nabla^2 J(k) = \frac{\partial^2 J}{\partial w^2(k)}, 0 < \eta \leq 1$$

6.3 典型神经网络模型IV——BP神经网络

变尺度法的算法:

$$w(k+1) = w(k) + \eta H(k)D(k)$$

$$H(k) = H(k-1) - \frac{\Delta w(k)\Delta w^T(k)}{\Delta w^T(k)\Delta D(k)} - \frac{H(k-1)\Delta D(k)\Delta D^T(k)H(k-1)}{\Delta D^T(k)H(k-1)\Delta D(k)}$$

$$\Delta w(k) = w(k) - w(k-1)$$

$$\Delta D(k) = D(k) - D(k-1)$$

6.3 典型神经网络模型IV——BP神经网络

(4) 变步长法

$$w(k+1) = w(k) + \eta(k)D(k)$$

$$\eta(k) = 2^\lambda \eta(k-1)$$

$$\lambda = \text{sgn}[D(k)D(k-1)]$$

以上公式说明，当连续两次迭代其梯度方向相同时，表明下降太慢，这时可使步长加倍；当连续两次迭代其梯度方向相反时，表明下降过头，这时可使步长减半。

6.3 典型神经网络模型IV——BP神经网络

需要引入动量项时，上述算法可修正改为

$$w(k+1) = w(k) + \eta(k)[(1-\eta)D(k) + \eta D(k-1)]$$

$$\eta(k) = 2^\lambda \eta(k-1)$$

$$\lambda = \text{sgn}[D(k)D(k-1)]$$

在使用该算法时，由于步长在迭代过程中自适应进行调整，因此对于不同的连接权系数实际采用了不同的学习率，也就是说误差代价函数 J 在超曲面上在不同的方向按照各自比较合理的步长向极小点逼近。

6.3 典型神经网络模型IV——BP神经网络

例 6-3 具有一个两个输入单元，两个隐含单元和一个输出单元的两层BP神经网络，如图6-19所示，设学习样本为 $N=4$ ，写出BP网络学习算法批处理（离线学习）的计算步骤。假设样本集的输入为 $X=\{x^1, x^2, x^3, x^4\}$ ，目标为 $T=\{t^1, t^2, t^3, t^4\}$ 。 $x^p=[x_1^p, x_2^p]$ ($p=1, 2, 3, 4$)。

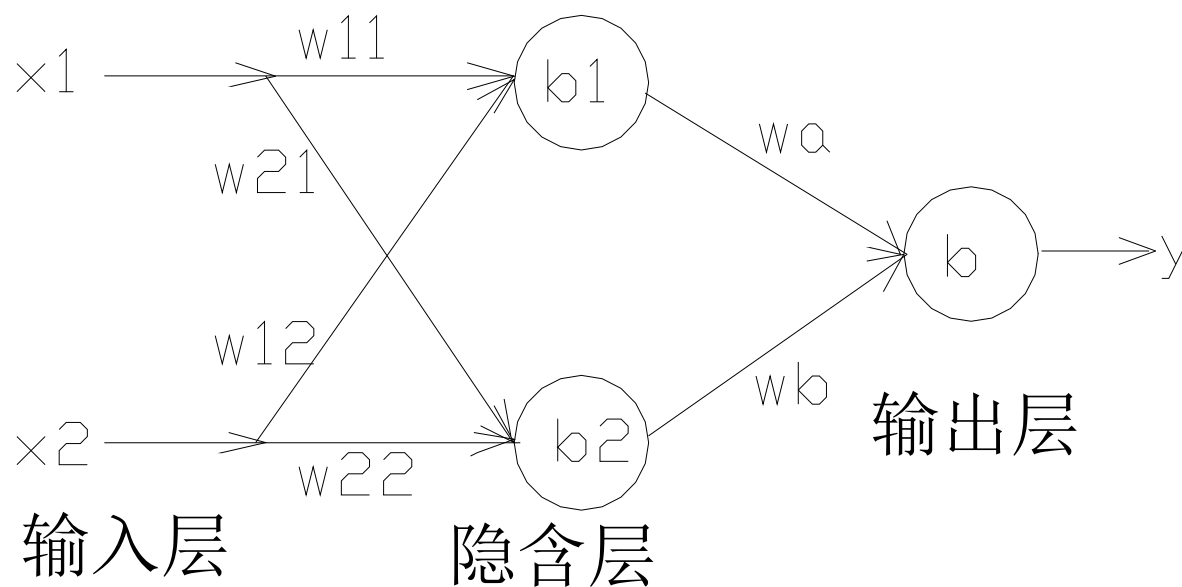


图 6-19 简单BP网络

6.3 典型神经网络模型IV——BP神经网络

解：(1) 置所有的加权系数及偏值的初始值为最小的随机数：

隐含层的加权系数及偏值的初始值为：

$$w_{11}(0), w_{12}(0), w_{21}(0), w_{22}(0), b_1(0), b_2(0)$$

输出层的加权系数及偏值的初始值为：

$$w_a(0), w_b(0), b(0)$$

6.3 典型神经网络模型IV——BP神经网络

(2) 根据式(6-12)和式(6-15)分别计算样本集中所有样本的隐含层和输出层各节点的输出值，即隐含层第1个神经元的输出为：

$$o_1^p = g(w_{11} \cdot x_1^p + w_{12} \cdot x_2^p + b_1) \quad (p=1,2,3,4)$$

隐含层第2个神经元的输出为：

$$o_2^p = g(w_{21} \cdot x_1^p + w_{22} \cdot x_2^p + b_2) \quad (p=1,2,3,4)$$

输出层神经元的输出为：

$$y^p = g(w_a \cdot o_1^p + w_b \cdot o_2^p + b) \quad (p=1,2,3,4)$$

6.3 典型神经网络模型IV——BP神经网络

(3) 根据式(6-19)和式(6-21)及目标值分别计算在所有样本作用下的各层误差，即输出层的误差为：

$$\delta^p = y^p (1 - y^p) (t^p - y^p) \quad (p=1,2,3,4)$$

隐含层第1个神经元的误差为：

$$\delta_1^p = \delta^p \cdot w_a \cdot o_1^p (1 - o_1^p) \quad (p=1,2,3,4)$$

隐含层第2个神经元的误差为：

$$\delta_2^p = \delta^p \cdot w_b \cdot o_2^p (1 - o_2^p) \quad (p=1,2,3,4)$$

6.3 典型神经网络模型IV——BP神经网络

(4) 根据式(6-25)和式(6-26)调整各层的加权系数及偏值，即

输出层的加权系数及阈值修正公式为：

$$w_a(k+1) = w_a(k) + \eta \sum_{p=1}^4 \delta^p o_1^p$$

$$w_b(k+1) = w_b(k) + \eta \sum_{p=1}^4 \delta^p o_2^p$$

$$b(k+1) = b(k) + \eta \sum_{p=1}^4 \delta^p$$

6.3 典型神经网络模型IV——BP神经网络

隐含层的加权系数及阈值修正公式为：

$$w_{11}(k+1) = w_{11}(k) + \eta \sum_{n=1}^4 \delta_1^p x_1^p$$

$$w_{12}(k+1) = w_{12}(k) + \eta \sum_{p=1}^4 \delta_1^p x_2^p$$

$$w_{21}(k+1) = w_{21}(k) + \eta \sum_{p=1}^4 \delta_2^p x_1^p$$

$$w_{22}(k+1) = w_{22}(k) + \eta \sum_{p=1}^4 \delta_2^p x_2^p$$

$$\theta_1(k+1) = \theta_1(k) + \eta \sum_{p=1}^4 \delta_1^p$$

$$\theta_2(k+1) = \theta_2(k) + \eta \sum_{p=1}^4 \delta_2^p$$

6.3 典型神经网络模型IV——BP神经网络

(5) 计算输出误差:

$$J = \frac{1}{2}[(t^1 - y^1)^2 + (t^2 - y^2)^2 + (t^3 - y^3)^2 + (t^4 - y^4)^2]$$

存在一个 $\varepsilon > 0$, 使 $J < \varepsilon$, 否则返回(2)重新计算, 直到误差满足要求为止。

6.3 典型神经网络模型IV——BP神经网络

例6-4 利用一个三输入两输出的两层BP神经网络训练一个输入为 $[1 \ -1 \ 1]^T$ ，希望的输出均为 $[1 \ 1]^T$ 的神经网络系统。激活函数取

$$g(x) = \frac{2}{1 + \exp(-x)} - 1$$

解：根据BP网络学习算法的计算步骤，用MATLAB语言编写的在线学习程序如下。

ex6_4

6.3 典型神经网络模型IV——BP神经网络

结果显示:

epoch =

3

0k =

0.9987

0.9904

$$w_{ki}(k+1) = w_{ki}(k) + \eta \delta_k^p o_i^p + \alpha [w_{ki}(k) - w_{ki}(k-1)]$$

$$w_{ij}(k+1) = w_{ij}(k) + \eta \delta_i^p o_j^p + \alpha [w_{ij}(k) - w_{ij}(k-1)]$$

6.3 典型神经网络模型 *IV*——BP神经网络

结果显示:

epoch =

2223

Ok =

0.9656

0.9714

6.3 典型神经网络模型Ⅰ——RBF神经网络

6.3.5 径向基神经网络

BP网络用于函数逼近时，权值的调节采用的是负梯度下降法，这种调节权值的方法有它的局限性，即存在着收敛速度慢和局部极小等缺点。

RBF (Radial basis function) 神经网络是由 J. Moody 和 C. Darken 于 20 世纪 80 年代末期提出的一种神经网络，它是具有单隐层的两层前馈网络。RBF 网络模拟了人脑中局部调整相互覆盖接收域（感受野，Receptive Field）的神经网络结构。

RBF神经网络无论在逼近能力、分类能力和学习速度等方面均优于**BP**网络。RBF网络比**BP**网络需要更多的神经元，但是它能够按时间片来训练网络。

6.3 典型神经网络模型V——RBF神经网络

径向基网络是一种局部逼近网络，已证明它能以任意精度逼近任一连续函数。当有很多的训练向量时，这种网络很有效果。

RBF网络的结构与多层前向网络类似，它是具有单隐层的一种两层前向网络。输入层由信号源节点组成。隐含层的单元数视所描述问题的需要而定。输出层对输入的作用作出响应。从输入空间到隐含层空间的变换是非线性的，而从隐含层空间到输出层空间变换是线性的。隐单元的变换函数是RBF，它是一种局部分布的对中心点径向对称衰减的非负非线性函数。

6.3 典型神经网络模型Ⅰ——RBF神经网络

1. 径向基函数网络模型

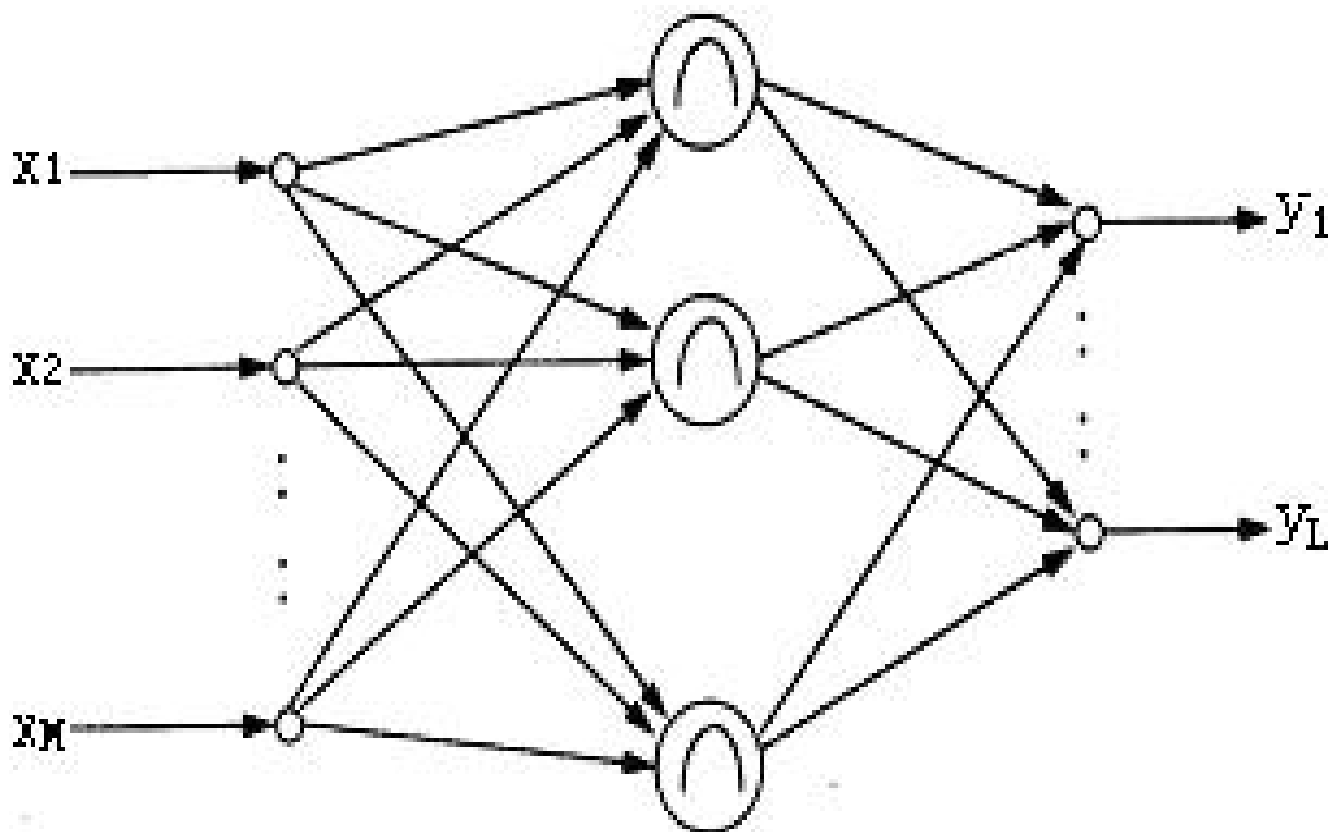


图6-20 RBF网络

6.3 典型神经网络模型Ⅳ——RBF神经网络

2. 网络输出

设网络输入 x 为 M 维向量，输出 y 为 L 维向量，输入输出样本对长度为 N 。

RBF网络的输入层到隐含层实现 $x \rightarrow u_j(x)$ 的非线性映射，径向基网络隐层节点的作用函数一般取下列几种形式：

$$f(x) = \exp[-(x/\delta)^2]$$

$$f(x) = \frac{1}{(\delta^2 + x^2)^a}, a > 0$$

$$f(x) = (\delta^2 + x^2)^\beta, a < \beta < 1$$

6.3 典型神经网络模型Ⅴ——RBF神经网络

RBF网络隐层第 i 个节点的输出可由下式表示：

$$u_i = \exp\left[-\frac{(x - c_i)^T (x - c_i)}{2\sigma_i^2}\right], i = 1, 2, \dots, q \quad (6-32)$$

其中， u_i 是第 i 个隐节点的输出， σ_i 是第 i 个隐节点的标准化常数， q 是隐层节点数， $x = (x_1, x_2, \dots, x_M)^T$ 是输入样本， c_i 是第 i 个隐节点高斯函数的中心向量，此向量是一个与输入样本 x 的维数相同的列向量，即 $c_i = (c_{i1}, c_{i2}, \dots, c_{iM})^T$ 。由上式可知，节点的输出范围在0和1之间，且输入样本愈靠近节点的中心，输出值愈大。当 $x = c_i$ 时， $u_i = 1$ 。

6.3 典型神经网络模型——RBF神经网络

采用高斯函数的优点：

- 表示形式简单，即使对于多变量输入也不增加太多的复杂性。
- 径向对称；
- 光滑性好，任意阶导数存在；
- 由于该基函数表示简单且解析性好，因而便于进行理论分析。

6.3 典型神经网络模型Ⅴ——RBF神经网络

RBF网络的隐含层到输出层实现 $u_i(x) \rightarrow y_k$ 的线性映射，即：

$$y_k = \sum_{i=1}^q w_{ki} u_i - \theta_k \quad (k=1, 2, \dots, L) \quad (6-34)$$

其中， u_i 是第 i 个隐层节点的输出， y_k 是第 k 个输出层节点的输出， w_{ki} 是隐含层到输出层的加权系数， θ_k 是输出层的阈值， q 是隐含层节点数。

6.3 典型神经网络模型Ⅳ——RBF神经网络

3. RBF网络的学习过程

设有 N 个训练样本，则系统对所有 N 个训练样本的总误差函数为

$$J = \sum_{p=1}^N J_p = \frac{1}{2} \sum_{p=1}^N \sum_{k=1}^L (t_k^p - y_k^p)^2 = \frac{1}{2} \sum_{p=1}^N \sum_{k=1}^L e_k^2 \quad (6-35)$$

式中 N 为模式样本对数； L 为网络输出节点数； t_k^p 表示在样本 p 作用下的第 k 个神经元的期望输出， y_k^p 表示在样本 p 作用下的第 k 个神经元的实际输出。

6.3 典型神经网络模型Ⅳ——RBF神经网络

RBF网络的学习过程分为两个阶段:

第一阶段，是无教师学习。是根据所有的输入样本决定隐层各节点的高斯核函数的中心向量 c_i 和标准化常数 σ_i 。

第二阶段，有教师学习。在决定好隐层的参数后，根据样本，利用最小二乘原则，求出隐含层和输出层的权值 w_{ki} 。有时在完成第二阶段的学习后，再根据样本信号，同时校正隐层和输出层的参数，以进一步提高网络的精度。

6.3 典型神经网络模型Ⅰ——RBF神经网络

1) 无教师学习阶段

无教师学习也称为非监督学习，是对所有样本的输入进行聚类，求得各隐层节点的RBF的中心向量 c_i 。

k -均值聚类算法调整中心向量，此算法将训练样本集中的输入向量分为若干族，在每个数据族内找出一个径向基函数中心向量，使得该族内各样本向量距该族中心的距离最小。

6.3 典型神经网络模型V——RBF神经网络

算法步骤如下：

(1) 给定各隐节点的初始中心向量 $c_i(0)$ ($i=1,2,\dots,q$) 和判定停止计算的阈值的 ε ；

(2) 计算距离（欧氏距离）并求出最小距离的节点

$$\begin{aligned} d_i(k) &= \|x(k) - c_i(k-1)\|, 1 \leq i \leq q \\ d_{\min}(k) &= \min d_i(k) = d_r(k) \end{aligned} \tag{6-36}$$

式中， k 为样本序号， r 为中心向量 $c_i(k-1)$ 与输入样本 $x(k)$ 距离最近的隐节点序号；

6.3 典型神经网络模型V——RBF神经网络

(3) 调整中心

$$c_i(k) = c_i(k-1), 1 \leq i \leq q, i \neq r$$

$$c_r(k) = c_r(k-1) + \beta(k) [x(k) - c_r(k-1)] \quad (6-37)$$

式中 $\beta(k)$ 是学习速率, $0 < \beta(k) < 1$ 。

$\beta(k) = \beta(k-1) / (1 + \text{int}(k/q))^{1/2}$, $\text{int}(\cdot)$ 表示对 (\cdot) 进行取整运算。可见, 每经过 q 个样本之后, 调小一次学习速率, 逐渐减至零;

(4) 判定聚类质量

对于全部样本 k ($k=1, 2, \dots, N$) 反复进行以上(2), (3)步, 直至满足以下条件, 则聚类结束。

$$J_e = \sum_{i=1}^q \|x(k) - c_i(k)\|^2 \leq \varepsilon \quad (6-38)$$

6.3 典型神经网络模型Ⅰ——RBF神经网络

2) 有教师学习阶段

RBF网络的隐含层至输出层之间的连接权值 $w_{ki}(k=1, 2, \dots, L; i=1, 2, \dots, q)$ 学习算法为

$$w_{ki}(k+1) = w_{ki}(k) + \eta(t_k - y_k)u_i(x) / u^T u \quad (6-39)$$

其中 $u = [u_1(x) \ u_2(x) \ \dots \ u_q(x)]^T$ ， $u_i(x)$ 为高斯函数。 η 为学习速率，可以证明当 $0 < \eta < 2$ 时可保证该迭代学习算法的收敛性，而实际上通常只取 $0 < \eta < 1$ 。 t_k 和 y_k 分别表示第 k 个输出分量的期望值和实际值。

6.3 典型神经网络模型Ⅰ——RBF神经网络

当 x 远离 c_i 时, $u_i(x)$ 非常小, 因此可作为0对待。因此实际上只当 $u_i(x)$ 大于某一数值 (例如0.05) 时才对相应的权值 w_{ki} 进行修改。经这样处理后RBF神经网络也同样具备局部逼近网络学习收敛快的优点。

6.3 典型神经网络模型Ⅳ——RBF神经网络

4. RBF网络有关的几个问题

(a) 从理论上而言，RBF网络和BP网络一样可近似任何的连续非线性函数。两者的主要不同点是在非线性映射上采用了不同的作用函数。BP网络中的隐层节点使用的是Sigmoid函数，其函数值在输入空间中无限大的范围内为非零值，即作用函数为全局的；而RBF网络中的隐层节点使用的是高斯函数，即它的作用函数则是局部的。

(b) 已证明RBF网络具有唯一最佳逼近的特性，且无局部极小；

(c) 求RBF网络隐节点的中心向量 c_i 和标准化常数 σ_i 是一个困难的问题；

6.3 典型神经网络模型Ⅰ——RBF神经网络

(d) 径向基函数，即径向对称函数有多种。对于同一组样本，**如何选择合适的径向基函数，如何确定隐节点数，以使网络学习达到要求的精度，目前还未解决。当前，用计算机选择、设计、再检验是一种通用的手段；**

(e) RBF 网络用于非线性系统辨识与控制，虽具有唯一最佳逼近的特性，以及无局部极小的优点，但**隐节点的中心难求，这是该网络难以广泛应用的原因；**

(f) **与BP网络收敛速度慢的缺点相反，RBF网络学习速度很快，适用于在线实时控制。这是因为RBF网络把一个难题分解成两个较易解决的问题的缘故。首先，通过若干个隐节点，用聚类方式覆盖全部样本模式；然后，修改输出层的权值，以获得最小映射误差。**

6.3 典型神经网络模型Ⅰ——RBF神经网络

(g) 图6-21是径向基函数神经元传输函数 $\text{radbas}()$ 的示意图，从图中可以注意到 RBF 网络的输入同前面介绍的神经网络的表达式有所不同。其网络输入为权值向量 W 与输入向量 x 之间的向量距离乘以阈值 b ，即 $d=\text{radbas}(\text{dist}(W, x)*b)$ 。

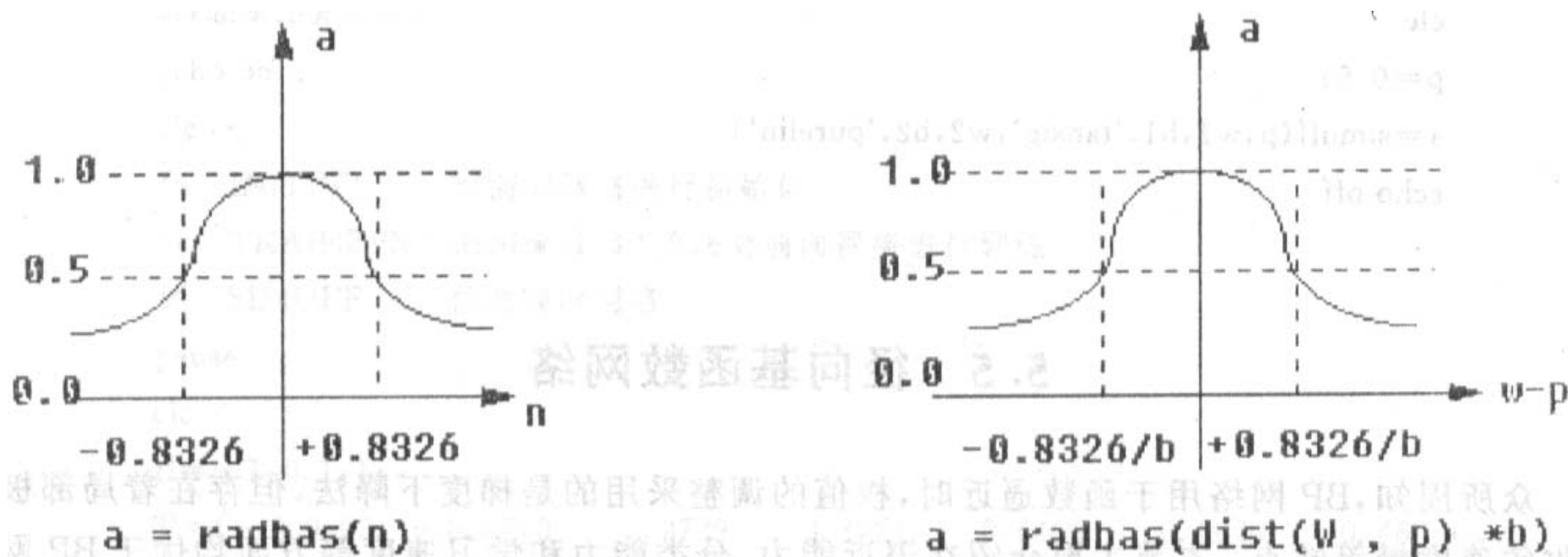


图6-21 传输函数 $\text{radbas}()$ 的示意图

6.3 典型神经网络模型——RBF神经网络

例 6-5 RBF网络设计实例

考虑结构为1-5-1的RBF网络，取网络输入为 $x = x_1$
令

$$\sigma = [\sigma_1 \quad \sigma_2 \quad \sigma_3 \quad \sigma_4 \quad \sigma_5]^T \quad \mathbf{c} = [c_1 \quad c_2 \quad c_3 \quad c_4 \quad c_5]$$
$$u = [u_1 \quad u_2 \quad u_3 \quad u_4 \quad u_5]^T \quad \mathbf{w} = [w_1 \quad w_2 \quad w_3 \quad w_4 \quad w_5]^T$$

则网络输出为

$$y_m(t) = \mathbf{w}^T u = w_1 u_1 + w_2 u_2 + w_3 u_3 + w_4 u_4 + w_5 u_5$$

取网络的输入为 $\sin(t)$ 时，网络的输出如图6-23所示，网络隐含层的输出如图6-24所示。仿真程序为chap6_5sim.mdl, chap6_5rbf.m, chap6_5plot。

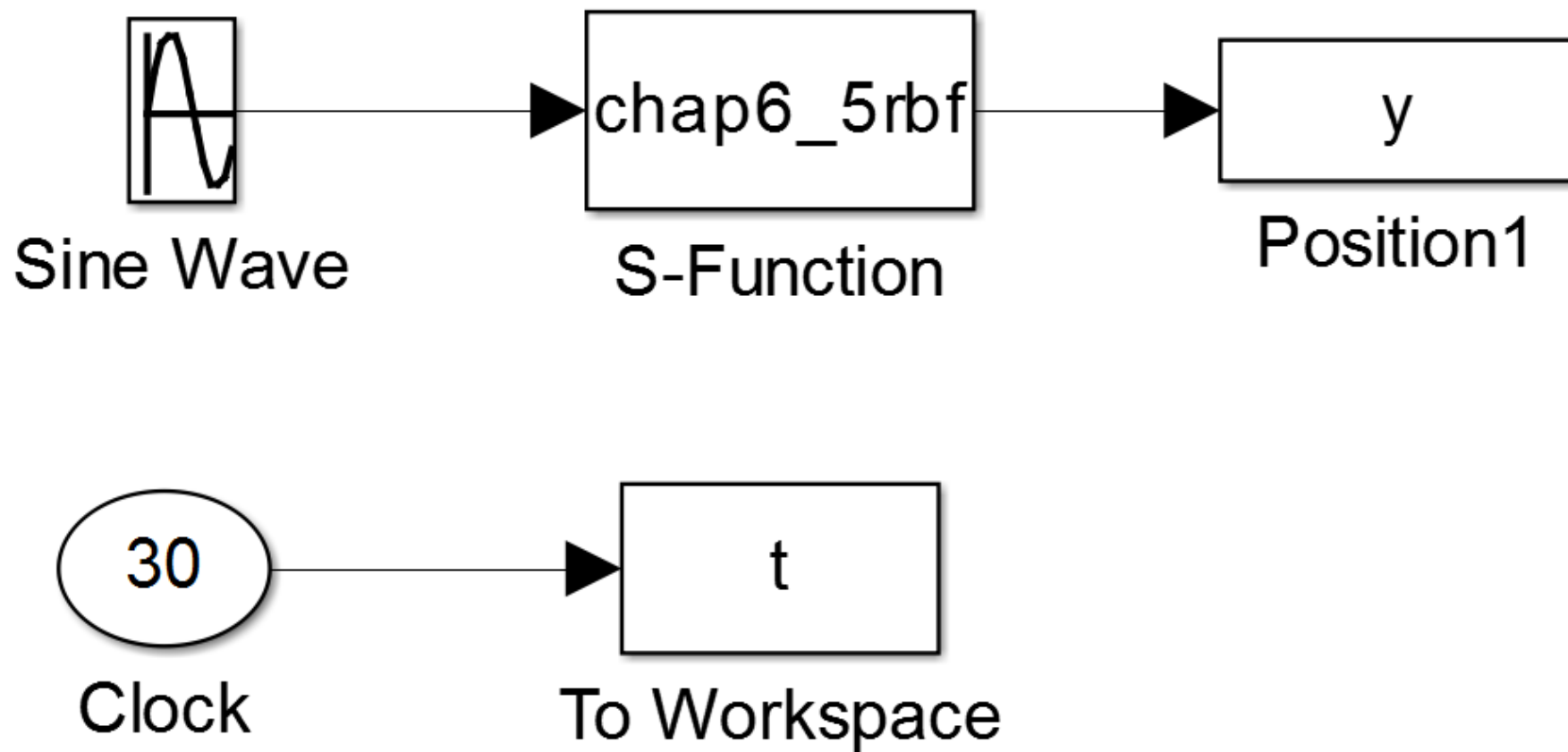


图6-22 chap6_5sim.mdl

6.3 典型神经网络模型Ⅳ——RBF神经网络

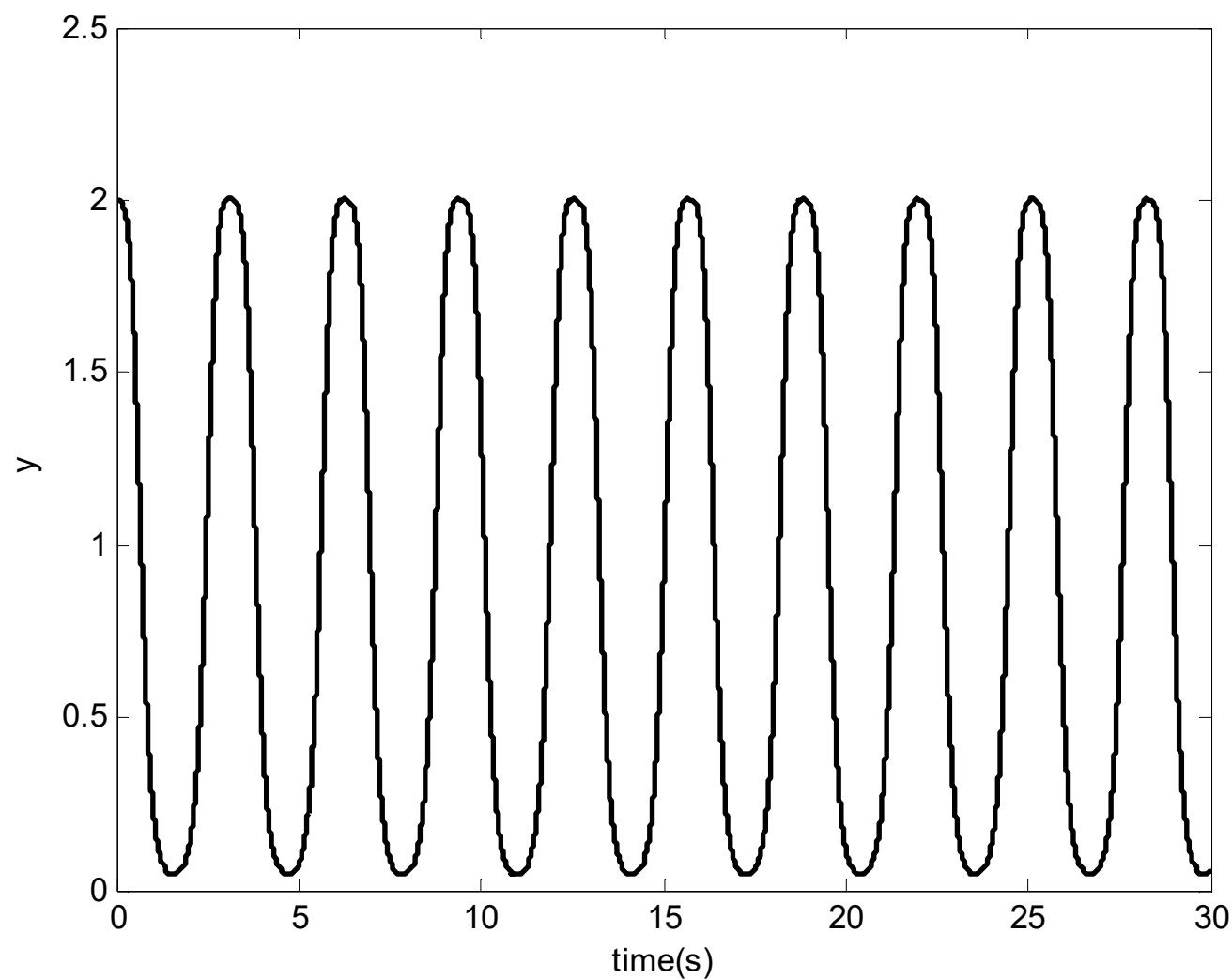


图6-23 RBF网络输出

6.3 典型神经网络模型Ⅰ——RBF神经网络

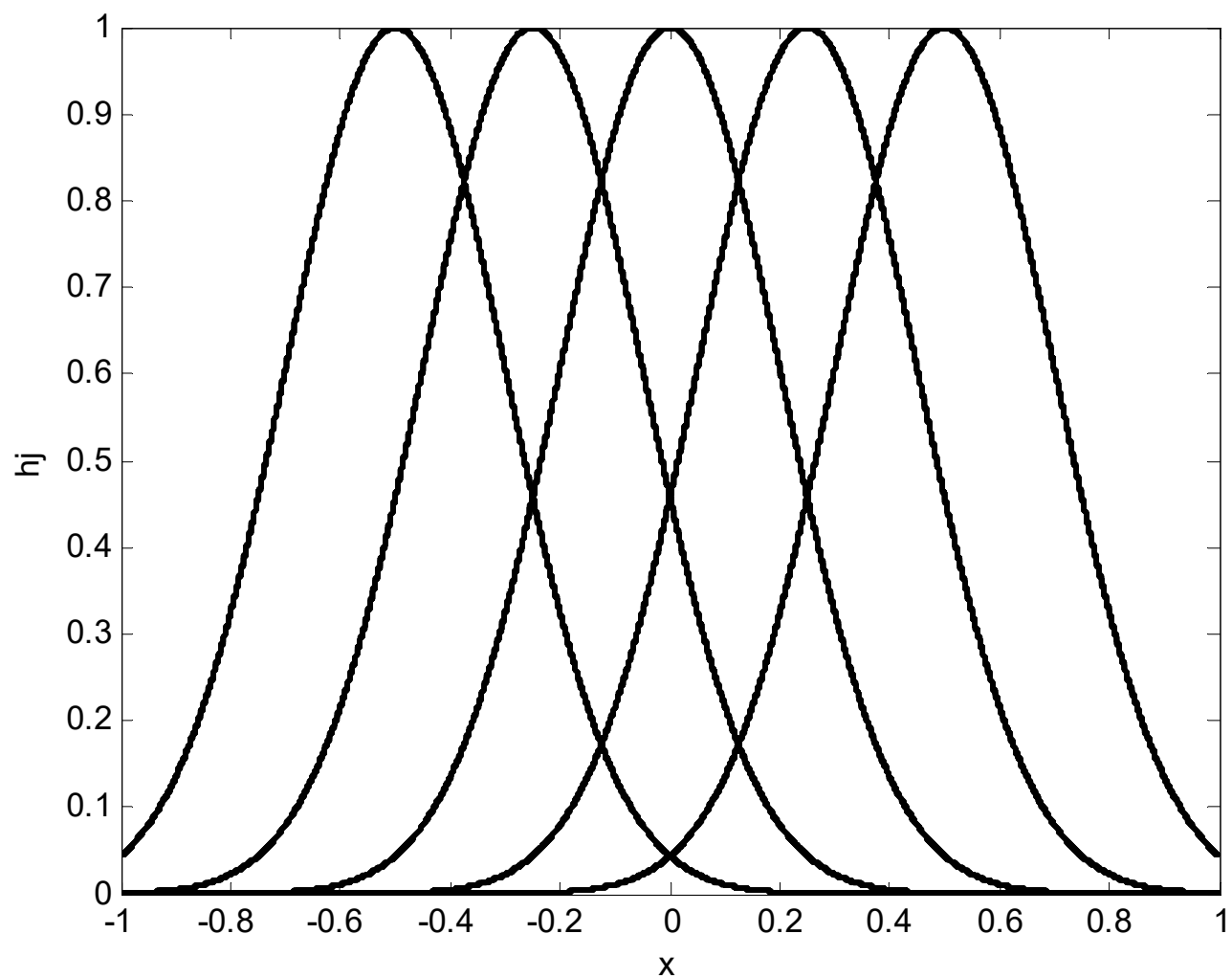


图6-24 RBF网络隐层的输出

6.3 典型神经网络模型——RBF神经网络

例6-6：仿真实例

采用RBF网络对如下模型进行逼近

$$G(s) = \frac{133}{s^2 + 25s}$$

网络结构为2-5-1，取 $x(1)=u(t)$ ， $x(2)=y(t)$ ， $\alpha = 0.05$
 $\eta = 0.5$ 网络的初始权值取0至1之间的随机值。考虑到网络的第一个输入范围为[0,1]，离线测试可得第二个输入范围为[0,10]，取高斯基函数的参数取值为

$$\mathbf{c} = \begin{bmatrix} -1 & -0.5 & 0 & 0.5 & 1 \\ -10 & -5 & 0 & 5 & 10 \end{bmatrix}^T \quad \sigma_i = 1.5 \quad i = 1, 2, 3, 4, 5$$

网络的第一个输入为 $u(t)=\sin(t)$ ，仿真中，只调节权值 w ，取固定的 c_i 和 σ 。仿真程序为chap6_6sim.mdl，chap6_6rbf.m，chap6_6plant.m, chap6_5plot。

6.3 典型神经网络模型Ⅰ——RBF神经网络

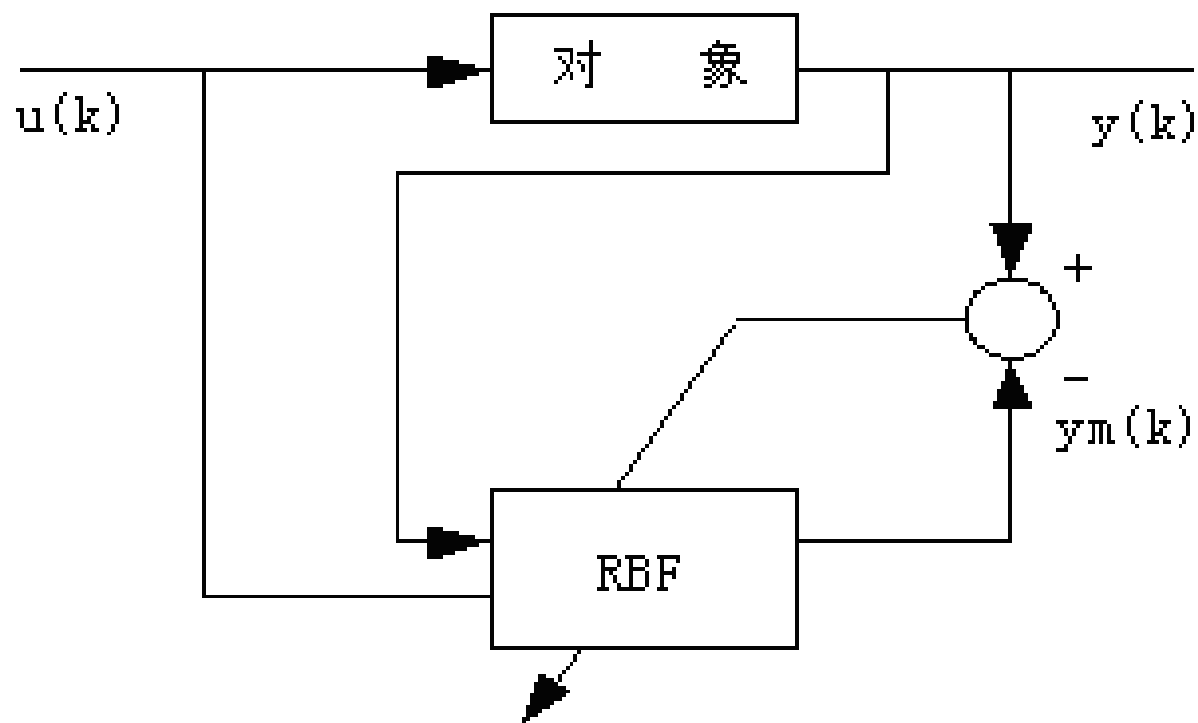


图6-25 RBF神经网络逼近

6.3 典型神经网络模型Ⅰ——RBF神经网络

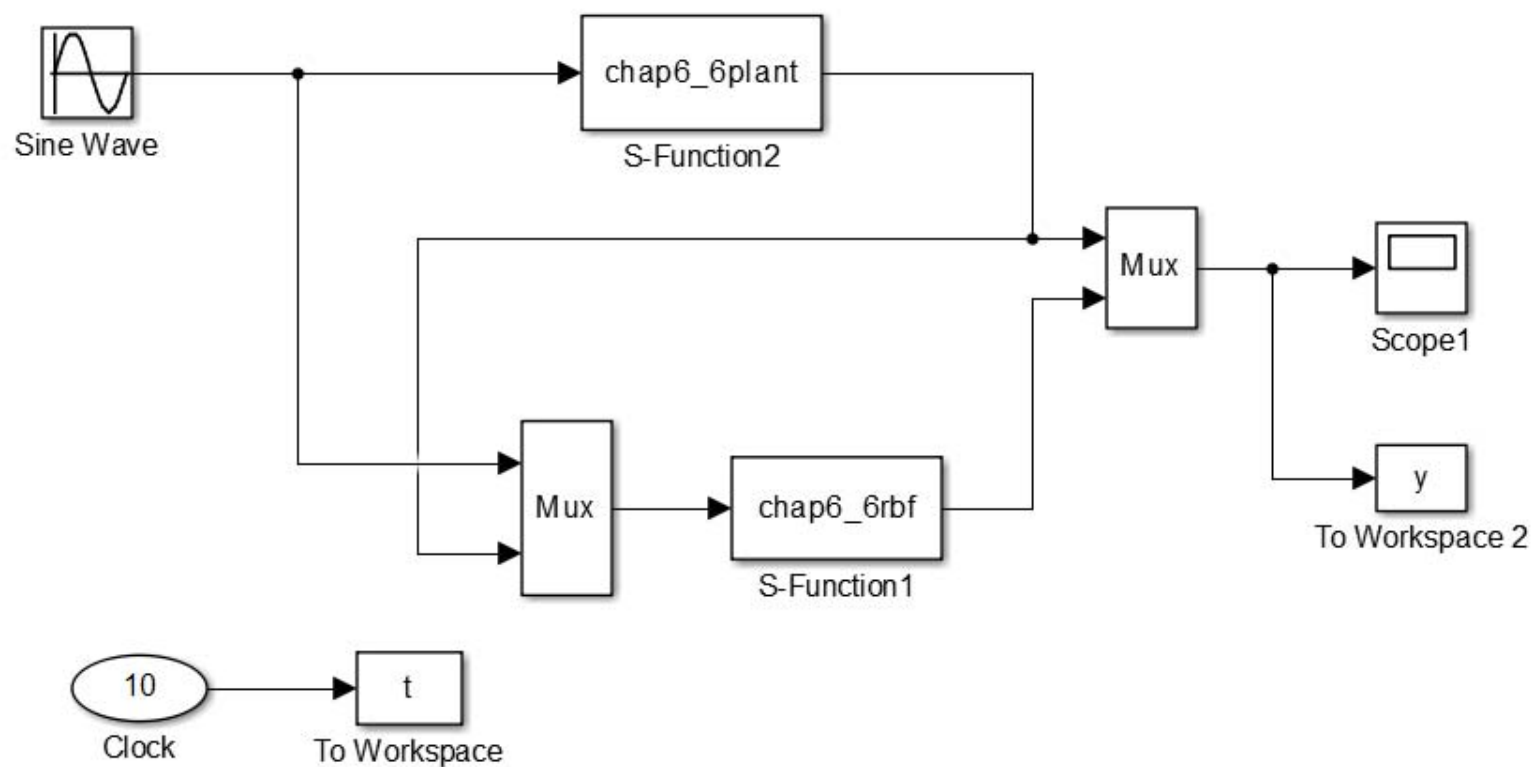


图6-26 chap6_6sim.mdl

6.3 典型神经网络模型——RBF神经网络

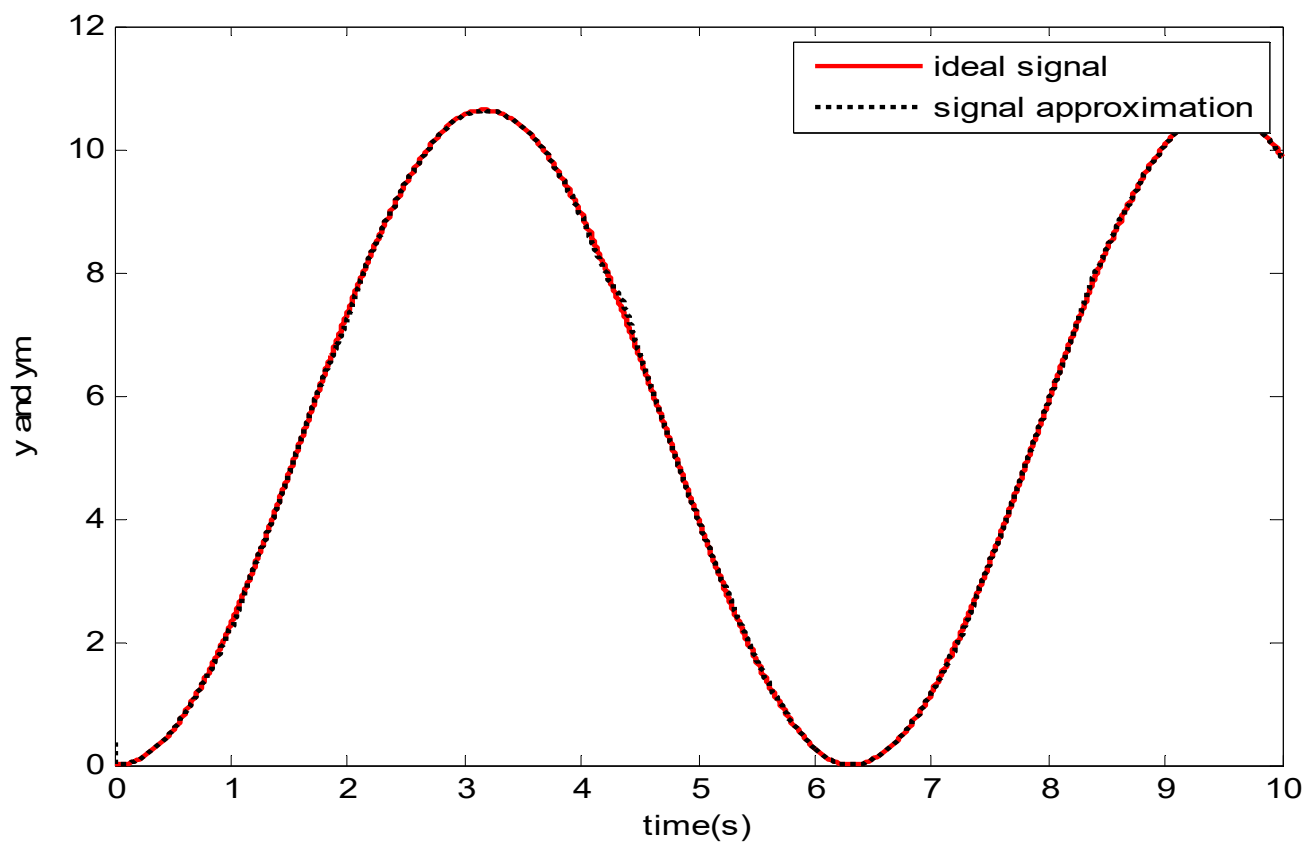


图6-27 基于权值调节的RBF网络逼近

6.3 典型神经网络模型V——RBF神经网络

在RBF网络设计中，需要注意的是将 c_i 和 σ_i 值设计在网络输入有效的映射范围内，否则高斯基函数将不能保证实现有效的映射，导致RBF网络失效。如果将 c_i 和 σ_i 的初始值设计在有效的映射范围内，则只调节网络的权值便可实现RBF网络的有效学习。

6.3 典型神经网络模型V——RBF神经网络

5. 高斯基函数的参数对RBF网络逼近的影响

c_i 和 σ_i 的设计原则如下：

(1) σ_i 为隐含层第 i 个神经元高斯基函数的宽度。 σ_i 值越大，表示高斯基函数越宽。高斯基函数宽度是影响网络映射范围的重要因素，高斯基函数越宽，网络对输入的映射能力越大，否则，网络对输入的映射能力越小。一般将 σ_i 值设计为适中的值。

(2) c_i 为隐含层第 i 个神经元高斯基函数中心点的坐标向量。 c_i 值离输入越近，高斯函数对输入越敏感，否则，高斯函数对输入越不敏感；

$$u_i = \exp\left[-\frac{(x - c_i)^T (x - c_i)}{2\sigma_i^2}\right], i = 1, 2, \dots, q$$

6.3 典型神经网络模型——RBF神经网络

(3) 中心点坐标向量 c_i 应使高斯基函数在有效的输入映射范围内。例如，RBF网络输入为 $[-3,+3]$ ，则 c_i 为 $[-3,+3]$ 。

仿真中，应根据网络输入值的范围来设计 c_i 和 σ_i ，从而保证有效的高斯基函数映射，如图6-28为5个高斯基函数。仿真程序为chap6_7.m。

6.3 典型神经网络模型Ⅰ——RBF神经网络

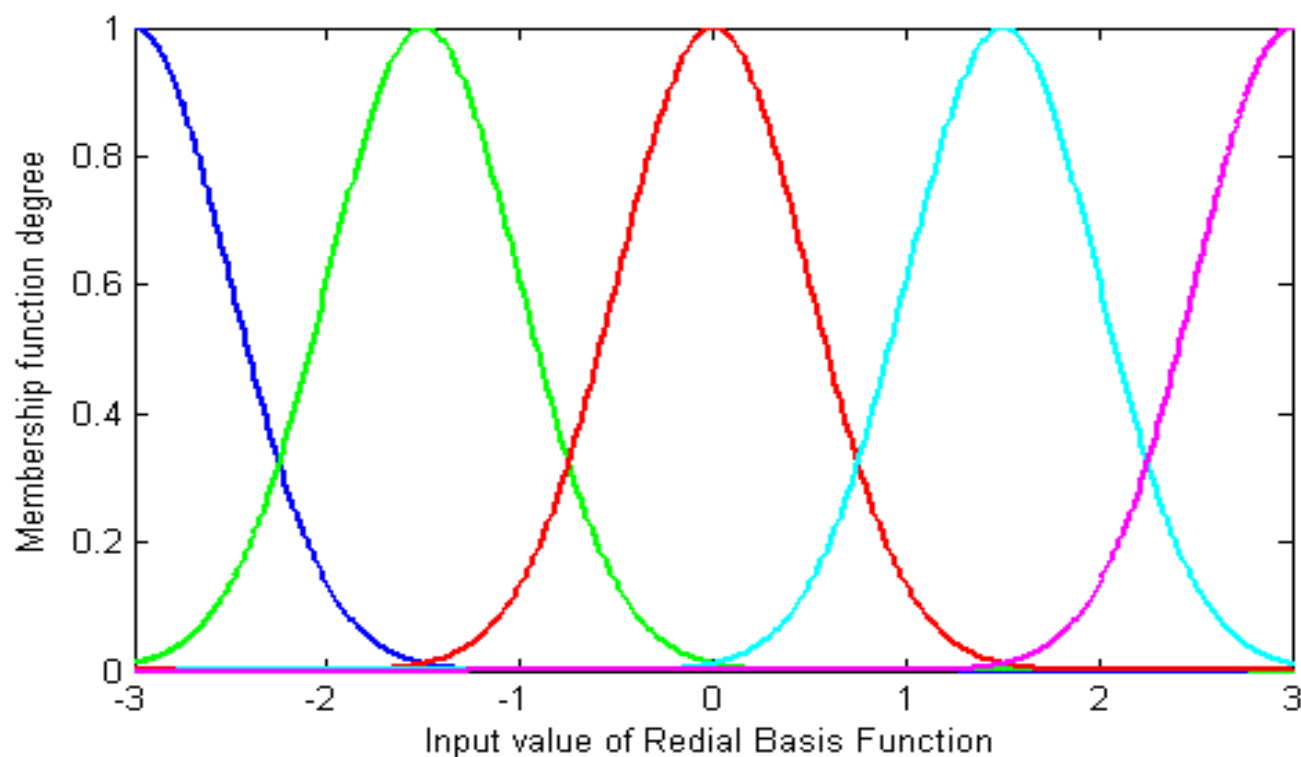


图6-28 5个高斯隶属函数

6.3 典型神经网络模型V——RBF神经网络

采用RBF网络对如下离散模型进行逼近(参考程序chap6_8.m)

$$y(k) = u(k)^3 + \frac{y(k-1)}{1 + y(k-1)^2}$$

仿真中，取RBF网络输入为 $0.5 \sin(2\pi t)$ ，网络结构为2-5-1，通过改变高斯基函数 c_i 和 σ_i 值，可分析 c_i 和 σ_i 对RBF网络逼近性能的影响，具体说明如下：

- (1)合适的 σ_i 和 c_i 值对RBF网络逼近的影响($M\sigma=1$, $Mc=1$);
- (2) 不合适的 σ_i 和 合适的 c_i 值对RBF网络逼近的影响($M\sigma=1$, $Mc=2$);

6.3 典型神经网络模型Ⅰ——RBF神经网络

(3) 合适的 σ_i 与不合适的 c_i 值对RBF网络逼近的影响($M\sigma=1$, $Mc=2$) ;

(4) 不合适的 σ_i 和 c_i 值对RBF网络逼近的影响($M\sigma=2$, $Mc=2$) 。

6.3 典型神经网络模型Ⅳ——RBF神经网络

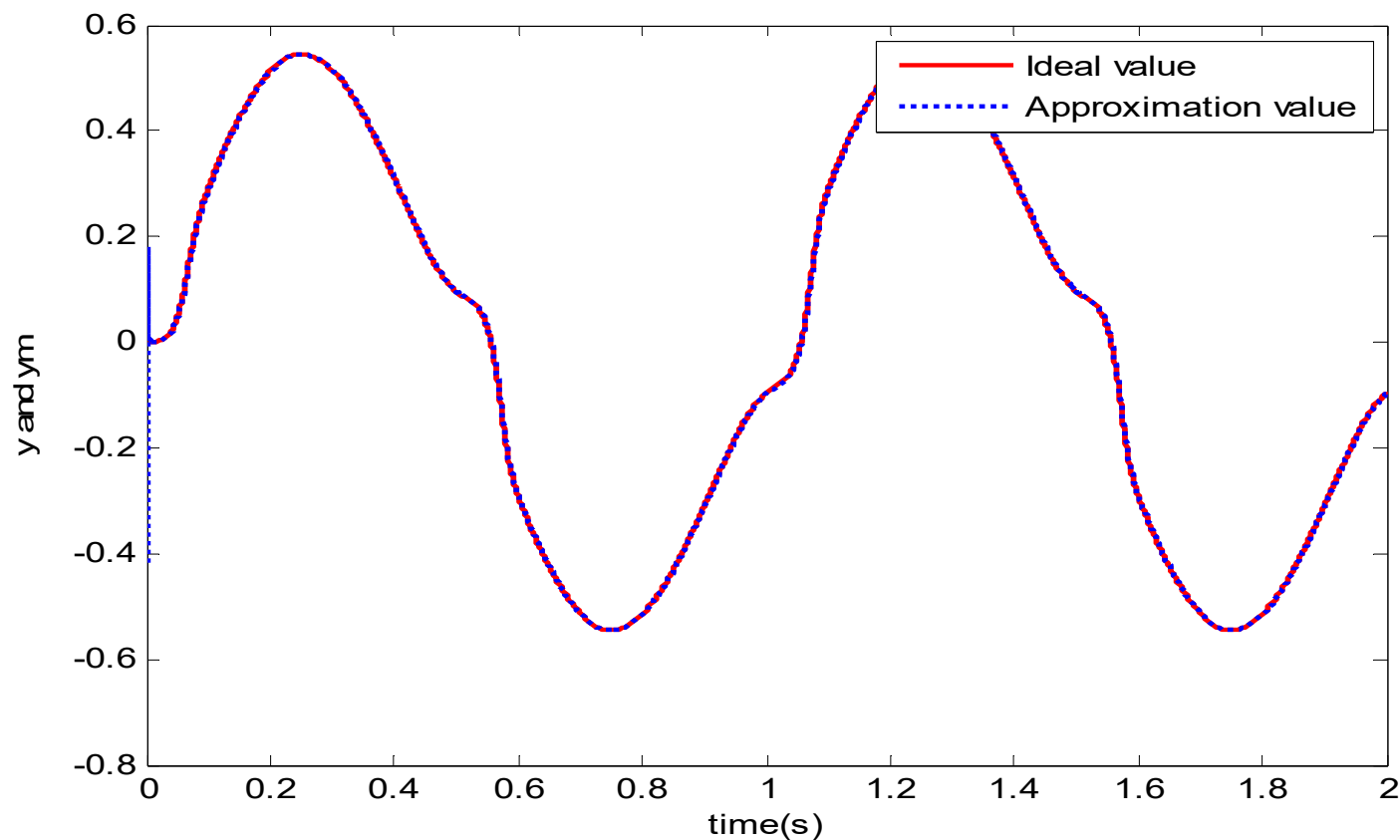


图6-29 合适 σ_i 和 c_i 值的RBF网络逼近RBF ($M\sigma=1$, $Mc=1$)

6.3 典型神经网络模型Ⅰ——RBF神经网络

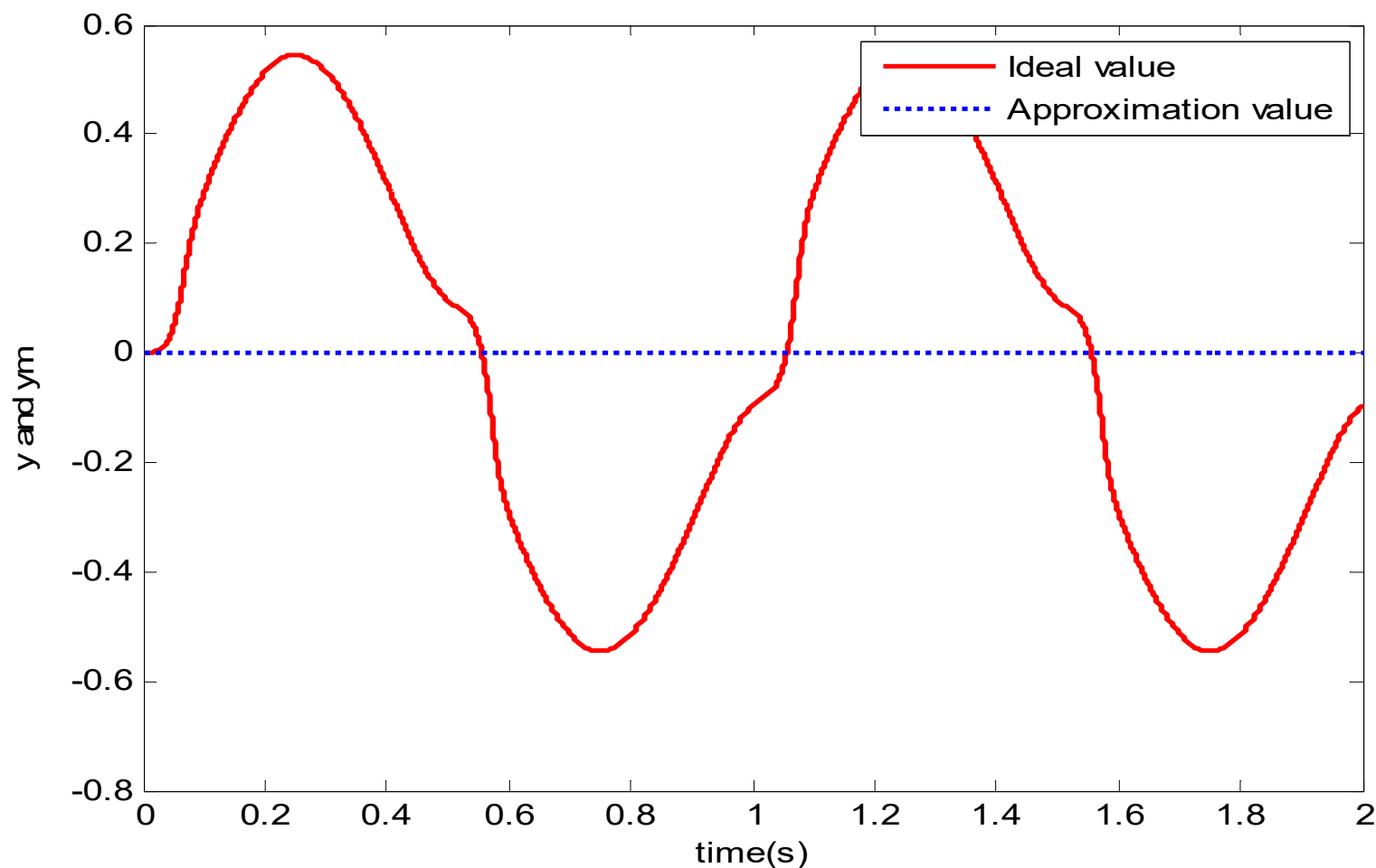


图6-30 不合适 σ_i 与合适 c_i 值的RBF网络逼近 ($M\sigma=2$, $Mc=1$)

6.3 典型神经网络模型Ⅳ——RBF神经网络

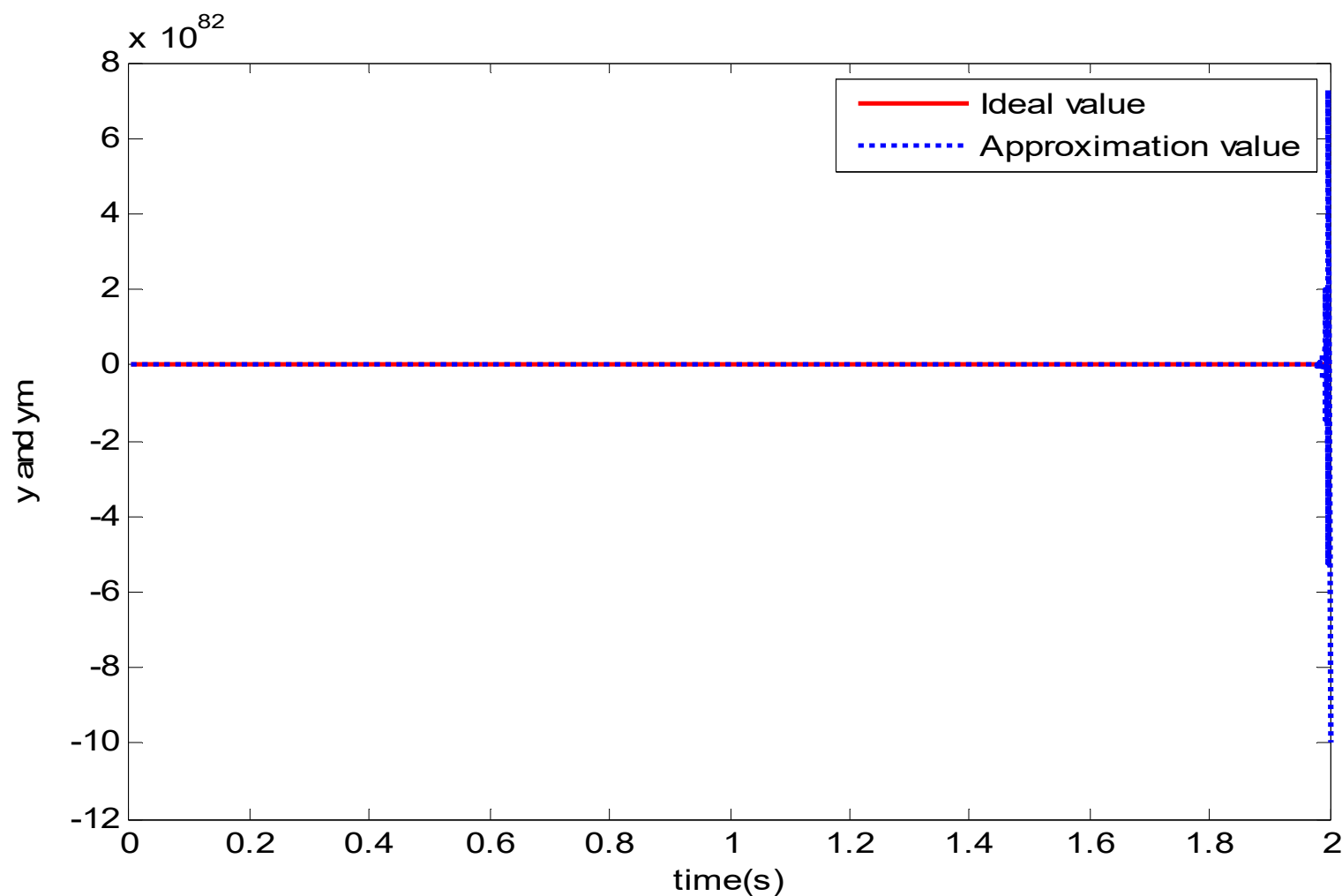


图6-31 合适的 σ_i 与不合适的 c_i 值的RBF网络逼近 ($M\sigma=1$, $Mc=2$)

6.3 典型神经网络模型Ⅰ——RBF神经网络

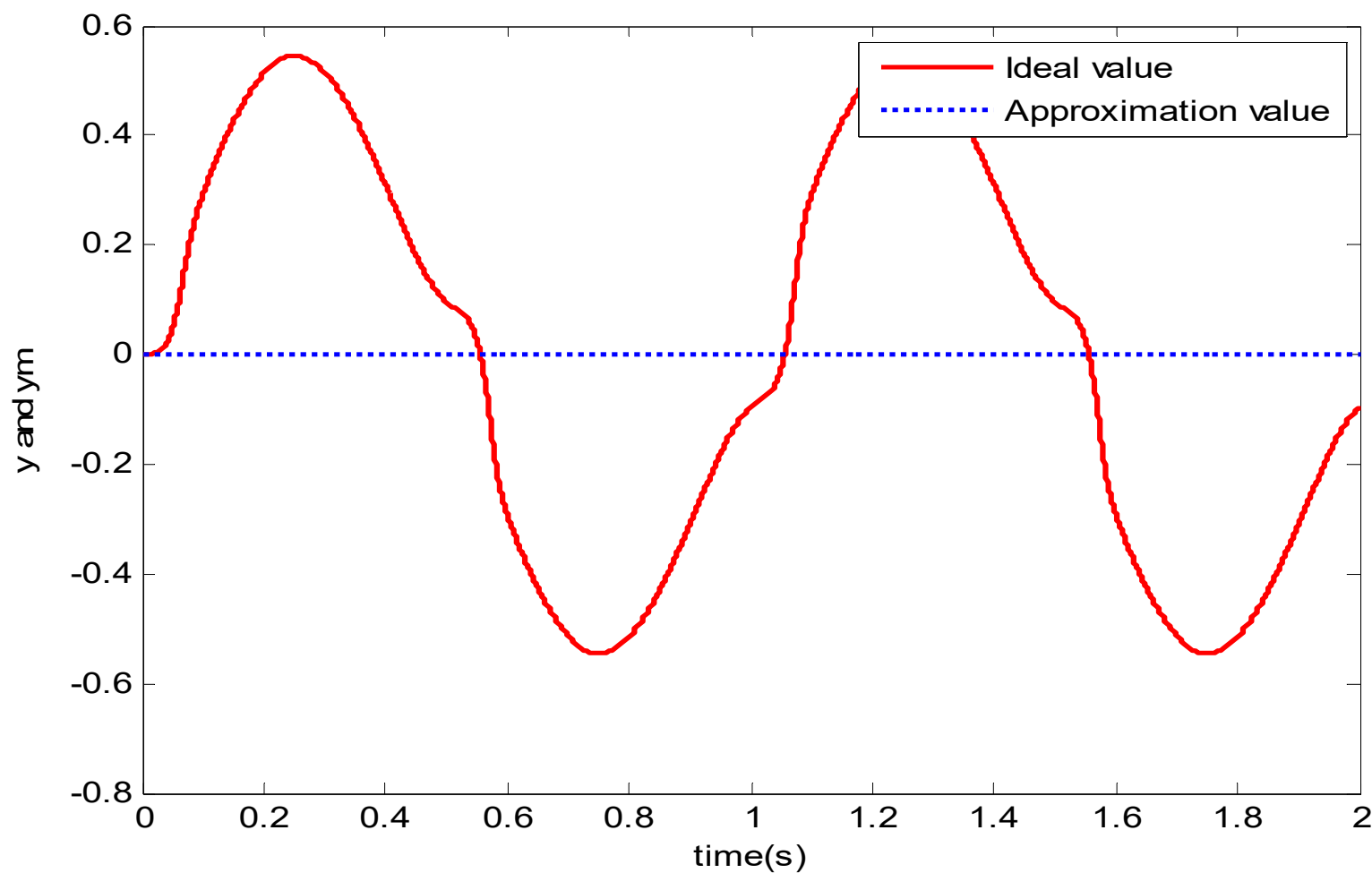


图6-32 不合适 σ_i 和 c_i 值的RBF网络逼近($M\sigma=2$, $Mc=2$)

6.3 典型神经网络模型 VI——竞争学习神经网络

6.3.6 竞争学习神经网络

竞争学习是一种典型无导师学习策略，是指同一神经元层次上各个神经元相互之间进行竞争，竞争胜利的神经元修改与其相联的连接权值，它模拟人类根据过去经验自动适应无法预测的环境变化。

在无监督学习中，只向网络提供一些学习样本，而不提供理想的输出。网络根据输入样本进行自组织，并将其划分到相应的模式类中。因为没有教师信号，所以这类网络通常利用竞争的原则进行。学习时只须给定一个输入模式集作为训练集，网络自行组织训练模式，并将其分成不同类型。

与BP学习相比，这种学习能力进一步拓宽了神经网络在模式识别、分类方面的应用。

6.3 典型神经网络模型 V/——竞争学习神经网络

竞争层

竞争学习网络的核心——竞争层是许多神经网络模型的重要组成部分，例如自组织神经网络（Self organizing NNs）、Kohonen提出的自组织特征映射网络（SOM）、Hecht—Nielson 提出的反传网络（CPN），Grossberg与 Carpenter提出的自适应共振理论（ART）网络模型等均包含竞争层。

6.3 典型神经网络模型 V/——竞争学习神经网络

1. 自组织竞争神经网络 (**Self Organizing NNs**)

2. 自组织特征映射神经网络 (**Self Organizing Feature Map, SOM**)

6.3 典型神经网络模型 V/——竞争学习神经网络

1. 自组织竞争神经网络（**Self organizing NNs**）

自组织竞争人工神经网络的形成是受生物神经系统的启发，之所以称为自组织竞争人工神经网络，是因为它一般是由输入层和竞争层构成的单层网络。自组织竞争神经网络能够识别成组的相似向量，常用于进行模式分类。

6.3 典型神经网络模型 V/——竞争学习神经网络

(1) 自组织竞争神经网络的形成

在生物神经系统中，存在一种“侧抑制”的现象，即一个神经细胞兴奋后，通过它的分支会对周围其他神经细胞产生抑制。这种侧抑制式神经细胞之间出现竞争，虽然开始阶段各个神经细胞都处于程度不同的兴奋状态，由于侧抑制的作用，各细胞之间相互竞争的最终结果是：兴奋作用最强的神经细胞所产生的抑制作用战胜了它周围所有其他细胞的抑制作用而“赢”了，其周围的其他神经细胞则全“输”了。

6.3 典型神经网络模型 VI——竞争学习神经网络

(2) 自组织竞争神经网络的结构

自组织竞争学习网络由输入层和竞争层组成。输入层由接收输入模式的处理单元构成；竞争层的竞争单元争相响应输入模式，胜者表示输入模式的所属类别。输入层单元到竞争层单元的连接为全互连方式，连接权是可调节的。对于给定的一个输入模式，**只调节获胜单元的连接权。**

6.3 典型神经网络模型 VI——竞争学习神经网络

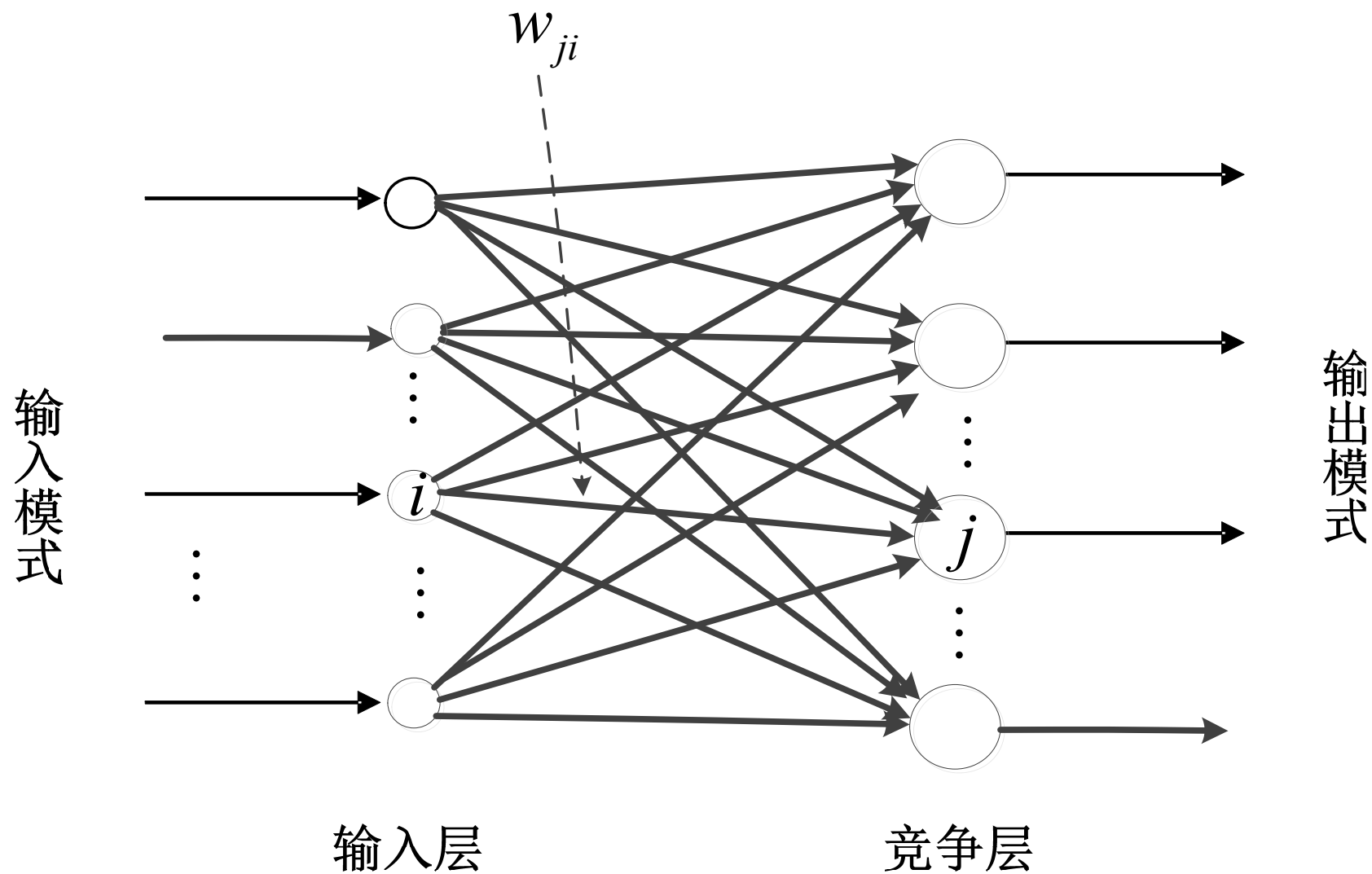


图6-33 自组织竞争学习网络

6.3 典型神经网络模型 VI——竞争学习神经网络

(3) 竞争学习机理

自组织竞争学习网络中，每个竞争单元和输入层单元都有一个连接权，其取值在0与1之间。为了简化网络，假设任意一个给定竞争单元的权值和总是为1，即

$$\sum_i w_{ji} \approx 1 \quad (6-40)$$

网络学习时，初始权值一般满足上式的一组小的随机数；输入模式是二进制的0 / 1 向量。

6.3 典型神经网络模型 V/——竞争学习神经网络

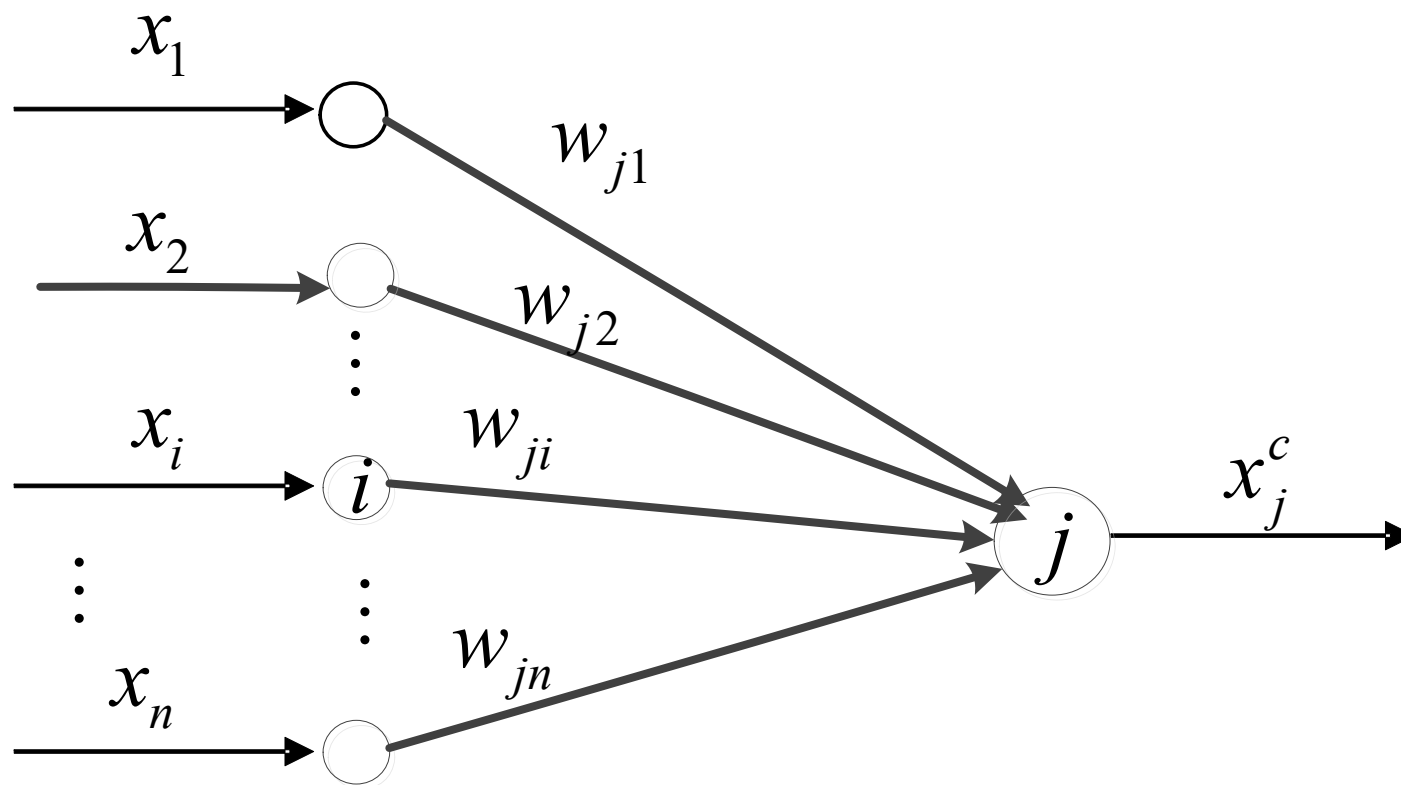


图 6-34 竞争层处理单元

6.3 典型神经网络模型 V/——竞争学习神经网络

首先计算每个单元输入的加权和，然后在竞争中产生输出。对于第 j 个竞争单元，其输出总和为

$$S_j = \sum_i w_{ji} x_i \quad (6-41)$$

当竞争层所有单元的输入总和计算完毕，便开始竞争。根据“胜者为王，败者为寇”的道理，竞争层中具有最高输入总和的单元被确定为胜者，其输出状态为1，其他各单元状态为0，即

$$x_j^c = \begin{cases} 1, s_j > \max(S_k, k \neq j, k = 1, 2, \dots, q) \\ 0, else \end{cases} \quad (6-42)$$

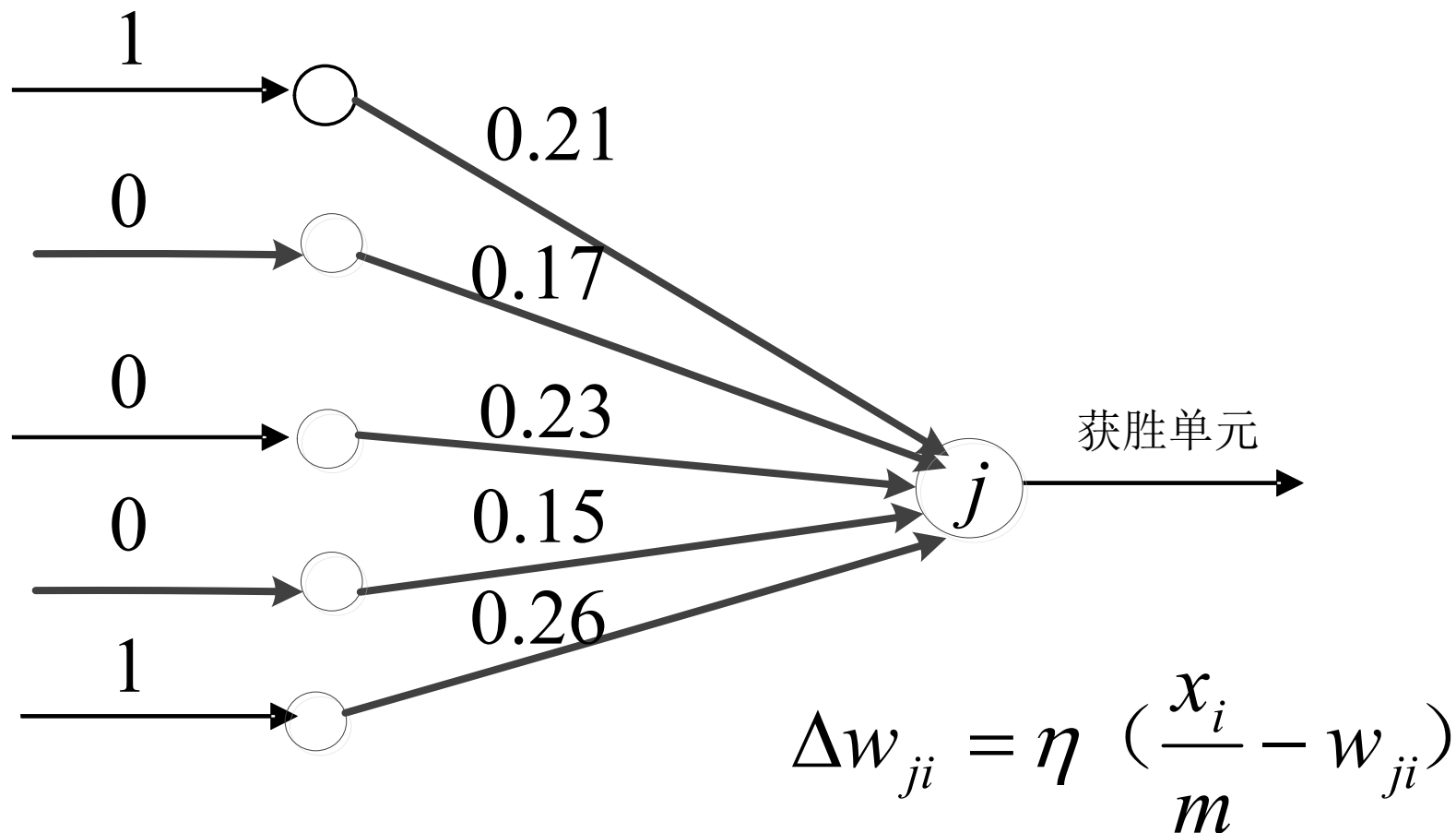
式中， x_j^c 表示竞争层第 j 个单元输出状态。特别地，当 $S_j = S_k (k \neq j)$ 时，通常取位于左边的处理单元状态为1。

6.3 典型神经网络模型 VI——竞争学习神经网络

对于某一输入模式，当竞争胜者单元被确定后，更新权值，也只有获胜单元的权值被增加一个量，使得再次遇到该输入模式时，该单元有更大的输入总和。权值更新规则表示为

$$\Delta w_{ji} = \eta \left(\frac{x_i}{m} - w_{ji} \right) \quad (6-43)$$

式中， η 表示学习因子， $0 < \eta < 1$ ，反映权值更新速率，一般取值为 $0.01 \sim 0.3$ ； m 表示输入层状态为 1 的单元个数。



输入模式 $[1 \ 0 \ 0 \ 0 \ 1]$

初始权值 $[0.21 \ 0.17 \ 0.23 \ 0.15 \ 0.26]$

$\eta = 0.1$

更新权值 $[0.239 \ 0.153 \ 0.207 \ 0.135 \ 0.284]$

图6-35 竞争学习中修正权值

6.3 典型神经网络模型 V/——竞争学习神经网络

按照权值修正算式(6-43)式，获胜单元的一些权值减小一个量，另一些则增大一个量，其结果获胜单元的权值之和仍然满足为1的约束。

$$\Delta w_{ji} = \eta \left(\frac{x_i}{m} - w_{ji} \right) \quad (6-43)$$

$$\sum_{i=1}^n \Delta w_{ji} = \sum_{i=1}^n \eta \left(\frac{x_i}{m} - w_{ji} \right) = \eta \left(\frac{\sum_{i=1}^n x_i}{m} - \sum_{i=1}^n w_{ji} \right) = 0$$

6.3 典型神经网络模型 V/——竞争学习神经网络

2. 自组织特征映射网络（SOM）

自组织特征映射网络（Self-Organizing feature Map, SOM 网络）是由芬兰赫尔辛基大学神经网络专家Kohonen教授在1981年提出的，他认为一个神经网络接受外界输入模式时，将会分为不同的区域，各区域对输入模式具有不同的响应特征，同时这一过程是自动完成的。

各神经元的连接权值具有一定的分布。最邻近的神经元互相刺激，而较远一些的则具有较弱的刺激作用。这种网络模拟大脑神经系统自组织特征映射的功能，它是一种竞争式学习网络，在学习中能无监督地进行自组织学习。

6.3 典型神经网络模型 VI——竞争学习神经网络

(1) 自组织特征映射网络的结构

自组织特征映射网络（**SOM**）的基本网络结构也是由输入层和竞争层组成，其竞争层按二维网络阵列方式组织，而且权值更新的策略也不同。

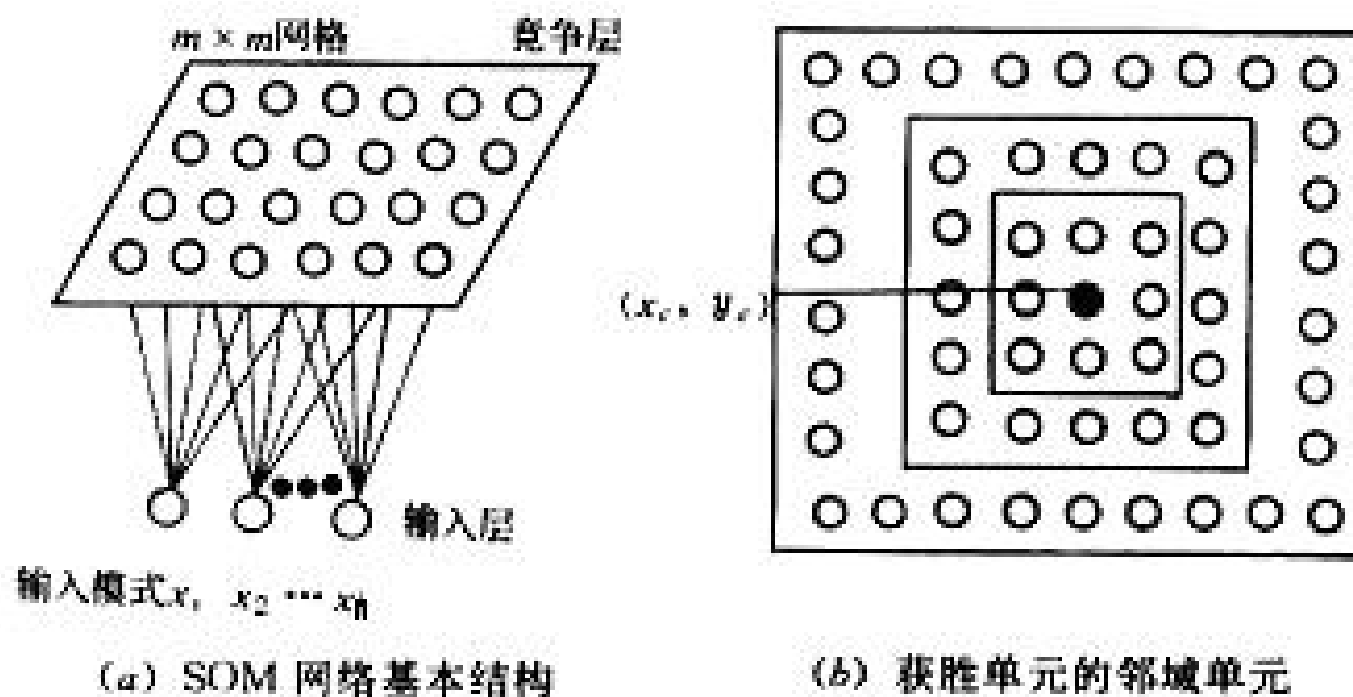


图 6-36 SOM网络结构

6.3 典型神经网络模型 V/——竞争学习神经网络

两种连接权:

- 神经元对外部输入反应的连接权值;
- 另一种是神经元之间的连接权值, 它的大小控制着神经元之间的交互作用的大小。

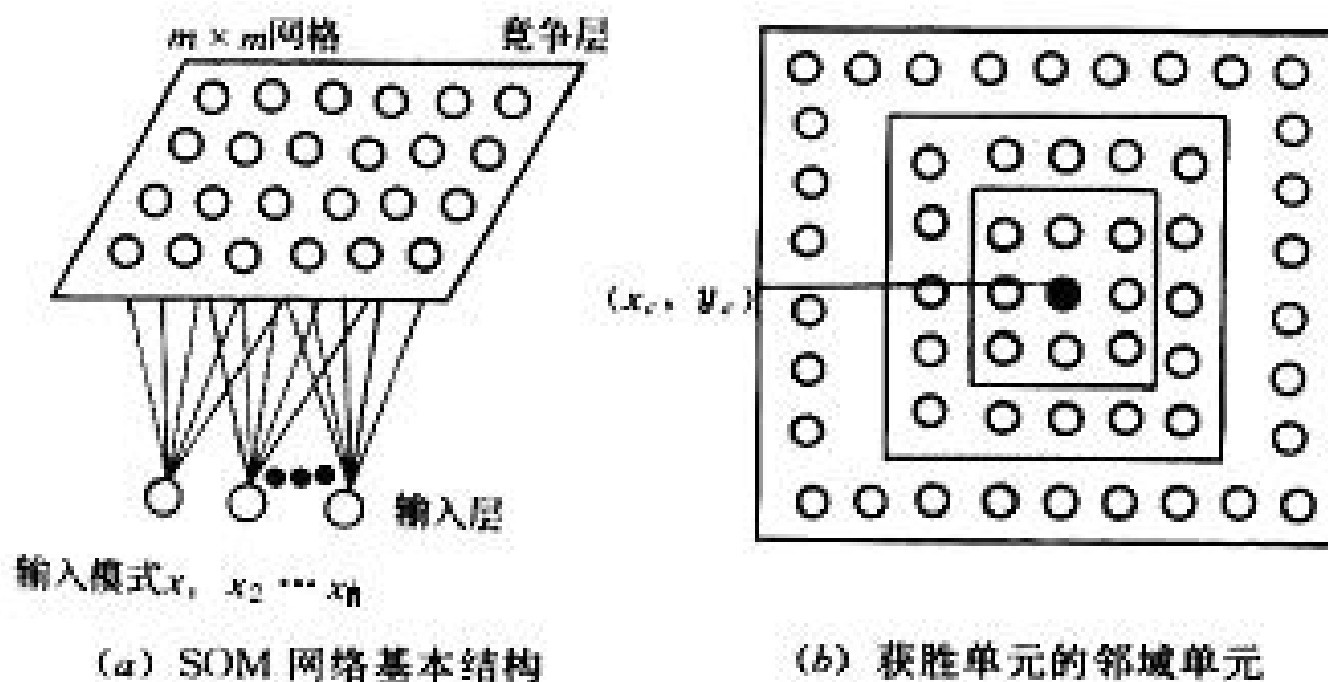


图 6-36 SOM网络结构

6.3 典型神经网络模型 V/——竞争学习神经网络

(2) 自组织特征映射网络的学习机理

自组织特征映射算法是一种无教师示教的聚类方法，它能够将任意输入模式在输出层映射成一维或二维离散图形，并保持其拓扑结构不变。即在无教师示教的情况下，通过对输入模式的自组织学习，在竞争层将分类结果表示出来。

此外，网络通过对输入模式的反复学习，可以使连接权矢量空间分布密度与输入模式的概率分布趋于一致，即连接权矢量空间分布能反映输入模式的统计特征。

6.3 典型神经网络模型 V/——竞争学习神经网络

自组织映射网络的竞争学习过程:

第一, 对于给定输入模式, 确定竞争层获胜单元;

第二, 按照学习规则修正获胜单元及其邻域单元的连接权值;

第三, 逐渐减少邻域及学习过程中权值的变化量。

竞争获胜单元及其邻域单元权值更新规则可表达为:

$$w_{ji}(k+1) = w_{ji}(k) + \Delta w_{ji}$$
$$\Delta w_{ji} = \begin{cases} \eta_c (x_i - w_{ji}), & \text{对获胜单元} \\ \eta_N (x_i - w_{ji}), & \text{获胜邻域单元} \end{cases} \quad (6-44)$$

式中 η_c, η_N 分别表示获胜单元及其邻域单元的权值学习因子, 取值在(0,1)区间且 $\eta_c > \eta_N$ 。 x_i, w_{ji} 表示竞争获胜单元的输入及连接权。

6.3 典型神经网络模型 V/——竞争学习神经网络

在学习过程中，权值学习因子 η_c 及竞争层获胜单元的邻域 N_c 的大小是逐渐减小的。学习因子 η_N 的初值一般取得比较大，随着学习过程的迭代 η_N 值逐渐减少，常用的一个调度策略表达式为

$$\eta_N = \eta_{N_0} \left(1 - \frac{t}{T}\right) \quad (6-45)$$

式中 η_{N_0} 表示 η_N 初始值，通常在 0.2~0.5 之间取值； t 表示迭代次数； T 表示整个迭代设定次数。

6.3 典型神经网络模型 V/——竞争学习神经网络

邻域的宽度也是随着学习过程的迭代而减少，因此，由竞争获胜单元到邻域单元的距离 d 相应减少。假设 d 的初值记 d_0 ，一般取值为 $1/2$ 或 $1/3$ 竞争层宽度， d 的减小策略可表达为

$$d = d_0 \left(1 - \frac{t}{T}\right) \quad (6-46)$$

自组织映射网络的学习使竞争获胜单元的邻域单元受到激励，邻域之外较远的单元受到抑制。

6.3 典型神经网络模型 V/——竞争学习神经网络

3. 竞争学习网络的特征

- 1) **竞争层**：它采用无导师学习策略，每个竞争单元都相当于一个特征分类器。
- 2) **模式分类**：在竞争学习方案中，网络通过极小化簇内模式距离及极大化不同簇间的距离（海明距离）实现模式分类。
- 3) 竞争网络能将给定的模式分成几类预先是不知道的，只有在学习以后才知道，这种分类能力在许多场合是很有用的

6.3 典型神经网络模型 V/——竞争学习神经网络

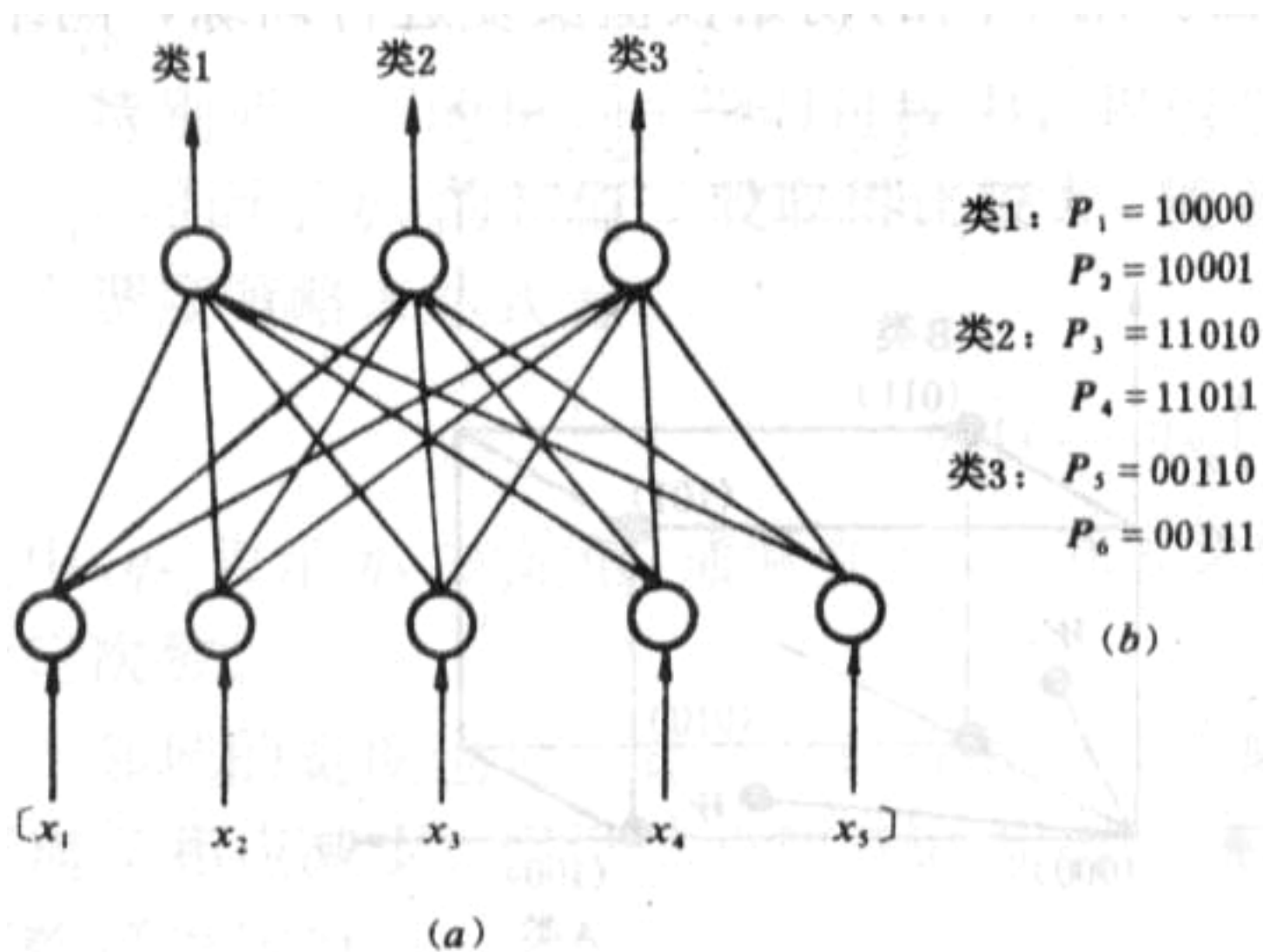


图 6-37 竞争学习聚类分析

6.3 典型神经网络模型 V/——竞争学习神经网络

例6-7 给定一个竞争网络，如图 6-38，要求通过训练将输入模式集划分为两大类。设输入模式记为

$$(101)=P_1, (100)=P_2, (010)=P_3, (011)=P_4$$

观察每两个模式之间的海明距离——两个二进制输入模式不同状态的个数，得到以下关系矩阵。

$$P = (p_{ij}) = \begin{bmatrix} 0 & 1 & 3 & 2 \\ 1 & 0 & 2 & 3 \\ 3 & 2 & 0 & 1 \\ 2 & 3 & 1 & 0 \end{bmatrix}$$

6.3 典型神经网络模型 V/——竞争学习神经网络

输入模式自然分成两类，该网络训练完后得到如下两类

A类: $P1 = (101)$, $P2 = (100)$;

B类: $P3 = (010)$, $P4 = (011)$

每一类包含两个输入模式，同一类模式海明距离为1，不同类模式的海明距离为 2 或 3。网络的分类原则来源于输入模式的固有特征。用不同的初始权值反复进行训练，网络便能自组织学习，完成正确的模式分组。

6.3 典型神经网络模型 V/——竞争学习神经网络

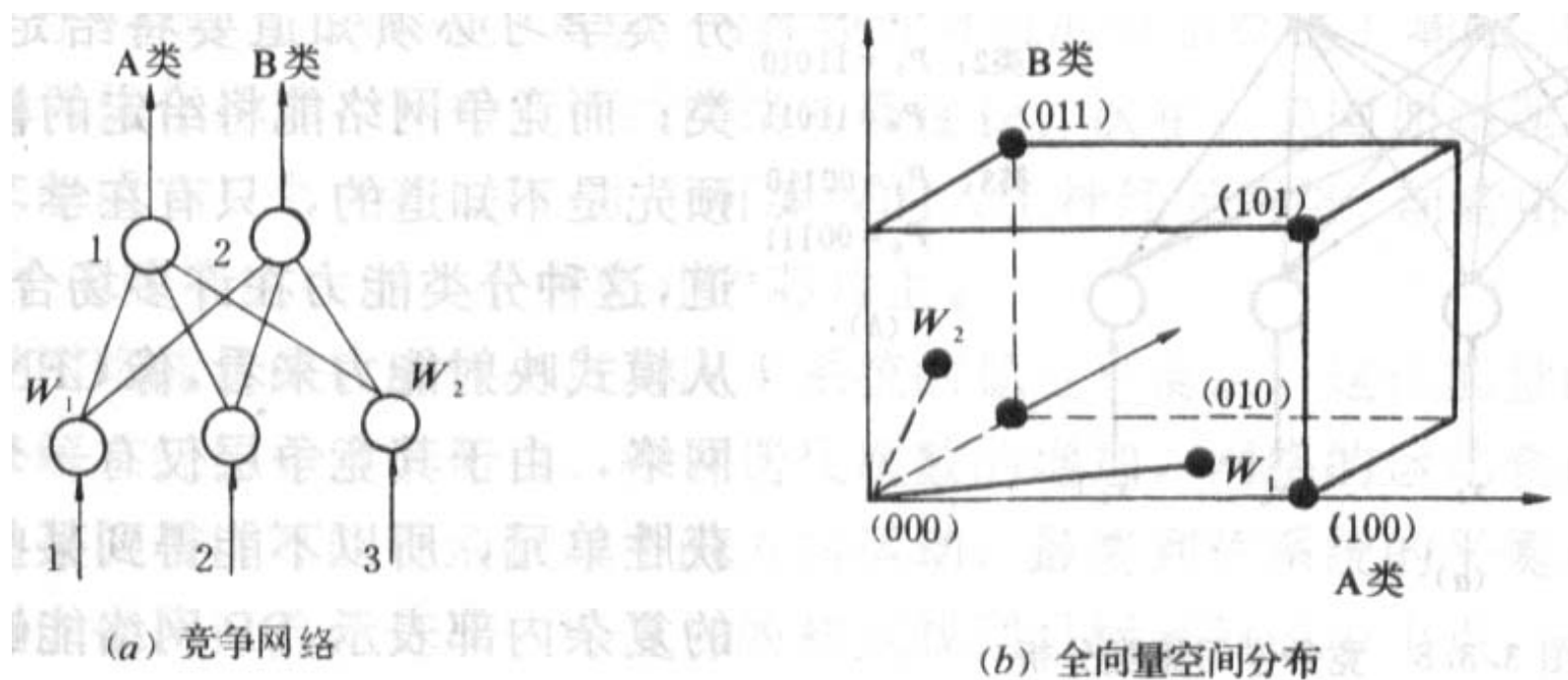


图 6-38 竞争分类

6.3 典型神经网络模型 V//——Hopfield神经网络

6.3.7 Hopfield神经网络

Hopfield神经网络是1982年美国物理学家Hopfield首先提出来的。前面讨论的BP前向神经网络的特点是网络中没有反馈连接，不考虑输出与输入间在时间上的滞后影响，其输出与输入间仅是一种映射关系。Hopfield神经网络采用反馈连接，考虑输出与输入间在时间上的传输延迟，所以所表示的是一个动态过程，需要用差分方程或微分方程来描述，因而Hopfield网络是一种由非线性元件构成的反馈系统，其稳定状态的分析比BP网络要复杂得多。

6.3 典型神经网络模型 V//——Hopfield神经网络

加权系数的调整

BP前向网络采用的是一种有监督的误差均方修正方法，学习计算过程较长，收敛速度较慢。

Hopfield网络则是采用有监督的Hebb学习规则（用输入模式作为目标模式）。计算的收敛速度很快。该网络主要用于联想记忆和优化计算。

6.3 典型神经网络模型 V//——Hopfield神经网络

在Hopfield网络中，每一个神经元都和所有其他神经元相连接，所以又称为全互连网络。研究表明，当连接加权系数矩阵无自连接并具有对称性质，即

$$w_{ii} = 0, w_{ij} = w_{ji} \quad (i, j=1, 2, \dots, n)$$

时，算法是收敛的。

根据网络的输出是离散量或是连续量，Hopfield网络分为离散和连续的两种。

6.3 典型神经网络模型 V//——Hopfield神经网络

1. 离散型Hopfield网络

1) 网络的结构和工作方式

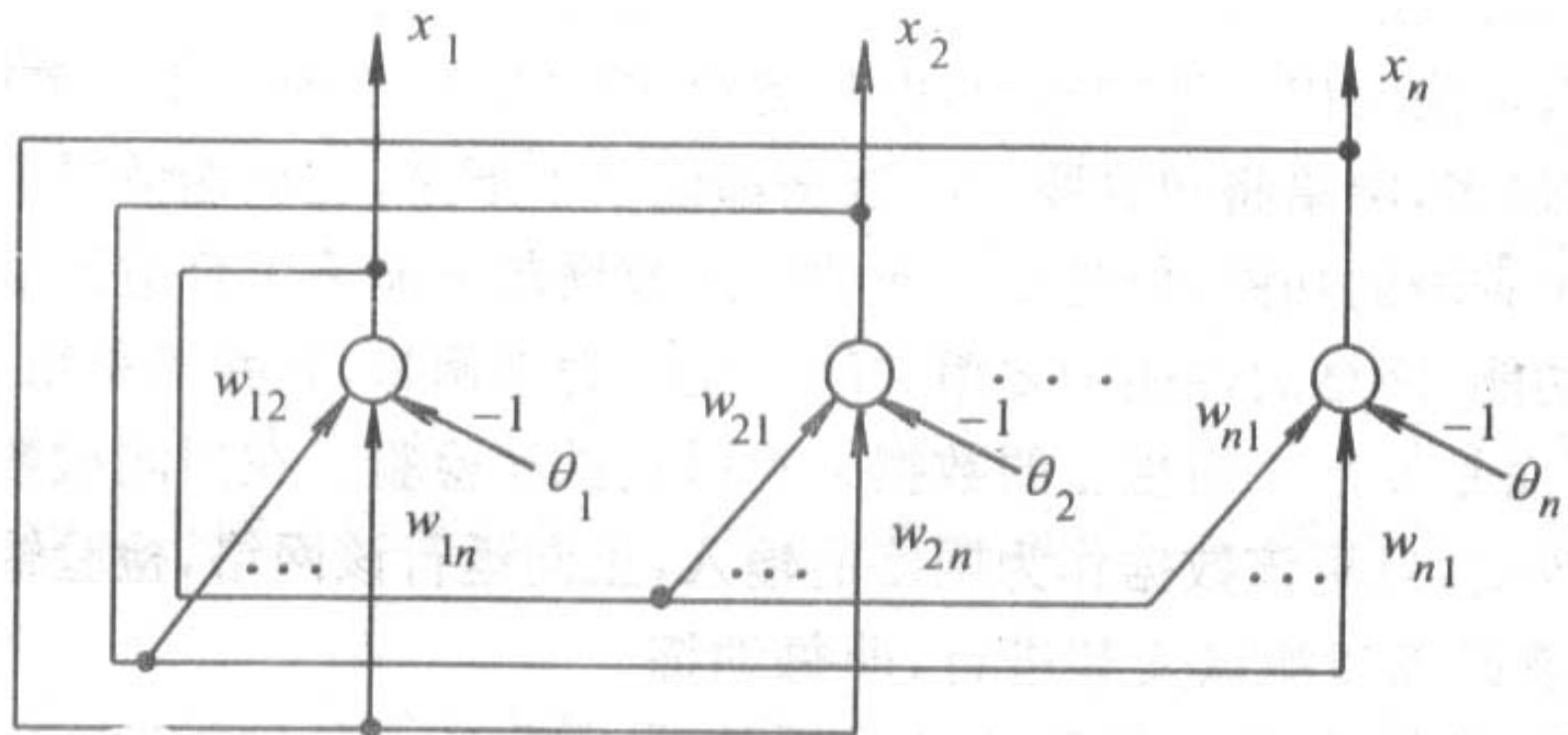


图6-39 离散型Hopfield网络

6.3 典型神经网络模型 V//——Hopfield神经网络

对于每一个神经元节点，其工作方式为

$$\begin{cases} s_i(k) = \sum_{j=1}^n w_{ij} x_j(k) - \theta_i \\ x_i(k+1) = f(s_i(k)) \end{cases} \quad (6-47)$$

其中， $w_{ii} = 0$ ， θ_i 为阈值， $f(\cdot)$ 是变换函数。对于离散Hopfield网络， $f(\cdot)$ 通常取为二值函数，即

$$f(s) = \begin{cases} 1, s \geq 0 \\ -1, s < 0 \end{cases} \quad f(s) = \begin{cases} 1, s \geq 0 \\ 0, s < 0 \end{cases} \quad (6-48)$$

6.3 典型神经网络模型 V//——Hopfield神经网络

整个网络有如下两种工作方式：

(1) 异步方式（串行工作方式）

在某一时刻只有一个神经元按照式(6-47)改变状态，而其余神经元的输出保持不变，这一变化的神经元可以按照随机方式或预定的顺序来选择。例如，若达到的神经元为第 i 个，则有

$$\begin{cases} x_i(k+1) = f\left(\sum_{j=1}^n w_{ij}x_j(k) - \theta_i\right) \\ x_j(k+1) = x_j(k), \quad j \neq i \end{cases} \quad (6-49)$$

其调整次序可以随机选定，也可按规定的次序进行。若以某种确定性的次序来选择神经元，使其按(6-49)变化，则称为顺序更新，若按照某种预先设定的概率来选择神经元，则称为随机更新。

6.3 典型神经网络模型 V//——Hopfield神经网络

(2) 同步方式(并行工作方式)

在某一时刻有 n_1 ($1 < n_1 \leq n$)个神经元按照式(6-47)改变状态, 而其余神经元的输出保持不变。当 $n_1 = n$ 时, 称为全并行方式, 此时所有神经元都照式(6-47)改变状态, 即

$$x_i(k+1) = f\left(\sum_{j=1}^n w_{ij}x_j(k) - \theta_i\right), i = 1, 2, \dots, n \quad (6-50)$$

6.3 典型神经网络模型 V//——Hopfield神经网络

上述同步计算方式也可写成如下的矩阵形式

$$x(k+1) = f(Wx(k) - \theta)$$

其中 $x(k) = [x_1(k), x_2(k), \dots, x_n(k)]^T$ 和 $\theta = [\theta_1, \theta_2, \dots, \theta_n]^T$ 是向量， W 是由 w_{ij} 所组成的 $n \times n$ 矩阵； $f(s)$ 是向量函数，它表示 $f(s) = [f(s_1), f(s_2), \dots, f(s_n)]^T$

该网络是动态的反馈网络，其输入是网络的状态初值

$$x(0) = [x_1(0), x_2(0), \dots, x_n(0)]^T$$

输出是网络的稳定状态 $\lim_{k \rightarrow \infty} x(k)$ 。

6.3 典型神经网络模型 V//——Hopfield神经网络

2) 稳定性和吸引子

(1) 稳定性

若网络的状态 x 满足 $x = f(Wx - \theta)$ 则称为网络的稳定点或吸引子。

定理 6.1 对于离散Hopfield网络，若按异步方式调整状态，且连接权矩阵 W 为对称阵，即 $w_{ij}=w_{ji}$ ，则对于任意初态，网络都最终收敛到一个吸引子。

证明：定义网络的能量函数为

$$E(k) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N w_{ij} x_j(k) x_i(k) + \sum_{i=1}^N \theta_i x_i(k) = -\frac{1}{2} x^T(k) W x(k) + x^T(k) \theta$$

由于神经元节点的状态只能取1和-1（或1和0）两种状态，因此上述定义的能量函数 $E(k)$ 是有界的。

6.3 典型神经网络模型 V//——Hopfield神经网络

令 $\Delta E(k) = E(k+1) - E(k), \Delta x(k) = x(k+1) - x(k)$
则

$$\begin{aligned}\Delta E(k) &= E(k+1) - E(k) \\&= -\frac{1}{2}[x(k) + \Delta x(k)]^T W[x(k) + \Delta x(k)] + [x(k) + \Delta x(k)]^T \theta \\&\quad - \left[-\frac{1}{2}x^T(k)Wx(k) + x^T(k)\theta \right] \\&= -\Delta x^T(k)Wx(k) - \frac{1}{2}\Delta x^T(k)W\Delta x(k) + \Delta x^T(k)\theta \\&= -\Delta x^T(k)[Wx(k) - \theta] - \frac{1}{2}\Delta x^T(k)W\Delta x(k)\end{aligned}$$

6.3 典型神经网络模型 V//——Hopfield神经网络

由于假定异步工作方式，因此可设第 k 时刻只有第 i 个神经元调整状态，即

$$\Delta x(k) = [0 \cdots 0 \quad \Delta x_i(k) \quad 0 \cdots 0]$$

代入上式则有

$$\Delta E(k) = -\Delta x_i(k) \left[\sum_{j=1}^n w_{ij} x_j(k) - \theta_i \right] - \frac{1}{2} \Delta x_i^2 w_{ii}$$

令 $s_i(k) = \sum_{j=1}^n w_{ij} x_j(k) - \theta_i$ ，则

$$\Delta E(k) = -\Delta x_i(k) \left[s_i(k) + \frac{1}{2} \Delta x_i(k) w_{ii} \right] = -\Delta x_i(k) s_i(k) \quad (w_{ii} = 0)$$

设神经元节点取1和-1两种状态，则

$$x_i(k+1) = f[s_i(k)] = \begin{cases} 1 & s_i(k) \geq 0 \\ -1 & s_i(k) < 0 \end{cases}$$

6.3 典型神经网络模型 V//——Hopfield神经网络

下面考虑 $\Delta x_i(k)$ 可能出现的各种情况:

(a) $x_i(k) = -1, x_i(k+1) = f[s_i(k)] = 1$ 时,

有 $\Delta x_i(k) = 2, s_i(k) \geq 0,$

则 $\Delta E(k) \leq 0$

(b) $x_i(k) = 1, x_i(k+1) = f[s_i(k)] = -1$ 时,

有 $\Delta x_i(k) = -2, s_i(k) < 0,$

$$\Delta E(k) = -\Delta x_i(k)s_i(k)$$

则 $\Delta E(k) < 0$

(c) $x_i(k+1) = x_i(k)$ 时, 有 $\Delta x_i(k) = 0$

则 $\Delta E(k) = 0$

6.3 典型神经网络模型 V//——Hopfield神经网络

可见，在任何情况下具有 $\Delta E(k) \leq 0$ ，由于 $E(k)$ 有下界，所以将收敛到一常数。下面需考虑 $E(k)$ 收敛到常数时，是否对应于网络的吸引子。根据上述分析，当 $\Delta E(k) = 0$ 时，对应于以下两种情况之一。

$$(a) \quad x_i(k+1) = x_i(k) = 1 \quad \text{或} \quad x_i(k+1) = x_i(k) = -1$$

$$(b) \quad x_i(k) = -1, x_i(k+1) = 1, s_i(k) = 0$$

$$\Delta E(k) = -\Delta x_i(k) s_i(k)$$

6.3 典型神经网络模型 V//——Hopfield神经网络

上述分析时，假设 $w_{ii}=0$ ，实际上 $w_{ii}>0$ 时上述结论仍然成立，而且收敛过程更快。

证毕。

$$\Delta E(k) = -\Delta x_i(k) \left[\sum_{j=1}^n w_{ij} x_j(k) - \theta_i \right] - \frac{1}{2} \Delta x_i^2 w_{ii}$$

6.3 典型神经网络模型 V//——Hopfield神经网络

定理 6.2 对于离散Hopfield网络，若按同步方式调整状态，且连接权矩阵 W 为非负定对称阵，则对于任意初态，网络都最终收敛到一个吸引子。

证明：前已求得

$$\begin{aligned}\Delta E(k) &= E(k+1) - E(k) = -\Delta x^T(k)[Wx(k) - \theta] - \frac{1}{2}\Delta x^T(k)W\Delta x(k) \\ &= -\Delta x^T(k)s(k) - \frac{1}{2}\Delta x^T(k)W\Delta x(k) = \sum_{i=1}^n \Delta x_i(k)s_i(k) - \frac{1}{2}\Delta x^T(k)W\Delta x(k)\end{aligned}$$

6.3 典型神经网络模型 V//——Hopfield神经网络

前已证得，对于所有的 i ，有 $-\Delta x_i(k)s_i(k) \leq 0$ ，因此只要 W 为非负定阵即有 $\Delta E(k) \leq 0$ ，也即最终 $E(k)$ 将收敛到一个常数值，并按照如上面同样的分析可说明网络最终将收敛到吸引子。

对于同步方式，它对连接权矩阵 W 的要求更高了，若不满足 W 为非负定对称阵的要求，则网络可能出现自持震荡或极限环。

6.3 典型神经网络模型 V//——Hopfield神经网络

(2) 吸引子的性质

定理 6.3 若 x 是网络的一个吸引子，且对于所有的 i ， $\theta_i = 0, \sum_{j=1}^n w_{ij}x_i \neq 0$ ，则 $-x$ 也一定是该网络的吸引子

证明：

$$x = f(Wx)$$

$$f(W(-x)) = f(-Wx) = -f(Wx) = -x$$

6.3 典型神经网络模型 V//—Hopfield神经网络

定理6.4 若 $x^{(a)}$ 是网络的吸引子，则与 $x^{(a)}$ 的海明距离 $d_H(x^{(a)}, x^{(b)}) = 1$ 的 $x^{(b)}$ 一定不是吸引子，海明距离定义为两个向量中不相同的元素的个数。

证明 不失一般性，设 $x_1^{(a)} \neq x_1^{(b)}, x_i^{(a)} = x_i^{(b)}, (i = 2, 3, \dots, n)$
因为 $w_{11} = 0$ 所以有

$$x_1^{(a)} = f\left[\sum_{j=2}^n w_{1j}x_j^{(a)} - \theta_1\right] = f\left[\sum_{j=2}^n w_{1j}x_j^{(b)} - \theta_1\right] \neq x_1^{(b)}$$

所以 $x^{(b)}$ 一定不是网络的吸引子。

6.3 典型神经网络模型 V//——Hopfield神经网络

(3) 吸引域

为了能够实现联想记忆，对于每一个吸引子应该有一定的吸引范围，这个吸引范围便称为吸引域。下面给出较严格的定义。

(a) 若 $x^{(a)}$ 是吸引子，对于异步方式，若存在一个更新次序可以从 x 演变 $x^{(a)}$ ，则称 x 弱吸引到 $x^{(a)}$ ；若对于任意更新次序都可以从 x 演变到 $x^{(a)}$ ，则称 x 强吸引至 $x^{(a)}$ 。

6.3 典型神经网络模型 V//——Hopfield神经网络

(b) 对所有 $x \in R(x^{(a)})$ 均有 x 弱（强）吸引到 $x^{(a)}$ ，
则称 $R(x^{(a)})$ 为 $x^{(a)}$ 的弱（强）吸引域。

对于同步方式，由于无调整次序问题，所以相应的吸引域也无强弱之分。

对于异步方式，对同一个状态，若采用不同的调整次序，有可能弱吸引到不同的吸引子。

6.3 典型神经网络模型 V//——Hopfield神经网络

3) 连接权的设计

Hopfield网络没有与之相关的学习规则。它的权值不被训练，也不会自己学习。它的权值矩阵是事前用基于Lyapunov函数的设计过程采用Hebb规则来计算出来的。

在这种网络中，不断更新的不是权值，而是网络中各神经元的状态，网络演变到稳定时各神经元的状态便是问题的解。

6.3 典型神经网络模型 V//——Hopfield神经网络

为了保证Hopfield网络在异步方式工作时能稳定收敛，连接权矩阵 W 应是对称的。若要保证同步方式收敛，则要求 W 为非负定阵，这个要求比较高。因而设计一般只保证异步方式收敛。另外一个要求是对于给定的样本必须是网络的吸引子，而且要有一定的吸引域，这样才能正确实现联想记忆功能。

设给定 m 个样本 $x^{(k)} (k=1,2,\cdots,m)$ ，为了实现上述功能，通常采用有监督的Hebb规则（用输入模式作为目标模式）来设计连接权，连接权可按以下两种情况进行计算。

6.3 典型神经网络模型 V//——Hopfield神经网络

(1) 当网络节点状态为1和-1两种状态即 $x \in \{-1, 1\}^n$ 时，相应的连接权为

$$w_{ij} = \begin{cases} \sum_{k=1}^m x_i^{(k)} x_j^{(k)} & i \neq j \\ 0 & i = j \end{cases} \quad (6-51)$$

写成矩阵形式则为

$$\begin{aligned} W &= \begin{bmatrix} x^{(1)} & x^{(2)} & \cdots & x^{(m)} \end{bmatrix} \begin{bmatrix} x^{(1)T} \\ x^{(2)T} \\ \vdots \\ x^{(m)T} \end{bmatrix} - mI \\ &= \sum_{k=1}^m x^{(k)} x^{(k)T} - mI_m = \sum_{k=1}^m \left(x^{(k)} x^{(k)T} - I \right) \end{aligned} \quad (6-52)$$

其中 I 为单位矩阵。

6.3 典型神经网络模型 V//——Hopfield神经网络

(2) 当网络节点状态为1和0两种状态即 $x \in \{0,1\}^n$ 时，相应的连接权为

$$w_{ij} = \begin{cases} \sum_{k=1}^m (2x_i^{(k)} - 1)(2x_j^{(k)} - 1) & i \neq j \\ 0 & i = j \end{cases} \quad (6-53)$$

写成矩阵形式则为

$$W = \sum_{k=1}^m (2x^{(k)} - b)(2x^{(k)} - b)^T - mI \quad (6-54)$$

其中 $b = [1 \ 1 \ \dots \ 1]^T$

6.3 典型神经网络模型 V//——Hopfield神经网络

例 6-7 对离散Hopfield网络进行设计。其中网络节点数为 $n=4$, $\theta_i=0$ ($i=1, 2, 3, 4$), 样本数为 $m=2$, 两个样本为

$$x^{(1)} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, x^{(2)} = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$$

解 首先根据Hebb规则式(6-51)求得连接权矩阵为

$$W = x^{(1)} x^{(1)T} + x^{(2)} x^{(2)T} - 2I = \begin{bmatrix} 0 & 2 & 2 & 2 \\ 2 & 0 & 2 & 2 \\ 2 & 2 & 0 & 2 \\ 2 & 2 & 2 & 0 \end{bmatrix}$$

$$W = \sum_{k=1}^m \left(x^{(k)} x^{(k)T} - I \right)$$

6.3 典型神经网络模型 V//——Hopfield神经网络

例 6-8 对离散Hopfield网络进行设计。其中，网络节点数为 $n=4$, $\theta_i=0$ ($i=1,2,3,4$), 样本数为 $m=2$, 两个样本为

$$x^{(1)} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, x^{(2)} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

$$W = \sum_{k=1}^m (2x^{(k)} - b)(2x^{(k)} - b)^T - mI$$

6.3 典型神经网络模型 V//——Hopfield神经网络

解 首先根据Hebb规则式(6-58)求得连接权矩阵为

$$w_{12} = (2x_1^{(1)} - 1)(2x_2^{(1)} - 1) + (2x_1^{(2)} - 1)(2x_2^{(2)} - 1) = (1) \times (-1) + (-1) \times (1) = -2 = w_{21}$$

$$w_{13} = (2x_1^{(1)} - 1)(2x_3^{(1)} - 1) + (2x_1^{(2)} - 1)(2x_3^{(2)} - 1) = (1) \times (1) + (-1) \times (-1) = 2 = w_{31}$$

$$w_{14} = (2x_1^{(1)} - 1)(2x_4^{(1)} - 1) + (2x_1^{(2)} - 1)(2x_4^{(2)} - 1) = (1) \times (-1) + (-1) \times (1) = -2 = w_{41}$$

$$w_{23} = (2x_2^{(1)} - 1)(2x_3^{(1)} - 1) + (2x_2^{(2)} - 1)(2x_3^{(2)} - 1) = (-1) \times (1) + (-1) \times (1) = -2 = w_{32}$$

$$w_{24} = (2x_2^{(1)} - 1)(2x_4^{(1)} - 1) + (2x_2^{(2)} - 1)(2x_4^{(2)} - 1) = (-1) \times (-1) + (1) \times (1) = 2 = w_{42}$$

$$w_{34} = (2x_3^{(1)} - 1)(2x_4^{(1)} - 1) + (2x_3^{(2)} - 1)(2x_4^{(2)} - 1) = (1) \times (-1) + (-1) \times (1) = -2 = w_{43}$$

$$W = \begin{bmatrix} 0 & -2 & 2 & -2 \\ -2 & 0 & -2 & 2 \\ 2 & -2 & 0 & -2 \\ -2 & 2 & -2 & 0 \end{bmatrix}$$

6.3 典型神经网络模型 V//——Hopfield神经网络

2. 连续型 Hopfield 网络

1) 网络的结构和工作方式

连续型神经网络的各神经元是并行（同步）工作的。对于每一个神经元节点，其工作方式为

$$\begin{aligned} s_i &= \sum_{j=1}^n w_{ij} x_j - \theta_i \\ \frac{dy_i}{dt} &= -\frac{1}{\tau} y_i + s_i \\ x_i &= f(y_i) \end{aligned} \tag{6-55}$$

一阶微分方程，相当于一阶惯性环节。 s_i 是该环节的输入， y_i 是该环节的输出。

6.3 典型神经网络模型 V//——Hopfield神经网络

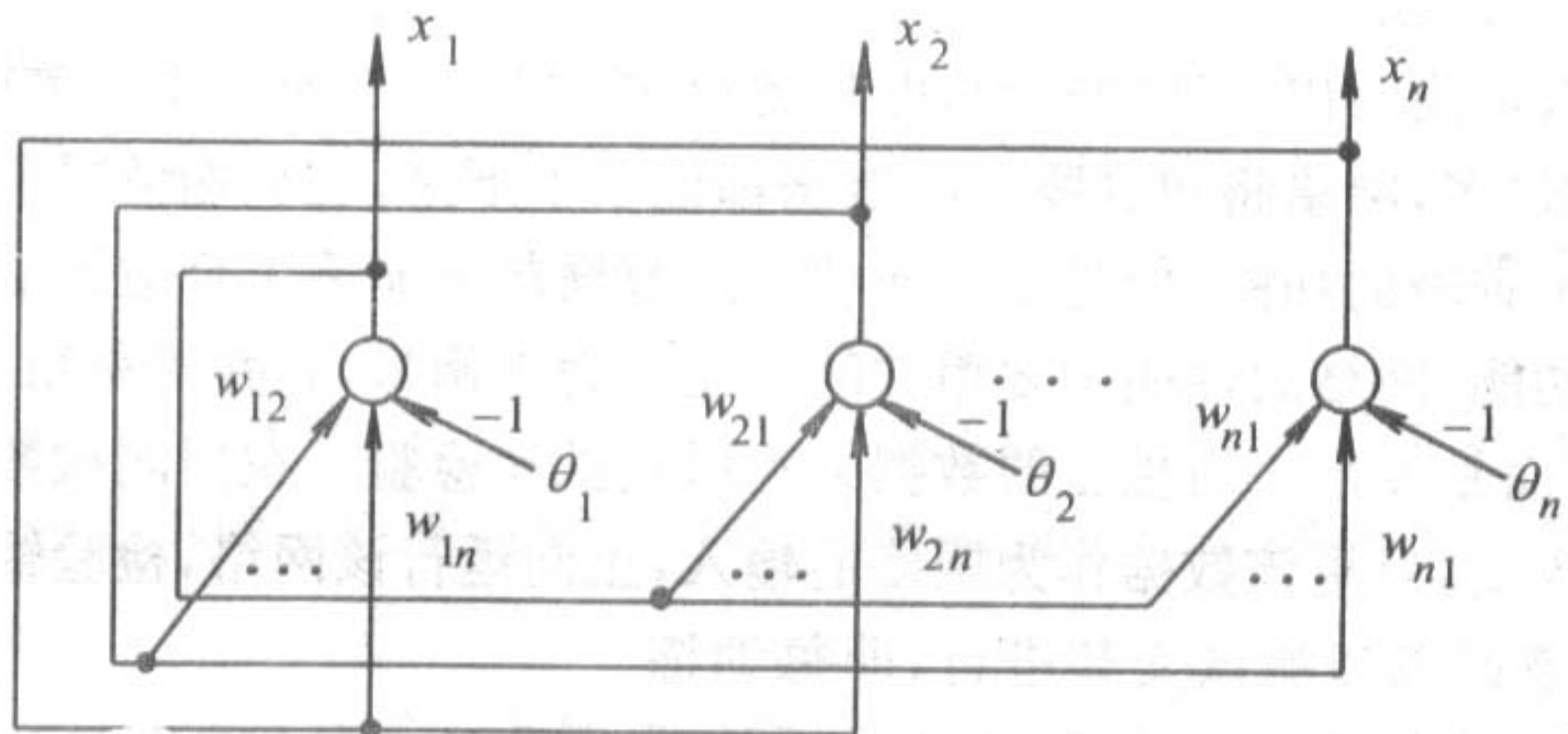


图6-39 连续型Hopfield网络

6.3 典型神经网络模型 V//——Hopfield神经网络

f 函数一般取S形函数，当 $x_i \in (-1,1)$ 时，取

$$x_i = f(y_i) = \frac{1 - e^{-\mu y_i}}{1 + e^{-\mu y_i}}$$

当 $x_i \in (0,1)$ 时，取 $x_i = f(y_i) = \frac{1}{1 + e^{-\mu y_i}}$

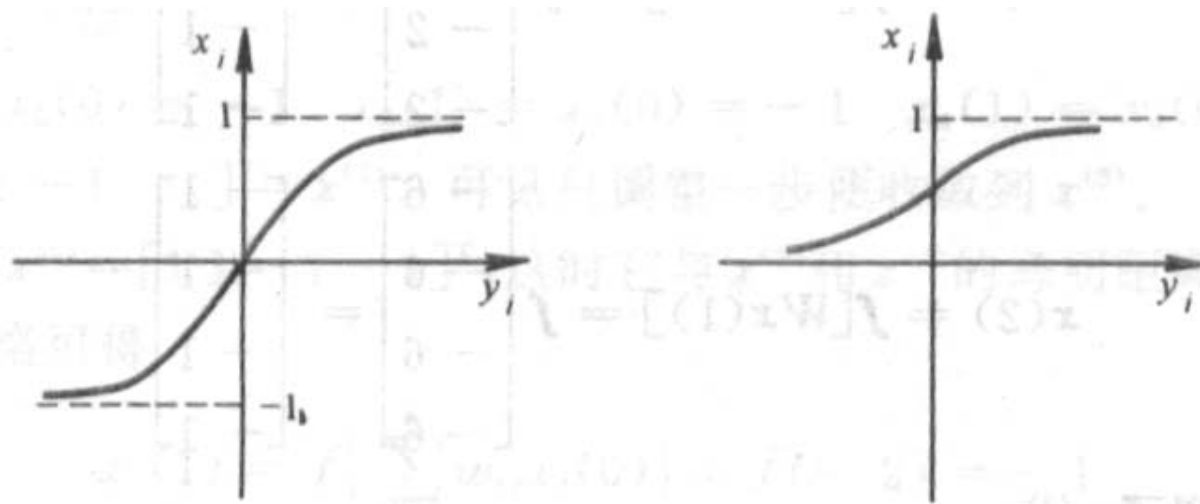


图6-40 变换函数（连续型Hopfield网络）

6.3 典型神经网络模型 V//——Hopfield神经网络

Hopfield利用模拟电路设计了一个连续Hopfield网络的电路模型。图6-41表示了其中由运算放大器电路实现的一个节点的模型。

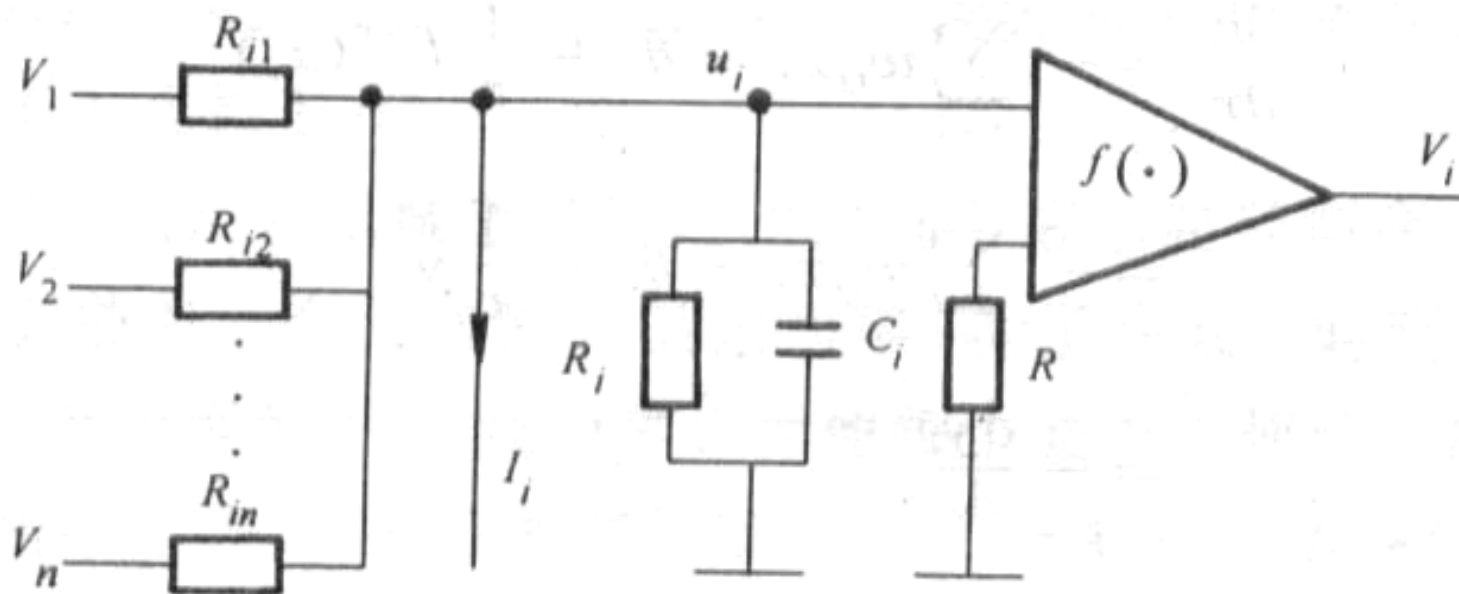


图6-41 连续Hopfield网络的电路模型

6.3 典型神经网络模型 V//——Hopfield神经网络

根据图 6-41 可以列出如下的电路方程：

$$\begin{cases} C_i \frac{du_i}{dt} + \frac{u_i}{R_i} + I_i = \sum_{j=1}^n \frac{V_j - u_i}{R_{ij}} \\ V_i = f(u_i) \end{cases}$$

经整理得

$$\begin{cases} \frac{du_i}{dt} = -\frac{1}{R'_i C_i} u_i + \sum_{j=1}^n \frac{1}{R_{ij} C_i} V_j - \frac{I_i}{C_i} \\ V_i = f(u_i) \end{cases}$$

其中

$$\frac{1}{R'_i} = \frac{1}{R_i} + \sum_{j=1}^n \frac{1}{R_{ij}}$$

6.3 典型神经网络模型 V//——Hopfield神经网络

若令 $x_i = V_i$, $y_i = u_i$, $\tau = R'_i C_i$, $w_{ij} = \frac{1}{R_{ij} C_i}$, $\theta_i = \frac{I_i}{C_i}$

则上式化为

$$\begin{cases} \frac{dy_i}{dt} = -\frac{1}{\tau} y_i + \sum_{j=1}^n w_{ij} x_j - \theta_i \\ x_i = f(y_i) \end{cases} \quad (6-56)$$

式中 $f(\cdot)$ 常用 **Sigmoid** 函数:

$$x_i = f(y_i) = \frac{1}{2} \left[1 + \tanh\left(\frac{y_i}{y_0}\right) \right]$$

$$\tanh(x) = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

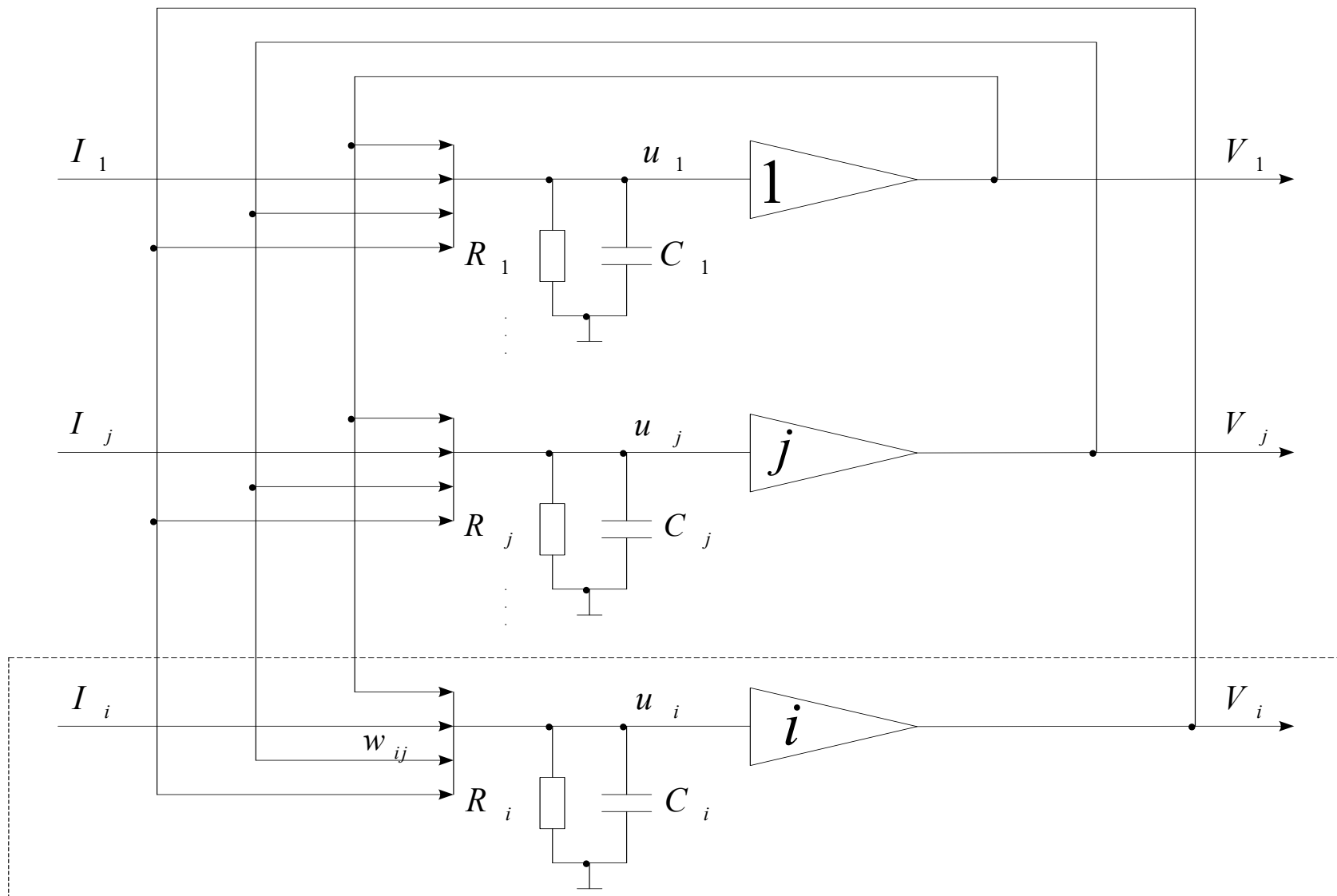


图 6-42 用运算放大器构造的连续型Hopfield网络

6.3 典型神经网络模型 V//——Hopfield神经网络

2) 稳定性

定义连续Hopfield网络能量函数为

$$\begin{aligned} E &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} x_i x_j + \sum_{i=1}^n x_i \theta_i + \sum_{i=1}^n \frac{1}{\tau_i} \int_0^{x_i} f^{-1}(\eta) d\eta \\ &= -\frac{1}{2} x^T W x + x^T \theta + \sum_{i=1}^n \frac{1}{\tau_i} \int_0^{x_i} f^{-1}(\eta) d\eta \end{aligned} \quad (6-57)$$

该能量函数的表达式与离散Hopfield网络的定义是完全相同的。对于离散Hopfield网络，由于 $f(\cdot)$ 是二值函数，所以第三项的积分项为零。

6.3 典型神经网络模型 V//——Hopfield神经网络

由于 $x_i \in (-1,1)$ 或 $x_i \in (0,1)$ ，因此上述定义的能量函数 E 是有界的，因此只需证得 $dE/dt \leq 0$ 既可说明系统是稳定的，因

$$\frac{dE}{dt} = \sum_{i=1}^n \frac{\partial E}{\partial x_i} \frac{dx_i}{dt}$$

根据式（6-57）所述 E 的表达式可以求得

$$\frac{\partial E}{\partial x_i} = -\sum_{j=1}^n w_{ij} x_j + \theta_i + \frac{1}{\tau_i} f^{-1}(x_i) = -\sum_{j=1}^n w_{ij} x_j + \theta_i + \frac{1}{\tau_i} y_i = -\frac{dy_i}{dt} \quad (6-58)$$

代入上式得

$$\frac{dE}{dt} = \sum_{i=1}^n \left(-\frac{dy_i}{dt} \frac{dx_i}{dt} \right) = -\sum_{i=1}^n \left(\frac{dy_i}{dx_i} \frac{dx_i}{dt} \frac{dx_i}{dt} \right) = -\sum_{i=1}^n \left(\frac{dy_i}{dx_i} \left(\frac{dx_i}{dt} \right)^2 \right)$$

6.3 典型神经网络模型 V//——Hopfield神经网络

前面已假设 $x_i = f(y_i)$ 是单调上升函数，显然它的反函数为单调上升函数，即有 $dy_i/dx_i \geq 0$ ，同时 $(dy_i/dt)^2 \geq 0$ 。因而有

$$\frac{dE}{dt} \leq 0 \quad (\text{所有 } x_i \text{ 均为常数时才取等号})$$

根据李雅普诺夫稳定性理论，该网络系统一定是渐近稳定的。即随着时间的演变，网络状态总是朝 E 减小的方向运动，一直到 E 取得极小值，这时所有的 x_i 变为常数，也即网络收敛到稳定状态。

6.3 典型神经网络模型 V//——Hopfield神经网络

连续 Hopfield 网络主要用来进行优化计算。因此，如何设计连接权系数及其它参数需根据具体问题来加以确定。

下面以连续型 Hopfield 神经网络应用于 TSP (Travelling Salesman Problem) 为例加以说明。TSP 问题是人工智能中的一个难题。

6.3 典型神经网络模型 V//——Hopfield神经网络

例6-9 旅行商问题 (Traveling Salesman Problem, 简称TSP) 可描述为:

已知N个城市之间的相互距离, 现有一推销员必须遍访这N个城市, 并且每个城市只能访问一次, 最后又必须返回出发城市。如何安排他对这些城市的访问次序, 使其旅行路线总长度最短。

解 这是一个典型的组合优化问题。下面要解决的问题是如何恰当地描述该问题, 使其适合于用 Hopfield网络来求解。正是由于 Hopfield 成功地求解了 TSP 问题, 才使得人们对神经网络再次引起了广泛的兴趣。

6.3 典型神经网络模型 V//——Hopfield神经网络

1 求解TSP问题的Hopfield网络设计

TSP问题是在一个城市集合 $\{A_c, B_c, C_c, \dots\}$ 中找出一个最短且经过每个城市各一次并回到起点的路径。为了将**TSP**问题映射为一个神经网络的动态过程，**Hopfield**采取了换位矩阵的表示方法，用 $N \times N$ 矩阵表示商人访问 N 个城市。例如，有四个城市 $\{A_c, B_c, C_c, D_c\}$ ，访问路线是：

$D_c \rightarrow A_c \rightarrow C_c \rightarrow B_c \rightarrow D_c$ 则**Hopfield**网络输出所代表的有效解用下面的二维矩阵表6-2来表示：

城 市 次 序	1	2	3	4
A_c	0	1	0	0
B_c	0	0	0	1
C_c	0	0	1	0
D_c	1	0	0	0

表6-2 四个城市的访问路线

6.3 典型神经网络模型 V//——Hopfield神经网络

表6-2 构成了一个 4×4 的矩阵，该矩阵中，各行各列只有一个元素为 1，其余为 0，否则是一个无效的路径。采用 V_{xi} 表示神经元 (x, i) 的输出，相应的输入用 U_{xi} 表示。如果城市 x 在 i 位置上被访问，则 $V_{xi}=1$ ，否则 $V_{xi}=0$ 。

针对TSP问题，Hopfield 定义了如下形式的能量函数：

$$\begin{aligned} E = & \frac{A}{2} \sum_{x=1}^N \sum_{i=1}^N \sum_{j=1}^N V_{xi} V_{xj} + \frac{B}{2} \sum_{i=1}^N \sum_{x=1}^N \sum_{y=x}^N V_{xi} V_{yi} \\ & + \frac{C}{2} \left(\sum_{x=1}^N \sum_{i=1}^N V_{xi} - N \right)^2 + \frac{D}{2} \sum_{x=1}^N \sum_{y=1}^N \sum_{i=1}^N d_{xy} V_{xi} (V_{y,i+1} + V_{y,i-1}) \end{aligned} \quad (6-59)$$

式中， A, B, C, D 是权值， d_{xy} 表示城市 x 到城市 y 之间的距离。

6.3 典型神经网络模型 V//——Hopfield神经网络

E 的前三项是问题的约束项，最后一项是优化目标项， E 的第一项为保证矩阵 V 的每一行不多于一个 1 时 E 最小（即每个城市只去一次）， E 的第二项保证矩阵的每一列不多于一个 1 时 E 最小（即每次只访问一城市）， E 的第三项保证矩阵 V 中 1 的个数恰好为 N 时 E 最小。

Hopfield将能量函数的概念引入神经网络，开创了求解优化问题的新方法。但该方法在求解上存在局部极小、不稳定等问题。为此，将TSP的能量函数定义为：

$$E = \frac{A}{2} \sum_{x=1}^N \left(\sum_{i=1}^N V_{xi} - 1 \right)^2 + \frac{A}{2} \sum_{i=1}^N \left(\sum_{x=1}^N V_{xi} - 1 \right)^2 + \frac{D}{2} \sum_{x=1}^N \sum_{y=1}^N \sum_{i=1}^N V_{xi} d_{xy} V_{y,i+1}$$

(6-60)

6.3 典型神经网络模型 V//——Hopfield神经网络

Hopfield 网络的动态方程为:

$$\begin{aligned}\frac{dU_{xi}}{dt} &= -\frac{\partial E}{\partial V_{xi}}(x, i=1, 2, \dots, N-1) \\ &= -A\left(\sum_{i=1}^N V_{xi} - 1\right) - A\left(\sum_{y=1}^N V_{yi} - 1\right) - D\sum_{y=1}^N d_{xy}V_{y,i+1}\end{aligned}\quad (6-61)$$

采用Hopfield网络求解TSP问题的算法描述如下:

- (1) 置初值: $t=0$, $A=1.5$, $D=1.0$, $\mu=50$; $V_{xi}(t) = \frac{1}{1 + e^{-\mu U_{xi}(t)}}$
- (2) 计算N个城市之间的距离 d_{xy} ($x, y=1, 2, \dots, N$);
- (3) 神经网络输入 $U_{xi}(t)$ 的初始化在0附近产生;
- (4) 利用动态方程 (6-61) 计算 $\frac{dU_{xi}}{dt}$;

6.3 典型神经网络模型 V//——Hopfield神经网络

(5) 根据一阶欧拉法计算

$$U_{xi}(t+1) = U_{xi}(t) + \frac{dU_{xi}}{dt} \Delta T$$

(6) 为了保证收敛于正确解，即矩阵 V 各行各列只有一个元素为 1，其余为 0，采用 Sigmoid 函数计算

$$V_{xi}(t) = \frac{1}{1 + e^{-\mu U_{xi}(t)}}$$

其中 $\mu > 0$ ， μ 值的大小决定了 Sigmoid 函数的形状。

6.3 典型神经网络模型 V//——Hopfield神经网络

- (7) 根据式 (6.60)，计算能量函数 E ；
- (8) 检查路径的合法性，判断迭代次数是否结束，如果结束，
则终止，否则返回到第 (4) 步；
- (9) 显示输出迭代次数、最优路径、最优能量函数、路径长度的值，并作出能量函数随时间的变化的曲线图。

6.3 典型神经网络模型 V//——Hopfield神经网络

3 仿真实例

在TSP的Hopfield网络能量函数（6.60）式中，取 $A=1.5$ ， $D=1.0$ 。采样时间取 $\Delta T=0.01$ ，网络输入 $U_{xi}(t)$ 初始值选择在 $[-0.001,+0.001]$ 间的随机值，在式（6.61）的函数中，取较大的 μ ，以使函数比较陡峭，从而稳态时 $V_{xi}(t)$ 能够趋于1或趋于0。

以8个城市的路径优化为例，其城市路径坐标保存在当前路径的程序 **cities8.txt** 中。如果初始化的寻优路径有效，即路径矩阵中各行各列只有一个元素为1，其余为0，则给出最后的优化路径，否则停止优化，需要重新运行优化程序。如果本次寻优路径有效，经过 **2000** 次迭代，最优能量函数为 **Final_E=1.4468**，初始路程为 **Initial_Length=4.1419**，最短路程为 **Final_Length =2.8937**。

6.3 典型神经网络模型 V//——Hopfield神经网络

由于网络输入 $U_{xi}(t)$ 初始选择的随机性，可能会导致初始化的寻优路径无效，即路径矩阵中各行各列不满足“只有一个元素为1，其余为0”的条件，此时寻优失败，停止优化，需要重新运行优化程序。仿真过程表明，在100次仿真实验中，有90次以上可收敛到最优解。

6.3 典型神经网络模型 V//——Hopfield神经网络

- **sumsqr(X)**: 求矩阵**X**中各元素的平方值之和;
- **Sum(X)**或**Sum(X,1)**为矩阵中各行相加, **Sum(X,2)**为矩阵中各列相加;
- **repmat**: 用于矩阵复制, 例如, $\mathbf{X} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, 则 $\text{repmat}(\mathbf{X}, 1, 1) = \mathbf{X}$

$$\text{repmat}(\mathbf{X}, 1, 2) = \begin{bmatrix} 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 4 \end{bmatrix} \quad \text{repmat}(\mathbf{X}, 2, 1) = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 1 & 2 \\ 3 & 4 \end{bmatrix}$$

- **dist(x,y)**: 计算两点间的距离, 例如 $\mathbf{x} = [1 \ 1]$, $\mathbf{y} = [2 \ 2]'$, 则

$$\text{dist}(\mathbf{x}, \mathbf{y}) = \sqrt{(2-1)^2 + (2-1)^2} = \sqrt{2}$$

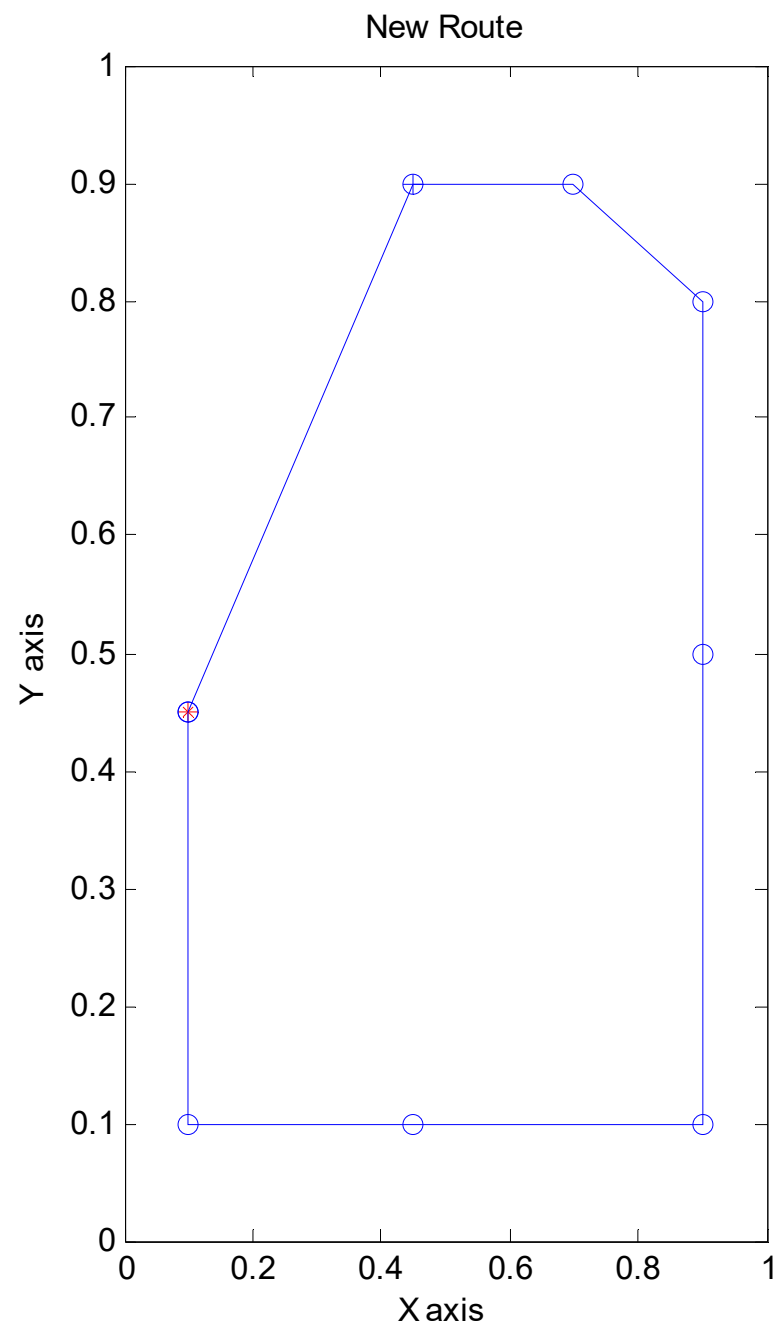
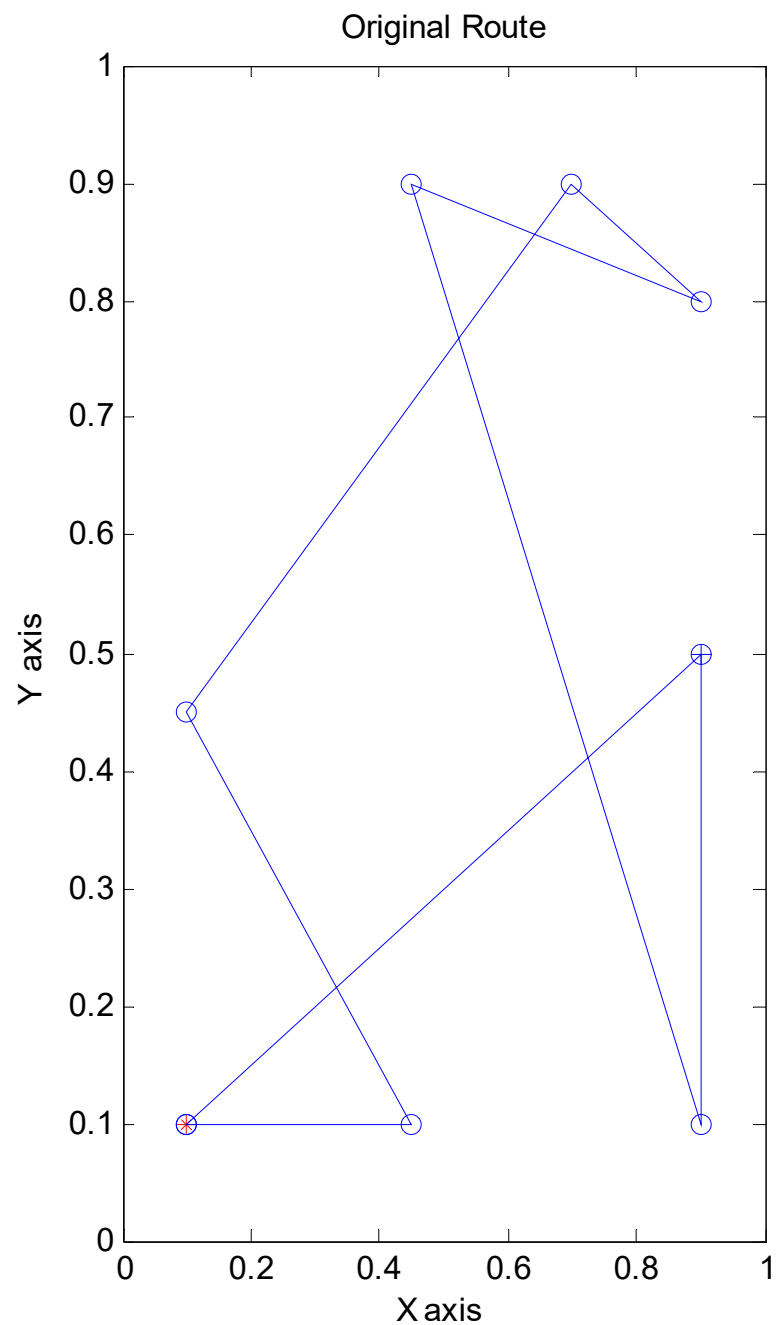


图6-43 初始路径及优化后的路径

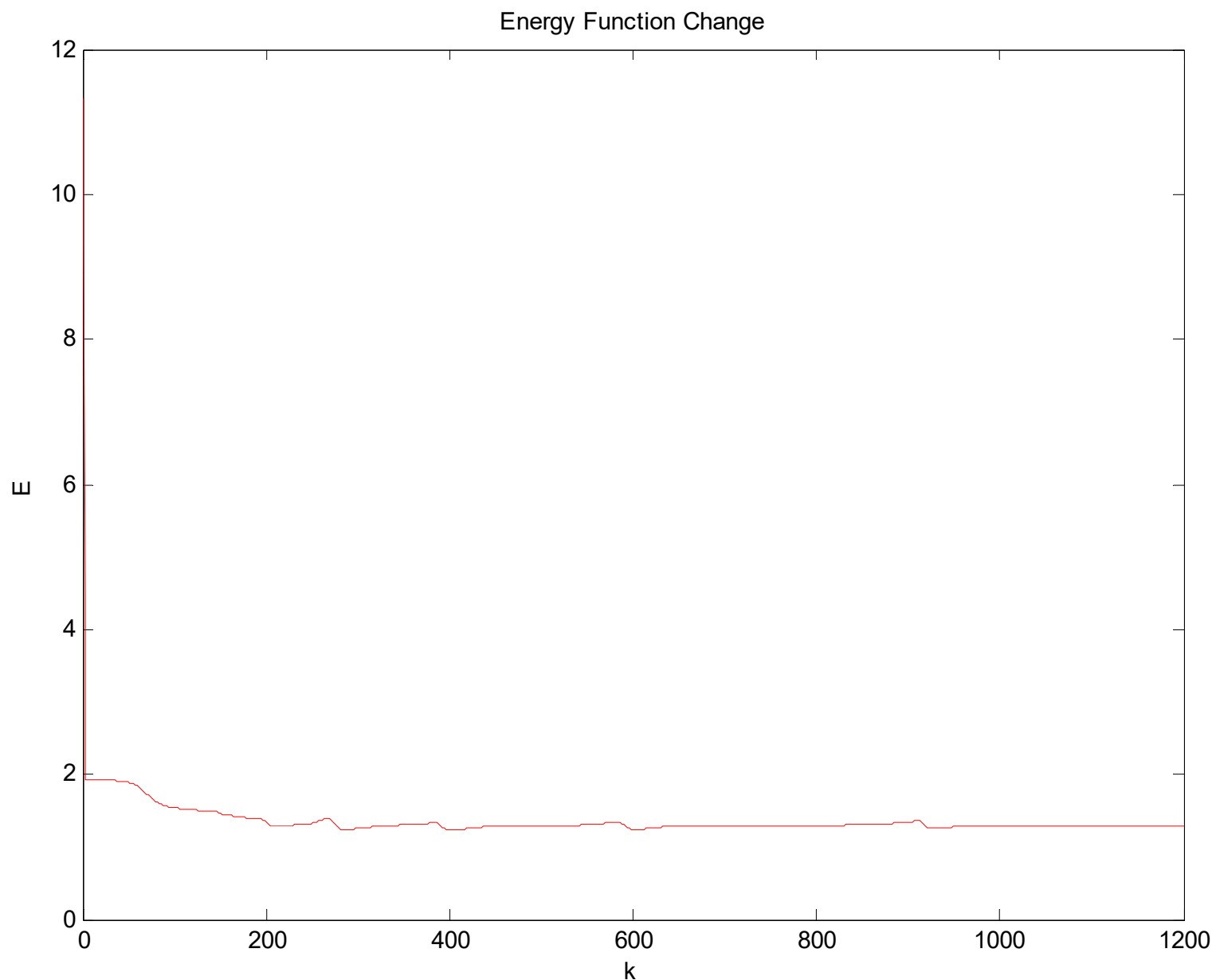


图6-44 能量函数随迭代次数的变化

6.4 神经网络的训练

神经网络可应用于许多方面，它主要有以下特点。

（1）具有自适应功能

根据所提供的数据，通过学习和训练，找出和输出之间的内在联系，从而求得问题的解答，而不是依靠对问题的先验知识和规则，因而它具有很好的适应性。

6.4 神经网络的训练

(2) 具有泛化功能

它能处理那些未经训练过的数据。而获得相应于这些数据的合适的解答。同样，它也能够处理那些有噪声或不完全的数据，从而显示了很好的容错能力。对于许多实际问题来说，泛化能力是非常有用的，因为现实世界所获得的数据常常受到一定的污染或残缺不全。

(3) 非线性映射功能

现实的问题常常是非常复杂的，各个因素之间互相影响，呈现出复杂的非线性关系，神经元网络为处理这些问题提供了有用的工具。

6.4 神经网络的训练

(4) 高度并行处理

神经网络的处理是高度并行的，因此用硬件实现的神经网络的处理速度可远远高于通常计算机的处理速度。与常规的计算机程序相比较，神经网络主要基于所测量的数据对系统进行建模、估计和逼近，它可应用于如分类、预测及模式识别等众多方面。

6.4 神经网络的训练

传统的计算机程序与神经网络

- ◆ 传统的计算机程序比较适合于那些需要高精度的数值计算或者需要符号处理的那些任务。例如财务管理和计算，它比较适合于采用计算机程序，而不适合于采用神经网络。
- ◆ 对于那些几乎没有规则，数据不完全或者多约束优化问题，则适合于用神经网络。例如用神经网络来控制一个工业过程便是这样的例子，对于这种情况很难定义规则，历史数据很多而且充满噪声，准确的计算是毫无必要的。

◆ 某些情况下应用神经网络会存在严重的缺点

- 当所给数据不充分或不存在可学习的映射关系时，神经网络可能找不到满意的解。
- 其次，有时很难估计神经网络给出的结果。神经网络中的连接权系数是千万次数据训练后的结果，对它的意义很难给出明显的解释，它对输出结果的影响也是非常复杂的。
- 神经网络的训练是很慢的，而且有时需要付出很高的代价，这一方面是由于需要收集、分析和处理大量的训练数据，同时还需要相当的经验来选择合适的参数。
- 神经网络在实际应用时的执行时间也是需要加以检验的。执行时间取决于连接权的个数，它大体与网络节点数的平方成正比。因此网络节点的稍许增加便可能引起执行时间的很大增长。

6.4 神经网络的训练

训练神经网络的步骤

(1) 产生数据样本集

原始数据的收集、数据分析、变量选择以及数据的预处理，只有经过这些步骤后，才能对神经网络进行有效的学习和训练。

- 首先要在大量的原始测量数据中确定出最主要的输入模式。
- 需进行尺度变换和预处理, 尺度变换常常将输入变换到 $[-1, 1]$ 或 $[0, 1]$ 的范围。通过对数据的预处理分析还可以检验其是否存在周期性、固定变化趋势或其它关系。

6.4 神经网络的训练

- 预留测试数据。将收集到的可用数据随机地分成两部分，譬如说其中三分之二用于网络的训练，另外三分之一用于将来的测试，随机选取的目的是为了尽量减小这两部分数据的相关性。
- 预先加以分类。影响数据大小的另一个因素是输入模式和输出结果的分布，对数据预先加以分类可以减少所需的数据量。相反，数据稀薄不匀甚至互相覆盖则势必要增加数据量。

6.4 神经网络的训练

(2) 确定网络的类型和结构

一般是从已有的网络类型中选用一种比较简单而又能满足要求的网络，若新设计一个网络类型来满足问题的要求往往比较困难。

在网络的类型确定后，剩下的问题是选择网络的结构和参数。以BP网络为例，需选择网络的层数、每层的节点数、初始权值、阈值、学习算法、数值修改频度、节点变换函数及参数、学习率及动量项因子等参数。

6.4 神经网络的训练

对于具体问题若确定了输入和输出变量后，网络输入层和输出层的节点个数也便随之确定了。对于隐层的层数可首先考虑只选择一个隐层。剩下的问题是如何选择隐层的节点数。其选择原则是：在能正确反映输入输出关系的基础上，尽量选取较少的隐层节点数，而使网络尽量简单。具体选择可有如下两种方法：

a. 先设置较少的节点，对网络进行训练，并测试网络的逼近误差（后面还将介绍训练和测试的具体方法），然后逐渐增加节点数，直到测试的误差不再有明显的减小为止。

6.4 神经网络的训练

b. 先设置较多的节点，在对网络进行训练时，采用如下的误差代价函数

$$J_f = \frac{1}{2} \sum_{p=1}^P \sum_{i=1}^{N_Q} (t_{pi} - x_{pi}^{(Q)})^2 + \varepsilon \sum_{q=1}^Q \sum_{i=1}^{N_Q} \sum_{j=1}^{N_{Q-1}} |w_{ij}^{(q)}| = J + \varepsilon \sum_{q,i,j} |w_{ij}^{(q)}|$$

其中 J 仍与以前的定义相同，它表示输出误差的平方和。引入第二项的作用相当于引入一个“遗忘”项，其目的是为了使训练后的连接权系数尽量小。可以求得这时 J_f 对 w_{ij} 的梯度为

$$\frac{\partial J_f}{\partial w_{ij}^{(q)}} = \frac{\partial J}{\partial w_{ij}^{(q)}} + \varepsilon \operatorname{sgn}(w_{ij}^{(q)}) \quad (6-62)$$

6.4 神经网络的训练

利用该梯度可以求得相应的学习算法。利用该学习算法，在训练过程中只有那些确实必要的连接权才予以保留，而那些不很必要的连接将逐渐衰减为零。最后可去掉那些影响不大的连接权和相应的节点，从而得到一个适当规模的网络结构。

若采用上述任一方法选择得到的隐层节点数太多。这时可考虑采用二个隐层。为了达到相同的映射关系，采用二个隐层的节点总数常常可比只用一个隐层时少。

6.4 神经网络的训练

(3)训练和测试

最后一步是对网络进行训练和测试，在训练过程中对训练样本数据需要反复地使用。对所有样本数据正向运行一次并反传修改连接权一次称为一次训练（或一次学习），这样的训练需要反复地进行下去直至获得合适的映射结果。通常训练一个网络需要成百上千次。

特别应该注意的一点是，并非训练的次数越多，越能得到正确的输入输出的映射关系。训练网络的目的在于找出蕴含在样本数据中的输入和输出之间的本质联系，从而对于未经训练的输入也能给出合适的输出，即具备泛化功能。由于所收集的数据都是包含噪声的，训练的次数过多，网络将包含噪声的数据都记录了下来，在极端情况下，训练后的网络可以实现相当于查表的功能。但是对于新的输入数据却不能给出合适的输出，也即并不具备很好的泛化功能。

网络的性能主要用它的泛化能力来衡量，它并不是用对训练数据的拟合程度来衡量，而是要用一组独立的数据来加以测试和检验。在用测试数据检验时，保持连接权系数不改变，只用该数据作为网络的输入，正向运行该网络，检验输出的均方误差。

6.4 神经网络的训练

实际操作时应该训练和测试交替进行，即每训练一次，同时用测试数据测试一遍，画出均方误差随训练次数的变化曲线，如图6-45所示。

从误差曲线可以看出，在用测试数据检验时，均方误差开始逐渐减小，当训练次数再增加时，测试检验误差反而增加。误差曲线上极小点所对应的即为恰当的训练次数，若再训练即为“过度训练”了。

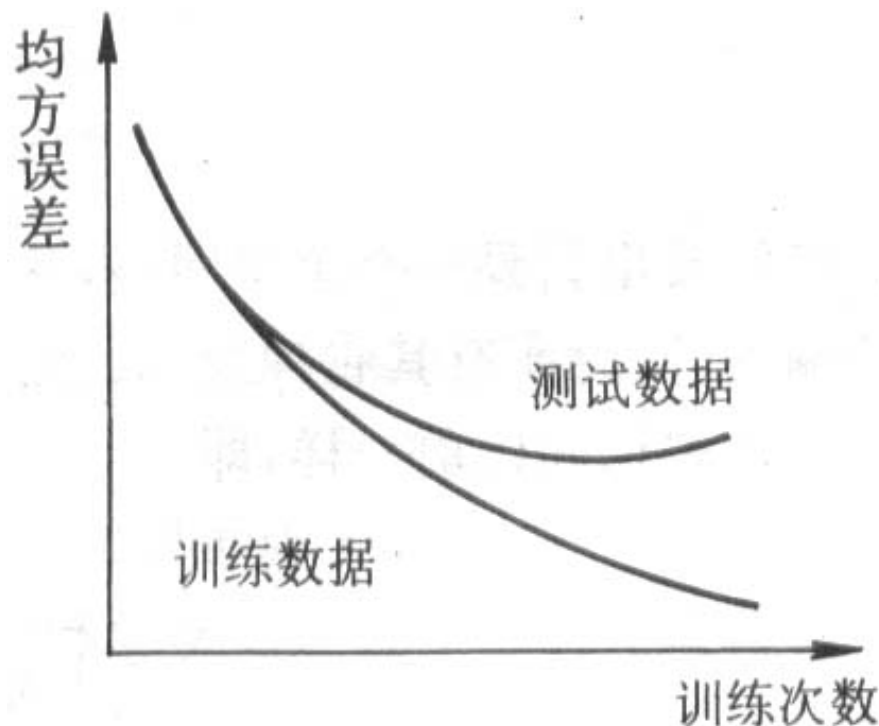


图6-45 均方误差曲线

6.4 神经网络的训练

对于网络隐层节点数的选择如果采用试验法，也必须将训练与测试相结合，最终也用测试误差来衡量网络的性能。均方误差与隐层节点数也有与图6-45相类似的关系，因此也不是节点数越多越好。

网络的节点数对网络的泛化能力有很大影响，节点数太多，它倾向于记住所有的训练数据，包括噪声的影响，反而降低了泛化能力；而节点数太少，它不能拟合样本数据，因也就谈不上有较好的泛化能力。选择节点数的原则是：选择尽量少的节点数以实现尽量好的泛化能力。

小 结

本章首先详细地介绍了神经网络的基本概念，包括生物神经元的结构与功能特点、人工神经元模型、神经网络的结构、神经网络的工作方式、神经网络的学习方法和神经网络的分类。最后重点介绍了十几种常用神经网络的拓扑结构、工作方式和学习算法及其利用**MATLAB**实现的方法。

作业

1. 采用**RBF**网络逼近非线性对象

$$y(k) = \frac{u(k-1) - 0.9y(k-1)}{1 + y^2(k-1)}$$

并进行**Matlab**仿真。

2. 构造30个城市的位置坐标，采用**Hopfield**网络实现30个城市路径的**TSP**问题优化，并进行**Matlab** 仿真。

谢 谢!