张立明 编著

复旦大学出版社

Models and Applications of Artificial Neural Networks

人工神经网络的模型及其应用

目　录

第一章 概 论

人工神经网络是最近发展起来的十分热门的交叉学科，它涉及生物、电子、计算机、数学和物理等学科，有着非常广泛的应用背景，这门学科的发展对目前和未来的科学技术的发展将有重要的影响。

长期以来，人们想方设法了解人脑的功能，用物理可实现系统去模仿人脑，完成类似于人脑的工作。计算机就是采用电子元件的组合来完成人脑的某些记忆、计算、判断功能的物理系统。用机器代替人脑的部分劳动是当今科学技术发展的重要标志。现代计算机中每个电子元件的计算速度为纳秒(ns)级，人脑中每个神经细胞的反应时间只有毫秒(ms)级。这样，似乎计算机的运算能力应为人脑的几百万倍。可是，迄今为止，计算机在解决信息初级加工时如视觉、听觉、嗅觉这类简单的感觉识别上却十分迟钝。人在识别文字、图像、声音等方面的能力大大超过计算机，现代计算机要花几十分钟(min)，甚至几小时(h)才能完成的识别任务，人只要零点几秒(s)就可以完成了。同样，在机器人控制、人工智能、思维、直觉等方面，就更无法与人相比了。人们希望去追求一种新型的计算机系统，它既有超越于人的计算能力，又有类似于人的识别、智能、联想的能力。

从人脑的结构看，它是由大量神经细胞组合而成的，这些细胞相互联接，每个细胞完成其某一种基本功能，如兴奋与抑制。从整体看，它们相互整合完成一种复杂的计算机思维活动，这些工作是并行的，有机的关联在一起，这种集体的功能就像用透镜得到图像的傅里叶变换一样，十分迅速，在人的日常生活中，每天都有成千上万的信息需要脑来处理，一个简单的动作如端一杯水，打一个电话，就牵涉到记忆、学习和相位变换等功能，而人却可以不加思索地完成，这就说明需要有一种新型的、类似于人脑结构的系统，来完成那些计算机做起来很慢或很困难的工作。

人工神经网络就是采用物理可实现的系统来模仿人脑神经细胞的结构和功能的系统。它是由很多处理单元有机地联接起来，进行并行的工作，它的处理单元十分简单，其工作则是"集体"进行的，它的信息传播，存贮方式与神经网络相似，它没有运算器、存贮器、控制器这些现代计算机的基本单元，而是相同的简单处理器的组合。它的信息是存贮在处理单元之间的连接上，因而，它是与现代计算机完全不同的系统。

本章主要介绍人工神经网络与生物体实际神经网络的比较、人工神经网络的特点、发展的历史及主要类型。

第一节 神经细胞及神经细胞组成的网络

一个神经细胞的构造如图 1-1 所示，主要包括细胞体、树突、轴突和细胞之间相互关联的突触。

细胞体是由细胞核、细胞浆、细胞膜等组成。在高等动物的神经细胞中，除了特殊的无"轴突"神经元外，一般每个神经元都由胞体的轴丘处发出一根粗细均匀、表面光滑的突起，

长度从几个 μm 到 1m 左右，称为轴突，它的功能是传出从细胞体来的神经信息。树突为细胞体向外伸出的很多其他突起，它们像树枝一样向四处分散开来，在胞体附近比轴突粗得多，但离开细胞体不远马上就很快分支变细，形成无数粗细不等的树突。它们的作用是向四方收集由其他神经细胞来的信息。信息流是从树突出发，经过细胞体，然后由轴突输出。突触是两个细胞之间连接的基本单元，主要有：

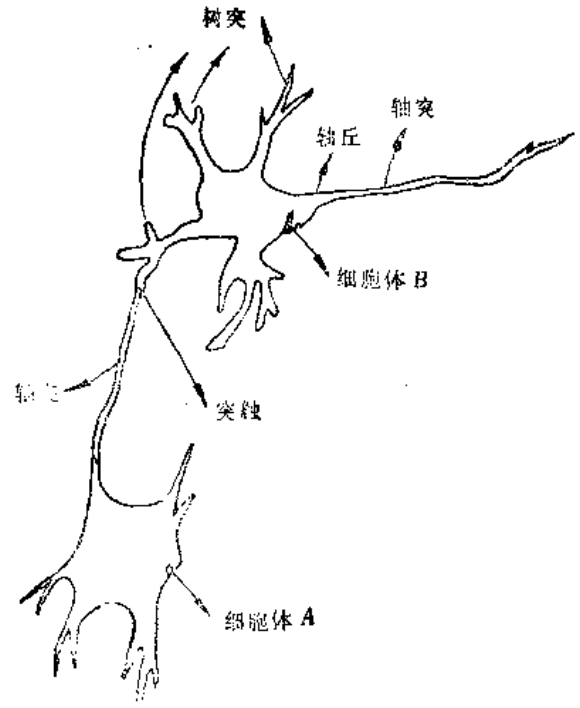


图 1-1　神经细胞的示意图

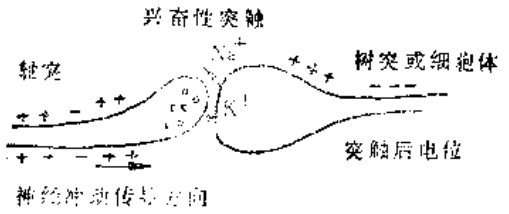


图 1-2　突触联接的示意图

(1) 一个神经细胞的轴突与另一个神经细胞的树突发生接触；

(2) 一个神经细胞的轴突与另一个神经细胞的胞体接触。

突触包含两个部分：

一个为突触前成分，表示在轴突的末梢；

一个是突触后成分，为树突的始端，或细胞体与轴突末端接触的部分。突触的联接如图 1-2 所示。

根据突触对下一个神经细胞的功能活动的影响，突触又可分为兴奋性的和抑制性的两种。兴奋性的突触：当神经细胞有冲动传递时，它能使突触后成分去极化，产生兴奋性突触后电位，可能引起下一个神经细胞兴奋。抑制性的突触能使突触后成分超极化，产生抑制突触后电位，使下一个神经细胞抑制。

神经细胞的种类很多，根据功能特性可以分为传入(感觉)神经细胞，中间(联络)神经细胞和传出(运动)神经细胞。

神经细胞单元的信息是宽度和幅度都相同的脉冲串。脉冲串的间隔是随机变化的，如某个神经细胞兴奋，其轴突输出的脉冲串在单位时间内的平均频率高(即发放率高)，如神经细胞抑制时，脉冲发放率减少，甚至无脉冲发放。每个神经细胞的输出脉冲发放率是与别的和它相联的神经细胞的发放情况及它们和该神经细胞突触联接情况有关。信息传递在突触处主要是发生化学和电的变化，(这里主要考虑化学变化)即在化学突触的传递过程中，突触前成分囊泡里的神经递质被释放到突触后膜，化学传递的突触传递神经冲动是通过释放神经递质来实现的，当脉冲到来时，贮存在突触囊泡内的神经递质进行排放，这样改变了突触后膜对钠离子(Na^+)、钾离子(K^+)和氯离子(Cl^-)的通透性，使突触后神经细胞相应发生电位变

化。突触传递信息需要一定的延迟时间，对温血动物，延迟时间为0.3～1.0ms。

兴奋性突触在脉冲刺激下，对下一个神经细胞产生兴奋性突触后的电位变化，抑制性突触在脉冲刺激下产生抑制性突触后的电位变化。很多神经细胞通过各自的突触对某一个神经细胞的作用，都形成该神经细胞的后电位变化，电位的变化是可以累加的，该神经细胞后电位是它所有的突触产生的电位总和，当该神经细胞的后电位升高到超过一个阈值，就会产生一个脉冲，从而总和的膜电位直接影响该神经细胞兴奋发放的脉冲数。一般说，每个神经细胞的轴突大约联接100～1000个其他神经细胞，神经细胞的信息就是这样从一个神经细胞传递到另一个神经细胞。

在人脑中大约有140亿个神经细胞单元，根据Stubbz的估计，这些神经细胞被安排进约1000个主要模块内，每个模块有上百个神经网络，每个网络约有10万个神经细胞，信息的传递是从一个神经细胞传到另一个神经细胞，从一种类型的神经细胞传到另一类神经细胞，从一个网络传到另一个网络，有时也从一个模块传到另一个模块。

对于化学突触的信息传递有以下几个原则：

(1) 只允许脉冲从突触前传向突触后，不允许逆向传递；

(2) 突触有延迟；

(3) 突触传递容易产生疲劳；

(4) 突触前传来的一次冲动常常不足以引起突触后神经细胞产生兴奋，同一个突触传来一系列脉冲，它们刺激间隔比较小，会引起突触后电位的变化，这就是时间上的总和；而很多突触前同时传来的脉冲也能引起突触后电位的变化，这是空间的总和；时间和空间的总和对突触后膜都会产生作用；

(5) 存在不应期，在不应期内，神经元对刺激不响应。

第二节　人工神经网络的特点

人工神经网络是采用物理可实现的器件或采用现有的计算机来模拟生物体中神经网络的某些结构与功能，并反过来用于工程或其他的领域。人工神经网络的着眼点不是用物理器件去完整地复制生物体中神经细胞网络，而是采纳其可利用的部分来克服目前计算机或其他系统不能解决的问题，如学习、识别、控制、专家系统等。随着生物和认知科学的发展，人们对人脑的结构及认知过程了解得越深入，这对人工神经网络的促进作用将会越大，越来越多的生物特性将被利用到工程中去。

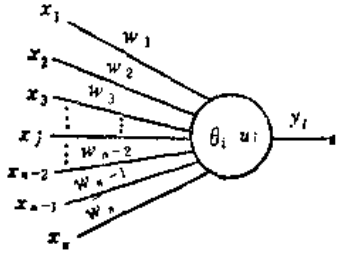


图1-3　人工神经元的示意图

图1-3是一个人工神经单元的示意图。图中$x_1,\cdots,x_n$表示其他神经元的轴突输出，$w_1,\cdots,w_n$为其他几个神经元与第i个神经元的突触联接，$w_1,\cdots,w_n$可以是正，也可以是负，分别表示为兴奋性突触和抑制性突触，数值的大小是根据突触不同的化学变化而各不相同。每个人工神经元满足：

$$S_i=\sum_{j=1}^{n} w_j x_i-\theta_i \tag{1-1}$$

$$u_i=g(s_i) \tag{1-2}$$

$$y_i=f(u_i) \tag{1-3}$$

其中式(1-1)表示神经元i突触后电位的累加值，θ_i为阈值。式(1-3)中，u_i为细胞i的状态，y_i为神经单元i的输出，$f(u_i)$是一个单调上升的函数，当u_i大时，y_i也大，但$f(u_i)$又是一个有限值函数，这是由于在生物体中神经细胞单元脉冲发放率有一个最大值，而不能永远无限上升的缘故。图1-3可以简单地表明上节所述神经细胞的基本功能，但是有以下几点不同：

(1) 在生物体内神经细胞中，来自其他几个神经细胞的轴突输出是发放脉冲串，用脉冲串在单位时间内的发放率来表示其兴奋活动情况，输出的脉冲串引起了突触化学成分的变化，然后引起第i个神经细胞突触后电位的变化，后电位的变化又引起y_i的脉冲发放率的变化，在人工神经网络中$x_1,x_2\cdots,x_n$和y_i大都用电压值来代替，y_i与u_i的关系也是电压与电压的关系。在人工神经网络的硬件实现时，也有人用数字脉冲频率的大小表示y_i与u_i，但与真正的神经网络还有较大的差距。

(2) 由于(1-1)式、(1-2)式和(1-3)式描写的方程，其信息的传递是采用电压的方式，因而大多数人工神经网络只有空间累加而没有时间累加。

(3) 在人工神经网络中，都没有考虑突触原则中的疲劳和不应期。

(4) 根据方程式(1-1)，由于x_j为电压，因而权w_j为导纳量纲，θ_i和s_i为电流量纲，u_i和y_i为电压量纲，这样，突触在人工神经网络中只要用一个电阻或相当于电阻的电子器件就可以实现。这种简单的权的表示与真正的生物突触中的化学反应和电反应也有很大的差距。

(5) 神经细胞的种类很多，但在一类人工神经网络中，神经元的种类通常仅为一种，少数有2～3种。

(6) 神经网络是由大量的神经细胞组成，而且神经细胞会不断死亡和增殖，在人工神经网络中，由于物理器件的限制，它的神经单元的数目大大小于真正神经网络中的细胞数，因而其功能也比真正神经网络差得多。这是两者最大的区别。当然，随着新器件的发展(如分子器件)会逐步缩小这个差距。

虽然人工神经网络与真正的神经网络有上述这些差别，但它与目前冯·诺伊曼计算机相比，由于它吸取了生物神经网络的部分优点，因而有其固有的特点：

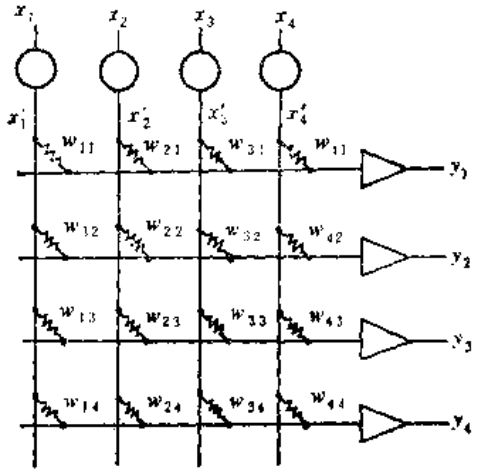


图1-4 简单的人工神经网络

(1) 人工神经网络在结构上与目前的计算机根本不同，它是由很多小的处理单元互相联接而成，每个处理单元的结构如图1-3所示，并由方程(1-1)、(1-2)、(1-3)来描述，每个单元的功能简单，但大量简单的处理单元集体的、并行的活动得到预期的识别、计算的结果，具有较快的速度。

(2) 人工神经网络具有很强的容错性，即局部的或部分的神经元损坏后，不影响全局的活动。

(3) 人工神经网络所记忆的信息是存贮在神经元之间的权中，从单个权中看不出其贮存信息的内容，因而是分布式的存贮方式。

(4) 人工神经网络具有十分强的学习功能，人工神经网络中的连接权和连接的结构都可以通过学习而得到。

现在，我们以图 1-4 所示的一个简单的人工神经网络的例子来说明上述的几个特点。图中，x_1, x_2, x_3, x_4 为输入，经感知细胞输出为 x_1', x_2', x_3', x_4'，然后经过突触权 w_{ij} 加到 y_1, y_2, y_3, y_4 的输入端，进行累加。为了简单起见，我们把式(1-1)，式(1-2)和式(1-3)的表达式改写为：

设：　$\theta_i=0$，则有

$$s_i=\sum_{j=1}^{n} w_{ij}x_j' \tag{1-1}'$$

$$u_i=s_i\cdot 1 \qquad \text{(量纲变换)} \tag{1-2}'$$

$$y_i=f(u_i)=\begin{cases} 1 & u_i\geqslant 0 \\ -1 & u_i<0 \end{cases} \tag{1-3}'$$

又设输入 x_j' $(j=1,\cdots,4)$ 也为二值变量(即 $x_j'=\pm 1$)，且：$x_j'=x_j$

x_j 是感知器的输入，我们用矢量 $x^1=[1,-1,-1,1]^T$ 表示眼看到花、鼻嗅到花香的感知输入，从 $x^1\to y^1$ 可通过一个联接矩阵 W_1 来得到。

$$W_1=\begin{bmatrix} -0.25 & +0.25 & +0.25 & -0.25 \\ -0.25 & +0.25 & +0.25 & -0.25 \\ +0.25 & -0.25 & -0.25 & +0.25 \\ +0.25 & -0.25 & -0.25 & +0.25 \end{bmatrix} \tag{1-4}$$

$$y^1=f(W_1x^1)$$

经计算：

$$y^1=[-1,-1,+1,+1]^T$$

这表示网络决策 x^1 为一朵花的概念。

我们看到从 $x^1\to y^1$ 的变化不是经过串型的计算得到的，因为矩阵 W_1 是一个可以用 VLSI 中的电阻矩阵来实现的，而 $y_i=f(v_i)$ 也可用一个简单的运算放大器来模拟，不管 x^1 和 y^1 的维数如何增加，整个计算都只用了一个运放的转换时间，网络的动作是并行的。

如果 $x^2=[-1,+1,-1,+1]^T$，表示眼看到炸猪排、鼻嗅到炸猪排的香味的感知器输入，通过矩阵

$$W_2=\begin{bmatrix} +0.25 & -0.25 & +0.25 & -0.25 \\ -0.25 & +0\ 25 & -0.25 & +0.25 \\ -0.25 & +0.25 & -0.25 & +0.25 \\ +0.25 & -0.25 & +0.25 & -0.25 \end{bmatrix} \tag{1-5}$$

得到 $y^2=[-1,+1,+1,-1]^T$，表示网络决策 x^2 为炸猪排。

从式(1-4)，(1-5)的权来看，我们并不知道其输出结果是什么，从局部权的分布看，也很难看出 W 中存贮什么，这是因为信息是分布式的存贮在权中，如果把(1-4)、(1-5)式相加，得到一组新的权：

$$W=W_1+W_2=\begin{bmatrix} 0 & 0 & 0.5 & -0.5 \\ -0.5 & 0.5 & 0 & 0 \\ 0 & 0 & -0.5 & 0.5 \\ 0.5 & -0.5 & 0 & 0 \end{bmatrix} \tag{1-6}$$

由 x^1 输入到这网络时，通过矩阵运算得到 y^1，由 x^2 输入得到 y^2，这说明 W 中存贮了两种信息，当然还可存贮多种信息，只要用多个已知的例子进行学习，在权中就可以存贮多种信息。

如果输入感知器中损坏了一个，设第三个损坏了，则 $x^1=[1,-1,0,1]^T$，经 W 算得 $y^1=[-1,-1,+1,+1]^T$，而 $x^2=[-1,+1,0,1]^T$ 经 W 算得 $y^2=[-1,+1,+1,-1]^T$，结果和前面的一样。这说明人工神经网络有一定的容错能力。

从上面的例子可以看出人工神经网络不像现在的计算机，有存贮单元、运算单元和控制单元，它的存贮单元和运算控制单元都融合在一个网络中，它的结构，工作步骤及方式完全与现代计算机不同，表 1-1 简单地列出了它们的差别，这里还要指出，人工神经网络与一般并行计算机或并行处理器的算法也不相同，因为并行机是多个 CPU 处理器并行工作，它的处理单元比人工神经元要复杂得多，它的算法和软件更比人工神经网络复杂得多。并行机只能解决计算机的处理速度问题，并没有上述的人工神经网络的特点和集体动作的功能。

表 1-1

	现代计算机	人工神经网络 (ANN)
元件间的连接	2～3	100～1000
实现结构	前馈式加很少反馈	前馈式 全反馈式
功　能	不能学习 串行，编程	能学习 并行、不编程或简单编程
记忆方式	存贮器 集中记忆	分布式记忆
容　错	元件损坏 无法工作	元件损坏能继续工作 或学习后继续工作
应　用	计算 逻辑判断 信息处理	识别感知 智能控制 专家系统等

第三节　人工神经网络的发展史

人工神经网络的发展可追溯到一个世纪前，根据文献[17]，可把这一历史分为四个时期。第一时期为初始发展期，自 1890 年至 1969 年；第二时期为低潮时期，自 1969 年至 1982 年；第三时期为复兴时期，自 1982 年至 1986 年；第四时期为发展高潮期，自 1986 年到现在。下面就每个时期的情况及有贡献的人物略作介绍。

一、初始发展期

1890 年，美国生理学家 W.James 出版了《生理学》[2] 一书，该书首次阐明了有关人脑结构及其功能，以及一些相关学习、联想记忆的基本规则，并指出：人脑中，当两个基本处理单元同时活动，或两个单元靠得比较近时，一个单元的兴奋会传递到另一个单元，而且一个单元的活动程度与它周围单元的活动数目和活动的密度是成正比的。这些论述有些是该书作者的推断，但在以后的发展中都证明这些推断是正确的。

大约经过半个世纪以后，McCulloch 和 Pitts 发表了十分有名的论文[5]。在这篇文章

中，他们用已知的神经细胞生物基础，描述了一个简单的人工神经元模型，这个神经元的活动是服从二值变化的，即兴奋和抑制，任何兴奋突触有输入激励后，使神经元兴奋，与过去神经元活动情况和神经元的位置无关，任何抑制突触被输入激励后，这个神经元即被抑制。突触的值是不变的，突触存在延迟时间为0.5ms。他们描述的模型后来被称为M-P模型。现在看来，这个模型是过于简单了，但他们的贡献在于：

(1) 证明了用M-P模型能完成任意有限的逻辑运算；

(2) 他们是在W. James后第一个采用集体并行计算结构来描述人工神经元和网络工作的；

(3) 他们的工作为以后进一步的研究提供了依据。

1949年Donala O.Hebb发表论著《行为自组织》[6]，首先定义了一种调整权的方法，称为Hebbian，他指出当一个A细胞的轴突，充分靠近细胞B，并持续不断地激励它，它们两个细胞的效应都增长，即B的活动增加了，A的活动也增加了。Hebb提出了很多有价值的观点，这对以后人工神经网络的结构和算法都有很大的影响，具体有以下几点：

(1) 他提出了在一个神经网络里信息是贮藏在突触联接的权中；

(2) 联接权的学习律是正比于两个被联神经细胞的活动状态值的乘积；

(3) 假设权是对称的，从A细胞到B细胞的权等于B细胞到A细胞的权；

(4) 细胞的互相联接的结构是它们权的改变创造出来的，例如一组弱联接的神经细胞同时活动，则细胞间的联系加强，形成强联接。直到现在，Hebb的学习算法仍在不少人工神经网络中应用。

1958年Frank Rosenblatt定义了一个神经网络结构[8]，称为感知器(perceptron)，这是第一个真正的人工神经网络，因为他在IBM704计算机上进行了模拟，从模拟结果看，perceptron是有能力通过调整权的学习达到正确分类的结果，因此它可以称为一个学习机。Rosenblatt用Perceptron来模拟一个生物视觉模型，输入是由随机的一组细胞组成，代表视网膜的一个小的范围，每个细胞又与下一层的神经细胞(称AU单元)相联，而AU又与第三层(RU)相联，RU为输出层，Perceptron的目的是通过学习，使对应的输入模板有正确的RU输出。初始的Perceptron学习机制是自组织的，因而响应的发生与初始的随机值有关，后来，他也加入一些教师进行训练，这些与后来在反向传输算法和Kohonen的自组织算法类似，因此Rosenblatt的思想有相当的活力。

1960年Bernard Widrow和Marcian Hoff发表了"自适应开关电路"的论文[9]，他们从工程的观点出发，不仅在计算机上模拟了这种神经网络，而且还做成了硬件，他们介绍的器件是一个称为"Adaline"的累加输出单元，输出的值是± 1的二值变量，权在Widrow和Hoff的文章中称为增益，他们主要提出了Widrow-Hoff算法，使增益的学习速度较快，而且还有较高的精度，后来这个算法被称为LMS算法，在数学上就是人们所知的速降法。Widrow-Hoff的算法不需要微分，其权的变化是正比于实际输出值与要求输出值之间的差和输入信号的符号，这种算法在以后反传网络(Back-Propagation)算法和其他信号处理系统中应用十分广泛。

二、低潮时期

1969年Marrin Minsky和Seymour Papert发表了名为"Perceptrons"的论著[7]，该书

分析了一些简单的单层 Perceptron，说明这些单层的 Perceptron 只能作线性划分，对于非线性或其他的分类会遇到很大的困难，Minsky 和 Papert 举了一个 XOR 逻辑分类的例子，说明用简单的单层 Perceptron 是不能正确分类的。Minsky 断言这种 Perceptron 无科学价值可言，包括多层的也没什么意义。这个结论对当时的人工神经网络的研究无疑是一沉重的打击。由于当时计算机的工具还不太发达，VLSI 尚未出现，人工神经网络的应用更没有展开，而人工智能和专家系统正处于发展的高潮，它们的问题和局限性尚未暴露，因此这个观点很快被不少人接受，很多领域的专家纷纷放弃了这个课题的研究。但在这个阶段，也还有不少科学家继续进行探索。

1972 年，两个研究者在不同的地方发表了他们的文章，他们是芬兰的 Helsinki 科技大学的 Teuro Kohonen 和美国 Brown 大学的 James Andorson。Kohonen 称神经网络的结构为"联想记忆"，Andorson 称他的网络为活动记忆，他们两人的网络在结构、学习算法和变换函数几乎相同。只是 Andorson 比较偏重于生理上的模型和学习方法，Kohonen 使用的网络的输入和输出都是连续值，[11] 它的权也是连续值，Kohonen 网络中输出神经元的数目大，代表分类的区域比较大，这使这个网络对噪声不太敏感，网络更有一般性，Kohonen 的学习方法是自组织的，不需要教师。

Boston 大学的 Stephen Grossberg，在此期间继续做了很多工作，他的工作似乎是更加集中在网络结构的生理背景。他和 Gail Carpenter 发展了一种自适应响理论 (ART)[10][12] 他提出了一些概念，包括一个兴奋神经元周围的其他神经元被强烈抑制等，都有一定的价值。他还提出了短期记忆和长期记忆的机制，形成了活动值和联接权之间的关系，对短期记忆，它们会随着时间的推移而被遗忘，而长期记忆被遗忘所需的时间更加长些，ART 网络的算法也是自组织的。后来，他们又发展了 ART_1 和 ART_2，分别针对二进制输入和连续输入的情况。

这个时期另一个研究者是东京 NHK 广播科学研究实验室大阪大学教授 Kunihiko Fukushima 博士，他提出了一个神经网络的结构称为"Neocognitron"，发表于 1980 年[13]。这个 Neocognitron 是一个视觉识别机制，它与生物视觉理论相符合，Fukushima 的目的是为了设计一个模型，它能像人一样进行模式识别，它是自组织的结构，不需要教师的学习。该模型主要针对图像处理，因而每一层都是二维的，它的特点是能够显示感兴趣的特征，这个网络对输入图象的位置和微小的变化都不太敏感，甚至对于几个字母组合的复杂图形也有能力在一个时间只选择其中一个字母进行识别，这样，就可以按次序对不同字母进行识别。Neocognitron 本身没有能应用到较广的程度，因为模型太复杂，根据 Fukushima 1980 年的文章，输入层为 256 个神经元(16×16)，而三个前馈网络分别为 8544，2400 和 1 个神经元，总计约有 11000个神经元，加上前馈和反馈的联接，因此只能在大型机上才能模拟，如果形成硬件，其规模也很大。但 Fukushima 提出的内部神经网络学习和计算的过程，他的网络每一层计算都给出了详细的公式，使这个识别过程是通过一层一层网络，一步一步计算进行的，而且模型又有一定的生物依据，他用这个网络完成了十个数字和英文字母的识别的例子，在一定程度上可以说明一种认知的过程。

三、复兴时期

在整个低潮时期，很多研究工作是一些生物学家、生理学家和其他研究者进行的，但在

1982年，加州技术学院的优秀物理学家John J.Hopfied博士发表了一篇十分重要的文章[14]，他所提出的全联接网络后来被称为Hopfield网络，在网络的理论分析和综合上达到了相当的深度，最有意义的是他的网络很容易用集成电路来实现，在1984年、1986年Hopfield连续发表了有关他的网络应用的文章[34][38][40][41]，他的文章得到了一些工程技术人员和计算机科学家的重视和理解。一个十分重要的思想是：对于一个给定的神经网络，他提出一个能量函数，这个能量函数是正比于每个神经元的活动值和神经元之间的联接权，而活动值的改变算法是向能量函数减少的方向进行，一直达到一个极小值为止。他证明了在一定条件下网络可到达稳定状态，他不仅讨论了离散的输出情况，而且还讨论连续变化时的情况，从而可以解出一些联想记忆问题和计算优化问题。Hopfield网络有比较完整的理论基础，他利用了物理中自旋玻璃子里的哈密顿算子，把这一思想推广到神经网络中来，虽然思想上并不新颖，但他的神经网络在设计和应用上起着不可估量的作用。继Hopfield的文章之后，不少搞非线性电路的科学家、物理学家、生物学家在理论和应用上对Hopfield网络进行了比较深刻的讨论和改进。Hopfield网络也引起了半导体工业界的重视，因为这个时期VLSI已经达到相当高的水平，从而有可能设计出一种神经网络芯片来，在1984年，Hopfield的文章发表三年以后，AT&T Bell实验室宣布第一个硬件人工神经网络芯片是在利用Hopfield理论的基础上实现的。后来加州技术学院的Carver meal继续从事芯片的研究工作，在耳蜗和视网膜的芯片上都得到比较好的成果。当然，在深入研究中也发现原始的Hopfield网络的一些问题，如网络的权是固定的，不能学习，在规模大的情况下实现困难，而且存在局部极小无法克服等等。但不可否认，Hopfield博士点亮了人工神经网络复兴的火把，掀起了各学科关心神经网络的一个热潮。

在这个时期，还有一个并行分布处理的研究小组，即Parallel Distributed Processing，简称PDP[1]，他们在1986年发表了二卷PDP的论著，共十六章。在1988年发表了带有软件程序的第三卷，这本书的主要作者是由David E. Rumelhart以及James L. McClelland和一个十多个人组成的小组，虽然这个小组在开始时很难确定这本书的方向，但在书中涉及到人工神经网络的三个主要属性：模型的结构，神经元的输入，输出变换函数以及算法，其中有些章节对以后的研究产生很大影响，例如第八章"误差反传的学习内部描述"和第七章"Boltzmann机"这本书对神经网络在各领域的推广起了很大作用。

四、发展高潮期

1987年，美国召开了第一届国际神经网络会议，当时参加的人数就在一千人以上，而且涉及到生物、电子、计算机、物理、控制、信号处理、人工智能等各个领域。这一年美国国防部对人工神经网络的研究作了调查，发表了一个Darpa报告，在报告中列出了很多应用的方向与实例。此后国际神经网络协会成立。在这个时期，人工神经网络的研究已经从美国推广到西欧、日本和我国等，同时各类的模型和算法纷纷出台，其中比较有名的是CNN网络，这是L. O. Chua[16]等人在Hopfield网络的基础上发展的局部联接网络，这种网络在视觉初级加工上得到广泛的应用。更为突出的是，这个时期在应用方面已达到一定的深度和广度，牵涉到20～30个方面。芯片也逐步出现，有的已形成产品。当然随着人工神经网络应用的深入，人们也发现原有的模型和算法所存在的问题，在理论的深入也碰到很多原来非线性理论、逼近论中的难点，可是人们相信，在深入、广泛应用的基础上，这个领域将会继续发展，并会对科

学技术有很大的促进作用。

第四节　人工神经网络的类型

人工神经网络的模型和算法种类很多，这里我们从神经元输入输出关系，网络结构和算法上归纳如下。

一、神经元变换函数的类型

神经元的关系满足方程(1-1)、(1-2)、(1-3)，神经元的状态 u_i 和输出 y_i 的关系是一个非线性的关系，一般有下面几种，如图 1-5 所示。

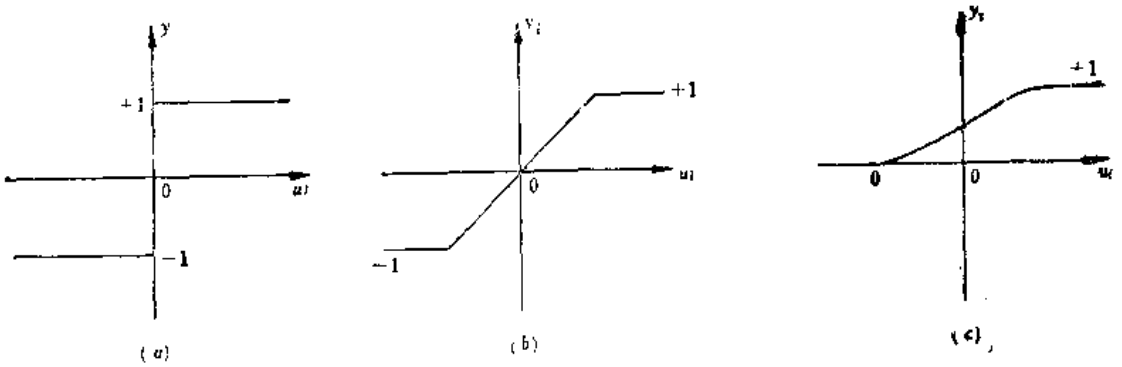


图 1-5　神经元的输入输出关系

(a) 硬函数；　(b) 分段线性函数；　(c) 非线性单调上升函数。

图 1-5a 表示神经元的兴奋活动输出值 y_i 是一个二值变量，如果在方程(1-1)中 x_j 也是二值变量，那么这种神经网络为离散型的网络，神经元的输出可以是 ±1，也可以为 0、1 二值。这种关系为硬函数，例如 Perceptron 模型、M-P 模型以及 Hopfield 在 1982 年所提出的模型，都是属于二值变量的硬函数。图 1-5b 表示输入输出关系是分段线性关系，在 CNN 网络中运用的就是这种模型。当然其值也可以从 0 到 1。图 1-5c 是一个单调上升的光滑的非线性函数，可以用公式(1-7)或(1-8)来表示：

$$y_i=f(u_i)=\frac{1}{1+e^{-u_i}} \tag{1-7}$$

$$y_i=f(u_i)=\frac{1}{2}\left(1+\text{th}\frac{u_i}{2}\right) \tag{1-8}$$

在 Fukushima 模型和 Back-Propagation 的模型中都是用这一类输入输出变换函数。

在方程(1-2)中，$u_i=g(s_i)$，在多数人工神经网络中，g 函数为一个线性函数，有的简化为 $u_i=s_i$，但有些模型中 s_i 与 u_i 满足一个状态方程。

二、人工神经网络的结构

1. 前馈式网络

前馈网络中神经元是分层排列的，每个神经元只与前一层的神经元相连，如图 1-6a 所示。最上一层为输出层，最下一层为输入层，还有中间层，中间层也称为隐层。隐层的层数也

可以是一层或多层。

2. 输入输出有反馈的前馈网络

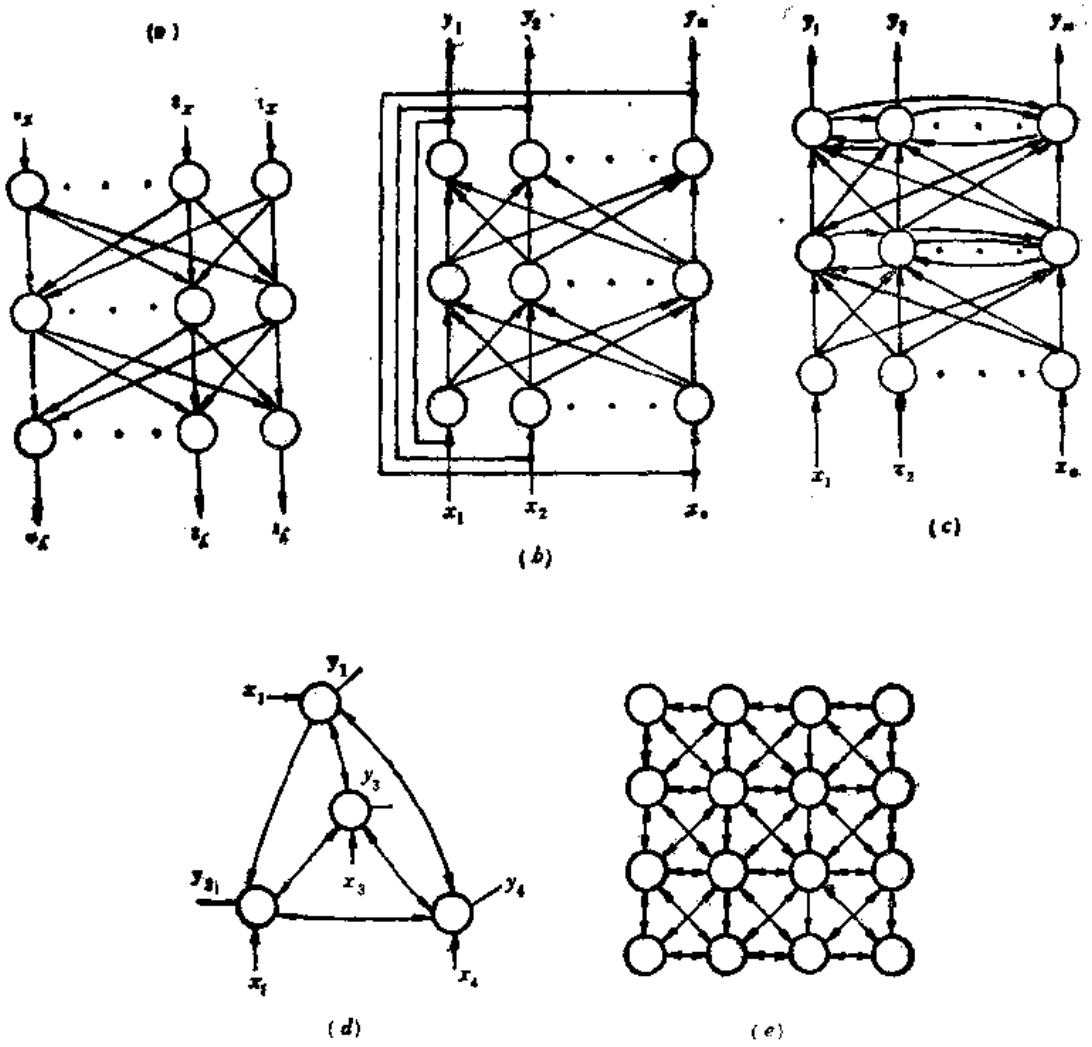


图 1-8 网络的结构

(a) 单纯前馈网络； (b) 有反馈的前馈网络； (c) 内层互联的前馈网络；
(d) 反馈型全互联网络； (e) 反馈型局部联接网络

图 1-6b 所示，在输出层上存在一个反馈回路到输入层，而网络本身还是前馈型的，如 Fukushima 的网络就是用这种反馈的方式达到对复杂图形的顺序选择和识别字符的。

3. 前馈内层互联网络

图 1-6c 所示的网络，在同一层内存在互相联接，它们可以形成互相制约，而从外部看还是一个前向网络，很多自组织网络，大都存在着内层互联的结构。

4. 反馈型全互联网络

图 1-6d 所示是一种单层全互联网络，每个神经元的输出都与其他神经元相联，如 Hopfield 网络和 Boltzmann 机都是属于这一类网络。

5. 反馈型局部联接网络

图 1-6e 所示是一种单层网络，它的每个神经元的输出只与其周围的神经元相联，形成反馈的网络，这类网络也可发展为多层的金字塔形的结构。

反馈型网络存在着一个稳定性问题，因而必须讨论其收敛性和稳定性的条件。

三、学习算法上的分类

在人工神经网络中，权是一个反应信息存贮的关键量，在结构和转换函数定了以后，如何设计权使网络达到一定的要求，这是人工神经网络必不可少的部分，大多数神经网络权的设计是通过学习得到的，目前可分为下列几种。

1. 死记式学习

网络的权是事先设计的，值是固定的。如 Hopfield 网络在做优化时，权可根据优化的目标函数和约束条件来设计，一旦设计好了就不能变动。早期的 M-P 模型也是用设计好的固定权来完成与、或、非等逻辑关系的。

2. δ 学习律

这种方法是用已知例子作为教师对网络的权进行学习，称为 δ 学习律。

设 $(\boldsymbol{x}^i,\boldsymbol{y}^i)i=1,\cdots,P$ 为已知的输入、输出例子，$\boldsymbol{x}^i,\boldsymbol{y}^i$ 为 n 和 m 维矢量，$\boldsymbol{y}^i=(y_1,y_2,\cdots,y_m)^T$ $\boldsymbol{x}^i=(x_1,x_2,\cdots,x_n)^T$，把 $\boldsymbol{x}^i$ 作为神经网络的输入，在权的作用下，可计算出实际神经网络的输出为 $\boldsymbol{y}^{i\prime}=(y_1',y_2'\cdots,y_j',\cdots,y_m')^T$，设任一个输入神经元 q 到 y_j' 的权为 w_{qj} 则其权的改变量为

$$\Delta w_{qj}=\eta\delta_j V_q \tag{1-9}$$

$$\delta_j=F(y_j-y_j') \tag{1-10}$$

其中 η 为步长，(y_j-y_j') 为误差(即要求值与实际值之差)，V_q 为第 q 个神经元的输出，$F(\cdot)$ 函数是根据不同的情况而定，多数人工神经网络 $\delta_j=(y_j-y_j')$；$F(x)=x$。

在很多神经网络中，都采用了这种 δ 学习律，如 Perceptron Adaline 和 Back-Propagation 算法等。

3. 自组织的学习和 Hebbian 学习律

两个神经元之间的连接权，正比于两个神经元的活动值，如 V_i,V_j 表示两个神经元的输出值，则他们之间的权的变化为

$$\Delta w_{ij}=\eta V_i V_j \tag{1-11}$$

这里 η 为步长或常数。

4. 相近学习

设 w_{ij} 为从神经元 i 到神经元 j 的权，V_i 为 i 神经元的输出，则

$$\Delta w_{ij}=\alpha(V_i-w_{ij}) \tag{1-12}$$

在这个学习中，使 w_{ij} 十分逼近 V_i 的值。如 Kohonen 和 ART 等都采用这类学习方法。

下面的几章，我们将详细讨论各类模型、算法和它们的应用。我们根据其结构讨论前馈式和反馈式的网络。自组织网络作为一章，其他类的网络也作为一章，每一章都讨论一些可以实用的例子，以使读者理解网络及其设计方法。最后给出硬件实现的几种方法。

第二章　前馈型人工神经网络

前馈型网络结构上采用的是其信息只能从输入层单元到它上面一层的单元，结构是分层的，第一层的单元与第二层所有的单元相联，第二层又与其上一层所有的单元相联，如图1-6*a*所示。在前馈网络中的神经单元其输入与输出关系，可采用线性阈值硬变换或单调上升的非线性变换，它们的连接权的算法用的都是有教师的δ学习律，但由于神经元输入输出变换函数的差异，学习律及某些结构上的区别分为各类不同的模型。

第一节　线性阈值单元组成的前馈网络

在上一章中，公式(1-1)、(1-2)、(1-3)表示神经单元输入输出的基本关系，对于第 j 个神经元，有

$$s_j=\sum_{i=1}^{n}w_ix_i-\theta_i$$

$$u_j=g(s_j)$$

$$y_j=f(u_j)$$

如令 $u_j=s_j$，$f(u_j)=\mathrm{sgn}(u_j)$，就形成了线性阈值单元，对于变换函数 $f(\cdot)$ 也可定义为

$$y_j=f(u_j)=\begin{cases}1 & u_j\geqslant 0\\ 0 & u_j<0\end{cases}\tag{2-1}$$

线性阈值单元的输入为一个 n 维的实数矢量 $x\in R^n$，其权 w 也是一个 n 维的矢量 $w\in R^n$，它们可为任何实数。而输出 y_j 必须是一个二值变量，但从输入与输出关系看，它们是满足线性关系 $u_j=\Sigma w_ix_i-\theta_j$，$y_j=\mathrm{sgn}(u_j)$。现对下面几类网络进行讨论。

一、M-P模型

M-P模型最初是由McCulloch和Pitts提出的，它是由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，如抑制性突触被激活，则神经元被抑制，输出为零。而兴奋性突触的数目比较多，兴奋性突触能否激活，则要看它的累加值是否大于一个阈

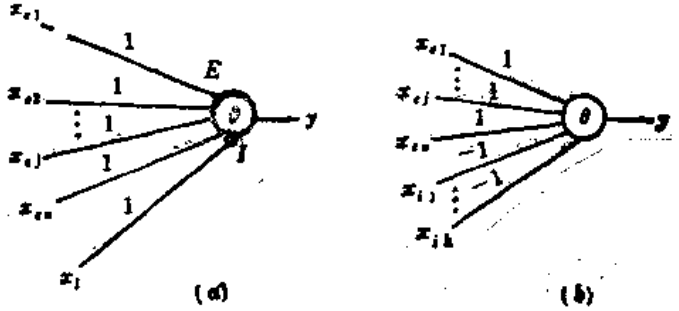


图 2-1　M-P模型中单个神经元示意图
(a) 早期模型。　(b) 具有±1突触的模型

值，大于该阈值神经元即兴奋。它的结构如图 2-1 所示，它的变换关系为

$$E=\sum_j x_{ej},\quad I=\sum_k x_{ik}$$

x_{ij} 为兴奋性突触的输入，$j=1,2,\cdots,n$，x_{ik} 为抑制突触的输入，$k=1,\cdots,n_1$，则输入与输出的转换关系为

$$\begin{cases} I=0 & E\geqslant\theta & y=1 \\ I=0 & E<\theta & y=0 \\ I>0 & & y=0 \end{cases} \tag{2-2}$$

M-P 模型中的权 $w_i=1$，这种模型是早期提出的，它可以用来完成一些逻辑关系。图 2-2a、b、c、d 分别表示与、或、非和一个逻辑关系式。如果兴奋与抑制突触用权 ±1 表示，而总的作用用输入加权的办法实现，兴奋为 1，抑制为 -1，则有

$$y=\begin{cases} 1 & \sum_j x_{ej}-\sum_k x_{ik}\geqslant\theta \\ 0 & \sum_j x_{ej}-\sum_i x_{ik}<\theta \end{cases} \tag{2-3}$$

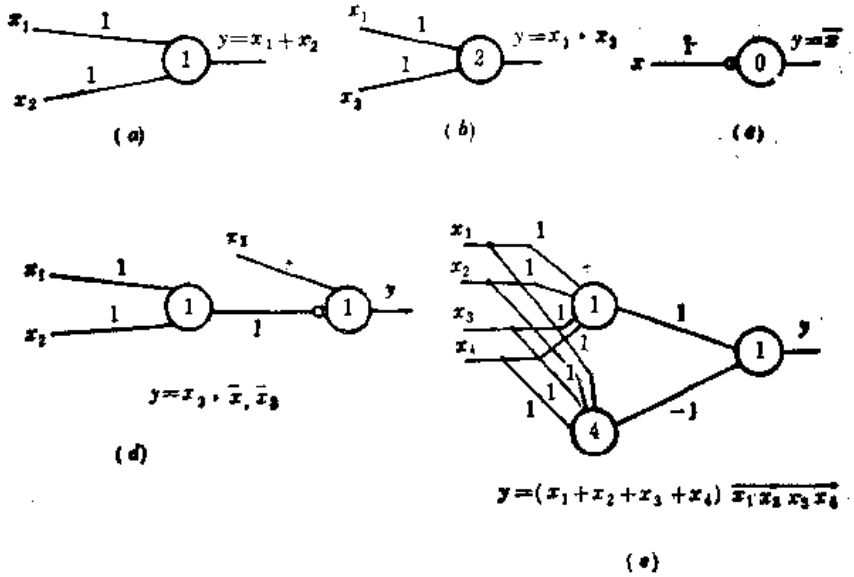


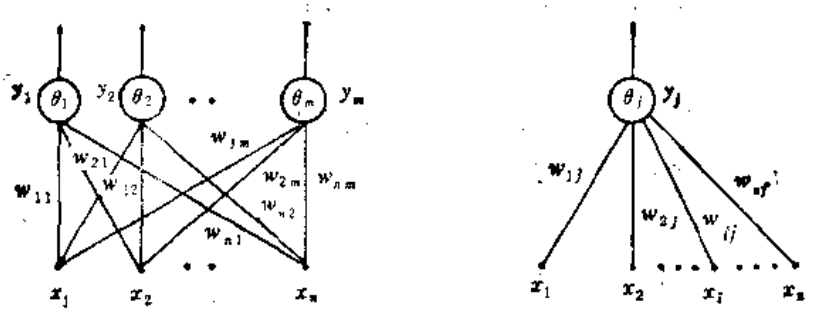
图 2-2　用 M-P 模型完成的布尔逻辑

图 2-1b 表示这种形式。图 2-2e 为这种形式组成的逻辑关系。

M-P 网络的权、输入、输出都是二值变量，这同由逻辑门组成的逻辑式的实现区别不大，又由于它的权无法调节，因而现在很少有人单独使用。

二、感知器组合的神经网络

感知器(perceptron)较 M-P 模型又进了一步，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。感知器可以对输入的样本矢量进行分类，而且多层的感知器，在某些样本点上对函数进行逼近，虽然它的分类和逼近在容差上和精度上都及不上以后讨论的 B-P 网络，但感知器是一个线性阈值单元组成的网络，在结构和算法上都成为其他前馈网络的基础，尤其它对隐单元的选取比其他非线性阈值单元组成的网络容易分析，而从感知器的讨论，可以对其他网络的分析提供依据。



(a)　　(b)

图 2-3

(a) 单层的感知器网络;　　(b) 单个神经元

(一) 单层的感知器网络

图 2-3a 所示的是一个单层的感知器网络，输入 $\boldsymbol{x}\in R^n$，$\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^T$每个输入节点 i 与输出层单元 y_j 的联接权为 w_{ij}，$i=1,\cdots,n$；$j=1,\cdots,m$，对于一个输出单元 j，它与所有 n 个输入单元的联接权为 $\boldsymbol{w}_j=(w_{1j},w_{2j},\cdots,w_{nj})^T$，也为一个 n 维空间的矢量，由于对不同的输出单元其联接权是独立的，因而可以把一个输出单元抽出来讨论。对第 j 个输出其转换函数为

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

$$f(u_j)=\begin{cases}1 & u_j\geqslant 0\\ -1 & u_j<0\end{cases} \qquad (2\text{-}4)$$

如果输入 $\boldsymbol{x}$ 有 k 个样本，即$\boldsymbol{x}^p$，$P=1,2,\cdots,k$，$\boldsymbol{x}\in R^n$，当将这些样本分别输入到单输出的感知器中，在一定的 w_{ij} 和 θ_j 下，输出 y_j 有两种可能，即 $+1$ 或 -1，把样本 $\boldsymbol{x}^p$ 看作为在 n 维状态空间中的一个矢量，那么 k 个样本为输入空间中的 k 个矢量，而方程(2-4)就是把这个 n 维输入空间分为 s_A，s_B 两个子空间，其分界线为 $n-1$ 维的超越平面，即用一个单输出的感知器，通过调节参数 w_{ij}，$i=1,\cdots,n$，以及 θ_j 来达到对 k 个样本的正确划分。如 $\boldsymbol{x}^A\in s_A$，$\boldsymbol{x}^B\in s_B(A=1,\cdots,l,B=l+1,\cdots,k,)$那么通过网络图 2-3b，使 $\boldsymbol{x}^A\to y_j=1$，$\boldsymbol{x}^B\to y_j=-1$，换句话说，如果存在一组权参数 w_{ij}，$i=1,2,\cdots,n$，和 θ_j 使公式 (2-4) 满足 $\boldsymbol{x}\in s_A$，$y_j=1$，$\boldsymbol{x}\in s^B$，$y_j=-1$，则称样本集为线性可分的，否则称为线性不可分的。

以一个两维输入空间为例，如图 2-4 所示，输入矢量 $\boldsymbol{x}=(x_1,x_2)^T$，权矢量 $\boldsymbol{w}=(w_1,w_2)^T$，则输出 $y=f(w_1x_1+w_2x_2-\theta)$，在两维的输入空间中用"O"代表 A 类样本，集中在平面的左上角上，用"·"表示 B 类样本，集中在平面的右下角，我们希望找到一根直线，把 A、B 两类样本分开，其分界线为

$$x_2=-\frac{w_1}{w_2}x_1+\frac{\theta}{w_2} \qquad (2\text{-}5)$$

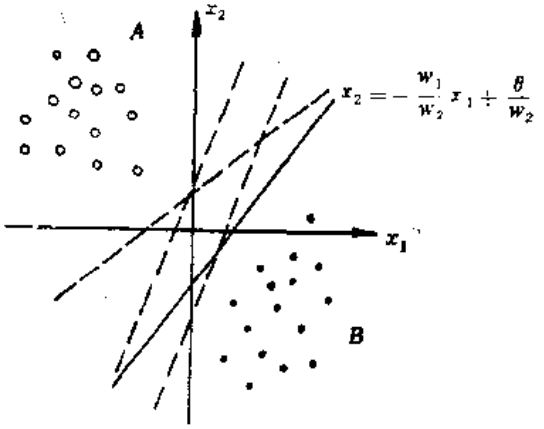


图 2-4　二维输入感知器及在状态空间中的划分

从图 2-4 中可以看出，如果 A、B 两类样本是线性可分的，而且如图那样有一段距离，那

么式(2-5)的解有无数个，如图 2-4 中虚线所示。对于 w_1、w_2、θ 的选取是根据下面的算法进行。

令

$$\boldsymbol{w}=(w_1,\cdots,w_{n+1})^T$$

其中 $w_{n+1}=\theta$，则在公式(2-4)中：

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

其中 $x_{n+1}=-1$。问题是已知样本集 $\boldsymbol{x}^A$，$\boldsymbol{x}^B$，分别属于 s_A 与 s_B 两类子空间，要求 $w_i, i=1,2,\cdots,n+1$，使

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

$$y=f\left(\sum_{i=1}^{n+1}w_ix_i^B\right)=-1$$

设存在一个教师 t_q，让 $\boldsymbol{x}^A$ 输入网络时 $t_q=1$，$\boldsymbol{x}^B$ 输入时 $t_q=-1$。那么整个学习的步骤如下：

(1) 网络中的初始化权和阈值，令

$$w_i(0)=a\,\mathrm{ran\,dom}(\cdot)\quad \forall i=1,\cdots,n+1$$

随机数前面的系数 a 的取值为不等于零的正小数，使初始化权比较小。

(2) 在 $\boldsymbol{x}^A$ 和 $\boldsymbol{x}^B$ 中，任选一个作为输入矢量，如选 $\boldsymbol{x}^A$ 中 $\boldsymbol{x}^l$，第 l 个样本作为输入，计算实际输出为

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

y^l 表示第 l 个样本输入后感知器的输出。

(3) 若 $y^l=1$，则权和阈值不需要调整，回到(2)重新选一个样本，如果 $y^l=-1$，此时教师为 $t_{q^A}=1$，就进行下一步。

(4) 调节权和阈值量，按下式进行，设 n_0 为迭代次数

$$w_i(n_0+1)=w_i(n_0)+2\eta[t_{q^A}-y^l_{(n_0)}]x_i^l(n_0) \tag{2-6}$$

(5) 重新在 $\boldsymbol{x}^A$ 或 $\boldsymbol{x}^B$ 集中选取另一个样本进行学习，即重复(2)、(3)、(4)、(5)一直进行到 $w_i(n_0+1)=w_i(n_0)$ 为止，对所有的 $i=1,\cdots,n+1$ 成立，则其学习结束。

在公式(2-6)中 η 为步长，$[t_q-\boldsymbol{y}(n_0)]$ 为误差，为了说明误差的情况，我们用 $e_q(n_0)$ 表示第 n_0 次迭代的误差值：

$$e_q(n_0)=t_q-y_q(n_0) \tag{2-7}$$

其中 q 表示为量化的结果，t_q 只可能为 ±1，$y_q^{(n_0)}$ 我们可定义为 $\mathrm{sgn}\left(\sum_{i=1}^{n+1}w_ix_i\right)$，因此 $y_q(n_0)$ 也为 ±1，因而 $e_q(n_0)$ 只可能为 $\pm2,0$。学习时样本的选取最好在 s_A 与 s_B 两个部分中轮流进行，以防止在权调整中的不均匀。关于学习样本顺序的选择，对学习的结果和收敛速度是有一定影响的，已有一些文章作过讨论，这里不赘述了。

(二) 单层感知器网络学习算法的讨论

公式(2-6)是一个按 δ 学习律调整权的公式，它是在梯度法的基础上，采用 LMS 算法而得到的。如果我们首先考虑 t_q 和 $y_q(n_0)$ 为两个非量化的量，用 t 表示学习的教师值，$y(n_0)$ 就表示为输入 x_i 的加权线性叠加，即

$$y(n_0)=\sum_{i=1}^{n+1} w_i x_i$$

这样可得到一个非量化的误差：

$$e(n_0)=t-y(n_0),\quad e^2(n_0)=[t-y(n_0)]^2$$

根据 LMS 的算法，权的修改值是向误差梯度的反方向进行，如此可得

$$\boldsymbol{w}(n_0+1)=\boldsymbol{w}(n_0)-\eta\nabla(n_0) \tag{2-8}$$

这里 η 为修改步长，$\nabla(n_0)$为 $e^2(n_0)$的梯度：

$$\nabla(n_0)=\frac{\partial e^2(n_0)}{\partial \boldsymbol{w}(n_0)}$$

因为

$$y(n_0)=\boldsymbol{w}^T(n_0)\boldsymbol{x}(n_0)$$

所以

$$\nabla(n_0)=-2e(n_0)\boldsymbol{x}(n_0) \tag{2-9}$$

继而可得

$$\boldsymbol{w}(n_0+1)=\boldsymbol{w}(n_0)+2\eta e(n_0)\boldsymbol{x}(n_0) \tag{2-10}$$

对于非量化的输出函数 $y(n_0)$，与 $w_i, x_i(i=1,\cdots,n+1)$是线性关系，因此 $e^2(n_0)$是有关 w_i 的椭圆函数，LMS 算法证明，这种椭圆函数只有一个极小值，因而用 LMS 的修改方法可以使 $e^2(n_0)$逐步达到该极小值。

现在我们回到量化的情况来讨论。$e_q(n_0)$是两个离散量之差，但我们仍可把它作为一个误差函数的梯度来讨论。

假设一个新的误差函数：

$$\xi(n_0)=2|y(n_0)|-2t_q y(n_0) \tag{2-11}$$

这个式子不失一般性，因为当 $y(n_0)=t_q$ 时，$\xi(n_0)=0$，当 $y(n_0)\neq t_q$ 时，$\xi(n_0)=4|y(n_0)|$；在 $y(n_0)=\pm 1$ 时与 $e_q^2(n_0)$相同。我们对 $\xi(n_0)$进行梯度运算得到：

$$\nabla(n_0)=2\mathrm{Sgn}[y(n_0)]\boldsymbol{x}(n_0)-2t_q\boldsymbol{x}(n_0)=-2e_q(n_0)\boldsymbol{x}(n_0) \tag{2-12}$$

将(2-12)式代入(2-8)式即可得到(2-6)式。

这里还有两个问题需要讨论：

1. 关于算法所求出的解

假设输入 $\boldsymbol{x}\in R^n$ 是一个高斯分布的随机向量，其均值为 0，相关矩阵 $\boldsymbol{R}=E[\boldsymbol{x}(n_0)\boldsymbol{x}^T(n_0)]$，$E$ 表示为求高斯分布的期望值，其结果 $y(n_0)$也是一个均值为 0 的高斯随机过程，y 的方差 $\sigma_y^2=\boldsymbol{w}^T\boldsymbol{R}\boldsymbol{w}$。

将 $|y(n_0)|=\mathrm{sgn}[y(n_0)]y(n_0)$代入(2-11)式得

$$\xi(n_0)=2y_q(n_0)y(n_0)-2t_q y(n_0)=-2e_q(n_0)y(n_0) \tag{2-13}$$

对(2-13)式中 $y_q(n_0)y(n_0)$求期望值，得

$$E[y_q(n_0)y(n_0)]=E[y^2(n_0)]/C\sigma_y=\sigma_y/C \tag{2-14}$$

其中 $C=\sqrt{\frac{\pi}{2}}$

(2-13)式中，$t_q y(n_0)=t_q\boldsymbol{w}^T\boldsymbol{x}(n_0)$

令

$$\boldsymbol{P}_q=E[\boldsymbol{x}(n_0)t_q]$$

在(2-13)式中，对误差函数 $\xi(n)$求期望值得

$$E[\xi(n_0)]=-2E[e_q(n_0)y(n_0)]=2\left(\frac{\sigma_y}{C}-\boldsymbol{w}^T\boldsymbol{P}_q\right) \tag{2-15}$$

对(2-15)式求梯度得

$$\nabla\xi=2\left(\frac{\boldsymbol{R}\boldsymbol{w}}{C\sigma_y}-\boldsymbol{P}_q\right) \tag{2-16}$$

当 $\nabla\xi=0$，表示对于 $\boldsymbol{w}$ 不再进行调整，则可得

$$\boldsymbol{w}=C\sigma_y\boldsymbol{R}^{-1}\boldsymbol{P}_q \tag{2-17}$$

对于一组随机变量作为输入，只要 $\boldsymbol{R}^{-1}$ 存在，$\boldsymbol{P}_q\neq 0$，那么 $\boldsymbol{w}$ 的解是存在的，但是 $\boldsymbol{w}$ 的解，并不一定是唯一的，因为对于不同的 σ_y 可能有不同的 $\boldsymbol{w}$ 解，从图 2-4 二维输入的划分上可以看到这个结论是正确的。

2．关于学习算法的收敛性

我们将(2-6)式写成矢量的形式：

$$\boldsymbol{w}(n_0+1)=\boldsymbol{w}(n_0)+2\eta e_q(n_0)\boldsymbol{x}(n_0) \tag{2-18}$$

其中 $e_q(n_0)$只可能为 0，±2，若 $e_q(n_0)=0$ 表示公式中的权不进行修改，而当 $e_q(n_0)\neq 0$，则 $\boldsymbol{w}(n_0+1)$将会被修改，但其修改的方向和矢量 $\boldsymbol{x}(n_0)$同方向，或者反方向，我们可用图 2-5a,b 来说明收敛性，假设输入矢量 $\boldsymbol{x}(n_0)$为某个固定的样本，图中用 $\boldsymbol{x}$ 表示。在阴影区内表示 $\boldsymbol{w}(n_0)$ 已调到正确的位置上。图 2-5a 中，$y_q<0$，$t_q>0$；矢量 $\boldsymbol{x}$ 和 $\boldsymbol{w}(n_0)$之间的夹角为钝角，$\boldsymbol{w}(n_0)\cdot\boldsymbol{x}<0$，而教师 $t_q>0$，因而 $e_q=+2$，在 η 很小的情况下，$\Delta\boldsymbol{w}(n_0)$在与 $\boldsymbol{x}$ 平行的方向上，通过矢量相加使$\boldsymbol{w}(n_0+1)$的方向朝 $\boldsymbol{x}$ 矢量靠近，一旦达到阴影区就满足 $e_q=0$ 的条件，同样在图 2-5b 则是 $e_q=-2$ 的情况，$\Delta\boldsymbol{w}(n_0)$在平行于 $\boldsymbol{x}$ 的反方向上，使 $\boldsymbol{w}(n_0+1)$向阴影区转动，一旦达到阴影区，其 $e_q=0$。对于不同的输入样本 $\boldsymbol{x}^P(P=1,\cdots,k)$，每一个都存在着一个权矢量应达到的阴影区，这些阴影区的交集就是满足 k 个输入样本正确划分的权矢量区域，如这个交集大，则得到的解的可能性就多，学习就比较容易，如没有交集，或为空集，则 k 个样本为线性不可分，即不能简单地用直线来划分。从以上分析可以看出，权矢量的方向是很重要的，而其大小只与矢量合成的幅度有关，如 $\boldsymbol{w}^*$ 是其权的最后解，只要它与输入样本矢量的夹角正确，而它的大小是无所谓的，$\alpha\boldsymbol{w}^*$ 也是其解，α 为一任意常数。

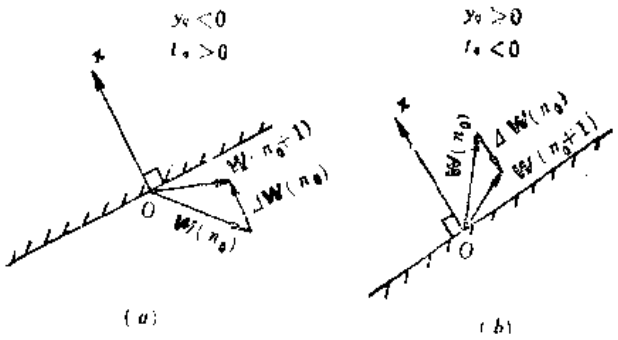


图 2-5　说明收敛的示意图

收敛性定理：若训练样本 $\boldsymbol{x}^P,P=1,\cdots,k$，是线性可分的，那么用公式(2-18)的 δ 学习律，在有限次的迭代后，必能达到所要求的正确解。

为了证明这个定理，我们作如下几个假设：

(1) 输入的样本 $\boldsymbol{x}^1,\cdots,\boldsymbol{x}^k$ 都是归一的，即 $\|\boldsymbol{x}^P\|=1$。

(2) 输出的教师值 t_q 可能为 ±1，如果我们令 $t_q=-1$ 的输入样本 $\boldsymbol{x}^P$，都取其相反方向 $-\boldsymbol{x}^P$，则 $\boldsymbol{w}^{T*}\boldsymbol{x}<0$，全变为 $\boldsymbol{w}^{T*}\boldsymbol{x}>0$，$\boldsymbol{w}^*$ 是要求的矢量值，所以我们只要证明对所有的 k^P 输入样本，通过(2-18)公式的迭代，必能找到一个解 $\boldsymbol{w}^*$，使 $\boldsymbol{w}^{T*}\boldsymbol{x}>0$。

(3) $\boldsymbol{w}^*$ 的大小不影响结果，可以令 $\|\boldsymbol{w}^*\|=1$。

证明：因为 k 个样本是线性可分的，所以必存在一个 $\boldsymbol{w}^*$ 满足对所有 k 个样本：

$$\boldsymbol{w}^*\cdot\boldsymbol{x}^P\geqslant\delta,\ \delta>0$$

用 $s(n)$ 表示 $\boldsymbol{w}(n_0)$ 与 $\boldsymbol{w}^*$ 之间的夹角的余弦，$\boldsymbol{w}(n_0)$ 表示第 n_0 次迭代的权值：

$$s(n_0)=\frac{\boldsymbol{w}^*\cdot\boldsymbol{w}(n_0)}{\|\boldsymbol{w}^*\|\ \|\boldsymbol{w}(n_0)\|} \tag{2-19}$$

利用(2-18)式可得

$$\boldsymbol{w}^*\cdot\boldsymbol{w}(n_0+1)=\boldsymbol{w}^*\cdot[\boldsymbol{w}(n_0)+\mu\boldsymbol{x}(n_0)]\geqslant\boldsymbol{w}^*\cdot\boldsymbol{w}(n_0)+\mu\delta \tag{2-20}$$

这里 $\mu=2\eta\cdot e_q=4\eta$，为一个常数。从上式可得

$$\boldsymbol{w}^*\cdot\boldsymbol{w}(n_0)\geqslant\boldsymbol{w}^*\cdot\boldsymbol{w}(0)+n_0\mu\delta$$

如选初值 $\boldsymbol{w}(0)$ 为输入 $\boldsymbol{x}^P$ 中任一个矢量，则 $\boldsymbol{w}^*\cdot\boldsymbol{w}(0)>0$，可得

$$\boldsymbol{w}^*\cdot\boldsymbol{w}(n_0)>n_0\mu\delta \tag{2-21}$$

由于在 $\boldsymbol{w}(n_0)$ 没有达到 $\boldsymbol{w}^*$ 时，有 $\boldsymbol{w}(n_0)\cdot\boldsymbol{x}(n_0)<0$，则有

$$\|\boldsymbol{w}(n_0+1)\|^2=\|\boldsymbol{w}(n_0)\|^2+2\boldsymbol{w}(n_0)\cdot\boldsymbol{x}(n_0)\mu+\mu^2<\|\boldsymbol{w}(n_0)\|^2+\mu^2$$

$$\|\boldsymbol{w}(n_0)\|^2<\|\boldsymbol{w}(0)\|^2+n_0\mu^2=1+\mu^2n_0 \tag{2-22}$$

将(2-21)和(2-22)代入(2-19)得

$$s(n_0)\geqslant\frac{n_0\mu\delta}{\sqrt{1+\mu^2n_0}} \tag{2-23}$$

上式在 $\boldsymbol{w}^*$ 与 $\boldsymbol{w}(n_0)$ 相等时，$s(n_0)=1$

由(2-23)得

$$n_0\leqslant\frac{1+\sqrt{1+\frac{4\delta^2}{\mu^2}}}{2\delta^2} \tag{2-24}$$

则可通过(2-18)式的有限次迭代达到收敛。

证毕。

三、多层的感知器网络

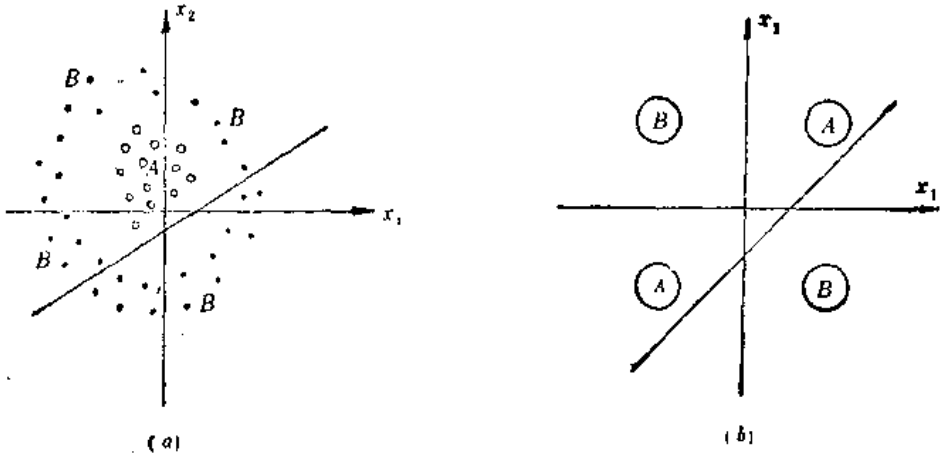


图 2-6　不能线性划分的样本集

单层感知器只能满足线性分类，如果有两类样本 A，B，它们不能用一个超平面将它分开，如图 2-6a，b 所示的二维平面中，A 类和 B 类是分布在此平面中的一些点或区域，图

2-6a 中 A 类分布在原点附近，B 类分布在 A 类的外部，不能用直线把两类分开。在图2-6b中，A 类是分布在第一象限和第三象限，而 B 类是分布在第二象限和第四象限，用一根直线同样不能把 A、B 类分开。

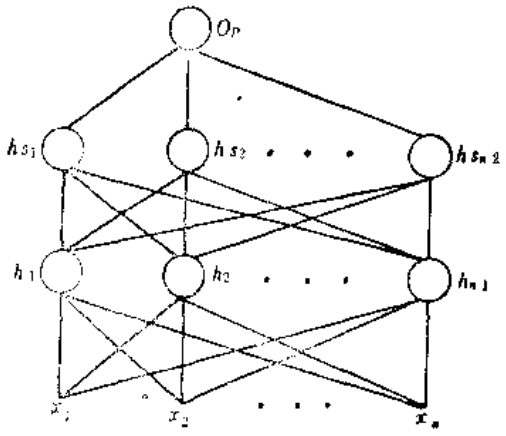


图 2-7　多层感知器网络

多层感知器网络是可以用来解决上述的问题。多层感知器网络如图 2-7 所示，输入与输出层之间存在一些隐层，输入为 n 个神经元 x_1，x_2，…，x_n，第一隐层为 n_1 个神经元 h_1, h_2，…，h_{n1}，第二隐层为 n_2 个神经元 $h_{s1}, h_{s2}, \cdots, h_{sn2}$，输出为 O_P。它们是以前馈的方式联接。

对第一隐层的第 j 个神经元，输出为

$$h_j=f\left(\sum_{i=1}^{n}w_{ij}x_i-\theta_j\right)\quad j=1,2,\cdots,n_1 \tag{2-25a}$$

对第二隐层的第 k 个神经元，输出为

$$h_{sk}=f\left(\sum_{j=1}^{n_1}w_{jk}h_j-\theta_k\right)\quad k=1,2,\cdots,n_2 \tag{2-25b}$$

对最高层神经元输出为

$$O_P=f\left(\sum_{k=1}^{n_2}w_k h_{sk}-\theta\right) \tag{2-25c}$$

其中：

$$f(\alpha)=\begin{cases}1 & \alpha\geqslant 0\\ -1 & \alpha<0\end{cases}$$

如果这里我们只讨论一层隐单元的情况，删去第二隐层，把(2-25c)式改为

$$O_P=f\left(\sum_{j=1}^{n_1}w_j h_j-\theta\right)$$

此时隐层与 n 个输入单元的关系如同单层的感知器网络一样，是形成一些 $n-1$ 维的超平面，把 n 维的输入空间划为一些小的子区域，例如 $n=2, n_1=3$ 的情况下，对于隐层，第 j 个($j=1,2,3$)个神经元的输出为

$$\begin{aligned}h_1&=f\left(\sum_{i=1}^{2}w_{i1}x_i-\theta_1\right)\\ h_2&=f\left(\sum_{i=1}^{2}w_{i2}x_i-\theta_2\right)\\ h_3&=f\left(\sum_{i=1}^{2}w_{i3}x_i-\theta_3\right)\end{aligned} \tag{2-26}$$

w_{i1} 为第 i 个输入到第一个隐单元的权，w_{i2}、w_{i3} 为第 i 个输入到第二、三个隐单元的权，h_1、h_2、h_3 为隐单元输出。

可以在二维输入空间中划出三根线，因为 w_{ii} 和 θ_i 的不同，此三线的斜率与截距各不相同，如图 2-8a 所示，对于图 2-6a 的问题，就是寻找一个区域使其内为 A 类，其外为 B 类，用这三个隐单元所得到的一个封闭区域就可满足其条件，从隐单元到输出层只要满足下式即可得到正确划分：

$$\begin{aligned}O_P=\{(x_1,x_2)/[&(w_{11}x_1+w_{21}x_2-\theta_1)>0\cap(w_{12}x_1+w_{22}x_2-\theta_2)\\ &>0\cap(w_{13}x_1+w_{23}x_2-\theta_3)>0]\}\end{aligned} \tag{2-27}$$

O_P 为第三层的输出。

十分明显，隐单元到输出单元之间为"与"的关系。

对于图 2-8*b* 和图 2-8*c*，我们可以分别采用四层或三层感知器网络来完成，第一层为输入层，第二层为线性划分作用，第三层为"与"组合，第四层为"或"组合。

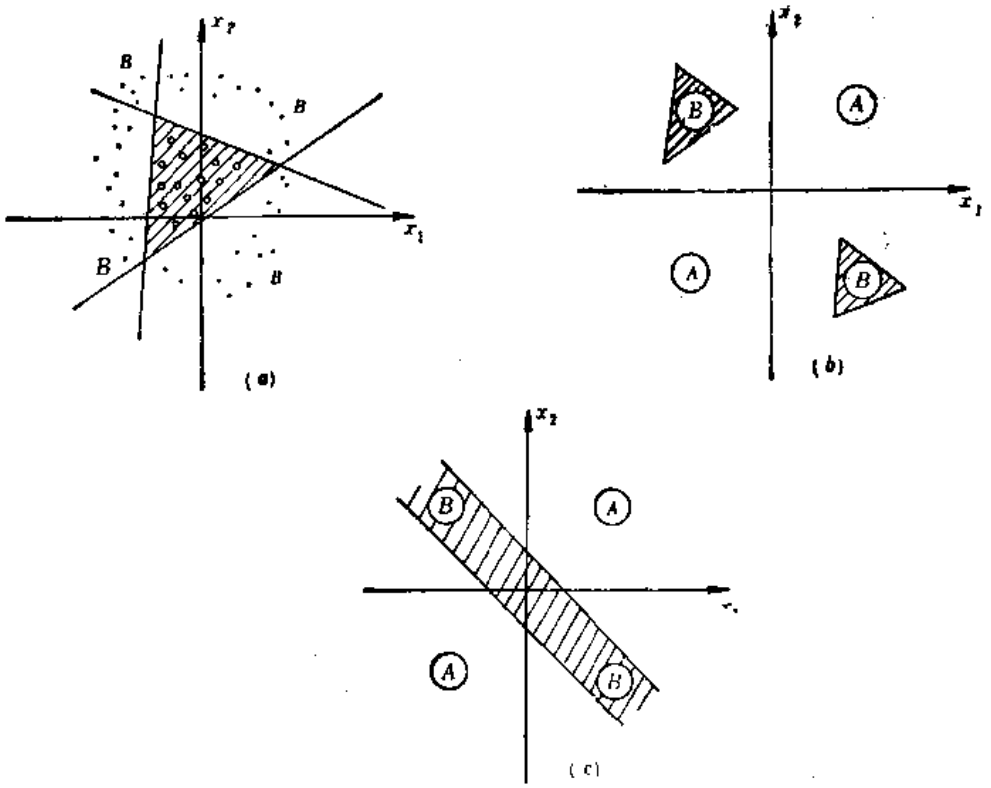


图 2-8 多层感知器网络在输入空间中的划分

对于图 2-8*b*，用四层感知器网络来完成。

$$O_P=\left\{(x_1x_2)/\left[\left(\bigcap_{j=1}^{3}(w_{1j}x_1+w_{2j}x_2-\theta_j)>0\right)\cup\left(\bigcap_{j=4}^{6}(w_{1j}x_1+w_{2j}x_2-\theta_j)>0\right)\right]\right\}$$

这里有六根直线，各用三根直线组成了两个三角形的区域，这是用"与"来完成，两个区域内部都属于 B 类，这又是"或"的关系，其他区域则为 A 类。

对于图 2-8*c*，用三层感知器网络来完成。

$$O_P=\{(x_1x_2)/[(w_{11}x_1+w_{21}x_2-\theta_1)>0\cap(w_{12}x_1+w_{22}x_2-\theta_2)<0]\}$$

用两根直线相"与"达到属于 B 的区域。

可以看出，多层感知器可通过单层感知器进行适当的组合达到任何形状的划分。

如何来设计多层感知器隐单元的个数，这里我们给出隐单元数的两种不同角度考虑作为上限和下限，以供设计时参考。

1. 隐单元数的上限

在(2-25)式中，令隐单元输出 $h_j=\mathrm{Sgn}\left(\sum_{i=1}^{n}w_{ij}x_i-\theta_j\right)$ 是一个线性阈值分割函数，每个隐单元把 n 维输入空间 S 划为 S^A、S^B 两个部分。设 g 是定义为 S 空间的一个函数，对于感知器网络对应的 k 个样本输入，希望其输出满足：

$$O_p^l=\mathrm{sgn}\left(\sum_{j=1}^{n_1}h_jw_j^*-\theta^*\right)=g(x^l),\quad l=1,2,\cdots,k \tag{2-28}$$

问题是能否找到隐单元数 n_1，通过调整输出单元的系数 w_j^*，θ^* 使(2-28)式满足条件，即可实现任何一个 g 函数的映照。

在一个输入空间 S 上，存在有 k 个样本，如果需要用 n_1 个超平面划分为 d 个区域，使 $d>k$，并保证每个区域对应于一个样本，则最大的隐单元数 n_1 满足：

$$n_1=k-1 \tag{2-29}$$

这个论断可以证明。我们假设一个三层感知器网络，其输入 $\boldsymbol{x}\subset R^n$，输出为一个单元，隐单元 $n_1=k-1$；k 为样本数。考虑一个 $k\times k$ 的矩阵 $\boldsymbol{D}$，当输入 $\boldsymbol{x}^1,\cdots,\boldsymbol{x}^k$ 样本时，让 $\boldsymbol{D}$ 矩阵第 j 列上的每个单元分别为第 j 个隐单元的输出($1\leqslant j\leqslant k-1$)值，而第 k 列每个单元都为 1。设每个隐单元把输入 S 空间划为两个子空间 S^A、S^B，分别对应为 $\{x_j\}$ 和 $\{x_{j+1},\cdots,x_k\}$，即第 j 个样本输入时，其对应隐单元 h_j 为 1，而 h_{j+1}，h_{j+2}，$\cdots$，h_{n1} 为 -1，这样令 $\boldsymbol{D}$ 矩阵的对角线上的值都为 1，在矩阵 $\boldsymbol{D}$ 中对角线以上的元素值不定，而对角线以下元素值为 -1，即

$$\boldsymbol{D}=\begin{bmatrix} 1 & \times & \times & \times & 1 \\ -1 & 1 & \times & \times & 1 \\ -1 & -1 & 1 & \times & 1 \\ \cdot & & \cdot & \cdot & \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix} \tag{2-30}$$

其中"×"表示为任意值，而第 k 列为全 1。十分明显，$\boldsymbol{D}$ 矩阵的形式可以保证 $\boldsymbol{D}^{-1}$ 存在，因为不可能存在线性相关的列。要求输出函数定义为 $g=[g(x^1),g(x^2),\cdots,g(x^{k-1}),g(x^k)]^T$，则可以得到一个系数矢量 $\boldsymbol{C}$：$\boldsymbol{C}\in R^k$

$$\boldsymbol{C}=\boldsymbol{D}^{-1}g \tag{2-31}$$

矢量 $\boldsymbol{C}$ 是输出神经元和隐单元的联接权和阈值 θ 组成的，满足(2-29)式的隐单元数，用三层感知器网络可对在 S 空间里定义的分类函数进行正确映照。这种隐层单元分割方法从几何上看，是把 n 维的 S 空间用 $k-1$ 个超平面不交叉地进行划分，得 k 个区间，而每个区间中包括一个样本，这样可经过组合达到正确分类。

2. 隐单元数的下限

在 k 个样本输入下，隐单元数为 $k-1$ 个是设计网络的上限，因为 $k-1$ 个超平面是不相交的，因此只可分为 k 个区域，但考虑到超平面可交的情况且只有一个隐层的情况下，输入为 n 个单元，n_1 个隐单元可把输入空间划分成一定数目的区域，这些区域是有限的 $n-1$ 维超超平面的交，如果这个区域是封闭的，我们称之为封闭区域，反之为开区域，图 2-9 为在二维空间中三个隐单元所划分的区域，在图 2-9a 中，可以有一个封闭区域 7 和 21 个开区域，但这些开区域并不是独立的，如在图 2-9a 中2,7合为一个区域，而2,7分别又为两个区域，现在我们讨论独立区域的情况。如果输入的每个样本都落到一个独立区域中，如图 2-9b 所示。那么，表示在每个样本输入时，对应的隐单元的输出是不相同的，从而可以通过"与""或"的组合得到正确的分类。这样对 n_1 数的求解，就变成求在 n 维空间中 n_1 的分割，能得到最大的区域数。在数学上可得到独立区域数为

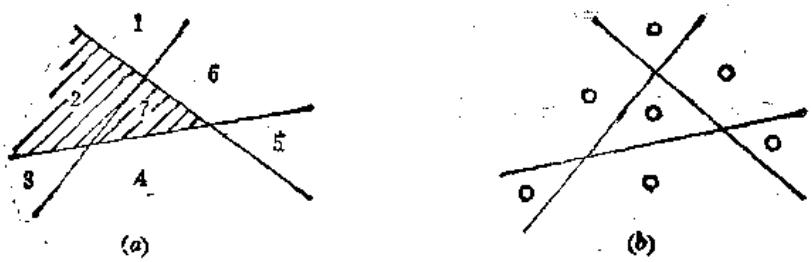


图 2-9

(a) 两维空间三个隐单元所划的区域；(b) 独立区域

$$P(n_1,n)=\sum_{i=0}^{n}\binom{n_1}{i},n_1\geqslant i \tag{2-32}$$

$$\binom{n_1}{i}=0,n_1<i$$

那么，对 k 个样本进行线性分割则要求：

$$n_1=\min[P(n_1,n)]\geqslant k \tag{2-33a}$$

在图 2-9a 中，$n=2,n_1=3$，则最大的独立区域数为

$$P(3,2)=\sum_{i=0}^{2}\binom{n_1}{i}=1+3+3=7$$

如果存在 7 个输入样本，它的线性划分可以用图 2-9b 表示，当然 7 个样本是否刚好落到 7 个区域中，必须通过对第一层与第二层之间的权进行调节而得到，仔细区分其独立封闭区域和开区域，对于有界封闭区域数为

$$P_a(n_1,n)=\begin{cases}0 & n_1\leqslant n\\ \binom{n_1-1}{n} & n_1>n\end{cases} \tag{2-33b}$$

对于独立开区域：

$$P_b(n_1,n)=\begin{cases}2n_1 & n_1\leqslant n\\ 2\sum_{i=0}^{n-1}\binom{n_1-1}{i} & n_1>n\end{cases} \tag{2-33c}$$

对于 k 个样本，隐单元取值满足(2-33x)式就可得到正确划分。

隐单元数 n_1 是否可再减小，可从图 2-9b 中看出，如减少后使两个样本落在一个线性分割区内，如这两个样本为同一类，则对分类结果不会产生影响，这里讨论的下限与类别数无关，所以下限就为(2-33a)式，在实际问题中有可能减小。

3. 多层感知器的设计

感知器网络的输入：根据需要分类的样本数据与表述来定，如输入是 n 维图像，或者是 n 个特征量，其输入为 n 维空间的矢量，输出为其类别数，或类别数的编码表述。感知器的隐单元数，可根据公式(2-29)和(2-33a)来确定，例如，用多层感知器来完成分类问题，输入两个神经元(x_1,x_2)分别为$(-1,-1)$、$(1,1)$时，其输出 $g=-1$；(x_1,x_2)分别为$(-1,1)$、$(1,-1)$时，输出 $g=+1$。

根据公式(2-33a)，$k=4$，要求$k\geqslant P(n_1,n)$，$n=2$，我们看到当$n_1=2$时，满足$k\geqslant P(n_1,n)$，可用两个隐单元来完成。同样也可根据上限的要求$n_1=k-1=3$，用三个隐单元来完成。用这两种方法得到的权和结构如图 2-10a、b 所示。

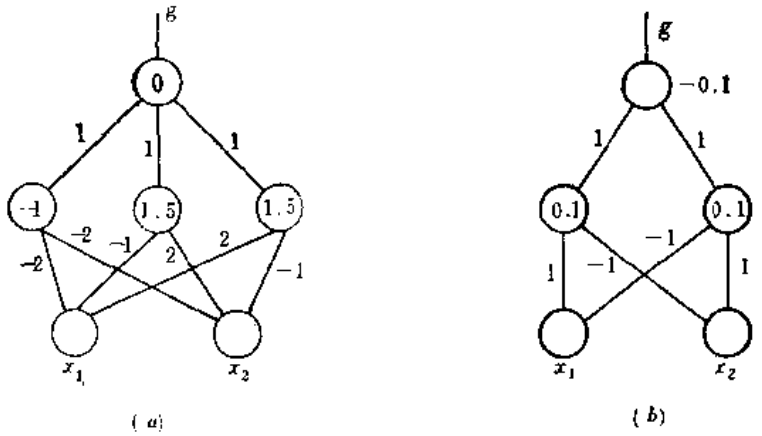


图 2-10

(a) 用上限计算的 XOR 感知器网络；(b) 用下限计算的 XOR 感知器网络

感知器的结构定了以后，可用学习的方法得到输入层到隐层的权，而输出与隐层之间的权可用"与""或"的方法。或用(2-31)式得到。

多层感知器可对输入空间矢量进行分类，也可对一些离散函数进行映照，它的输入、输出间的映照是比较确定的。它的抗干扰性和"强壮"性比较差，在样本的畸变，噪声加入的情况下，不能得到正确分类。

第二节　自适应线性单元组成的网络

自适应线性神经元(Adaline)是在 1961 年由斯坦福大学教授 Windrow 提出的，它是一个自适应可调的网络，适用于信号处理中的自适应滤波、预测和模型识别。构成这种网络的基本单元 Adaline 是一个类似于感知器单元，如图 2-11 所示。如在第 k 时刻，有向量 $\boldsymbol{x}_k$ 输入，权向量为$\boldsymbol{w}_k$，此时有模拟输出 y_k，$y_k=\boldsymbol{w}_k^T\boldsymbol{x}_k$ 及二进制输出 q_k。y_k与要求的理想响应的差值，通过 LMS 算法，修改 $\boldsymbol{w}_k$ 为 $\boldsymbol{w}_{k+1}$，从而减小了 y_k 及理想响应的误差。这个单元的输入与输出关系满足：

$$\begin{aligned} y_k&=\sum_{i=1}^{n}w_{ik}x_{ik}-\theta_k \\ q_k&=\operatorname{sgn}(y_k) \end{aligned} \tag{2-34}$$

$\boldsymbol{x}_k,\boldsymbol{w}_k\in R^n$，$\theta_k$ 为阈值，如使 $\theta_k=w_{0k}$，$x_0=-1$，那么

$$y_k=\sum_{i=0}^{n}w_{ik}x_{ik}$$

比较(2-34)式与(2-4)式，其差别只是输出分为模拟和数字两个部分。从数字部分看，与感知器单元完全相同，它可进行线性分割。从模拟输出看，它是作为误差调节之用，对单个 Adaline，其误差为模拟输出和要求响应输出之差，也为模拟量。用 LMS 算法能保证这种网络在自适应学习时的收敛性。Adaline 的特点是：可以根据外界输入的情况和要求

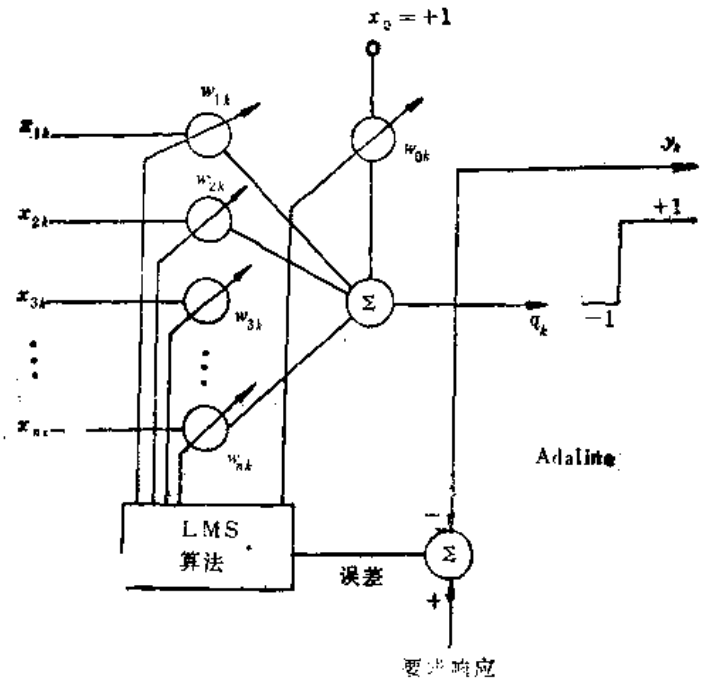


图 2-11 Adaline 原理图

的响应随时学习，十分灵活。

一、用于非线性分割的自适应网络

对于单层的 Adaline 网络，由于它的形式与感知器相同，因而它只能适用于一种线性分割的情况，为了要达到非线性的分割，可采用下列两种方法：

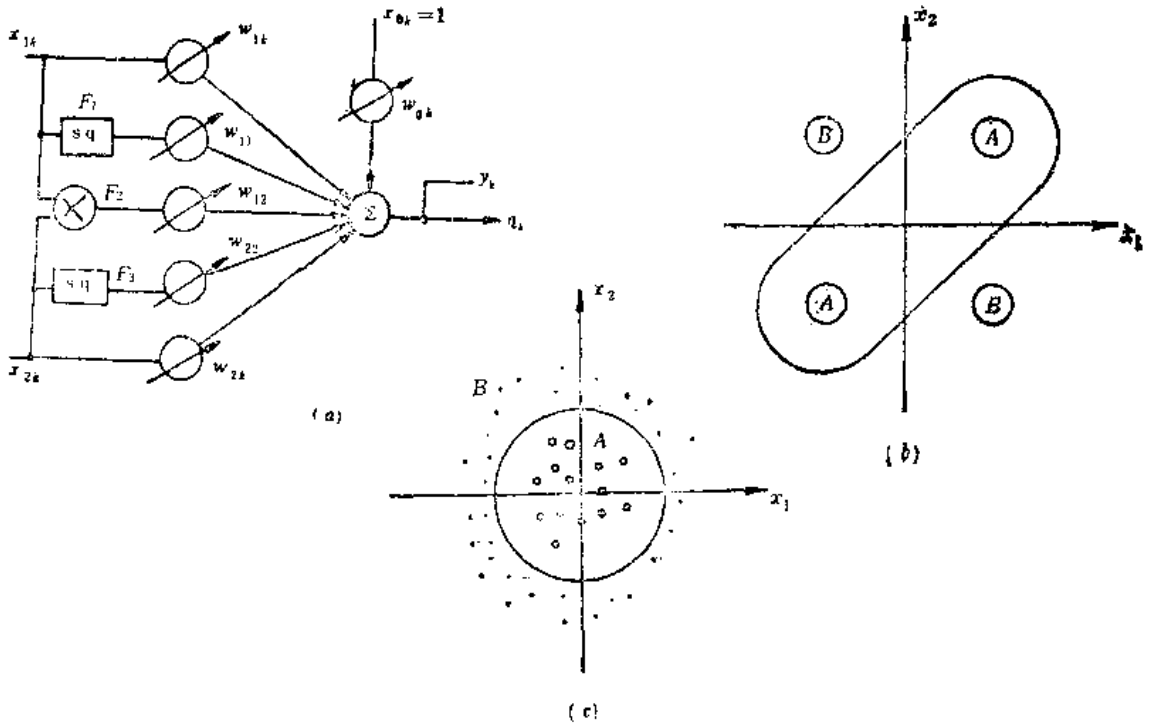


图 2-12

(a) 非线性分割的 adaline 结构；(b) 用椭圆分界面解 XOR 问题；(c) 用圆分界面分割

(1) 加一些辅助的输入单元，使输入矢量 $\boldsymbol{x}_k \in R^n$ 变为 $\boldsymbol{x}_k' \in R^{n+l}$，这 l 个输入单元的输入是 $\boldsymbol{x}_k$ 矢量中各单元 x_{ik} 的二次值，$i=1,2,\cdots,n$。如在两维的情况下，$n=2$，我们加上三个辅助输入节点 F_1、F_2、F_3，如图 2-12a 所示，使其 $F_1=x_{1k}^2$，$F_2=x_{1k}x_{2k}$，$F_3=x_{2k}^2$。这样我们得到其输出 y_k 为

$$y_k=\sum_{i=0}^{n} w_{ik}x_{ik}+w_{11}F_1+w_{12}F_2+w_{22}F_3$$

$$=w_{0k}+x_{1k}w_{1k}+x_{2k}w_{2k}+w_{11}x_{1k}^2+w_{22}x_{2k}^2+w_{12}x_{1k}\cdot x_{2k} \tag{2-35}$$

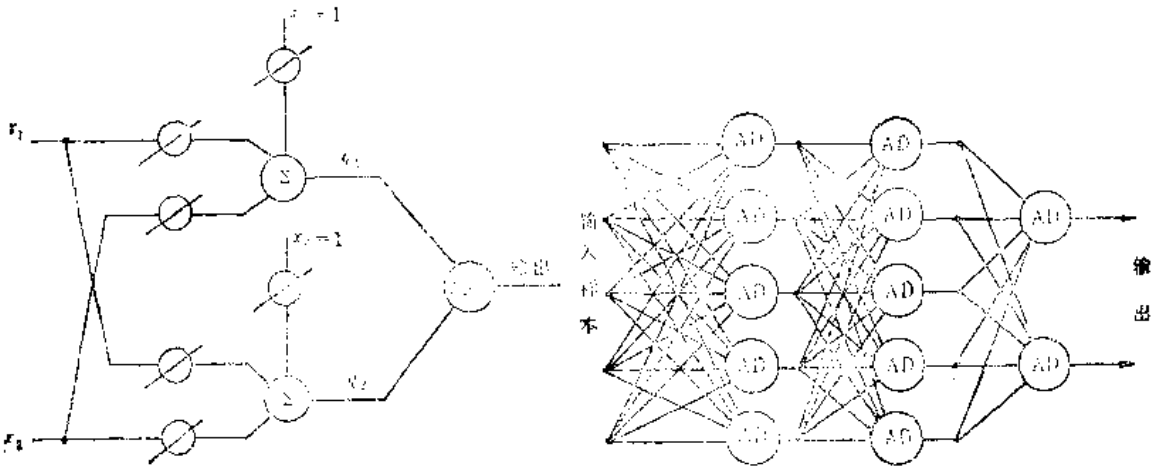
当 $y_k=0$ 时，在两维空间上的分界面由(2-35)式给出，现在是一个椭圆或是一个圆，可以通过调节权系数来达到这一类的非线性分割。例如在图 2-6a 和图 2-6b 的两种情况，采用辅助输入单元，可以得到正确的分类，如图 2-12b、c 所示。

当然，这种辅助输入单元的网络将会因为输入维数的增加，其辅助输入维数亦增加，如原来输入单元维数为 n，考虑二次情况，输入总的维数为

$$2n+C_n^2=2n+\frac{n(n-1)}{1\cdot 2}=\frac{n^2+3n}{2}$$

因而增加了复杂性。

(2) Madaline 网络，是由多个 Adaline 单元组合成的多层网络，图 2-13a 为两个 Adaline 和一个与门组合成的 Madaline 网络。多个 Adaline 组合成的多层网络如图 2-13b 所示，图中AD是表示 Adaline 的符号。



(a) Mabaline　　(b) 多层网络

图 2-13

(a) Madaline 结构；(b) 多层 Madaline 网络

它的非线性分割与多层感知器一样，如输入矢量为 n 维，利用组合很多的 $n-1$ 维的超越平面来实现。

二、多层 Adaline 网络的学习算法

多层 Adaline 的学习分两部分进行，先利用 MRII 的算法计算出每一层的输出要求，然后再采用单层的 LMS 算法来完成每一层间的权的学习。如果输入是一个模拟量，第一层的 Adaline 的输出是两个量，一是模拟量 y_k，提供给 LMS 算法用来调整该层的输入权，另一是

数字量 q_k，是输入到下一层的网络。这样从第二层 Adaline 开始，其输入矢量是二进制的量，MRII 算法就是建立在这些二进制量的基础上。它的思想是，从对网络干扰最小的二进制输出量，改变其符号，如改变第一层中某一个 Adaline 输出的符号，(从 $+1\rightarrow-1$，或从 $-1\rightarrow+1$)观察其总的网络的输出响应和要求的值之间的误差，如误差减小，就确认这种改变，若误差不减小，恢复原来符号，每次选中的 Adaline 应是那些模拟量最接近 0 的单元，这样可使网络的干扰最小。

这样我们可以列出 MRII 的学习步骤：

(1) 初始化多层 Adaline 网络中的权，用一个随机数作为各个权的初始权值。

(2) 输入一个样本矢量 $\boldsymbol{x}_k$ 和要求的输出 t_k，用式(2-34)按层一步一步地计算出实际输出，并求出要求的输出 t_k 和实际输出的误差。

(3) 根据最小干扰的原则，从第一层开始，找第一层中神经元模拟输出最接近于 0 的那个神经元，让它输出数字量改变符号。

(4) 计算网络最后的输出和要求响应之间的误差，如果误差减小，则这种改变被接受，如没有减小，上一步改变的输出数字量的符号恢复。

(5) 第一个神经元训练结束后，转入下一个神经元，即模拟输出第二个接近零的单元，仍按上面的规则训练。

(6) 当这一层单个神经元训练完了，再按两个一组地训练，然后再按三个一组，……直到输出与要求响应之间的距离减到最小为止。然后用同样的方法训练第二层、第三层，到相同时为止。

(7) 另一个新的样本输入，用同样方法进行训练，一直到达到误差最小为止。

(8) 通过 MRII 的学习可以得到每个样本输入对应的每一层相应的输出，从而对每一单层的 Adaline 可以用 LMS 算法进行学习，得到全部权的解。

单层的学习算法如下：

$$\boldsymbol{w}(n_0+1)=\boldsymbol{w}(n_0)+\frac{\alpha}{\|\boldsymbol{x}(n_0)\|}\varepsilon(n_0)\boldsymbol{x}(n_0) \tag{2-36}$$

这里，$\boldsymbol{w}(n_0+1)$为 $\boldsymbol{w}(n_0)$的下一个时刻的权矢量，$\boldsymbol{x}(n_0)$ 为当前输入样本矢量，$\varepsilon(n_0)$为当前的误差。n_0 为迭代的次数。

如果 $\boldsymbol{x}(n_0)$矢量的分量为 ±1，那么 $\|\boldsymbol{x}(n_0)\|$ 为

$$\|\boldsymbol{x}(n_0)\|^2=x_1^2(n_0)+x_2^2(n_0)+\cdots+x_{n+1}^2(n_0)$$

对于每个调节周期，可按照(2-36)进行调节。误差 $\varepsilon(n_0)$为

$$\varepsilon(n_0)=t(n_0)-\boldsymbol{x}^T(n_0)\boldsymbol{w}(n_0) \tag{2-37}$$

当权改变时，其误差改变为

$$\Delta\varepsilon(n_0)=\Delta[t(n_0)-\boldsymbol{x}^T(n_0)\boldsymbol{w}(n_0)]=-\boldsymbol{x}^T(n_0)\Delta\boldsymbol{w}(n_0) \tag{2-38}$$

$$\Delta\boldsymbol{w}(n_0)=\boldsymbol{w}(n_0+1)-\boldsymbol{w}(n_0)=-\frac{\alpha}{\|\boldsymbol{x}(n_0)\|}\varepsilon(n_0)\boldsymbol{x}(n_0) \tag{2-39}$$

综合(2-38)式和(2-39)式得

$$\Delta\varepsilon(n_0)=-\boldsymbol{x}^T(n_0)\frac{\alpha}{\|\boldsymbol{x}(n_0)\|}\varepsilon(n_0)\boldsymbol{x}(n_0)=-\alpha\varepsilon(n_0) \tag{2-40}$$

这说明，当网络的输入不变时，误差的变化是缩小当前误差的 α 倍，选择 α 因子可以控

制收敛的速度和稳定性，α 太大，误差将可能增大。一般取 $1.0>\alpha>0.1$。

三、Adaline 和 Madaline 的应用

（一）在信号处理中的应用

单个 Adaline，作为调节单元可以形成一个自适应滤波器，用于信号处理。

图 2-14a 是一个 Adaline 单元，它的输入 x_k 为一个随时间变化的信号，输入的信号通过延迟得到 $x_{k-1}, \cdots, x_{k-l}$，l 为延迟器的最大数目，它们是通过对输入信号的采样而得到的。因为信号随时间变化，因而当前采样值为 x_k，在 Adaline 的输入端分别为 $x_k, x_{k-1}, \cdots, x_{k-l}$，下一时刻为 $x_{k+1}, \cdots, x_{k-l+1}$；$x_{k+1}$ 为当前时刻，x_{k-l+1} 为延迟了 l 个节拍后的采样值，即上 l 个节拍前的采样值。Adaline 网络的输入 $l+1$ 维的变化的信号。自适应滤波的目的是通过要求的响应和实际输出的误差，调节其权而达到对输入信号滤波的目的。

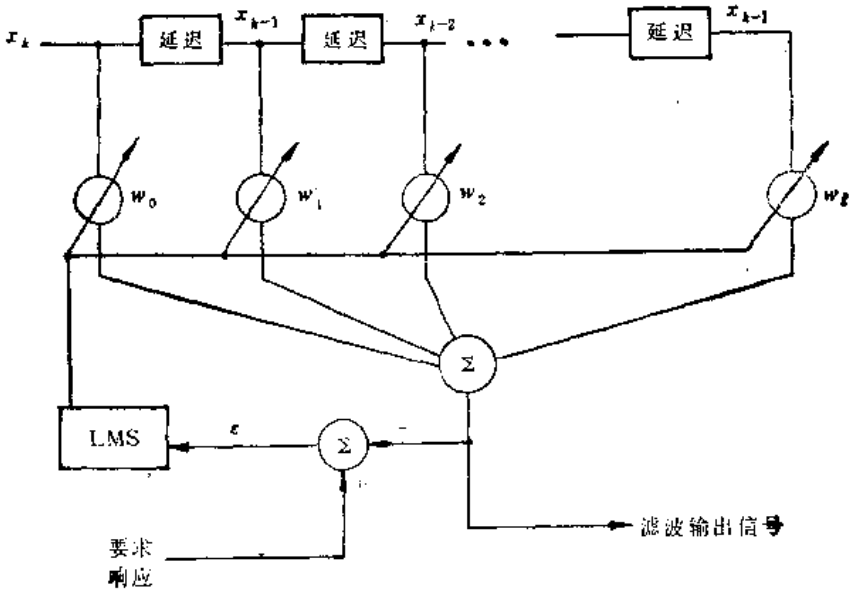


图 2-14(a)　Adaline 组成的自适应滤波器

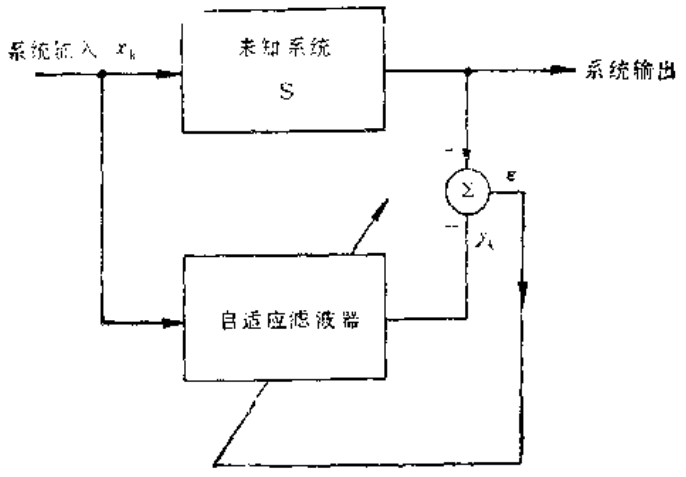


图 2-14(b)　用自适应滤波器进行建模的框图

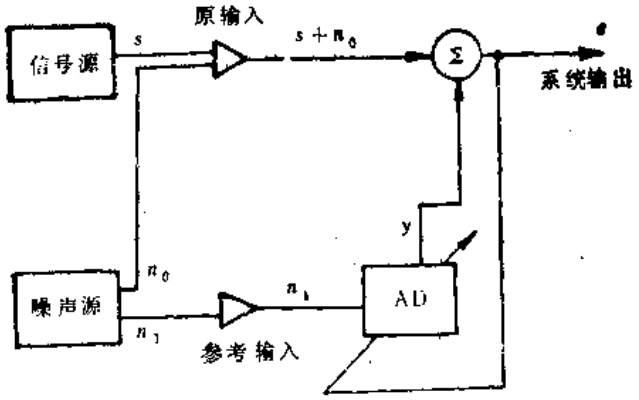


图 2-14(c)　噪声对消框图

其具体应用如下：

1. 系统的建模

图 2-14b 是一个采用自适应滤波器进行建模的框图，设 S 为一个未知系统，我们希望通过已知的输入信号 x_k 加入未知系统后，而得到的输出作为单个 Adaline 要求响应，而已知信号同时加到 Adaline 组成的自适应滤波器上，得到的输出 y_k，y_k 与未知系统的输出之间的误差来调节自适应滤波器的权，从而就可以得到 S 的数学模型为

$$S=\sum_{l=0}^{n} w_l x_{kl}$$

其中的 $w_l(l=0, \cdots, n)$ 是通过调节得到的，如果未知系统是一个稳定的时不变系统，输入的信号是频谱较宽的白噪声或 δ 脉冲，那么 w_l 是可以收敛到一个定值。这样，未知系统 S 就可用一个确定的线性模型来描述。

2. 噪声对消

对于一个最优的滤波器，希望通过滤波将信号中的噪声去掉，这对一般的滤波器很难完全做到。图 2-14c 是一个用 Adaline 来进行噪声对消应用的示意图。如果有一个输入信号 s，加上一个与 s 不相关的噪声 n_0 作为 Adaline 神经元的要求响应，而与噪声 n_0 相关的信号 n_1 输入 Adaline 组成的自适应滤波器，其输出为 y，y 与 n_1 相关，其误差为 e：

$$e=s+n_0-y$$

假设 s 是平稳的零均值的随机信号，n_0 是一个独立于 s 的随机噪声，可以得到

$$e^2=s^2+(n_0-y)^2+2s(n_0-y) \tag{2-41}$$

$$E(e^2)=E(s^2)+E[(n_0-y)^2]+2E[s(n_0-y)]=E(s^2)+E[(n_0-y)^2]$$

通过调节 AD(Adaline 网络)，得到

$$E_{\min}(e^2)=E_{\min}(s^2)+E_{\min}[(n_0-y)^2] \tag{2-42}$$

上式中，当 $E_{\min}[(n_0-y)^2]\to 0$，则说明 y 最靠近 n_0。此时系统的输出 e 为 s，其噪声 n_0 被抵消。采用这种系统来完成对胎儿心率的检测，得到十分满意的结果。由于测量胎儿的心率一定会受到母体心率的干扰，而且母亲心率很强，但与胎儿心率则是相互独立的，我们可将母体心率作为噪声源 n_1 进入 Adaline 中，混有噪声的胎儿的心率信号作为要求响应，通过对消后，系统就可以得到清晰的胎儿心率。

Adaline 除了上面两个例子外，还在电话机的回声对消、电话机内的均衡器中都有应

用，只要一个神经元就可以完成很多其他方法很难达到的效果，这种 Adaline 十分简单，而且用硬件很容易达到，因而成为应用非常广泛的神经元模型。

（二）在模式识别中的应用

一个用作字母识别的 Madaline 组成的系统如图 2-15 所示，输入为二维的文字，称作视网膜，它和生物体的视网膜一样能接受到外界的信息，二维的文字排列为一个方阵，阵中每个元作为不变性网络的输入单元，对文字说，输入是二值量 ±1，白的为 −1，黑的为 +1。

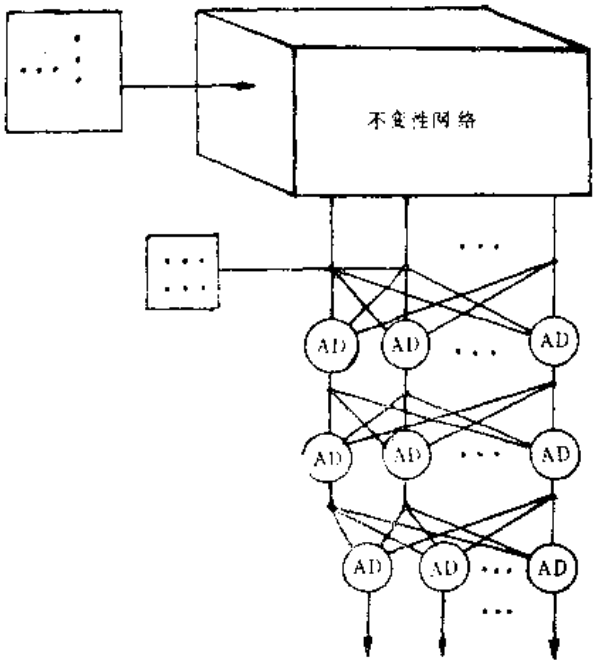


图 2-15　字母识别 Madaline 系统

不变性网络是由多个 Adaline 单元经过有规则的排列而得到的，其目的是，当输入文字样本发生平移、旋转时，其输出仍然不发生变化，因此不变性网络的输出只与不同输入模式相对应，而与同一个输入模式的位置、方向无关。从不变性网络输出的不同模式，经过一个多层的 Adaline 网络决策学习后便可取得正确的分类识别。

不变性网络由很多子平面组成。如果考虑旋转不变，那么几个子平面的组合，对应于一个输出，因而不变性网络包含很多子平面，并组合为一定的输出，每个子平面如图 2-16a 所示，图中的 (AD) 表示 Adaline，一个平面上有很多 (AD) 按方阵排列，输入视网膜上所有的元都与每个 (AD) 相联，而每个 (AD) 的输出都接到一个选择器 MAJ 上，如果在子平面 s_1 左上角上的 Adaline 与输入二维图像的连接权为 $\boldsymbol{W}_1$（这也是一个矩阵形的），那么在 s_1 子平面上其他 (AD) 与输入单元的连接权分布如下：

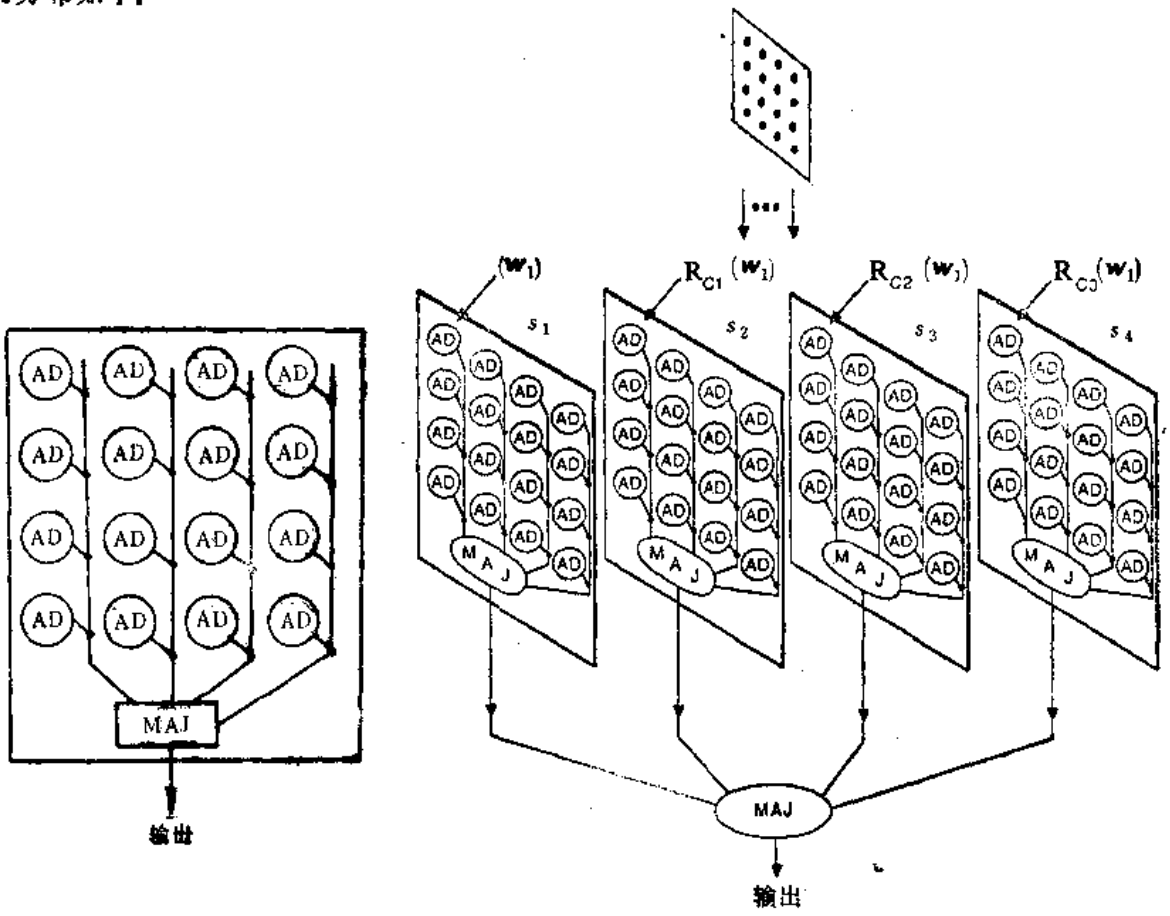
$$\begin{bmatrix} \boldsymbol{W}_1 & TR_1(\boldsymbol{W}_1) & TR_2(\boldsymbol{W}_1) & TR_3(\boldsymbol{W}_1) \\ TD_1(\boldsymbol{W}_1) & TD_1TR_1(\boldsymbol{W}_1) & TD_1TR_2(\boldsymbol{W}_1) & TD_1TR_3(\boldsymbol{W}_1) \\ TD_2(\boldsymbol{W}_1) & TD_2TR_1(\boldsymbol{W}_1) & TD_2TR_2(\boldsymbol{W}_1) & TD_2TR_3(\boldsymbol{W}_1) \\ TD_3(\boldsymbol{W}_1) & TD_3TR_1(\boldsymbol{W}_1) & TD_3TR_2(\boldsymbol{W}_1) & TD_3TR_3(\boldsymbol{W}_1) \end{bmatrix} \tag{2-43}$$

在上面的分布中，TR_1 的意义是 $\boldsymbol{W}_1$ 向右循环移一列，TD_1 的意义是 $\boldsymbol{W}_1$ 向下循环移一行，$TD_1TR_1(\boldsymbol{W}_1)$ 表示 $\boldsymbol{W}_1$ 向下循环移一行并向右循环移一列，即

$$W_1=\begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \\ w_{31} & w_{32} & w_{33} & w_{34} \\ w_{41} & w_{42} & w_{43} & w_{44} \end{bmatrix} \qquad TD_1(W_1)=\begin{bmatrix} w_{41} & w_{42} & w_{43} & w_{44} \\ w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \\ w_{31} & w_{32} & w_{33} & w_{34} \end{bmatrix}$$

$$TR_1(W_1)=\begin{bmatrix} w_{14} & w_{11} & w_{12} & w_{13} \\ w_{24} & w_{21} & w_{22} & w_{23} \\ w_{34} & w_{31} & w_{32} & w_{33} \\ w_{44} & w_{41} & w_{42} & w_{43} \end{bmatrix} \qquad TD_1TR_1(W_1)=\begin{bmatrix} w_{44} & w_{41} & w_{42} & w_{43} \\ w_{14} & w_{11} & w_{12} & w_{13} \\ w_{24} & w_{21} & w_{22} & w_{23} \\ w_{34} & w_{31} & w_{32} & w_{33} \end{bmatrix}$$

W_1 的权是随机产生的，一旦产生了，其他(AD)的权都按照上述的方案向下或向右循环移动 n 次，n 为 1, 2, …, n。因此这样的结构，不管输入样本发生上下左右的变化，在子平面上 Adaline 输出的 +1 或 −1 的数目是不变的，选择器 MAJ 总是把它们的数累加起来经一定的组合输出，所以选择器 MAJ 的输出对平移是不变的。对于旋转不变可用下面四个子平面的综合获得，图 2-16*b* 为表示输入样本在旋转 90°、180°、270° 时的情况，如 s_1 平面是最左面的平面，表示没有旋转，其 W_1 的分布如式(2-43)，那么 s_2 为旋转 90° 时的情况，其权的分布如下：



(a) (b)

图 2-16 网络结构示意图

(a) 不变性网络的子平面结构；(b) 旋转不变的网络结构

$$\begin{bmatrix} R_{C1}(\boldsymbol{W}_1) & TR_1R_{C1}(\boldsymbol{W}_1) & TR_2R_{C1}(\boldsymbol{W}_1) & TR_3R_{C1}(\boldsymbol{W}_1) \\ T_{D1}R_{C1}(\boldsymbol{W}_1) & TR_1T_{D1}R_{C1}(\boldsymbol{W}_1) & TR_2T_{D1}R_{C1}(\boldsymbol{W}_1) & TR_3T_{D1}R_{C1}(\boldsymbol{W}_1) \\ T_{D2}R_{C1}(\boldsymbol{W}_1) & TR_1T_{D2}R_{C1}(\boldsymbol{W}_1) & TR_2T_{D2}R_{C1}(\boldsymbol{W}_1) & TR_3T_{D2}R_{C1}(\boldsymbol{W}_1) \\ T_{D3}R_{C1}(\boldsymbol{W}_1) & TR_1T_{D3}R_{C1}(\boldsymbol{W}_1) & TR_2TD_3R_{C1}(\boldsymbol{W}_1) & TR_3T_{D3}R_{C1}(\boldsymbol{W}_1) \end{bmatrix} \tag{2-44}$$

$R_{C1}(\boldsymbol{W}_1)$表示 $\boldsymbol{W}_1$ 矩阵旋转 90° 的权矩阵。s_3、s_4 分别是旋转 180°、270° 时的平面，它们的输出又综合到一个总的 MAJ 输出。

这样对应于一个模式(样本)，可能需要有四个(∧)子平面来保证其平移和旋转不变。对于 k 个样本，最小需要的子平面数为 $k>4\log_2 k$，其中"4"为旋转四个方向，$\log_2 k$ 为能与样本一一对应的最小编码。

第三节　非线性变换单元组成的前馈网络

由非线性变换单元组成的前馈网络，简称 B-P 网络。

一、网络的结构与数学描述

对于非线性变换单元的神经元，其输入与输出关系满足非线性单调上升的函数，重写上一章的公式，把脚标 i 与 j 交换，g 函数为 1，则有:

$$s_j=\Sigma w_i x_i-\theta_j \tag{1-1}$$

$$u_j=s_j \tag{1-2}$$

$$y_j=f(u_j) \tag{1-3}$$

其中(1-3)式在非线性变换下为

$$f(u_j)=\frac{1}{1+e^{-u_j}}=\frac{1}{1+e^{-(\Sigma w_i x_i-\theta_j)}} \tag{2-45}$$

图 2-17a 所示的是函数 $f(u)$ 的图形，可以看到 $f(u)$ 是一个连续可微的函数，它的一阶导数存在，用这种函数来区分类别时，它的结果可能是一种模糊的概念，当 $u>0$ 时，其输出不为 1，而是一个大于 0.5 的数，而当 $u<0$ 时，其输出是一个小于 0.5 的数，对一个单元组成的分类器来说，这种 $f(u)$ 函数得到的概率为 80%(>50%)，它是属于 A 类的概率，或属于 B 类的概率(=20%)，这种分割具有一定的科学性。对于多层的网络，这种 $f(u)$ 函数所划分的区域不是线性划分，是由一个非线性的超平面组成的区域，它是比较柔和、光滑的任意界面，因而它的分类比线性划分精确、合理，这种网络的容错性较好。另外一个重要的特点是由于 $f(u)$ 是连续可微的，它可以严格利用梯度法进行推算，它的权的学习解析式十分明确，它的学习算法称为反向传输算法(Back-Propagation)，简称 B-P 算法，这种网络也称为 B-P 网络。

多层 B-P 网络的结构如图 2-17b 所示，输入矢量为 $\boldsymbol{x}\in R^n$，$\boldsymbol{x}=(x_0, x_2, \cdots, x_{n-1})^T$；第二层有 n_1 个神经元 $\boldsymbol{x}'\in R^{n_1}$，$\boldsymbol{x}'=(x'_0, x'_2, \cdots, x'_{n_1-1})^T$；第三层为 n_2 个神经元，$\boldsymbol{x}''\in R^{n_2}$，$\boldsymbol{x}''=(x''_0, x''_2, \cdots, x''_{n_2-1}]^T$；最后输出神经元 $\boldsymbol{y}\in R^m$，有 m 个神经元，$\boldsymbol{y}=[y_0, \cdots, y_{m-1}]^T$，如输入与第二层之间的权为 w_{ij}，阈值为 θ_j，第二层与第三层之间的权为 w'_{jk}，阈值为 θ'_k，第三层与最后层的权为 w''_{ki}，阈值为 θ''_i，那么各层神经元的输出满足:

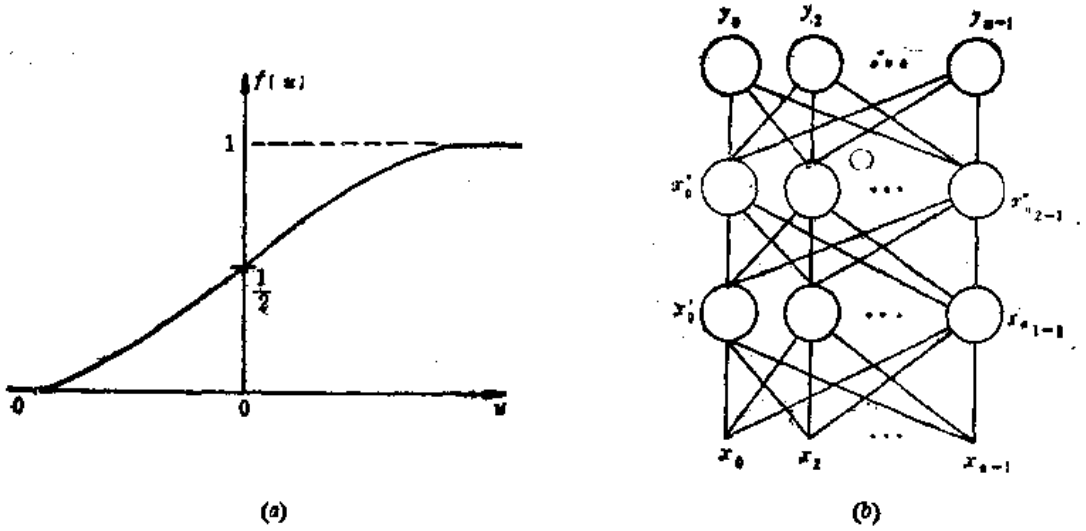


图 2-17

(a) 输入输出非线性函数；(b) 多层 B-P 网络

$$\begin{cases} y_i=f\left(\sum\limits_{k=0}^{n_2-1} w''_{ki}x''_k-\theta''_i\right) \\ x''_k=f\left(\sum\limits_{j=0}^{n_1-1} w'_{jk}x'_j-\theta'_k\right) \\ x'_j=f\left(\sum\limits_{i=0}^{n-1} w_{ij}x_i-\theta_j\right) \end{cases} \tag{2-46}$$

其中函数 f 满足(2-45)式。$f(u)$ 中的 u 是用各层输出加权求和的值，B-P 网络是完成 n 维空间向量对 m 维空间的近似映照。

若近似映照函数为 F，$\boldsymbol{x}$ 为 n 维空间的有界子集，$F(\boldsymbol{x})$ 为 m 维空间的有界子集，$\boldsymbol{y}=F(\boldsymbol{x})$ 可写为

$$F:\boldsymbol{x}\subset R^n\rightarrow\boldsymbol{y}\subset R^m$$

通过 P 个实际的映照对 $(\boldsymbol{x}^1,\boldsymbol{y}^1),(\boldsymbol{x}^2,\boldsymbol{y}^2),\cdots,(\boldsymbol{x}^P,\boldsymbol{y}^P)$ 的训练，其训练的目的是得到神经元之间的连接权 w_{ij},w'_{jk},w''_{kl} 和阈值 $\theta_j,\theta'_k,\theta''_l(i=0,1,2,\cdots,n-1;j=0,1,2,\cdots,n_1-1;k=0,1,2,\cdots,n_2-1;l=0,1,2,\cdots,m-1)$，使其映照获得成功。训练后得到的连接权，对其他不属于 $P_1=1,2,\cdots,P$ 的 $\boldsymbol{x}$ 子集进行测试，其结果仍能满足正确映照。

如果输入第 P_1 个样本对 $(\boldsymbol{x}^{P_1},\boldsymbol{y}^{P_1})$，通过一定方式训练后，得到一组权 $\boldsymbol{W}^{P_1}$，$\boldsymbol{W}^{P_1}$ 包括网络中所有的权和阈值，此时 $\boldsymbol{W}^{P_1}$ 的解不是唯一的，而是在权空间中的一个范围，也可为几个范围。对于所有的学习样本 $P_1=1,2,\cdots,P$ 都可以满足：

$$\boldsymbol{y}^{P_1}=F(\boldsymbol{x}^{P_1},\boldsymbol{W}^{P_1}) \tag{2-47}$$

各自的解为 $\boldsymbol{W}^1,\boldsymbol{W}^2,\cdots,\boldsymbol{W}^P$，通过对样本集的学习，得到满足所有样本正确映照的解为

$$\boldsymbol{W}=\bigcap_{P_1=1}^{P}\boldsymbol{W}^{P_1} \tag{2-48}$$

学习的过程就是求解 $\boldsymbol{W}$ 的过程，因为学习不一定要求很精确，所以得到的是一种近似解。这种解 $\boldsymbol{W}$ 是通过学习而得到的。假设(2-47)是一个线性方程，而且要求的未知数 $\boldsymbol{W}$ 和样本数相同，如同为 n_P，则可以直接用线性代数的方法解出这些未知数 $\boldsymbol{W}$。可是这种解没有一点容错，即在测试样本输入时，它很难联想到应该对应的输出。幸好这里 $F(\cdot)$ 不是一个线性函数，而是一个十分复杂的非线性关系，而且 $\boldsymbol{W}$ 的维数和样本数不相同，因而 $\boldsymbol{W}$

不是唯一解，而是有一定的容错范围，这使 B-P 网络比一般的线性阈值单元的网络有更大的灵活性。

二、B-P 的学习算法

B-P 算法属于 δ 学习律，是一种有教师的学习算法，输入学习样本为 P 个，$x^1, x^2, \cdots, x^P$，已知与其对应的教师为 $t^1, t^2, \cdots, t^P$，学习算法是将实际的输出 $y^1, y^2, \cdots, y^P$ 与 $t^1, t^2, \cdots, t^P$ 的误差来修改其连接权和阈值，使 y^{P_1} 与要求的 t^{P_1} 尽可能的接近。

为了方便起见，在图 2-17b 的网络中，把阈值写入连接权中去，令：$\theta''_l=w''_{n_2,l}$；$\theta'_k=w'_{n_1,k}$；$\theta'_j=w_{n_j}$；$x_{n_2}=-1$；$x_{n_1}=-1$；$x_n=-1$；则方程(2-46)改为

$$y_l=f\left(\sum_{k=0}^{n_2} w''_{kl}x''_k\right) \tag{2-46a}$$

$$x''_k=f\left(\sum_{j=0}^{n_1} w'_{jk}x'_j\right) \tag{2-46b}$$

$$x'_j=f\left(\sum_{i=0}^{n} w_{ij}x_i\right) \tag{2-46c}$$

第 P_1 个样本输入到图 2-17b 所示的网络，得到输出 y_l，$l=0, 1, \cdots, m-1$，其误差为各输出单元误差之和，满足：

$$E_{P_1}=\frac{1}{2}\sum_{l=0}^{m-1}(t_l^{P_1}-y_l^{P_1})^2$$

对于 P 个样本学习，其总误差为

$$E_{总}=\frac{1}{2}\sum_{P_1=1}^{P}\sum_{l=0}^{m-1}(t_l^{P_1}-y_l^{P_1})^2 \tag{2-49}$$

设 w_{sq} 为图 2-17b 网络中任意两个神经元之间的连接权，w_{sq} 也包括阈值在内，$E_{总}$ 为一个与 w_{sq} 有关的非线性误差函数。令

$$\varepsilon=\frac{1}{2}\sum_{l=0}^{m-1}(t_l^{P_1}-y_l^{P_1})^2=E_{P_1}$$

$$E_{总}=\sum_{P_1=1}^{P}\varepsilon(W, t^{P_1}, x^{P_1})$$

$$W=(w_{11}, \cdots, w_{sq}, \cdots, w]^T$$

采用梯度法，对每个 w_{sq} 元的修正值为

$$\Delta w_{sq}=-\sum_{P_1=1}^{P}\eta\frac{\partial\varepsilon}{\partial w_{sq}} \quad \text{其中 } \eta \text{ 为步长}$$

$$\Delta E_{总}=+\sum_{P_1=1}^{P}\sum_{sq}\frac{\partial\varepsilon}{\partial w_{sq}}\Delta w_{sq}=-\eta\sum_{P_1=1}^{P}\sum_{sq}\left(\frac{\partial\varepsilon}{\partial w_{sq}}\right)^2\leqslant 0 \tag{2-50}$$

这里用梯度法可以使总的误差向减小的方向变化，直到 $\Delta E_{总}=0$ 为止，这种学习方式其矢量 W 能够稳定到一个解，但并不保证是 $E_{总}$ 的全局最小解，可能是一个局部极小解。

具体学习算法的解析式推导如下：

令 n_0 为迭代次数，根据(2-49)式和梯度算法，可得到每一层的权的迭代公式为

$$w''_{kl}(n_0+1)=w''_{kl}(n_0)-\eta\frac{\partial E_{总}}{\partial w''_{kl}} \tag{2-51a}$$

$$w'_{jk}(n_0+1)=w'_{jk}(n_0)-\eta\frac{\partial E_{总}}{\partial w'_{jk}} \tag{2-51b}$$

$$w_{ij}(n_0+1)=w_{ij}(n_0)-\eta\frac{\partial E_{总}}{\partial w_{ij}} \tag{2-51c}$$

从(2-51a)式可以看出，w''_{kl}是第k个神经元与输出层第l个神经元之间的连接权，它只与输出层中一个神经元有关，将(2-49)式代入(2-51a)中的第二项，利用公式(2-45)得：

$$\frac{\partial E_{总}}{\partial w''_{kl}}=\sum_{P_1=1}^{P}\frac{\partial E_{P_1}}{\partial y_l^{P_1}}\frac{\partial y_l^{P_1}}{\partial u_l''^{P_1}}\frac{\partial u_l''^{P_1}}{\partial w''_{kl}}=\sum_{P_1=1}^{P}(t_l^{P_1}-y_l^{P_1})f'(u_l''^{P_1})x_k''^{P_1} \tag{2-52}$$

这里 $u_l''^{P_1}=\sum_{k=0}^{n_3}w''_{kl}x_k''^{P_1}$

$x_k''^{P_1}$为P_1样本输入网络时，x''_k的输出值。

$$f'(u_l''^{P_1})=\frac{e^{-u_l''^{P_1}}}{(1+e^{-u_l''^{P_1}})^2}=f'(u_l''^{P_1})[1-f(u_l''^{P_1})]=y_l^{P_1}(1-y_l^{P_1}) \tag{2-53}$$

将(2-53)、(2-52)代入(2-51a)，得

$$w''_{kl}(n_0+1)=w''_{kl}(n_0)+\eta\sum_{P_1=i}^{P}\delta_{kl}^{P_1}x_k''^{P_1} \tag{2-54}$$

这里的$\delta_{kl}^{P_1}=(t_l^{P_1}-y_l^{P_1})y_l^{P_1}(1-y_l^{P_1})$

对于中间隐层，根据(2-51b)式有：

$$\Delta w'_{jk}=-\eta\frac{\partial E_{总}}{\partial w'_{jk}}$$

而

$$\begin{aligned}\frac{\partial E_{总}}{\partial w'_{jk}}&=-\sum_{P_1=1}^{P}\sum_{l=0}^{m-1}(t_l^{P_1}-y_l^{P_1})\frac{\partial y_l^{P_1}}{\partial u_l''^{P_1}}\frac{\partial u_l''^{P_1}}{\partial x_k''^{P_1}}\frac{\partial x_k''^{P_1}}{\partial u_j'^{P_1}}\frac{\partial u_j'^{P_1}}{\partial w'_{jk}}\\&=-\sum_{P_1=1}^{P}\sum_{l=0}^{m-1}(t_l^{P_1}-y_l^{P_1})f'(u_l''^{P_1})w''_{kl}x_k''^{P_1}(1-x_k''^{P_1})x_j'^{P_1}\\&=-\sum_{P_1=1}^{P}\sum_{l=0}^{m-1}\delta_{kl}^{P_1}w''_{kl}x_k''^{P_1}(1-x_k''^{P_1})x_j'^{P_1}\\&=-\sum_{P_1=1}^{P}\delta_{jk}^{P_1}x_j'^{P_1}\end{aligned} \tag{2-55}$$

其中：

$$\delta_{jk}^{P_1}=\sum_{l=0}^{m-1}\delta_{kl}^{P_1}w''_{kl}x_k''^{P_1}(1-x_k''^{P_1})$$

所以

$$w'_{jk}(n_0+1)=w'_{jk}(n_0)+\eta\sum_{P_1=1}^{P}\delta_{jk}^{P_1}x_j'^{P_1} \tag{2-56}$$

同理可得

$$w_{ij}(n_0+1)=w_{ij}(n_0)+\eta\sum_{P_1=1}^{P}\delta_{ik}^{P_1}x_j'^{P_1} \tag{2-57}$$

其中：

$$\delta_{ij}^{P_1}=\sum_{k=0}^{n_2}\delta_{jk}^{P_1}w'_{jk}x_j'^{P_1}(1-x_j'^{P_1})$$

公式(2-54)、(2-56)、(2-57)是多层B-P网络各层之间权修正的基本表达式，由于权的修正是在所有样本输入后，计算其总的误差后进行的，这种修正也被称为批处理。批处理修正可以保证其$E_{总}$向减小的方向变化，在样本数多的时候，它比分别处理时的收敛速度快。

整个网络学习过程分为两个阶段。第一个阶段是从网络的底部向上进行计算，如果网络的结构和权已设定，输入已知学习样本，可按公式(2-46a、b、c)计算每一层的神经元输

出。第二个阶段是对权和阈值的修改，这是从最高层向下进行计算和修改，从已知最高层的误差修改与最高层相联的权，然后按公式(2-54)、(2-56)、(2-57)修改各层的权，两个过程反复交替，直到达到收敛为止，具体步骤如下，网络的结构仍如图 2-17*b* 所示：

(1) 在初始的时候，图 2-17*b* 各层的权和阈值用一个随机数加到各层上，作为初值，使

$$w_{sq}(0)=\text{Random}(\cdot)\quad sq\text{ 为 }ij,jk,kl$$

(2) 在已知 P 个学习样本中，顺序取样本输入到图 2-17*b* 的网络中，先取一个输入 $P_1=1$。

(3) 按公式(2-46*a*、*b*、*c*)计算 x_j',x_k'',y_l。

(4) 求出各层的误差，对已知样本的教师可得

$$\delta_{kl}^{P_1}=(t_l^{P_1}-y_l^{P_1})y_l^{P_1}(1-y_l^{P_1})$$

$$\delta_{jk}^{P_1}=\sum_{l=0}^{n-1}\delta_{kl}^{P_1}w_{kl}''x_k''^{P_1}(1-x_k''^{P_1})$$

$$\delta_{ij}^{P_1}=\sum_{k=0}^{n_2}\delta_{jk}^{P_1}w_{jk}'x_j'^{P_1}(1-x_j'^{P_1})$$

并记下各个 $x_k''^{P_1},x_j'^{P_1},x_i^{P_1}$ 的值。

(5) 记下学习过的样本集次数 P_1，即计数为 P_1+1，看 P_1+1 是否等于 P，如没有达到 P，回到步骤(2)继续计算，否则再从第一个输入样本开始让 $P_1=1$，进行步骤(6)。

(6) 按公式(2-54)、(2-56)、(2-57)修改各层的权和阈值。

(7) 按新的权计算 x_j',x_k'',y_l 和 E_{P_1}，根据要求如果对每个 P_1 和 l 都满足：

$$|t_l^{P_1}-y_l^{P_1}|<\varepsilon \tag{2-58}$$

若 ε 为大于 0 的一个给定小数，则学习停止，否则重复步骤(2)，重新修改权，直到(2-58)式满足为止。

在(2-58)式中的 ε，是根据要求的精度和分类的可信度来定的，例如用作分类器的 B-P 网络，希望输出 $t_l=1$ 表示属于某一类的，而 $t_l=0$ 表示不属于某一类的，但是从输入输出函数 $f(u)$ 的(2-45)式中可看出，只有 $u=\pm\infty$ 时才能达到 1 和 0，因此 1和0 可以根据用户定义，例如 $y_l>0.7$ 为 1，$y_l<0.3$ 为 0，此时误差 $\varepsilon=(0.3)^2=0.09$，对用于函数逼近的 B-P 网络可根据逼近精度来定义 ε。

[例 2-1] 图 2-18 是具有两个输入，两个隐单元和一个输出的多层神经网络，如学习样本为 $P=4$，它的权的标号如图上所示，按上述的步骤计算：

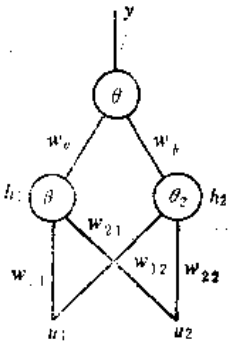


图 2-18 简单 B-P 网络

(1) 先任意取初始值：

$w_{11}(0),w_{12}(0),w_{21}(0),w_{22}(0),\theta_1(0),\theta_2(0),\theta(0)$ 和 $w_a(0),w_b(0)$

(2) 在样本集中取 $u_1^1,u_1^1,\cdots,u_1^4,u_1^4$ 输入图 2-18 的网络，正向计算隐单元输出：

$$h_1^{P_1}=f(w_{11}u_1^{P_1}+w_{21}u_1^{P_1}-\theta_1)\quad P_1=1,2,3,4$$

$$h_2^{P_1}=f(w_{12}u_1^{P_1}+w_{22}u_1^{P_1}-\theta_2)$$

输出单元为

$$y=f(w_ah_1^{P_1}+w_bh_2^{P_1}-\theta)$$

(3) 根据要求的输出值 t^1,t^2,t^3,t^4 计算各层的误差，输出 y 的误差：

$$\delta^{P_1}=y^{P_1}(1-y^{P_1})(t^{P_1}-y^{P_1})$$

中间隐单元：

$$\delta_i^{P_1}=\delta^{P_1}w_a\cdot h_i^{P_1}(1-h_i^{P_1})$$
$$\delta_i^{P_1}=\delta^{P_1}w_b\cdot h_i^{P_1}(1-h_i^{P_1})$$

(4) 权的修改公式，可以表示如下：

$$w_a(n_0+1)=w_a(n_0)+\eta\sum_{P_1=1}^{4}\delta^{P_1}h_i^{P_1}$$

$$w_b(n_0+1)=w_b(n_0)+\eta\sum_{P_1=1}^{4}\delta^{P_1}h_i^{P_1}$$

$$\theta(n_0+1)=\theta(n_0)+\eta\sum_{P_1=1}^{4}\delta^{P_1}(-1)$$

$$w_{11}(n_0+1)=w_{11}(n_0)+\eta\sum_{P_1=1}^{4}\delta_i^{P_1}u_i^{P_1}$$

$$w_{21}(n_0+1)=w_{21}(n_0)+\eta\sum_{P_1=1}^{4}\delta_i^{P_1}u_i^{P_1}$$

$$w_{12}(n_0+1)=w_{12}(n_0)+\eta\sum_{P_1=1}^{4}\delta_i^{P_1}u_i^{P_1}$$

$$w_{22}(n_0+1)=w_{22}(n_0)+\eta\sum_{P_1=1}^{4}\delta_i^{P_1}u_i^{P_1}$$

$$\theta_1(n_0+1)=\theta_1(n_0)+\eta\sum_{P_1=1}^{4}\delta_i^{P_1}\cdot(-1)$$

$$\theta_2(n_0+1)=\theta_2(n_0)+\eta\sum_{P_1=1}^{4}\delta_i^{P_1}\cdot(-1)$$

(5) 计算输出误差：

$$E_{总}=\frac{1}{2}[(t^1-y^1)^2+(t^2-y^2)^2+\cdots+(t^4-y^4)^2]$$

存在一个 ε，使 $(t^{P_1}-y^{P_1})^2<\varepsilon$，否则重复上面步骤，直到满足要求为止。

三、B-P 网络的误差曲面讨论

B-P 网络的误差公式为

$$E_{总}=\frac{1}{2}\sum_{P_1}\sum_{i}(t_i^{P_1}-y_i^{P_1})^2,$$

$y_i=f(u_i)$ 是一个非线性函数，而多层的 B-P 网络中 u_i 又是前一层神经元的非线性函数，用 ε 表示其一个样本 P_1 的误差，则

$$\varepsilon(\boldsymbol{W},\boldsymbol{t}^{P_1},\boldsymbol{x}^{P_1}),\quad E_{总}=\frac{1}{2}\sum_{P_1}\varepsilon(\boldsymbol{W},\boldsymbol{t}^{P_1},\boldsymbol{x}^{P_1}) \tag{2-59}$$

$E_{总}$ 与权 $\boldsymbol{W}$ 有关，与输入的学习样本和输出样本有关，如暂且不考虑样本的问题，$E_{总}$ 是一个与权矢量相关的函数，在多层 B-P 网络中，权空间的维数为 n_w：

$$n_w=[i(j+1)]+[j(k+1)]+(k(m+1)]$$

在 n_w+1 维的空间中，误差 $E_{总}$ 是一个具有极其复杂形状的曲面，如果再考虑输入的样本，则 $E_{总}$ 的形状就更难想象了。对这样的曲面求其梯度，其结果不像线性阈值单元的网络那么简单，例如：对权空间某一维求梯度：

$$\frac{\partial E_{总}}{\partial w_{sq}}=-\sum_{P_1}\sum_{l}(t_l^{P_1}-y_l^{P_1})y_l^{P_1}(1-y_l^{P_1})\frac{\partial u_l^{P_1}}{\partial w_{sq}} \tag{2-60}$$

再简单一些，就对输出层第 l 单元的 w''_{kl} 来说，因为 l 已经定了，因而对 w''_{kl} 的梯度与其他输出无关，式(2-60)为

$$\frac{\partial E_{总}}{\partial w''_{kl}}=-\sum_{P_1}(t_l^{P_1}-y_l^{P_1})y_l^{P_1}(1-y_l^{P_1})x''_k$$

从上面的梯度公式可以看出，该式包含两个因子，一个是$(t_l^{P_1}-y_l^{P_1})$因子，另一个是$y_l^{P_1}(1-y_l^{P_1})$因子，在(2-60)式中对于中间层的单元，偏微分 $\frac{\partial u_l^{P_1}}{\partial w_{sq}}$ 中还有 $x_k''^{P_1}(1-x_k''^{P_1})$ 因子。当梯度为零时，并不完全说明网络已经达到要求的 $E_{总}$ 的最小点。梯度为零可能有以下几种情况：

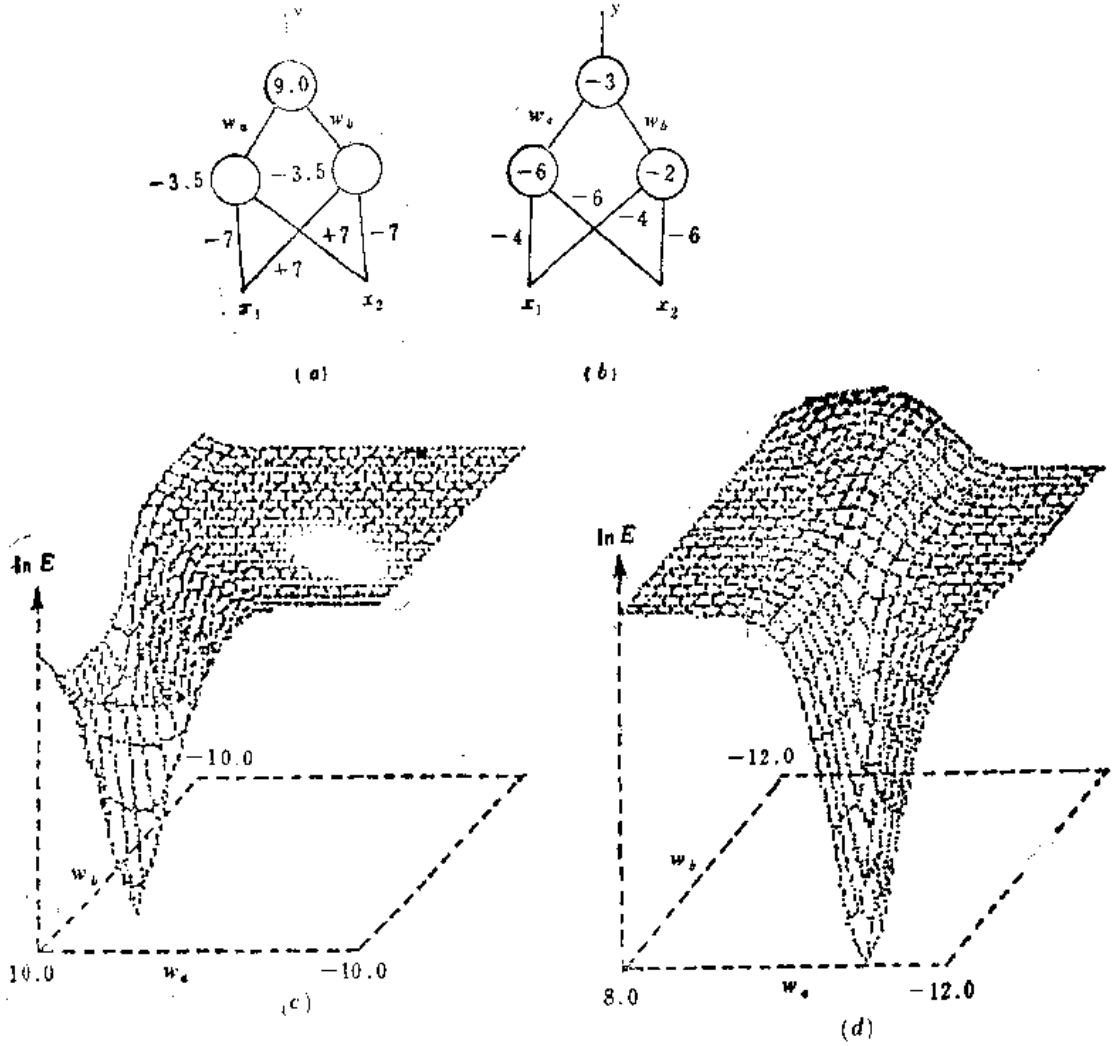


图 2-19 在不同权初值条件下，异或非的网络及其误差曲面

(a) 同或的 B-P 网络；(b) 异或非的 B-P 网络；(c) 网络(a)在 w_a, w_b 变化时的误差曲面；(d) 网络(b)在 w_a, w_b 变化时的误差曲面

(1) $(t_l^{P_1}-y_l^{P_1})\to 0$，对每一个 P_1 和 l 都满足。这种情况下，$E_{总}$ 是可以减得比较小，达到 $E_{总}<\varepsilon'$ 或 $|t_l^{P_1}-y_l^{P_1}|<\varepsilon$ 的目的。此时的解为全局最小点。通过梯度法我们可以取得满足要求的一组解，但是由于在曲面上存在很多满足 $E_{总}<\varepsilon'$ 的解，因而在不同初始权的条件下，

学习出来的解很不相同。图 2-19*a*,*b* 表示在两个不同初值条件下、同一结构的 B-P 网络学习同或的情况。设输入(x_1,x_2)分别为(0,0)、(1,1)时，输出为 1；输入(x_1, x_2)分别为(1, 0)、(0, 1)时，输出为 0。在固定网络的其他参数的条件下，讨论 w_a，w_b 与误差的关系，用 $\ln E$ 表示网络输出的对数误差，当 w_a,w_b 改变的时候，其误差曲面如图 2-19*c*,*d* 所示。对图 2-19*a* 的网络，改变 w_a，w_b 引起的 $\ln E$ 变化如图 2-19*c* 所示。对于图 2-19*b* 的网络，改变 w_a，w_b 引起的 $\ln E$ 变化如图 2-19*d* 所示。这里我们只讨论了两个权。图 2-19*a*，*b* 网络，它有 9 个权，可以想象它的解有无数个。

(2) 在式(2-60)中，还有一个因子是 $y_i^{P_1}(1-y_i^{P_1})\to 0$，这是由于 $y_i=f(\Sigma w''_{ki}x''_k)$ 中 $\Sigma w''_{ki}x''_k$ 的绝对值很大而造成的，$f(\cdot)$是一个指数型函数，在 $e^{-\Sigma w''_{ki}x''_k}$ 中，$|\Sigma w''_{ki}x''_k|>3$ 时，指数上升和下降十分缓慢，因而 $f(\cdot)$处于接近于 +1 或 0 的平坦线上，此时 $\Sigma w''_{ki}x''_k$ 的变化对输出函数不太敏感，存在一些平坦区域，如图 2-19*c*、*d* 上，都可以看到这种平坦区域，此时用梯度法调整权的范围很小，而$(t_i^{P_1}-y_i^{P_1})$仍然很大，$E_{总}$ 没有达到要求而小于 e' 值，但是梯度却已经很小了；若增加迭代次数，只要调整的方向正确，那么经过一段较长的时间后，总可以从这个平坦区中退出来，进入一些全局的近似极小点，如图 2-19*c* 中的第三象限。图 2-19*d* 中的第一象限，当初值落到这些平坦区，但调整方向是向着那些全局极小点的，只要有足够的迭代次数，仍可落到所要求的解上。在有很多输出单元的 B-P 网络中，$E_{总}$ 是每个单元误差之和，如其中有某些单元进入了平坦区，又是在不同的时刻退出平坦区，在误差曲面上会出现一个一个的阶梯，如一个单元退出了平坦区落到梯度比较大的区域，$E_{总}$ 会突然减小一个阶梯，随后进入等二个平坦区，又经过一个长过程，再很快减小，这样最后还可以收敛到要求的值。对于不同初值的情况下，收敛的方向及在误差曲面上的情况可能是完全不相同的。这种阶梯对于不同的样本 P_1 同样存在。

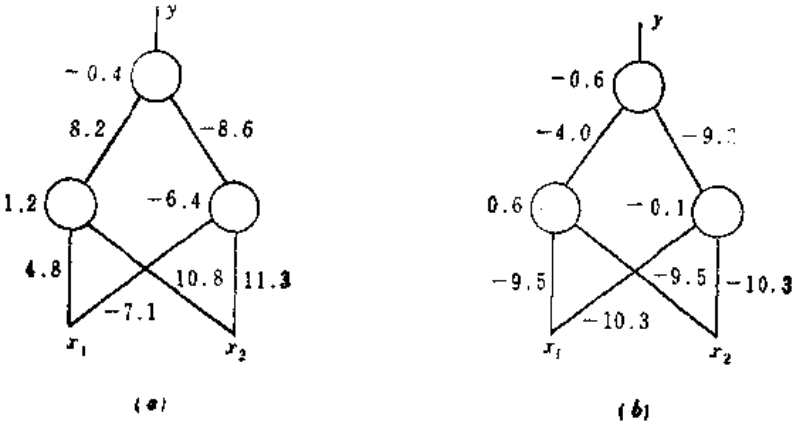


图 2-20　存在局部最小 XOR 网络

(8) 在误差曲面上存在一些局部极小点，当收敛到这种局部极小点，无论经过多长时间，学习不能达到其要求的解上，局部极小点主要分为两类，一类是：虽然$(t_i^{P_1}-y_i^{P_1})$不为零，$y^{P_1}(1-y^{P_1})$也不为零，但是其对各样本的和为零，这种局部极小点称为第一类局部最小。如图 2-20 中的两种情况下的 XOR 问题，设输入(x_1, x_2)分别为(0, 0)、(1, 1)，其输出应为 0，输入(x_1, x_2)分别为(0, 1)、(1, 0)时，其输出应为 1，但他们却落入了局部最小，对应 *a* 组的总的误差值为 $E_{总}=0.5$，*b* 组为 $E_{总}=0.664$。分析图 2-20*a*，对于输入(x_1, x_2)分别为(0, 0)

(1, 0)时，能够产生正确的输出，而对应于输入(x_1, x_2)为(1, 1)和(0, 1)时，其输出值非常接近0.5，对于不同的输入样本(1, 1)和(0, 1)，其隐单元产生几乎一样的输出，这导致网络的输出值也相同。若不考虑两个正确样本，此时$E_{总}=\frac{1}{2}[(1-y)^2+y^2]$，$y=0.5$时，$E_{总}\to\min$。当然这一类局部最小的原因也可能是因为隐单元已进入平坦区而造成的。第一类局部极小点满足：

$$\frac{\partial E_{总}}{\partial w_{sq}}=\sum_{P_1}\delta_{P_1}x_k''\to 0$$

$$\text{而 } \delta_{P_1}x_k''\neq 0$$

在误差曲面上存在这些第一类局部极小点，但陷入第一类局部极小点的概率不大，尤其是样本数很多的情况下很难碰到。

第二类局部最小是处在误差曲面的平坦区上，而梯度方向是离开全局极小点，这种局部最小点，使权向 $\pm\infty$ 的方向上趋近，其极小点的位置在无穷远。下面用一个简单的例子来说明。

一个一维的分类器，它的样本是5个点：$(x_1, x_2, x_3, x_4, x_5)$，在一个实轴上，其值为0.5，2.5，5，10和25。25这一点属于一类，其神经元输出为1，而其他点为另一类，其神经元输出为0，用单输入的单个神经元及B-P的算法进行学习，其方程为

$$u(x_i)=wx_i+\theta$$

$$y(u_i)=\frac{1}{1+e^{-[u(x_i)]}}$$

误差：

$$E=\frac{1}{2}\sum_{i=1}^{5}[t_i-y(x_i)]^2$$

这里简单地将θ取正号以便于说明。

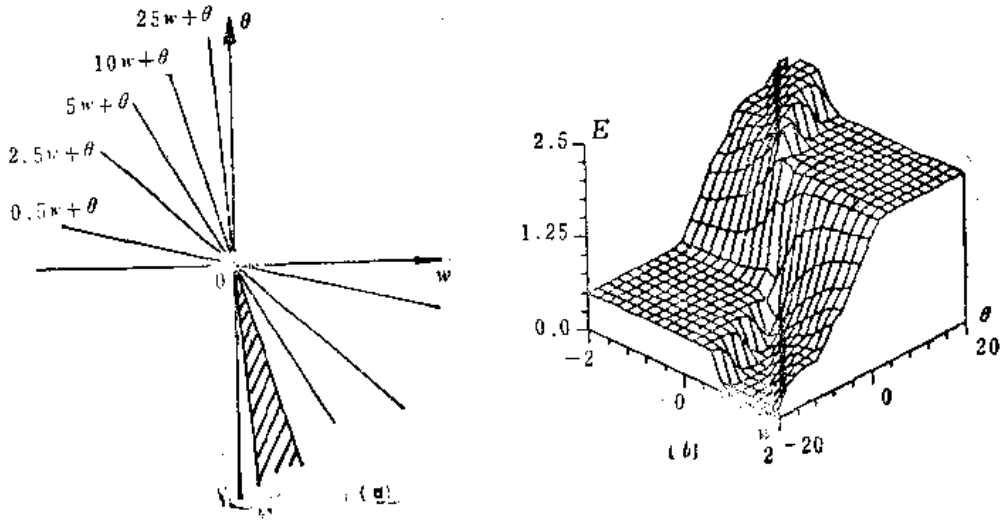


图 2-21

(a) 系数空间；(b) 误差曲面

要调节的参数为w和θ，在一个二维参数空间上，不同的样本输入，都是此空间的一根直线，如图2-21a所示，满足条件的解很多，在第四象限阴影部分都能满足分类器的要求，以

0.5 为界线，当 x_1, x_2, x_3, x_4 输入时，可使 $y(x_i)<\frac{1}{2}$，x_5 输入时，可使 $y(x_5)>\frac{1}{2}$。其 E 的梯度为

$$\frac{\partial E}{\partial w}=-\sum_{i=1}^{5}[t-y(x_i)]y(x_i)[1-y(x_i)]x_i$$

$$\frac{\partial E}{\partial \theta}=-\sum_{i=1}^{5}[t-y(x_i)]y(x_i)[1-y(x_i)]$$

考虑参考初值落在参数空间第三象限中间的部分，即 $\theta<0$，$w<0$，并且 θ 和 w 的大小相近，可以计算出其梯度 $\frac{\partial E}{\partial w}$ 和 $\frac{\partial E}{\partial \theta}$ 都是大于零的，因而使用下式：

$$w(n_0+1)=w(n_0)-\eta\frac{\partial E}{\partial w}$$

$$\theta(n_0+1)=\theta(n_0)-\eta\frac{\partial E}{\partial w}$$

进行修改时，w 和 θ 向更小的方向发散，使 w、θ 越来越负，直到使 (w, θ) 达到 $(-\infty, -\infty)$ 时，E 也会变为零，但是此时当 x_5 输入时，其 $y(x_5)=0$，不满足分类要求，从图 2-21b 看到不同的 w、θ 时 E 的误差曲面，有两个平坦区，而全局最小落在第四象限，在参数 w、θ 的初值落在第三象限大部分或第二象限一部分的区域时，用 B-P 算法迭代后使 w、θ 达到了负无穷大，都可能落到这种局部最小值上。

分析一下这一类局部最小点的出现，是由于 $x_5=25$ 这个很大的输入样本致使第五个样本输入时 $y(x_i)$ 很快达到与零很接近的值，此时 $y(x_5)[1-y(x_5)]$ 变得很小，这个样本对梯度的贡献很小，而初值又落在第三象限的平坦区，且其他样本对梯度的贡献较大，最终结果被其他样本拉到了一边，而使学习陷入了局部极小。

对于 B-P 网络的误差曲面，有以下三个特点：第一，有很多全局的最小的解。第二，存在一些平坦区，在此区内误差改变很小，这些平坦区多数发生在神经元的输出接近于 0 或 1 的情况下，对于不同的映照，其平坦区的位置、范围各不相同，有的情况下，误差曲面会出现一些阶梯状。第三，存在不少局部最小点，在某些初始值的条件下，算法的结果会陷入局部最小，使算法不收敛。

四、算法的改进

1. 变步长的算法

B-P算法是在梯度法的基础上推算出来的，在一般最优梯度法中，步长 η 是由一维搜索求得的，求解有下面几个步骤：

(1) 给定初始权的点 $\boldsymbol{w}(0)$ 和允许误差 $\varepsilon>0$。

(2) 计算误差 $E_总$ 的负梯度方向 $\boldsymbol{d}^{(n)}=-\nabla E_总(\boldsymbol{w})$。

(3) 若 $\|\boldsymbol{d}^{(n)}\|<\varepsilon$，则停止计算；否则从 $\boldsymbol{w}(n_0)$ 出发，沿 $\boldsymbol{d}^{(n)}$ 作一维搜索，求出最优步长 $\eta(n_0)$，$E_总[\boldsymbol{w}(n_0)+\eta(n_0)\boldsymbol{d}(n_0)]=\min E_总[\boldsymbol{w}(n_0)+\eta\boldsymbol{d}(n_0)]$。

(4) 进行权的迭代：$\boldsymbol{w}(n_0+1)=\boldsymbol{w}(n_0)+\eta(n_0)\boldsymbol{d}(n_0)$ 并转 (2)。但是在 B-P 算法中步长 η 是不变的，其原因是由于 $E_总$ 是一个十分复杂的非线性函数，很难通过最优求极小的方向得到最优的步长 η，同时如果每一步都要求计算 y，这使计算量变得很大。可是从 B-P

网络的误差曲面看出，有平坦区存在，如在平坦区上 η 太小使迭代次数增加，而当 w 落到误差剧烈变化的地方，步长太大又使误差增加，反而使迭代次数增加影响了学习收敛的速度。变步长的方法可以使步长得到合理的调节。这里推荐一种方法：先设一初始步长，若一次迭代后误差函数 E 增大，则将步长乘以小于 1 的常数 β 沿原方向重新计算下一个迭代点，若一次迭代后误差函数 E 减小，则将步长乘一个大于 1 的常数 φ，这样既不增加太多的计算量，又使步长得到合理的调整。

$$\begin{aligned} &\eta=\eta\varphi \quad \varphi>1 \qquad \text{当 } \Delta E<0 \\ &\eta=\eta\beta \quad \beta<1 \qquad \text{当 } \Delta E>0 \end{aligned} \tag{2-61}$$

这里 φ、β 为常数，$\Delta E=E_s(n_0)-E_s(n_0-1)$。

当然变步长也可用其他方法进行。很多文章都讨论了这种步长改变和选择的方法。确定了步长后，可得到其迭代公式：

$$\boldsymbol{w}(n_0+1)=\boldsymbol{w}(n_0)+\eta(n_0)\boldsymbol{d}(n_0) \tag{2-62}$$

2．加动量项

为了加速收敛和防止振荡，在许多文献中都建议引入一个动量因子 α：

$$\boldsymbol{w}(n_0+1)=\boldsymbol{w}(n_0)+\eta(n_0)\boldsymbol{d}(n_0)+\alpha\Delta\boldsymbol{w}(n_0) \tag{2-63}$$

其中第三项是记忆上一时刻权的修改方向，而在时刻 (n_0) 的修改方向为 (n_0-1) 时刻的方向与 (n_0) 时刻方向的组合。将 (2-63) 式改写为

$$\begin{aligned} \boldsymbol{w}(n_0+1)&=\boldsymbol{w}(n_0)+\eta(n_0)\left[\boldsymbol{d}(n_0)+\frac{\alpha}{\eta(n_0)}\Delta\boldsymbol{w}(n_0)\right] \\ &=\boldsymbol{w}(n_0)+\eta(n_0)\left[\boldsymbol{d}(n_0)+\frac{\alpha\eta(n_0-1)}{\eta(n_0)}\boldsymbol{d}(n_0-1)\right] \end{aligned}$$

上式的形式类似于共轭梯度法的算式，但这里的 $\boldsymbol{d}(n_0-1)$ 和 $\boldsymbol{d}(n_0)$ 并不是共轭的，而 $0<\alpha<1$，因此建议在 $\eta(n_0)$ 进行调整时，碰到 $\Delta E>0$，η 要减小时，让 $\alpha=0$，然后调节到 η 增加时使 α 恢复。

3．加入 $\gamma_l^{P_1}$ 因子的情况

从 (2-60) 式可以看到，对于任一个 P_1 和 l 碰到 $y_l^{P_1}(1-y_l^{P_1})$ 的因子趋于零，而 $(t_l^{P_1}-y_l^{P_1})\neq 0$ 时，会产生第二类局部最小，这就希望对所碰到的局部最小或平坦区的误差函数有一定的改变，使 $y_l^{P_1}$ 迅速退出不灵敏区。这儿加入一个因子 $\gamma_l^{P_1}$，使输出为

$$\begin{aligned} y_l^{P_1}&=\frac{1}{1+\exp(-\alpha)} \\ &=\frac{1}{1+\exp\left[-\sum_k(w_{lk}''x_k''^{P_1}+\theta_l'')/\gamma_l^{P_1}\right]} \end{aligned} \tag{2-64}$$

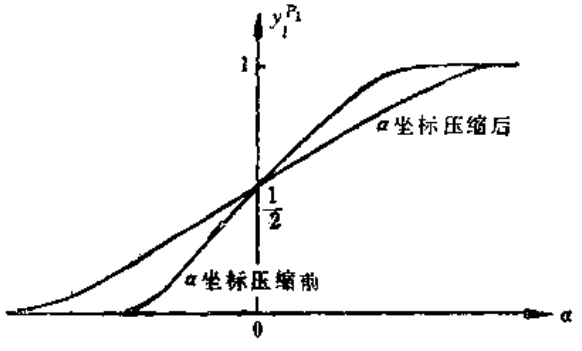


图 2-22　α 压缩前后的神经元输出曲线

在对任一个 l，P_1，碰到进入第二类的局部极小，或平坦区，使 w_{lk}'' 和 θ_l'' 同时缩小一个因子，$\gamma_l^{P_1}>1$，这样可使 $y_l^{P_1}$ 梯度脱离零值，离开平坦区，图 2-22 表示 α 坐标改变前后的神经元输出函数曲线。在离开局部最小后，再恢复 $\gamma_l^{P_1}=1$，这个方法对于避免第二类局部最小十分有效，由于第二类局部最

小的概率大，所以用这方法可以避免大部分的局部极小值。而使算法的收敛速度变快。

4. 模拟退火方法

在所有的权上加一个噪声，改变误差曲面的形状，再用模拟退火的方法，退出局部最小，这种方法可以避免陷入局部最小，但收敛速度很慢，现在很少有人采用。

综上所述，可以得到改进的 B-P 学习算法：

$$w_{sq}(n_0+1)=w_{sq}(n_0)+\eta(n_0)\sum_p\delta_{sq}^{p_1}x_i^{p_1}+\alpha\Delta w_{sq}(n_0)$$

对输出层：$sq=kl$，

$$\delta_{kl}^{p_1}=(t_l^{p_1}-y_l^{p_1})y_l^{p_1}(1-y_l^{p_1})/\gamma_l^{p_1} \tag{2-65}$$

对隐层：$sq=jk, ij$，

$$\left.\begin{aligned}\delta_{jk}^{p_1}&=\sum_{l=0}^{m-1}\delta_{kl}^{p_1}w_{kl}''x_{kl}''^{p_1}(1-x_k''^{p_1})/\gamma_k^{p_1}\\ \delta_{ij}^{p_1}&=\sum_{k=0}^{n_1}\delta_{jk}^{p_1}w_{jk}'x_j'^{p_1}(1-x_j'^{p_1})/\gamma_j^{p_1}\end{aligned}\right\} \tag{2-66}$$

$$\left.\begin{aligned}&\text{当 } \Delta E_{总}<0 \quad \eta(n_0+1)=\eta(n_0)\cdot\varphi \quad \alpha=\alpha\\ &\text{当 } \Delta E_{总}>0 \quad \eta(n_0+1)=\eta(n_0)\cdot\beta \quad \alpha=0\end{aligned}\right\} \tag{2-67}$$

其中 $\varphi>1, \beta<1, \Delta E(n_0)=E(n_0)-E(n_0-1)$

在遇到局部最小时，可以通过调节 $\gamma_l^{p_1}, \gamma_k^{p_1}, \gamma_j^{p_1}$ 来克服：将 $\gamma_l^{p_1}, \gamma_k^{p_1}$ 和 $\gamma_j^{p_1}$ 分别都调节为大于 1。

五、B-P 网络的设计考虑

1. 输入与输出层的设计

B-P 网络的输入、输出层维数是完全根据使用者的要求来设计，如果 B-P 网络用作分类器，其类别数为 m 个，那么输出一般取 m 个神经元，其训练样本集中的 x^{p_1} 属于第 j 类，要求其输出为

$$y=(0,0,0,\overset{j}{1},0,0,\overset{m}{0})^T$$

即第 j 个输出为 1，其他输出为 0，因而对一个 n 维的输入，$x\in R^n$ 进行分类映照 $\rightarrow y\in R^m$，满足：

$$y_l=1 \quad x^{p_1}\text{属于 } l \text{ 类}$$
$$y_l=0 \quad x^{p_1}\text{不属于 } l \text{ 类}$$

输出神经元还可根据类别进行编码，即 m 类的输出只要用 $\log_2 m$ 个输出单元即可。

输入的神经单元可以根据需要求解的问题和数据表示的方式而定，如果输入的是电压波形，那么输入单元可根据电压波形的采样数值和采样点数来决定输入单元的维数，也可以用一个单元输入，但输入样本为采样的时间序列。如果输入为图像，则输入单元可以为图像的像素，也可以为经过处理后的图像特征。总之问题确定之后，则输入与输出层的单元数就定了。只是在设计中应注意尽可能减小系统的规模，使学习的时间和系统的复杂性减小。

2. 隐层的数目

在本节开始时，隐层是用两层来进行的。1989 年 Robert Hecht-Nielson 证明了对于任何在闭区间内的一个连续函数都可以用一个隐层的 B-P 网络来逼近，因而一个三层的 B-P 网络可以完成任意的 n 维到 m 维的映照。

以数学上 Weierstrass 的两个逼近定理为依据，这两个定理是：

定理 A 任意给定一个连续函数 $g\in C(a,\ b)$及 $\varepsilon>0$，存在一个多项式 $P(x)$，使 $|g(x)-P(x)|<\varepsilon$，对每个 $x\in[a,b]$成立。

定理 B 任意给定一个函数 $g\in C_{2\pi}$($C_{2\pi}$ 是以 2π 为周期的连续函数)，及 $\varepsilon>0$，存在三角函数多项式 $T(x)$，使得 $|g(x)-T(x)|<\varepsilon$，对每个 $x\in R$ 成立。

推理 C 在 n 维空间中，任一向量 x 都可以由基集$\{e_i\}$表示，$x=C_1e_1+C_2e_2+\cdots+C_ne_n$，同样在有限区间内 $a\in[a,b]$ 的一个函数 $g(x)$，可以用一个正交函数序列 $\{\phi_i(x)\}$ 来表示。如果基函数可以扩展到任意大，那么

$$g(x)=C_1\varphi_1(x)+C_2\varphi_2(x)+\cdots+C_n\varphi_n(x)$$

如果正交基函数是有限项，那么

$$g(x)=C_1\varphi_1(x)+C_2\varphi_2(x)+\cdots+C_n\varphi_n(x)+\varepsilon$$

$\{\varphi_i(x)\}$是正交的，可以用傅里叶级数的三角函数展开，$C_1,\cdots,C_n$ 为傅里叶级数的系数。

利用推理，对于一个任意给定的一维连续函数 $g(x)$，$x\in[0,1]$，可以用一个傅里叶级数来近似，表示为

$$g_F(x)=\sum_k C_k\exp(2\pi ikx) \tag{2-68a}$$

其中

$$C_k=\int_{[0,1]}g(x)\exp(-2\pi ikx)\,\mathrm{d}x$$

则有

$$|g(x)-g_F(x)|<\varepsilon,\quad \text{对每个 } x \text{ 成立。}$$

进一步考虑 x 为一个 n 维空间的矢量，在$[0,1]^n\in R^n$ 进行映照 $g':[0,1]^n\in R^n\to R$，如果积分$\int_{[0,1]^n}|g'(x)|^2\mathrm{d}x$ 存在，可以根据傅里叶级数理论，仍旧存在一个级数：

$$\begin{aligned}g'_F(x,N,g')&=\sum_{k_1=-N}^{N}\sum_{k_2=-N}^{N}\cdots\sum_{k_n=-N}^{N}C_{k_1k_2\cdots k_n}\exp\Big(2\pi i\sum_{i=1}^{n}k_ix_i\Big)\\&=\sum_K C_K\exp(2\pi ik\cdot x)\end{aligned} \tag{2-68b}$$

$$C_{k_1,k_2,\cdots,k_n}=C_k=\int_{[0,1]^n}g'(x)\exp(-2\pi ik\cdot x)\,\mathrm{d}x$$

当 $N\to\infty$ 时，满足

$$\lim_{N\to\infty}\int_{[0,1]^n}|g'(x)-g'_F(x,N,g')|\,\mathrm{d}x=0$$

现考虑对一个任意多维函数的映照，给定一个函数 $h(x)$，$x\in R^n$，$[0,1]^n\subset R^n\to R^m$，其 $h(x)=[h_1(x),\ h_2(x),\cdots,h_m(x)]^T$，则 h 中的每一个分量也都可以用(2-68a)式中的傅里叶级数来近似，那么可以得到下面的定理：

映照定理：给定任一个 $\varepsilon>0$，一个连续函数矢量 h，其矢量中的每个元满足 $\int_{[0,1]^n}|h_i(x)|^2\mathrm{d}x$ 存在，h：$[0,1]^n\subset R^n\to R^m$，必定存在一个三层 B-P 神经网络来逼近函数 h，使逼近误差在 ε 之内。

证明 在 h 中取其分量 $h_i(x)$，根据(2-68a)，则有

$$\int_{[0,1]^n}|h_i(x)-\sum_k C_k\exp(2\pi ik\cdot x)|^2\mathrm{d}x<\delta_1 \tag{2-69}$$

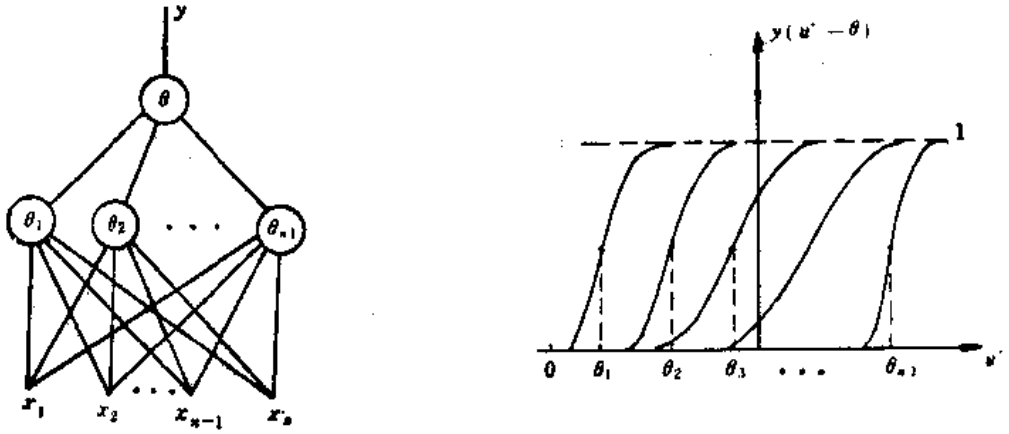
这里 $x\in[0,1]^n$，$\delta_1>0$

如果证明傅里叶级数中的任意三角函数项可以用三层 B-P 的子网络来逼近，那么就可以保证用三层 B-P 子网络来逼近任意函数 $h_i(x)$，即可逼近多维输出 $h(x)$。考虑子网络为 n 个

输入节点，n_1 个隐节点和 1 个输出节点。用该子网络来逼近一个正弦或余弦函数，子网络的输出节点是隐单元输出加权的线性叠加，子网络的输出为

$$y=\sum_{n_1} w_{n_1} f\left(\sum_{i=1}^{n} w_{in_1} x_i-\theta_{n_1}\right) \tag{2-70}$$

w_{n_1} 为第 n_1 个隐单元到输出单元的权，w_{in_1} 为第 i 个输入单元到第 n_1 个隐单元的权，θ_{n_1} 为隐单元的阈值，图 2-23a 表示这个子网络的结构。现在用此网络来逼近一个正弦函数。



(a) (b)

图 2-23

(a) 用于函数逼近的 B-P 子网络 (b) θ_{n1} 对隐层输出函数的影响

令 $u'_{n_1}=\sum_{i=1}^{n} w_{in_1}x_i$，$y(u'_{n_1}-\theta_{n_1})\sim u'$，即 $y(u'-\theta)\sim u'$ 的关系如图 2-23b 所示。不同的 θ_{n_1}，使 $y(\cdot)$ 在 u' 轴上的位置是不同的。

又令 $u'_{n_1}-\theta_{n_1}=\beta_{n_1}(\omega-\alpha_{n_1})$，$\beta_{n_1}\omega=u'_{n_1}$，$\beta_{n_1}$ 为图 2-23b 上各个 $f(\cdot)$ 曲线的斜率，β_{n_1} 可正可负，图中我们画的都是正的。$\alpha_{n_1}=\theta_{n_1}/\beta_{n_1}$，为 $f_{n_1}(\cdot)$ 在横轴上的位移，假设用(2-70)中的 y 去逼近傅里叶级数中的一个正弦函数或余弦函数，设 $\sin(2\pi i\Sigma k_i\cdot x_i)=\sin(u)$，给定一个 $\delta_2>0$，讨论是否存在 $w_{n_1}, w_{in_1}, \theta_{n_1}$ 满足：

$$\left|\sin(u)-\sum_{n_1} w_{n_1} f_{n_1}\left(\sum_{i=1}^{n} w_{in_1}x_i-\theta_{n_1}\right)\right|<\delta_2 \tag{2-71}$$

令

$$s(\alpha,\beta,w_l,\boldsymbol{x})=\sum_{n_1} w_{n_1} f_{n_1}(\beta_{n_1}(\omega-\alpha_{n_1})) \tag{2-72}$$

w_l 是与 w_{n_1}, w_{in_1} 及 n_1 隐单元的子网络有关的量，α，β 与 α_{n_1}，β_{n_1} 有关，从而 α，β，w_l 为待定，(2-71)式可写为

$$|\sin(u)-s(\alpha,\beta,w_l,\boldsymbol{x})|<\delta_2 \tag{2-73}$$

u 是一个有界变量($d\leqslant u\leqslant e$)，为了满足(2-73)式，对位移量作区间分割如下：

$$d\leqslant\alpha_0<\alpha_1<\alpha_2\cdots<\alpha_{n_1-1}<\alpha_{n_1}\leqslant e$$

而 β_{n_1} 足够大，此时，$f_{n_1}(\beta_{n_1}(\omega-\alpha_{n_1}))$ 为一个硬函数，是一阶跃函数，它的范围在[0, 1]之间，而 $w_{n_1}f_{n_1}(\beta_{n_1}(\omega-\alpha_{n_1}))$ 是一个高度为 w_{n_1} 的阶梯函数，w_{n_1} 可正可负，其阶梯的长度为 $(\alpha_i-\alpha_{i-1})$，$i=1,2,\cdots,n_1$，其阶梯的高度为 w_{n_1}，对于任意的 $\delta_2>0$，$s(\alpha,\beta,w_l,\boldsymbol{x})$ 总可以通过选择足够的 n_1，调节 w_{n_1}，和 $\alpha_i-\alpha_{i-1}$ 的宽度达到 δ_2 范围内的逼近，而满足式(2-73)。

对于函数 $\boldsymbol{h}(\boldsymbol{x})$ 是 m 维空间的矢量，对其中一个分量 $h_i(\boldsymbol{x})$ 的逼近，可用式(2-69)中的傅里叶级数进行，改变(2-69)中的系数 C_K 为 $a(K,e)$，$b(K,l)$，是实部或虚部三角函数系数，

用 $y(\boldsymbol{x})$表示由 l 个子网络组合成的三层 B-P 网络，输出 $y(\boldsymbol{x})$为单个神经元，用来逼近输出矢量 $\boldsymbol{h}$ 中的一个分量 $h_i(\boldsymbol{x})$，用 $h_F(\boldsymbol{x})$表示 $h_i(\boldsymbol{x})$的傅里叶变换，则有

$$\begin{aligned}h_F(\boldsymbol{x})-y(\boldsymbol{x})=&\sum_{K\in w_l}a(K,l)[\sin(u)-s(\alpha,\beta,w_l,\boldsymbol{x})]\\&+i\sum_{K\in w_l}b(K,l)[\cos(u)-s'(\alpha,\beta,w_l,\boldsymbol{x})]\end{aligned}\tag{2-74a}$$

用 $y(\boldsymbol{x})$来逼近 $h_i(\boldsymbol{x})$，利用(2-69)、(2-74)式可得

$$\begin{aligned}\int_{[0,1]^n}|h_i(\boldsymbol{x})-y(\boldsymbol{x})|^2\mathrm{d}\boldsymbol{x}=&\int|h_i(\boldsymbol{x})-h_F(\boldsymbol{x})+h_F(\boldsymbol{x})-y(\boldsymbol{x})|^2\mathrm{d}\boldsymbol{x}\\\leqslant&\int_{[0,1]^n}|h_i(\boldsymbol{x})-h_F(\boldsymbol{x})|^2\mathrm{d}\boldsymbol{x}+\int_{[0,1]^n}|h_F(\boldsymbol{x})-y(\boldsymbol{x})|^2\mathrm{d}\boldsymbol{x}\\\leqslant&\delta_1+\delta_2\Sigma(a^2(K,l)+b^2(K,l))\leqslant\varepsilon\end{aligned}\tag{2-74b}$$

因为 δ_1,δ_2 是可任意选取的小数，$\delta_1>0,\delta_2>0,a^2+b^2>0$，因此给定任一小的正数 $\varepsilon>0$，总可找到 δ_1，δ_2 及有界的傅里叶级数的系数满足(2-74b)式，那么利用(2-69)，(2-73)可得到 $h_i(\boldsymbol{x})$与 $y(\boldsymbol{x})$的任意小数 ε 内的逼近，从而能得到输出矢量 $\boldsymbol{h}(\boldsymbol{x})$的逼近，证毕。

在证明中，把 $f_{n_1}(\cdot)$取为阶跃函数，这过于严格，其实 $f(\cdot)$函数本身是一个连续函数，可以展开为多项三角函数的叠加，这样 n_1 的数目不一定要很大。同样可用逼近定理 A 来证明其三层 B-P 网络可对任意函数的映照都成立，这里就不再讨论了。

3. 隐单元数的选择

对于隐层单元数的选择是一个十分复杂的问题，根据最近 Eberhart 的书[17]中阐述，称"这是一种艺术"，因为没有很好的解析式来表示，可以说隐单元数与问题的要求，输入输出单元的多少都有直接的关系。

对于用作分类的 B-P 网络，可以参照感知器中间隐单元数的公式(2-29)和(2-83a)，可是由于 B-P 网络的隐单元的输入和输出之间是单调上升的非线性函数，它的输出是一个软函数，因此比感知器要求的隐单元数少，事实上隐单元数网络太少可能不能训练出来，或网络不"强壮"，不能识别以前没有看到的样本，容错性差，但隐单元数太多又使学习时间过长，误差也不一定最佳，因此存在一个最佳的隐单元数，如何求解，下面几个公式可作参考。

$$(1)\quad k<\sum_{i=0}^{n}C\binom{n_1}{i}\tag{2-75}$$

式中 k 为样本数，n_1 为隐单元数，n 为输入单元数，如 $i>n_1,C\binom{n_1}{i}=0$

$$(2)\quad n_1=\sqrt{n+m}+a\tag{2-76}$$

其中 m 为输出神经元数，n 为输入神经元数，a 为 1～10 之间的常数。

$$(3)\quad n_1=\log 2n\tag{2-77}$$

n 为输入神经元数。

以上这些参考公式，对于分类器来说，主要由于它的输入和输出基本上是二值函数，往往输入是二值图像像素或特征，而输出则为 0,1 等情况，故而可用感知器的模型来参照。

对于用作函数逼近的 B-P 网络，中间层的单元数与要逼近的函数的精度和函数本身的波动情况有关，例如要求逼近精度高，要求逼近三角函数或多项式的项数要增加，因而隐单元的个数也高，同样函数在闭区间内波动越多，要求三角函数的频率也高，从而项数也多，这样隐单元数也高。但这只是一种粗略的说明，并没有十分严格的关系，事实上，由于隐单元本身也可展开为很多多项式或三角函数相加，这样隐单元个数还与逼近函数的本身性质有关。

对于用作数据压缩情况下的B-P网络(以后在应用中讨论)，隐单元与输入单元的比为其数据的压缩比，它可参照公式(2-77)来考虑。

还有一种考虑是使隐单元的数目可变，或初始放入足够多的隐单元，然后把学习后那些不起作用的隐单元逐步去掉，一直减少到不可收缩为止。也可在初始放入比较少的隐单元，学习一定的次数后，不成功再增加隐单元数，一直达到比较合理的隐单元数为止。这样做对于用硬件完成B-P多层网络有一定的好处，但对于结构的选定所花的时间较长。

4. 初始值的选取

由于系统是非线性的，初始值对于学习是否到达局部最小和是否能收敛的关系很大，一个重要的要求是希望初始权在输入累加时使每个神经元的状态值接近于零，这样可保证一开始时不落到那些平坦区上。权一般取随机数，而且权的值要求比较小，这样可以保证每个神经元一开始都在它们转换函数变化最大的地方进行。

对于输入样本同样希望能够进行归一，使那些比较大的输入仍落在神经元转换函数梯度大的那些地方。

六、B-P网络的应用举例

B-P网络是目前神经网络中应用最广的一类网络之一，它的应用主要为三个方面：

其一，模式识别、分类。用于语言、文字、图像的识别，用于医学特征的分类、诊断等。

其二，函数逼近。用于非线性控制的函数的建模、拟合非线性控制曲线、机器人的轨迹控制及其他工业控制等。

其三，数据压缩。在通信中的编码压缩和恢复，图像数据的压缩和存贮及图像特征的抽取等。

这里分别叙述三种应用的实例。

1. 用于模式识别

[例2-2] 对于一些手写数字进行模式识别。

由于手写数字变化很大，用传统的统计模式识别或句法模式识别很难得到高的识别率，B-P网络是综合这两类模式识别的特点，它的权是通过对训练样本的学习而得到的，包括统计和句法配合的问题，因而有较高的识别率。图2-24为不同人写的手写数字，由于字的体形大小不同，这儿取其特征作为B-P网络的输入，用传统的方法例如考虑端点特征，1,2,3,7都有两个端点，0,6, 8, 9都是由圈构成；在1, 2, 3, 7这组数字中，"2"的两个端点为上左下右，"3"的两个端点为两个都在一边，而在0,6,8,9这组数字中，"9"的圈在上，"6"的圈在下。每个特征在输入神经元上取1，而无此特征取0，这里共取了34个特征作为可区分不同的字型的依据，当然随着学习的样本数的增加，可能要增加其输入的特征数。如果输出神经元 $m=10$，对应10个数字，其输出只有一个为"1"，其他为"0"，这样的分类输入和输出都是0、1二值，中间层是[0 ,1]之间的模拟量，因此，可用三层B-P网络来完成，其隐单元数可根据公式(2-75)，

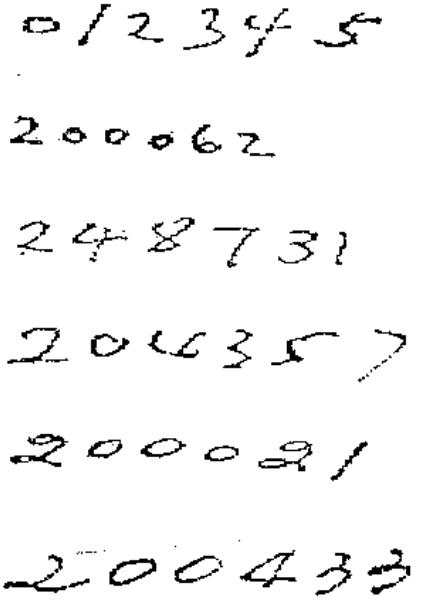


图2-24 手写体数字

(2-76), (2-77)来选取。考虑输入样本数 P，对于可能的输入样本有 2^{34} 个，但是很多样本是不会出现的，根据200人写的数字统计，样本数约为 $P=1000$，根据(2-75)式，取 $n_1=10$：

$$\sum_{i=0}^{n} C\binom{n_1}{n}=\binom{10}{0}+\binom{10}{1}+\cdots+\binom{10}{10}=1108>1000$$

根据(2-76)式：

$$n_1=\surd\, m+n\ +a=\surd\, 390\ +a=8\sim17$$

式中 $a=1\sim9$

根据(2-77)式：

$$n_1=\log_2 n=6$$

按上述几个公式并考虑到对测试样本的容差性，取 $n_1=14$。通过对样本的学习，并对6000多个字的测试，其识别率 $>95\%$。

[例 2-3] T-C 模式分类问题。

T-C 分类是一个十分典型的问题。用 B-P 学习的方法得到一些有效的权，它是输入一个 3×3 的两维图像，黑色为 +1，白色为 0。图 2-25 表示 T 和 C 的两个字母，分别旋转0°, 90°, 180°, 270° 时的情况，希望对图 2-25 的样本，输入 T 时其输出为 1，输入 C 时其输出为 0，保持平移和旋转不变。这里隐单元并不与所有的输入单元相连，每个隐单元只与输入 3×3 个输入单元相连接，其连接权对每个隐单元都是相同的，为了使隐单元能覆盖输入的整个平面，隐单元数与输入单元几乎相等，如输入为 $n\cdot n$，而隐单元数为 $(n-1)\cdot(n-1)$，输出单元为 $m=1$，其 B-P 网络如图 2-26 所示，输出单元与每个隐单元相连，因此，不管字母放在输入的什么地方，总有一些隐单元被输入的黑色 1 激励，只要适当地调节输出阈值和权就可满足其要求。图 2-27 表示对于不同的初值学习后得到的四种不同的输入权。在图 2-27a 的权值情况下，字母"T"在输入二维平面上，无论放在什么方向上(四种不同方向)，总存在一个隐单元，它的输入和等于 1，在 2-27a 上表示两个不同方向"T"输入后，输入为 1 的地方与权相乘、求和大于 0，等于 1。而同样是 2-27a 的这套权，对于字母 C 输入，可以算出，它不存在一个隐单元的加权和大于 0。因此，只要适当选取图中输出单元的阈值 $\theta=0.5$，

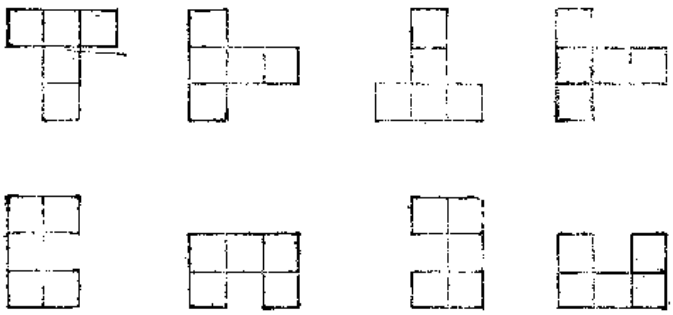


图 2-25 输入不同方向的"T""C"字母

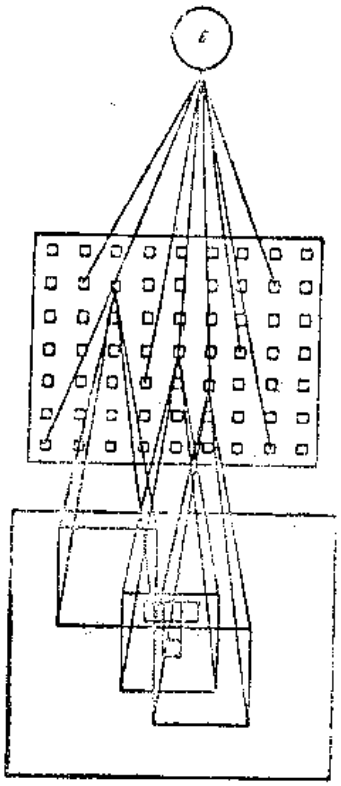


图 2-26 识别"T"与"C"的 B-P 网络

而输出单元的权为1，就可使字母"T"输入时，输出单元为1，字母"C"输入时为0。同理，在图2-27*b*的这套权的情况下，对任意方向上字母"T"输入，其有一个隐单元的加权和为2，如图中表示的两个不同方向的"T"，它们满足一个隐单元的加权和为2，而同样这套权，对任何方向上的"C"字母输入最多有一个隐单元的加权和为1，只要选$\theta=1.5$，就可使"T"输入时输出为1，"C"输入时输出为0。图2-27*c*中"T"输入时有五个隐单元被激活，即加权和大于0，而"C"输入时只有三个隐单元被激活。选$\theta=4$，可区分"T"和"C"。图2-27*d*是具有相同的负权-2，在"T"输入时，25个隐单元中有21个不兴奋，4个兴奋，在"C"输入时，25个隐单元中有20个不兴奋，5个兴奋，在隐单元和输出单元之间选负权，如-1，而阈值为-4.5，就可把字母"T"和"C"区别开来。

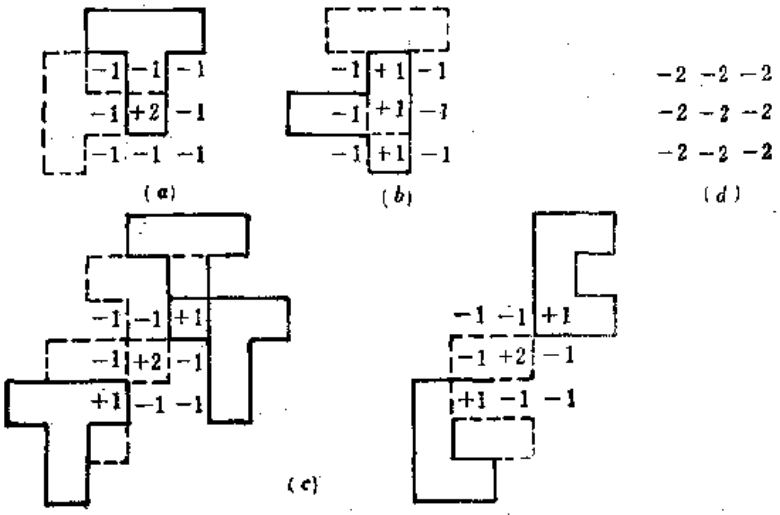


图2-27 学习得到不同类型的权

2．函数逼近

[例2-4] 非线性曲线的拟合。

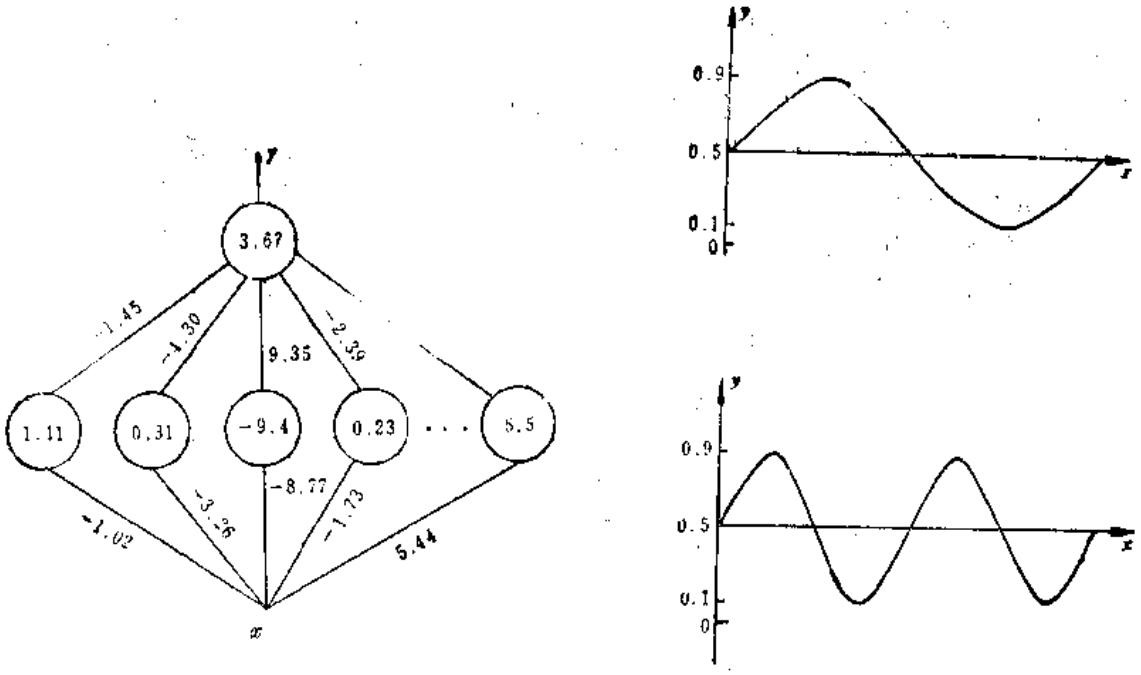
在控制中往往希望产生一些非线性的输入输出曲线，例如，已知一个机械臂取物的轨迹，根据这个轨迹可计算出各臂关节的角度θ_1和θ_2(这里是有两个关节的情况)，按照机械臂的θ要求应该反演计算出驱动马达的力或脉冲频率，这是一个复杂的数学计算，用B-P网络学，可以不需要已知机械臂的动力学模型，而直接用样本计算得到这些曲线的拟合，即$y=g(x)$，x为要求的θ角，y为θ角对应的马达驱动，在实际机器人的运动中取出n个点，只要把曲线$g(x)$上的每一对(x_1,y_1)，(x_2,y_2)，…，(x_n,y_n)作为输入与输出的教师样本，那么学习后就可得到精度较高的曲线拟合，用B-P网络就可以代替轨迹跟踪中的复杂计算。

图2-28*a*表示用一个输入单元x，8个隐单元和一个输出单元y来拟合一个正弦曲线，它同样也可拟合两个周期、三个周期的正弦曲线，图2-28*b*表示一个0.1～0.9之间的单个周期和两个周期的正弦曲线。

$$y=0.4\sin(2\pi x)+0.5 \qquad \text{单个周期}$$

其中$x\in[0,1]$的连续值

$$y=0.4\sin(4\pi x)+0.5 \qquad \text{两个周期}$$



(a) 函数逼近网络　　(b) 要求逼近的函数

图 2-28　函数逼近示意图

十分明显，对于同样的隐单元数，单周期函数的训练次数少、精度高，单个周期和两个周期在 $\eta=0.2,\alpha=0.8$ 时，单个周期的训练次数为 2864 次，误差为 0.001，而两个周期训练次数为 30000 次，误差为 0.01。这说明隐单元的数目与函数拟合的精度很有关系。对于函数逼近的隐单元取法应在公式(2-75)、(2-76)、(2-77)的基础上，按精度要求和频率波动的情况加以调整。

3．数据压缩

数据压缩，又称数据特征抽取，是一种由三层神经元组成的 B-P 网络，其输入与输出层的神经元数完全相同，若均为 n 个，其中间层根据公式(2-77)为 $n_1=\log_2 n$ 个，当输入各个学习样本后，通过 B-P 网络的学习使输入与输出样本完全相同，由于中间神经元数目远远小于输入和输出神经元，则中间神经元便可成为输入样本的一种压缩。图 2-29 表示这种数据压缩的 B-P 网络。将中间神经元的输出存放在内存里，要用时只需乘上权就可以恢复到原来的数据。在图像存贮上，可存贮这些压缩数据，减少内存的耗费。在代码通信时，可以在发射方面发出压缩数据，在接收时乘上输出权，其代码可得到复原。这种数据压缩的中间单元，也可以作为输入样本的某些特征。

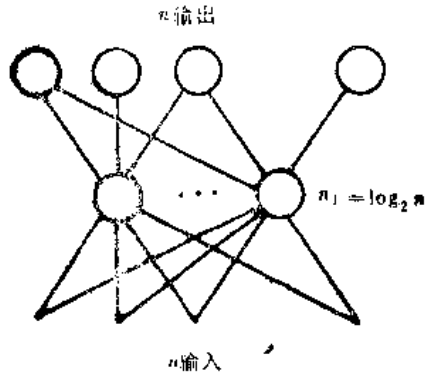


图 2-29　用于数据压缩的 B-P 网络

一个简单的例子如表 2-1 所示，输入样本是一串 8 个 bit 的代码，其中只有一个为 1，其他为 0，输出与输入相同，中间采用三个神经元，它们的输出如表上所示。在计算机内只要存入中间神经元的输出，那么需用时，就可以很快恢复其原来的代码。

表 2-1

输入								中间			输出							
1	0	0	0	0	0	0	0	0.5	0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	0.5	0	1	0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0	1	0	0.5	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0	0	0.5	0	0	0	0	0	0	0	1

第三章　反馈式人工神经网络

反馈式人工神经网络的结构如图 3-1 所示，我们首先考虑单层全反馈网络，这种网络中的每个神经元的输出都与其他神经元的输入相连，它的输入与输出关系为

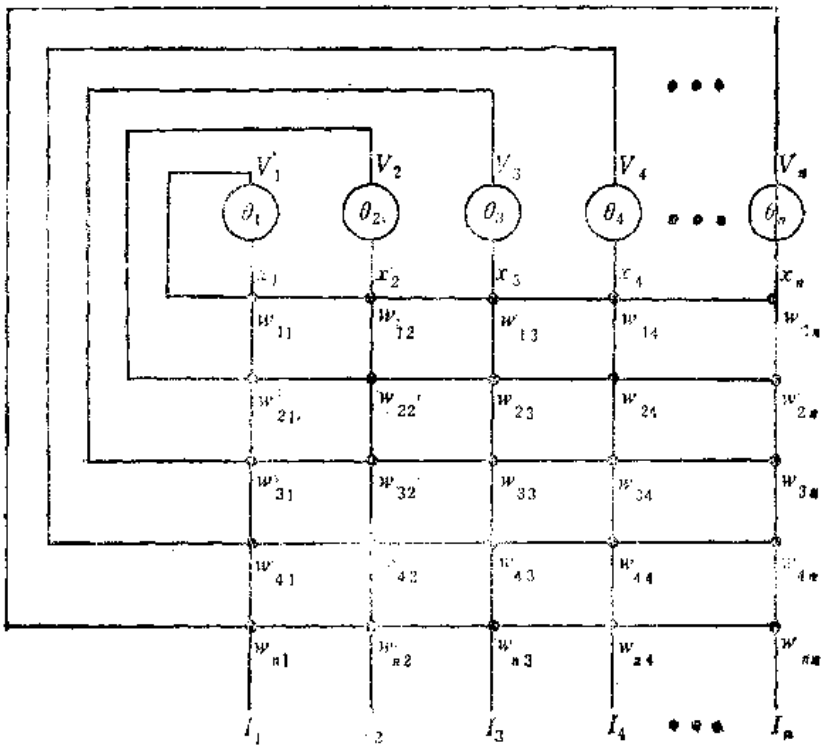


图 3-1　反馈式网络

$$s_j=\sum w_{ij}V_i+I_j \tag{3-1}$$

$$x_j=g(s_j) \tag{3-2}$$

$$V_j=f(x_j) \tag{3-3}$$

这里 i, j 分别为 $1,2,\cdots,n$，x_j 为第 j 个神经元的输入状态，由于反馈网络的输出同上一时刻的状态有关，因此它与前馈网络不同，它的 x_j 的作用比较突出。在反馈网络中如果 $x_j=s_j$，并且 $f(x_j)$ 是一个二值的硬函数，$V_j=\mathrm{sgn}(x_j)$，那么这种网络为离散型的反馈网络。如果式 (3-2) 用状态方程来表述 $x_j=-\dfrac{\mathrm{d}x_j}{\mathrm{d}t}+s_j$，$V_j=f(x_j)$ 中 $f(\cdot)$ 为一个连续单调上升的有界函数，这类网络为连续型的反馈网络。反馈式的单层网络有多种，这里主要讨论由 Hopfield 提出的一种网络，称为 Hopfield 网络。Hopfield 网络又分为两类，对离散型的称为 DHNN (Discrete Hopfield Neural Network)，对连续型的称为 CHNN (Continuous Hopfield Neural Network)。这种单层网络输入与输出的神经元数是相同的，对于一个由 n 个神经元组成的反馈网络，在某一时刻 t，其状态矢量 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^T$，$\boldsymbol{x}\in R^n$，它的输出

矢量为 V, $V=(V_1,V_2,\cdots,V_n)^T$, $V\in R^n$, 用 $x(t)$, $V(t)$来表示。通过式 (3-1)、(3-2), 由 $V(t)$得下一时刻的 x, 即 $x(t+1)$, 而 $x(t+1)$又引起 $V(t+1)$的变化, 这种反馈演化的过程, 使状态矢量 $x(t)$随时间发生变化。在一个 n 维状态空间上, 可以用一条轨迹来描述, 从初始值 $x(t_0)$ 出发, $x(t_0+\Delta t)\rightarrow x(t_0+2\Delta t)\cdots x(t_0+m\Delta x)\cdots$这些在空间上的点组成的确定轨迹, 是演化过程中所有可能状态的集合, 我们称这个状态空间为相空间。

图 3-2 描述了一个三维相空间上三条不同的轨迹, 对于 DHNN, 因为 $V(t)$ 中每个元只可能为 ±1, 或$\{0,1\}$, 对确定的权 w_{ij}, 其轨迹是跳跃的阶梯式, 如图 3-2 中 A 所示。对于 CHNN, 因为 $x(t)$ 是连续的, 因而其状态轨迹也是连续的, 如图 3-2 中的 B、C 所示。

对于网络的不同联接权 w_{ij} 和输入 $I_j(i,j=1,2,\cdots,n)$, 其状态轨迹可能出现以下几种情况:

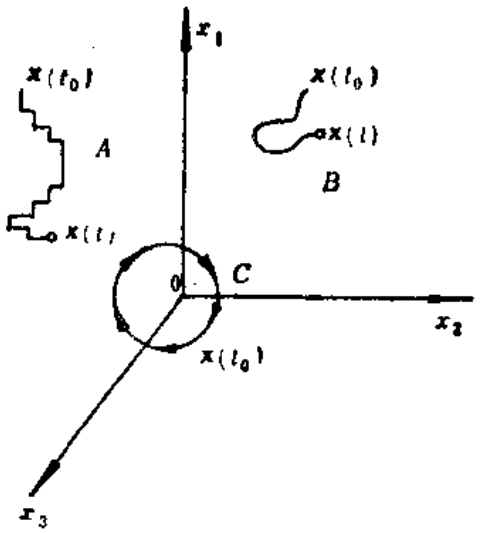


图 3-2 三维空间三个不同的状态轨迹

(1) 轨迹经过一定的时间 t 后, $t>0$, 此轨迹不会延伸, 而永远停留在$x(t_0+t)$, 这时我们称网络收敛到一个稳定点。轨迹的起始点 $x(t_0)$ 是系统在 t_0 时状态的初值, 从 $x(t_0)$ 开始, 状态经过 t 后, 到达 $x(t_0+t)$, 如果 $x(t_0+t+\Delta t)=x(t_0+t)$, $\Delta t>0$, 则 $x(t_0+t)$ 为网络的稳定点, 成平衡点。由于 $x(t)$ 不再变化, 相对应的 $V(t+t_0)$ 也达到了稳定值。对于非线性系统来说, 不同的初始值 $x(t_0)$, 可能有不同的轨迹, 到达不同的稳定点, 这些稳定点, 也可以认为是人工神经网络的解。在一个反馈网络中, 存在很多稳定点, 根据不同情况, 这些稳定点可分为:

(i) 渐近稳定点 x_s 如果在稳定点 x_s 周围的 $s(\sigma)$ 区域内, 从任一个初态 $x(t_0)$ 出发的每个运动, 当 $t\rightarrow\infty$ 时都收敛于 x_s。这样, 它不仅存在一个稳定点 x_s, 而且存在一个稳定域, 有时也称这个稳定点为吸引子, 对应的稳定域为吸引域。

(ii) 不稳定的平衡点 x_{en}, 在某些特定的轨迹演化过程中, 能够使网络达到稳定点 x_{en}, 但对 x_{en} 的其他方向上, 任一个小的区域 $s(\sigma)$, 不管 $s(\sigma)$取多么小, 其轨迹在时间 t 以后总是偏离 x_{en}。

(iii) 网络的解, 如果网络最后是稳定到设计人员期望的稳定点上, 而且该稳定点又是渐近稳定点, 那么这个点即是网络的解。

(iv) 网络的伪稳定点, 网络最终稳定到一个渐近稳定点上, 但这个稳定点不是网络设计所要求的解, 这个稳定点为伪稳定点。

在一个非线性的反馈网络中存在着这些不同类型的稳定点, 而网络设计的目的是希望网络落到所要求的稳定点上, 并且还要有一定的稳定域。

(2) 轨迹为环状, 称为极限环。如果在某些参数的情况下, $x(t)$ 的轨迹是一个圆, 或为一个环, 状态 $x(t)$ 沿着环重复旋转, 永远不会停止, 这时输出 $V(t)$ 也出现周期变化, 说明系统出现了振荡, 如图 3-2 中 C 的轨迹就是这种极限环的情况。对于 DHNN, 轨迹变化可能在两种状态下来回跳动, 其极限环为 2, 如在 m 种状态下循环变化, 称其极限环为 m。

(3) 如果 $x(t)$ 的轨迹在某个确定的范围内运动，但既不重复，又不能停下来，状态变化为无穷多个，而轨迹也不发散到无穷远，这种现象称为混沌(Chaos)，在出现混沌的情况下，系统输出变化为无穷多个，并且随时间推移不能趋向稳定，但又不发散。图 3-3 所示是一个三状态组成的连续动态系统中，取两状态描出的状态轨迹。这种现象越来越引起人们的重视，因为在脑电波的测试中已发现这种现象，而在真正神经网络中存在的这种现象，也应在人工神经元中加以考虑。

图 3-3 相空间中的混沌图形

(4) 如果状态 $x(t)$ 的轨迹随时间一直延伸到无穷远，此时状态发散，而系统的输出也发散。在人工神经网络中，由于输入、输出关系函数 $f(\cdot)$ 是一个有界函数，虽然状态是发散的，但其输出 $V(t)$ 还是稳定的，而 $V(t)$ 的稳定反过来又限制了状态的发散，一般的非线性人工神经网络中发散现象不会发生，除非神经元的输入、输出关系是线性的。

对于一个由 n 个神经元组成的反馈系统，它的行为就是由这些状态轨迹的情况来决定，目前的人工神经网络是利用第一种情况来解某些问题的，如果把系统的稳定点视作一个记忆的话，那么从初态朝这个稳定点流动的过程就是寻找该记忆的过程，初态可以认为是给定的有关该记忆的部分信息，状态 $x(t)$ 流动的过程是从部分信息去寻找全部信息，这就是联想记忆的过程。如果把系统的稳定点考虑为一个能量函数的极小点，在状态空间中，从初始态 $x(t_0) \rightarrow x(t+t_0)$ 最后到达 x^*，若 x^* 为稳定点，则可以看作是 x^* 把 $x(t_0)$ 吸引了过

来，在 $x(t_0)$ 时，能量比较大，而吸引到 x^* 时能量比较小了，那么能量的极小点就可以作为一个优化目标函数的极小点，状态变化的过程就是优化某一个目标函数的过程。因此反馈网络的状态流动是一种计算联想记忆或优化的过程，而它的解并不需要真的去计算，而只要去形成这一类反馈神经网络，如果适当地设计其权 w_{ij}、输入 I_j，就可以达到这个目的。

图 3-4 表示这种状态随时间流动的过程图，其中"x^*"为状态的稳定点，如果初始状态 $x(t_0)$ 落到在稳定点 x^* 周围的 $s(\sigma)$ 范围内：

$$\|x(t_0)-x^*\|<s(\sigma)$$

当 $t\to\infty$ 时，$x(t_0+t)=x^*$

状态最后就稳定到 x^* 上，而 $s(\sigma)$ 称为该稳定点 x^* (或称吸引子)的吸引域。

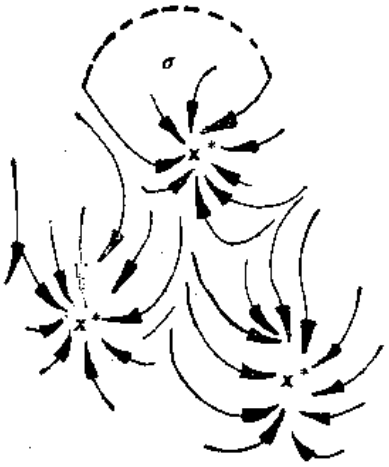


图 3-4 状态空间的状态流动的区域过程

通过对神经元之间的权和阈值的设计，要求单层的反馈网络达到：

(1) 网络系统能够达到稳定收敛。即讨论在什么条件下，系统不会出现振荡和混沌现象。

(2) 网络的稳定点。一个非线性网络可能有很多个稳定点，权的设计要求其中的某些稳定点是所要求的解，对于用作联想记忆的反馈型网络，希望稳定点都是一个记忆，那么记忆容量就与稳定点的数量有关，希望记忆的量越大，那么，稳定点的数目也越大。但稳定点数目的增加可能会引起吸引域的减小，从而使其联想功能减弱。对于用作优化的反馈网络，由于目标函数(即在系统中的能量函数)往往要求只有一个全局最小，那么稳定点越多，陷入局部最小的可能就越大，因而希望系统(非要求的)稳定点越少越好。

(3) 吸引域的设计，希望的稳定点有尽可能大的吸引域，而非希望解的稳定点的吸引域要尽可能小。因为状态空间是一个多维空间，状态随时间的变化轨迹也是多种形状，吸引域就很难用一个明确的解析式来表达，这在设计时要尽可能考虑的。

在本章内，讨论 DHNN，CHNN 和 CNN 三种网络模型，CNN 是由 Hopfield 全联接网络派生出来的局部联接网络。本章主要分析这三种模型的生物背景、结构、稳定性、稳定点和吸引域，同时结合网络的设计给出了三种网络的应用。

第一节　离散的单层反馈网络模型

DHNN 网络是一种单层的，其输入、输出为二值的反馈式网络，它主要用于联想记忆。当输入的向量 I 作为一个初值时，网络通过反馈演化，从网络输出端得到一个向量 V，V 是从初值 I 演化而联想到的一个稳定记忆。

一、基本公式

在公式(3-2)中令 $x_j=s_j$，$f(\cdot)$为一个硬函数，用 f_n 表示，则它的方程可写为

$$x_j(t+1)=\sum_i w_{ij}V_i(t)+I_j\quad\forall i,\ j=1,2,\cdots,n \tag{3-4}$$

$$V_j(t+1)=f_n[x_j(t)]=\mathrm{sgn}[x_j(t)] \tag{3-5}$$

$$\mathrm{sgn}[x_j(t)]=\begin{cases}+1 & x_j(t)\geqslant 0\\ -1 & x_j(t)<0\end{cases} \tag{3-6}$$

或者

$$V_j(t+1)=f_n[x_j(t)]=0.5+0.5\mathrm{sgn}[x_j(t)] \tag{3-7}$$

$$V_j(t+1)=\begin{cases}1 & x_j(t)\geqslant 0\\ 0 & x_j(t)<0\end{cases} \tag{3-8}$$

这里我们就采用式(3-4)、(3-5)、(3-6)来进行分析，至于输出为{0, 1}的情况，可按式(3-7)、(3-8)自行类推。

在式(3-4)中的 I_j 可以作为 $x_j(t)$ 的一个初值，也可作为输入，由于网络是反馈演化的，I_j 在输入后，就可撤去，而网络仍然可以演化下去，如果 I_j 永远接在输入端，则 I_j 可以作为一个阈值，设为 $-\theta_j$，$I_j=-\theta_j$，则式(3-4)可化为

$$x_j(t)=\sum_i w_{ij}V_i-\theta_j$$

它表示 $\Sigma w_{ij}V_i\geqslant\theta_j$ 时，$x_j(t)\geqslant 0$，V_j 兴奋，反之 V_j 抑制。

DHNN 网络主要有两种工作方式：

(1) 异步方式：在某一时刻 t，只有一个神经元按照公式(3-4)、(3-5)进行变化，而其余的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也可以按照预定的顺序来选择。如选到的神经元为第 j 个，则有

$$\left.\begin{aligned}V_j(t+1)&=\mathrm{sgn}[x_j(t)]\\ V_i(t+1)&=V_i(t)\qquad i\neq j\end{aligned}\right\} \tag{3-9}$$

(2) 同步方式：在任何时刻 t，所有的神经元同时按照式(3-4)、(3-5)式进行变化，即

$$V_i(t+1)=\mathrm{sgn}[x_i(t)]\qquad i=1,2,\cdots,n \tag{3-10}$$

二、稳定点

网络从一个初始态 $t=t_0$、$\boldsymbol{x}(t_0)$ 时开始，经过一个有限的时刻 t，网络的输出不再发生变化，这些输出为网络的稳定点，用公式表示为

$$\boldsymbol{V}(t_0+t+\Delta t)=\boldsymbol{V}(t_0+t)\qquad \Delta t>0$$

$\boldsymbol{V}\in B^n$，n 为神经元的数目。

当 $\Delta t=1$、$t_0=0$ 时，可写为

$$\boldsymbol{V}(t+1)=\boldsymbol{V}(t) \tag{3-11}$$

当网络处在稳定点时，每个神经元的输出满足：

$$V_j(t+1)=V_j(t)=\mathrm{sgn}[\sum_{i=1}^{n} w_{ij}V_i(t)+I_j] \tag{3-12}$$

即

$$V_j(t)\cdot x_j(t)>0\qquad j=1,2,\cdots,n$$

$$x_j(t)=\sum_{i=1}^{n} w_{ij}V_i(t)+I_j$$

对于不同的输入样本向量，$\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots\boldsymbol{V}^k\cdots\boldsymbol{V}^m$，如果希望每一个输入的样本为系统的初值，最后又都能演化到自己，即每一个样本向量都是网络最终的稳定点，那么必须使每个样

本满足：

$$\boldsymbol{W}\cdot\boldsymbol{V}^k=\beta\boldsymbol{V}^k \qquad k=1,2,\cdots,m \tag{3-13}$$

$$\boldsymbol{W}\in R^{n\times n},\ \boldsymbol{W}=\begin{bmatrix} w_{11} & w_{21} & \cdots & w_{n1} \\ w_{12} & w_{22} & \cdots & w_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ w_{1n} & w_{2n} & \cdots & w_{nn} \end{bmatrix}$$

$$\boldsymbol{V}^k\in R^n \qquad \boldsymbol{V}^k=[V_1^kV_2^k\cdots V_n^k]^T$$

$$\beta\in R^{n\times n} \qquad \beta=\begin{bmatrix} \beta_1 & \cdots & 0 \\ \vdots & \beta_2 & \vdots \\ 0 & \cdots & \beta_n \end{bmatrix}$$

$$\beta_i>0 \qquad i=1,2,\cdots,n$$

如果把每个 $\boldsymbol{V}^k$ 矢量都包括在内($k=1,2,\cdots,m$)，方程(3-13)可以写成：

$$\boldsymbol{W}[\boldsymbol{V}^1\boldsymbol{V}^2\cdots\boldsymbol{V}^m]=[\beta^1\boldsymbol{V}^1\beta^2\boldsymbol{V}^2\cdots\beta^m\boldsymbol{V}^m] \tag{3-14}$$

$$\boldsymbol{WZ}=\boldsymbol{B}$$

$$\boldsymbol{Z}\in R^{n\times m},\boldsymbol{B}\in R^{n\times m},\boldsymbol{W}\in R^{n\times n}$$

对(3-14)式两边取转置，得到

$$\boldsymbol{Z}^T\boldsymbol{W}^T=\boldsymbol{B}^T \tag{3-15}$$

取式(3-15)中 $\boldsymbol{W}^T$ 的第 j 列，得到

$$\boldsymbol{Z}^T\boldsymbol{W}_j=\boldsymbol{B}_j \tag{3-16}$$

其中

$$\boldsymbol{Z}^T=\begin{bmatrix} V_1^1 & V_2^1 & \cdots & V_n^1 \\ V_1^2 & V_2^2 & \cdots & V_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ V_1^m & V_2^m & \cdots & V_n^m \end{bmatrix};\ \boldsymbol{W}_j=\begin{bmatrix} w_{1j} \\ w_{2j} \\ \vdots \\ w_{nj} \end{bmatrix};\ \boldsymbol{B}_j=\begin{bmatrix} B_j^1 & V_j^1 \\ B_j^2 & V_j^2 \\ \vdots & \vdots \\ B_j^m & V_j^m \end{bmatrix}$$

式(3-16)有 m 个方程，有 n 个未知数，如果 $m\leqslant n$，并且 $\boldsymbol{Z}$ 是满秩时(即各个矢量线性独立)，那么总可以解出 $\boldsymbol{W}_j$ 来满足稳定点为 $\boldsymbol{V}^1,\cdots,\boldsymbol{V}^m$ 的条件。这表示网络能够"记住"所有的样本集。

在网络中，我们不仅要求"记住"所存的样本集，还要求每个样本周围都有足够大的"吸引域"，才能对诸如模糊、畸变、不完整的输入有联想识别的能力。虽然网络能够在样本集输入时收敛到自己，但并不说明整个系统是稳定的，因为在某些非样本矢量输入时，系统可能会出现振荡(极限环)，当然由于 DHNN 是一个具有二值输出的网络，它的状态值是有限的，因此不可能出现混沌现象。

三、网络的稳定性

网络存在多个稳定点，其中有些即使是渐近稳定点，只要存在一个极限环，此网络就不可避免地在某些初值时会出现振荡，对于一个非线性二值反馈网络，如何用一个解析判据来分析系统的稳定性呢？这里借用了铁磁材料中哈密顿函数的形式来定义一个离散反馈网络的能量函数。在铁磁体中自旋只有两个方向，即铁磁分子自旋 $P_i\in\{1,-1\}$，$i=1,2,\cdots,n$，在这种材料中的哈密顿函数为

$$H=-\frac{1}{2}\sum_i\sum_j J_{ij}P_iP_j-\sum_i H_iP_i \tag{3-17}$$

P_i,P_j只有两种方向 ±1,J_{ij}表示第 i 分子与第 j 个分子之间的作用 $J_{ij}=J_{ji}$,H_i 是外加的随机场,整个物质的相互作用结果是使哈密顿函数 H 达到最小。由于在 DHNN 中的输出也是二值量,而相互作用由权 w_{ij} 表示,外加输入为 I_i,所以 Hopfield 定义了一个系统的能量函数:

$$E=-\frac{1}{2}\sum_i\sum_j w_{ij}V_iV_j-\sum_i I_iV_i \tag{3-18}$$

由于 V_i,V_j 只可能为 $+1,-1$,w_{ij}、I_i 有界,$i,j=1,2,\cdots,n$ 能量 E 也是有界的。

$$|E|\leqslant\frac{1}{2}\sum_i\sum_j|w_{ij}||V_i||V_j|+\sum_i|I_i||V_i|$$

$$=\frac{1}{2}\sum_{ij}|W_{ij}|+\sum_i|I_i| \tag{3-19}$$

从任意一个初始状态开始,在每次迭代时都能满足 $\Delta E\leqslant0$,那么网络的能量将会越来越小,最后趋向于稳定点 $\Delta E=0$。它的物理意义是,在那些渐近稳定点的吸引域内,当状态离吸引点越远,其能量越大,当状态越接近于稳定点时,其能量越小。而能量 E 的单调下降,说明状态的运动从远离吸引子跑到了吸引子上,最后使网络达到稳定。

我们再分两种工作方式来讨论它的稳定性。

1. 异步方式

(1) 当网络工作在异步方式下,满足 $W_{ij}=W_{ji},W_{ii}=0,i,j=1,2,\cdots,n$,则其能量函数能够单调下降,且网络必定稳定。

证明 对于 DHNN 网络的某一个神经元 i,它的输出变化可能为

$$\Delta V_i=V_i(t+1)-V_i(t)=\begin{cases}0 & V_i(t+1)=V_i(t)\\ +2 & V_i(t+1)=1,V_i(t)=-1\\ -2 & V_i(t+1)=-1,V_i(t)=1\end{cases}$$

根据异步方式的定义,每个时刻只有一个神经元变化,如第 i 个神经元变化,而其他神经元不变,据公式(3-18)可得

$$\Delta E=-\frac{1}{2}\sum_{j=1}^{n}w_{ij}V_j\Delta V_i-\frac{1}{2}\sum_{j=1}^{n}w_{ji}V_i\Delta V_i-I_i\Delta V_i$$

由于

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

所以

$$\Delta E=-(\sum_{j=1}^{n}w_{ji}V_j+I_i)\Delta V_i=-x_i(t)\Delta V_i$$

由于 $V_i(t+1)=\mathrm{sgn}[x_i(t)]$,$\Delta V_i=V_i(t+1)-V_i(t)$

因此 当 $x_i(t)\geqslant0$ $\Delta V_i\geqslant0$ $\Delta E\leqslant0$

当 $x_i(t)<0$ $\Delta V_i\leqslant0$ $\Delta E\leqslant0$

所以网络无论在什么初始条件下都得保证 $\Delta E\leqslant0$,这样就保证了网络的稳定性和收敛性。

(2) 当网络工作在异步方式下,若 $w_{ij}=w_{ji}$,且 $w_{ii}>0$,则网络的能量函数也单调下降,

网络必定稳定。

对于 DHNN 网络的第 i 个神经元发生变化，如果 $w_{ij}=w_{ji}$，且 $w_{ii}>0$ 则

$$\begin{aligned}\Delta E&=-\frac{1}{2}\sum_{j\neq i}w_{ij}V_j\Delta V_i-\frac{1}{2}\sum_{j\neq i}w_{ji}V_j\Delta V_i\\&\quad-w_{ii}[V_i^2(t+1)-V_i^2(t)]-I_i\Delta V_i\\&=-(\sum_j w_{ij}V_j+I_i)\Delta V_i-w_{ii}\Delta V_i(V_i(t+1)+V_i(t))\\&=-x_i(t)\Delta V_i-w_{ii}\Delta V_i(V_i(t+1)+V_i(t))\end{aligned}$$

由于 $w_{ii}>0$，且 ΔV_i 与 $V_i(t+1)$ 是同号的，即仍能保证 $\Delta E\leqslant 0$，这也证明了网络的稳定性和收敛性。

2. 同步方式

在全并行工作时，如果满足 $w_{ij}=w_{ji}$，网络将收敛于一个稳定点，或者网络收敛于一个周期解，其极限环为 2。

证明 在全并行工作时，其能量函数可用下式表示：

$$E=-\frac{1}{2}\sum_i\sum_j w_{ij}V_i(t+1)V_j(t)-\frac{1}{2}\sum_i I_i[V_i(t)+V_i(t+1)] \tag{3-20a}$$

可写成矩阵形式：

$$E=-\frac{1}{2}\boldsymbol{V}^T(t+1)\boldsymbol{W}\boldsymbol{V}(t)-\frac{1}{2}\boldsymbol{I}^T[\boldsymbol{V}(t+1)+\boldsymbol{V}(t)] \tag{3-20b}$$

$$\boldsymbol{V}\in R^n;\boldsymbol{W}\in R^{n\times n};\boldsymbol{I}\in R^n$$

$$\begin{aligned}\Delta E&=-\frac{1}{2}\boldsymbol{V}^T(t+1)\boldsymbol{W}\boldsymbol{V}(t)-\frac{1}{2}\boldsymbol{I}^T[\boldsymbol{V}(t+1)+\boldsymbol{V}(t)]\\&\quad+\frac{1}{2}\boldsymbol{V}^T(t)\boldsymbol{W}\boldsymbol{V}(t-1)+\frac{1}{2}\boldsymbol{I}^T[\boldsymbol{V}(t)+\boldsymbol{V}(t-1)]\\&=-\frac{1}{2}[\boldsymbol{V}^T(t)\boldsymbol{W}][\boldsymbol{V}(t+1)-\boldsymbol{V}(t-1)]-\frac{1}{2}\boldsymbol{I}^T[\boldsymbol{V}(t+1)-\boldsymbol{V}(t-1)]\\&=-\frac{1}{2}[\boldsymbol{V}^T(t)\boldsymbol{W}+\boldsymbol{I}^T][\boldsymbol{V}(t+1)-\boldsymbol{V}(t-1)]\\&=-\frac{1}{2}[\boldsymbol{x}(t)]^T[\boldsymbol{V}(t+1)-\boldsymbol{V}(t-1)]\end{aligned}$$

由于在 $\boldsymbol{x}(t)$ 中每个分量 $x_i(t)$ 与在 $\boldsymbol{V}(t+1)$ 中每个分量 $V_i(t+1)$ 同号，因而

$$[\boldsymbol{x}(t)]^T[\boldsymbol{V}(t+1)-\boldsymbol{V}(t-1)]\geqslant 0 \qquad \forall i \text{ 成立}$$

所以 $\Delta E\leqslant 0$。现在考虑在稳定点的情况，即 $\Delta E=0$ 的情况：

若 $\boldsymbol{V}(t)=\boldsymbol{V}(t+1)=\boldsymbol{V}(t-1)$，则 $\Delta E=0$，且网络达到稳定。

若 $\boldsymbol{V}(t)\neq\boldsymbol{V}(t+1)=\boldsymbol{V}(t-1)$，则 $\Delta E=0$，且网络到达周期为 2 的极限环。

证毕。

推论：(1) 如 $\boldsymbol{W}$ 正定，$I_i=0,\forall i$ 成立；则

$\boldsymbol{V}(t+1)=\boldsymbol{V}(t)$；$\boldsymbol{V}(t+1)=\mathrm{sgn}[\boldsymbol{W}\boldsymbol{V}(t)]=\boldsymbol{V}(t)$；

网络必定达到稳定收敛。

(2) 如 $\boldsymbol{W}$ 负定，$I_i=0,\forall i$；则

$$V(t+1)\fallingdotseq \mathrm{sgn}[WV(t)]=-V(t)$$
$$V(t+1)=-V(t)=V(t-1)$$

网络周期振荡，振荡极限环为 2。

四、外积型 DHNN 权的设计和讨论

用输入样本矢量的外积来设计 DHNN 的权，这种方法我们称为外积型法。

1. 外积型设计权的基本公式

外积型设计的权应符合 Hebb 学习律，主要考虑存贮 m 个记忆样本 V^k，$k=1,2\cdots,m$，$V^k\in R^n$；其每个分量为 $V_i^k, i=1,2,\cdots,n$。利用已知需存贮的样本来设计 n 个神经元之间的联接权，如 i、j 表示两个不同的神经元，他们间的权为

$$\begin{cases} w_{ij}=\alpha\sum\limits_{k=1}^{m}V_i^kV_j^k & i\fallingdotseq j \\ w_{ii}=0 & i=j \end{cases} \tag{3-21}$$

α 为一个正常数，设初始时 $w_{ij}=0$，当一个样本出现时，在权上就加上一个修改量 $w_{ij}=w_{ij}+\alpha V_i^kV_j^k$，当第 k 个样本的 V_i^k 与 V_j^k 同时兴奋或同时抑制时，$\alpha V_i^kV_j^k>0$，当 V_i^k, V_j^k 中一个兴奋一个抑制时，$\alpha V_i^kV_j^k<0$，这就和 Hebb 提出的生物中神经细胞之间的规律相同。用 Hebb 学习律得到的权可以满足上面提到的 $w_{ij}=w_{ji}$ 的对称条件，从而在异步工作时可保证系统收敛，在同步工作时系统也会收敛或出现极限环为 2。公式 (3-21) 是在输出 $V_i, V_j\in\{1,-1\}$ 的情况下讨论的，当 $V_i, V_j\in\{0,1\}$ 的二值时，式(3-21)可写为

$$w_{ij}=\sum_{k=1}^{m}(2V_i^k-1)(2V_j^k-1) \tag{3-22}$$

我们仍以式(3-21)的形式进行讨论。

把式(3-21)写成矩阵形式，且 $\alpha=1$，对于需存贮的样本 $V^k, k=1,2,\cdots,m$，其权为

$$W=[V^1V^2\cdots V^m]\begin{bmatrix} V^{1T} \\ V^{2T} \\ \vdots \\ V^{mT} \end{bmatrix}-mU$$

$$=\begin{bmatrix} V_1^1 & V_1^2 & \cdots & V_1^k & \cdots & V_1^m \\ V_2^1 & V_2^2 & \cdots & V_2^k & \cdots & V_2^m \\ \vdots & \vdots & & \vdots & & \vdots \\ V_n^1 & V_n^2 & \cdots & V_n^k & \cdots & V_n^m \end{bmatrix}\begin{bmatrix} V_1^1 & V_2^1 & \cdots & V_n^1 \\ V_1^2 & V_2^2 & \cdots & V_n^2 \\ \vdots & \vdots & & \vdots \\ V_1^m & V_2^m & \cdots & V_n^m \end{bmatrix}$$

$$-m\begin{bmatrix} 1 & \cdot & \cdot & \cdot & 0 \\ \cdot & 1 & & & \cdot \\ \cdot & & 1 & & \cdot \\ \cdot & & & 1 & \cdot \\ 0 & \cdot & \cdot & \cdot & 1 \end{bmatrix}$$

$$=\begin{bmatrix} \sum_{k=1}^{m}(V_1^k)^2 & \sum_{k=1}^{m}V_1^kV_2^k & \cdots & \sum_{k=1}^{m}V_1^kV_n^k \\ \sum_{k=1}^{m}V_2^kV_1^k & \sum_{k=1}^{m}(V_2^k)^2 & \cdots & \vdots \\ \vdots & \vdots & \ddots & \\ \sum_{k=1}^{m}V_1^kV_n^k & \cdots & & \sum_{k=1}^{m}(V_n^k)^2 \end{bmatrix}$$

$$-m\begin{bmatrix} 1 & \cdots & 0 \\ & 1 & \\ \vdots & & \vdots \\ & 1 & \\ & & 1 \\ 0 & \cdots & 1 \end{bmatrix}$$

由于 $$\sum_{k=1}^{m}(V_1^k)^2=\sum_{k=1}^{m}(V_2^k)^2\cdots=\sum_{k=1}^{m}(V_n^k)^2=m$$

这儿 U 是一个单位矩阵。

所以

$$W=\begin{bmatrix} 0 & \sum_{k=1}^{m}V_1^kV_2^k & \cdots & \sum_{k=1}^{m}V_1^kV_n^k \\ \sum_{k=1}^{m}V_2^kV_1^k & 0 & \cdots & \vdots \\ \vdots & & \ddots & \\ \sum_{k=1}^{m}V_1^kV_n^k & \cdots & & 0 \end{bmatrix} \tag{3-23}$$

式(3-23)与式(3-21)是相同的，满足：$w_{ii}=0; w_{ij}=w_{ji}$，且设外界输入 $I=0$，我们再来分析用外积的 Hebb 学习律得到的权是否能收敛到所要求的存贮样本上来。

2. 稳定点的讨论

(1) 如果要求存贮的样本是两两正交，对于样本 $V^k, k=1,2,\cdots,m$，V^i、V^j 为 V^k 中任意两个不同的矢量，且满足 $(V^i)^T(V^j)=0, i,j\in k, i\neq j, \forall i,j$ 成立。

在神经元为二值输出的情况下：$V_i\in\{-1,1\}$，当二个 n 维样本矢量的各个分量中有 $\frac{n}{2}$ 个是相同的，有 $\frac{n}{2}$ 个为相反的，就能满足这两个矢量正交。用上面外积型法所得到的权进行迭代计算，在输入样本 $V^k, k=1,2,\cdots,m$ 中任取一个 V^i 作为初始输入，可得

$$WV^i=[V^1V^2\cdots V^i\cdots V^m]\begin{bmatrix} V^{1T} \\ V^{2T} \\ \vdots \\ V^{iT} \\ \vdots \\ V^{mT} \end{bmatrix}V^i-mUV^i$$

$$=[\boldsymbol{V}^1\boldsymbol{V}^2\cdots\boldsymbol{V}^i\cdots\boldsymbol{V}^m]\begin{bmatrix}0\\0\\\boldsymbol{V}^{iT}\boldsymbol{V}^i\\0\\0\\0\end{bmatrix}-m\boldsymbol{U}\boldsymbol{V}^i=n\boldsymbol{V}^i-m\boldsymbol{V}^i=(n-m)\boldsymbol{V}^i$$

只要满足 $n>m$，则 $\mathrm{sgn}[\boldsymbol{W}\boldsymbol{V}^i]=\boldsymbol{V}^i$，则 V^i 为网络的一个稳定点。

(2) 如果输入 m 个记忆样本不是两两正交，$\boldsymbol{V}^k, k=1,2,\cdots,m$，为 n 维矢量，其联接权仍按 Hebb 学习律设计，在 m 个记忆样本中任选一个 $\boldsymbol{V}^i$ 样本输入：

$$\boldsymbol{W}\boldsymbol{V}^i=[\boldsymbol{V}^1\boldsymbol{V}^2\cdots\boldsymbol{V}^i\cdots\boldsymbol{V}^m]\begin{bmatrix}\boldsymbol{V}^{1T}\\\boldsymbol{V}^{2T}\\\vdots\\\boldsymbol{V}^{iT}\\\vdots\\\boldsymbol{V}^{mT}\end{bmatrix}\boldsymbol{V}^i-m\boldsymbol{V}^i$$

通过上式可求得新的输出 $\boldsymbol{V}^{i'}=\mathrm{sgn}(\boldsymbol{W}\boldsymbol{V}^i)$，取 $\boldsymbol{V}^{i'}$ 的第 j 个分量：

$$V_j^{i'}=\mathrm{sgn}[\sum_{i=1}^{n}w_{ji}V_i^i]=\mathrm{sgn}[\sum_{i=1}^{n}V_i^1V_j^1V_i^i+\sum_{i=1}^{n}V_i^2V_j^2V_i^i+\cdots$$

$$+\sum_{i=1}^{n}V_i^mV_j^mV_i^i]=\mathrm{sgn}[\sum_{i=1}^{n}V_j^i(V_i^i)^2+\sum_{\substack{k=1\\k\neq i}}^{m}\sum_{i=1}^{n}V_i^kV_j^kV_i^i]$$

$$=\mathrm{sgn}(s_j+n_j)\qquad(3\text{-}24)$$

式中 $$s_j=nV_j^i;\ n_j=\sum_{\substack{k=1\\k\neq i}}^{m}\sum_{i=1}^{n}V_i^kV_j^kV_i^i$$

设 n_j 为一个零均值的随机变量，$V_i^k, V_j^k, V_i^i\in\{-1,1\}$，而 n_j 的方差 $\sigma^2=(m-1)n$，$\sigma=\sqrt{(m-1)n}$

对于非正交的学习样本，如果满足：$n>\sqrt{(m-1)n}$，则网络仍可收敛到其存贮样本上。

从上面的分析可以看出，对于正交样本，按照 Hebb 学率，它的稳定点与要记忆的样本相同，对于非正交的记忆样本，网络不能保证收敛到所希望的记忆样本上，只有当 $s_j>n_j$ 时，其输出近似为所要求的输出。

3. 外积型设计的 DHNN 网络的性质

(1) 若 $\boldsymbol{V}^i$ 是一个稳定的记忆，$\boldsymbol{V}^i\in R^n$，则 $-\boldsymbol{V}^i$ 也是一个稳定记忆。($\boldsymbol{W}\cdot\boldsymbol{V}^i=0$ 的情况例外)

证明：$\boldsymbol{V}^i$ 是一个稳定点，则 $\boldsymbol{V}^i=\mathrm{sgn}(\boldsymbol{W}\cdot\boldsymbol{V}^i)$

$\boldsymbol{W}\in R^{n\times n}$

用 $-\boldsymbol{V}^i$ 替代 $\boldsymbol{V}^i$，可得

$$\mathrm{sgn}[\boldsymbol{W}(-\boldsymbol{V}^i)]=\mathrm{sgn}(-\boldsymbol{W}\boldsymbol{V}^i)$$

$$=-\mathrm{sgn}(\boldsymbol{W}\boldsymbol{V}^i)=-\boldsymbol{V}^i$$

(2) 用 $d_H(\boldsymbol{V}^i,\boldsymbol{V}^k)$ 表示两个矢量 $\boldsymbol{V}^i$、$\boldsymbol{V}^k$ 之间的海明距离，定义为

$$d_H(\boldsymbol{V}^i,\boldsymbol{V}^k=No.\ of\ i:\ V_i^i\neq V_i^k\quad \forall i \tag{3-25}$$

它等于两个矢量中不相同的分量数目。如果 $d_H(\boldsymbol{V}^i,\boldsymbol{V}^k)=1$，或 $d_H(\boldsymbol{V}^i,\boldsymbol{V}^k)=(n-1)$，则 $\boldsymbol{V}^i$ 是网络的稳定点，$\boldsymbol{V}^k$ 一定不是网络的稳定点。

证明：若 $d_H(\boldsymbol{V}^i,\boldsymbol{V}^k)=n-1$，则 $d_H(-\boldsymbol{V}^i,\boldsymbol{V}^k)=1$，根据性质(1)，$\boldsymbol{V}^i$ 为网络的稳定点，$-\boldsymbol{V}^i$ 也是网络的稳定点，因此我们只需考虑 $d_H(\boldsymbol{V}^i,\boldsymbol{V}^k)=1$ 的情况。为了简单起见，设 $V_1^i\neq V_1^k$，而 $V_i^i=V_i^k, i=2,3,\cdots,n$，因为 $w_{11}=0$，所以有：

$$V_1^i=\mathrm{sgn}\left(\sum_{i=2}^{n}w_{i1}V_i^i\right)$$

由于 $\boldsymbol{V}^i$ 为网络的稳定点，所以有：

$$V_1^i=-V_1^k=\mathrm{sgn}\left(\sum_{i=2}^{n}w_{i1}V_i^i\right)=\mathrm{sgn}\left(\sum_{i=2}^{n}w_{i1}V_i^k\right)\neq V_1^k$$

所以矢量 $\boldsymbol{V}^k$ 不会是系统的稳定点。

(3) 设 m 个记忆样本矢量 $\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m,\boldsymbol{V}\in R^n$ 满足：

$$\alpha n\leqslant d_H(\boldsymbol{V}^i,\boldsymbol{V}^j)\leqslant(1-\alpha)n\qquad \text{对一切 } i\neq j \text{ 成立}$$

$0<\alpha<\frac{1}{2}$；$\boldsymbol{V}^i$，$\boldsymbol{V}^j$ 是样本集中两个不同矢量，$i,j=1,2,\cdots,m$。

若满足：$m\leqslant M\triangleq 1+\left|\frac{1}{1-2\alpha}\right|$，则当 n 足够大时，$\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m$ 为外积型设计的权所形成的网络的稳定点。

证明：由于 $d_H(\boldsymbol{V}^i,\boldsymbol{V}^j)\geqslant\alpha n$

所以

$$(\boldsymbol{V}^i)^T\cdot\boldsymbol{V}^j\begin{cases}\leqslant n-2\alpha n & i\neq j\\ =n & i=j\end{cases} \tag{3-26}$$

用 $\beta_1,\beta_2,\cdots,\beta_m$ 表示第 $\boldsymbol{V}^i$ 个样本与其他 m 个样本之间的内积，计算在 $\boldsymbol{V}^i$ 输入时与 $\boldsymbol{W}$ 的乘积：

$$\boldsymbol{W}\boldsymbol{V}^i=[\boldsymbol{V}^1\boldsymbol{V}^2\cdots\boldsymbol{V}^i\cdots\boldsymbol{V}^m]\begin{bmatrix}\boldsymbol{V}^{1T}\\ \boldsymbol{V}^{2T}\\ \vdots\\ \boldsymbol{V}^{iT}\\ \vdots\\ \boldsymbol{V}^{mT}\end{bmatrix}\boldsymbol{V}^i-m\boldsymbol{U}\boldsymbol{V}^i$$

其中 $\boldsymbol{U}$ 为单位矩阵，$\boldsymbol{U}\in R^{n\times n}$

$$\boldsymbol{W}\boldsymbol{V}^i=[\boldsymbol{V}^1\boldsymbol{V}^2\cdots\boldsymbol{V}^i\cdots\boldsymbol{V}^m]\begin{bmatrix}\beta_1\\ \beta_2\\ \vdots\\ n\\ \vdots\\ \beta_m\end{bmatrix}-m\boldsymbol{U}\boldsymbol{V}^i$$

$$=[\boldsymbol{V}^1\boldsymbol{V}^2\cdots\boldsymbol{V}^i\cdots\boldsymbol{V}^m]\left\{\begin{bmatrix}0\\0\\\vdots\\n\\\vdots\\0\end{bmatrix}+\begin{bmatrix}\beta_1\\\beta_2\\\vdots\\0\\\vdots\\\beta_m\end{bmatrix}\right\}-m\boldsymbol{U}\boldsymbol{V}^i$$

$$=(n-m)\boldsymbol{V}^i+\begin{bmatrix}\sum_{j=1}^{m}V_1^j\beta_j\\\sum_{j=1}^{m}V_2^j\beta_j\\\vdots\\\sum_{j=1}^{m}V_n^j\beta_j\end{bmatrix}_{j\neq i} \tag{3-27}$$

式 (3-27) 中第 2 项的每个元为

$$\sum_{\substack{j=1\\j\neq i}}^{m}V_i^j\beta_j\leqslant\left|\sum_{\substack{j=1\\j\neq i}}^{m}V_i^j\beta_j\right|\leqslant\sum_{\substack{j=1\\j\neq i}}^{m}|\beta_j|\leqslant(m-1)n(1-2\alpha) \tag{3-28a}$$

其中 $|\beta_j|=|\boldsymbol{V}^{jT}\boldsymbol{V}^i|\leqslant n(1-2\alpha)$

若要求网络能收敛到记忆样本集上，即 $\boldsymbol{V}^i=\mathrm{sgn}[\boldsymbol{W}\boldsymbol{V}^i]$ 则有：

$$(n-m)\geqslant(m-1)n(1-2\alpha)$$

$$\frac{n-m}{n}\geqslant(m-1)(1-2\alpha)$$

在 n 充分大时，可得

$$1\geqslant(m-1)(1-2\alpha)$$

而

$$m\leqslant1+\frac{1}{1-2\alpha}$$

所以，当 $0<\alpha<\frac{1}{2}$ 时，有

$$m\leqslant1+\left|\frac{1}{1-2\alpha}\right|$$

证毕。

(4) 给定一组正交矢量 $\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m$，以及另一个矢量 $\boldsymbol{x}$，如果 $d_H(\boldsymbol{x},\boldsymbol{V}^i)<\frac{n-m}{2m}$，则 $\boldsymbol{x}$ 将被吸引到 $\boldsymbol{V}^i$ 上。

证明：设 $\boldsymbol{x}=\boldsymbol{V}^i-2\boldsymbol{b}$，其中 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^T,\boldsymbol{x}\in R^n,\boldsymbol{b}=(b_1,b_2,\cdots,b_n)^T,\boldsymbol{b}\in R^n$。

对 $\boldsymbol{b}$ 每个分量满足 $|b_i|=\begin{cases}1;&x_i\neq V_i^i\\0;&x_i=V_i^i\end{cases}$

因为存在 $d_H(\boldsymbol{x},\boldsymbol{V}^i)$，表示 $\boldsymbol{x}$ 与 $\boldsymbol{V}^i$ 有 d_H 个单元不相等。令 $\boldsymbol{b}=[b_1,b_2,\cdots,b_{d_H},0,\cdots,0]^T$

给网络输入 $\boldsymbol{V}^i$，可计算得到：

$$WV^i=[V^1V^2\cdots V^m]\begin{bmatrix}V^{1T}\\V^{2T}\\\vdots\\V^{mT}\end{bmatrix}V^i-mUV^i$$

$$=(n-m)V^i \tag{3-28b}$$

再给网络输入x，则有

$$Wx=[V^1V^2\cdots V^m]\begin{bmatrix}V^{1T}\\V^{2T}\\\vdots\\V^{mT}\end{bmatrix}(V^i-2b)-mU(V^i-2b)$$

$$=WV^i-W2b \tag{3-28c}$$

其中$2Wb$展开为

$$2Wb=2W\begin{bmatrix}b_1\\b_2\\\vdots\\b_{d_H}\\0\\0\\0\end{bmatrix}=2\begin{bmatrix}\sum_{i=1}^{d_H}b_iw_{1i}\\\sum_{i=1}^{d_H}b_iw_{2i}\\\vdots\\\sum_{i=1}^{d_H}b_iw_{ni}\end{bmatrix} \tag{3-28d}$$

因为
$$w_{ji}=\begin{cases}\sum_{k=1}^{m}V_i^kV_j^k & i\neq j\\0 & i=j\end{cases}$$

所以
$$|w_{ji}|\leqslant\sum_{k=1}^{m}|V_i^k||V_j^k|=m$$

代入(3-28d)矢量中的一个分量，则有

$$\sum_{i=1}^{d_H}b_iw_{ji}\leqslant\sum_{i=1}^{d_H}|b_i||w_{ji}|=d_Hm \tag{3-28e}$$

由式(3-28b)、(3-28c)可得

$$Wx=(n-m)V^i-W2b$$

利用式(3-28d)、(3-28e)，若x被V^i吸引，对每个分量有$|(n-m)V_i^i|\geqslant2\sum_{i=1}^{d_H}b_iw_{ji}$，因为$V_i^i=1$，则有

$$(n-m)\geqslant2d_Hm$$

所以
$$d_H\leqslant\frac{n-m}{2m}$$

证毕。

(5) 若$V^1,V^2,\cdots,V^m$是网络的稳定点，x是由$\{V^1,V^2,\cdots,V^m\}$线性组合的n维矢量，则x也为网络的稳定点。

证明：设$x=\sum_{i=1}^{m}a_iV^i$

则
$$\boldsymbol{W}\boldsymbol{x}=\sum_{i=1}^{m}a_i\boldsymbol{W}\boldsymbol{V}^i=\sum_{i=1}^{m}a_i\boldsymbol{V}^i=\boldsymbol{x}$$

即
$$\mathrm{sgn}(\boldsymbol{W}\boldsymbol{x})=\boldsymbol{x}$$

由外积型设计的 DHNN 系统除了有这五个性质外，还具有以下的特点：

(1) 系统存在不少的伪稳定点，这些稳定点并不是系统要求的吸引子，我们称作伪稳定解。如 $\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m$ 是系统要求的稳定点，则 $-\boldsymbol{V}^1,-\boldsymbol{V}^2,\cdots,-\boldsymbol{V}^m$ 也是系统的稳定点，$\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m$ 的线性组合亦为其稳定点。如果这两类稳定点不是系统要求的，那么这些就是伪稳定点。

(2) 如果一个系统设计后，使 $\boldsymbol{V}^i$ 为其演化的最终稳定点，那么，那些与 $\boldsymbol{V}^i$ 只差一个海明距离的矢量 $\boldsymbol{V}^j,d_H(\boldsymbol{V}^j,\boldsymbol{V}^i)=1$ 就不能成为系统的稳定点。

(3) 要求设计的记忆样本的海明距离拉得越远，那么记忆的数量就可能越大。

(4) 对于正交的输入记忆样本，每个样本不仅能够收敛到自己，而且存在一个吸引域，此吸引域至少为
$$d_H\leqslant\frac{n-m}{2m}$$

4. 外积型求 DHNN 权的举例

用外积型设计 DHNN 主要用于联想记忆，步骤如下：

(1) 根据需要记忆的样本 $\boldsymbol{V}^1,\cdots,\boldsymbol{V}^m$，用外积型设计权
$$w_{ij}=\begin{cases}\sum_{k=1}^{m}V_i^kV_j^k & i\neq j\\ 0 & i=j\end{cases}$$
$$I_j=0$$

(2) 令输入样本、或测试样本作为网络输出的初值，令矢量 $\boldsymbol{V}^i$ 为任意的输入矢量，使
$$\boldsymbol{V}(t_0)=\boldsymbol{V}^i,\text{即 }V_i(t_0)=V_i^i,i=1,2,\cdots,n$$
$\boldsymbol{V}$ 为网络的输出，$\boldsymbol{V}\in R^n$

(3) 用下面的迭代公式进行演算：
$$V_i(t+1)=f_n[\sum_{j=1}^{n}w_{ij}V_j(t)]$$
$$f_n(x)=\mathrm{sgn}(x)$$

(4) 重复迭代直至每个输出单元不变为止，即
$$V_i(t+1)=V_i(t)\qquad\forall i\text{ 成立}$$

此时 $\boldsymbol{V}^i$ 被吸引到已学习过的记忆样本的某个吸引子上。

[例 3-1] 对 $n=5$ 的 DHNN 网络，其要求的记忆样本为

$\boldsymbol{V}^1=(1,1,1,1,1)^T$　　$\boldsymbol{V}^2=(1,-1,-1,1,-1)^T$　　$\boldsymbol{V}^3=(-1,1,-1,-1,-1)^T$

它们并不满足正交条件，按外积型计算其权矩阵可得到 $\boldsymbol{W}=\sum_{i=1}^{3}\boldsymbol{V}^i\boldsymbol{V}^{iT}-3\boldsymbol{U}$；$\boldsymbol{U}$ 为一个单位矩阵 $\boldsymbol{U}\in R^5$。

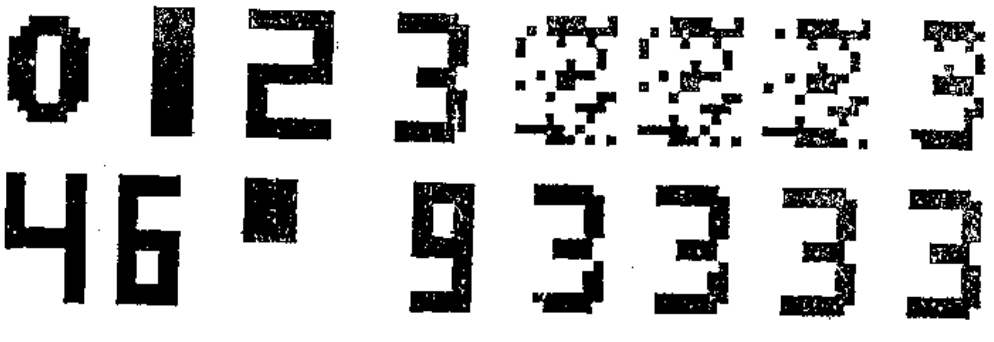
$$W=\begin{bmatrix} 0 & -1 & 1 & 3 & 1 \\ -1 & 0 & 1 & -1 & 1 \\ 1 & 1 & 0 & 1 & 3 \\ 3 & -1 & 1 & 0 & 1 \\ 1 & 1 & 3 & 1 & 0 \end{bmatrix}$$

计算可以得到 $\mathrm{sgn}(WV^1)=V^1, \mathrm{sgn}(WV^2)=V^2, \mathrm{sgn}(WV^3)=V^3$,

该网络可能的输出状态为 $2^n=32$ 种矢量，而分析一下其稳定点的情况，系统共有四个稳定点，分别为 $V^1=(1,1,1,1,1)^T$，$V^2=(1,-1,-1,1,-1)^T$，$V^3=(-1,1,-1,-1,-1)^T$，$V^4=(-1,1,1,-1,1)^T$，V^1、V^2、V^3 为要求的稳定点，V^4 为伪稳定点，$V^4=-V^2$。在用串行方式工作的情况下，其中 8 个初始态收敛到 V^1，9 个初始态收敛到 V^2，5个初使态收敛到 V^3，10 个初使态收敛到 V^4，四个稳定点都是渐近稳定点。在并行方式工作情况下，其中 10 个初始态收敛到 V^1，1 个初始态收敛到 V^2，2 个初始态收敛到 V^3，1 个初始态收敛到 V^4，在其他 18 个初始态情况下，网络都陷入了极限环，因此其收敛域明显减小。

[例 3-2] 外积型设计权。

对于 n 比较大的情况，其联想效果比较明显，图 3-5a 是 8 个要求记忆的样本，通过外积型权的设计，可得到它的 120 个神经元之间的联接权，通过学习后的网络它在此 8 个样本输入时能收敛到自己。图 3-5b 是加有噪声的一个样本，噪声是加在字母"3"上，通过网络的迭代，此网络能联想到一个字母"3"上，从而达到联想记忆的目的。



(a) (b)

图 3-5

(a) 输入的记忆样本；(b) 有噪声干扰的样本能迭代收敛

五、用其他方法进行权的设计

用 Hebb 规则设计的权能够保证其网络在异步工作时，它能稳定收敛，尤其在记忆样本是正交的条件下，它保证了每个记忆样本能够收敛到自己，并且有一定范围的吸引域，但对于那些不正交的记忆样本，它不一定能收敛到本身。下面介绍的两种方法改进了这种情况，具有各自的特点。

1. 伪逆法

设一个 n 维矢量输入到 n 个神经元，考虑存在着 m 个样本 $V^1, V^2, \cdots, V^m$，则把样本按序排列成一个矩阵：

$$X=[V^1, V^2, \cdots V^k, \cdots, V^m]; X\in R^{n\times m}$$

设神经元的输出可写为一个矩阵 $\boldsymbol{Y}$，与 $\boldsymbol{X}$ 相对应，$\boldsymbol{Y}' \in R^{n\times m}$，输入与输出之间是用一个 $n\times n$ 的权矩阵 $\boldsymbol{W}$ 来映照，$\boldsymbol{W}\in R^{n\times n}$，且 $\boldsymbol{I}=0$，则有

$$\boldsymbol{WX}=\boldsymbol{Y}';\boldsymbol{Y}=\mathrm{sgn}(\boldsymbol{Y}')$$

可得

$$\boldsymbol{W}=\boldsymbol{Y}'\boldsymbol{X}^{*} \tag{3-29}$$

$\boldsymbol{X}^{*}$ 为 $\boldsymbol{X}$ 的伪逆，且 $\boldsymbol{X}^{*}$ 满足

$$\boldsymbol{X}^{*}=(\boldsymbol{X}^{T}\boldsymbol{X})^{-1}\boldsymbol{X}^{T}$$

在输入 m 个样本时，如样本之间是线性无关的，则其 $(\boldsymbol{X}^{T}\boldsymbol{X})$ 为满秩，$\mathrm{Rank}(\boldsymbol{X}^{T}\boldsymbol{X})=m$，$(\boldsymbol{X}^{T}\boldsymbol{X})^{-1}$ 存在，则 $\boldsymbol{W}$ 就可以解出来。

用伪逆法求出的权 $\boldsymbol{W}$，可以保证在自己输入时仍能收敛到样本自己。如果 $\boldsymbol{Y}'$ 与输入 $\boldsymbol{X}$ 完全相同，则 $\boldsymbol{W}$ 也可以是对称的，因而满足稳定工作的条件。其实只要满足 $\boldsymbol{Y}$ 矩阵中每一个元与 $\boldsymbol{WX}$ 矩阵中的每个元有相同的符号，就可以满足收敛到本身。

例如在〔例 3-1〕中，$\boldsymbol{V}^{1}=[1,1,1,1,1]^{T}$，$\boldsymbol{V}^{2}=[1,-1,-1,1,-1]^{T}$，$\boldsymbol{V}^{3}=[-1,1,1,-1,-1,]^{T}$。

$$\boldsymbol{X}=\begin{bmatrix}1 & 1 & -1\\ 1 & -1 & 1\\ 1 & -1 & 1\\ 1 & 1 & -1\\ 1 & -1 & -1\end{bmatrix}\quad \text{设}\quad \boldsymbol{Y}'=\begin{bmatrix}0.5 & 0.5 & -0.1\\ 0.5 & -0.1 & 0.5\\ 0.5 & -0.1 & 0.5\\ 0.5 & 0.5 & -0.1\\ 0.1 & -0.5 & -0.5\end{bmatrix}$$

则

$$\boldsymbol{W}=\boldsymbol{Y}'\boldsymbol{X}^{*}=\boldsymbol{Y}(\boldsymbol{X}^{T}\boldsymbol{X})^{-1}\boldsymbol{X}^{T}=\begin{bmatrix}0.25 & 0.1 & 0.1 & 0.25 & -0.2\\ 0.1 & 0.25 & 0.25 & 0.1 & -0.2\\ 0.1 & 0.25 & 0.25 & 0.1 & -0.2\\ 0.25 & 0.1 & 0.1 & 0.25 & -0.2\\ -0.1 & -0.1 & -0.1 & -0.1 & 0.5\end{bmatrix}$$

$\boldsymbol{W}$ 可以保证输入 $\boldsymbol{V}^{1},\boldsymbol{V}^{2},\boldsymbol{V}^{3}$ 时收敛到自己。

2. 正交化的权设计

这一方法是由 L_i 和 Mechel 提出来的，其出发点为：

(1) 要保证系统在异步工作时的稳定性，则它的权是对称的，满足 $w_{ij}=w_{ji}$，$i,j=1,2,\cdots,n$（如果系统由 n 个神经元组成）。

(2) 要保证所有的要求的记忆样本都能收敛到自己，不会出现错误的其他收敛值。

(3) 要求伪稳定点的数目尽可能地少。

(4) 要求稳定点其吸引域尽可能地大。

其转换公式为

$$\mathrm{sgn}[\boldsymbol{WV}(t)+\boldsymbol{I}]=\boldsymbol{V}(t+1);\quad \boldsymbol{I}\neq 0 \tag{3-30}$$

正交化的权的计算公式推导如下：

(1) 已知有 m 个记忆样本矢量为 $\boldsymbol{V}^{1},\boldsymbol{V}^{2},\cdots,\boldsymbol{V}^{m},\boldsymbol{V}\in R^{n}$，计算 $n\times(m-1)$ 阶矩阵 $\boldsymbol{Y}\in R^{n\times(m-1)}$：

$$\boldsymbol{Y}=[\boldsymbol{V}^{1}-\boldsymbol{V}^{m},\boldsymbol{V}^{2}-\boldsymbol{V}^{m},\cdots,\boldsymbol{V}^{m-1}-\boldsymbol{V}^{m}]^{T}$$

(2) 对 Y 进行奇异值及酋矩阵分解，如存在两个正交阵 P，Q 和一个对角值为 Y 的奇异值的对角矩阵 A，满足：

$$Y=PAQ^T$$

$$Y=[Y_1,Y_2,\cdots,Y_{m-1}]^T \qquad Y\in R^{n\times(m-1)}$$

$$P=[P_1,P_2,\cdots,P_n]^T \qquad P\in R^{n\times n}$$

$$Q=[Q_1,Q_2,\cdots,Q_{m-1}]^T \qquad Q\in R^{(m-1)\times(m-1)}$$

$$A=\begin{bmatrix} \lambda_1 & \cdots & 0 \\ & \ddots & \\ \vdots & \lambda_k & \vdots \\ & \ddots & \\ 0 & \cdots & 0 \end{bmatrix} \qquad A\in R^{n\times(m-1)}$$

k 维空间为 n 维空间的子空间，它由 k 个独立基组成：

$$k=\mathrm{Rank}(A)$$

设 $\{P_1,P_2,\cdots P_k\}$ 为 Y 的正交基，而 $\{P_{k+1},P_{k+2},\cdots,P_n\}$ 为在 n 维空间中的补充正交基，从 P 矩阵中派生出来的权可以进一步按下面的方法设计。

(3) 计算这 n 维网络的权。定义：

$$W^+=[w_{ij}]^+=\sum_{i=1}^{k}P_iP_j^T \qquad W^+\in R^{n\times n}$$

$$W^-=[w_{ij}]^-=\sum_{\lambda=k+1}^{n}P_iP_j^T \qquad W^-\in R^{n\times n}$$

而总的联接权可以设计为

$$W_\tau=W^+-\tau W^- \tag{3-31}$$

τ 为一个参数，$\tau>-1$。

(4) 网络存在一个阈值 I_τ。定义：

$$I_\tau=V^m-W_\tau V^m \tag{3-32}$$

按上述正交化的方法设计权可以保证满足开始提出的四个条件。下面我们来逐一分析。

首先，网络的权是用两部分的权 W^+ 与 W^- 相加而成的，而每一部分用的都是类似于外积型法得到的，只是用的不是原始要求记忆的样本，而是分解后正交阵 P 的分量，这两部分权都是满足对称条件的，即

$$w_{ij}^+=w_{ji}^+; \quad w_{ij}^-=w_{ji}^-$$

因而 W_τ 中的分量也满足对称条件，这就保证了系统在异步工作时收敛并且不会出现极限环。

第二，虽然 $V^1,V^2,\cdots,V^m$ 不是正交的，但是在记忆样本集中的每个矢量都能收敛到自己。

证明： 在样本集中任选一个记忆矢量 V^i，因为 (V^i-V^m) 是 Y 中的一个矢量，它是属于 A 的秩所定义的 k 个基空间中的矢量，必存在一些系数 $\sigma_1,\sigma_2,\sigma_3,\cdots,\sigma_k$，使

$$V^i-V^m=\sigma_1P_1+\sigma_2P_2+\cdots\sigma_kP_k \tag{3-33}$$

$$V^i=\sigma_1P_1+\sigma_2P_2+\cdots+\sigma_kP_k+V^m \tag{3-34}$$

对于 $P_1, P_2, \cdots, P_M$ 中任一个基 P_i 有

$$W_\tau P_i = W^+ P_i - \tau W^- P_i = P_i \tag{3-35}$$

因而当 $V^i(t)$ 输入时，有输出为

$$V^i(t+1) = \mathrm{sgn}(W_\tau V^i_{(t)} + I_\tau)$$

其中 $W_\tau V^i_{(t)} + I_\tau = W^+ V^i_{(t)} - \tau W^- V^i_{(t)} + V^m_{(t)} - W^+ V^m_{(t)} + \tau W^- V^m_{(t)}$

$$= W^+ (V^i_{(t)} - V^m_{(t)}) - \tau W^- (V^i_{(t)} - V^m_{(t)}) + V^m(t)$$

根据式(3-31)、(3-33)、(3-35)得

$$W_\tau V^i + I_\tau = V^i_{(t)} - V^m_{(t)} + V^m_{(t)} = V^i(t)$$

所以

$$V^i(t+1) = \mathrm{sgn}(W_\tau V^i_{(t)} + I_\tau) = V^i(t)$$

如在记忆样本中选第 m 个样本作为输入，代入式(3-30)并利用式(3-32)，则得

$$V^m(t+1) = \mathrm{sgn}[W_\tau V^m_{(t)} + I_\tau]$$

$$= \mathrm{sgn}[W_\tau V^m_{(t)} + V^m(t) - W_\tau V^m(t)]$$

$$= \mathrm{sgn}[V^m(t)] = V^m(t)$$

证毕。

第三，可通过参数 τ 的改变来减少伪稳定点的数目，而使收敛域的范围变大。

如果存在一个 V^i，不是记忆样本，其输出

$$V = \mathrm{sgn}(W_\tau V^i + I_\tau)$$

$$= \mathrm{sgn}[(W^+ - \tau W^-) V^i + I_\tau]$$

因为 V^i 不是学习记忆样本，必有

$$(W^+ - \tau W^-) V^i \neq V^i$$

τ 是一个可调参数，调节 τ 改变 $(W^+ - \tau W^-) V^i + I_\tau$ 中每个分量的符号，则得

$$\mathrm{sgn}(W V^i + I_\tau) \neq V^i,$$

使 V^i 不能收敛到本身。

利用参数 τ 的调节可以改变伪稳定点的数目，在串行工作的情况下，伪稳定点的减少就相当于每个要求稳定点的稳定域的扩大。对于任意一个不在记忆样本中的 V^i，总可以设计一个 τ，把 V^i 排除在稳定点之外。

表 3-1 给出的是一个 $n=10$，有 5 个学习记忆样本的系统，采用了上面方法进行权的设计，在不同 τ 的时候其稳定点的数目，同时给出外积型权网络的比较。

表 3-1

	正交化方法		外积型权设计
	$\tau=1$	$\tau=10$	
稳定点数	8	5	4
错误稳定点数	0	0	0

六、记忆容量的讨论

一个 DHNN 能记忆的样本矢量数称为这种网络的记忆容量，对于联想记忆来说，重要的是吸引域，而容量和吸引域往往是矛盾的，没有吸引域就没有联想的作用，所以我们并不

过分强调记忆容量的大小。在 n 维空间中的吸引域分析比较困难，我们只能用统计和定性分析的方法来讨论。对于用外积型设计的网络，如果输入样本是正交的，n 个神经元的网络最多有 n 个可能正交的矢量，这样最多能记住 n 个样本，但在大多数的情况下，学习样本不可能正交，例如：要记住字母、文字，这些样本是不可能正交的，而正交样本又没有什么图形上的意义。但如果把样本进行正交变换，或者如上面讨论的方法进行变换，网络的设计就比较复杂，下面主要讨论外积型设计权的网络的记忆容量及提高容量的方法。

1. 外积型法设计的 DHNN 的容量

对于外积法设计的网络，从统计实验看 Hopfield 提出了一个数量级的范围，其记忆容量为 $0.13\sim0.15n$，n 为神经元的数目，初看起来，这个数值太小，因为 n 个神经元输出可能的矢量为 2^n 个，而网络真正能够记忆的仅仅是 $0.13\sim0.15n$。在 n 很大时，两者相差甚大，但从生物的角度看，人脑的神经细胞数目很大，那么记忆的数量还是很可观的。

为了分析非正交学习样本输入时网络的记忆容量，我们考虑式(3-24)：

对于记忆样本集 $\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m$，取出矢量 $\boldsymbol{V}^i$ 作为 DHNN 的初始值 $\boldsymbol{V}^i(t_0)$，在 n 维神经网络中，第 j 个分量输出为

$$V_j^i(t+1)=\operatorname{sgn}(s_j+n_j)\qquad \boldsymbol{V}^i\in R^n \tag{3-24}$$

其中

$$s_j=nV_j^i(t)$$

$$n_j=\sum_{\substack{k=1\\k\neq i}}^{m}\sum_{t=1}^{n}V_t^iV_j^kV_t^i$$

希望 $\boldsymbol{V}^i$ 在 t 时刻后能够收敛到本身，则有

$$V_j^i(t+1)=\boldsymbol{V}_j^i(t)=\boldsymbol{V}^i(t_0)$$

对于一切 j 成立。如果 n_j 比较大，而且 n_j 与 s_j 异号，使得某些分量 $V_j^i(t+1)\neq V_j^i(t)$，那么矢量 $\boldsymbol{V}^i$ 输入到网络后就不能收敛到自己，这些我们称为联想出错。对于某一个分量 j 出错的可能是那些 s_j 与 n_j 异号，且 $|s_j|<|n_j|$ 的情况。如果出错的概率为高斯分布，它的均值为零，方差为 σ，$\sigma=\sqrt{n(m-1)}$，那么 n_j 与 s_j 异号时，其错误概率分布为

$$\varphi(n_j)=\frac{1}{\sqrt{2\pi\sigma}}\int_{-nV_j^i}^{\infty}\mathrm{e}^{-\frac{Z^2}{2\sigma^2}}\,\mathrm{d}Z\qquad \text{对于 } V_j^i<0,n_j>0 \tag{3-36}$$

或

$$\varphi(n_j)=\frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{-nV_j^i}\mathrm{e}^{-\frac{Z^2}{2\sigma^2}}\,\mathrm{d}Z\qquad \text{对于 } V_j^i>0,n_j<0$$

图 3-6 表示出错的范围，其中阴影部分表示 n_j 与 s_j 同号，或 $|s_j|>|n_j|$ 的情况，非阴影部分就是出错可能的范围，而从概率分布的情况看，积分应该从现有的 s_j 地方开始，例如图 3-6 中的 A、B 两点。表示 n_j 与 s_j 的关系的两个点，它们在 x、y 轴上的分量为 A_x、A_y、B_x、B_y，如果 $s_j=A_x$，那么从 A_y 向 $-\infty$ 积分，非阴影部分就包含着所有可能的错误 n_j，同样，如果 $s_j=B_x$，那么从 B_y 开始向 $+\infty$ 积分的非阴影部分包含着所有的可能错误，这样用(3-36)两个公式，对于一个确定的 s_j，只需要一个积分就可以了。从

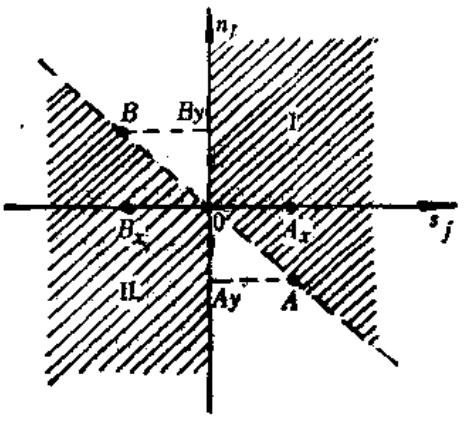


图 3-6 出错范围图

图 3-6 还可以看出从 A_y 和从 B_y 的积分值是相同的，积分从 $|n_j|=|s_j|$ 开始，得到其错误的概率分布为

$$\varphi(s_j)=\int_{+s_j}^{\infty}\frac{1}{\sqrt{2\pi}\,\sigma}\,e^{-\frac{Z^2}{2\sigma^2}}\,dZ \tag{3-37}$$

$$s_j>0,\ s_j=nV_j';\ V_j'>0$$

因为 V_j' 只可能为 ±1，因而 $s_j=n$。

将 σ 放入 Z 中，变量改为 $\frac{Z}{\sigma}$，则(3-37)式为

$$\varphi\left(\sqrt{\frac{n}{m-1}}\right)=\frac{1}{\sqrt{2\pi}}\int_{\sqrt{\frac{n}{m-1}}}^{\infty}e^{-\frac{Z^2}{2}}\,dZ$$

假设正确率服从泊松分布，对 V_i' 中的每一个分量总的错误率为 $n\varphi\left(\sqrt{\frac{n}{m-1}}\right)$，则其正确率为

$$\beta=\exp\left(-n\varphi\left(\sqrt{\frac{n}{m-1}}\right)\right) \tag{3-38}$$

当 $\varphi(0)=0,\beta=1$，表示无错误时，正确率为 100%，将(3-38)式改写为

$$\frac{-\ln\beta}{n}=\varphi\left(\sqrt{\frac{n}{m-1}}\right)=\frac{\alpha}{n}$$

由于 β 总是小于 1，$\alpha=-\ln\beta>0$

得

$$\varphi^{-1}\left(\frac{\alpha}{n}\right)=\sqrt{\frac{n}{m-1}}$$

$$m=\frac{n}{\left[\varphi^{-1}\left(\frac{\alpha}{n}\right)\right]^2}+1$$

当 $\sqrt{\frac{n}{m-1}}>3.5$ 时，$\varphi^{-1}\left(\frac{\alpha}{n}\right)\approx\sqrt{2\ln\frac{n}{\alpha}}$

得

$$m=\frac{n}{2\ln\frac{n}{\alpha}}+1 \tag{3-39}$$

对于一定的正确率，α 为一个常数，可近似地令 $\alpha=1$，则可从式(3-39)得到 $m\approx0.12-0.15n$。从式(3-39)的计算与用统计方法计算的近似容量有同样的数量级。

2. 提高记忆容量和吸引域的方法介绍

用外积型得到权的 DHNN 系统，它的记忆容量是十分小的，用伪逆法设计权的公式为

$$\boldsymbol{W}=\boldsymbol{Y}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X};\ \text{其中}\ \boldsymbol{X}^T=[\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m]$$

对于 m 个样本，每个样本由 n 个分量组成 $n>m$，其 $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ 的秩最大为 m，只要所有的样本是线性无关，那么最大的容量为 m，但并没有考虑其吸引域。

用正交分解的方法，可以讨论两个极限情况，如 $\tau\fallingdotseq0$，则对于任何非正交样本 $\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m$，只要 $m\leqslant n$，通过正交化得到权 $\boldsymbol{W}_\tau=\boldsymbol{W}^+-\tau\boldsymbol{W}^-$，都可以收敛到自己，这样可以收敛到自己的样本数为 m 个，但如果 $m>n$，则 $\boldsymbol{Y}$ 矩阵的最大秩为 n，在任意的 $\tau\fallingdotseq0$ 的情况下，

它们能收敛的稳定点仍为 n 个样本，在正交化方法中其伪稳定点可以通过控制 τ 的大小来排除，因此它比伪逆法和外积法更有优越性，但所有的方法都很难控制其吸引域。下面的几种改进方法都着重于扩大吸引域，抑制伪稳定点。

(1) 高阶关联的网络：用高阶阈值逻辑单元组成的单层反馈网络为高阶关联的DHNN，它的输入、输出关系为

$$y_j=\mathrm{sgn}(u_j)=\mathrm{sgn}[\sum T_i(j)] \tag{3-40}$$

$$T_1(j)=\sum_i w_{ij}V_i$$

$$T_2(j)=\sum_i\sum_k w_{ijk}V_iV_k$$

$$\vdots$$

$$T_m(j)=\sum_{i1}\sum_{i2}\cdots\sum_{im} w_{i_1,i_2,\cdots i_m}\prod_{i=1}^{m}V_i$$

对于 w_{ijk} 也可以用 Hebb 学习律得到，对于不同的样本 V_i^p，$(P=1,2,\cdots,m)$其权为

$$w_{ijk}=\sum_{p=1}^{m}V_i^pV_j^pV_k^p$$

$$i=1,2,\cdots,n;\quad j=1,2,\cdots,n;\quad k=1,2,\cdots,n$$

依次类推，可得到不同阶数的高阶关联网络，网络的输入也是由输入样本的高阶次获得的，下面我们讨论一个比较特殊的两阶输入情况，输入样本为 $\boldsymbol{V}^i$，让输出矢量 $\boldsymbol{Y}$ 写为

$$\boldsymbol{Y}=\mathrm{sgn}\left[(\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m)\begin{bmatrix}\boldsymbol{V}^{1T}\\ \boldsymbol{V}^{2T}\\ \vdots\\ \boldsymbol{V}^{mT}\end{bmatrix}\boldsymbol{V}^i\begin{bmatrix}\boldsymbol{V}^{1T}\\ \boldsymbol{V}^{2T}\\ \vdots\\ \boldsymbol{V}^{mT}\end{bmatrix}\boldsymbol{V}^i\right]$$

$$=\mathrm{sgn}[(\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m)\boldsymbol{V}'^2] \tag{3-41}$$

其中 $\boldsymbol{V}'$ 为

$$\boldsymbol{V}'=\begin{bmatrix}\boldsymbol{V}^{1T}\\ \boldsymbol{V}^{2T}\\ \vdots\\ \boldsymbol{V}^{mT}\end{bmatrix}\boldsymbol{V}^i \tag{3-42}$$

如果输入样本 $\boldsymbol{V}^i$ 不在学习样本集内，它离样本集中 $\boldsymbol{V}^j$ 最近，则 $\boldsymbol{V}'$ 为

$$\boldsymbol{V}'=\begin{bmatrix}\boldsymbol{V}^{1T}\boldsymbol{V}^i\\ \boldsymbol{V}^{2T}\boldsymbol{V}^i\\ \vdots\\ \boldsymbol{V}^{jT}\boldsymbol{V}^i\\ \vdots\\ \boldsymbol{V}^{mT}\boldsymbol{V}^i\end{bmatrix}=\begin{bmatrix}\beta_1\\ \beta_2\\ \vdots\\ \beta_j\\ \vdots\\ \beta_m\end{bmatrix}$$

其中 $\beta_i=\boldsymbol{V}^{iT}\boldsymbol{V}^i;\ i=1,2,\cdots,m$

因为 $\beta_j>\beta_i(i=1,2,\cdots,m),i\neq j$，则 $\beta_j^2\gg\beta_i$，通过平方以后 β_j 的分量突出了，而其他分量被抑制了，所以用 $(\boldsymbol{V}')^2$ 去乘矩阵 $(\boldsymbol{V}^1,\boldsymbol{V}^2,\cdots,\boldsymbol{V}^m)$后，就使 $\boldsymbol{V}^j$ 这个矢量突出了，通过多次迭代，使网络的输出稳定到 $\boldsymbol{V}^j$ 上。这相当于使 $\boldsymbol{V}^j$ 的吸引域加大，让那些非学习的矢

量被拉到了已学习过的那些样本上。同样可以用到更加高的阶数，使 Y 的输出为

$$Y=\mathrm{sgn}\left[(V^1,V^2,\cdots,V^m)\left(\begin{pmatrix}V^{1T}\\V^{2T}\\\vdots\\V^{mT}\end{pmatrix}V^t\right)^k\right] \tag{3-43}$$

这种高阶的非线性网络使系统的吸引域大大增加，而伪稳定点的数目得以减少。

(2) 采用带参数的加强网络：此网络的权矩阵仍然采用外积型，记为 W^t。对于任意一个矢量 V^t 输入，其输出矢量 Y 为

$$\begin{aligned}Y&=\mathrm{sgn}(u)=\mathrm{sgn}[W^tV^t]\\&=\mathrm{sgn}[P(V^{t'}V^{t'T}-U)V^t+\sum_{\substack{k=1\\k\neq t'}}^{m}W_kV^t]\end{aligned} \tag{3-44}$$

其中 $W_k=(V^kV^{kT}-U)$

在式(3-44)中，$V^{t'}$ 为一个与 V^t 最靠近的矢量(即海明距离最小)，而希望 V^t 输入后，最后能收敛到 $V^{t'}$。P 为一个加重参数，当 $P=1$ 时，式(3-44)就是一般的外积求权的DHNN，当 $P>1$ 时，意味着对于重点内容强化记忆，P 愈大，重点内容强化记忆的程度越高，在外积法网络中的权 W 是根据记忆样本进行学习的，一旦学习完了，每个权值就固定了。而在这种参数加强网络中，参数 P 的引入使网络在联想过程中，能依据情况，对重点内容强化记忆。与一般外积法网络相比，它从 i 神经元到 j 神经元的权为

$$w_{ij}=w_{ij}^H+(P-1)V_i^{t'}V_j^{t'} \tag{3-45}$$

其中 w_{ij}^H 为一般外积法的权，$V_i^{t'}V_j^{t'}$ 为第 t' 个样本中第 i 个神经元输出与第 j 个神经元输出之间的乘积，参数 P 的作用使样本 t' 的吸引域扩大了，从而减小了伪稳定点的吸引域，客观上提高了网络的容量和容错能力。

(3) 采用 δ 学习率对网络的权进行调整：在网络的能量函数 E 的讨论中，我们看到的是利用了有明确物理意义的哈密顿函数，其权的对称条件也是因为自旋材料中的分子相互作用(对称的)而借用过来的，能量函数随时间单调下降而要求 $w_{ij}=w_{ji}$，它只是充分条件而不是必要条件，如果其权取得不对称，也可保证其收敛的条件，如：存在一个已知的对称权矩阵 W，满足对所有样本的收敛，即有

$$V^k(t+1)=\mathrm{sgn}(WV^k(t)] \qquad k=1,2,\cdots,m \tag{3-46}$$

如存在一个正定的对角阵 L，则 LW 与 W 性质相同，得

$$\begin{aligned}W'&=\begin{bmatrix}\lambda_1&&&0\\&\lambda_2&&\\&&\ddots&\\0&&&\lambda_n\end{bmatrix}\begin{bmatrix}w_{11}&w_{12}&\cdots&w_{1n}\\\vdots&&&\vdots\\w_{n1}&w_{n2}&\cdots&w_{nn}\end{bmatrix}\\&=\begin{bmatrix}\lambda_1w_{11}&\lambda_1w_{12}&\cdots&\lambda_1w_{1n}\\\lambda_2w_{21}&\lambda_2w_{22}&\cdots&\lambda_2w_{2n}\\\vdots&\vdots&\ddots&\vdots\\\lambda_nw_{n1}&\lambda_nw_{n2}&\cdots&\lambda_nw_{nn}\end{bmatrix}\end{aligned}$$

上式中 W' 不是一个对称阵，但仍然满足式(3-46)，这说明对称的要求不是必要的，因

此可以用迭代法对网络进行调整，最后得到满足要求的权。

第二节　连续的单层反馈网络

连续的单层反馈网络有 Hopfield 提出的 CHNN 网络和 Grossberg 的网络，它的每个神经元的输入与输出关系为连续可微的单调上升函数，它的每个神经元的输入是一个随时间变化的状态变量，它与外界的输入和其他神经元来的信号有直接关系，同时也与其他神经元同它之间的连接权有关系。状态变量直接影响了输入变量，使系统变成一个随时间变化的动态系统。本节主要从 Hopfield 的网络出发来讨论这一类网络的生物背景、数学模型、稳定性以及权的设计和应用。

一、连续的 Hopfield 网络的生物背景和数学模型

在第一章中我们已经看到了一个神经细胞的示意图，神经细胞之间的信息传递是由一串不等间隔的神经冲动(即脉冲)来进行的，在图中，树突接收外界的信号，这里的外界，可以是从其他神经细胞来的脉冲串经过突触后传到树突，也可以是直接由外界的光、声等刺激而传递来的信息，因此突触是神经细胞之间接触的主要部分，突触的结构如图 3-7 所示，它可以是一个神经细胞的轴突末梢与下一个神经细胞树突发生的功能性接触，也可以直接与细体接触，在突触底部有很多小的囊泡，当一个神经冲动到来时，贮存在突触囊泡内的神经

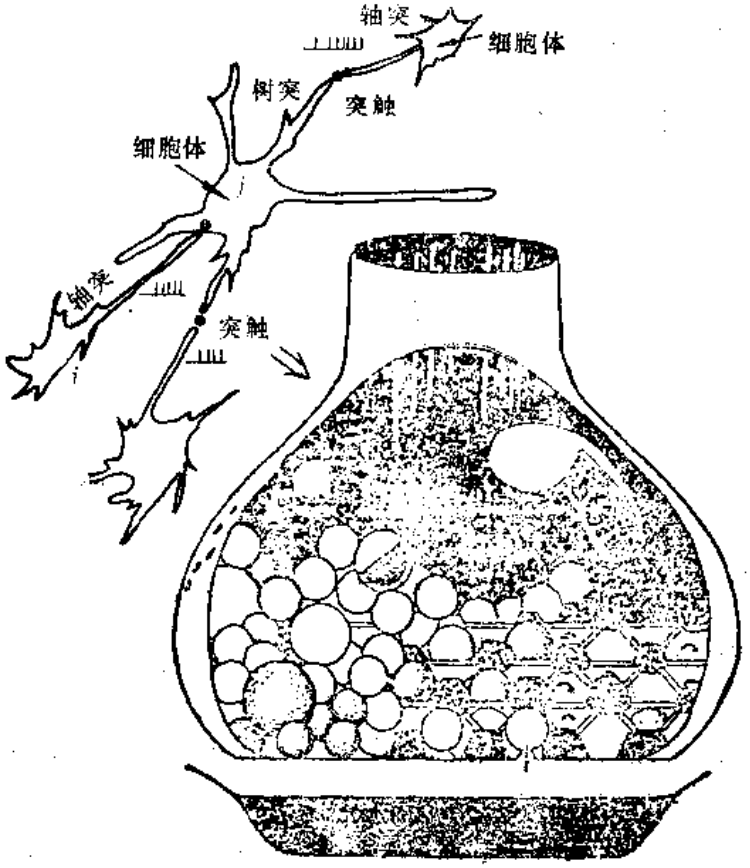


图 3-7　神经细胞中突触结构及位置

递质被释放，从而引起了突触后成分的去极化或超极化。反映在突触后电位变化有累加的作用，即同时既有从不同的其他神经细胞突触来的膜电位的累加，又有同一个突触在不同时间的神经冲动造成的膜电位的累加，累加到一定值 θ，使神经细胞兴奋，这个值即为阈值。同时发出一个脉冲，然后再继续进行累加，累加的波形如图 3-8 所示。如果输入脉冲串的频率越高，则产生的膜电位也大，其累加值随时间上升的速率也越快，超过阈值的时间也越早，这使输出脉冲的频率也加快，在神经细胞轴突上的输出是阈值 θ 以上的脉冲串，然后将这个脉冲再传递到其他神经细胞上。根据神经细胞这一个十分细致的信息传递过程，我们用一些数学模型来描述，如果我们考虑第 i 个神经细胞上的第 j 个突触，在 t_i 时刻接受到一个脉冲，所产生的突触后电位用 u_i 来表示，它可以用一个指数形式来近似：

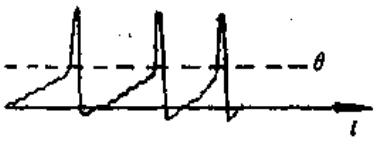


图 3-8　突触后电位累加波形

$$u_i(t)=u_{ij0}e^{-(t-t_i)/\tau} \qquad \forall t>t_i$$
$$u_i(t)=0 \qquad \forall t<t_i$$

u_{ij0} 为突触传递的系数，它与突触的性质有关(对兴奋性突触 $u_{ij0}>0$，对于抑制性突触 $u_{ij0}<0$)，与突触中囊泡内的递质有关，实质上它反映了第 j 个突触前的刺激能对第 i 个神经细胞产生多少贡献。τ 为一个时间常数。如果输入为一串脉冲，$l=1, 2, \cdots$，那么在突触 j 上的后电位可表示为

$$u_j=\sum_l u_i=\sum_l u_{ij0}e^{-(t-t_l)/\tau}$$
$$=\int_{-\infty}^{t} dt' u_{ij0}e^{-(t-t')/\tau}\sum_l \delta(t'-t_l) \tag{3-47}$$

式中 $\delta(t'-t_l)$ 满足

$$\delta(t'-t_l)=\begin{cases}\infty & t'=t_l\\ 0 & t'\neq t_l\end{cases} \tag{3-48}$$

根据 $\delta(\cdot)$ 函数的性质：任意函数与 δ 函数的积分就等于该函数本身，因此可得

$$\int_{-\infty}^{t} g(t-t')\delta(t'-t_l)dt'=g(t-t_l)$$

所以(3-47)式成立。

对于第 i 个神经细胞上存在很多与其他神经细胞相联的突触，其 u_{ij0} 是互不相同的，这样膜电位的总和为

$$x_i(t)=\sum_j u_j=\sum_j u_{ij0}\int_{-\infty}^{t} dt' e^{-(t-t')/\tau}\sum_l \delta(t'-t_l) \tag{3-49}$$

令 $f_j(t)$ 为第 j 个神经细胞输出的脉冲串为

$$f_j(t)=\sum_l \delta(t'-t_l) \qquad l=1,2,\cdots$$

把 $x_i(t)$ 作为第 i 个神经细胞的状态，它的数值直接反映了输出脉冲的发放频率，对 $x_i(t)$ 进行微分得到：

$$\frac{dx_i(t)}{dt}=\sum_j u_{ij0}\sum_l \delta(t'-t_l)+\sum_j u_{ij0}\frac{-1}{\tau}\int_{-\infty}^{t} dt' e^{-(t-t')/\tau}\sum_l \delta(t'-t_l)$$
$$=-\frac{1}{\tau}x_i(t)+\sum_j u_{ij0}f_j(t) \tag{3-50}$$

为了方便起见，让$f_j(t)$的量纲从脉冲频率改为电压量纲，而u_{j0}表示第j个神经细胞与第i个神经细胞之间的权，而状态$x_i(t)$与$f_i(t)$之间的关系是服从真实神经细胞中的单调上升关系，那么式(3-50)可以重写为

$$\begin{cases} \dfrac{\mathrm{d}x_i(t)}{\mathrm{d}t} = -\dfrac{1}{\tau}x_i(t) + \sum\limits_j w_{ji}V_j & (3\text{-}51a) \\ V_i = F[x_i(t)] & (3\text{-}51b) \end{cases}$$

$F(\cdot)$是一个单调上升的可微函数，其关系如图3-9所示。

如果存在外界的刺激，用电流I来表示，它将改变$x_i(t)$变化的速率，式(3-51a)可改为

$$\frac{\mathrm{d}x_i(t)}{\mathrm{d}t} = -\frac{1}{\tau}x_i(t) + \sum_j w_{ji}V_j + I_i \qquad (3\text{-}51c)$$

在第i个神经细胞的数学模型中，我们看到输入的累加值为

$$s_i = \sum_j w_{ji}V_j + I_i$$

它与(3-1)式相一致。

状态值x_i与累加值的关系用一个状态方程来表示：

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = -\frac{x_i}{\tau} + s_i$$

它与(3-2)式相一致。

输出与状态之间的关系为一个单调上升的有限函数：

$$V_i = F(x_i)$$

它与(3-3)式相一致

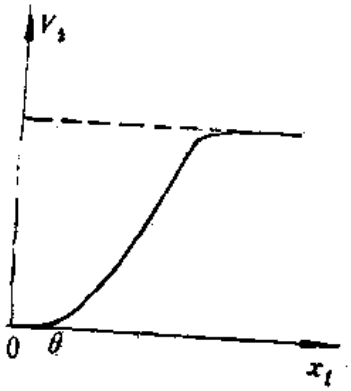


图3-9 V_i与x_i之间关系

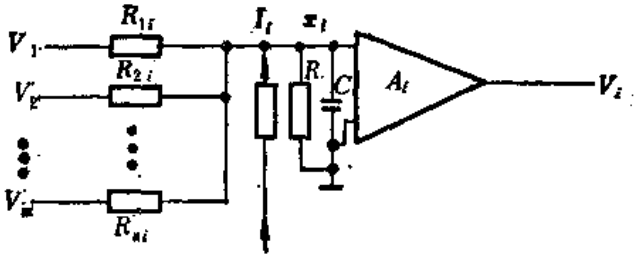


图3-10 运算放大器模拟神经的电路模型

从这几个方程的形式看，如果输入输出信息传递从脉冲变为电压，则这种神经细胞可以用一个简单的运算放大器来仿真形成人工神经元，在图3-10中，第i个运算放大器的输入为状态x_i，它与输出V_i之间的关系满足如图3-9所示的单调上升函数，其电源电压为输出V_i的最高值，利用克希霍夫定律可得

$$\begin{cases} C\dfrac{\mathrm{d}x_i}{\mathrm{d}t}=-\dfrac{x_i}{R}+\sum\limits_{j=1}^{n}\dfrac{1}{R_{ji}}(V_j-x_i)+I_i \\ V_i=F(x_i) \end{cases} \tag{3-52}$$

比较(3-52)式和(3-51c)式，则(3-52)式可改写为

$$\frac{\mathrm{d}x_i}{\mathrm{d}t}=-\left(\frac{1}{RC}+\sum_{j=1}^{n}\frac{1}{R_{ji}C}\right)x_i+\frac{1}{C}\sum_{j=1}^{n}\frac{1}{R_{ji}}V_j+I_i/C$$

对照(3-51)式，可得

$$\frac{1}{\tau}=\frac{1}{RC}+\sum_{j=1}^{n}\frac{1}{R_{ji}C};\quad w_{ji}=\frac{1}{CR_{ji}}$$

对于具有 n 个互相联接的人工神经元组成的网络，每个神经元都满足(3-52)方程，可将(3-52)式写成矩阵形式，当 $C=1$ 时为

$$\begin{bmatrix}\dot{x}_1(t)\\ \dot{x}_2(t)\\ \vdots\\ \dot{x}_n(t)\end{bmatrix}=-\frac{1}{\tau}\begin{bmatrix}x_1(t)\\ x_2(t)\\ \vdots\\ x_n(t)\end{bmatrix}+\begin{bmatrix}w_{11}&w_{12}&\cdots&w_{1n}\\ w_{21}&w_{22}&\cdots&w_{2n}\\ \vdots&\vdots&\cdots&\vdots\\ w_{n1}&w_{n2}&\cdots&w_{nn}\end{bmatrix}\begin{bmatrix}V_1(t)\\ V_2(t)\\ \vdots\\ V_n(t)\end{bmatrix}+\begin{bmatrix}I_1\\ I_2\\ \vdots\\ I_n\end{bmatrix} \tag{3-53a}$$

$$\begin{bmatrix}V_1(t)\\ V_2(t)\\ \vdots\\ V_n(t)\end{bmatrix}=\begin{bmatrix}F(x_1)\\ F(x_2)\\ \vdots\\ F(x_n)\end{bmatrix} \tag{3-53b}$$

这是一个 n 维的非线性微分方程，它与离散系统比较，当 $\dot{x}_i(t)=0$ 时，$i=1, 2, \cdots, n$，(3-53a)式可写为

$$\begin{bmatrix}x_1(t)\\ x_2(t)\\ \vdots\\ x_n(t)\end{bmatrix}=\tau\begin{bmatrix}w_{11}&w_{12}&\cdots&w_{1n}\\ w_{21}&w_{22}&\cdots&w_{2n}\\ \vdots&\vdots&\ddots&\vdots\\ w_{n1}&w_{n2}&\cdots&w_{nn}\end{bmatrix}\begin{bmatrix}V_1(t)\\ V_2(t)\\ \vdots\\ V_n(t)\end{bmatrix}+\tau\begin{bmatrix}I_1\\ I_2\\ \vdots\\ I_n\end{bmatrix} \tag{3-54}$$

把矢量 $\boldsymbol{I}=[I_1, I_2, \cdots, I_n]^T$ 看作阈值，那么上式就与 DHNN 的形式相同；同时，如果代表 $F(\cdot)$ 函数的运算放大器的放大倍数足够大，那么就变为一个二值的硬函数，可以说 DHNN 是 CHNN 的一个特例。

二、CHNN 方程的解及稳定性讨论

方程(3-53a、b)是一个非线性方程，方程的状态随时间的变化为零的那些解称为方程的平衡点，我们定义为 $\dot{\boldsymbol{x}}(t)=0$，可以通过解(3-54)和(3-53b)得到，但是平衡点并不是系统的最后归宿，因为状态变量对时间的微分为零，并不说明系统能到达这些点，平衡点的性质可能有如下几种情况：

设 $\boldsymbol{x}_0$ 是一个系统的平衡点，以 $\boldsymbol{x}_0$ 为圆心，r 为半径在状态空间划出一个超球体区域 $s(\boldsymbol{x}_0, r)$，则有：

(1) 对于任意给的 $\varepsilon>0$, ($\varepsilon<R_0$) 存在一个区域 $\delta(0<\delta<\varepsilon)$，如图 3-11 所示，使得当 $x(0)$ 位于 $s(x_0,\delta)$ 之内，对一切时刻 $t>0$，有 $x(t)$ 位于 $s(x_0,\varepsilon)$ 内部，称这个平衡点 x_0 是李亚普诺夫(Lyapunov)的稳定。

(2) 如果条件(1)是满足的，并且其状态 $x(t)$ 随着时间 t 的增加趋向于 x_0，则称为按李亚普诺夫的渐近稳定。

在图 3-11 中表示出了这种渐近稳定点。

(3) 临界稳定，我们把不是渐近稳定的稳定类型称为临界稳定。

(4) 状态最终跑出了 $s(x_0,\varepsilon)$ 以外，此平衡点是不稳定的。

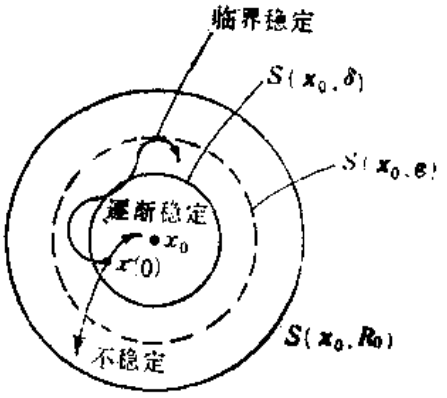


图 3-11　不同稳定意义的平衡点

在平衡点周围的情况可用图 3-11 来说明，从 $x(0)$ 出发的三条状态轨迹分别表示为不稳定、渐近稳定和临界稳定。在状态平衡点周围小范围内我们可以近似用线性方程的情况来讨论，因此我们首先把(5-53a、b)中的非线性函数 $F(\cdot)$ 用一个线性关系来代替，假设：

$$V_i=x_i \qquad i=1,2,\cdots,n$$

则方程(3-53a)就是一个线性状态方程，可写为标准形式：

$$\dot{x}=Ax+\mathrm{I} \tag{3-55}$$

式中

$$\dot{x}=[\dot{x}_1,\ \dot{x}_2,\ \cdots,\ \dot{x}_n]^T \qquad \in R^n;$$

$$x=[x_1,\ x_2,\ \cdots,\ x_n]^T \qquad \in R^n;$$

$$\mathrm{I}=[I_1,\ I_2,\ \cdots,\ I_n]^T \qquad \in R^n;$$

$$A=\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nn} \end{bmatrix} \in R^{n\times n};$$

现在用线性方程的方法来判断平衡点的性质。式(3-55)的特征方程为：$|A-\lambda U|=0$，U 为单位矩阵，可解出其特征值 $\lambda_1,\lambda_2,\cdots,\lambda_n$，而方程(3-55)的解为

$$x=e^{Ax}x(0)+\int_0^t e^{A(t-\tau)}\mathrm{I}\,d\tau \tag{3-56}$$

因此 $\lambda_1,\lambda_2,\cdots$ 的情况不同，其状态矢量在状态空间中的轨迹是不同的，因而影响了解的性质。

(i) 当 $\lambda_1,\lambda_2,\cdots,\lambda_n$ 的实部小于零，(3-56)式中 x 矢量是随时间指数下降，最后到达一个稳定值，方程(3-55)是收敛的，系统也是稳定的。

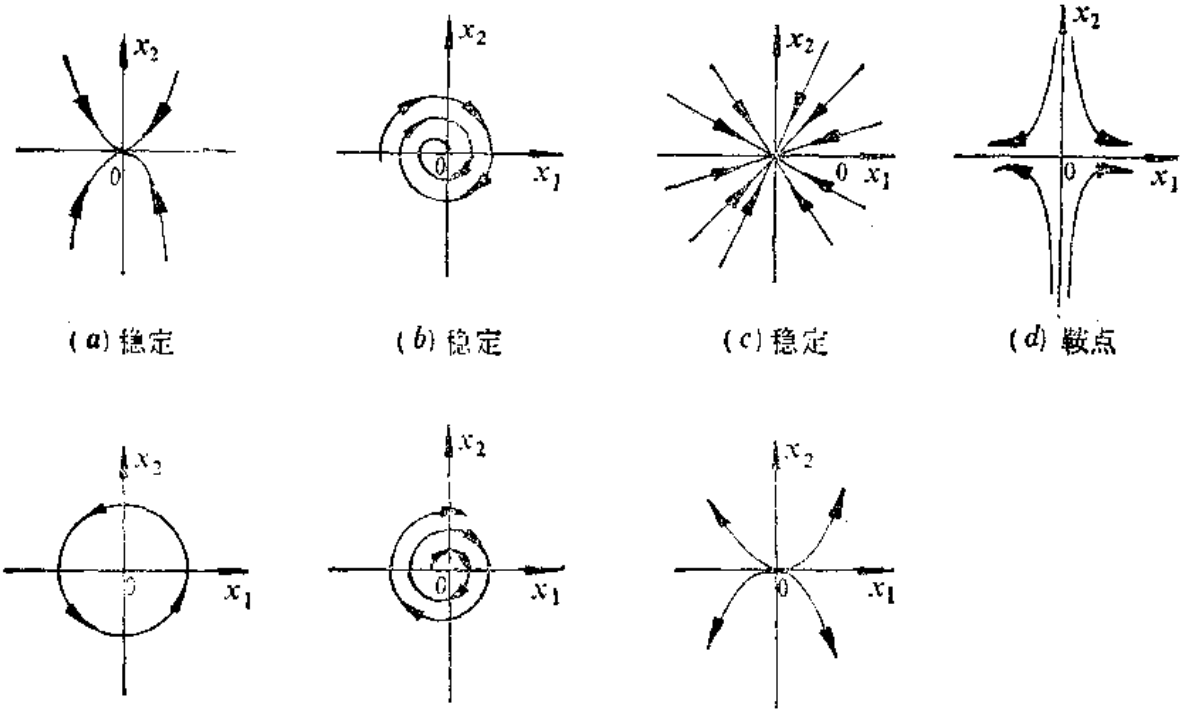
(ii) 当 $\lambda_1,\lambda_2,\cdots,\lambda_n$ 中存在异号实根，则系统出现了鞍点，即状态随时间变化的轨迹从某些方向是向着平衡点靠近，在另一些方向上是远离平衡点，因而系统不能稳定到一个稳定点。

(iii) 当 $\lambda_1,\lambda_2,\cdots,\lambda_n$ 中有些特征值有零实部，且存在非零虚部，则系统在状态空间上出现极限环。

(iv) 当 $\lambda_1,\lambda_2,\cdots,\lambda_n$ 的实部大于零，则系统发散。

在二维情况下，不同的特征根使状态对应在状态空间的轨迹如图 3-12 所示，平衡点为 $x_1=0,x_2=0$。

图 3-12 中 a、c 都是所有特征值 λ_1,λ_2 为实数，且小于零的情况。因而能稳定收敛到平衡点。图 3-12b 是 λ_1,λ_2 的实部小于零，而虚部不等于零，在状态空间上的轨迹是螺旋形地到达平衡点。图 3-12d 为鞍点，它对应的特征值为一个大于零，一个小于零的实数。图 3-12e 是 λ_1,λ_2 是纯虚数，因而出现了极限环。图 3-12f、g 则是不稳定的情况，它们的特征值都有大于零的实部，对应的系统随时间发散。



(a) 稳定　(b) 稳定　(c) 稳定　(d) 鞍点

(e) 极限环　(f) 不稳定　(g) 不稳定

图 3-12　不同特征值时 x 在状态空间的轨迹

在非线性的情况下，$V_i=F(x_i)\quad i=1,2,\cdots,n$，此时可以把状态方程 (3-53$a$、$b$) 写成下列形式：

$$\dot{x}=-\frac{1}{\tau}x+WF(x)+I=g(x,t) \tag{3-57}$$

其中 $\dot{x},x\in R^n;W\in R^{n\times n};F(x)=[F(x_1),F(x_2),\cdots,F(x_n)]^T$

$g(x,t)$是一个非线性函数，也可以写成：

$$g(x,t)=[g_1(x,t),g_2(x,t),\cdots,g_n(x,t)]^T$$

这里

$$g_i=(x,t)=-\frac{1}{\tau}x_i+\sum_{j=1}^{n}w_{ji}F(x_j)+I_i$$

我们可以让 $g(x,t)=0$，解出 x 的值，但由于非线性的存在，平衡点的数目可能很多，如果把 $F(x_j)$ 化为 l 段线性的话，那么对每一个 $F(x_j)$ 可能有 l 个线性区，系统可能组合出 l^n 个方程，每个方程有不同的平衡点解，平衡点的性质又各不相同，CHNN 网络主要关心的是那些渐近稳定平衡点，因为这些平衡点是系统的最后解，如果能设计出 W、I，使系统最后

解与要求的解一致，那么系统就可以完成联想或优化等功能。(3-57)式的平衡点的性质可以用在平衡点附近的一阶微分来近似，设系统平衡点为 x^e，如果 $g(x,i)$ 在平衡点附近连续可微，它的多阶导数存在，那么对于矢量 $x=(x_1,x_2,x_3,\cdots,x_n)^T$ 中的任一个分量 x_i，$i=1,2,\cdots,n$ 满足：

$$\dot{x}_i=g_i(x^e)+\sum_{j=1}^{n}\frac{\partial g_i}{\partial x_j}(x_j-x_j^e)+O_n \quad i=1,2,\cdots,n\cdots$$

若忽略了 O_n 高次微分项，则上式可写为

$$g_i(x^e)=0 \qquad \dot{x}_i=\sum_{j=1}^{n}\frac{\partial g_i}{\partial x_j}\Big|_{x_j=x_j^e}(x_j-x_j^e) \tag{3-58}$$

$\left.\frac{\partial g_i}{\partial x_j}\right|_{x_j=x_j^e}$ 为雅可比(Jacobian)行列式的一个元，这样非线性方程就化为线性方程，其解的性质就可以用线性方程的情况来讨论。下面举一个例子来说明平衡点解的情况。

[例 3-3] 图 3-13 是由两个运算放大器组成的两个人工神经元的互联电路，它们的状态方程如下：

$$\left.\begin{aligned} C\frac{dx_1}{dt}&=\frac{1}{R}(V_2-x_1)-\frac{x_1}{r}=g_1(x_1,x_2)\\ C\frac{dx_2}{dt}&=\frac{1}{R}(V_1-x_2)-\frac{x_2}{r}=g_2(x_1,x_2)\end{aligned}\right\} \tag{3-59}$$

其中

$$V_1=\mathrm{th}(\alpha x_1)$$
$$V_2=\mathrm{th}(\alpha x_2)$$

每个神经元的输入与输出关系为一个单调上升的函数。当 $x_1\to+\infty$，$x_2\to+\infty$，则 $V_1\to1$，$V_2\to1$；当 $x_1\to-\infty$，$x_2\to-\infty$，则 $V_1\to-1$，$V_2\to-1$；当 α 很小时，可以近似为 $V_i=\beta_i x_i$，$\beta_i=g_i'(x_i)x_i=x_i^e$，$\beta_i$ 为运算放大器的增益。如果我们考虑在 x^e 附近的情况下，状态方程(3-59)式可以用一个雅可比行列式展开，这个雅可比行列式为

$$\left|\begin{matrix}\frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2}\\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2}\end{matrix}\right|_{x=x^e}$$

在 x^e 附近展开，其中 $\beta_i=g_i'(x^e)$，利用(3-59)式，状态方程变为

$$C\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\frac{1}{R}\begin{bmatrix}-1-\frac{R}{r} & \beta_1\\ \beta_2 & -1-\frac{R}{r}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} \tag{3-60}$$

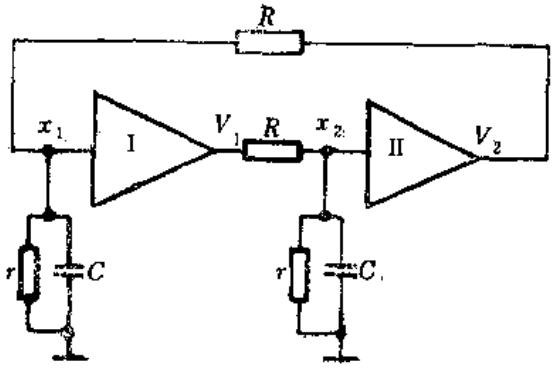


图 3-13 两个神经的互联电路

因为每个神经元的输入输出关系是相同的，因而 $\beta_1=\beta_2=\beta$，其平衡点可解，在 $\dot{x}_1=\dot{x}_2=0$ 时，得 $x_1=x_2=0$。若 $RC=1$，$r=\infty$，可求出方程(3-60)的特征方程为

$$(\lambda+1)^2-\beta^2=0;\lambda=\pm\beta-1$$

根据 β 的情况来讨论在 α 很小时解的性质。若 $\beta>1$，则 $\lambda_1>0, \lambda_2<0$，为一个鞍点，系统不能稳定在平衡点 $(0,0)$ 上，在状态空间中，如图 3-14a 所示。在 $x_1>0, x_2>0$ 的第一象限中，$\dot{x}_1>0, \dot{x}_2>0$；在 $x_1>0$，$x_2<0$ 的第四象限中，$\dot{x}_1<0, \dot{x}_2<0$；在 $x_1<0, x_2>0$ 的第二象限中，$\dot{x}_1>0, \dot{x}_2<0$；同理，在第三象限内 $\dot{x}_1<0, \dot{x}_2>0$；只有在 $x_1=-x_2$ 的直线上才会收敛到平衡点 $x_1=x_2=0$，其他都偏离平衡点。

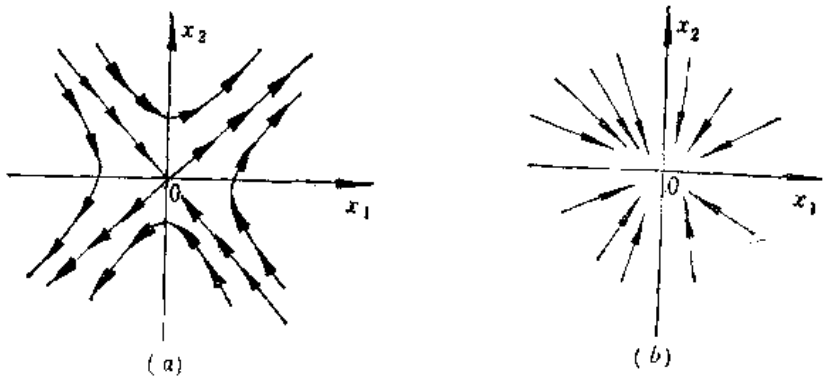


图 3-14 β 不同时，两个神经元组成网络的平衡点
(a) $\beta>1$　　(b) $\beta<1$

当 $\beta<1$ 时，$\lambda_1<0, \lambda_2<0$，系统能收敛到平衡点 $x_1=x_2=0$，其状态空间上的轨迹如图 3-14b 所示。

进一步讨论 $\beta>1$ 的情况。当 $\beta>1$，在 $x_1=0, x_2=0$ 的平衡点不稳定，在远离平衡点 x^* 的区域是否存在稳定点呢？将方程(3-59)改写为下式：

$$\left.\begin{aligned}\frac{\mathrm{d}x_1}{\mathrm{d}t}&=-x_1+\alpha_1 f(x_2)\\ \frac{\mathrm{d}x_2}{\mathrm{d}t}&=-x_2+\alpha_2 f(x_1)\end{aligned}\right\} \tag{3-61a}$$

其中 $RC=1; r=\infty; f(x_i)=\mathrm{th}(\alpha x_i)$

令

$$\frac{\mathrm{d}x_1}{\mathrm{d}t}=\frac{\mathrm{d}x_2}{\mathrm{d}t}=0$$

得

$$\left.\begin{aligned}x_1&=\alpha_1 f(x_2)\\ x_2&=\alpha_2 f(x_1)\end{aligned}\right\} \tag{3-61b}$$

若 $\alpha_1=\alpha_2=1$，则式(3-61b)x_1 与 x_2 之间的关系在状态空间中用两条曲线表示，其曲线的交点就是平衡点，而平衡点的稳定性可以进一步地分析，图 3-15 中的 A、B、C，是由(3-61b)两式相交得到的三个平衡点。A 平衡点为 $x_1=0, x_2=0$，从上面的分析可以看出，因为 $\beta>1$，而在 A 点是鞍点，不稳定。B 和 C 平衡点是落在 $f(\cdot)$ 比较平坦的部分，因此 $\beta>1$，可以分析 B 和 C 是稳定点，AP 是经过平衡点 A 的一条直线，当初态落在直线的右侧，$x(t)$ 收敛到 B 点，初态落在直线的左侧，$x(t)$稳定收敛到 C 点，B、C 是两个渐近稳定点，它们的吸引域大小相同，并包括了整个状态空间。

[例 3-4]　将[例 3-3]中的第Ⅱ个运算放大器的输出取反，如图 3-16 所示，并且

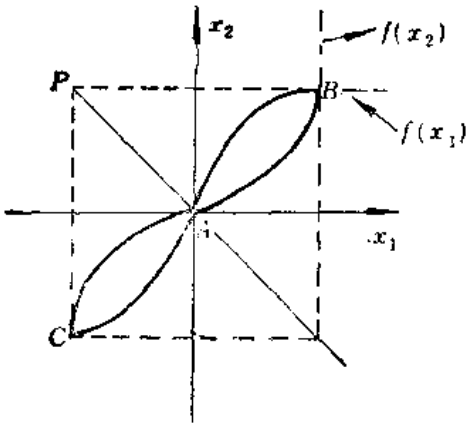


图 3-15　$\beta>1$时网络的平衡点

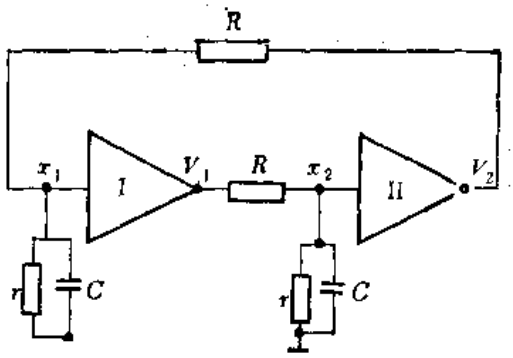


图 3-16　[例 3-4]中两个神经元组成的电路

$r\to\infty$, $RC\fallingdotseq1$, 那么方程(3-59)中的 V_2 改为 $V_2=-\mathrm{th}(\alpha x_2)=-\beta x_2$, 对 (3-59) 式进行雅可比行列式展开, 可得

$$C\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}-\frac{1}{R}-\frac{1}{r} & -\frac{\beta}{R}\\ \frac{\beta}{R} & -\frac{1}{R}-\frac{1}{r}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}\tag{3-62}$$

其特征方程为

$$\lambda^2+2\frac{R+r}{RrC}\lambda+\left(\frac{R+r}{RrC}\right)^2+\frac{\beta^2}{R^2C^2}=0$$

解上述方程得到

$$\lambda_{1,2}=-\frac{R+r}{RrC}\pm\sqrt{\left(\frac{R+r}{RrC}\right)^2-\left(\frac{R+r}{Rrc}\right)^2-\frac{\beta^2}{R^2C^2}}$$

$$=-\frac{1}{C}\left(\frac{1}{R}+\frac{1}{r}\right)\pm\mathrm{i}\frac{\beta}{RC}$$

式(3-62)的特征值有负实部和非零虚部，因而系统的状态最终能平衡到一个稳定点。这个系统状态的轨迹是一个螺旋递减的曲线，随着时间的增加，越来越靠近平衡点，直至达到平衡点为止($t\to\infty$)，系统的平衡点是 $x_1=x_2=0$。

上面我们是用拟线性的雅可比行列式来解非线性的问题，在一个范围内，系统的稳定性可以用线性方程中的特征值的情况来分析，其平衡点在某一个小的线性区域中其性质也就比较清楚了。对于多维的状态方程，当 $n>3$，其特征方程是一个高阶的多项式，很难解出 λ 的值，如下式：

$$a_0\lambda^n+a_1\lambda^{n-1}+a_2\lambda^{n-2}+\cdots+a_n\lambda+a_n=0$$

可以采用胡尔威次判据来判断具有上述特征方程系统的稳定性，渐近稳定的充要条件为多阶主子行列式大于零，即

$$\Delta_1=a_1>0 \qquad \Delta_2=\begin{vmatrix} a_1 & a_0 \\ a_3 & a_2 \end{vmatrix}>0$$

$$\Delta_3=\begin{vmatrix} a_1 & a_0 & 0 \\ a_3 & a_2 & a_1 \\ a_5 & a_4 & a_3 \end{vmatrix}>0 \qquad \cdots \Delta_n=\begin{vmatrix} a_1 & a_0 & 0 & 0 & \cdots & 0 \\ a_3 & a_2 & a_1 & a_0 & \cdots & 0 \\ a_5 & a_4 & a_3 & a_2 & \cdots & 0 \\ a_7 & a_6 & a_5 & a_4 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & a_n \end{vmatrix}>0 \tag{3-63}$$

三、李雅普诺夫稳定性定理

用拟线性方法来讨论系统的稳定性，对于有些问题并不能得到完全正确的分析，更加一般的方法是用李雅普诺夫定理进行，对于下列的状态方程组：

$$\frac{\mathrm{d}x_i}{\mathrm{d}t}=f_i(x_1,x_2,\cdots,x_n,t) \qquad i=1,2,\cdots,n \tag{3-64}$$

(1) 如果可以找到一个在原点邻域内满足下列条件的可微函数 $V(x_1,x_2,\cdots,x_n)$ 该函数称为李雅普诺夫函数，它满足：

$$V(x_1,x_2,\cdots,x_n)\geqslant 0 \quad (\text{或}\leqslant 0)$$

在 $x_i=0(i=1,2,\cdots,n,)$ 时，$V=0$

(2) 当 $t>t_0$ 时，V 的积分曲线的全导数

$$\frac{\mathrm{d}V}{\mathrm{d}t}=\sum_{i=1}^{n}\frac{\partial V}{\partial x_i}\frac{\mathrm{d}x_i}{\mathrm{d}t}=\sum_{i=1}^{n}\frac{\partial V}{\partial x_i}f_i(x_1,x_2,\cdots,x_n,t)\leqslant 0 \quad (\text{或}\geqslant 0)$$

则方程(3-64)在平衡点 $x_i=0(\forall i=1,2,\cdots,n)$ 是稳定的。

又因为在 $x_i=0(i=1,2,\cdots,n)$ 周围的邻域内都满足：

$$\frac{\mathrm{d}V}{\mathrm{d}t}<0 \quad (\text{或}>0)$$

所以 $x_i=0(i=1,2,\cdots,n)$ 为渐近稳定点。

李雅普诺夫稳定定理可解更一般的问题，例如下列方程：

$$\begin{aligned} \dot{x}_1&=-4x_2-x_1^3 \\ \dot{x}_2&=3x_1-x_2^3 \end{aligned} \tag{3-65}$$

在 $x_1=x_2=0$ 的平衡点附近的雅可比行列式为

$$J=\begin{vmatrix} 0 & -4 \\ 3 & 0 \end{vmatrix}$$

它的特征方程：

$$\lambda^2+12=0$$

则有

$$\lambda_{1,2}=\pm\sqrt{12}\,i$$

可能判为在平衡点 $x_1=x_2=0$ 上不稳定，出现了极限环，但是选用李雅普诺夫函数 $V(x_1x_2)=3x_1^2+4x_2^2$ 满足：

(1) $V(0,0)=0$

(2) $V(x_1,x_2)>0$

(3) $\dfrac{dV}{dt}=\dfrac{\partial V}{\partial x_1}\dfrac{dx_1}{dt}+\dfrac{\partial V}{\partial x_2}\dfrac{dx_2}{dt}=-6x_1^4-8x_2^4\leqslant 0$

式(3-65)的平衡点 $x_1=0$, $x_2=0$ 是属于李雅普诺夫稳定，因而系统是稳定的，因为在 $x_1=0,x_2=0$ 周围的任意大的区域内都可以满足，因此系统是渐近稳定的。

严格按照李雅普诺夫函数考虑 CHNN 模型，要求系统的原点为平衡点，将式(3-53a、b)改写为下式：

$$\dot{\boldsymbol{x}}=-\frac{1}{\tau}\boldsymbol{x}+\boldsymbol{W}\boldsymbol{F}(\boldsymbol{x})+\boldsymbol{I}$$

$$=-\boldsymbol{A}\boldsymbol{x}+\boldsymbol{W}\boldsymbol{F}(\boldsymbol{x})+\boldsymbol{I}$$

式中 $\boldsymbol{x}\in R^n$; $x_i(i=1,2,\cdots,n)$是第 i 个神经元的状态变量; $\boldsymbol{F}(\boldsymbol{x})=[F(x_1),F(x_2),\cdots,F(x_n)]^T$; $F(x_i)$是连续可微的; $\boldsymbol{I}=(I_1,I_2,\cdots,I_n)^T$是网络的输入向量; $\boldsymbol{W}$ 为 $n\times n$ 实际数矩阵; $\boldsymbol{A}$为一个对角阵，对角元为$\dfrac{1}{\tau}$；与实际电路相比，τ 是一个与电阻、电容有关的量，$\tau>0$。

把 (3-53a、b) 式写成分量形式，w_{ji} 为第 j 个神经元与第 i 个神经元之间的联接权，CHNN 模型状态分量的微分方程为

$$c_i\frac{dx_i}{dt}=-\frac{x_i}{R_i}+\sum_{j=1}^{n}w_{ji}F(x_j)+I_i$$

令$c_i=1$，上式可写为

$$\dot{x}_i=-a_{ii}x_i+\sum_{j=1}^{n}w_{ji}F(x_j)+I_i \qquad (3\text{-}66a)$$

$$i=1,2,\cdots,n$$

设 $\boldsymbol{x}^e=(x_1^e,x_2^e,\cdots,x_n^e)^T$为网络的平衡点，这个平衡点不一定是原点，代入式(3-66a)得

$$-a_{ii}x_i^e+\sum_{j=1}^{n}w_{ji}F(x_j^e)+I_i=0 \qquad (3\text{-}66b)$$

令 $u_i=x_i-x_i^e$,

$$h(x_i^e+u_j)=F(x_j^e+u_j)-F(x_j^e) \qquad (3\text{-}66c)$$

令 $\dot{u}_i=(x_i-x_i^e)'=\dot{x}_i$

$$=-a_{ii}(u_i+x_i^e)+\sum_{j=1}^{n}w_{ij}F(x_j^e+u_j)+I_i$$

再利用式(3-66a)、(3-66b)、(3-66c)，可得

$$\dot{u}_i=\dot{x}_i=-a_{ii}u_i+\sum_{j=1}^{n}w_{ji}h(x_j^e+u_j)=g_i(\boldsymbol{u}) \qquad (3\text{-}66d)$$

令 $\boldsymbol{u}\in R^n$ $\boldsymbol{u}=(u_1,u_2,\cdots,u_n)^T$

式(3-66d)可写成

$$\begin{cases}\dot{u}_i=g_i(\boldsymbol{u}) & i=1,2,\cdots,n\\ g_i(0)=0\end{cases} \qquad (3\text{-}67)$$

当 $\boldsymbol{u}=0$，表示 $x_i=x_i^e$, $i=1,2,\cdots,n$，这意味着式(3-67)是平衡点在原点的方程。对于式(3-67)，可定义如下的李雅普诺夫函数：

$$V(\boldsymbol{u})=\boldsymbol{g}^T(\boldsymbol{u})\boldsymbol{B}\boldsymbol{g}(\boldsymbol{u}) \qquad (3\text{-}68)$$

$\boldsymbol{g}(u)=[g_1(\boldsymbol{u}),g_2(\boldsymbol{u}),\cdots,g_n(\boldsymbol{u})]^T$，如果 $V(\boldsymbol{u})>0$，且 $\dot{V}(\boldsymbol{u})<0$，则满足李雅普诺夫稳定性定理。因为方程(3-67)已移到了原点，如满足 $V(\boldsymbol{u})>0$，只要 $\boldsymbol{B}$ 为正定，就能满足；要达到

$\dot{V}(u)<0$，必须对(3-68)式求全微分，如果求微分用近似线性的方法进行，则定义一个矩阵 T（B 为单位矩阵）为

$$T=g^T(u)[J^T(u)+J(u)]g(u)$$

那么 T 的负定就能保证 $\dot{V}(u)<0$，$J^T(u)$ 是 $J(u)$ 的转置矩阵，$J(u)$ 是函数矢量 $g(u)$ 的雅可比矩阵：

$$J(u)=\begin{bmatrix} \frac{\partial g_1(u)}{\partial u_1} & \frac{\partial g_1(u)}{\partial u_2} & \cdots & \frac{\partial g_1(u)}{\partial u_n} \\ \frac{\partial g_2(u)}{\partial u_1} & \frac{\partial g_2(u)}{\partial u_2} & \cdots & \frac{\partial g_2(u)}{\partial u_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_n(u)}{\partial u_1} & \frac{\partial g_n(u)}{\partial u_2} & \cdots & \frac{\partial g_n(u)}{\partial u_n} \end{bmatrix} \tag{3-69}$$

要使 T 为负定，则有 $[J^T(u)+J(u)]$ 负定。因此对于 (3-67) 式的稳定性讨论就变为使 $(J^T(u)+J(u))$ 矩阵的负定问题。若一个矩阵为负定，则要求其特征值为负，对于任一个矩阵 T，如果满足 $|T_{ii}|>\sum_{\substack{j=1\\ j\neq i}}^{n}|T_{ij}|$，$i=1,2,\cdots,n$，则 T 矩阵为占优矩阵，在占优矩阵中，T 的特征值实部正、负的个数与 T_{ii} 的正负个数相同。

要使 $J^T(u)+J(u)$ 为负定，要保证它的对角线元素为负的占优矩阵，则对于 (3-66a)、(3-66d) 式和 (3-67) 式要达到占优的负定矩阵，应要求：

$$-a_{ii}+w_{ii}\frac{\partial F(x_i^e+u_i)}{\partial u_i}<0 \tag{3-70a}$$

$$2\left|w_{ii}\frac{\partial F(u_i+x_i^e)}{\partial u_i}-a_{ii}\right| > \left|\sum_{i\neq j}\left(w_{ij}\frac{\partial F(u_j+x_j^e)}{\partial u_j}+w_{ji}\frac{\partial F(u_i+x_i^e)}{\partial u_i}\right)\right| \tag{3-70b}$$

上式又称为克拉索夫斯基定理。

我们给出一个利用克拉索夫斯基定理来讨论 n 个神经元组成的网络的稳定情况的例子。

[例 3-5] 用 n 维的 CHNN 网络来实现一个多端抑制器，其电路如图 3-17 所示，这是一种最简单的具有多维状态方程的系统。利用克希霍夫定理，它的状态方程和输入、输出的关系如下：

$$c\frac{dx_i}{dt}=-\frac{x_i}{r}+\sum_{j=1}^{n}w_{ji}(\overline{V}_j-x_i)+I_i \quad i=1,2,\cdots,n \tag{3-71a}$$

其中：$w_{ji}=+1 \quad \forall i.j$

$w_{ii}=0$

$$V_i=F(x_i)=\begin{cases} 0 & x_i<-\frac{1}{2\alpha}; & \text{区域 } A \\ +(\alpha x_i+0.5) & -\frac{1}{2\alpha}\leqslant x_i\leqslant\frac{1}{2\alpha}; & \text{区域 } B \\ +1 & x_i>\frac{1}{2\alpha}; & \text{区域 } C \end{cases} \tag{3-71b}$$

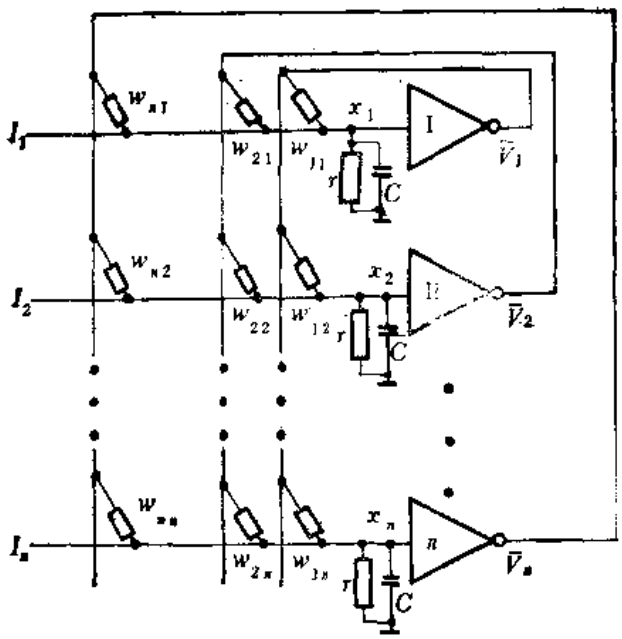


图 3-17　多个神经元网络的电路图

上式中 $\overline{V}_i$，引入了"一"为"非"符号，是为了解决 w_{ji} 的符号问题，在电路中的权是电阻，因而没有符号。这里 $\overline{V}_i=-V_i$。

首先考虑在 $I_i=0$ 的情况。因为 I_i 对方程(3-69)是没有影响的，而影响方程(3-66)的平衡点具体数值。代入(3-71a)方程中有关多端抑制器的条件：$w_{ji}=+1$，$w_{ii}=0$，$\forall i,j$，且设 $cr=1$，$c=n-1$，方程(3-71a)可写为

$$\dot{x}_i=-x_i+\frac{1}{n-1}\sum_{\substack{j=1\\ j\neq i}}^{n}\overline{V}_j \tag{3-71c}$$

写成矩阵形式为

$$\dot{\boldsymbol{x}}=-\boldsymbol{A}\boldsymbol{X}+\boldsymbol{W}\boldsymbol{V}$$

其中

$$\boldsymbol{A}=\begin{bmatrix}+1 & 0 & 0 & 0\\ 0 & +1 & 0 & 0\\ 0 & 0 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & +1\end{bmatrix}$$

$$\boldsymbol{W}=\begin{bmatrix}0 & -\frac{1}{n-1} & -\frac{1}{n-1} & \cdots & -\frac{1}{n-1}\\ -\frac{1}{n-1} & 0 & \cdots & & -\frac{1}{n-1}\\ \vdots & \vdots & \ddots & & \vdots\\ -\frac{1}{n-1} & -\frac{1}{n-1} & \cdots & & 0\end{bmatrix}$$

$\boldsymbol{A}$ 为对角阵。

考虑(3-71a)是工作在 $-\frac{1}{2a}\leqslant x_i\leqslant\frac{1}{2a}$，即 B 区域中，将(3-71b)代到(3-71c)式，可

以通过解方程 $\dot{x}_i=0$，得到

$$x_i^e=-\frac{1}{n-1}\sum_{j=1}^{n}(\alpha x_j+0.5)$$

令 $$x^e=(x_1^e,x_2^e,\cdots,x_n^e)^T$$

将平衡点代入(3-71c)式，让 $u_i=x_i-x_i^e$；$h(x_j^e+u_j)=F(x_j^e+u_j)-F(x_j^e)$，得

$$\dot{u}_i=-u_i+\sum_{j=1}^{n}-\frac{1}{n-1}h(x_j^e+u_j)$$

$$=-u_i+\sum_{\substack{j=1\\i\neq j}}^{n}-\frac{\alpha}{n-1}u_j$$

令 $\beta=\frac{\alpha}{n-1}$，则上述方程的雅可比矩阵为

$$J(u)=\begin{bmatrix}-1 & -\beta & -\beta & \cdots & -\beta\\ -\beta & -1 & -\beta & \cdots & -\beta\\ -\beta & -\beta & -1 & \cdots & -\beta\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ -\beta & -\beta & -\beta & \cdots & -1\end{bmatrix}$$

要使方程(3-71c)在区域 B 内稳定，必须要满足式(3-70a)和式(3-70b)，即

(i) $-a_{ii}+0<0$ 即要求：$a_{ii}>0$

(ii) $$2|a_{ii}|>\left|\sum_{i\neq j}\left[w_{ij}\frac{\partial F(u_j+x_j^e)}{\partial u_j}+w_{ji}\frac{\partial F(u_i+x_i^e)}{\partial u_i}\right]\right|$$

$$2>2\frac{\alpha(n-1)}{n-1}$$

得 $$\alpha<1。$$

由此可见，在 $\alpha<1$ 在的条件下，在(3-71b)式区域 B 内的平衡点是满足渐近稳定条件的。

对于本例的系统，我们希望系统的平衡点不是在区域 B 内，而是在区域 C 和 A 内，一个多稳态电路希望 n 个神经元中只有一个单元输出为 1，其他为 0，这一方面可以使系统在区域 B 不稳定，同时，再加上输入来满足，如果希望第 e 个神经元的输出为 1，而其他神经元的输出为零，在方程(3-71c)中加上输入 I_i，可得

$$\dot{x}_i=-x_i+\frac{1}{n-1}\sum_{\substack{j=1\\i\neq j}}^{n}V_j+\frac{I_i}{n-1}$$

要求：当 $j\neq e$，$V_j=0$；当 $j=e$，$V_j=1$。

在平衡时，对于第 e 个神经元，它的方程如下($V_j=0$)：

$$\dot{x}_e=-x_e+\frac{I_e}{n-1}=0$$

要满足其输出为 $V_e=1$，根据(3-71b)式，x_e 落在区域 C 上，则有

$$x_e=\frac{I_e}{n-1}>\frac{1}{2\alpha}$$

$$I_e>\frac{n-1}{2\alpha}=\frac{1}{2\beta}$$

对于其他神经元，$i\neq e$，$i=1,2,\cdots,n$ 其状态方程为

$$\dot{x}_i=-x_i-\frac{1}{n-1}+\frac{I_i}{n-1}$$

在 $\dot{x}_i=0$ 时，要使 x_i 落在区域 A，则有

$$x_i=-\frac{1}{n-1}+\frac{I_i}{n-1}<-\frac{1}{2\alpha}=-\frac{1}{2\beta(n-1)}$$

因此可得

$$I_i<\left(1-\frac{1}{2\beta}\right)$$

可以通过输入值来控制其输出的某一个为 1，某一个为 0，当然对 w_{ij} 的调整也可以调整其输出的平衡点的情况，w_{ij} 的设计我们还要详细讨论。

用李雅普诺夫方法来讨论一个非线性系统的稳定情况是一种直接的方法，比较适合于一个复杂的系统，而李雅普诺夫函数对一个系统来说不是唯一的，因此用李雅普诺夫来决策可以保证系统稳定，但找不到李雅普诺夫函数并不说明系统不稳定，李雅普诺夫函数只是一个充分条件而不是一个必要条件。用克拉索夫基方法还只是解决在平衡点附近区域的稳定性问题，它利用了李雅普诺夫函数的稳定方法，因此它比一般的雅可比方法使用面略广一些。在我们讨论的例子中，对于$[\boldsymbol{J}^T(\boldsymbol{u})+\boldsymbol{J}(\boldsymbol{u})]$矩阵负定性的判断还是比较麻烦的，尤其是在维数比较多的情况下，更是如此。采用占优矩阵来分析，其条件也过于苛刻，因为占优矩阵只有在对角元全小于零时，可保证负定，但不是占优矩阵也同样可以使矩阵负定，因此这是一种可以讨论、尚待改进的方法，并不完全能分析系统的全局情况，但在有些问题中可以采用。

四、Hopfield 的能量函数和稳定性判别

Hopfield 在 80 年代初提出了一个对单层反馈动态神经网络的稳定性判别函数，这个函数有确定的物理意义，是建立在能量基础上的，它的离散形式就是在自旋材料中的一种哈密顿能量，同时它又是李雅普诺夫函数的一种推广，它是广义的李雅普诺夫函数。

对于状态方程组其输出：

$$c_i\frac{\mathrm{d}x_i}{\mathrm{d}t}=-\frac{x_i}{R_i}+\sum_{j=1}^{n}w_{ij}V_j+I_i \quad i,j=1,2,\cdots n \tag{3-72a}$$

$$V_i=F(x_i)$$

它的能量函数定义为

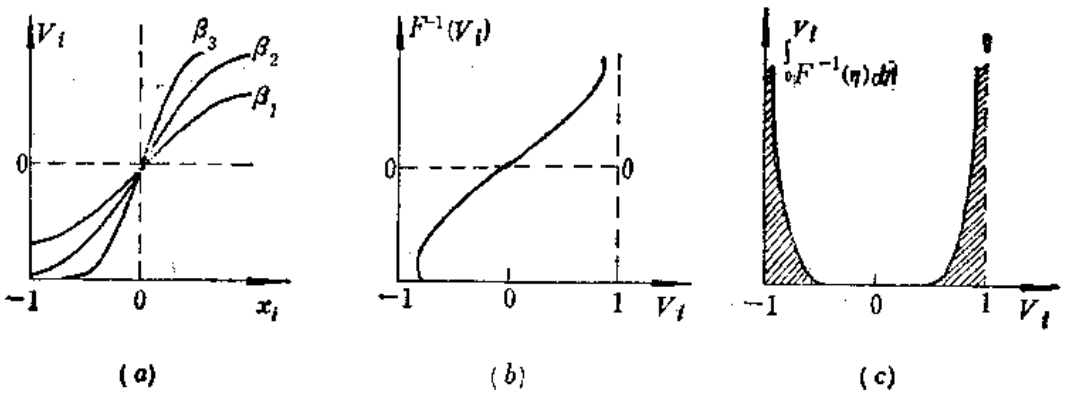
$$E=-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ij}V_iV_j-\sum_{i=1}^{n}V_iI_i+\sum_{i=1}^{n}\frac{1}{R_i}\int_0^{V_i}F^{-1}(\eta)\mathrm{d}\eta \tag{3-72b}$$

其中

$$\frac{1}{R_i}=\frac{1}{r}+\sum_j w_{ij}$$

状态方程完全和 CHNN 的电路模型紧密结合起来了，如前面图 3-17 的多维神经元模型，r 是神经元——运算放大器的输入电阻，w_{ij} 是第 i 个神经元输出与第 j 个神经元输入之间连接的导纳，c_i 是第 i 个运算放大器的输入总电容，x_i 为第 i 个运算放大器输入状态，V_i 为它的输出，而输入 I_i 为外加电流。输入状态 x_i 与输出 V_i 之间的关系是一个单调上升的函数关系，如图 3-18a 所示。β 表示了运算放大器的放大倍数，图 3-18a 表示在不同的放大倍数时，输入与输出之间的关系。在能量函数 E 的公式(3-72b)中的 $F^{-1}(V_i)$表示为函数 V_i的逆，它实际上是把图 3-18a 中的坐标颠倒后得到：$F^{-1}(V_i)=x_i$，它们的关系如图 3-18b 所示，在能量函数 E 中的积分项表示一种输入状态和输出值关系的能量项，其积分结果用图

3-18c 表示。把式(3-72b)与离散情况下 DHNN 的能量(3-18)式比较，唯一的区别是增加了第三项。为了便于同离散的 Hopfield 网络能量函数相比较，我们把神经元的输出 V_i 限制在 -1 和 $+1$ 之间，与离散的网络比较，V_i 是一个在 -1 和 $+1$ 之间的任意实数，而不是二值量，即定义为 $V_i\in[-1,1]$。当然也可以定义 V_i 为 0～1 之间的任意实数，其情况与 ±1 相同。当第三项积分为零，连续 Hopfield 的能量函数的形式与离散的哈密顿函数形式基本类似。



(a)　　(b)　　(c)

图 3-18　输入输出函数关系

考虑(3-72b)式，如果 E 是一个有界量，即 $|E|<E_{\max}$，那么 E 就可以像一个微分方程的李雅普诺夫函数一样，只不过这里的 E 可正可负，但有界。因此只要能量 E 随时间单调下降，就总可以达到能量的极小点，此时系统方程式(3-72a)也达到平衡，则能量极小点也是系统的稳定点。

如果满足：(i) $w_{ij}=w_{ji}$；(ii) 输入、输出关系是单调上升函数(如图 3-18a 所示)，那么对(3-72b)式求全导数可得

$$\begin{aligned}\frac{dE}{dt}&=-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ij}V_j\frac{dV_i}{dt}-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}w_{ji}V_j\frac{dV_i}{dt}\\&\quad-\sum_{i=1}^{n}I_i\frac{dV_i}{dt}+\sum_{i=1}^{n}\frac{1}{R_i}x_i\frac{dV_i}{dt}\\&=-\sum_{i=1}^{n}\frac{dV_i}{dt}\left(\sum_{j=1}^{n}w_{ij}V_j+I_i-\frac{x_i}{R_i}\right)\\&=-\sum_{i=1}^{n}c_i\frac{dV_i}{dt}\frac{dx_i}{dt}\\&=-\sum_{i=1}^{n}c_i\frac{dV_i}{dx_i}\left(\frac{dx_i}{dt}\right)^2\end{aligned}\tag{3-73}$$

由于输入输出关系是单调上升函数，所以有

$$\frac{dV_i}{dx_i}>0,$$

$$\frac{dE}{dt}\leqslant0$$

当 $\dfrac{dE}{dt}=0$ 时，则有

$\left(\dfrac{dx_i}{dt}\right)=0$，$i=1,2,\cdots,n$　表明能量极小点与 $\dfrac{dx_i}{dt}=0$ 的平衡点一致。

现在我们再讨论一下 E 函数是否有界，这里最重要的是 E 不能下降到负无限大，E 在负方向要有界，这样当 E 随时间减小必定会到达一个极限值，此极限值是 E 的极小值，而且也是系统的稳定解。

考虑 (3-72b) 式的第一项 $-\frac{1}{2}\sum_i\sum_j w_{ij}V_iV_j$，因为 $|V_i|\leqslant 1$，$|V_j|\leqslant 1$，因而不管 w_{ij} 组成的权矩阵是正定还是负定，它必是一个有界量。该式第二项 $-\sum_{i=1}^{n} I_iV_i$，I_i 是外加输入信号，V_i 有限，I_i 也是有限的，因此第二项也是有限的。考虑最后一项 $\sum_i \frac{1}{R_i}\int_0^{V_i} F^{-1}(\eta)\mathrm{d}\eta$，这个积分项中的 $F(\cdot)$，是神经网络的输入、输出关系函数，也是运算放大器的输入、输出的函数，用 β 代表运算放大器的增益，那么状态 x_i 与 V_i 的关系可用 $V_i=F(x_i,\beta)$ 来代替 $V_i=F(x_i)$，可得

$$x_i=\left(\frac{1}{\beta}\right)F^{-1}(V_i) \tag{3-74}$$

将(3-74)式代入(3-72b)式的第三项得

$$\frac{1}{\beta}\sum_{i=1}^{n}\frac{1}{R_i}\int_0^{V_i} F_i^{-1}(\eta)\mathrm{d}\eta \tag{3-75}$$

从图 3-18c 可以看到，(3-75)式的积分对一个 i 来说在 $V_i=0$ 时为零，而在其他情况下为正(不管 V_i 是正值还是负值)，同时当 $\beta\to\infty$ 时，这一项的作用很小而可以忽略，这时能量函数就由第一，第二两项的和决定，与离散时的情况相同。在图 3-18a 中可以看出 β 越大，其斜率也大，当 $\beta\to\infty$ 时，图 3-18a 中 $V_i(x_i)$ 是一个符号函数。如果 β 比较小，第三项为正，它的贡献主要是在靠近 $V_i\approx\pm 1$ 的边缘，因为边缘部分的能量大，能量的最小点是落在 $|V_i|<1$ 的那些区域上，为超立方体的内部。在 β 十分小的情况下，能量的极小值就到达 $V_i=0(i=1,2,\cdots,n)$ 的点上。β 较大而且有限时，可能存在多个能量极小点，我们用状态和能量的关系图来表示，对于二维状态下，能量用地形图来表示，其结果如图 3-19 示，这时稳

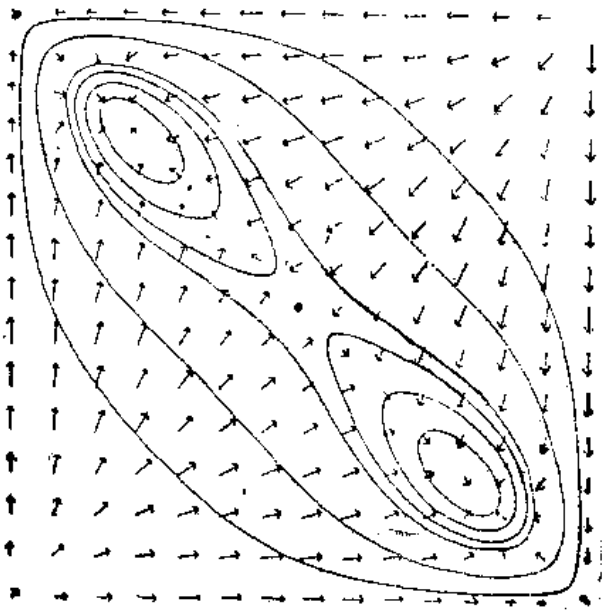


图 3-19　二维状态能量地形图

定点是在右下角和左上角上，而所有的边缘都是不稳定的，它们有比较大的能量，并逐渐向两个能量的极小值靠拢。该图是在 $w_{12}=w_{21}=1; w_{ii}=0, \beta=1.4$ 的情况下得到的，其能量函数的极小点偏负，十分靠近零。

Hopfield 选择的能量函数，也只是保证系统稳定和渐近稳定的充分条件，而不是必要条件，其能量函数也不是唯一的。为了证明 $\frac{\mathrm{d}E}{\mathrm{d}t}\leqslant 0$ 时，Hopfield 对设计权 w_{ij} 有一个对称性的要求，即 $w_{ij}=w_{ji}$ 和 $\frac{\mathrm{d}V_i}{\mathrm{d}x_i}>0$ 的要求，权的对称条件与实际神经网络的实验并不符合，实际神经细胞之间的联接权没有对称性要求。不少文章阐述了即使不满足联接权权矩阵对称的条件下，仍然可以达到系统稳定。我们这里证明：一个非对称的权矩阵如果能拆成一个对角正定阵与一个对称阵的乘积，那么系统仍能满足 E 是有限的、且 $\frac{\mathrm{d}E}{\mathrm{d}t}\leqslant 0$ 的条件。

证明 先将(3-72b)改写为矩阵形式，其能量函数为

$$E=-\frac{1}{2}\boldsymbol{V}^T\boldsymbol{W}\boldsymbol{V}-\boldsymbol{I}^T\boldsymbol{V}+\boldsymbol{R}^T\boldsymbol{S}(u) \tag{3-76}$$

上式中

$$\boldsymbol{V}(V_1,V_2,\cdots,V_n)^T \quad \boldsymbol{V}\in R^n$$

$$\boldsymbol{I}=(I_1,I_2,\cdots,I_n)^T \quad \boldsymbol{I}\in R^n$$

$$\boldsymbol{R}=\left(\frac{1}{R_1},\frac{1}{R_2},\cdots,\frac{1}{R_n}\right)^T \quad \boldsymbol{R}\in R^n$$

$$\boldsymbol{S}(\boldsymbol{V})=\left[\int_0^{V_1}F^{-1}(\eta)\mathrm{d}\eta,\int_0^{V_2}F^{-1}(\eta)\mathrm{d}\eta,\cdots\int_0^{V_n}F^{-1}(\eta)\mathrm{d}\eta\right]^T$$

$$\boldsymbol{W}\in R^{n\times n}$$

$\boldsymbol{W}$ 为一个对称阵。

对应的状态方程为：

$$\boldsymbol{C}\dot{\boldsymbol{X}}=-\boldsymbol{P}(\boldsymbol{V})\boldsymbol{R}+\boldsymbol{W}\boldsymbol{V}+\boldsymbol{I} \tag{3-77a}$$

式中 $\dot{\boldsymbol{X}}\in R^n$；$P(V)$为对角阵

即

$$\boldsymbol{P}(V)=\begin{bmatrix} F^{-1}(V_1) & 0 & \cdots & 0 \\ 0 & F^{-1}(V_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & F^{-1}(V_n) \end{bmatrix}$$

从神经元的输入、输出关系可得

$$\frac{\mathrm{d}V_i}{\mathrm{d}t}=\frac{\partial V_i}{\partial x_i}\frac{\mathrm{d}x_i}{\mathrm{d}t}$$

$$\frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}=\boldsymbol{Q}\frac{\mathrm{d}x}{\mathrm{d}t} \tag{3-77b}$$

其中：

$$\boldsymbol{Q}=\begin{bmatrix} F'(V_1) & 0 & 0 & \cdots & 0 \\ 0 & F'(V_2) & 0 & \cdots & 0 \\ 0 & 0 & F'(V_3) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & & F'(V_n) \end{bmatrix}$$

$$\boldsymbol{C}=\begin{bmatrix} c_1 & 0 & \cdots & 0 \\ 0 & c_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_n \end{bmatrix}$$

对式(3-76)进行全微分，则有

$$\frac{\mathrm{d}E}{\mathrm{d}t}=-\left(\frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}\right)^{T}[\boldsymbol{C}\boldsymbol{Q}^{-1}(V)]\frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}\leqslant 0 \tag{3-78}$$

为了保证(3-78)式成立，在(3-78)式中乘一个任意的对角正定矩阵 $\boldsymbol{D}$，其结果不会改变，因而可得

$$\frac{\mathrm{d}E}{\mathrm{d}t}=-\left(\frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}\right)^{T}(\boldsymbol{C}\boldsymbol{Q}^{-1}\boldsymbol{D})\ \frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}\leqslant 0 \tag{3-79}$$

展开上式且利用(3-77a)、(3-77b)式，可得

$$\begin{aligned}\frac{\mathrm{d}E}{\mathrm{d}t}&=-\left(\frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}\right)^{T}[\boldsymbol{C}\boldsymbol{Q}^{-1}\boldsymbol{D}]\{\boldsymbol{Q}\boldsymbol{C}^{-1}[\boldsymbol{W}\boldsymbol{V}+\boldsymbol{I}-\boldsymbol{P}(\boldsymbol{V})\boldsymbol{R}]\}\\&=-\left(\frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}\right)^{T}\{\boldsymbol{D}[\boldsymbol{W}\boldsymbol{V}+\boldsymbol{I}-\boldsymbol{P}(\boldsymbol{V})\boldsymbol{R}]\}\\&=-\left(\frac{\mathrm{d}\boldsymbol{V}}{\mathrm{d}t}\right)^{T}[\boldsymbol{D}\boldsymbol{W}\boldsymbol{V}-\boldsymbol{D}\boldsymbol{I}-\boldsymbol{D}\boldsymbol{P}(\boldsymbol{V})\boldsymbol{R}]\end{aligned}$$

令 $\boldsymbol{B}+\boldsymbol{B}^{T}=\boldsymbol{D}\boldsymbol{W}$；保证 $\boldsymbol{D}\boldsymbol{W}$ 在能量函数中对称：

$$\boldsymbol{I}'=\boldsymbol{D}\boldsymbol{I};\boldsymbol{R}'=\boldsymbol{D}\boldsymbol{R}$$

则 Hopfield 能量函数可写为

$$E(V)=-\boldsymbol{V}^{T}\boldsymbol{B}\boldsymbol{V}-\boldsymbol{I}'\boldsymbol{V}+\boldsymbol{R}'^{T}\boldsymbol{S} \tag{3-80}$$

(3-80)式对时间的全微分为(3-79)式，可保在它$\leqslant 0$，而 $E(\boldsymbol{V})$ 是一个有限函数，即使 $\boldsymbol{W}$ 不对称仍可保证其稳定。在运用中，如果(3-77)状态方程中 $\boldsymbol{W}$ 是非对称的，但只要找到一个对角正定阵 $\boldsymbol{D}$，使 $\boldsymbol{D}\boldsymbol{W}$ 能够得到一个对称阵，这样也就从(3-80)式得到了能量函数 $E(\boldsymbol{V})$，由此也就可以决定系统的稳定与否了。

我们注意到 Hopfield 能量函数在一般情况下与李雅普诺夫函数的区别在于：李雅普雅夫函数只限制在大于零的范围内，而能量函数则可以小于零，但要负向有界；同时李雅普诺夫函数要求在零点的值为零，即$V(0)=0$；而能量函数则不一定，当能量函数 E 满足$E(V)>0$，$E(0)=0$，$\frac{\mathrm{d}E}{\mathrm{d}t}\leqslant 0$ 的条件时，Hopfield 能量函数就是李雅普诺夫函数，在以后的设计中，我们主要从能量函数出发来讨论，有时也运用李雅普诺夫条件。

五、能量函数与优化计算

所谓优化问题是求满足一定约束条件下的目标函数极小问题，优化的传统算法很多，如梯度法、单纯型法等；在某些情况由于条件过于复杂，变量的维数较多，使优化工作耗费过多而达不到预期的结果。对于 CHNN 网络，能量函数是一个反映多维神经元状态的一标量函数，而且直接可以用简单的电路形成人工神经元网络，它们的互联形成了并联计算的机制，而系统可以随时间的变化自然地收敛到那些渐近稳定点上，在这些稳定点上能量函数达到极小，因此如果人为地设计神经网络之间的联接权矩阵 $\boldsymbol{W}$ 和输入 $\boldsymbol{I}$，把优化问题中的目标函数、约束条件与 Hopfield 的能量函数联系起来，那么电路达到的平衡点就是能量函数的

极小点，也是优化中满足约束条件下的目标函数的极小点，这就可以利用人工神经网络来完成优化问题，由于人工神经网络是并行计算，其计算量不随维数的增加而发生指数性质的"爆炸"，因而对于高速的优化处理特别有效。

下面我们给出能量函数设计的一般方法。

设优化的目标函数为$f(\boldsymbol{x})$，$\boldsymbol{x}\in R^n$为人工神经网络的状态，也是目标函数中的变量，$g(\boldsymbol{x})=0$为约束条件，优化问题就归结为满足约束条件下使目标函数最小，从而设计出能量函数E：

$$E=f(\boldsymbol{x})+C|g(\boldsymbol{x})|>0$$

这里，$C|g(\boldsymbol{x})|$也称为惩罚函数，因为在约束条件不能满足时，$C|g(\boldsymbol{x})|$值将很大，造成E很大，这意味着一定要满足约束条件。

若E的全导数要小于零，则要求满足：

$$\frac{\mathrm{d}x_i}{\mathrm{d}t}=-\frac{\partial E(x)}{\partial x_i}\quad\forall i \tag{3-81}$$

因为

$$\frac{\mathrm{d}E}{\mathrm{d}t}=\sum_i\frac{\partial E}{\partial x_i}\frac{\mathrm{d}x_i}{\mathrm{d}t}=\sum_i-\left(\frac{\mathrm{d}x_i}{\mathrm{d}t}\right)^2\leqslant 0$$

按照 Hopfield 能量函数的要求，只要E在负的方向有界：$E>E_{\min}$，同时$\frac{\mathrm{d}E}{\mathrm{d}t}\leqslant 0$，则系统最后总能达到$E$最小且$\frac{\mathrm{d}E}{\mathrm{d}t}=0$的点，同时又是系统的稳定点，$\frac{\mathrm{d}x_i}{\mathrm{d}t}=0$。

对于目标函数$f(\boldsymbol{x})$，一般总取一个要求值和实际值之间的误差的平方或绝对值，这样$f(\boldsymbol{x})$总是大于零的，而约束条件也可以设计为一个大于零的数，例如要求$g(\boldsymbol{x})>0$，$E>E_{\min}=0$，那么E可以设计为

$$E=f(\boldsymbol{x})+\mathrm{e}^{-kg(\boldsymbol{x})}$$

在要求$g(\boldsymbol{x})<0$的情况下，能量函数E为

$$E=f(\boldsymbol{x})+\mathrm{e}^{kg(\boldsymbol{x})}$$

在有多个约束条件下，可以把能量函数E设计为

$$E=f(\boldsymbol{x})+\sum_{\alpha=1}^{P}C_\alpha\rho[g_\alpha(\boldsymbol{x})]$$

其中P为约束条件的数目，C_α为常数，$\rho[g_\alpha(\boldsymbol{x})]$为与约束条件有关的函数。

在此以前，我们已经看到 Hopfield 能量函数不是直接与状态变量$\boldsymbol{x}$有关，而是与输出变量$\boldsymbol{V}$直接有关，但是输出变量$\boldsymbol{V}$与$\boldsymbol{x}$的关系为

$$V_i=F(x_i)\qquad\frac{\partial V_i}{\partial x_i}=F'(x_i)$$

$$\frac{\mathrm{d}V_i}{\mathrm{d}t}=\frac{\partial V_i}{\partial x_i}\frac{\mathrm{d}x_i}{\mathrm{d}t}=F'(x_i)\frac{\mathrm{d}x_i}{\mathrm{d}t}$$

或

$$V_i=F(x_i)\qquad x_i=F^{-1}(V_i)$$

只要$\frac{\partial V_i}{\partial x_i}>0$，$F(x_i)$在$E$函数中采用什么形式，其结果都是相同的。

具体设计步骤如下：

(1) 根据要求的目标函数，写出能量函数的第一项$f(\boldsymbol{x})$；

(2) 根据约束条件$g(\boldsymbol{x})$，写出惩罚函数，使其满足约束条件时惩罚函数最小，作为能量

函数的第二项。

(3) 加上一项$\sum_i \frac{1}{R_i}\int_0^{V_i} F^{-1}(\eta)\mathrm{d}\eta$。这一项是人为的，因为在神经元状态方程中，存在一项$-\frac{x_i}{R_i}$，它是人工神经网络电路设计中产生的，为了使设计优化的结果能在电路中得以实现，在能量函数中加上了这一项$\sum_i \frac{1}{R_i}\int_0^{V_i} F^{-1}(\eta)\mathrm{d}\eta$，这是一个正值函数，表现出在边缘上能量大，并且$V_i$值越小，能量值也越小，在运算放大器放大倍数足够大时，又可忽略，因而它对能量E和优化问题的结果影响不大。

(4) 根据能量函数E求得状态方程，满足(3-81)式：

$$\frac{\partial E}{\partial x_i}=-\frac{\mathrm{d}x_i}{\mathrm{d}t} \quad \forall i$$

(5) 根据状态方程，对照公式(3-72a)求出$w_{ij}, I_i, i, j=1,2,\cdots,n$。

(6) 用类似于图3-10的电路实现：

$$\frac{1}{R_{ij}}=w_{ij} \quad \forall i.j$$

为了说明上述的设计方法，我们举一个简单的例子来说明如何根据要求设计能量函数，继而由能量函数来得到电路的。

[例3-6] 用人工神经网络来设计一个A/D变换器，要求输入是一个连续的模拟量，输出为0,1的数字量$V_i\in\{0,1\}$，V_i代表第i个神经元的输出$V_i=F(x_i)$，F为单调上升有限函数。

(1) 如果考虑四位A/D，那么其目标函数$f(\boldsymbol{V})$为

$$f(\boldsymbol{V})=\frac{1}{2}\left(A-\sum_{i=0}^{3}V_i2^i\right)^2>0$$

式中：A为输入的模拟量；V_i是数字量，对于$i=0,1,2,3$，只可能为1或0；2^i表示数字量的二进制位数。目标函数大于零，因此在$f(\boldsymbol{V})$存在极小界线。

(2) 考虑约束条件

$$g(\boldsymbol{V})=-\frac{1}{2}\sum_{i=0}^{3}(2^i)^2(V_i-1)V_i$$

$g(\boldsymbol{V})$是保证输出V_i只能为0或1，因为只有在$V_i=0$，或$V_i=1$时，$g(\boldsymbol{V})=0$，而V_i在0与1之间的任意实数$g(\boldsymbol{V})>0$，因此约束条件是输出为数字量的保证。

(3) 写出总的能量函数E，这里需要考虑到电路设计时的具体条件而加上一个大于零的积分项：

$$t_P=\sum_i\frac{1}{R_i}\int_0^{V_i}F^{-1}(\eta)\mathrm{d}\eta$$

可得

$$\begin{aligned}E&=f(\boldsymbol{V})+g(\boldsymbol{V})+t_P(\boldsymbol{V})\\&=\frac{1}{2}\left(A-\sum_{i=0}^{3}V_i2^i\right)^2-\frac{1}{2}\sum_{i=0}^{3}(2^i)^2(V_i-1)V_i\\&\quad+\sum_i^1\int_0^{V_i}\frac{1}{R_i}F^{-1}(\eta)\mathrm{d}\eta\end{aligned}$$

十分明显，E是大于零的，但$E(0)\neq 0$，$E(0)=\frac{1}{2}A^2$，E在低端是有界的，$E_{\min}\geqslant 0$。

(4) 计算$\frac{dV_i}{dt}$，这里V_i是输出值，与状态值x_i有一点差距，$\frac{dV_i}{dt}=F'(x_i)\frac{dx_i}{dt}$，$F'(x_i)>0$在放大区内近似为一个常数，假如令在放大区内的$F'(x)=C_i$，则有

$$\begin{aligned}\frac{\partial E}{\partial V_i}&=-\frac{dV_i}{dt}=-C_i\frac{dx_i}{dt}\\&=-2^iA+\frac{1}{2}\sum_j 2^{i+j}V_j+\frac{1}{2}\sum_i 2^{j+i}V_i-(2^j)^2+2^{2i-1}+\frac{1}{R_i}x_i\\&=-\frac{x_i}{R_i}+\sum_{\substack{j\\(\neq i)}}2^{i+j}V_j+2^{2i-1}-2^iA\end{aligned}\tag{3-82}$$

(5) 将(3-82)式与(3-72a)比较，可得

$$w_{ij}=(-2^{i+j});\quad I_i=(-2^{2i-1}+2^iA)$$

(6) 根据(5)可得

$$w_{12}=w_{21}=-2^3;\quad w_{13}=w_{31}=-2^4=-16;\quad w_{01}=w_{10}=-2;$$
$$w_{02}=w_{20}=-4;\quad w_{30}=w_{03}=-8;\quad w_{11}=w_{22}=w_{33}=w_{00}=0$$

输入由两个量组成，一个是模拟量输入后得到的，一个是固定的输入量，其状态方程满足：

$$C_i\frac{dx_i}{dt}=-\frac{x_i}{R_i}+\sum_j w_{ij}V_j+I_i$$

根据上面的设计，我们可以用电路来实现，其电路如图3-20所示，权的负值是用两个倒相的运算放大器来完成，图中标出的数值是以导纳表示的w_{ij}值，其结果如图3-21所示，输入为0～15V，而输出为0000～1111，电路正确地反映了这种输入模拟量A和输出数字量D之间的关系。

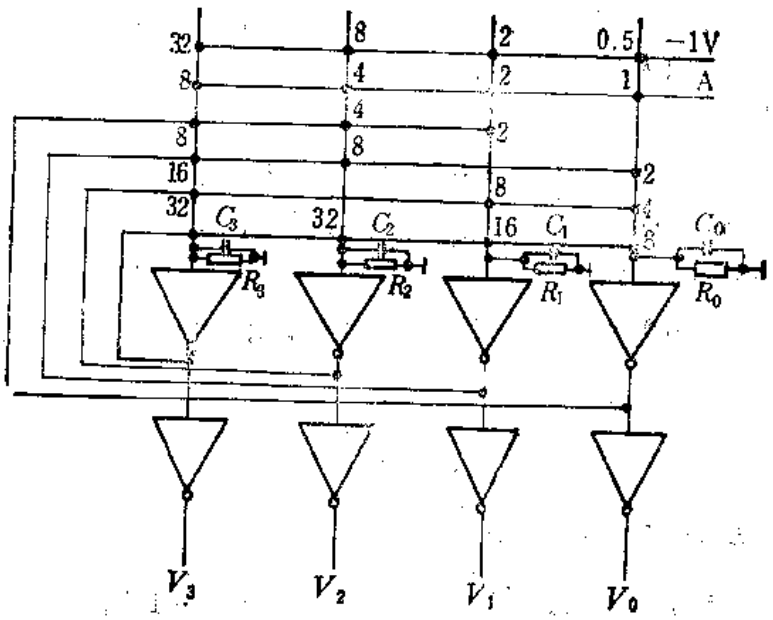


图3-20 A/D变换的电路

从这个例子可以看出，优化问题的设计是设法将问题转化为能量函数，这种转化没有现成的公式，而是一种艺术，转化好了，电路也可能随之设计出来了，同时解也可得到。下面我

们将介绍利用 Hopfield 能量函数法解各种优化或近似优化问题，它具有相当的实用价值。

六、应用举例

(一) TSP问题(Travelling Salesman Problem)

TSP 是一种十分难的优化问题：一个推销员去 n 个城市推销他的产品，从他所在的城市出发访问 $n-1$ 个城市，最后又回到原来的城市，要求每个城市只经过一次，并且所有的城市都有走到，要求他经过的路程为最短。如果已知城市 $A,B,C,D,\cdots$ 间的距离为 $d_{AB},d_{AC},\cdots$，那么总的距离 $d=d_{AB}+d_{AC}+_{AD}+\cdots$，对于这种动态规划去求得 $\min(d)$ 的解，在城市数目少的情况下尚可以计算，当城市数目增加，计算量就会增加到无法进行的地步，是一个 NP 完全问题，对于这个问题如果采用 Hopfield 能量函数来设计，在城市数 $n<100$ 时，采用电路来实现时，它的计算速度非常快。

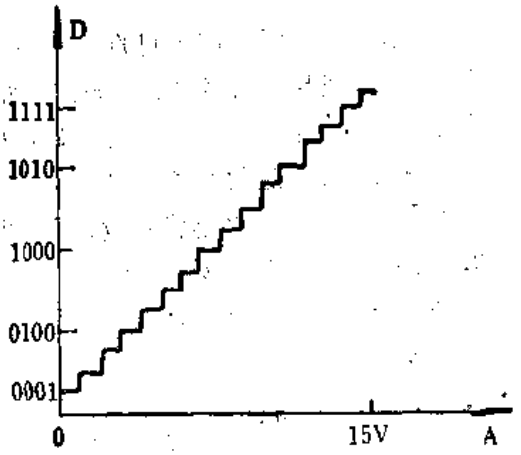


图 3-21　A/D变换的结果

这里我们着重讨论如何把这个问题转化为一个能量函数，再由能量函数转化为状态方程和电路，问题就解决了。

要解n城市的 TSP 问题，是把问题映照到一个复杂的神经网络上，假定每个神经元的放大器有很高的放大倍数，而神经元的输出被限制在二值 0 和 1 上，那么映照问题是用一个关联矩阵进行，关联矩阵中的每个元只能为 0 和 1（如果系统最后收敛的话），如以 5 个城市 A、B、C、D、E 为例，它的关联矩阵可表示为

	1	2	3	4	5
A	0	1	0	0	0
B	0	0	0	1	0
C	1	0	0	0	0
D	0	0	0	0	1
E	0	0	1	0	0

n 个城市用 n^2 个神经元表示，其中 n 行表示 n 个城市，n 列表示推销员到达某城市的先后，在上述关联矩阵中反映推销员到达城市的顺序为

$$C\rightarrow A\rightarrow E\rightarrow B\rightarrow D\rightarrow C$$

而推销员所走的距离为

$$d=d_{CA}+d_{AE}+d_{EB}+d_{BD}+d_{DC}$$

为了保证每个城市只去一次（不包括初始出发城市），则在关而矩阵上每一行只能有一个为 1，其他均为零，对于关联矩阵的次序上，也是每一列只能有一个元为 1，其他为零，把按要求最后得到的关联矩阵的情况化为 Hopfield 能量函数。

1. 目标函数 $f(\boldsymbol{V})$

把关联矩阵的每个元用符号 $V_{xi},V_{yj},\cdots$ 来表示，下标 x,y 表示城市 $A,B,C,D,\cdots$ 下标

i,j 表示顺序，这样目标函数 $f(V)$ 表示为

$$f(V)=\frac{P}{2}\sum_{x}\sum_{\substack{y\\ y\neq x}}\sum_{i}d_{xy}V_{xi}(V_{y,i+1}+V_{y,i-1})$$

d_{xy} 表示两个不同城市之间的距离，如果 $V_{xi}=0$，那么这个输出对 $f(V)$ 没有贡献，如果 $V_{xi}=1$，那么寻找与 i 相邻次数的城市，如在关联矩阵中 $V_{A2}=1$，那么就可以找到 V_{C1} 和 V_{E3} 的两个与 V_{B2} 相邻列上的非零值，此时在 $f(V)$ 中得到 d_{AB} 和 d_{BE} 两个相加的量，依次类推，那些推销员走过的距离都累加起来，在 $f(V)$ 中只不过是每个距离计算了两次，在目标函数 $f(V)$ 中前面的 P 是常系数，而 1/2 是为了除去重复计算而设的。

2. 约束条件(惩罚函数) $g(V)$

这是一个约束条件问题，其约束条件是要保证关联矩阵每一行和每一列只有一个值为1，其他为零，总的惩罚函数用三项表示：

$$g(V)=\frac{Q}{2}\sum_{i}\sum_{\substack{x\\ y\neq x}}\sum_{y}V_{xi}V_{yi}+\frac{S}{2}\sum_{x}\sum_{\substack{i\\ j\neq i}}\sum_{j}V_{xi}V_{xj}$$
$$+\frac{T}{2}\left(\sum_{x}\sum_{i}V_{xi}-n\right)^{2}$$

第一项表示同一次只能到达一个城市，因此满足约束条件时，这一项为零，第二项表示每个城市只能去一次，满足约束条件时第二项为零，第三项表示总的 V_{xi} 为 1 的数目最多为 n，因而在满足约束条件的情况下 $g(V)=0$

3. 总的能量函数

这里，由于使用了高增益放大器，因此能量函数中的积分项可以忽略不计，得能量函数：

$$E=\frac{P}{2}\sum_{x}\sum_{\substack{y\\ y\neq x}}\sum_{i}d_{xy}V_{xi}(V_{y,i+1}+V_{y,i-1})+\frac{Q}{2}\sum_{i}\sum_{\substack{x\\ x\neq y}}\sum_{y}V_{xi}V_{yi}$$
$$+\frac{S}{2}\sum_{x}\sum_{\substack{i\\ i\neq j}}\sum_{j}V_{xi}V_{xj}+\frac{T}{2}\left(\sum_{x}\sum_{i}V_{xi}-n\right)^{2}$$

系数 P、Q、S、T 和距离 d_{xy} 都是大于零的，因而 E 满足 $E>0$，同时按照前面提到的设计步骤(4)来考虑状态方程，就可以保证 $\frac{dE}{dt}\leqslant 0$，由电路组成的人工神经网络按 (4) 的权和输入的要求，系统最后将收敛到一些稳定点，它们都是较优路径的解。

(二) 利用电路计算运动物体的速度场

速度场是物体运动时，从两幅不同时间拍摄的照片上计算出物体上每个像素(或称格点)的速度方向和大小，以表示物体的运动情况，这种各格点速度方向的组合形成的图形称为速度场。由于照片是二维的，这里主要考虑三维图像速度场的计算，它的条件为视角小，使物体没有偏离相机的视平面。考虑物体是刚性的，物体的光强在运动时不变，那么运动物体的速度场也可以采用神经网络来计算。

1. 目标函数

物体是刚性的，因而速度场是连续的，考虑速度的变化率应该最小，即如果图形的 x、y 方向的速度分别用

$$u=\frac{dx}{dt};\qquad v=\frac{dy}{dt}$$

来表示，那么其输出速度场的速度在二维平面上是一个连续函数，得到目标函数为

$$\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial u}{\partial y}\right)^2+\left(\frac{\partial v}{\partial x}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2$$

达到最小。

2. 约束条件

如果在一张照片的二维图像上每个点的光强为 $I(x,y)$，那么总光强为

$$I=\sum_{x,y} I(x,y)$$

由于总光强也是不变的，因而有

$$\frac{\mathrm{d}I}{\mathrm{d}t}=0$$

可得

$$\sum_{x,y}\left(\frac{\partial I}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t}+\frac{\partial I}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t}+\frac{\mathrm{d}I}{\mathrm{d}t}\right)=0$$

令

$$\frac{\partial I}{\partial x}=I_x;\quad \frac{\partial I}{\partial y}=I_y;\quad \frac{\mathrm{d}I}{\mathrm{d}t}=I_t$$

得到能量函数为

$$E(u,v)=\iint\left\{(I_xu+I_yv+I_t)^2+\lambda\left[\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial u}{\partial y}\right)^2+\left(\frac{\partial v}{\partial x}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2\right]\right\}\mathrm{d}x\mathrm{d}y \tag{3-83}$$

式中 λ 是一个常数。

在二维图像上不同的 x 和 y 时，其速度大小、速度变化和光强的变化不同，对于一点来说，能量变得没有意义，而总的能量分别为各点的能量之和，上式的积分形式就是其累加的结果。

对(3-83)式进行变分运算，在稳定时 $\delta E=0$，得到两个线性方程：

$$\left.\begin{aligned}I_x^2u+I_xI_yv-\lambda\nabla^2u+I_xI_t=0\\ I_xI_yu+I_y^2v-\lambda\nabla^2v+I_yI_t=0\end{aligned}\right\} \tag{3-84a}$$

把图像分为许多小的格点，它们均匀分布在二维空间内如图 3-22 所示，对某一个 i,j 格点，表示为第 i 行第 j 列，它与周围格点之间的关系可用式(3-84a)写成下列形式：

$$\begin{aligned}I_{xij}^2u_{ij}+I_{xij}I_{yij}v_{ij}-\lambda(u_{i+1j}+u_{ij+1}\\ +u_{i-1j}+u_{ij-1}-4u_{ij})+I_{xij}I_{tij}=0\end{aligned} \tag{3-84b}$$

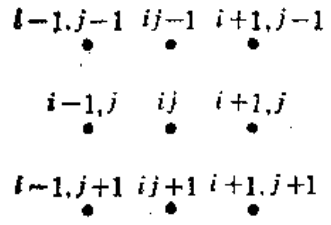


图 3-22 图像小格点的位置

$$I_{xij}I_{yij}u_{ij}+I_{yij}^2v_{ij}-\lambda(v_{i+1j}+v_{ij+1}+v_{i-1j}+v_{ij-1}-4v_{ij})+I_{yij}I_{tij}=0 \tag{3-84c}$$

其中 I_{xij}、I_{yij}、I_{tij} 表示在 ij 格点上的光强随 x、y、t 的变化率，u_{ij}、v_{ij} 为 ij 格点的 x、y 方向的速度，在(3-84a)式中的 ∇^2u、∇^2v，可以用两阶差分方程来表示

$$\nabla^2u=\frac{\partial^2u}{\partial x}+\frac{\partial^2u}{\partial y}\quad \text{对 } ij \text{ 格点为}$$

$$\begin{aligned}&[(u_{ij-1}-u_{ij})-(u_{ij}-u_{ij+1})]-[(u_{ij}-u_{i+1j})-(u_{i-1j}-u_{ij})]\\ &=(u_{ij-1}+u_{ij+1}+u_{i+1j}+u_{i-1j})-4u_{ij}\end{aligned}$$

$\nabla^2 v=\frac{\partial^2 v}{\partial x}+\frac{\partial^2 v}{\partial y}$；对 ij 格点为

$$[(v_{ij-1}-v_{ij})-(v_{ij}-v_{ij+1})]-[(v_{ij}-v_{i+1j})-(v_{i-1j}-v_{ij})]$$
$$=(v_{ij-1}+v_{ij+1}+v_{i+1j}+v_{i-1j})-4v_{ij}$$

根据前面设计方法的步骤(4)得到

$$\frac{\partial E}{\partial u_{ij}}=-\frac{\mathrm{d}u_{ij}}{\mathrm{d}t}$$

因此有

$$C\frac{\mathrm{d}u_{ij}}{\mathrm{d}t}=T(u_{i+1j}+u_{ij+1}+u_{i-1j}+u_{ij-1}-4u_{ij})+g_{ij}^{u}(E_{ij}-u_{ij})$$
$$+T_{e-ij}(v_{ij}-u_{ij}) \tag{3-85a}$$

$$C\frac{\mathrm{d}v_{ij}}{\mathrm{d}t}=T(v_{i+1j}+v_{ij+1}+v_{i-1j}+v_{ij-1}-4v_{ij})+g_{ij}^{v}(E_{ij}-v_{ij})$$
$$+T_{e-ij}(u_{ij}-v_{ij}) \tag{3-85b}$$

其中：

$$T=\lambda;\quad g_{ij}^{u}=I_{xij}(I_{xij}+I_{yij})$$
$$g_{ij}^{v}=I_{yij}(I_{xij}+I_{yij})$$
$$E_{ij}=\frac{-I_{zij}}{I_{xij}+I_{yij}}$$
$$T_{e-ij}=-I_{xij}I_{yij}$$

比较(3-84b)和(3-85a)式、(3-84b)和(3-85b)式，显然它们是一致的。要完成(3-85a、b)的网络可采用光敏元件和电阻网络相结合的电路，E_{ij}, g_{ij}^{u}, g_{ij}^{v} 都是与相片上的光强有关的量，每个光敏器件也是格型排列的，它们所产生的导纳和电压直接加到格型的电阻网络上。图3-23a就是完成(3-85a)状态方程的一个电阻网络，在格点 ij 上输入 E_{ij}，并接入导纳 g_{ij}^{u} 和电容 C，并且电阻 T 与其他格点相连，总的网络形状如图 3-23b 所示，它是由两层如图3-23a的网络组成，都与光敏器件相连，两层之间又用 T_{e-ij} 相连接，当运动物体在光敏器件发出 E_{ij}、g_{ij}^{u}、g_{ij}^{v}；加入到网络，在网络的各格点上输出为 u_{ij}、V_{ij} 的值，使之达到迅速计算速度场的目的。

该电阻网络已制成芯片用于卫星上。

3. 组合式的 Hopfield 网络

我们可以用一个能量函数来表示优化问题中的目标函数和约束条件，以设计电路完成优化问题。这里我们介绍运用一种组合形式的电路来完成约束条件下的优化问题，假定目标函数是一个线性方程，要完成在一个约束条件下的线性规划问题，求目标函数最小。

令 π 为一个线性规划的目标函数，它是由系数矢量 $\boldsymbol{A}$ 和 $\boldsymbol{V}$ 组成，希望在 π 最小时求 $\boldsymbol{V}$ 的值。

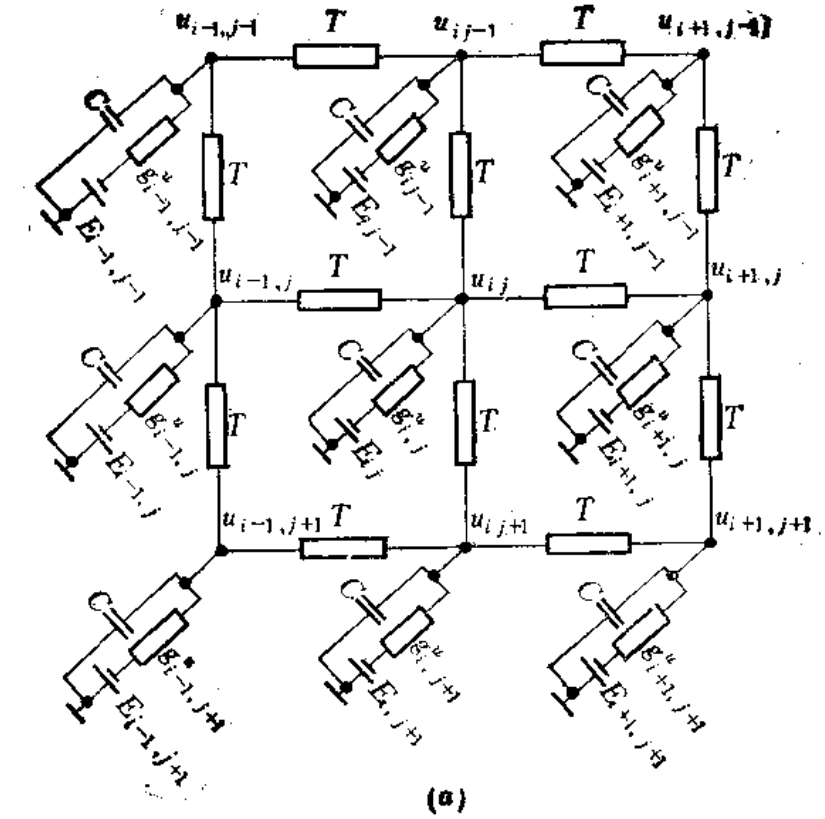
$$\pi=\boldsymbol{A}^{T}\boldsymbol{V}\quad \boldsymbol{A}\in R^{n};\ \boldsymbol{V}\in R^{n}$$

其中 $\boldsymbol{A}$ 为 n 维的系数矢量，$\boldsymbol{V}$ 为 n 维的变量，并且满足 M 个约束条件：

$$\boldsymbol{D}_{j}^{T}\boldsymbol{V}\geqslant B_{j},\quad j=1,2,\cdots,M$$

式中：$\boldsymbol{D}_{j}=[D_{j1},D_{j2},\cdots,D_{jn}]^{T}$

可以用图 3-24 所示的电路来完成这个线性规划的问题，它是由两部分神经元网络组合而成的，左面的网络是完成目标函数到达极小，即 $\min(\boldsymbol{A}^{T}\boldsymbol{V})$，右面部分的网络是约束条件



(a) 完成状态方程(3-85a)的电阻网络; (b) 速度场计算的电阻网络

图 3-23

的网络，它们相互制约，最后达到平衡点。在图 3-24 中，权 D_{ij} 是约束条件方程的系数，按约束条件方程的数目来决定神经元 ψ_j 的数目，而目标函数的系数矢量 $\boldsymbol{A}$ 为左面网络的输入。

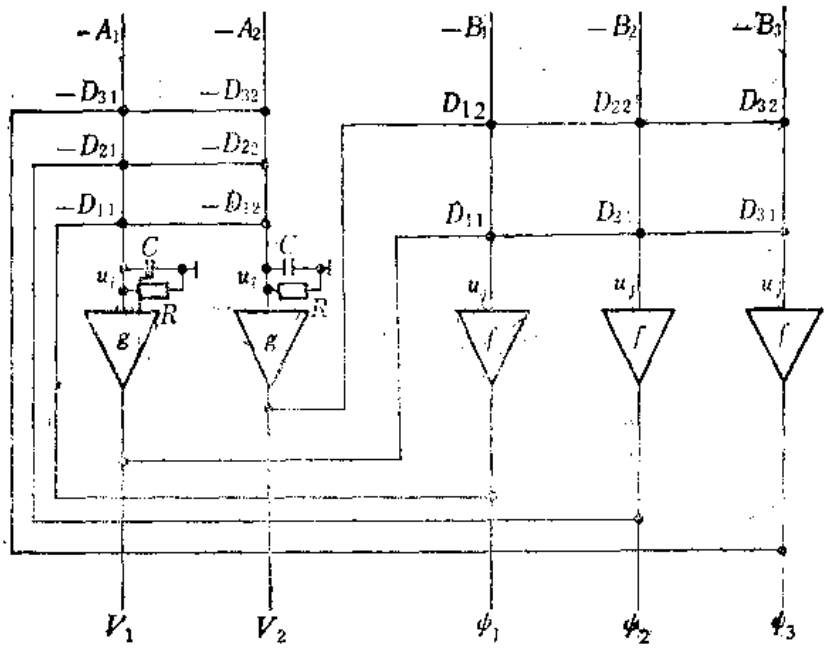


图 3-24　组合式 Hopfield 网络

在图 3-24 中 g 函数是线性函数，$V_i=g(u_i)=u_i$，u_i 是左面网络的状态。图中的 f 函数与 z 值大小有关。对于右面约束条件神经网络的状态为 u_j，$u_j=\boldsymbol{D}_j^T\boldsymbol{V}-B_j$，则输出为

$$\psi_j=f(u_j)$$

$$f(z)=0;\quad z\geqslant 0$$

$$f(z)=-z;\quad z<0$$

因此当约束条件满足时，右面网络的输出等于零，而左面网络在动态过程下达到极小，事实上由于组合网络相互牵制，约束网络的输出不可能完全等于零。

按图 3-24，目标函数网络的状态方程为

$$C\frac{\mathrm{d}u_i}{\mathrm{d}t}=-A_i-\frac{u_i}{R}-\sum_j D_{ji}f(u_j)$$

$$=-A_i-\frac{u_i}{R}-\sum_j D_{ji}f(D_j\cdot V-B_j) \tag{3-86}$$

其能量函数为

$$E=\boldsymbol{A}^T\boldsymbol{V}+\sum_j F(\boldsymbol{D}_j^T\boldsymbol{V}-B_j)-\sum_i\frac{1}{R}\int_0^{V_i}g^{-1}(V)\mathrm{d}V \tag{3-87}$$

如果 $\boldsymbol{D}_j^T\boldsymbol{V}-B_j=z$，则有

$$f(z)=\frac{\mathrm{d}F(z)}{\mathrm{d}z}$$

对式(3-87)全微分得

$$\frac{\mathrm{d}E}{\mathrm{d}t}=-\left[\frac{u_i}{R}+A_i+\sum D_{ji}f(\boldsymbol{D}_j\boldsymbol{V}-B_j)\right]$$

$$=-\sum_i C_i \frac{dV_i}{dt}\frac{du_i}{dt}=-\sum_i C_i \frac{dV_i}{du_i}\left(\frac{du_i}{dt}\right)^2$$

$$=-\sum_i C_i g^{-1}(V_i)\left(\frac{dV_i}{dt}\right)^2 \leqslant 0$$

网络满足稳定条件，同时也满足线性规划的要求。如果在满足约束条件的情况下，$f(x)=0$，$F(x)$也为0，则可将(3-87)式写成

$$E=A^TV+\sum_i \frac{1}{R}\int_0^{V_i} g^{-1}(\eta)\,d\eta$$

上式第二项在放大器放大倍数很大时，可以忽略，E主要由第一项决定，E的极小值就相当于A^TV的极小，而满足$\pi\to\min$。

利用图3-24的电路可以完成DFT（离散傅里叶变换）的运算，这里是首先利用组合网络来完成离散哈特利变换，然后组合离散哈特利变换的结果达到离散傅里叶变换。

如果一个随时间变化的模拟函数$b(t)$，在一个有限时间内对此函数进行采样，共采入N点，用$b(\tau)$表示，τ为离散的时间，$b(\tau)$为离散时间上的采样值，函数$b(t)$的离散哈特利变换(DHT)为$V(\nu)$，ν为频率的离散量，有

$$\left.\begin{aligned} V(\nu)&=\frac{1}{N}\sum_{\tau=0}^{N-1} b(\tau)\left[\cos\left(\frac{2\pi\nu\tau}{N}\right)+\sin\left(\frac{2\pi\nu\tau}{N}\right)\right]\\ b(\tau)&=\sum_{V=0}^{N-1} V(\nu)\left[\cos\left(\frac{2\pi\nu\tau}{N}\right)+\sin\left(\frac{2\pi\nu\tau}{N}\right)\right]\end{aligned}\right\} \tag{3-88}$$

$b(\tau)$是$V(\nu)$的反哈特利变换(IDHT)。

哈特利变换可以写成矩阵的形式，只要引入N阶方阵D，使每个单元为

$$D_{ij}=\cos\left(\frac{2\pi ij}{N}\right)+\sin\left(\frac{2\pi ij}{N}\right)$$

此时DHT可以写成

$$\boldsymbol{V}=\boldsymbol{D}^{-1}\boldsymbol{b}\quad \boldsymbol{V}\in R^n;\ \boldsymbol{b}\in R^n;\ \boldsymbol{D}\in R^{n\times n}$$

其反变换为

$$\boldsymbol{b}=\boldsymbol{D}\boldsymbol{V}$$

因此实现DHT问题就可以归结为如何实现变换矩阵$\boldsymbol{D}$以及$\boldsymbol{D}$与向量$\boldsymbol{V}$的乘法计算，这个计算可以化为一个优化问题，使其目标函数$\boldsymbol{DV}-\boldsymbol{b}^2\to\min$，变换矩阵$\boldsymbol{D}$有如下三个性质：(1) $\boldsymbol{D}=\boldsymbol{D}^T$，(2) $\boldsymbol{D}^2=N\boldsymbol{u}$；$\boldsymbol{D}^{-1}=\frac{1}{N}\boldsymbol{D}$；(3) $\boldsymbol{D}$的特征值为$\pm\sqrt{N}$。这里$\boldsymbol{U}$为单位矩阵。

由DHT可以得到离散傅里叶变换DFT。

DFT的定义为

$$f(\nu)=\frac{1}{N}\sum_{\tau=0}^{N-1} b(\tau)\left[\cos\frac{2\pi\nu\tau}{N}-j\sin\frac{2\pi\nu\tau}{N}\right]=P(\nu)+jO(\nu)$$

式中：$P(\nu)$、$O(\nu)$分别为DFT的实部和虚部。

观察

$$V(\nu)+V(N-\nu)=\frac{1}{N}\sum_{\tau=0}^{N-1} b(\tau)\left[\left(\cos\frac{2\pi\nu\tau}{N}+\sin\frac{2\pi\nu\tau}{N}\right)+\left(\cos\frac{2\pi(N-\nu)\tau}{N}+\sin\frac{2\pi(N-\nu)\tau}{N}\right)\right]=\frac{2}{N}\sum_{\tau=0}^{N-1} b(\tau)\cos\frac{2\pi\nu\tau}{N}=2P(\nu)$$

所以

$$P(\nu)=\frac{V(\nu)+V(N-\nu)}{2}$$

同理

$$O(\nu)=\frac{-V(\nu)+V(N-\nu)}{2}$$

这意味着一旦得到 DHT 的结果 $V(\nu)$，那么只要做简单的加法就可以方便地得到 DFT 的实部和虚部。

用图 3-24 的网络来实现 DHT，只要把 $(\boldsymbol{DV}-\boldsymbol{b})$ 看作约束条件，令 $\boldsymbol{A}=0$，则 $A_1, A_2\cdots$ 为零，即目标函数网络的输入为零，而约束条件的输入则对应为模拟信号 $b(t)$ 的采样数量 $b(\tau)$，如取 N 个采样点，则 $b(\tau)$ 为 N 个不同时间的采样值，约束网络的神经元个数为 N，由于目标函数网络的输出为 DHT 变换的 $V(\nu)$，所以目标函数网络的神经元数也为 N 个。利用 $g(u_i)=\beta u_i$（β 为放大器的放大倍数），β 为常数，$g(\cdot)$ 函数是一个线性函数，而约束方程中：

$$f(z)=\alpha z;$$

利用公式(3-87)得到能量函数为

$$E=\sum_{j=0}^{N-1}F(\boldsymbol{D}_j^{\mathrm{T}}\boldsymbol{V}-B_j)+\sum_{i=0}^{N-1}\frac{1}{R_i}\int_0^{V_i}\frac{1}{\beta}V\mathrm{d}V$$

其中

$$F(\boldsymbol{D}_j^{\mathrm{T}}\boldsymbol{V}-B_j)=\int f(\boldsymbol{D}_j^{\mathrm{T}}\boldsymbol{V}-B_j)=\frac{\alpha}{2}\left(\sum_{j=0}^{N-1}D_{ij}V_i-B_j\right)^2$$

所以能量函数 E 为

$$E=\sum_{j=0}^{N-1}\frac{\alpha}{2}\left(\sum_{j=0}^{N-1}D_{ij}V_i-B_j\right)^2+\sum_{i=0}^{N-1}\frac{1}{R_i}\int_0^{V_i}\frac{1}{\beta}V\mathrm{d}V$$

$$=-\frac{\alpha}{2}\left|\boldsymbol{DV}-\boldsymbol{B}\right|^2+\frac{1}{\beta R}|\boldsymbol{V}|^2,\ R_i=R,\ i=0,1,\cdots,N-1$$

在稳定点时能量为最小，得

$$\boldsymbol{DV}-\boldsymbol{B}=0\quad \boldsymbol{V}=\boldsymbol{D}^{-1}\boldsymbol{B}$$

网络的输出 V_i 就是我们要求的 DHT 的输出，经过组合可得 DFT 的输出。

Hopfield 的能量函数可以用来解优化问题，它的主要优点是用一个人工神经网络的电路就可以完成优化工作，因此计算速度快，而且避免了复杂性问题。但是 Hopfield 网络的电路本身还存在以下两个问题：

(1) 联接权的数目随神经元数而增加，有 N 个神经元组成的网络，则其联接权数目为 N^2 个，这对电路的实现会产生较大的困难，在规模比较大的情况下，无法用电路来完成，于是 CHNN 网络解决了计算上的 NP 问题，但又带来了实现上的 NP 问题。

(2) 网络是非线性的，因而随着 N 的增加，其局部极小点的数目也随之增加，使优化的结果不能达到全局的最优。这就影响了优化计算的精度，当然，也可以把问题退回到线性范围内进行，如在前面 DFT 计算的应用中，就是完全用线性的输入、输出关系来实现优化的，此时不会出现局部最小的问题，可是多数的优化问题一般存在多个约束条件，这就使非线性不可避免的存在，这也使得网络的规模不能做得过大。

第三节　细胞神经网络

Hopfield 网络要求每个神经元与其他神经元全联接，可是在真实神经网络里，并没有这种要求，在由 10^{11} 数量的神经细胞组成的人脑中，每个神经细胞只与其周围的 10^3 左右的神经细胞相联，在视觉初级加工的神经网络中，每个神经细胞与其相近的神经细胞之间的联接较强，而远离该神经细胞的联接权弱，视觉处理就是利用这种联接权的方向进行方向检测、边缘提取等工作，CNN(Cellular Neural Nefwork)网络就是以神经细胞的这种联接方式为背景，来实现一种局部联接的、权可设计的人工神经网络。这种网络对二维图像的初级加工特别有用，现已形成了一个新的学科分支。它的实现也比 Hopfield 网络容易，网络的芯片也已不断出现，这是一个值得注意的领域。

一、CNN 网络的模型

CNN 网络的基本单元称为人工细胞，它是由线性电容、线性电阻、线性控制元件和非线性控制元件组成，如同一个细胞自动机，它只同它周围的神经元相接，这是一个连续的动态系统，组成二维的形式如图 3-25 所示。用 $C(i,j)$ 或 C_{ij} 来表示第 i 行、第 j 列的神经元，$C(i,j)$ 只与 $C(i+1,j)$，$C(i-1,j)$，$C(i,j+1)$，$C(i,j-1)$，$C(i+1,j+1)$，$C(i+1,j-1)$，$C(i-1,j+1)$，$C(i-1,j-1)$ 相联，而与其他神经元不相联，这样 n 个神经元组成的网络联接权为 $9n$ 个。用 $N_r(i,j)$ 表示 $C(i,j)$ 神经元的邻近其他神经元的集合，在一个 $m\times n$ 的二维神经元排列空间内：

$$N_r(i,j)=\{C(k,l)/\max[|k-i|,|j-l|]\leqslant r\} \qquad (3\text{-}89)$$
$$1\leqslant k\leqslant m;\ 1\leqslant l\leqslant n$$

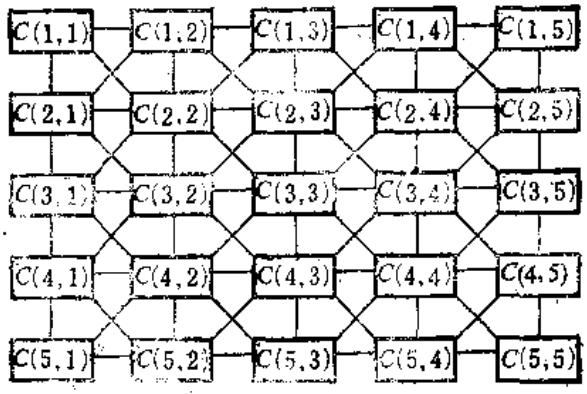


图 3-25　CNN 网络的结构

CNN 网络中每个神经元与周围 r 范围内的神经元相联，$r=1$，表示一个神经元除了与本身相连外，还与周围 8 个其他神经元相接，$r=2$ 表示与周围 24 个神经元相联，如图 3-26 所示。$r=l$，则与 $(2l+1)^2-1$ 个其他神经元相联，这里首先把这些相邻神经元定义为对称型的，即如果 $C(i,j)\in N_r(k,l)$，那么 $C(k,l)\in N_r(i,j)$ 对所有的 $C(i,j)$、$C(k,l)$ 都成立。对于一

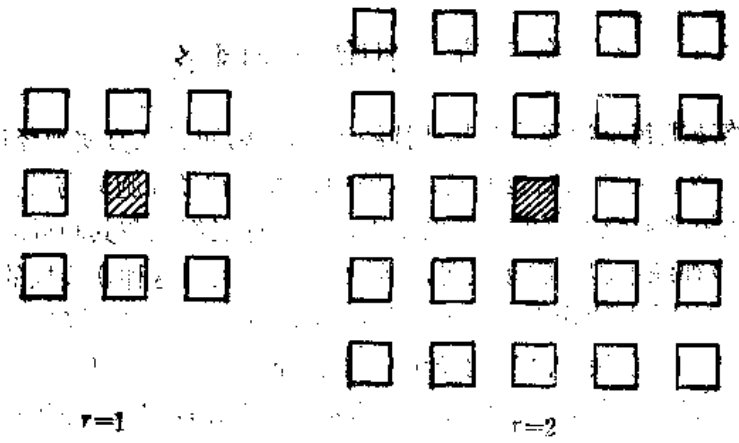


图 3-26　CNN 网络中 $N_r(i,j)$的范围

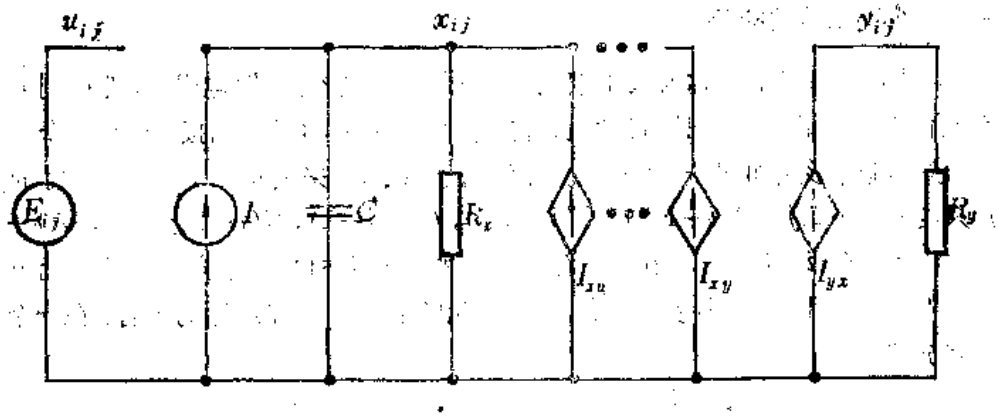


图 3-27　单个神经元的等效电路

个 $C(i,j)$，我们用一个等效电路来描述，这个等效电路可以用运算放大器来实现。图 3-27 是一个 $C(i,j)$ 的等效电路，u_{ij} 表示输入，用一个独立的电压源 E_{ij} 来表示，如果 $C(i,j)$ 有外界输入时，则 $E_{ij}\neq 0$，x_{ij} 表示神经元的状态，它是由电容 C 上的电压决定的，与邻近神经元的关系则由 $2P$ 个线性电压控制电流源来决定，如果邻近神经元 $C(k,l)$ 的输入为 u_{kl}，输出为 y_{kl}，那么 I_{xy}、I_{xu} 可表示为

$$\begin{aligned} I_{xy}(i,j;k,l) &= A(i,j;k,l)y_{kl} \\ I_{xu}(i,j;k,l) &= B(i,j;k,l)u_{kl} \end{aligned} \tag{3-90}$$

$A(i,j;k,l)$表示第 $C(k,l)$的输出与 $C(i,j)$之间的联接权，$B(i,j;k,l)$表示第 $C(k,l)$的输入与 $C(i,j)$之间的联接权，它们可以通过在电容上并接一个压控电流源来实现，如邻近有关联的神经元数为 P 个，$P=(2l+1)^2$，那么压控电源有 $2P$ 个。设由 $A(i,j;k,l)$组成的矩阵为 $\boldsymbol{A}$，由 $B(i,j;k,l)$组成的矩阵为 $\boldsymbol{B}$，它们由 $(2r+1)^2$ 个元组成，而每个神经元与相邻神经元之间的联接都是由 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 中的元所决定的。R_x、R_y 是线性电阻，I 为独立的电压源，R_x、R_y 的存在是由于电路实现时的阻抗决定的，一般是一个常数。y_{ij} 为 $C(i,j)$的输出，它是由一个非线性电压控制电流源组成 $I_{yx}=\left(\frac{1}{R_y}\right)f(x_{ij})$，$f(x_{ij})$的特性如图 3-28 所示，它是一个分段线性函数，满足下面的公式：

$$f(x_{ij})=\frac{1}{2}[|x_{ij}+1|-|x_{ij}-1|]$$

$$f(x_{ij})=\begin{cases}1, & x_{ij}\geqslant 1\\ -1, & x_{ij}\leqslant -1\\ x_{ij}, & -1<x_{ij}<1\end{cases} \tag{3-91}$$

$$y_{ij}=R_yI_{yx}=f(x_{ij})$$

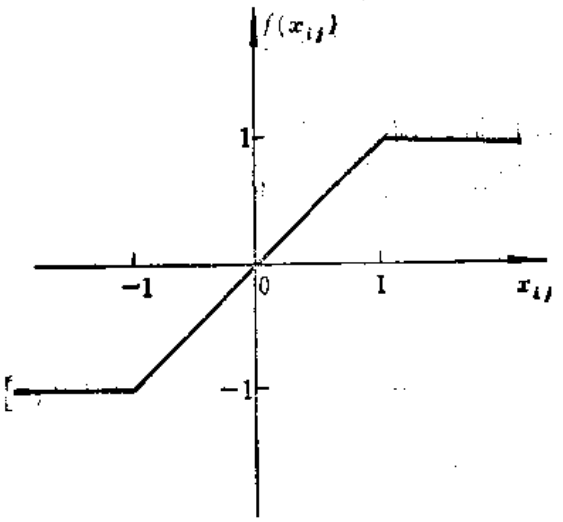


图 3-28　分段线性的 $f(x_{ij})$

对于图 3-27 的电路，利用克希霍夫电流和电压定律，得到每个人工细胞单元满足下列的状态方程：

$$C\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=-\frac{1}{R_x}x_{ij}(t)+\sum_{C(k,l)\in Nr(i,j)}A(i,j;k,l)y_{kl}(t)+\sum_{C(k,l)\in Nr(i,j)}B(i,j;k,l)u_{kl}+I \tag{3-92}$$

式中　$1\leqslant i\leqslant m$; $1\leqslant j\leqslant n$; $C>0$; $R_x>0$

输入方程为

$$u_{ij}=E_{ij},\ 1\leqslant i\leqslant m,\ 1\leqslant j\leqslant n \tag{3-93a}$$

约束条件为

$$|x_{ij}(0)|\leqslant 1,\ 1\leqslant i\leqslant m;\ 1\leqslant j\leqslant n \tag{3-93b}$$

$$|u_{ij}|\leqslant 1,\ 1\leqslant i\leqslant m;\ 1\leqslant j\leqslant n \tag{3-93c}$$

系统的对称条件为

$$A(i,j;k,l)=A(k,l;i,j) \tag{3-93d}$$

CNN 网络就是按这些单个神经元的模型直接联接而成的单层的动态网络，CNN网络模型有如下特点：

(1) 所有人工细胞都是由图 3-27 形式的电路组成，每个细胞的电路都是相同的。

(2) 每个细胞只与 $(2r+1)^2$ 个相邻的细胞相联，每个细胞都接受自己和邻近其他单元的反馈信号，反馈的多少由 A、B 两个矩阵决定，A、B 矩阵称为模板。

(3) 所有细胞的输入、输出关系是非线性单调上升函数，在 CNN 网络中采用了分段线性来描述，如公式(3-91)。

(4) 网络由一组非线性的状态方程来描写，即式(3-91)式(3-92)，是一个动态的系统。

二、CNN 网络系统的分析

1. CNN 网络的状态 $x_{ij}(t)$ 的有界性

网络的状态 $x_{ij}(t)$ 在动态过程中是有界的，它的最大值为

$$|x_{max}|=1+R_x|I|+R_x\max\Big[\sum_{C(k,l)\in Nr(i,j)}|A(i,j;k,l)|+|B(i,j;k,l|\Big]$$
$$1\leqslant i\leqslant m,\ 1\leqslant j\leqslant n \tag{3-94}$$

证明：

先将公式(3-92)写成下列的状态方程式：

$$\frac{dx_{ij}(t)}{dt}=\frac{-1}{R_xC}x_{ij}(t)+f_{ij}(t)+qb_{ij}(u)+\hat{I} \tag{3-95}$$

其中

$$f_{ij}(t)=\frac{1}{C}\sum_{C(k,l)\in Nr(i,j)}A(i,j;k,l)y_{kl}(t)$$

$$qb_{ij}(u)=\frac{1}{C}\sum_{C(k,l)\in Nr(i,j)}B(i,j;k,l)u_{kl}$$

$$\hat{I}=\frac{I}{C}$$

$$1\leqslant i\leqslant m,\ 1\leqslant j\leqslant n$$

式(3-95)的解为

$$x_{ij}(t)=x_{ij}(0)e^{-\frac{t}{R_xC}}+\int_0^t e^{-(t-\tau)/R_xC}[f_{ij}(\tau)+qb_{ij}(u)+\hat{I}]d\tau \tag{3-96}$$

在(3-96)式两边取绝对值可得

$$|x_{ij}(t)|\leqslant|x_{ij}(0)e^{-t/R_xC}|+\Big|\int_0^t e^{-(t-\tau)/R_xC}[f_{ij}(\tau)+qb_{ij}(u)+\hat{I}]d\tau\Big|$$

$$\leqslant|x_{ij}(0)||e^{-t/R_xC}|+\int_0^t e^{-(t-\tau)/R_xC}[|f_{ij}(\tau)|+|qb_{ij}(u)|+|\hat{I}|]d\tau$$

$$\leqslant|x_{ij}(0)|+R_xC[F_{ij}+QB_{ij}+|\hat{I}|]$$

$$F_{ij}=\max|f_{ij}(t)|$$

$$\leqslant\frac{1}{C}\sum_{C(k,l)\in Nr(i,j)}|A(i,j;k,l)|\max|y_{kl}(t)|$$

$$=\frac{1}{C}\sum_{C(k,l)\in Nr(i,j)}|A(i,j;k,l)|$$

$$QB_{ij}=\max|qb_{ij}(u)|$$

$$\leqslant\frac{1}{C}\sum_{C(k,l)\in Nr(i,j)}|B(i,j;k,l)|\max|u_{kl}|$$

$$=\frac{1}{C}\Sigma|B(i,j;k,l)|$$

利用(3-93a、b、c)得到

$$|x_{ij}(t)|\leqslant1+R_x\Big[\sum_{C(k,l)\in Nr(i,j)}|A(i,j;k,l)|+|B(i,j;k,l|+|\hat{I}|\Big]=x_{max}$$

证毕。

2. 网络的能量函数 E

定义一个能量函数 E 为 CNN 系统的能量函数

$$E(t)=-\frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}A(i,j;k,l)y_{ij}(t)y_{kl}(t)+\frac{1}{2R_x}\sum_{(i,j)}y_{ij}^2(t)$$
$$-\sum_{(i,j)}\sum_{(k,l)}B(i,j;k,l)y_{ij}(t)u_{kl}-\sum_{(i,j)}Iy_{ij}(t) \tag{3-97}$$

这个能量函数反映了图 3-27 类型的电路的能量，它与 u、y 有关，这是电路中的电压，虽然它不反映状态变量 $x_{ij}(t)$，但在 $y_{ij}(t)$、$y_{kl}(t)$ 中包含了 $x_{ij}(t)$ 的信息，这个量是可以收敛到一个最小点的。用式(3-97)的能量函数来分析 CNN 网络，若要求网络收敛到一个稳定点，即能量最小点，其条件是模板 A 矩阵对称，$A(i,j;k,l)=A(k,l;i,j)$。

按照能量函数的定义，如果 E 有界，并且随着时间 t 增加 E 单调下降，则 CNN 系统是稳定的。以下我们分别证明 $E(t)$ 是有界的，以及在模板对称情况下 E 随时间单调下降。

(1) $E(t)$ 的有界性，即

$$\max|E|\leqslant E_{\max};\text{ 且 }E_{\max}\text{ 满足}$$
$$E_{\max}=\frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}|A(i,j;k,l)|+\sum_{(i,j)}\sum_{(k,l)}|B(i,j;k,l)|$$
$$+m\cdot n\left(\frac{1}{2R_x}+|I|\right) \tag{3-98}$$

对于 $m\times n$ 个神经元构成的网络成立。

证明： 将(3-97)式两边取绝对值得

$$|E(t)|\leqslant\frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}|A(i,j;k,l)||y_{ij}(t)||y_{kl}(t)|$$
$$+\frac{1}{2R_x}\sum_{(i,j)}y_{ij}^2(t)+\sum_{(i,j)}\sum_{(k,l)}|B(i,j;k,l)||y_{ij}(t)||u_{kl}|$$
$$+\sum_{(i,j)}|I||y_{ij}(t)|$$

由(3-91)式 $|y_{kl}(t)|\leqslant 1$，$|y_{ij}(t)|\leqslant 1$，取 $|y_{kl}(t)|=1$，$|y_{ij}(t)|=1$；且 $u_{kl}=1$

可得

$$|E(t)|\leqslant\frac{1}{2}\sum_{(i,j)}\sum_{(k,l)}|A(i,j;k,l)|+m\cdot n\frac{1}{2R_x}$$
$$+\sum_{(i,j)}\sum_{(k,l)}|B(i,j;k,l)|+m\cdot n|I|$$

所以

$$|E(t)|\leqslant E_{\max}$$

证毕。

(2) 当 $A(i,j;k,l)=A(k,l;i,j)$，$\frac{dE(t)}{dt}\leqslant 0$

证明：

$$\frac{dE(t)}{dt}=-\sum_{(i,j)}\sum_{(k,l)}A(i,j;k,l)\frac{\partial y_{ij}}{\partial x_{ij}}\frac{dx_{ij}}{dt}y_{kl}$$
$$+\frac{1}{R_x}\sum_{(i,j)}\frac{\partial y_{ij}}{\partial x_{ij}}\frac{dx_{ij}}{dt}y_{ij}-\sum_{(i,j)}\sum_{(k,l)}B(i,j;k,l)\frac{\partial y_{ij}}{\partial x_{ij}}\frac{dx_{ij}}{dt}u_{kl}$$
$$-\sum_{(i,j)}I\frac{\partial y_{ij}}{\partial x_{ij}}\frac{dx_{ij}}{dt}=-\sum_{ij}\frac{\partial y_{ij}}{\partial x_{ij}}\frac{dx_{ij}}{dt}\Big[\sum_{c(k,l)\in Nr(i,j)}A(i,j;k,l)y_{kl}$$

$$-\frac{1}{R_x}x_{ij}+\sum_{C(k,l)\in Nr(i,j)}B(i,j;k,l)u_{kl}+I\Big]$$

$$=-\sum_{(i,j)}C\left(\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}\right)^2\left(\frac{\partial y_{ij}}{\partial x_{ij}}\right)$$

由图 3-28 得

$$\frac{\partial y_{ij}}{\partial x_{ij}}=\begin{cases}1 & |x_{ij}|<1\\ 0 & |x_{ij}|\geqslant 1\end{cases}$$

所以 $$\frac{\mathrm{d}E(t)}{\mathrm{d}t}\leqslant 0$$

因此 CNN 网络在模板 A 对称情况下是稳定的。

3. 单个 CNN 神经元的稳定点

将(3-92)式改写为

$$C\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=f'[x_{ij}(t)]+g(t) \tag{3-99}$$

其中：
$$f'[x_{ij}(t)]=\frac{1}{2}A(i,j;i,j)\left(|x_{ij}(t)+1|-|x_{ij}(t)-1|\right)-\frac{1}{R_x}x_{ij}(t)$$

$$g(t)=\sum_{\substack{C(k,l)\in Nr(i,j)\\ C(k,l)\neq C(i,j)}}[A(i,j;k,l)y_{kl}(t)+B(i,j;k,l)u_{kl}]+I$$

我们首先分析在 $A(i,j;i,j)>\frac{1}{R_x}$ 时的情况，并设 $g(t)=0$，$f'[x_{ij}(t)]$ 与 x_{ij} 的关系曲线如图 3-29 所示(是带有箭头的折线)，因为 $g(t)=0$，因此 $\frac{1}{C}f'[x_{ij}(t)]=\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}$，令 $C=1$，$A(i,j;i,j)=2$，$R_x=1$，得到图 3-29 的曲线。$\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}=0$ 时，共有三个平衡点：A、B、C，对于 A、C 两点，是稳定平衡点。初值在 0～2 之间时，$\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}>0$，而 $x_{ij}(t)>0$，这样轨

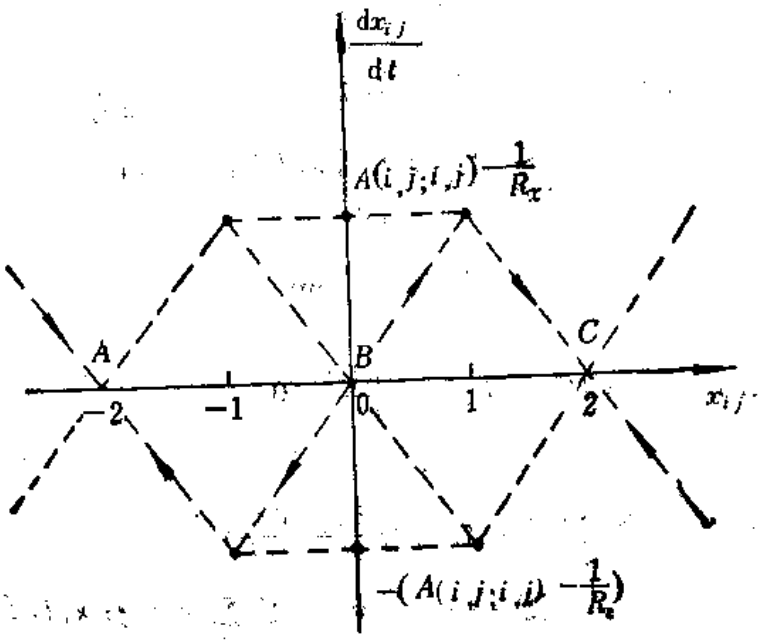


图 3-29 $g(t)=0$ 时，$\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}$ 与 x_{ij} 的关系

迹可沿着图 3-29 的箭头直达 C 点，而 $x_{ij}(t)>2$ 时，因为 $\frac{dx_{ij}(t)}{dt}<0$，而使 $x_{ij}(t)$ 向着 C 点靠近，因此 C 是稳定平衡点，而 A 也是稳定平衡点，当初值为：$-2<x_{ij}(0)<0$，则 $\frac{dx_{ij}(t)}{dt}<0$，$x_{ij}(t)$ 向减小的方向运动，直达 A 点。而 B 点是不稳定的平衡点，在 A、C 两点上 $|x_{ij}(t)|>1$，因此 $y_{ij}=\pm 1$。因此要使 CNN 网络最后稳定到 $y_{ij}(t)=\pm 1$ 的那些点，必须满足

$$A(i,j;i,j)>\frac{1}{R_x} \tag{3-100}$$

当 $g(t)\neq 0$，则图 3-29 的曲线会向上，或向下平移，因此其他神经元对 i,j 神经元的影响主要体现在 $g(t)$ 的移动上，图 3-30a、b 表示在加入 $g(t)$ 时，图 2-29 曲线移动的情况，$g(t)$ 的移动在小范围内对 A、B、C 平衡点没有多大影响，如图 3-30b 中，B 点仍为不稳定点，A、C 仍为稳定点，而输出对应为：$y_{ij}(t)=\pm 1$，如果 $g(t)$ 的移动范围大，如图 3-30a 所示，此时 B、C 平衡点不存在，只有 A 是一个稳定点，$y_{ij}(t)=-1$。

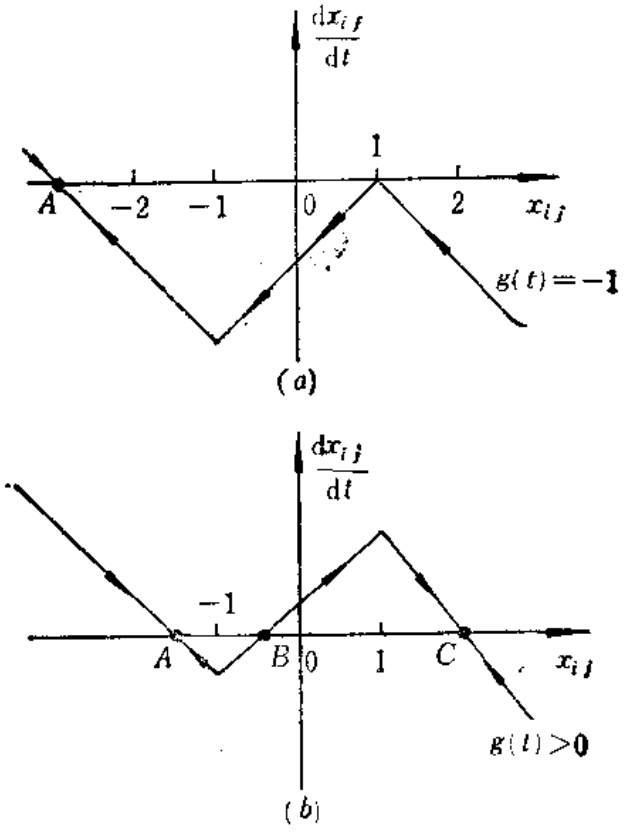


图 3-30　$g(t)\neq 0$ 时，$\frac{dx_{ij}}{dt}$ 与 x_{ij} 的关系

如果 $A(i,j;i,j)<\frac{1}{R_x}$，$g(t)=0$ 时，则稳定点为 B 点，如图 3-29 中无箭头的虚线所示，即 $y_{ij}(t)=0$，一般来说，这不是我们所期望的，此时，系统不会稳定到 A、C 点，当然在 $g(t)$ 加入后，B 点也在移动，$y_{ij}(t)=x_{ij}(t)\neq 0$，也可以通过输入 I 的调节，使网络达到稳定点 $y_{ij}(t)=\pm 1$，这里我们主要讨论了 $A(i,j;i,j)>\frac{1}{R_x}$ 的情况。

三、非对称模板条件下的系统稳定性

从上面的分析看，CNN 网络的稳定性要求 A 模板对称，应该指出，这种对称性与前面 CHNN 网络中所讨论的权矩阵对称是不完全相同的，例如他们的对称元就不相同。将方程(3-92)写成矩阵形式，令 $R_x=1, C=1$

$$\dot{X}(t)=-X(t)+A'Y(t)+B'u+I \tag{3-101}$$

其中 A' 是有 $(m\times n)^2$ 个元组成的方阵，它与模板 A 有直接联系，A' 对角线上的元为 A 的中心元素，$A(i,j;i,j)$ 用 A_{∞} 表示，A' 的非对角线上的元素是 A 中非中心素组成，B' 也是由 B 组成，其组成方式与 A' 相同。在 A 不对称的情况下，满足一定条件的 CNN 网络的系统也可以是稳定的，下面讨论一个比较特殊的非对称 A 的系统，看一看它的稳定性如何。系统的模板 A 的形式为

$$A=\begin{bmatrix} 0 & S & 0 \\ 0 & Q & 0 \\ 0 & -S & 0 \end{bmatrix};\quad A=\begin{bmatrix} 0 & 0 & 0 \\ S & Q & -S \\ 0 & 0 & 0 \end{bmatrix};$$

$$A=\begin{bmatrix} S & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & -S \end{bmatrix};\quad A=\begin{bmatrix} 0 & 0 & -S \\ 0 & Q & 0 \\ S & 0 & 0 \end{bmatrix}$$

它是以矩阵中心反对称的，甚至可以得到某些行或列上不对称，例如在 A 中某一行为 $S\quad Q\quad -r, S\neq r, Sr>0$，此时系统仍然可以达到稳定。首先考虑(3-93$b$)的条件，系统状态初值是处于 $|x_{ij}(0)|<1,\ 1<i<n,\ 1<j<m$，此时，$X(t)=Y(t)$，系统处于线性状态下(3-101)为线性方程，它的雅可比矩阵为

$$J=-U+A'P \tag{3-102}$$

U 是一个单位矩阵，P 是一个正对角矩阵，它的对角元素为 $\frac{\partial y_{ij}}{\partial x_{ij}}>0$，因此雅可比行列式的性质是由 A' 决定的，把式(3-101)改写为

$$\dot{X}(t)=-UX(t)+A'Y(t)+b;\quad b=B'u+I$$

在特殊模板的情况下，如 A 为

$$A=\begin{bmatrix} 0 & 0 & 0 \\ S & Q & -S \\ 0 & 0 & 0 \end{bmatrix};$$

则

$$A'=\begin{bmatrix} Q & S & 0 & 0 & \cdots \\ S & Q & -S & 0 & \cdots \\ 0 & S & Q & -S & \cdots \\ 0 & 0 & S & Q & \cdots \\ \cdots & & \cdots & & Q \quad -S \\ \cdots & & \cdots & & S \quad Q \end{bmatrix}$$

考虑到线性条件 $\boldsymbol{X}=\boldsymbol{Y}$，有

$$\dot{\boldsymbol{X}}(t)=(\boldsymbol{A}'-\boldsymbol{U})\boldsymbol{X}(t)+\boldsymbol{b} \tag{3-103}$$

$(\boldsymbol{A}'-\boldsymbol{U})$是一个与 $\boldsymbol{A}'$ 相同的矩阵，除了在对角线项 Q 改为 $Q-1=q$ 外，其他元都是相同的。根据(3-100)的要求，$A_{00}>\frac{1}{R_x}$，这里 $A_{00}=Q$，$\frac{1}{R_x}=1$，这样 $q=(A_{00}-1)>0$，在上述模板 $\boldsymbol{A}$ 的情况下，$(\boldsymbol{A}'-\boldsymbol{U})$矩阵是一个正定矩阵：

$$(\boldsymbol{A}'-\boldsymbol{U})=\begin{bmatrix} q & -S & 0 & \cdot & \cdot & \cdot \\ S & q & -S & \cdot & \cdot & \cdot \\ 0 & S & q & \cdot & \cdot & \cdot \\ & & & q & -S & \\ & & & S & q & \end{bmatrix}$$

它具有子行列式：

$$\det^1=q>0;\quad \det^2=q^2+S^2>0$$

$$\det^3=q\det^2+S^2\det^1>0$$

依次类推，对第 k 阶行列式($k>0$)也是正的：

$$\det^{k+1}=q\det^k+S^2\det^{k-1}>0$$

同理，在不对称的情况下，若 $\boldsymbol{A}=\begin{bmatrix} 0 & 0 & 0 \\ r & Q & -S \\ 0 & 0 & 0 \end{bmatrix}$，$Q>1$，$r>0, S>0$，也同样满足。当$(\boldsymbol{A}'-\boldsymbol{U})$为正定阵时，其所有特征值大于零。设一个由 $m\times n$ 个神经元组成的 CNN 网络，具有相反符号的模板 $\boldsymbol{A}$ 矩阵，而且它们每个神经元是用一个分段线性的输入、输出函数来描写，我们把神经元工作的线性区内称为 α 区，而非线性区为 β 区，那么 $m\times n$ 个动态方程满足(3-93b、c)条件下，有下列几种情况：

(1) 如果所有的神经元工作在线性区 α 区内，在任何一个 $t=t_i$ 的时刻与此系统关联的雅可比矩阵所有的特征值有正实部，因此所有的状态解向 $+\infty$ 处发散，在 α 区内没有稳定点。

(2) 如果所有的神经元工作在 β 区内，那么所有的特征值为 -1，因此他们稳定于一个正常数，也就是这个区域的稳定点，根据上面分析，只要 $A(i,j;i,j)>\frac{1}{R_x}$，那么，当 $t\to\infty$ 时，$|x_{ij}|>1$ 总能达到，在 $|x_{ij}|>1$ 的情况下，由于所有的特征值为 -1，因而一定稳定在 β 区域。

(3) 如果有 l 个量在 β 区域内，而余下来的在 α 区域内，那么 β 区内的那些变量变成了常数，而不落入雅可比行列式中，例如在下面的情况下，第 k 个神经网络的输出落到 β 区，那么(3-103)式可写为

$$\dot{\boldsymbol{X}}(t)=\boldsymbol{A}^{\alpha\beta}\boldsymbol{X}(t)+\boldsymbol{b}^k \tag{3-104}$$

式中

$$A^{\alpha\beta}=\begin{array}{c} \\ k-1 \\ k \\ k+1 \end{array}\begin{array}{c} \begin{array}{cccccccc} k-1 & k & k+1 \end{array} \\ \left[\begin{array}{cccccccc|c|ccccccc} q & -s & 0 & 0 & \cdots & 0 & 0 & 0 & & & & & & & \\ s & q & -s & 0 & \cdots & 0 & 0 & 0 & & & & & & & \\ 0 & s & q & -s & \cdots & 0 & 0 & 0 & & & & 0 & & & \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & & & & & & \\ 0 & 0 & 0 & 0 & \cdots & s & q & 0 & & & & & & & \\ \hline 0 & 0 & 0 & 0 & \cdots & 0 & s & -1 & -s & 0 & & & & & \\ \hline & & & & & & & 0 & q & -s & 0 & 0 & \cdots & 0 & 0 \\ & & & & & & & 0 & s & q & -s & 0 & \cdots & 0 & 0 \\ & & & 0 & & & & 0 & 0 & s & q & -s & \cdots & 0 & 0 \\ & & & & & & & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ & & & & & & & 0 & 0 & 0 & 0 & 0 & \cdots & s & q \end{array}\right] \end{array}\begin{array}{c} \\ \\ \\ \\ \\ \\ \\ 0 \\ 0 \\ \\ \\ \\ \\ \\ \end{array}$$

$$b^k=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{k-1}-sh \\ b_k+Qh \\ b_{k+1}+sh \\ b_{k+2} \\ \vdots \\ b_{m\times n} \end{bmatrix}$$

其中 h 为第 k 个神经元的输出 h 为 ± 1。

对于 $A^{\alpha\beta}$，可以分为两个小矩阵 A_1，A_2 与一个单行组合而成，可写为

$$A^{\alpha\beta}=\begin{array}{c} \\ k \\ \\ \end{array}\begin{array}{c} \begin{array}{ccc} & k & \end{array} \\ \left[\begin{array}{ccc} A_1 & & 0 \\ \hline & s \quad -1 \quad -s & \\ \hline 0 & & A_2 \end{array}\right] \end{array}$$

其中 $A_1\in R^{(k-1)\times(k-1)}$，$A_2\in R^{(mn-k)\times(mn-k)}$，$A_1$ 和 A_2 满足与 $(A'-U)$ 相同的正定条件，(3-104)式的特征方程为

$\det(\lambda U-A^{\alpha\beta})=\det(\lambda U_{k-1}-A_1)(\lambda+1)\det(\lambda U_{mn-k}-A_2)$ U_{k-1} 是 $(k-1)\times(k-1)$ 的单位矩阵，U_{mn-k} 是 $(mn-k)\times(mn-k)$ 的单位矩阵，因此，对于没有达到非线性区(β 区)的其他神经元的状态，按 $(A'-U)$ 正定的条件向非线性区运动，对于已达到 β 区的第 k 个神经元，在 b_k 的第 k 个分量上加一个 Qh 量，如 $h=1$，那么相当于在状态变量 $\dot{x}_{ij}(t)$ 上加一个正量，使 $\dot{x}_{ij}(t)$ 的值增加，如图 3-30b 所示的情况，此时稳定点 C 的稳定域加大，因而在 $y_k=1$ 的情况下，使 k 的输出更加稳定，相反若 $h=-1$，$Qh<0$，使图 3-30 的曲线向下，如图 3-30a 所示，此时 A 点的稳定域增加，使输出保持 $y_k=-1$。因此对这种非对称网络，系统仍然可以达到稳定。

对于这种特殊的非对称模板的分析方法，也可以推广到其他非对称模板。

四、网络权的设计

用一个 CNN 网络来检测一个目标的运动情况，根据动物视网膜的实验，细胞对运动物体的方向和速度都有检测作用。图 3-31 表示了一个实验的结果。图中第一列 (a) 表示在 t 时刻物体的初始状态，第二列 (b) 表示在 $t+\Delta t$ 时刻物体运动的位置：在(1) 的情况下，t 和 $t+\Delta t$ 时刻的位置相同，物体没有运动；在(2)的情况下，$t+\Delta t$ 时刻物体产生了运动，运动方式为平移，移动一个像素；在(3)的情况下，物体虽然动运了，但速度过快，细胞不能检测出来；在(4)的情况下，物体虽然运动了，但不是平移，而是向下向右移动了一个像素；除了第二行 (2) 能检测出物体运动以外，其他都不能检测出来。第三列 (c) 表示最后检测的结果。这个现象可用 CNN 网络来完成其测试，将图 3-31a 作为 CNN 网络的输入量，将图 3-31b 作为 CNN 网络的状态量，采用设计 $\boldsymbol{A}$、$\boldsymbol{B}$ 和 I 的关系方程，来达到其输出量为图 3-31c 的目的。我们采用的模板 $\boldsymbol{A}$、$\boldsymbol{B}$ 和输入量 I 为

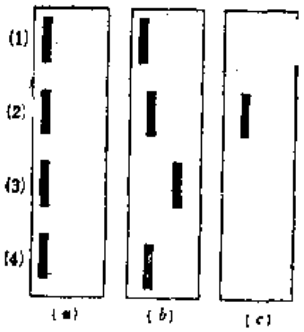


图 3-31　物体运动检测示意图

$$\boldsymbol{A}=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0.3 & 0.3 & 0.3 & 0 \\ 0 & 0.3 & 4.2 & 0.3 & 0 \\ 0 & 0.3 & 0.3 & 0.3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{B}=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -0.3 & -0.3 & -0.3 & 0 & 0 \\ -0.3 & 3.1 & -0.3 & 0 & 0 \\ -0.3 & -0.3 & -0.3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$I=-6.0$$

我们假设：图 3-31 中的黑色为 $+1$，白色为 -1，物体长度为三个像素，宽度为一个像素，利用式(3-92)，考虑图中(2)的情况：将图中(a)作为输入，当 $\boldsymbol{B}$ 矩阵与输入 u_{kl} 相乘时，只有 $\boldsymbol{B}$ 中第二列与 $+1$ 相乘，而其他都与 -1 相乘。将图中(b)作为$x_{kl}(t_0)$，$C(k,l)$，$\in N_r(i,j)$ 考虑其输出，在 $\boldsymbol{A}$ 矩阵，只有第三列与 $+1$ 相乘，而其他都与 -1 相乘，如果 $C(i,j)$是物体中间的那个细胞，利用(3-92)式，令电容 $C=1$，电阻 $R_x=1$；有

$$\begin{aligned}\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t} &= -x_{ij}(t)+\sum_{C(k,l)\in Nr(i,j)} A(i,j;k,l)y_{kl}(t)+\sum_{C(k,l)\in Nr(i,j)} B(i,j;k,l)u_{kl}+I \\ &= -1+6\times 0.3\times(-1)+2\times 0.3\times(+1)+4.2\times(+1) \\ &\quad +6\times(-0.3)\times(-1)+2\times(-0.3)\times(1)+3.1\times 1+(-6.0)=+0.3>0\end{aligned}$$

因为 $\dfrac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}>0$，则 $y_{ij}(t)$在初值 $x_{ij}(0)=1$ 的条件下，$\Delta x_{ij}(t)>0$，则得 $y_{ij}(t)=1$。

对于图 3-31 中的(1)、(3)、(4)的情况，代入(3-92)式，都使 $\dfrac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}<0$，使状态向减少的方向移动，有时初始值 $x_{ij}(t_0)=1$，也会使其减少，直到 $y_{ij}(t)=-1$ 为止。从 $\boldsymbol{A}$ 矩阵来看，由于 $A(i,j;i,j)>\dfrac{1}{R_x}$，稳定点必然在 $y_{ij}(t)=\pm 1$ 上，适当选择 $\boldsymbol{A}$、$\boldsymbol{B}$、I 的参数，可以使输出达到图 3-31 所要求的情况，完成物体速度和方向检测。对于这些参数的设计我们先考虑(3-99)式的形式，仍设 $R_x=1$，$C=1$。

在(3-99)式中，将 $g(t)$ 写成一个有上下限的形式，对每个神经元 $C(i,j)$ 则有

$$g_{ij}(t)=\sum_{\substack{C(k,l)\in Nr(i,j)\\ C(k,l)\neq C(i,j)}} A(i,j;k,l)y_{kl}(t)+\sum_{C(k,l)\in Nr(i,j)} B(i,j;k,l)u_{kl}+I$$

在系统的状态变化中 $g_{ij}(t)$ 的最大值为 $g_{ij}^{+}(t)$，最小值为 $g_{ij}^{-}(t)$，十分明显，I、$B(i,j;k,l)$、$A(i,j;k,l)$、u_{kl} 都是确定的，$g_{ij}^{+}(t)$ 和 $g_{ij}^{-}(t)$ 是与周围的神经元 $y_{kl}(t)$ 的输出有关的，因此 $g_{ij}^{+}(t_0)$ 与 $g_{ij}^{-}(t_0)$ 是一个与 t 无关的量。在(3-99)式中：

$$\begin{aligned} f'(x_{ij}) &= A(i,j;i,j)y_{ij}(t)-x_{ij}(t)\\ &= A_{00}y_{ij}(t)-x_{ij}(t) \end{aligned}$$

还是用前面的图 3-29 来表示，当 $A_{00}-1>0$，那么对于神经元 $C(i,j)$ 其稳定点为 A 和 C，$g_{ij}(t)$ 的加入使图 3-29 的曲线向上或向下平移，因而改变了它的吸引域的大小，使系统达到某一个稳定点，因此如果设计者希望其输出保证能达到要求，若 t_0 为初始时刻，而且 $t_0\geqslant 0$，那么设计可按下面的原则进行：

(1) 如果满足 $(A_{00}-1)y_{ij}(t_0)+g_{ij}^{+}(t_0)<0$，即 $g_{ij}^{+}(t_0)$ 为一个很负的数，如图 3-30a 所示，当 $t\geqslant T$ 时，则

$$y_{ij}(t)=-1$$

这时 T 对于不同的初始条件下是不同的。

$$\left.\begin{array}{lll} \text{若} & y_{ij}(t_0)=-1, & \text{则}\quad T=t_0\\ \text{若} & |y_{ij}(t_0)|<1, & \text{则}\quad T=t_1\\ \text{若} & y_{ij}(t_0)=1, & \text{则}\quad T=t_2 \end{array}\right\} \tag{3-105}$$

这说明 $g_{ij}^{+}(t_0)$ 加入后，自然能够使 $\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}$ 与 x_{ij} 的曲线在坐标轴 x_{ij} 以下，如图 3-30a 所示，此时不管 $y_{ij}(t_0)$ 初始在什么位置上，都可以通过一定的时间 T，到达稳定点 A，只是 T 的时间不同，$t_0<t_1<t_2$。

(2) 如果满足 $(A_{00}-1)y_{ij}(t_0)+g_{ij}^{-}(t_0)>0$，$g_{ij}^{-}(t_0)$ 是一个大的正数，如图 3-30b 所示，当 $t\geqslant T$ 时，则

$$y_{ij}(t)=+1$$

同样可得

$$\left.\begin{array}{ll} y_{ij}(t_0)=1, & T=t_0\\ |y_{ij}(t_0)|<1, & T=t_1\\ y_{ij}(t_0)=-1, & T=t_3 \end{array}\right\} \tag{3-106}$$

根据以上的原则，并根据问题的要求，设计 $\boldsymbol{A}$、$\boldsymbol{B}$、I 的方法如下：

(1) 对所需解决的问题定出一系列规则；

(2) 给出模板 $\boldsymbol{A}$、$\boldsymbol{B}$ 的基本形式；

(3) 对每一条规则，让其满足上面两个原则(3-105)式和(3-106)式；

(4) 得到一组不等式组，并解出其 $\boldsymbol{A}$、$\boldsymbol{B}$ 模板的各个系数；

(5) 计算机模拟，验证。

虽然上面的设计 CNN 的方法、原则不是唯一的，但不少问题可以根据图 3-29、图 3-30 来决定一些设计规则和所要求的参数。

下面给出图 3-31 中物体运动检测的例子，说明 $\boldsymbol{A}$、$\boldsymbol{B}$、I 的设计。图中黑色代表 1，白色代表 -1，以 i 为行，j 为列，以图 3-31b 作为网络的初值用 $x_{ij}(t_0)$ 来表示，图 3-31c 作为网

络的输出值用 $y_{ij}(t)$ 表示，图 3-31*a* 为网络的输入值，一般是不变的，用 u_{ij} 表示。在初始时刻，因为 CNN 的条件为式(3-93*b*)，$|x_{ij}(t_0)|\leqslant 1$，因此 $y_{ij}(t_0)=x_{ij}(t_0)$，根据图 3-31 中(1)，(2)，(3)，(4)四种情况，可以总结出如下规则：

(1) 在时刻 t_0，$x_{ij}(t_0)=y_{ij}(t_0)=-1$，那么在任何 $t>t_0$ 时刻，$y_{ij}(t)=-1$，因为在图 3-31*b* 中 $x_{ij}(t_0)=-1$ 的地方，其输出为图 3-31*c*，$y_{ij}(t)$ 必为 -1。

(2) 在时刻 t_0，$x_{ij}(t_0)=1$，而 $u_{i,j-1}=-1$，此时表示物体以错误的速度平移，因而其输出 $y_{ij}(t)=-1$。

(3) $x_{ij}(t_0)=1$，对 $i=i_1,i_1+1,i_1+2$ 成立，而 $u_{i',j-1}=1$，对 $i'=i_1+1,i_1+2,i_1+3$(或 $i'=i_1-1,i_1,i_1+1$)成立，$i'\neq i$，即在中间 $C(i,j)$ 神经元周围的 $C(i-1,j)$，$C(i+1,j)$，有一个元 $x_{i-1,j}(t_0)\neq u_{i-1,j-1}$，这表明物体移动方向发生了偏离，不是严格的平移，此时输出也应为 $y_{ij}(t)=-1,y_{i-1,j}(t)=-1,y_{i+1,j}(t)=-1$。

(4) $x_{ij}(t_0)=1,u_{ij-1}=1$，对于每一个 $C(k,l)\in N_r(i,j)$，都满足 $x_{kl}(t_0)=1$ 时，$u_{k,l-1}=1$，这说明目标移动的位置正确，则有 $y_{ij}(t)=1$。

在设计中，首先考虑在运动物体检测的 CNN 网络中，相邻像素对中心像素具有相同的作用。设中心像素为 $C(i,j)$，并且设与 $C(i,j)$ 相邻像素的权是相同的，即

$$A(i,j;k,l)=A_n,\quad \forall i\neq k,\quad j\neq l$$

成立，同时对于输入模板 $\boldsymbol{B}$，也是与中心点对称的，只是与模板 $\boldsymbol{A}$ 的位置比较，向左移了一列，即

$$B(i,j-1;k,l-1)=B_n$$

从图 3-31*c* 看，$C(i,j)$的输出 $y_{ij}(t)$ 与 $u_{i,j-1}$ 和 $x_{ij}(t_0)$有关，与 $x_{ij}(t_0)$ 周围的神经元无关，这样，当 $u_{i,j-1}=1$，$x_{ij}(t_0)=1$ 时，必有

$$A(i,j;k,l)y_{kl}+B(i,j-1;k,l-1)u_{k,l-1}=0$$

$$C(k,l)\neq C(i,j)\qquad C(k,l)\in N_r(i,j)$$

$$A_n=-B_n$$

因此，可得 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 的形式为

$$\boldsymbol{A}=\begin{bmatrix}0&0&0&0&0\\0&A_n&A_n&A_n&0\\0&A_n&A_{00}&A_n&0\\0&A_n&A_n&A_n&0\\0&0&0&0&0\end{bmatrix}\qquad \boldsymbol{B}=\begin{bmatrix}0&0&0&0&0\\-A_n&-A_n&-A_n&0&0\\-A_n&B_{0-1}&-A_n&0&0\\-A_n&-A_n&-A_n&0&0\\0&0&0&0&0\end{bmatrix}$$

从公式(3-99)中得

$$g_{ij}(t)=A_n\sum_{\substack{C(k,l)\in N_r(i,j)\\C(k,l)\neq C(i,j)}}[y_{kl}(t)-u_{k,l-1}]+B_{0,-1}u_{i,j-1}+I \tag{3-107}$$

假定 $A_n>0$，利用规则(1)～(4)和原则(1)和(2)，找出 A_{00}，A_n，$B_{0,-1}$ 和 I，则可完成 CNN 网络的设计。

根据规则(1)，在某一时刻 t_0，$t_0\geqslant 0$，如 $x_{ij}(t_0)=-1$，则无论 t 为何值，$y_{ij}(t)=-1$，根据(3-107)式，得

$$g_{ij}^{+}(t_0)=16A_n+|B_{0,-1}|+I$$

在 $t\geqslant t_0$ 时，$y_{ij}(t)=-1$，由原则(1)得

$$\frac{dx_{ij}}{dt}=+1-A_{00}+16A_n+|B_{0,-1}|+I\leqslant 0$$

使用规则(2)，当 $x_{ij}(t_0)=1$，而 $u_{i,j-1}=-1$，则 $y_{ij}(t)=-1$，此时 $g_{ij}^{+}(t)$ 根据(3-107)式，可得

$$g_{ij}^{+}(t_0)=16A_n-B_{0,-1}+I$$

根据原则(1)得

$$\frac{dx_{ij}}{dt}=A_{00}-1+16A_n-B_{0,-1}+I<0$$

$t>t_0$ 时，满足上式。

根据规则(3)，$x_{ij}(t_0)=1$，$u_{i,j-1}=1$，仍有至少一个邻近元 $C(k,l)$ 有 $u_{m_1,n_1-1}=1$，而 $x_{m_1,n_1}(t_0)=-1$，即 $y_{m_1,n_1}(t_0)=-1$，则对 $t>t_0$ 时，有 $y_{ij}(t)=-1$。利用(3-107)式，有

$$g_{ij}(t_0)=A_n\sum_{\substack{C(k,l)\in N_r(i,j)\\ C(k,l)\neq C(i,j)\\ C(k,l)\neq C(m,n)}}[y_{kl}(t_0)-u_{k,l-1}]+A_n[y_{m_1,n_1}(t_0)-u_{m_1,n_1-1}]+B_{0,-1}+I$$

$y_{m_1,n_1}(t)$ 为 $C(k,l)$ 中的一个神经元，它有 $u_{k,l-1}=1$，$x_{k,l}(t_0)=-1$，而其他的 $C(k,l)$ 细胞都满足 $u_{k,l-1}=1$，$x(t_0)=1$ 则

$$g_{ij}^{+}(t)=-2A_n+B_{0,-1}+I$$

根据规则(1)得

$$\frac{dx_{ij}}{dt}=(A_{00}-1)-2A_n+B_{0,-1}+I<0$$

根据规则(4)，$x_{ij}(t_0)=u_{i,j-1}=1$，对所有 $C(k,l)\in N_r(i,j)$ 都满足，得到

$$g_{ij}^{-}(t)=A_n\Sigma 0+B_{0,-1}u_{i,j-1}+I=B_{0,-1}+I$$

按原则(2)得到

$$\frac{dx_{ij}}{dt}=(A_{00}-1)+B_{0,-1}+I>0$$

综合上述 n 个规则，得到

$$\begin{aligned}
&A_{00}-1-16A_n-B_{0,-1}-I>0\\
&1-A_{00}-16A_n+B_{0,-1}-I>0\\
&1-A_{00}+2A_n-B_{0,-1}-I>0\\
&A_{00}-1+B_{0,-1}+I>0\\
&A_{00}-1>0\\
&A_n>0
\end{aligned}$$

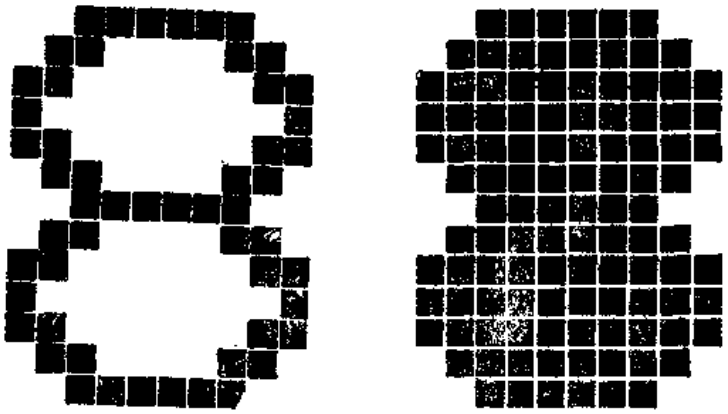
解上述不等式，得到 $A_n=0.3$；$B_{0,-1}=3.1$；$A_{00}=4.2$，$I=-6.0$，用计算机验证所获得的结果与设计相符。

五、CNN 网络的应用举例

1. 空洞滤波器(Hole-Filter)

这种 CNN 网络的目的是将一个二值文字中的空洞全部填满，如图 3-32*a* 的一个数字 8，通过 CNN 网络把中间部分的空洞补满，形成图 3-32*b* 的形式。

为了达到所要求的结果，我们假设 $R_x=1$，$C=1$，图像为二值 ± 1，将图 3-32*a* 作为两维的输入图像加入到 CNN 的输入端，$\boldsymbol{u}=\{u_{ij}\}$，$1\leqslant i\leqslant m$；$1\leqslant j\leqslant n$，取模板 $\boldsymbol{A}$、$\boldsymbol{B}$、I 的参数为



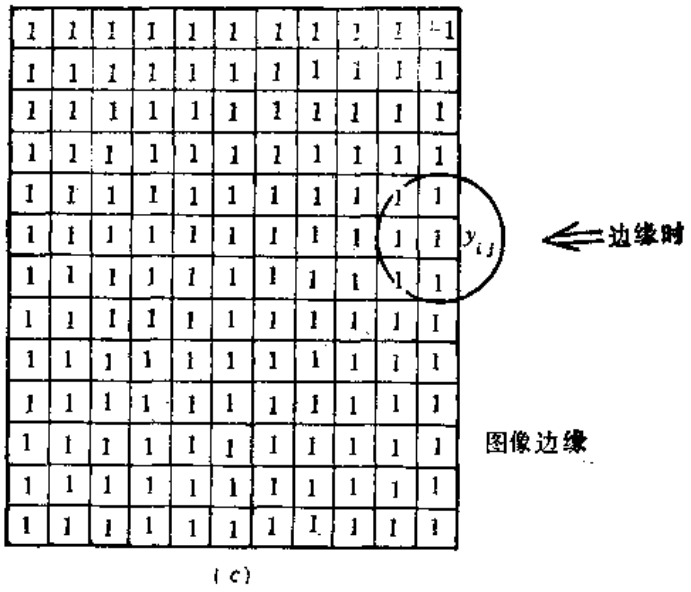


图 3-32　滤波前后的图像

(a) 滤波前；　(b) 滤波后；　(c) 初始值$y_{ij}(t_0)=1$

$$\boldsymbol{A}=\begin{bmatrix}0&1&0\\1&2&1\\0&1&0\end{bmatrix};\qquad \boldsymbol{B}=\begin{bmatrix}0&0&0\\0&b&0\\0&0&0\end{bmatrix}=b=4I=-1$$

并设所有的初值 $x_{ij}(t_0)=1$，根据(3-92b)式的条件，在初始时所有的 $y_{ij}(t_0)=1$，如图3-82c所示，$1\leqslant i\leqslant m$；$1\leqslant j\leqslant n$。

利用(3-92)式，状态方程可写成

$$C\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=-\frac{1}{R_x}x_{ij}(t)+\boldsymbol{A}*y_{ij}(t)+bu_{ij}+I \tag{3-108}$$

式中"*"表示为卷积。

$$\boldsymbol{A}*y_{ij}(t)=\sum_{C(kl)\in N_r(i,j)}A(i,j;k,l)y_{kl}$$

下面，我们分三种情况讨论。

(1) 当 $u_{ij}=-1$ 时，即图 3-32a 中图像为白色的点。在 $y_{i,j+1}(t_0)$，$y_{i,j-1}(t_0)$，$y_{i-1,j}(t_0)$ $y_{i+1,j}(t_0)$ 不全为 1 的情况下，则

$$\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}<0$$

图像上原为白色的，选代后仍为白色，$y_{ij}(t)=-1$。

由于所有初始值设定为：$x_{ij}(t_0)=1$，$y_{ij}(t_0)=x_{ij}(t_0)=1$，方程(3-108)可写成

$$C\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=-x_{ij}(t)+y_{i-1,j}(t)+y_{i,j-1}(t)+2y_{ij}(t)+y_{i,j+1}(t)$$
$$+y_{i+1,j}(t)+4u_{i,j-1} \qquad (3\text{-}109a)$$

用初始值 $y_{ij}(t_0)=1$，$u_{ij}=-1$ 代入，上式右边为

$$-x_{ij}(t_0)+y_{i-1,j}(t_0)+y_{i,j-1}(t_0)+2y_{ij}(t_0)+y_{i,j+1}(t_0)$$
$$+y_{i+1,j}(t_0)-4-1=0$$

因此，只有 $y_{i,j+1}(t_0)$，$y_{i,j-1}(t_0)$，$y_{i+1,j}(t_0)$，$y_{i-1,j}(t_0)$ 中不完全为 1 时，才能保持 $\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}<0$，显然，只有在图像的边缘上才满足条件。如图 3-32c 表示的那些边缘上的 y_{ij} 点，图像为 $n\times m$ 个像素，只有 $i=1$，或 $i=n$，及 $j=1$ 或 $j=m$ 的那些点，在一开始满足 $\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}<0$，空洞滤波器从图像的边缘开始搜索，因此，边缘上 $y_{ij}(t)=-1$，一旦 $y_{ij}(t)=-1$ 以后，那么靠着边缘的点又不满足 $y_{i,j+1}(t)$，$y_{i,j-1}(t)$，$y_{i-1,j}(t)$，$y_{i+1,j}(t)$ 全为 1 的条件，这样新的 $y_{ij}(t)$，又可为 -1，一直到输入图像 $u_{ij}(t)=1$ 为止。

(2) 当 $u_{ij}=1$，不管什么条件下，都能保证 $\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}\geqslant 0$，当 $u_{ij}=1$ 时，式 (3-108) 的右边可写成：

$$4+y_{i-1,j}(t)+y_{i+1,j}(t)+y_{i,j-1}(t)+y_{i,j+1}(t)+2y_{ij}(t)>0$$

(3) 当 $u_{ij}=-1$，即图像中间的空洞部分，因为此时，$y_{i-1,j}(t)$，$y_{i+1,j}(t)$，$y_{i,j+1}(t)$，$y_{i,j-1}(t)$ 都为 1，因此(3-108)式的右边为 0，$C\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=0$，而原先假设 $y_{ij}(t_0)=1$，状态不变，因此在 $y_{ij}(t)$ 的点上仍然保持为 1，这样在 $u_{ij}=1$ 包围的内部的 -1 点被填上 $y_{ij}(t)=1$，因此，空洞就被填满了。我们先从边缘的神经元考虑，继而向中间扩展，就像波浪一样随时间传播，直到 $u_{ij}(t)=1$ 时为止，而 $u_{ij}(t)=1$ 以内的 $y_{ij}(t_0)$ 则原样保持。这样，结果如图 3-32b 所示。空洞滤波器的模板 A 是对称的，因此，也必定能达到稳定的状态，而不会出现振荡现象。

2. 相连单元检测器(Connected Component Detector)

在文字识别中，特征的选择往往是识别的重要手段，如文字与垂直线、水平线相交的次数，在经典的模式识别中，还将文字与一些特殊方向直线相交的次数作为文字识别的特征，但由于经典处理是串行进行的，它只能考虑 n 根线的相交。如果用特殊设计的CNN网络，则可以把文字与多条平行线相交的情况映照到另一个图像上，此图像较原始图像更容易表示各种不同样本的特征，因而可以区分不同的样本。

在 CNN 网络的公式(3-92)中，若采用模板为

$$\boldsymbol{A}=\begin{bmatrix}0 & 0 & 0\\ 1 & 2 & -1\\ 0 & 0 & 0\end{bmatrix};\ \boldsymbol{B}=0;\ I=0$$

因为 $\boldsymbol{A}$ 模板是反对称的，根据上面讨论反对称权条件下稳定性分析，$\boldsymbol{A}$ 模板组成的 CNN 网络具有稳定性，并考虑初始值 $x_{ij}(t_0)$ 为要处理的图形，这样公式(3-92)简化为只与同一行的相邻神经元有关，写成

$$C\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}=\frac{1}{-R_x}x_{ij}+y_{i,j-1}+2y_{ij}-y_{i,j+1}\quad i=1,2,\cdots n \tag{3-109b}$$

采用上述公式对一行信号进行处理，如图 3-33 所示，图 3-33a 为处理前的情况，黑色为 $+1$，白色为 -1，图 3-33b 为经过 CNN 网络后的情况。它的特点是，最终状态黑白相间数与最初状态的黑白相间数相等，但它把图形压缩到左面，同时，在边缘上的神经元其初始状态在 CNN 网络动态处理后保持不变。对于图 3-33a 中第一行(1)的两个边缘神经元其初始状态为 -1，那么终态 b，两个边缘神经元也为 -1，而在初始图形(a)中，凡是黑白间隔超过 1 个像素的，都在终态(b)中，压缩到间隔只有一个的状态。

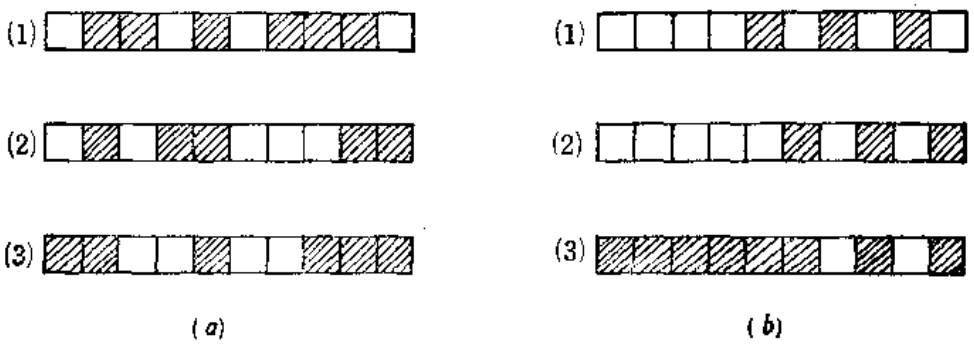


图 3-33 检测前后的图形
(a) 检测前；(b) 检测后

在式(3-109b)中，令 $R_x=1, C=1$，分析下面几种情况：

(1) 如果从左到右的一行图像像素中，初始 $x_{i,j+1}(t_0)$，$x_{i,j-1}(t_0)$ 都是相同的值，则 $y_{i,j-1}(t_0)=y_{i,j+1}(t_0)$ 例如为 $+1$，那么(3-109b)式为

$$\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=-x_{ij}(t_0)+2y_{ij}(t_0)=1>0\quad (x_{ij}(t_0)=1, y_{ij}(t_0)=1)$$

得到 $y_{ij}(t)=1$，将保持原来的状态。

同理 $x_{ij}(t)=x_{i,j-1}(t)=x_{i,j+1}(t)=-1$，其结果：

$$y_{ij}(t)=-1$$

(2) 如果 $x_{i,j-1}(t_0)=-1, x_{ij}(t_0)=+1, x_{i,j+1}(t_0)=+1$，在初始的输出 $y_{i,j-1}(t_0)=-1$，$y_{ij}(t_0)=+1, y_{i,j+1}(t_0)=+1$，那么由(3-109$b$)式得

$$\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=-x_{ij}(t_0)+0<0$$

由于 $y_{ij}(t_0)>0$；而 $\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}<0$，则 $y_{ij}(t)=-1$，使其输出向反方向改变，原来为 $+1$，现在变为 -1。

同样，$y_{i,j-1}(t_0)=1; y_{ij}(t_0)=-1; y_{i,j+1}(t_0)=-1;$

则得 $$y_{ij}(t)=1$$

(3) 如果 $x_{i,j-1}(t_0)=-1$; $x_{ij}(t_0)=+1$; $x_{i,j+1}(t_0)=-1$, 则有

$$\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=-x_{ij}(t_0)+2y_{ij}(t)>0$$

则保持原状态，因为 $\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}$ 的方向与 $x_{ij}(t)$ 一致。

相联单元检测器可以采用以下四种模板:

$$\boldsymbol{A}_1=\begin{bmatrix}0&0&0\\1&2&-1\\0&0&0\end{bmatrix}\quad \boldsymbol{A}_2=\begin{bmatrix}0&1&0\\0&2&0\\0&-1&0\end{bmatrix}$$

$$\boldsymbol{A}_3=\begin{bmatrix}1&0&0\\0&2&0\\0&0&-1\end{bmatrix}\quad \boldsymbol{A}_4=\begin{bmatrix}0&0&-1\\0&2&0\\1&0&0\end{bmatrix}$$

得到的图为四幅不同方向的压缩图形，如图 3-34 示。图中为不同方向对字母 "5" 的压缩结果。

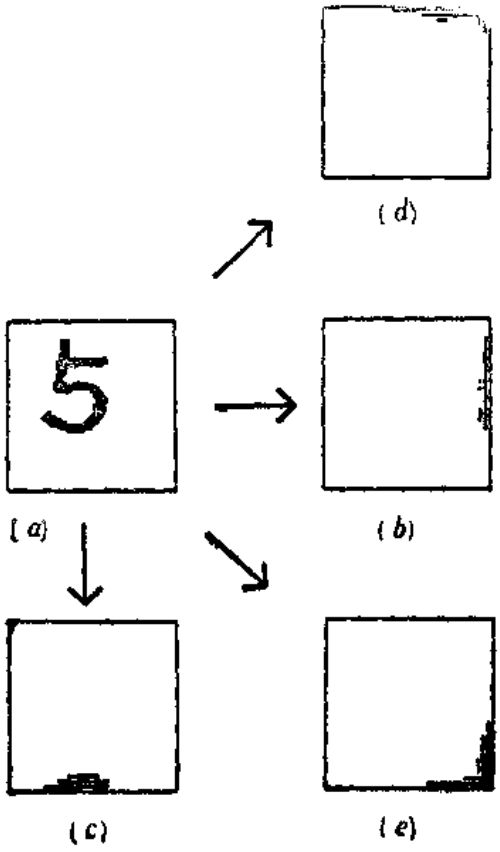


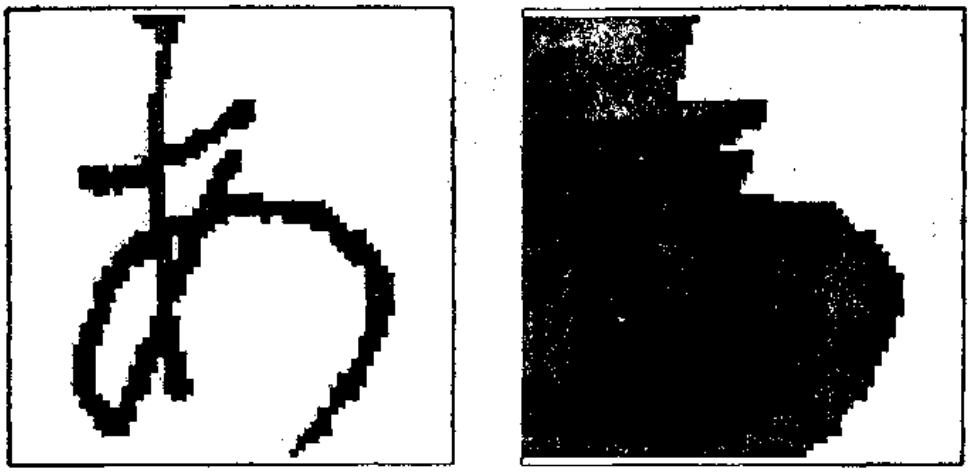
图 3-34 CNN 网络的输出

使用 $\boldsymbol{A}_1$ 模板得到了图 3-34*b* 的结果，图 3-34*c* 则是使用 $\boldsymbol{A}_2$ 模板的结果，图 3-34*d* 为使用 $\boldsymbol{A}_4$ 模板的结果，图 3-34*e* 则为使用 $\boldsymbol{A}_3$ 模板的结果，用图 3-34*b*、*c*、*d*、*e* 作为特征，可得到比较好的识别结果。

3. 阴影检测器(Shadow Detector)

阴影检测是从某一个方向上，如右面将字母的边缘形状突出，而另一个方向上则用阴影涂黑，它像阳光从一个方向上射向字母上，在背阳面则留下阴影一样。从文字识别看，将字母在某一个方向上，如右面的轮廓突出，可以按最右面为黑的位置来判断字母的类别，图3-35为一个日文字母"あ"字及CNN网络处理的结果，采用模板为

$$\boldsymbol{A}=\begin{bmatrix}0&0&0\\0&2&2\\0&0&0\end{bmatrix};\quad \boldsymbol{B}=\begin{bmatrix}0&0&0\\0&2&0\\0&0&0\end{bmatrix};\quad \boldsymbol{I}=\boldsymbol{0}$$



(a)　　(b)

图 3-35　阴影CNN处理

(a) 处理前；(b) 处理后

其状态方程在 $R_x=1, C=1$ 时为

$$\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}=-x_{ij}+2y_{ij}+2y_{i,j+1}+2u_{ij} \tag{3-110}$$

此时，$\boldsymbol{A}$ 模板不是对称模板，但同样可以证明其系统的状态方程雅可比行列式是正定的，同样可以得到上述不对称模板 A 情况下系统仍然可以达到稳定的结论。

对于(3-110)式，设网络的初值为 $x_{ij}(t_0)=1$，$\forall i,j$ 成立。图形由 u_{ij} 输入，在图形上黑色的像素 $u_{ij}=1$，白色的像素 $u_{ij}=-1$，先从右面开始讨论。

(1) 从图 3-35a 的最右面开始，此时图像最边缘的是 u_{ij}，如果它是白色的，即有 $u_{ij}=-1$；$y_{i,j+1}(t_0)=0$；$y_{ij}(t_0)=1$；$x_{ij}(t_0)=1$，则(3-110)式为

$$\frac{\mathrm{d}x_{ij}(t)}{\mathrm{d}t}=-1+2-2<0$$

得到

$$y_{ij}(t)=-1$$

因此从右面开始 $u_{ij}=-1$ 的地方保持其为 -1。

(2) 在 $u_{ij}=-1$；$y_{i,j+1}(t_0)=-1$ 的情况下，这是在最右面的神经元开始计算后，移动次右面一列得到的，在 $y_{ij}(t_0)=1$；$x_{ij}(t_0)=1$ 的初值条件下，式(3-110)可得

$$\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}=-1+2-2-2<0, y_{ij}(t)=-1$$

如此从最右面开始，向左扩张，一直保持其输出 $y_{ij}(t)=-1$ 直到达到 $u_{ij}=1$ 的情况为止。

(3) 在 $u_{ij}=1$；$y_{ij}(t_0)=1$；$x_{ij}(t_0)=1$；$y_{i,j+1}(t_0)=-1$ 的情况下，由式 (3-110) 得到

$$\frac{dx_{ij}}{dt}=-1+2-2+2>0;\ y_{ij}(t)=1$$

此时 $y_{ij}(t)$ 的输出与 u_{ij} 相同。

(4) 在 $u_{ij}=-1$; $y_{ij}(t_0)=1$; $x_{ij}(t_0)=1$; $y_{i,j+1}(t_0)=1$ 的情况下，此为在图 3-35*a* 黑色笔画的左面，由(3-110)式得

$$\frac{dx_{ij}}{dt}=-1+2+2-2>0;\ y_{ij}(t)=1$$

此时在黑色笔画的左面全部变黑；如图 3-35*b* 所示，因而，由(3-110)式表示的 CNN 网络可以达到阴影检测的作用。

4. 用于细化的 CNN 网络

"细化"是在字母识别中常用的抽取骨架的方法，去除那些多余的 +1 像素，得到字母的骨架，它又保留了字母中的主要信息量。如图 3-36 所示，这是将一个比较粗的中文字母，细化为骨架，图 3-36*b* 为其结果。它是采用下面八个 5×5 的不对称模板组成 CNN 网络，同时作用于图像上进行细化的，其八个模板分别为

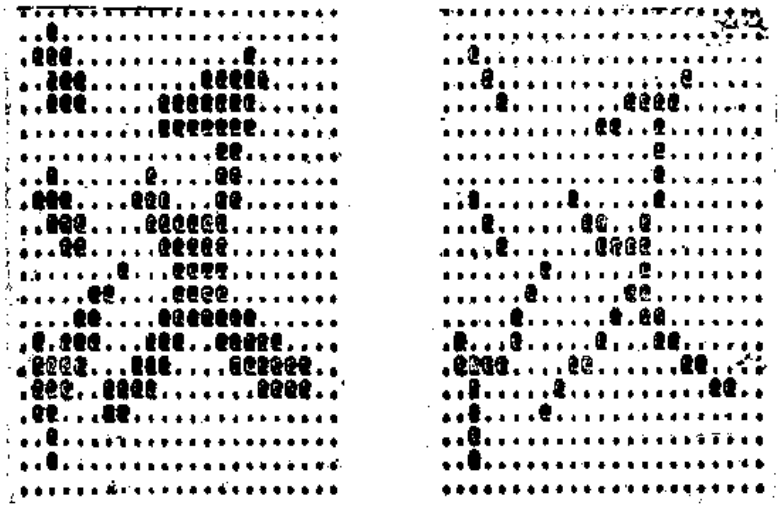


图 3-36 图像细化的结果

(*a*) 细化前； (*b*) 细化后

$$\boldsymbol{A}_1=\begin{bmatrix}0&0&0&0&0\\0&1&1&0&0\\0&1&5&-1&0\\0&0&-1&0&0\\0&0&0&0&0\end{bmatrix};\quad \boldsymbol{A}_2=\begin{bmatrix}0&0&0&0&0\\0&1&2&1&0\\0&0&9&0&0\\0&-1&-2&-1&0\\0&0&0&0&0\end{bmatrix};$$

$$\boldsymbol{A}_3=\begin{bmatrix}0&0&0&0&0\\0&0&1&1&0\\-1&-1&6&1&0\\0&0&-1&0&0\\0&0&0&0&0\end{bmatrix};\quad \boldsymbol{A}_4=\begin{bmatrix}0&0&0&0&0\\0&-1&1&1&0\\1&-2&11&2&0\\0&-1&1&1&0\\0&0&0&0&0\end{bmatrix};$$

$$A_5=\begin{bmatrix}0&-1&-1&0&0\\0&0&-1&0&0\\-1&-1&8&1&0\\0&0&1&1&0\\0&0&0&0&0\end{bmatrix};\quad A_6=\begin{bmatrix}0&0&1&0&0\\0&-1&-2&-1&0\\0&1&11&1&0\\0&1&2&1&0\\0&0&0&0&0\end{bmatrix}$$

$$A_7=\begin{bmatrix}0&0&-1&0&-1\\0&0&-1&-1&0\\0&1&9&-1&0\\0&1&1&0&0\\0&0&0&0&0\end{bmatrix};\quad A_8=\begin{bmatrix}0&0&0&0&0\\0&1&0&-1&0\\0&2&9&-2&0\\0&1&0&-1&0\\0&0&0&0&0\end{bmatrix}\quad I=-1\ \ B=0$$

此时(3-92)式可写为

$$C\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}=-\frac{x_{ij}}{R_x}+A_P*y_P+I \tag{3-111}$$

"*"表示卷积，A_P 为上列八个模板($P=1,2,\cdots,8$)，y_P 是与 A_P 有关的输出，其系统的特点为：

(1) 这八个模板分别组成八个 CNN 网络，可以同时对输入图像进行细化，观察每个模板，它们的中间系数 A_{00} 特别大，而模板非中心的系数用 $A_{i_1j_1}$ 表示，$A_{00}\geqslant\sum_{i_1j_1\neq 00}|A_{i_1j_1}|$，因此这八个模板分别形成的八个 CNN 系统的雅可比行列式，因为对角元的值 $A_{00}>1$，并且满足比其他元之和大，因此它们都是正定阵，在 $|x_{ij}|<1$ 的区域中没有有稳定点。

(2) 采用状态方程(3-111)式来处理图像，考虑每一个中心点 $C(i,j)$，因为相邻的神经元对 $C(i,j)$ 的联接权中正符号的系数与负符号的系数近似相同，因此 A_P 如果卷积一个全黑或全白的图像，其结果会保证中心点 $C(i,j)$ 的像素不变(此时 A_{00} 起主要作用)。

(3) 在(3-111)式中，设 $C=1,R_x=1$，中心联接权为 A_{00}，x_{ij} 的初值为 ±1，(3-111)式可写为

$$\frac{\mathrm{d}x_{ij}}{\mathrm{d}t}=(A_{00}-1)x_{ij}+g_{ij}(t)$$

$$g_{ij}^{+}(t)=\sum_{i_1j_1\neq 00}|A_{i_1j_1}|-1$$

$$g_{ij}^{-}(t)=-\sum_{i_1j_1\neq 00}|A_{i_1j_1}|-1$$

当 x_{ij} 初始值为 -1 时，

由于

$$A_{00}\geqslant\sum_{i_1j_1\neq 00}|A_{i_1j_1}|$$

所以

$$-(A_{00}-1)+g_{ij}^{+}(t)\leqslant 0$$

因此，在任何条件下都不能把 -1 改变到 $+1$ 上来。

当 x_{ij} 初始值为 $+1$ 时，对八个不同模板满足

$$A_{00}=\sum_{i_1j_1\neq 00}|A_{i_1j_1}|,$$

或者 $A_{00}=\sum_{i_1j_1\neq 00}|A_{i_1j_1}|+1$ 时，

$$\frac{dx_{ij}}{dt}=(A_{00}-1)+g_{\bar{i}\bar{j}}(t)=(A_{00}-1)-\sum_{i_1j_1\neq 00}|A_{i_1j_1}|-1<0$$

因而在 $g_{ij}(t)=g_{\bar{i}\bar{j}}(t)$，或 $g_{ij}(t)=g_{\bar{i}\bar{j}}(t)+1$ 的条件下，$C(i,j)$ 的状态可改变符号，其输出 y_{ij} 从 $+1\to-1$。

这样的过程与细化的要求是一致的，细化是消去那些多余的 $+1$ 像素，而绝不改变 -1 到 $+1$，因此 $y_{ij}(t)$ 的改变是单向的，只能从 $+1\to-1$，而不能从 $-1\to+1$，因此虽然模板是不对称的，但还能保证 CNN 网络是稳定的。

(4) 模板的形式是与细化要求消去的点有关，在模板设计中不仅根据细化的位置来设计，而且还要考虑优先级，还要防止图形被完全细化，这些设计考虑的规则比较复杂，这里不详细讨论了，有兴趣的读者可查阅有关的参考文献。[52]

第四章　自组织竞争人工神经网络

第一节　概　　述

在实际的神经网络中，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，这种侧抑制在脊髓和海马中存在，在人眼的视网膜中也存在。这种抑制使神经细胞之间出现竞争，一个兴奋最强的神经细胞对周围神经细胞的抑制也强，虽然一开始各个神经细胞都处于兴奋状态，但最后是那个输出最大的神经细胞"赢"了，而其周围的神经细胞"输"了。

另外，在认知过程中除了从教师那儿得到知识外，还有一种不需要教师指导的学习，例如：婴儿出生后，听到外界声音的刺激，他自然也会发出声，并自己在外界环境中学习抓东西、走路等。在动物中这种现象更为普遍。这种直接依靠外界刺激，"无师自通"达到的功能有时也称为自学习、自组织的学习方法。

自组织竞争人工神经网络就是基于上述两种生物结构和现象的基础上生成的，它的权是经过 Hebb 规则或类似 Hebb 规则学习后得到的。

图 4-1 是一个简单的自组织竞争人工神经网络的示意图，它是由多层的前馈网络组成，但在同一层内，神经元之间又有相互抑制、相互竞争的作用。

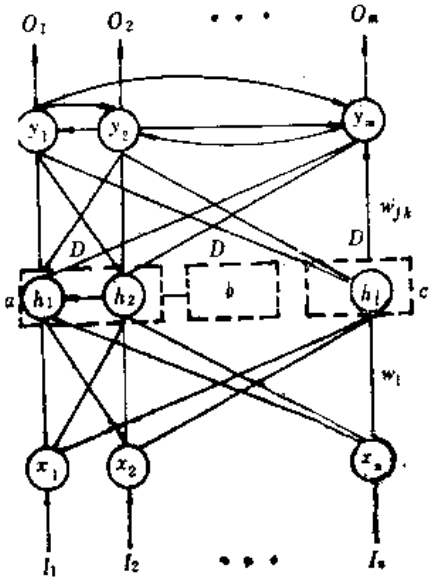


图 4-1　多层竞争网络

它的每个神经元的输入与输出也满足第一章中的(1-1)、(1-2)、(1-3)式。

我们令：$x \in R^n$ 为输入层神经元的状态；

$s \in R^n$ 为输入层神经元的输出；

$h \in R^l$ 为中间层神经元的状态；

$s' \in R^l$ 为中间层神经元的输出；

$y \in R^m$ 为输出层神经元的状态；

$o \in R^m$ 为输出层神经元的输出，每个神经元都满足：

对输入层：

$$\left.\begin{aligned} x_i &= I_i \\ s_i &= f(x_i) \end{aligned}\right\} \tag{4-1}$$

对中间层：

$$\left.\begin{aligned} h_j &= \sum_{i=1}^{n} w_{ij} x_i + g_j \\ s'_j &= f(h_j) \end{aligned}\right\} \tag{4-2}$$

对输出层：

$$\left.\begin{aligned} y_k &= \sum_{j=1} w_{jk} s'_j + g' \\ O_k &= f(y_k) \end{aligned}\right\} \tag{4-3}$$

其中 g 和 g' 都是同一层中周围其他神经元的输入，如果中间层中分为很小的区域，如图 4-1 中虚线所示，竞争是在小区域内进行，竞争结果，每一区域内最多只有一个神经元"赢"。

现在我们针对这种网络的两个特点作进一步的讨论。

一、竞争

在图 4-1 中我们考虑从第一层输入层到中间层的情况，假设 $s(x_i)=x_i=I_i$，则输入矢量 $\boldsymbol{I}=\boldsymbol{x}$。对于中间层的神经元，其状态矢量为 $\boldsymbol{h}\in R^l$，$\boldsymbol{h}$ 中每个分量 h_j，都接受输入层的信息，从第一层第 i 个神经元到中间层第 j 个神经元之间的权为 w_{ij}，对于一个分量 h_j 来说，它接受第一层的信息是通过权矢量 $\boldsymbol{m}_j$，$\boldsymbol{m}_j=(w_{1j}, w_{2j}, \cdots, w_{nj})^T$，如果输入为 $\boldsymbol{x}$ 则 h_j 得到从第一层来的信息为 $\boldsymbol{x}^T\boldsymbol{m}_j$。同时 h_j 又要接受同一层来的抑制信息，如果同一层神经元之间的联接权为 s_{Pj}，那么用公式来表示：

$$g_j=\sum_{P\in D} s_{Pj} s'_P(h_P) \tag{4-4}$$

这里 D 表示在图 4-1 上的一个小区域 D 内，如果 D 表示为整个第二层所有的神经元，那么竞争后，在第二层中只有一个神经元兴奋，如果 D 为一个区域，那么竞争后此区域内只有一个神经元兴奋，在第二层中有 k_1 个小区域，则可能有 k_1 个神经元兴奋。在(4-4)式中 s_{Pj} 为同层第 P 个神经元与第 j 个神经元之间的连接权。$s'_P(h_P)$ 为第二层第 P 个神经元的输出，$s'_P(\cdot)$ 是一个单调上升函数，其输出 s'_P 是 0 到 1 之间的模拟量 $s'_P\in[0,1]$。如果 s'_P 在竞争后"赢"了，则 $s'_P=1$，如果 s'_P 在竞争后"输"了，则 $s'_P=0$. 综合式(4-2)、(4-4)得

$$s'_j(h_j)=s'_j(\boldsymbol{x}^T\boldsymbol{m}_j+g_j)=D_j(\boldsymbol{x}) \tag{4-5}$$

这里 $D_j(x)$ 表示在 D 区域内第 j 个神经元的输出。在(4-4)中权 s_{Pj} 是对称的，$s_{Pj}=s_{jP}$，同时满足：$s_{11}=s_{22}=\cdots s_{PP}>0$；而不同神经元之间的权 $s_{Pj}<0$，$P\neq j$。如果在竞争中第 j 个神经元"赢"了，$s'_j(h_j)=1$，$g_j=s_{jj}>0$。如果在竞争中 $s'_j(h_j)$ 输了，则 $g_j<0$，$s_{lj}<0$，l 是周围"赢"了的那个神经元的标号。在没有输入信号时，因为 $\boldsymbol{x}^Tm_j=0$，$s'_j(h_j)$ 都是很小的，g_j 也几乎为 0，当网络输入 $\boldsymbol{x}$ 后，$\boldsymbol{x}^Tm_j\neq 0$，$s'_j(h_j)$ $(j=1, \cdots, l)$ 都产生一定的信号，此时，g_j 就起作用，通过(4-5)式达到竞争。

除了采用(4-5)式讨论竞争外，有时可以根据输入的情况直接讨论第二层神经元竞争的结果，其算法如下：

若 $s_i(x_i)=x_i$，则

$$s'_j(h_j)=\begin{cases} 1; & \|\boldsymbol{x}-\boldsymbol{m}_j\|=\min\limits_{k'}\|\boldsymbol{x}-\boldsymbol{m}_{k'}\| \\ 0; & \|\boldsymbol{x}-\boldsymbol{m}_j\|>\min\limits_{k'}\|\boldsymbol{x}-\boldsymbol{m}_{k'}\| \end{cases} \tag{4-6}$$

这里 $\|\boldsymbol{x}-\boldsymbol{m}_j\|$ 为欧氏距离；$\|\boldsymbol{Z}\|=Z_1^2+Z_2^2+\cdots Z_l^2=\boldsymbol{Z}^T\boldsymbol{Z}$

从上式看，如果 $\|\boldsymbol{x}-\boldsymbol{m}_j\|=\min\limits_{k'}\|\boldsymbol{x}-\boldsymbol{m}_{k'}\|$，则第 j 个神经元才会"赢"，这时可得到

$$(\boldsymbol{x}-\boldsymbol{m}_j)^T(\boldsymbol{x}-\boldsymbol{m}_j)=\min_{k'}(\boldsymbol{x}-\boldsymbol{m}_{k'})^T(\boldsymbol{x}-\boldsymbol{m}_{k'})$$

$$x^Tx+m_j^Tm_j-2x^Tm_j=\min_{k'}(x^Tx+m_{k'}^Tm_{k'}-2x^Tm_{k'})$$

如果 $|m_1|=|m_2|=\cdots|m_j|=|m_l|$，则有

$$-2x^Tm_j=2\min_{k'}(-x^Tm_{k'}) \quad k'=1,2,\cdots,l$$

所以

$$x^Tm_j=\max_{k'}(x^Tm_{k'}) \quad k'=1,2,\cdots,l \tag{4-7}$$

从(4-7)式可以看到，一个神经元要竞争到"赢"，那么其输入向量与该神经元相联的权向量的欧氏距离最小，即为 x 向量与 m_j 向量的内积最大。如果 $|x|=|m_j|=1$，则(4-7)式可化为

$$x^Tm_j=|x||m_j|\cos(x,m_j) \quad (|x|,|m_j| \text{ 为矢量的模}) \tag{4-8}$$

当

$$\cos(x,m_j)=1 \tag{4-9}$$

则(4-8)式为最大。

二、自组织网络的学习规律

自组织网络权有两类，一类是层与层之间的权，在图 4-1 中用 $\{w_{ij}\}$、$\{w'_{jk}\}$ 表示，这些权是可学习的。另一类是层内互相抑制的权 $\{s_{pj}\}$，一般讲这一类权是固定的，如满足一定的分布关系，距离近的抑制强，距离远的抑制弱，它是一种对称权。$\{w_{ij}\}$，$\{w'_{jk}\}$ 的学习是用 Hebb 学习律或竞争学习律得到的，若两个神经元输出兴奋，则他们之间的联接权增加，反之减少，按其学习公式来分，大致有以下几种学习规则：

1. 信号 Hebb 学习规则

$$\dot{w}_{ij}=-w_{ij}+s_i(x_i)s'_j(h_j) \tag{4-10}$$

w_{ij} 代表第 i 个输入神经元与第 j 个中间神经元之间的权，如果 $w_{ij}>0$ 为兴奋性权，$w_{ij}<0$ 为抑制性权，$s_i(x_i)$ 与 $s'_j(h_j)$ 为两个不同层上的神经元，$s_i(x_i)\in[0,1]$，$s'_j(h_j)\in[0,1]$，它们为 0 与 1 之间的模拟量，如果两个神经元都兴奋，那么 w_{ij} 就会增加，否则 w_{ij} 会减少，甚至达到遗忘。例如 $s'_j(h_j)=0$ 时，$\dot{w}_{ij}=-w_{ij}$，此时 w_{ij} 的值会减少，直到零为止。

2. 竞争学习规律

$$\dot{w}_{ij}=s'_j(h_j)[s_i(x_i)-w_{ij}]\forall i \tag{4-11}$$

如果 $s'_j(h_j)$ 是一个单调上升的函数，$s'_j(h_j)=\frac{1}{1+e^{-ch_j}}$，$c>0$，$s'_j\in[0,1]$，因此如果在某一时刻 t，s'_j "赢" 了，$s'_j(h_j)=1$，那么与第 j 个神经元相联的所有的权都发生变化，w_{ij} 的改变是使 $s_i(x_i)\approx w_{ij}$ $\forall i=1,2,\cdots,n$. 如果 $s_j(h_j)$ "输" 了；即 $s'_j(h_j)=0$，则 $w_{ij}=0$，权 w_{ij} 不进行调整。考虑一个特殊的情况：$s_i(x_i)=x_i$，那么式(4-11)就变为

$$\dot{w}_{ij}=s'_j(h_j)(x_i-w_{ij})$$

化为矢量形式：

$$\dot{m}_j=s'_j(h_j)(x-m_j) \tag{4-12}$$

3. 微分 Hebb 学习律

$$\dot{w}_{ij}=-w_{ij}+s_i(x_i)s'_j(h_j)+\dot{s}_i(x_i)\dot{s}'_j(h_j) \tag{4-13}$$

$\dot{s}_i$ 定义为

$$\frac{ds_i(x_i)}{dt}=\frac{\partial s_i}{\partial x_i}\frac{dx_i}{dt}=s_i^{(1)}\dot{x}_i$$

$s_i^{(1)}$ 表示 s_i 对 x_i 的偏导数。

比较 (4-10) 与 (4-13)，微分的 Hebb 规则是多了一个与状态变化率有关的量 $\dot{s}_i(x_i)\cdot \dot{s}'_j(h_j)$，因为 $\dot{s}_i$ 与 $\dot{s}'_j$ 有正有负，它改变了权 w_{ij} 的变化情况，$\dot{s}_i$ 与 $\dot{s}'_j$ 大，即状态变化速率大，则权的改变也大，当 $\dot{s}_i=0$，或 $\dot{s}'_j=0$ 则 (4-13) 式与 (4-10) 式相同。

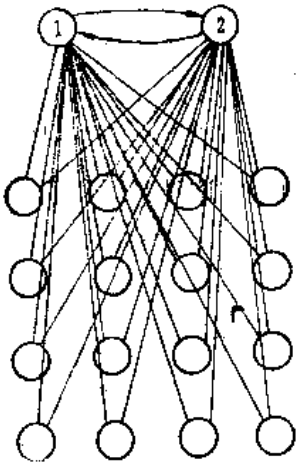


图 4-2　两层自组织网络

4. 微分竞争学习规律

$$\dot{w}_{ij}=\dot{s}'_j(h_j)[s_i(x_i)-w_{ij}] \tag{4-14}$$

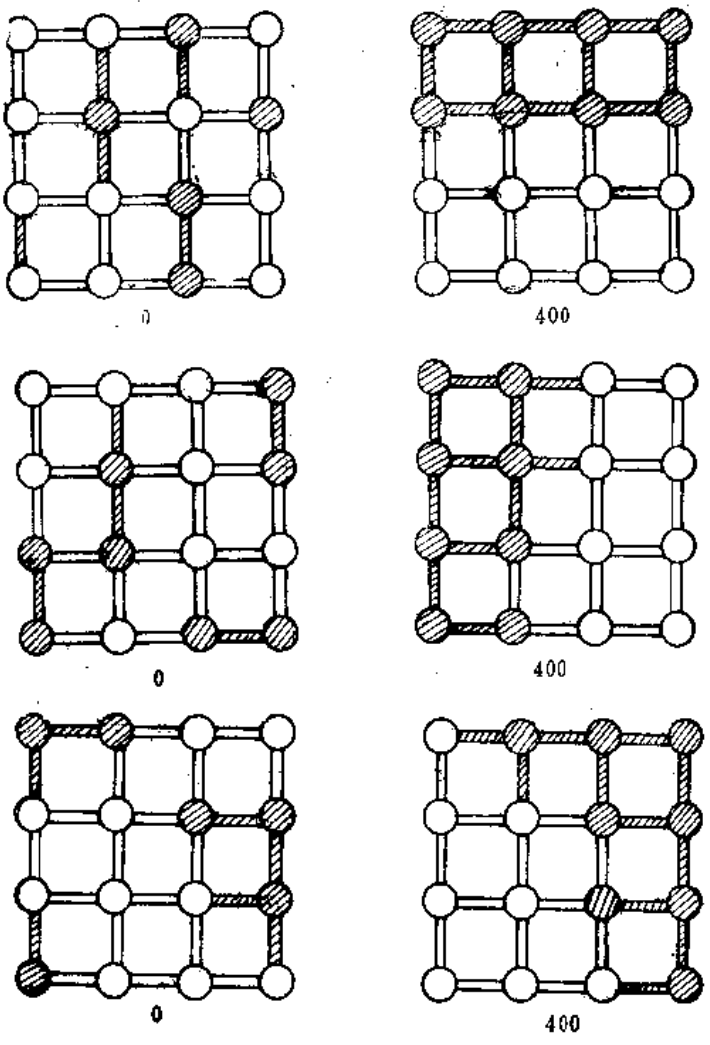
微分竞争学习只与输出的变化有关，当 $\dot{s}'_j(h_j)$ 达到平坦区，$\dot{s}'_j(h_j)=0$，则 $\dot{w}_{ij}=0$，w_{ij} 也不会改变了。同样当输入 $s_i(x_i)=x_i$ 时，(4-14) 式可写为

$$\dot{w}_{ij}=\dot{s}'_j(h_j)(x_i-w_{ij})$$

或写成矢量形式

$$\dot{\boldsymbol{m}}_j=\dot{s}_j(h)(\boldsymbol{x}-\boldsymbol{m}_j)$$

以上讨论的四个学习规律是在不同的自组织网络中运行



(a)　　(b)

图 4-3

(a) 初态；(b) 400 次迭代后的终态

的，它们是没有外加教师的，因此这些学习是有一定的随机性。当网络达到稳定时，能完成一定的功能，但它并没有指定那一个神经元兴奋或抑制。下面的例子能说明这种自组织网络的特点。

图 4-2 是一个两层的自组织网络，输入层为 $m\times n$ 个神经元，以方阵形式排列，$m\times n=16$，输入的样本满足这样的模式：$m\times n$ 个神经元中只有两个神经元兴奋，其他神经元都不兴奋，而兴奋的神经元必定在输入两维的格子中，形成偶极子。在 $n\times m$ 两维格子中存在 $n(m-1)+m(n-1)$ 个可能的样本，输出神经元为两个，如图中的 1 和 2，它们相互竞争，这里 $m=4, n=4$，则有 24 个可能输入的样本。图 4-3 为输入平面，其中用黑白两色表示，当输入偶极子落在两个都是黑的格子上，则输出神经元 1 兴奋，当输入样本落在两个都白的格子上，输出神经元 2 兴奋。如果某一个样本落在一个黑一个白的两个格点上，则通过竞争，这两个输出神经元都有可能"赢"。图 4-3*a* 为初始权的情况下，样本落在格点上其输出的情况，图 4-3*b* 为在经过 400 次学习后得到的结果。虽然没有教师，但两个神经元分工得十分好，在图 4-3*b* 的第一行，上半平面的偶极子使神经元 1 兴奋，下半平面的偶极子使神经元 2 兴奋。第二行为左半平面的偶极子使神经元 1 兴奋，右半平面的偶极子使神经元 2 兴奋；第三行则以对角线为分界，右上平面使神经元 1 兴奋，左下平面使神经元 2 兴奋。这些结果的不同是因为权的初值不同而造成的，初值是随机的，其结果也是随机的。但用 Hebb 学习和竞争的方法得到的结果是依据偶极子的位置把输入平面均匀地一分为二，每个神经元只对它"管辖"的区域兴奋，因而很自然地抓住了输入样本的位置特征。

第二节　自适应共振理论模型

一、ART 的模型结构

自适应共振理论 ART (Adaptive Resonance Theory) 模型是一种比较接近于实际神经系统的一种模型，它的记忆容量可以随学习样本的增加而增加，记忆形式也与生物中的记忆形式类似，它的结构如图 4-4 所示。

ART 模型一般由两层神经元和一些控制部分互相结合而成的。在图 4-4 的结构图中，

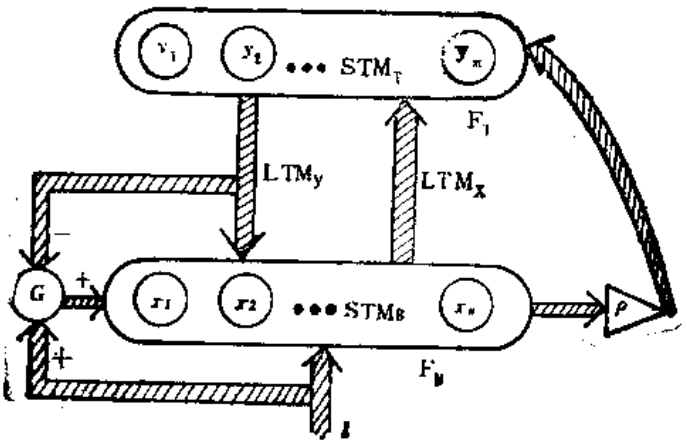


图 4-4　ART 模型结构

底层为 F_B 层，由 n 个神经元组成，它接受外界输入信号 I，同时还接受上一层来的信号，并且受增益控制器 G 的控制，上层为输出层用 F_T 表示，接受来自 F_B 的信号，同时又受控制器 ρ 的输出所控制。F_T 的神经元数为 m 个，m 的数目多少决定了 ART 网络的容量。

ART 网络存在两种记忆方式。一种称为短期记忆，用 STM(Short time memory)表示，它是由 F_B 和 F_T 中神经元的状态来决定的，在图 4-4 中为 x_i 和 y_j。x_i 是由输入到 F_B 的三个信号所决定，因此 x_i 的值在网络运行的过程中是变化的，当输入 I 的样本改变时 x_i 的值也会改变，因此，x_i 在某一固定输入时刻暂时被记忆在 F_B 中，我们用 STM_B 来表示，y_j 的状态在网络工作时也是不断变动的，如果 y_j 是经过竞争后产生的"赢"的最大值，在运行过程中，这个最大的 y_j 还会有变化，在稳定时产生的最大的 y_j 与外界输入的样本也有关系，不同的 I 输入，其 y_j 的值也不同，y_j 也是短期记忆，用 STM_T 表示。另一种称为长期记忆 LTM (long time memory)，它是由两组权来决定，一组是由下向上的权用 LTM_x 表示，另一组是由上向下的权用 LTM_y 表示，一旦网络经过学习后，对样本的记忆留在两组权中，即使输入样本改变了，这两组权依然存在，而且当输入的样本 I 为已经记忆的那个样本，那么这两组长期记忆将输出神经元回忆到原来的状态。

F_T、F_B 的输出信号是由信号的流向来控制的，在 F_B 中的神经元满足 2/3 规则。这个规则是：F_B 的输入由三方面信号控制，即输入信号 I、增益控制 G 和从 F_T 输出返回的信号，如果这三个信号中存在两个信号输入，则 F_B 中的 x_i 就可以改变。从图 4-5a 可以看出，如果外界有信号输入 I，那么 I 会向增益控制器 G 输入一个加强信号，此时 F_T 无信号输出，G 有一个加强信号输入到 F_B，此时 F_B 还是受到两个信号的控制，F_B 内的神经元状态 x_i $i=1,2,\cdots,n$ 会受到 I 的影响而变化。从图 4-5b 可以看出，如果 F_T 有信号输出，要控制 F_B 和 G，在 F_B 上除了接受外加的 I 信号外，还接受从 F_T 输出返回来的信号，此时从 F_T 输出一个抑制信号到 G，使 G 对 F_B 的输出进行抑制，这样 F_B 中神经元的状态则由输入 I 和从 F_T 输出返回的信号所控制，x_i 也会发生改变。除了这两种情况外，其他的情况，即只有一方面的信号控制，F_B 都不能使状态 x_i 改变。这个规则可以防止信号流对 F_B 的控制发生混乱，尤其在用硬件实现 ART 网络时特别有用。

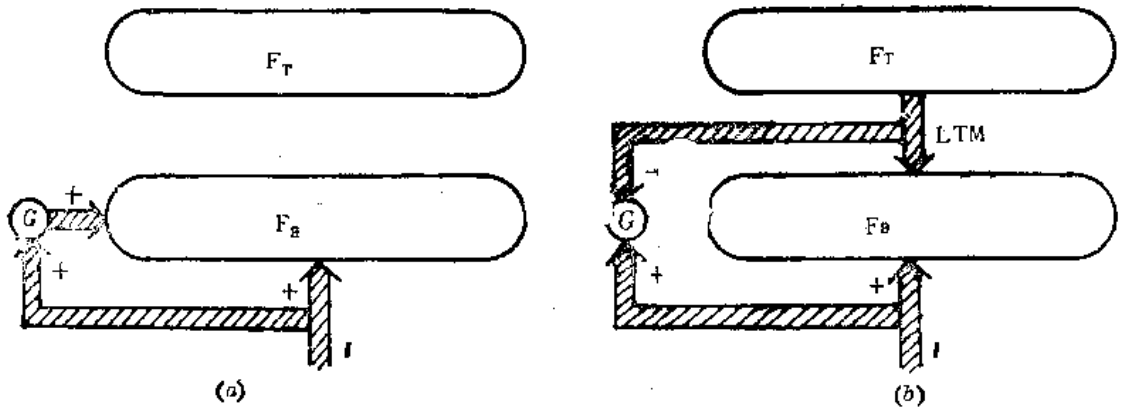


图 4-5　两种不同情况的 2/3 规则

在图 4-4 中，还有一个控制输出器 ρ，ρ 是当 F_B 在 I 和 F_T 返回到 F_B 信号存在的时候产生作用的，ρ 称为 ART 匹配控制器。当 F_T 输出返回到 F_B 与外界输入 I 的匹配程度比

较好时，则匹配控制器无输出。当 F_T 返回信号与外界输入样本不匹配时，则匹配控制器输出，并将 F_T 中"赢"的那个神经元抑制，此时新的竞争重新开始。

二、ART 的工作流程

1. 自 F_B 向 F_T 的流程

输入 I 样本，$I \in R^n$，它的每个分量分别加到 F_B 的每个神经元上，产生的状态为 $x \in R^n$，因为输入 I 存在时，就向增益控制器 G 输入一个加强量，增益控制器对 F_B 中每个神经元有输入，因此根据 2/3 规则每个神经元都接受 I 的输入，最简单的则是 $x_i = I$ $(i=1, 2, \cdots, n)$ 如图 4-5a 所示，此时 F_B 对 ρ 控制器输入为负值，抑制了 ρ 控制器的输出。

X 状态经过 F_B 中神经元的变换，得到其输出为 s，$s_i(x_i)$ 为每个 F_B 中的神经元的输出，$s \in R^n$，$s_i(x_i)$ 与 F_T 中每个神经元相联，联接权为 w_{ij}，表示第 i 个 F_B 中神经元与第 j 个 F_T 中神经元之间的联接权，w_{ij} 为长期记忆(LTM_X)，它是通过学习得到的，从 s 经过加权累加在 F_T 中的每个神经元输入端的值为

$$T_j = \sum_{i=1}^{n} s_i(x_i) w_{ij} \quad j = 1, 2, \cdots, m \tag{4-15}$$

$$T = (T_1, \cdots, T_j, T_m)^T \quad T \in R^m$$

当 T 输入后通过 F_T 得到一个新的短期记忆 $Y \in R^m$，因为对每个 Y 的分量 y_j 的输入 T_j 是不相同的，因而 y_j 的状态也各不相同。归纳上述情况我们得到从 F_B 到 F_T 的工作过程为

$$I \longrightarrow X \longrightarrow s \longrightarrow T \longrightarrow Y$$

$$(STM_B) \quad (LTM_X) \quad (STM_T)$$

在流程中，增益控制器 G 被打开，而匹配控制器 ρ 被抑制，在 F_T 中的神经元状态 y_j $(j=1, 2, \cdots, m)$ 被设置。如图 4-6a 所示

2. 在 F_T 中的竞争过程

在 F_T 中的状态 y_j 可以经过竞争算法或排序方法得到一个最大的输出 y_l，并抑制了其他的输出，假定状态 y_i 的输出为 $U_i(y_i)$，那么 $U_i(y_i)=1$，而 $U_j(y_j)=0$，$j \neq l$。

3. 从 F_T 反馈回 F_B 的过程

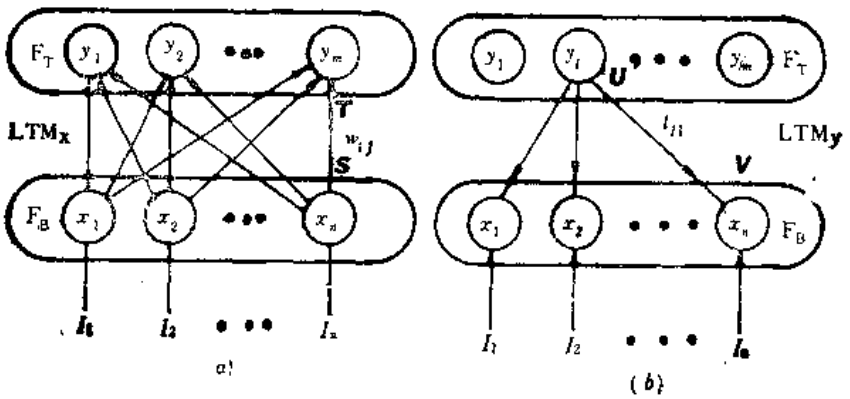


图 4- A[illegible]模型的长期记忆

(a) [illegible]，(b) LTM_Y

从 $U_i(y_i)=1$ 回到 F_B 是经过一个长期记忆 LTM_Y 进行的，图 4-6*b* 表示从 F_T 回到 F_B 的情况，长期记忆 LTM_Y 用权 $\{t_{ii}\}$ 来表示，它的输出乘上 t_{ii} 加到 F_B 中的各个神经元 i 输入端 $i=1,2,\cdots,n$，这种映照为 $\boldsymbol{V}$，根据图 4-5*b*，F_T 的输出还产生一个抑制信号，该信号去抑制增益控制器 G。使输入到 F_B 中的信号主要是由两部分组成，一部分是 $\boldsymbol{V}$，另一部分为 $\boldsymbol{I}$。两个信号的输入根据 2/3 规则，将改变 F_B 中的状态 $\boldsymbol{X}$ 为 $\boldsymbol{X}^*$。从 F_T 到 F_B 的过程又可以归纳为

$$\boldsymbol{Y}\longrightarrow\boldsymbol{U}\longrightarrow\boldsymbol{V}\longrightarrow\boldsymbol{X}^*$$

$$(STM_T)\quad(LTM_Y)\quad(STM_B)$$

4. 匹配控制器 ρ 对 F_T 状态的控制

当 $\boldsymbol{I}$ 输入到 F_B 时，F_B 对 ρ 控制器发出一个抑制信号，使 ρ 控制器没有输出，而 $\boldsymbol{V}$ 与 $\boldsymbol{I}$ 同时输入至 F_B 时，考虑 $\boldsymbol{V}$ 与 $\boldsymbol{I}$ 匹配的程度，如果匹配程度低，则 F_B 发出增加信号，使 ρ 控制器有输出，将 y_i 恢复到 0 值，此时，因为 y_i 没有输出，也没有信号去控制 G，$\boldsymbol{X}$ 还是由 $\boldsymbol{I}$ 和 G 的输出决定，按第 1 点中讨论的步骤进行，此时 y_i 不可能再"赢"，因为被 ρ 控制器抑制了，F_T 中神经元有新的状态出现，重复上述过程直达到 ρ 控制器没有输出为止。

三、权的设计及学习

在 ART 中存在着两种权，一个是自下而上的权，由 w_{ij} 表示 $\{w_{ij}\}$，为长期记忆用 LTM_X 表示，一个是自上而下的权用 $\{t_{ji}\}$ 表示，也是长期记忆用 LTM_Y 表示，这两类权都是联接两个不同层的神经元，它们的形式服从关联衰减规则和 Weber 规则。

关联衰减规则：如果在 F_B 内的神经元，在输入样本 $\boldsymbol{I}$ 加入后，有 n_1 个神经元不兴奋，那么与这些不兴奋神经元相关的权 t_{ji}、w_{ij} $(i=1,2,\cdots,n_1)$，为零或近似于零的很小值。

由于输出神经元 y_j，$j=1,2,\cdots,m$，每个都代表着一种记忆，它对应于某一类样本，竞争输出为 y_i 兴奋，与 y_i 相联的权就是对该样本的记忆，n_1 个不兴奋神经元被记忆到与 y_i 相联权上，即 t_{ii}，w_{ii} 中有关的 n_1 个权都为零。

Weber 规则：输入 $\boldsymbol{I}\in R^n$，用 $\boldsymbol{I}^1,\boldsymbol{I}^2,\cdots,\boldsymbol{I}^P$ 表示 P 个不同输入样本，如果 $s(x_i)=x_i=I_i^l$，$l=1,2,\cdots,P$，那么 LTM_X 的权满足下列公式：

$$\frac{\alpha}{\beta+|\boldsymbol{I}|}\tag{4-16}$$

这里 α 和 β 为常数，$|\boldsymbol{I}|$ 为输入分量的模。

用一个例子说明上述权的意义。为了简单起见，输入样本 $\boldsymbol{I}$ 是由 $\{0,1\}$ 二值变量组成，$|\boldsymbol{I}|$ 为矢量 $\boldsymbol{I}$ 中分量为 1 的个数，考虑输入为 $\boldsymbol{I}^1$，$\boldsymbol{I}^2$ 两个样本，已知 $|\boldsymbol{I}^1|<|\boldsymbol{I}^2|$，且 $\boldsymbol{I}^2$ 包含了 $\boldsymbol{I}^1$，其输出在 F_T 中的神经元 y_1，y_2，分别对应 $\boldsymbol{I}^1$，$\boldsymbol{I}^2$ 两个样本。据 Weber 规则，F_B 到 y_1 的联接权为 $\frac{\alpha}{\beta+|\boldsymbol{I}^1|}$，到 y_2 的联接权为 $\frac{\alpha}{\beta+|\boldsymbol{I}^2|}$，已被 LTM_X 记忆在权中，同时按关联衰减规则：$\boldsymbol{I}^1$ 中为 0 的分量与 y_1 对应的权为0，$\boldsymbol{I}^2$ 中为0 的分量与 y_2 对应的权为 0，当 $\boldsymbol{I}^1$ 输入至 F_B 时，对应在 y_1 与 y_2 的输入分别为

$$T_{11}=\frac{\alpha|\boldsymbol{I}^1|}{\beta+|\boldsymbol{I}^1|}$$

$$T_{12}=\frac{\alpha|\boldsymbol{I}^1|}{\beta+|\boldsymbol{I}^2|}$$

T_{11} 是 y_1 的输入，T_{12} 是 y_2 的输入，因为 $|I^1|<|I^2|$，因而有 $T_{11}>T_{12}$，按竞争的原则，若 $y_1>y_2$，则 $U_1(y_1)=1$，$U_2(y_2)=0$。如果输入 F_B 为样本 I^2，在 y_1 与 y_2 的输入端对应的值为 T_{21} 和 T_{22}：

$$T_{21}=\frac{\alpha|I^1|}{\beta+|I^1|}$$

$$T_{22}=\frac{\alpha|I^2|}{\beta+|I^2|}$$

在 T_{21} 的分子上，α 乘的不是 $|I^2|$，而是 $|I^1|$，这是因为关联衰减规则的缘故；$I^2\cap I^1=I^1$，那些 I^1 矢量中为 0 的分量的权为 0，因此 I^2 输入后在 T_{21} 上得到的还是 $|I^1|$。很明显，因为 $|I^2|>|I^1|$，所以 $T_{22}>T_{21}$，通过竞争 $U_2(y_2)=1$，$U_1(y_1)=0$。

从上例可以看出，LTM_x 是按照上述两个规则来建立记忆的，对于 LTM_y，它的形式与 LTM_x 相似，由于 LTM_y 受到了 2/3 规则的限制，为了保证由上向下的信号对 F_B 中的状态进行修改，因而它的记忆权值比 LTM_x 大。

权的学习公式如下：

$$\frac{dw_{ij}}{dt}=U_j(y_j)[\ -F_{ij}(x)w_{ij}+G_{ij}(x)s_i(x_i)\ ] \tag{4-17}$$

这里 $F_{ij}(x)$，$G_{ij}(x)$ 为与状态有关的量。

从上式我们可以看到，学习只有 $U_j(y_j)=1$ 时才会进行，而 $U_j(y_j)=0$ 的那些权都不进行学习。

(4-17)式也说明，当学习结束后 $\frac{dw_{ij}}{dt}=0$，则有

$$w_{ij}=\frac{G_{ij}(x)}{F_{ij}(x)}s_i(x_i) \tag{4-18}$$

令

$$F_{ij}(x)=s_i(x_i)+L^{-1}\sum_{\substack{k=1\\k\neq i}}^{n}s_k(x_k)$$

式中 L 为 >1 的实数，我们有

$$\begin{aligned}w_{ij}&=\frac{G_{ij}(x)}{F_{ij}(x)}s_i(x_i)\\&=\frac{G_{ij}(x)}{s_i(x_i)+L^{-1}\sum_{\substack{k=1\\k\neq i}}^{n}s_k(x_k)}s_i(x_i)\\&=\frac{LG_{ij}(x)}{\sum_{k=1}^{n}s_k(x_k)+(L-1)s_i(x_i)}s_i(x_i)\\&=\frac{\alpha}{\beta+\sum_{k=1}^{n}s_k(x_k)}s_i(x_i)\end{aligned}$$

这里 $\alpha=LG_{ij}(x)$，$\beta=(L-1)s_i(x_i)$。

若 $s_i(x_i)=0$，则 $w_{ij}=0$；若 $s_i(x_i)=1$，则

$$w_{ij}=\frac{\alpha}{\beta+\sum_{k=1}^{n}s_k(x_k)}$$

满足 Weber 和关联衰减规则。

对于从上到下的权 t_{ji} 的学习公式，则和式(4-17)相似；若 $F_{ij}(x)$ 和 $G_{ij}(x)$ 为常数，t_{ji} 的学习公式为

$$\frac{\mathrm{d}\,t_{ji}}{\mathrm{d}t}=U_j(y_j)[-t_{ji}+s_i(x_i)] \tag{4-19}$$

如果 $s_i(x_i)$ 与 $U_j(y_j)$ 都兴奋时，则 t_{ji} 增加；当 $s_i(x_i)=0$ 时，t_{ji} 减少，直到等于零时为止；当 $U_j(y_j)=0$ 时，则 t_{ji} 不变。从(4-19)式可以看出 t_{ji} 并不满足 Weber 规则，它比 w_i 的权要大，以保证达到 2/3 规则。

四、ART 的实现

ART 模型可以用算法予以实现，也可以用硬件实现，用算法进行学习时，在权的修改上忽略了权调整的动态过程，如果按上述的工作流程进行，算法可以归纳为：

(1) 初始化权：$t_{ij}(0)=1$，$w_{ij}(0)=\frac{1}{n+1}$；$i=1,2,\cdots,n;j=1,2,\cdots,m,0\leqslant\rho\leqslant1$；这里假设输入第一层神经元为 n 个，输出第二层为 m 个。

(2) 输入 $\boldsymbol{I}$，计算 x_i 和 $s_i(x_i)$，得到 $\mathrm{F_B}$ 内的每个神经元的输出。

(3) 计算 T，　$T_j=\sum_{i=1}^{n}w_{ij}\,s_i(x_i)$　$1\leqslant j\leqslant m$

(4) 选择一个最大的 T_j 为 T_j^*，即

$$T_j^*=\max_j\ (T_j)$$

(5) 由 T_j^* 得到一个输出 $U_j(y_j)=1$，而其他为0。

(6) 从 y_j 出发计算 $t_{ji}U_j(y_j)=t_{ji}$，并从 $\mathrm{F_T}$ 出发返回 $\mathrm{F_B}$ 的输入，此时 $U_j(y_j)$ 将 G 控制器的输出抑制。如图 4-5b 示。

(7) 在 $\mathrm{F_B}$ 中计算新的状态 $\boldsymbol{X}^*$。设原来的状态为 $\boldsymbol{X}$，系由外界输入 $\boldsymbol{I}$ 所决定，受 t_{ji} 的影响得到新的状态 X_i^*。

(8) 计算匹配因子：定义 $|\mathbf{X}|=\sum_{i=1}^{n}x_i$，　$|\mathbf{X}^*|=\sum_{i=1}^{n}x_i^*$；若 $|\mathbf{X}^*|/|\mathbf{X}|>\rho$，则转向(9)；若 $|\mathbf{X}^*|/|\mathbf{X}|<\rho$，则 ρ 控制器发出一个复位信号使 $\mathrm{F_T}$ 中的 $U_j(y_j)=0$，此时 G 控制器的抑制信号为增强信号 $\mathbf{X}^*=0$，又可用 (3) 进行计算，此时 $U_j(y_j)$ 不参加竞争，从 (3)、(4)、(5)、(6)、(7)直到 $|\mathbf{X}^*|/|\mathbf{X}|>\rho$ 为止。

(9) 修改权 w_{ij}，t_{ji}，j 为"赢"的神经元标号，$i=1,2,\cdots,n$。

$$t_{ji}^*\ (t+1)=t_{ji}^*\ (t)s_i\ (x_i)$$

$$w_{ij}(t+1)=\frac{t_{ji}^*s_i(x_i)}{\beta+\sum_{i=1}^{n}x_{ji}s_i(x_i)}$$

(10) 转向(2)，接受新的输入样本。

通过上面的算法可以将输入的样本 $\boldsymbol{I}^P P=1,2,\cdots,l$ 都记忆到权 w_{ij} 和 t_{ji} 上，形成长期记忆，ρ 的选择大小可以调节对输入样本分类的能力，ρ 接近于 1，那么输入样本与已学习记忆过的样本略有不同，就属于新的一类样本，被记忆到长期记忆中，如 ρ 比较小，那么输入样本有畸变和干扰仍可属于已记忆的样本一类，因此 ρ 的选择与要解的问题情况有关。

用硬件来实现 ART 时，F_T 和 F_B 内的神经元都用电路来实现，其长期记忆的权是用压控的CMOS 电路来完成，整个电路的动作是用状态方程来描述，例如在 F_T 中的状态方程可用下式进行：

$$\frac{d}{dt}y_j=-A_jy_j+(B_j-C_jy_j)\quad T_j^+-(E_j+F_jy_j)T_j^- \tag{4-20}$$

其中 B_j, C_j, E_j, F_j 为一个个自定常数，T_j^+, T_j^- 分别为 y_j 神经元输入端兴奋信号总和及抑制信号总和。状态方程可用相应的电路来完成；(4-20)中 T_j^+ 是一个正号的信号在方程(4-20)中使状态值 y_j 增加，T_j^- 是一个负的值，它使 y_j 减小，为了保证 y_j 是一个有界的数，(4-20) 式中当状态 $y_j>B_j/C_j$ 时，T_j^- 的作用反而使 y_j 减少，当 $y_j<-E_j/F_j$ 时，T_j^+ 反而使 y_j 状态增大，这样 y_j 状态的最大范围为：$[-E_j/F_j, B_j/C_j]$，当 $\frac{d}{dt}y_j=0$ 时，则状态达到了稳定。E_j, F_j, B_j 和 C_j 都是常数，设计者可根据要求设计状态的极大和极小值。

具体电路我们这里不详细讨论了，读者可以根据 ART 的算法及神经元的状态方程自行设计完成。

第三节　自组织映照模型

自组织映照模型是由 Kohonen 提出来的，模型基于实际神经细胞中的一种特征敏感细胞，在外界信号刺激下，形成对某一种特征特别敏感，它们这种特性的形成是通过自学习而得到的。神经细胞模型中还存在一种细胞聚类的功能柱，它是由多个细胞聚合而成的，在接受外界刺激后，它们会自动形成，一个功能柱中的细胞完成同一种功能。生物中这些细胞结构和现象在 Kohonen 的模型中有所反映。当外界输入不同的样本到人工的自组织映照网络中，一开始时，输入样本引起输出兴奋细胞的位置各不相同，但自组织后会形成一些细胞群，它们分别代表了输入样本，反映了输入样本的特征。这些细胞群，如果在二维输出空间，则是一个平面区域，样本自学习后，在输出神经元层中排列成一张二维的映照图，功能相同的神经元靠得比较近，功能不相同的神经元分得比较开。这个映照的过程是用一个简单的竞争算法来完成的，其结果可以使一些无规则的输入自动排序，在联接权的调整中可使权的分布与输入样本的概率密度分布相似，同时它又是一种样本特征检测器，在样本排序、样本分类、样本检测方面有广泛的应用。

一、自组织映照模型的结构和工作过程

1. 结构与公式

Kohonen 网络是由输入层和输出层两层神经网络组成的。如图 4-7所示。输入层中的每一个神经元，通过权与输出层的每一个神经元相联，输出层中的神经元一般是以二维形式排列的，它们中的每个神经元是输入样本的代表。在输出层中竞争是这样进行的：对于"赢"的那个神经元 c，在其周围 N_c 的区域内神经元在不同程度上得到兴奋，而在以 N_c 以外的神经元都被抑制，这个 N_c 区域可以是正方形，也可以为六角形，如图 4-8 所示。N_c 是 t 的函数，随着 t 的增加，N_c 的面积成比例地缩小，最后只剩下一个神经元，也可能是一个组的神经元，它们反映了一类样本的属性。如果输入样本用矢量 x^P 表示，$P=1, 2, \cdots, k$，共有 k 个样本，而每个样本的分量 x_i^P，与输入的第 i 个神经元相联，x_i^P 与输出第 j 个神经元之

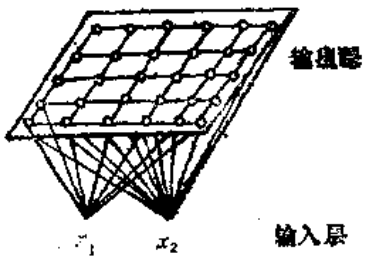


图 4-7 Kohonen 网络的结构

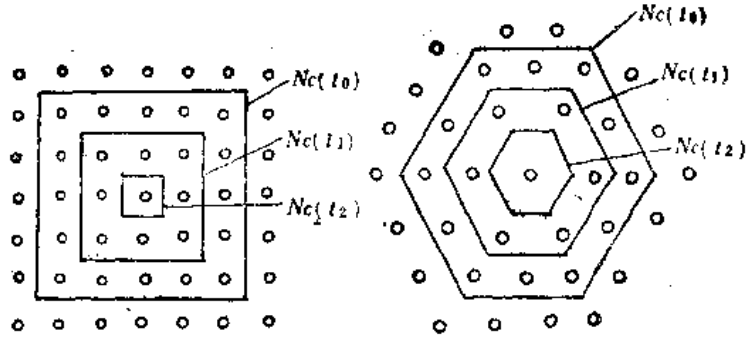


图 4-8 $N_c(t)$的形状和变化情况

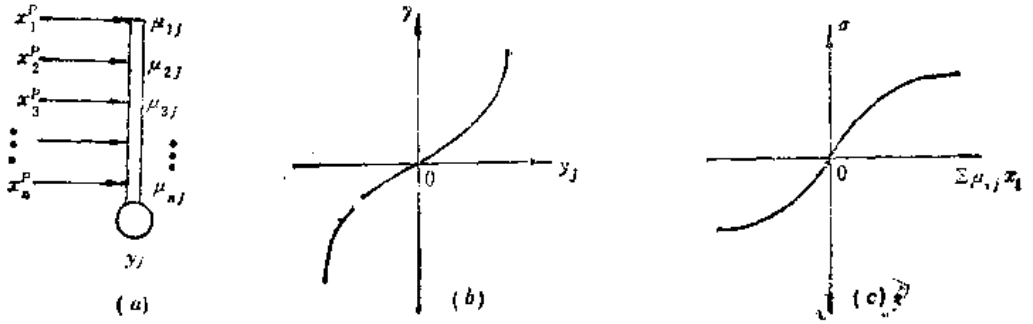


图 4-9

(a) 单个输出神经元结构； (b) $y_j \sim \gamma$ 的非线性关系； (c) $\sigma \sim \sum u_{ij}x_i$

间的权为 μ_{ij}，而其输出为 y_j。对于一个输出神经元 y_j 可用图 4-9a 来表示，x_1^p, x_2^p, …, x_n^p 是由外界的输入样本决定，其输入与输出满足下列状态方程：

$$\frac{dy_j}{dt}=\sum_{i=1}^{n}\mu_{ij}x_i^p-\gamma(y_j) \tag{4-21}$$

式中第一项为输入的加权累加，第二项是一个与 y_j 有关的函数，此函数可采用不同的形式，但为非线性项，它的效果是使 y_j 的变化速率变慢，当外界没有输入或输入的量与权的乘积比较小的时候，输出单元 y_j 的值减小，直到 0 为止。在式 (4-21) 中的第一项比较大时，$\frac{dy_j}{dt}$增长快，但 y_j 的增加又会引起 γ 的增加，图 4-9b 为一个非线性的 $\gamma(y_j)$ 关系，直到 $\frac{dy_j}{dt}=0$。y_j 可用下式描述：

$$\begin{gathered} y_j=\sigma\left(\sum_{i=1}^{n}\mu_{ij}x_i\right) \\ \sigma(\cdot)=\gamma^{-1}(\cdot) \end{gathered} \tag{4-22}$$

$\sigma(\cdot)$是一个单调上升的非线性函数，如图 4-9c 所示，与一般神经元输入、输出关系相似。竞争是在 y_j 之间进行，那个 y_j 最大的神经元就是竞争后“赢”的神经元。μ_{ij} 的学习是满足 Hebb 规则的，变化正比于输入与输出状态值的乘积：

$$\frac{d\mu_{ij}}{dt}=\alpha y_j x_i^p-\beta(y_j)\mu_{ij} \tag{4-23}$$

式中 α 是一个常数，前一项是服从 Hebb 规则的，当输入 x_i 与 y_j 都兴奋时，μ_{ij} 增长较快，后

一项 $\beta(y_j)\mu_{ij}$ 是一个遗忘因子，当外界没有输入时，权 μ_{ij} 会随时间的增大而减小，这反映了遗忘这一特性。在设计中，直接应用(4-21)式、(4-23)式是较少的，因为竞争在算法中只要对比 x_i 与 μ_{ij} 之间的相近程度就可得到。在式(4-23)中考虑一个特殊情况，当 $y_j \in \{0,1\}$ 为 0，1 的二值函数，且满足 $y_i=0$ 时，$\beta(y_j)=0$；$y_j=1$ 时，$\beta(y_j)=\alpha$，则方程(4-23)变为

$$\begin{cases} \dfrac{\mathrm{d}\mu_{ij}}{\mathrm{d}t}=\alpha y_j x_i^P-\beta(y_j)\mu_{ij}=\alpha(x_i^P-\mu_{ij}) & y_j=1 \\ \dfrac{\mathrm{d}\mu_{ij}}{\mathrm{d}t}=0 & y_j=0 \end{cases} \tag{4-24}$$

由上式派生出来的算法其结果是：使学习后的权 μ_{ij} 越来越靠近输入的 x_i^P，把 (4-24) 式写成矢量的形式，即

$$\boldsymbol{m}_j=(\mu_{1j},\mu_{2j},\cdots,\mu_{nj})^T$$
$$\boldsymbol{x}^P=(x_1^P,\ x_2^P,\ \cdots,\ x_n^P)^T$$

则

$$\dot{\boldsymbol{m}}_j=\alpha(\boldsymbol{x}^P-\boldsymbol{m}_j) \tag{4-25}$$

这里 α 是与时间和距离都有关系的量，而且，在(4-24)式中权的调整是在“赢”的神经元周围 N_c 区域内进行的，比较靠近“赢”的神经元的那些神经元的权调整系数 α 大，而远离的那些神经元的 α 小，α 随距离的变化如图 4-10 所示，图中 r_0 表示“赢”的那个神经元所在的位置，在其周围其他神经元的位置用半径 r 表示，α_0 为权调整的步长。图 4-10b 表示在 r_0 周围 $|r-r_0|<R$ 的范围内，α_0 与 $|r-r_0|$ 之间的关系如同一个墨西哥草帽的函数，图 4-10b 是由两个正态形状的曲线组合而成的，如图 4-10a 所示。可用公式表示为

$$\alpha_0(r)=a_1\mathrm{e}^{-\frac{(r-r_0)^2}{\sigma_1}}-a_2\,\mathrm{e}^{-\frac{(r-r_0)^2}{\sigma_2}} \tag{4-26}$$

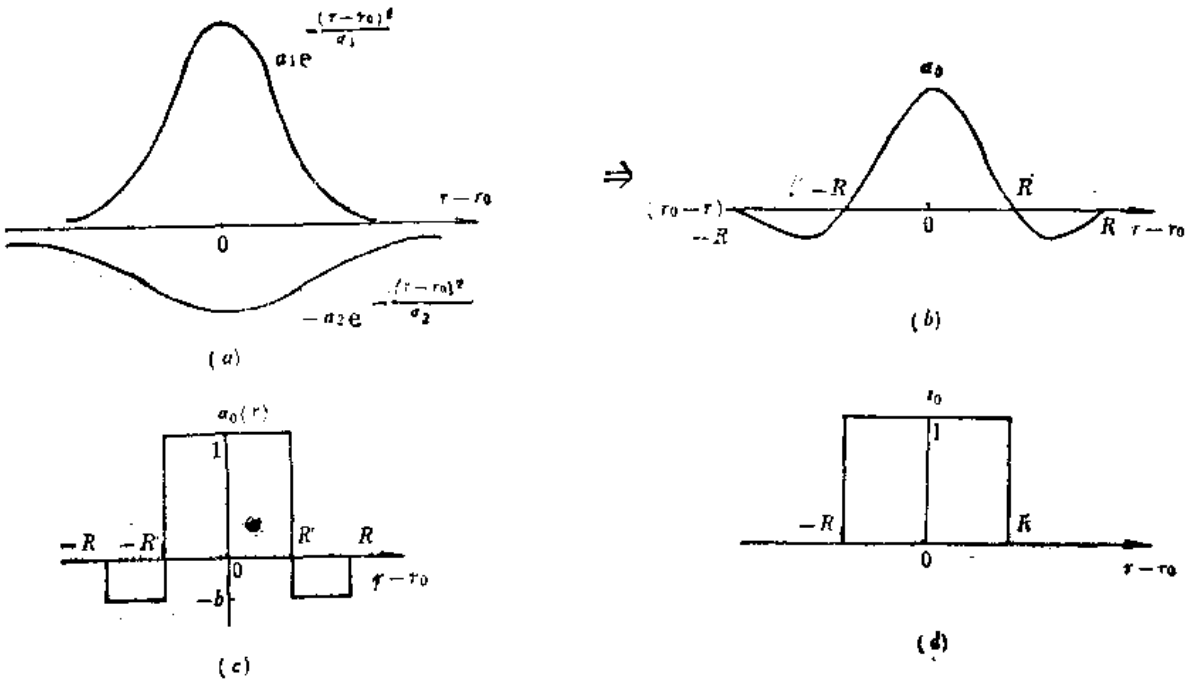


图 4-10　α_0 随距离 $|r-r_0|$ 变化的情况

(a) 两个正态函数相减；(b) 墨西哥草帽函数，由(a)中两个函数组合而成；(c) 火炉管函数；(d) 礼帽函数

式中：a_1, a_2, σ_1, σ_2 为常数，$a_1>a_2$；$\sigma_2>\sigma_1$，十分明显，在 $|r-r_0|<R'$ 的范围内，α_0 为正值，并且随距离增大而减小，在 $R'<|r-r_0|<R$ 的范围内，α_0 为负值，对在 $|R'|$ 以外的神经元，其联接权调整使 μ_{ij} 变化减小，在 $|R|$ 以外的权不进行调整。图 4-10c 是一个火炉管函数，它满足：

$$\alpha_0(r)=\begin{cases}1 & |r-r_0|<R' \\ -b & R'<|r-r_0|<R \\ 0 & |r|>R\end{cases} \tag{4-27}$$

图 4—10d 是一个礼帽函数，满足：

$$\alpha_0(r)=\begin{cases}1 & |r-r_0|\leqslant R \\ 0 & |r-r_0>R\end{cases} \tag{4-28}$$

考虑到(4-24)式中的 $\alpha_0(r)$ 又是 t 的函数，可以写为 $\alpha_0(r,t)$，随着 t 的增加，α_0 的值减小，可以用下式子表示 α_0 与 t 的关系：

$$\left.\begin{aligned}&\alpha_0(t)=A\mathrm{e}^{-t/\tau_1} \\ \text{或}\quad&\alpha_0(t)=0.9\left(1-\frac{t}{1000}\right)\end{aligned}\right\} \tag{4-29}$$

由于"赢"的那个神经元周围的区域 N_c 也是时间 t 的函数，可得

$$N_c(t)=A_1+A_2\mathrm{e}^{-t/\tau_2} \tag{4-30}$$

其中：τ_2、A_1、A_2 为常数。

$N_c(t)$ 也是随着时间的变化而逐步减小，直到只剩下一个神经元为止，如图 4-8 所示。

2. 算法

对于图 4-7 那样的 Kohonen 网络，输入神经元数与问题的要求有关，设输入为 n 个神经元，输出为一个二维平面，希望在输出平面上的神经元数足够多，设为 n_1 个，$n_1\gg n$，定义一个 d 函数，表示输入样本矢量 $\boldsymbol{x}$ 与权 $\boldsymbol{m}_j(j=1,2,\cdots,n_1)$ 匹配的程度，用欧氏距离表示：

$$d(\boldsymbol{x},\boldsymbol{m}_j)=\|\boldsymbol{x}-\boldsymbol{m}_j\| \tag{4-31}$$

如果在输出层中有一个神经元与输入 x 的匹配最好，记为 c，则

$$d(\boldsymbol{x}-\boldsymbol{m}_c)=\min_j d\ (\boldsymbol{x}-\boldsymbol{m}_j)$$

$\boldsymbol{m}_c$ 所对应的输出为

$$y_c=\max_j y_j$$

而权的修正则对 $\boldsymbol{m}_c$ 和 $y_i\in N_c$ 中的 $\boldsymbol{m}_j$ 进行，具体算法为：

(1) 初值权 $\boldsymbol{m}_j=\boldsymbol{m}_j(0)$ 为一个小的随机量。

(2) 在样本 $\boldsymbol{x}^1,\boldsymbol{x}^2,\cdots,\boldsymbol{x}^P$ 中，任取一个样本作为 Kohonen 网络的输入。

(3) 计算 $\|d\|=\|\boldsymbol{x}^p-\boldsymbol{m}_j\|=\sum_{i=1}^{n}(x_i^p-\mu_{ij})^2$，$j=1,2,\cdots,n_1$，取其中最大的 $\|d\|$，对应于 y_c 最大，作为竞争得胜的神经元。

(4) 对权 $\boldsymbol{m}_j$ 进行修改：

$$\begin{cases}\boldsymbol{m}_j(t+1)=\boldsymbol{m}_j(t)+\alpha_0(t,r_0)\,(\boldsymbol{x}-\boldsymbol{m}_j(t)) & i\in N_c(t) \\ \boldsymbol{m}_j(t+1)=\boldsymbol{m}_j(t) & i\notin N_c(t)\end{cases} \tag{4-32}$$

修正的区域 $N_c(t)$，且一开始很大，约为 1/2 的输出平面，中心点为 y_c，形状可用正方形或

六角形，然后每次迭代都按(4-30)式的形式减小。

(5) $\alpha_0(t,r_0)$ 可采用 (4-26) 式或 (4-27)、(4-28)和 (4-29) 式进行，如果 $\alpha_0(t,r_0)$ 用 (4-28)式，那么 $\alpha(r_0,t)=\alpha(t)$。

(6) 回到步骤(2)，反复(2)、(3)、(4)、(5)，直到在输出神经元平面上的兴奋神经元与输入样本稳定对应为止。

步骤(3)的竞争还可以用下面的形式进行，即输入样本 $\boldsymbol{x}$ 与权 $\boldsymbol{m}_j$ 的内积最大的值为竞争优胜者：

$$\boldsymbol{x}^T\boldsymbol{m}_j(t)=\max_j[\boldsymbol{x}^T\boldsymbol{m}_j(t)]$$

它的效果与步骤(3)相似，都能找出与输入样本 $\boldsymbol{x}^P$ 匹配得最好的那个输出神经元 y_c。公式(4-32)可用一个归一化的公式来代替

$$\boldsymbol{m}_j(t+1)=\begin{cases}\dfrac{\boldsymbol{m}_j(t)+\alpha'(t)\boldsymbol{x}(t)}{\|\boldsymbol{m}_j(t)+\alpha'(t)\boldsymbol{x}(t)\|} & j\in N_c(t)\\ \boldsymbol{m}_j(t) & j\notin N_c(t)\end{cases}\tag{4-33}$$

在(4-33)式中，当 $t\to\infty$ 时，$\alpha'(t)\to 0$，此时迭代稳定后得到的

$$\boldsymbol{m}_j(t+1)=\frac{\boldsymbol{m}_j(t)}{\|\boldsymbol{m}_j(t)\|}$$

为一个归一化的权。(4-33)、(4-32)式都是从(4-24)式派生出来的，其效果相近。

从上面讨论的 Kohonen 网络的公式和算法来看，Kohonen 网络有如下几个特点：

(i) 网络中的权是输入样本的记忆。如果输出神经元 j 与输入 n 个神经元之间的联接权用 $\boldsymbol{m}_j$ 表示，对应某一类样本 $\boldsymbol{x}^P$ 输入，使 y_j 达到匹配最大，那么 $\boldsymbol{m}_j$ 通过学习后十分靠近 $\boldsymbol{x}^P$，因此以后当 $\boldsymbol{x}^P$ 再次输入时，y_j 这个神经元必定会兴奋，y_j 是样本 $\boldsymbol{x}^P$ 的代表。

(ii) 网络学习时对权的调整，不只是对兴奋的那个神经元所对应的权进行，而对其周围 N_c 区域内的神经元同时进行调整，因此对于在 N_c 内的神经元可以代表不只是一个样本 $\boldsymbol{x}^P$，而是与 $\boldsymbol{x}^P$ 比较相近的样本都可以在 N_c 内得到反映，因此这种网络对于样本的畸变和噪声的容差大。

(iii) 网络学习的结果使比较相近的输入样本在输出二维平面的位置上也比较接近。

二、Kohonen 网络的工作原理

用上面提到的网络结构和算法可以完成对输入样本的排序、映照和分类等功能，但网络为什么具有这些功能则未提及，为了进一步了解 Kohonen 网络的原理，我们从输出一维的情况下来讨论，一维情况如图 4-11 所示，二维情况下的分析则可类推。

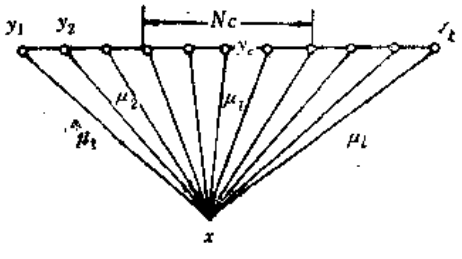


图 4-11 输出为一维情况下的 Kohonen 网络

1. 排序问题

用 Kohonen 的学习方法可以使输出神经元的兴奋位置与输入样本的大小有关，按样本值的大小，或从左到右排序，或从右到左排序。例如输入 x 是一个随机数，为 $x(t_1),x(t_2),\cdots,x(t_P)$，代表不同的随机数样本，如果输出为 $y_1,y_2,y_3,\cdots,y_t$，它们排列在一条直线上，其联接权用 $\mu_1,\mu_2,\cdots,$

μ_i 代表。$x\in R$，且当 $i\neq j$ 时，$x(t_i)\neq x(t_j)$，利用前面给出的算法，对任何一个输入 $x(t_i)$，可以找到一个匹配度最大的权 μ_c 使

$$\|x(t_i)-\mu_c\|=\min_i \|x(t_i)-\mu_i\|$$

成立。我们定义一个区域 N_c：

$$N_c=\{\max(1,c-1),\ c,\ \min(l,c+1)\}$$

对于任一个输出单元 y_j，它的最相邻的输出为 $i-1$，$i+1$ 两个，只有在两个端点 y_1 和 y_l，它们相邻的神经元仅为一个，前者为 y_2，后者为 y_{l-1}。如果只考虑最近的近邻神经元，按方程(4-24)，在一维情况下可得

$$\begin{aligned}&\frac{\mathrm{d}\mu_i}{\mathrm{d}t}=\alpha(t)(x-\mu_i) && i\in N_c\\ &\frac{\mathrm{d}\mu_i}{\mathrm{d}t}=0 && i\notin N_c\end{aligned}\tag{4-34}$$

这里 $\alpha(r,t)=\alpha(t)$，用的是图 4-10d 的函数。上述的权调整算法，当无序的输入随机数样本学习后，在输出神经元上得到有序的排列，这种排列可以是单调递增，也可以是单调递减的。显然，$y_1,y_2,\cdots,y_l$ 对输入样本的有序排列主要是那些与 y_i 相联的权的有序排列造成的，用一个因子 D 来表示其输出规律的程度：

$$D=\sum_{i=2}^{l}|\mu_i-\mu_{i-1}|-|\mu_l-\mu_1|\tag{4-35}$$

在一般情况下，$D\geqslant 0$，而如果 μ_i 是有序排列的，那么 $D=0$，而无序排列 $D>0$，对 D 求导得

$$\frac{\mathrm{d}D}{\mathrm{d}t}=\sum_{i=2}^{l}\frac{\mathrm{d}}{\mathrm{d}t}|\mu_i-\mu_{i-1}|-\frac{\mathrm{d}}{\mathrm{d}t}|\mu_l-\mu_1|\tag{4-36}$$

如果 $3\leqslant c\leqslant l-2$，而且 N_c 的区域为 $\mu_c,\mu_{c+1},\mu_{c-1}$，能够用(4-34)式调整，在(4-36)式中只有五项，$\frac{\mathrm{d}\mu_{c+2}}{\mathrm{d}t}=0,\frac{\mathrm{d}\mu_{c-2}}{\mathrm{d}t}=0$，则有

$$\begin{aligned}\frac{\mathrm{d}D}{\mathrm{d}t}=&-\frac{\mathrm{d}}{\mathrm{d}t}|\mu_{c+1}|+\frac{\mathrm{d}}{\mathrm{d}t}|\mu_{c-1}|+\frac{\mathrm{d}}{\mathrm{d}t}|\mu_{c+1}-\mu_c|\\&+\frac{\mathrm{d}}{\mathrm{d}t}|\mu_c-\mu_{c-1}|-\frac{\mathrm{d}}{\mathrm{d}t}|\mu_l-\mu_1|\end{aligned}\tag{4-37}$$

上式中 μ_l，μ_1只有在 μ_c 为 μ_2,或为 μ_{l-1} 时才可能改变,对于(4-37)式中有各种可能的组合，如 $\mu_c>\mu_{c+1}$ 或 $\mu_{c+1}>\mu_c$，$\mu_{c-1}>\mu_c$ 或 $\mu_{c-1}<\mu_c$，$\mu_{c-1}>0\cdots$等，共有 31 种组合，其中的 13 种组合，具有$\frac{\mathrm{d}D}{\mathrm{d}t}<02$,种组合,$\frac{\mathrm{d}D}{\mathrm{d}t}=0$,16 种组合,$\frac{\mathrm{d}D}{\mathrm{d}t}>0$,虽然在初始时不能保证 $D=0$，但是当有序的规则一旦建立，则 D 保持常数，$\frac{\mathrm{d}D}{\mathrm{d}t}=0$的组合可能性变得越来越大，而且一旦 D 变成了 0，则 D 就永远为 0，即一个有序排列的系统不能变到无序，这可用(4-34)代入(4-37)可得。

下面我们用一个例子来说明：

设 μ_i，μ_{i-1}，μ_{i+1} 组成一个随机初值，而权的调整范围 N_c 只有 c，$c-1$，$c+1$ 三个，如果初始时：

$$\mu_{i-1}<\mu_{i-2}<\mu_i$$

是非顺序排列，而输入的随机数 $x(t_i)$ 使 $\mu_i \cdot x(t_i)$ 为最大，那么根据算法将对 μ_i，μ_{i-1}，μ_{i+1} 进行调整，而 μ_{i-2} 不调整，此时 μ_{i-1} 可以调整到在 μ_{i-2} 与 μ_i 之间，得

$$\mu_{i-2}<\mu_{i-1}<\mu_i$$

如果 $x(t)$ 是一个随机输入，$x(t)$ 的样本和输出的神经元数都足够多，那么经过自组织的 Kohonen 算法，其 μ_i 必定按照大小进行排列，因为输入 $x(t)$ 足够多，则每一个 μ_i 都可能成为优胜者，其优胜的结果就可把周围非顺序的排列恢复过来，一旦非顺序排列恢复后，按(4-37)式的讨论，一个有序的排列在 $D=0$ 后就不可能回到无序，因此只要学习次数足够，就一定能得到一种顺序的排列。

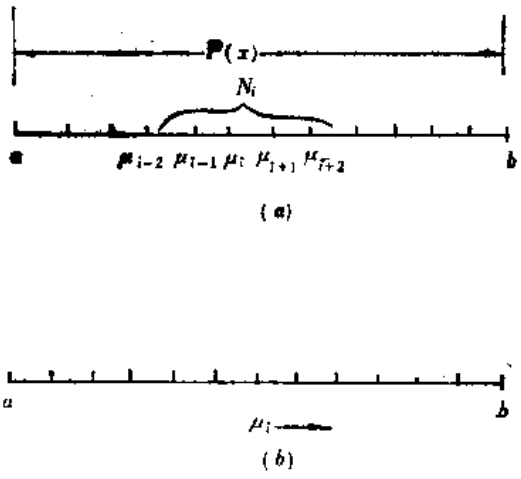


图 4-12 μ_i 的分布

2. 自组织学习的收敛性及权的分布

通过学习，μ_i 变成了一种有序的排列，如果继续学习最后会收敛。μ_i 的分布与输入样本的分布十分近似。这个现象可用下面一维的例子来分析。

图 4-12a 中 x 轴是 a 与 b 之间的区域，在输入样本 x 的情况下，每个输出神经元对应的权为 μ_1，μ_2，…，μ_l。因为 μ_i 是有序的，因此它在[a. b]区间内的排列也是有序的，但其分布还不清楚。假设输入随机样本 $x(t_i)$，$i=0$，1，…，t_n 是按概率密度 $P[x(t_i)]$ 分布的，而样本 $x(t_i)$ 输入后引起 μ_i 对应神经元 y_i 兴奋，那么权可调节的范围 N_i 可用下述方法获得。

当 $3\leqslant i\leqslant l-2$：

$$N_i=\left[\frac{1}{2}\left(\mu_{i-2}+\mu_{i-1}\right),\ \frac{1}{2}\left(\mu_{i+1}+\mu_{i+2}\right)\right]$$

当 $i=1$：　　$N_i=\left[a,\ \frac{1}{2}\left(\mu_2+\mu_3\right)\right]$

当 $i=2$：　　$N_i=\left[a,\ \frac{1}{2}\left(\mu_3+\mu_4\right)\right]$

当 $i=l-1$：　　$N_i=\left[\frac{1}{2}\left(\mu_{l-3}+\mu_{l-2}\right),\ b\right]$

当 $i=l$：　　$N_i=\left[\frac{1}{2}\left(\mu_{l-2}+\mu_{l-1}\right),\ b\right]$　　(4-38)

按公式(4-24)，并取期望值，得

$$\langle\dot{\mu}_i\rangle=E\{\dot{\mu}_i\}=\alpha[E\{x/x\in N_i\}-\mu_i]\qquad(4\text{-}39)$$

这里 $E\{x/x\in N_i\}$ 是在 N_i 的区域里，当 x 的分布 $P(x)$ 已知情况下的期望值。为了简单起见，先设 $P(x)$ 是一个常数，即为均匀分布，则(4-39)式可以展开为

当 $3\leqslant i\leqslant l-2$：　　$\langle\dot{\mu}_i\rangle=\frac{\alpha}{4}\left(\mu_{i-2}+\mu_{i-1}+\mu_{i+1}+\mu_{i+2}-4\mu_i\right)$

当 $i=1$：　　$\langle\dot{\mu}_1\rangle=\frac{\alpha}{4}\left(2a+\mu_2+\mu_3-4\mu_1\right)$

当 $i=2$:　　$\langle\dot{\mu}_2\rangle=\frac{\alpha}{4}(\ 2a+\mu_3+\mu_4-4\mu_2)$

当 $i=l-1$:　　$\langle\dot{\mu}_{l-1}\rangle=\frac{\alpha}{4}(\ \mu_{l-3}+\mu_{l-2}+2b-4\mu_{l-1})$

当 $i=l$:

$$\langle\dot{\mu}_l\rangle\ =\frac{\alpha}{4}(\mu_{l-2}+\mu_{l-1}+2b-4\mu_l) \tag{4-40}$$

假设初始条件为 $\mu_i(0)$，则平均的 $\mu_i(t)$ 可以通过解下列方程而得到：

$$\frac{\mathrm{d}\boldsymbol{Z}}{\mathrm{d}t}=\boldsymbol{F}\boldsymbol{Z}_T+\boldsymbol{h} \tag{4-41}$$

其中

$$\boldsymbol{Z}=[\mu_1(t),\mu_2(t)\cdots\mu_n(t)]^T$$

$$\boldsymbol{h}=\frac{\alpha}{2}[a,a,0,0,\cdots 0,b,b]^T$$

$$\boldsymbol{F}=\frac{\alpha}{4}\begin{bmatrix}
-4 & 1 & 1 & 0 & 0 & 0 & \cdots & \cdots & \cdots & 0 & 0 & 0 & 0 & 0 & 0\\
1 & -4 & 1 & 1 & 0 & 0 & \cdots & \cdots & \cdots & 0 & 0 & 0 & 0 & 0 & 0\\
1 & 1 & -4 & 1 & 1 & 0 & \cdots & \cdots & \cdots & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 1 & 1 & -4 & 1 & 1 & \cdots & \cdots & \cdots & 0 & 0 & 0 & 0 & 0 & 0\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
0 & 0 & 0 & 0 & 0 & 0 & \cdots & \cdots & \cdots & 1 & 1 & -4 & 1 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & \cdots & \cdots & \cdots & 0 & 1 & 1 & -4 & 1 & 1\\
0 & 0 & 0 & 0 & 0 & 0 & \cdots & \cdots & \cdots & 0 & 0 & 1 & 1 & -4 & 1\\
0 & 0 & 0 & 0 & 0 & 0 & \cdots & \cdots & \cdots & 0 & 0 & 0 & 1 & 1 & -4
\end{bmatrix}$$

当初始条件 $\boldsymbol{Z}(0)=[\mu_1(0),\mu_2(0),\cdots\mu_l(0)]^T$，那么从(4-41)式可得

$$\boldsymbol{Z}=\boldsymbol{Z}_0+\mathrm{e}^{\boldsymbol{F}t}\cdot[\boldsymbol{Z}(0)-\boldsymbol{Z}_0] \tag{4-42}$$

而 Z_0 是方程(4-41)的稳定点：

$$\boldsymbol{Z}_0=-\boldsymbol{F}^{-1}\boldsymbol{h} \tag{4-43}$$

如果(4-42)式中 $\boldsymbol{F}$ 为负定，这一点从 $\boldsymbol{F}$ 矩阵中很容易得到，因为在对角元上的值都是大于其他元上的值，而且是负的。因而(4-42)式得到的 $\boldsymbol{Z}$，将收敛到 $\boldsymbol{Z}_0$。很明显，$\boldsymbol{F}$ 矩阵的系数与输入样本的分布很有关系，我们一开始假设的样本 $x(t)$ 是均匀分布，因而最后得到的 $\boldsymbol{Z}_0$ 如图 4-12b 所示，μ_i 的分布也近似为一种均匀分布。从(4-39)式看，在 $\langle\dot{\mu}_i\rangle=0$ 时：

$$\mu_i\approx E\{x/x\in N_i\}$$

因而 μ_i 的分布可以近似与 $x\in N_i$ 内的均匀分布相似，因此而输入样本足够多，而在输出神经元数也足够多的情况下，其权的分布近似于输入样本的分布，这样输出 y_i 的位置与输入的分布有关。这是一种十分有趣的结果。

3．自组织学习在分类上的分析

如前所述，当一些随机样本输入到 Kohonen 网络时，如果样本足够多，那么在权的系数分布上可以近似于输入随机样本的概率密度分布，在输出神经元上也反映了这种分布，即概率大的样本集中在输出空间的某一个区域，如果输入的样本有几种类型，则它们各自会根据

其概率分布集中到输出空间的各个不同的区域，同一个区域内，代表同一类的样本，当然这个区域可以逐步减小，使区域划分越来越明显，在这种情况下，不管输入样本是多少维的，都可以投影到低维的数据空间的某个区域上，这种形式也称作为数据压缩，同时如果在高维空间中比较相近的样本，则在低维空间中的投影也比较相近，这样就可以从中取得样本空间中比较多的信息。我们再用一个简单的例子来说明这种方法可以与贝叶斯分类器相似，也可以得到比较好的划分。

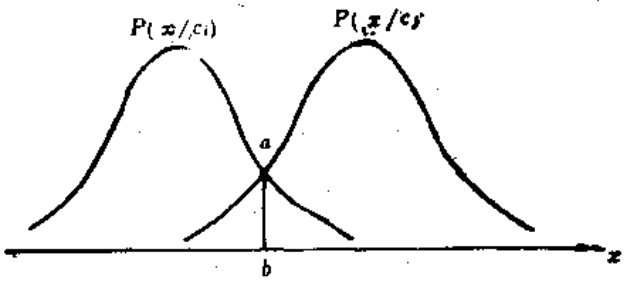


图 4-13　贝叶斯分类

考虑一个两类的分类器，初始权可以通过自组织学习的方法，得到与输入样本相对应的样本密度分布，现在进一步可用自组织学习方法得到两类样本的最优分界面。如果两类样本的输出密度分布存在着重叠，如图 4-13 所示，用 c_i 表示第 i 类，用 c_j 表示为第 j 类，根据理论上的贝叶斯分类器，如果已知 c_i 和 c_j 的先验概率分别为 $P(c_i)$ 和 $P(c_j)$，而条件概率分别为 $P(x/c_j)$ 和 $P(x/c_i)$，那么在图 4-13 中的最优分界面为

$$P(x/c_j)P(c_j)=P(x/c_i)P(c_i)$$

则在图中使两边错误概率为最小的分界线为直线 ab。如果把 c_i 类和 c_j 类反映到 Kohonen 网络的输入样本上，输入

$$x=(x_1,x_2)^T$$

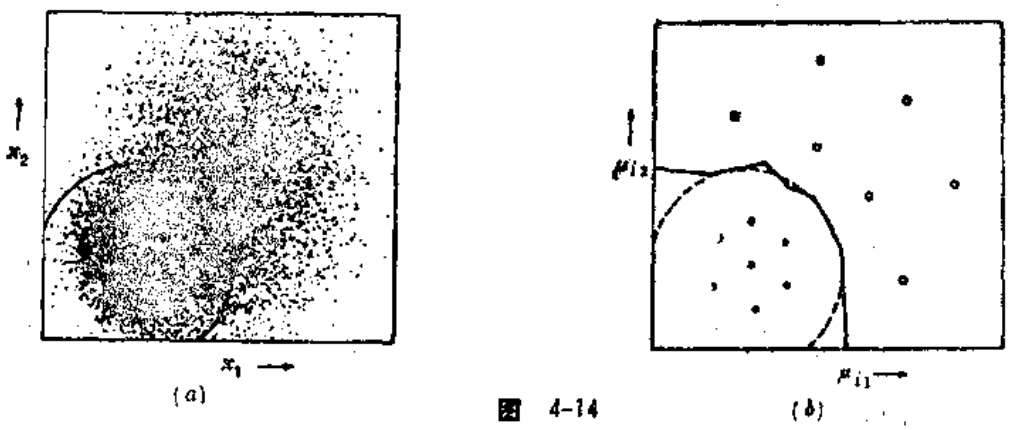


图 4-14

(a) 输入样本的贝叶斯决策；　(b) Kohonen 算法的决策

如图 4-14a 所示，它是一个二维平面上的一些点，它们是按图 4-13 的分布形成的，白的"+"为 c_i 类的中心，c_j 类的中心被埋在黑点中，从图中可以看到这两类样本在界面上是有交叠的，图中黑色的圈表示了理想的贝叶斯决策界面。采用 Kohonen 的学习方法，如果已知 x 是属于那一类的，那么可以把(4-32)式写为：

若 x 属于正确分类，且 $i\in N_c$，则有

$$m_i(t+1)=m_i(t)+\alpha(t,r)[x(t)-m_i(t)] \tag{4-44a}$$

若 x 不属于正确分类，且 $i\in N_c$，则有

$$m_i(t+1)=m_i(t)-\alpha(t,r)[x(t)-m_i(t)] \tag{4-44b}$$

$$m_i(t+1)=m_i(t) \quad i\notin N_c \tag{4-44c}$$

对于 $m_i(t)$ 的调整结果是使那些分布在交叠区域内的矢量，向着正确分类的那一面靠近，这样通过 Kohonen 网络的学习，使交叠区内能在竞争中获胜的神经元数越来越少，其分界面(这里是实线)也变得越来越明确，图 4-14b 描绘的是通过(4-44)式算法学习后的权 μ_{i1} 与 μ_{i2} 的情况，其学习后的决策分界面用实线表示，贝叶斯决策面用虚线表示，两者比较类似。可以想象，如果输出神经元数足够大，那么在输出平面上可以得到各类样本的映照区，而在两个十分接近的映照区中，即使产生了样本交叠，也可以通过(4-44)式的学习达到较优的划分，这样 Kohonen 网络可以用于分类或特征的映照。

三、Kohonen 网络的应用举例

Kohonen 网络应用十分广泛，这里我们举三个例子来说明它在拓扑映照、分类等方面的应用。

1. 字母的拓扑排序

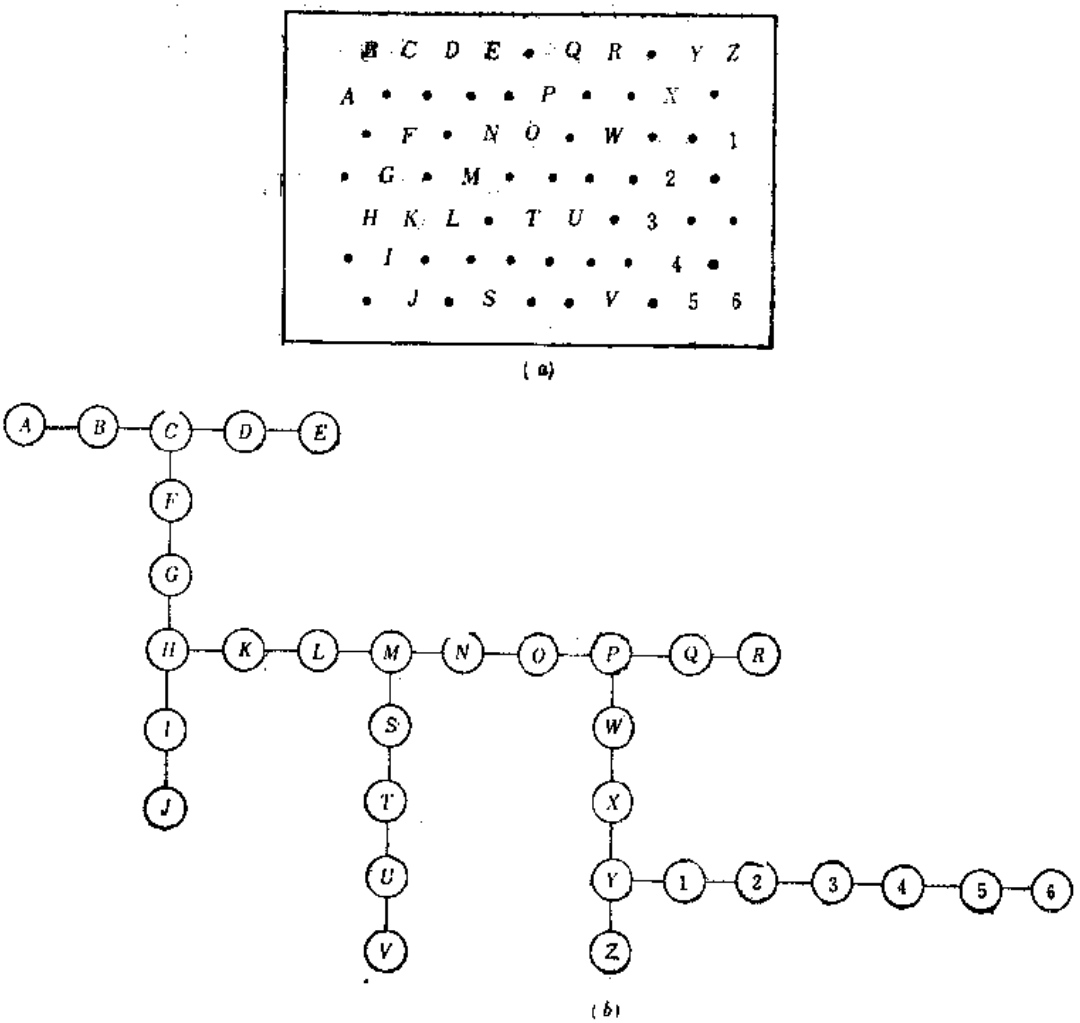


图 4-15

(a) Kohonen 网络输出对应于输入样本的映照　　(b) 映照输出的拓扑结构

输入为 26 个英文字母和 1、2、3、4、5、6数字，共有 32 个样本，它们各自用5维矢量来表示

如下的数据矩阵：

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	1	2	3	4	5	6
d_1	1	2	3	4	5	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
d_2	0	0	0	0	0	1	2	3	4	5	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
d_3	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	3	3	3	3	3	3	3	6	6	6	6	6	6	6	6	6	6
d_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	4	1	2	3	4	2	2	2	2	2	2
d_5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	4	5	6

它们通过自组织学习后得到的结果如图 4-15a 示，其中"·"表示对任何输入样本都不发生兴奋，输入的字母没有语义，但经自组织学习后形成比较有规则的拓扑结构，$A\,B\,C\,D\,E$ 为一类，它们的 $d_2=d_3=d_4=d_5=0$，$F\,G\,H\,I\,J$ 为一类，其拓扑结构与输入样本空间是完全一致的，如图 4-15b 所示。

2. TSP 问题

在第三章中，我们利用 Hopfield 网络来解 TSP 问题，权的设计是采用最优化的方法得到的，在 Kohonen 网络中，我们利用自组织学习方法得到在输出神经元空间中的拓扑映照。如果把输入样本看作是一个城市的集合，其中每一个城市是定位在二维平面上的，如同地图上表示城市的位置坐标一样，在集合C中每个样本用 $C_j(x_j,y_j)$ 来表示，因此这种网络的输入为两个神经元，表示城市的 x、y 坐标，这种网络的输出为 100 个以上的神经元，组成了一个平面。网络中的联接权初始时为一个随机数，按本节前面的算法进行学习，其邻域 N_c 一开始取得很大，而随时间减小，$\alpha(t,r)$ 也随之改变。由于 Kohonen 网络中的映照特性和聚类特性，使得在地图上相近的城市，在输出平面中的位置也比较相近，把相近的点连成链状，这个链无疑是按照城市远近排列的，于是，这个链便是 TSP 问题的解了，具体如图4-16 所示。

图 4-16 TSP 问题的解

3. 声音信号识别

Kohonen 研制了一台可以听写的打字机，他是使用自组织网络实现音素的分类功能的。芬兰语有 21 个音素，其中 3 个爆破音很难区分，就计为一类，他把语音分为 19 类，输出用 8×12 个二维神经元阵列，声音信号的预处理是用 5.3Hz 低通模拟滤波器去噪声以及前置放大来完成的。用 12bit 的 A/D 转换器，采样频率为 13.02kHz，并且用 DFT 求其频谱，然后对数化、归一化，形成了 15 个实数谱系数为 9.33 ms 时间内声音信号的特征，作为网络的输入。每个音素取 50 个样本进行训练，其输出神经元的结果如图 4-17 所示，绝大多数输出神经元给出了唯一的答案，个别神经元代表两种样本。学习后的网络，对于连续语音输入后在输出神经元上得到了响应，经过后处理程序把音素组合成单词和句子，全部硬件用两个 TMS320C10 芯片和一个神经网络组成，在 PC 机上训练，权在 TMS320C10 中，该网络组成的系统与打字机相联形成一个听写系统，当发言者发出声音后，打字机可直接打出文字，对不同的人，系统要求对其 100 个单词进行十分钟的学习，用 1000 个单词识别，其识别率为 93～98%。

Kohonen 自组织映照网络能将输入映照到低维空间，减少了维数，其聚类性、排列性类

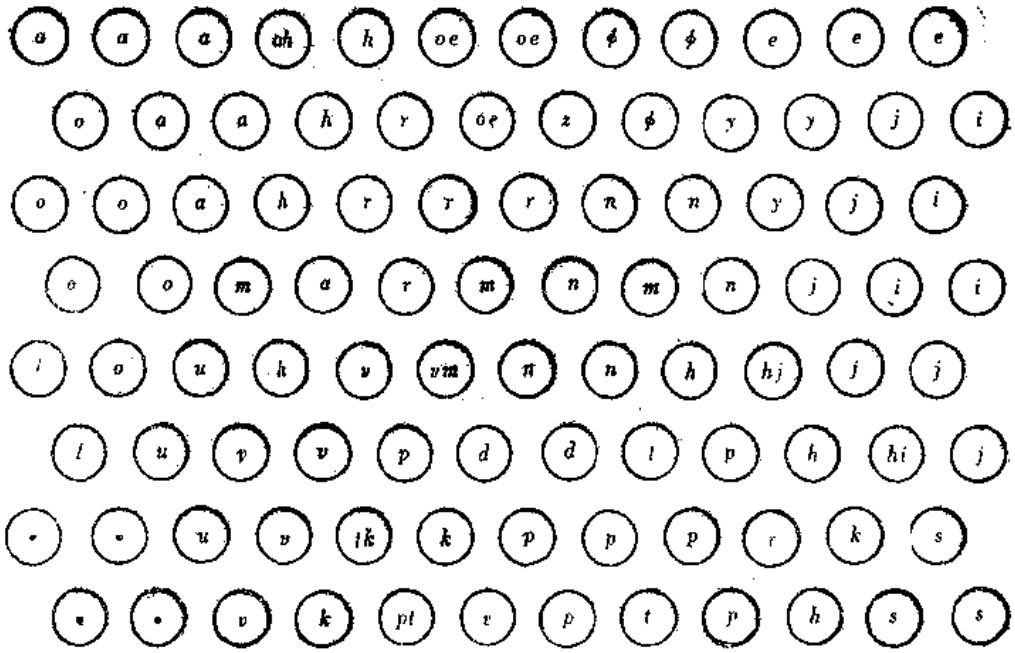


图 4-17　音素学习后在输出平面上的映照结果

似于生物系统，这种网络在对随机信号的分类和处理上具有显著的优势，因而获得了较广泛的应用。遗憾的是，网络在高维映照到低维时会出现畸变，压缩比越大，其畸变尤甚；另外网络要求的输出神经元数很大，因而其联接权数也很大，这使它比其他神经网络（如 B-P 网络）的规模要大。

第四节　Fukushima 网络模型

Fukushima 网络是一种用于模式识别的人工视觉神经网络模型，它对于输入样本的位移、畸变和噪声干扰等都具有较强的抵抗能力。网络是由若干结构相同的神经元模块加上输入层而构成的，由于处理的是视觉信息，所以每个模块都是由一些不同的二维阵列的神经元层组成的。网络采用自组织竞争算法进行学习，自动调节一些神经元的可变的联接权，并对学习样本集进行分类。学习结束后，联接权不再发生变化，这时，它们中存贮的就是关于学习样本集的一些特征信息。所以网络的学习过程也就是一个从学习样本中摘录特征的过程，而网络的识别就是对输入的识别样本进行多层的特征匹配，最后决定识别样本是否属于学习样本集中的一类。

Fukushima 网络包括两个模型：人工神经认识机模型（即 Neocognitron）和基于人工神经认知机的有选择注意力的识别模型（Selective attention）

一、人工神经认知机模型

1. 人工神经认知机中的神经元

人工神经认知机模型中有四种不同特性的神经元：s 神经元、c 神经元、V_s 神经元和 V_c 神经元。它们的输入、输出值均为非负的模拟量。

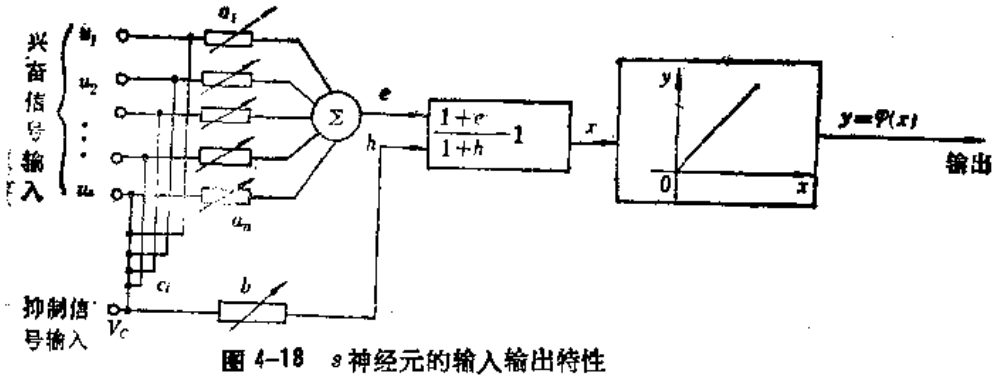


图 4-18 s 神经元的输入输出特性

s 神经元有很多兴奋信号输入端和一个抑制信号输入端，如图 4-18 所示，其输出可表示为

$$y=\varphi\left(\frac{1+\sum_{i=1}^{n}a_i u_i}{1+bV_c}-1\right) \tag{4-45}$$

$$\varphi(x)=\max(x,0) \tag{4-46}$$

在(4-45)式中，a_i 是 s 神经元的可调兴奋权重，b 为可调抑制权重。兴奋信号的加入可以增大 s 神经元的 y 输出，抑制信号的加入使 s 神经元的输出值 y 减小。这是一个分流模型，是由福岛(Fukushima)在早期提出的。

记加入到兴奋信号输入端的信号加权和为 e，加到抑制信号输入端的信号加权和为 h，即

$$e=\sum_{i=1}^{n}a_i u_i \tag{4-47}$$

$$h=b\cdot V_c \tag{4-48}$$

则(4-45)式可写为

$$y=\varphi\left(\frac{1+e}{1+h}-1\right)=\varphi\left(\frac{e-h}{1+h}\right) \tag{4-49}$$

当抑制输入很小时($h\ll1$)，则有 $y=\varphi(e-h)$，这时 s 神经元的特性与模拟阈值元件的特性一致，即当 $e>h$ 时，$y=e-h$，$e<h$ 时，$y=0$。随着学习的进行，可调权重 a_i 和 b 将逐渐增大，使 $e\gg1$，$h\gg1$，这时有

$$y\approx\varphi\left(\frac{e}{h}-1\right),$$

如果在学习时规定 a_i 和 b 以同一比例增大，则 s 神经元的输出 y 将稳定在某个值附近而不会发散。

若将兴奋输入 e 和抑制输入 h 写成

$$e=\varepsilon x;\qquad h=\eta x$$

则当 $\varepsilon>\eta$ 时，而且 $\frac{1+e}{1+h}>1$时，有

$$y=(\varepsilon-\mu)\frac{x}{1+\eta x}=\frac{\varepsilon-\eta}{2\eta}\left(1-\frac{1+\eta x-2\eta x}{1+\eta x}\right)$$

$$=\frac{\varepsilon-\eta}{2\eta}\left(1+\frac{\sqrt{\eta x}-\frac{1}{\sqrt{\eta x}}}{\sqrt{\eta x}+\frac{1}{\sqrt{\eta x}}}\right)=\frac{\varepsilon-\eta}{2\eta}\left(1+\frac{e^{\frac{1}{2}\ln\eta x}-e^{-\frac{1}{2}\ln\eta x}}{e^{\frac{1}{2}\ln\eta x}+e^{-\frac{1}{2}\ln\eta x}}\right)$$

$$=\frac{\varepsilon-\eta}{2\eta}\left[1+\mathrm{th}\left(\frac{1}{2}\ln\eta x\right)\right]=\frac{\varepsilon-\eta}{2\eta}\left[1+\mathrm{th}(\beta)\right] \tag{4-50}$$

式(4-50)表明，y 与 x 的关系也是一种单调上升的函数关系，如图 4-19b 所示，y 是从 0 到 1 的有界函数；式(4-50)中的双曲函数如图 4-19a 所示。这与神经生理学中用于描述动物感觉系统的经验公式是一致的。

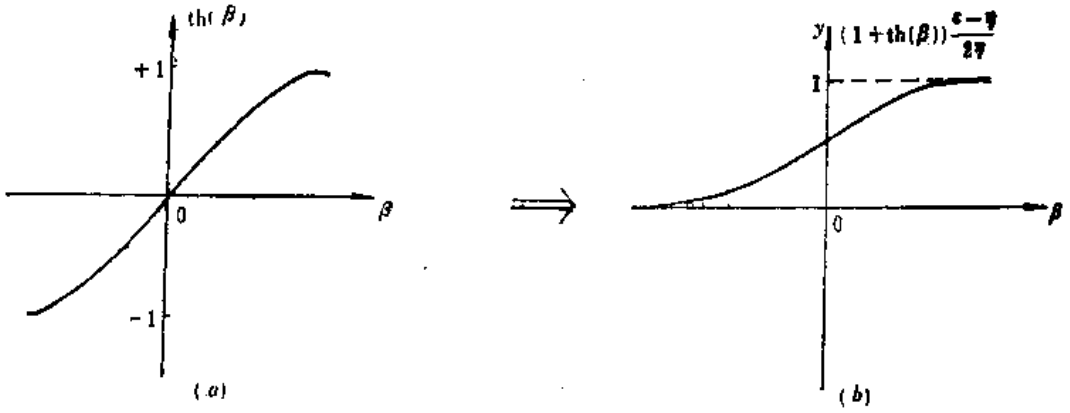


图 4-19　输出函数的曲线

c 神经元的结构与 s 神经元相似，如图 4-20 所示，其输入也是由两部分组成，兴奋端为 $u_{s1}, u_{s2}, \cdots, u_{sn}$，抑制端为 V_s，满足下列关系：

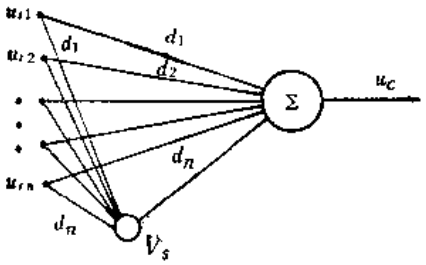


图 4-20　c 神经元的特性

$$u_c=\psi\left(\frac{1+\sum_{i=1}^{n}d_i u_{si}}{1+V_s}-1\right)$$

$$\psi(x)=\begin{cases}\dfrac{x}{\alpha+x} & x\geqslant 0\\ 0 & x<0\end{cases} \tag{4-51}$$

d_i 为兴奋联接权，抑制联接权为 1，α 为常数。

这里，d_i 是 u_{si} 到 u_c 的权，一般情况下在人为设定后就不变动了。

s 神经元和 c 神经元的输出都只能送到其他神经元的兴奋信号输入端。而 V_s 和 V_c 神经元的输出则只能送到其他神经元的抑制信号输入端。

V_s 神经元和 V_c 神经元的输入、输出特性较复杂。V_s 神经元只有兴奋信号输入端，并且它的输出正比于输入信号的加权和，V_c 神经元也只有兴奋信号输入端。它们的公式为

$$V_c=\sqrt{\sum_{i=1}^{n}c_i u_{ci}^2} \tag{4-52}$$

$$V_s=\frac{1}{K_L}\sum_{k_l=1}^{k_L}\sum_{i=1}^{n}d_i u_{si} \tag{4-53}$$

2. 网络的结构

一个人工神经认知机单元包括三个串联的模块：

$$u_0\rightarrow\underbrace{u_{s1}\rightarrow u_{c1}}_{\text{模块 1}}\rightarrow\underbrace{u_{s2}\rightarrow u_{c2}}_{\text{模块 2}}\rightarrow\underbrace{u_{s3}\rightarrow u_{c3}}_{\text{模块 3}}$$

每个模块包括两层不同类型的神经元，u_s 为 s 神经元，u_c 为 c 神经元，u_0 为外界光感受器的输入，如果把 V_s 和 V_c 神经元都考虑进去，其结构如图 4-21 示。图中所有的"○"表示一个神经元层，用大写字母 U_s U_c 表示 s 神经元层和 c 神经元层，共有三个模块 U_1，U_2，U_3，这里 V_{s1}，V_{s2}，V_{s3} 为不同模块中抑制神经元 V_s 的组合，V_{c1}，V_{c2}，V_{c3} 为不同模

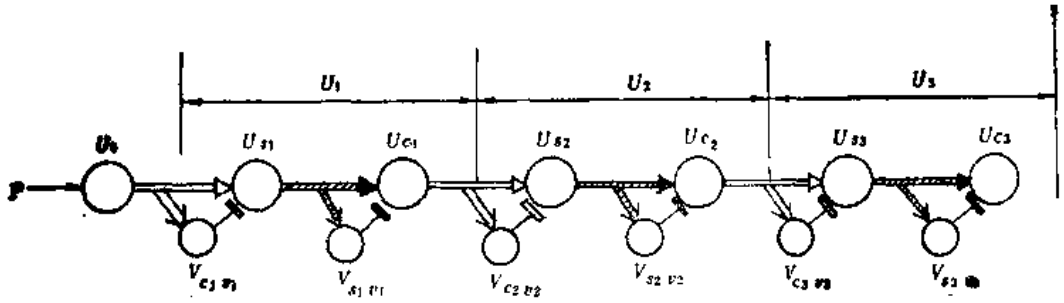


图 4-21　网络的结构图

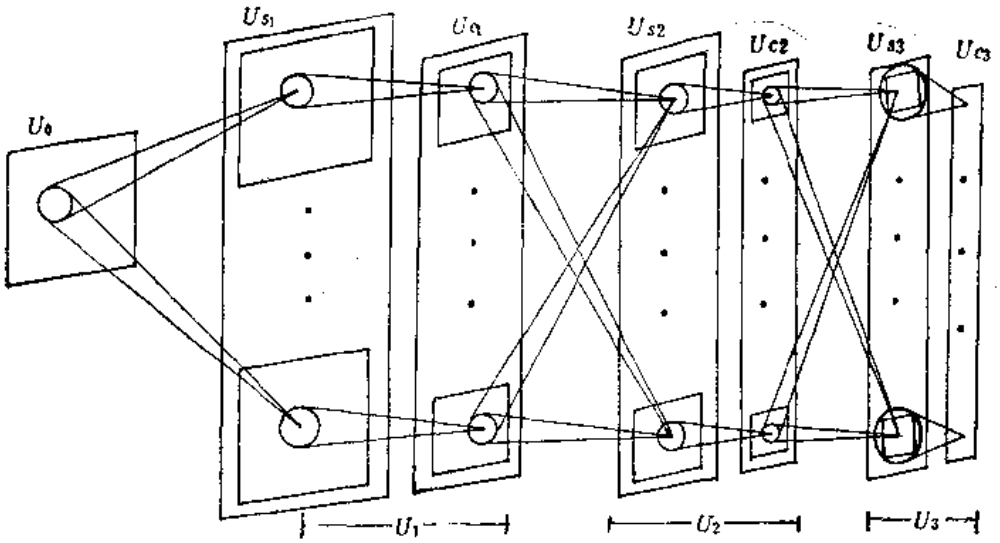


图 4-22　神经认知机中层与层间的联接

块中 V_c 神经元的组合。"⊿"表示兴奋输入，"▲"表示抑制输入，空心的"箭头"和"平头"表示权可调，如果把图 4-21 画成神经元层的形式，略去神经元 V_s 和 V_c 只画出 U_s 和 U_c，如图 4-22 所示。可以看到每个模块都由 s 神经元组成的 U_s 层和 c 神经元组成的 U_c 层构成，每个神经元层又可分成若干个子神经元平面，用 k_L 表示第 L 模块子平面的序号。模型设定：同一个子平面上的神经元都有相同的输入权重，在神经认知机中，只有 s 神经元的输入联接权重是可调的，其他神经元的输入权重都是固定的。从图 4-22 可以看出，网络是前馈式的，每个神经元只与前一层中的神经元相联，同时不是全联接的。在 U_{cL} 中第 k_L 子平面中的 c 神经元 u_{cL} 只与前一层 U_{sL} 中相应的第 k_L 子平面上某一个区域 D_L 相联，用坐标 $\boldsymbol{n}$ 表示 U_{cL} 的坐标，它与前一层的 $\boldsymbol{n}+D_L$ 中的神经元相联，图 4-23a 表示这种联接的区域。而 U_{sL} 中 s 神经元是与前一层 U_{cL-1} 层中每个子平面上对应的区域 S_{L-1} 内的神经元相联。我们用 $u_{sL}(k_L,\boldsymbol{n})$ 表示第 L 个模块中第 k_L 个 s 神经元子平面，坐标为 $\boldsymbol{n}$ 的 s 神经元的输出，如图 4-23b 所示，其余表示依此类推。利用公式(4-45)，则 s 神经元的输出为

$$u_{sL}(k_L,\boldsymbol{n})=r_L\varphi\left[\frac{1+\sum_{k_{L-1}=1}^{k_{L-1}}\sum_{V\in S_{L-1}}a_L(k_{L-1},V,k_L)\cdot u_{cL-1}(k_{L-1},\boldsymbol{n}+V)}{1+\frac{r_L}{1+r_L}b_L(k_L)V_{cL}(\boldsymbol{n})}-1\right] \tag{4-54}$$

其中 $\varphi(x)$ 由式(4-46)表示。在 $L=1$ 时，$u_{cL-1}(k_{L-1},\boldsymbol{n})$ 即为 $U_0(\boldsymbol{n})$，且 $k_{L-1}=1$，$a_L(k_{L-1},$

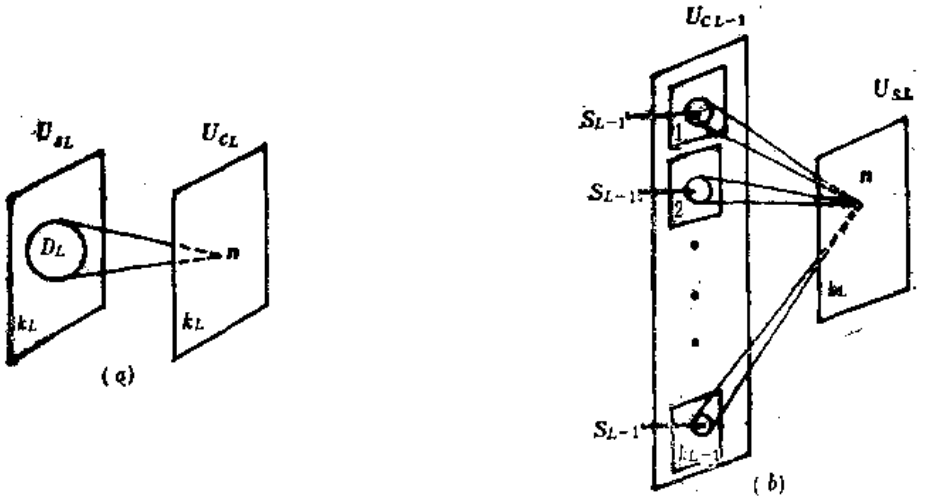


图 4-23

(a) U_{cL} 对 U_{sL} 的联接；　　(b) U_{sL} 与 U_{cL-1} 的联接

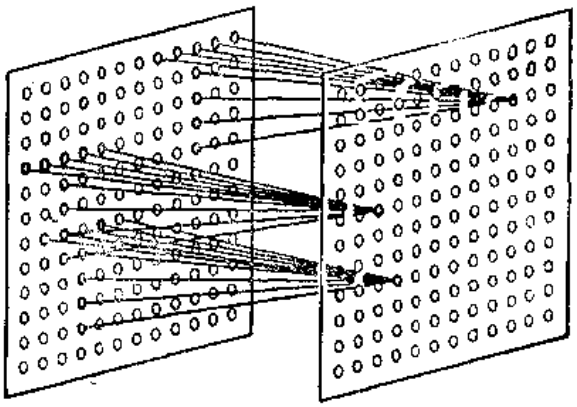


图 4-24　在同一子平面上平移不变

V, k_L)和$\tilde{b}_s(k_L)$表示 s 神经元的兴奋输入权和抑制输入权。对于同一个子平面 k_L 上的神经元都有相同的输入权重，因而 s 神经元输入权的表示与 n 无关。这种同一子平面上的神经元的输入权相同，可以保证在图像输入时的平移不变性，如图 4-24 所示，不管输入字母"T"在什么位置上，在某一个 k_L 子平面上总有一个神经元接受到比较大的输入信号，因此一个子平面表示一个输入图像的特征摘录。参数 r_L 用以控制输入和 s 神经元摘录特征之间允许的差异，将在稍后作较详细的讨论。

向 s 神经元送抑制输入信号的 $V_{cL}(n)$ 神经元，其输入与 $U_{sL}(k_L, n)$ 的输入相同，利用公式(4-52)，其输出为

$$V_{cL}(n)=\sqrt{\sum_{k_{L-1}=1}^{k_{L-1}}\sum_{V\in SL}c_{L-1}(V)U^2_{c_{L-1}}(k_{L-1},n+V)} \tag{4-55}$$

式中 $c_{L-1}(V)$ 为固定联接权，随$|V|$的增大而单调递减，并且满足：

$$\sum_{k_{L-1}=1}^{k_{L-1}}\sum_{V\in SL}c_{L-1}(V)=1 \tag{4-56}$$

从 s 神经元到 c 神经元的联接权是固定的。如图 4-22 所示，每个 c 神经元只和同一模块中具有相同序号 k_L 的 s 神经元平面上的一个邻域 S_L 相连，固定联接权的设计使得只要

在连接区 S_L 内有一个 s 神经元兴奋，c 神经元即兴奋。由于同一神经元子平面上所有 s 神经元的输入联接权是一样的，即它们摘录的是不同位置上的同一特征，所以在输入样本略有位移时，原来兴奋的 c 神经元仍将保持兴奋。因此，c 神经元和作为它输入的 s 神经元摘录的特征是相同的，但 c 神经元对特征位移的敏感性却减弱了。c 神经元的输出为

$$u_{cL}(k,n)=\psi\left[\frac{1+\sum\limits_{V\in DL} d_L(V)\cdot u_{sL}(k_L,n+V)}{1+V_{sL}(n)}-1\right] \tag{4-57}$$

$\psi(x)$由式(4-51)定义。

V_s 神经元给 c 神经元抑制输入信号，利用公式(4-58)其输出为

$$V_{sL}(n)=\frac{1}{K_L}\sum_{k_L=1}^{k_L}\sum_{V\in DL} d_L(V)u_{sL}(k_L,n+V) \tag{4-58}$$

式中 $d_L(V)$ 为固定联接权，随 $|V|$ 增加而递减。

s 神经元和 c 神经元的输入连接区域 s_L 和 D_L 设置成随 L 的增大而增大。

3. 网络的自组织学习

人工神经认知机网络的学习采用无教师的自组织竞争算法。在学习过程中，竞争体现在两个方面：

(i) 对于同一层 U_{sL} 的 K_L 个子平面中，同一个区域 S_L，最多只能有一个神经元兴奋，而其他都被抑制。

(ii) 对于同一个子平面 k_L 上最多只能有一个神经元兴奋，其他都被抑制。那些最后兴奋的神经元称为"代表"，权的学习只在兴奋的 s 神经元的输入权上进行，一旦学习完成，那个"代表"所在的子平面上的其他神经元的输入权被完全复制，因此同一个子平面上的神经元都有相同的输入权。

其过程是这样进行的：在学习样本输入后，从每个 s 神经元层中选出一些"代表"，先比较同一个 S 神经元层内不同神经元子平面上具有相同或几乎相同输入连接区的 s 神经元的值，选出输出最大的神经元作为候选；然后再在有候选神经元的神经元平面内比较各个候选神经元的值，选出输出最大的候选神经元作为"代表"，没有候选神经元的神经元子平面不产生"代表"。

"代表"神经元的输入联接权将根据其输入进行调整。设 $u_{sL}(k_L,n)$ 被选为"代表"，则有

$$\begin{aligned}\Delta a_L(k_{L-1},V,k_L)&=q_L\cdot c_{L-1}(V)\cdot u_{cL-1}(k_{L-1},n+V)\\ \Delta b_L(k_L)&=q_L V_{cL}(n)\end{aligned} \tag{4-59}$$

其中 q_L 为正常数，其大小决定了学习的速度。与"代表"神经元同子平面的其余 s 神经元的联接权作同样的调整。

由于每个神经元子平面最多只能产生一个"代表"，这就保证了每个神经元子平面都只对一个特定的输入敏感；而候选步骤的加入则避免了多个神经元子平面摘录同一特征的现象。

下面用一个简单的例子来说明这种网络的自组织学习的过程。图 4-25a 所示的是一个简单的两层自组织的网络，输入为 9 个神经元组成的 U_0 平面；而输出为 U_s 层，它由三个子平面组成，取子平面中的 U_a、U_b 和 U_c 表示此子平面中的神经元，V 是抑制输出神经元，它接收输入到 U_0 来的信号，输出与 U_a、U_b、U_c 相连，其权为 b_a、b_b、b_c。初始权为 a_{ia}、a_{ib}、a_{ic}，$i=1,2,\cdots,9$，是小的正数，而 b_a、b_b、b_c 取为 0。如一开始用图 4-25b 中的样本(i)输入，由于初始值为 0，因此根据式(4-54)得

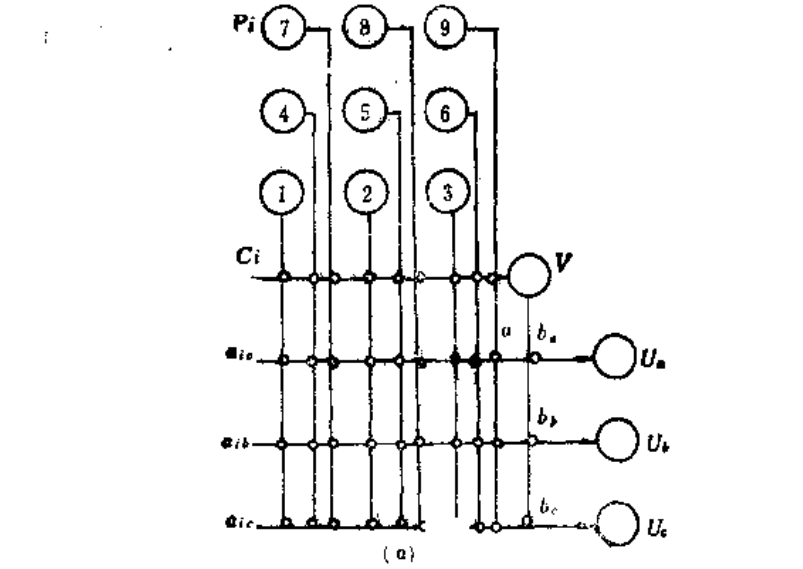


图 4-25

(a) 两层自组织网络; (b) 输入样本

$$U_x=r\varphi\left(\frac{1+\sum a_{ix}P_i}{1+\frac{r}{1+r}b_xV}-1\right)=r\sum_{i=1}^{9}a_{ix}P_i>0,\quad x=a,b,c$$

即 U_a、U_b、U_c 都兴奋，按上面竞争的原则，只取其中最大的为"代表"，如其"代表"为 U_a，那么只有 a_{ia} 和 b_a 的权可以调整，按照(4-59)式得

$$\Delta a_{ia}=q\cdot c\cdot P_i$$

$$\Delta b_a=q\cdot V$$

如果 q 很大，那么在样本(i)中不等于零的 P_8、P_5、P_2 对应的权 a_{8a}、a_{5a}、a_{2a} 变大，$b_a\fallingdotseq 0$，因此 U_a 是对应着(i)的这种样本。当样本(ii)输入，此时 $b_a\fallingdotseq 0$，$b_b=b_c=0$，因此在(4-45)式中，U_b 和 U_c 对样本(ii)的响应大，而 U_a 因其分母 b 很大，而可能使 $\varphi(\cdot)=0$，这样样本(ii)会被 U_b 或 U_c 选中，一旦被 U_b 选中，那么样本(iii)必定只能被 U_c 选中而作为其"代表"。因此竞争的算法使每个子平面只能对一个特征敏感，不可能出现两个子平面同时响应一个特征的现象。这里也可以看到，s 神经元层的作用主要是通过权的学习，使它上面每个子平面分别摘录输入样本不同位置上的特征，通过三个模块的特征组合来识别二维的文字和图形。

4. s 神经元的特征检测性能

由于各模块的结构是相同的，因此在这里的叙述中略去下标。考虑任一被选为"代表"

的 s 神经元，设此时联接区内的输入值为 $P(V)$，根据(4-55)、(4-59)式，则其联接权改变为

$$\Delta a(V)=q\cdot c(V)\cdot P(V)$$
$$\Delta b=q\sqrt{\sum_V c(V)\cdot P^2(V)}$$

设该 s 神经元联接权经 N 次加强，由于其初值为一很小数，故将其忽略，有

$$a(V)=N\cdot q\cdot c(V)P(V) \tag{4-60}$$
$$b=N\cdot q\cdot\sqrt{\sum_V c(V)P^2(V)} \tag{4-61}$$

设目前该 s 神经元的输入为 $p(V)$，则其输出为

$$u=r\cdot\varphi\left[\frac{1+\sum_V a(V)p(V)}{1+\frac{r}{1+r}b\cdot V}-1\right] \tag{4-62}$$

令

$$T=\frac{\sum_V a(V)\cdot p(V)}{b\cdot V} \tag{4-63}$$

在 $a(V)$ 和 b 都足够大时，式(4-62)可化为

$$u\approx r\varphi\left[\frac{r+1}{r}T-1\right] \tag{4-64}$$

将式(4-60)和式(4-61)代入式(4-63)得

$$T=\frac{\sum_V c(V)\cdot P(V)\cdot p(V)}{\sqrt{\sum_V c(V)P^2(V)}\ \sqrt{\sum_V c(V)p^2(V)}} \tag{4-65}$$

若将 $P(V)$、$p(V)$ 排列成矢量形式，用 $\mathbf{P}$、$\boldsymbol{p}$ 表示，T 就代表了这两矢量在矢量空间中夹角的余弦，所以在 $\mathbf{P}=\boldsymbol{p}$ 时，$T=1$；在 $\mathbf{P}\neq\boldsymbol{p}$ 时，从式(4-64)可以看出，在输入为 $\boldsymbol{p}(V)$ 时，若 $T>(r/1+r)$，则有 $u>0$；反之，则 $u=0$，即 s 神经元不响应，我们将 s 神经元经过学习后获得的这种特性称它为对特征的检测性能。这意味着，那些能使 $T>(r/1+r)$ 的输入 $\boldsymbol{p}(V)$ 将被认为是与 $\mathbf{P}(V)$ 相似，而使 $T\leqslant(r/1+r)$ 的输入 $\boldsymbol{p}(V)$ 被认为与 $\boldsymbol{P}(V)$ 不同。

由于 $(r/1+r)$ 是 r 的增函数，所以增大 r 值可以提高 s 神经元对特征的选择性。但是，很高的选择性是不必要的，因为它减弱了 s 神经元对输入畸变的容差。也就是说，有时，我们希望在 $\boldsymbol{p}(V)$ 和 $\boldsymbol{P}(V)$ 有一定差别时，s 神经元仍能将 $\boldsymbol{p}(V)$ 看成是 $\boldsymbol{P}(V)$。所以参数 r 往往是取在选择性和容差间折衷的数值。

综上所述，s 神经元的联接权是样本 $\boldsymbol{p}(V)$ 连续输入 N 次后的数值。在实际学习时，所有的学习样本可以轮流输入，因为随着学习的进行，每个 s 神经元平面都变得只对一个与众不同的特征敏感。所以，在输入样本与 $\boldsymbol{P}(V)$ 有较大差别时，对 $\boldsymbol{P}(V)$ 敏感的 s 神经元平面将没有神经元兴奋，s 神经元的输出值较小，此时的输入端样本将不会引起该 s 神经元平面联接权的调整。

5. 网络的工作原理和计算机模拟的结果

s 神经元能够摘录特征，在 n 个模块相串而形成的多层自组织网络中，第一个模块的作用是将输入样本中的局部特征摘录到 U_{s1} 层神经元的子平面上，如图 4-26 中"A"是输入的样本，在 U_{s1} 层中对"A"字母的"$\wedge$"、"$\nearrow$"、"$\rightarrow$"、"$\searrow$"、"$\vee$"等特征敏感，而在图 4-27 中 U_{s1} 所示的五个子平面对五个特征兴奋的位置，U_{c1} 可使 U_{s1} 子平面上的特征有一定的位置容错，如

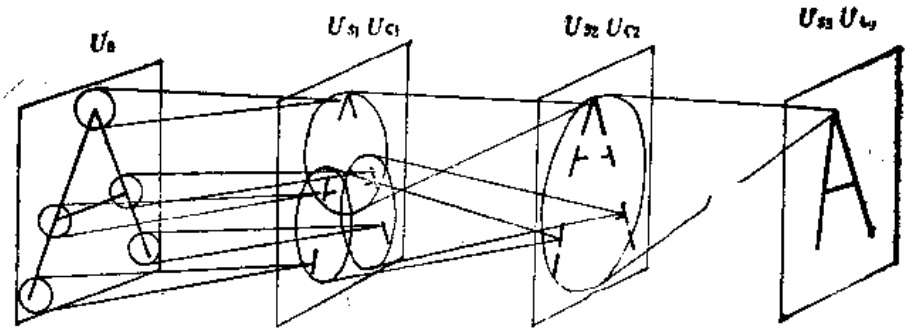


图 4-26　多层自组织网络对特征的组合

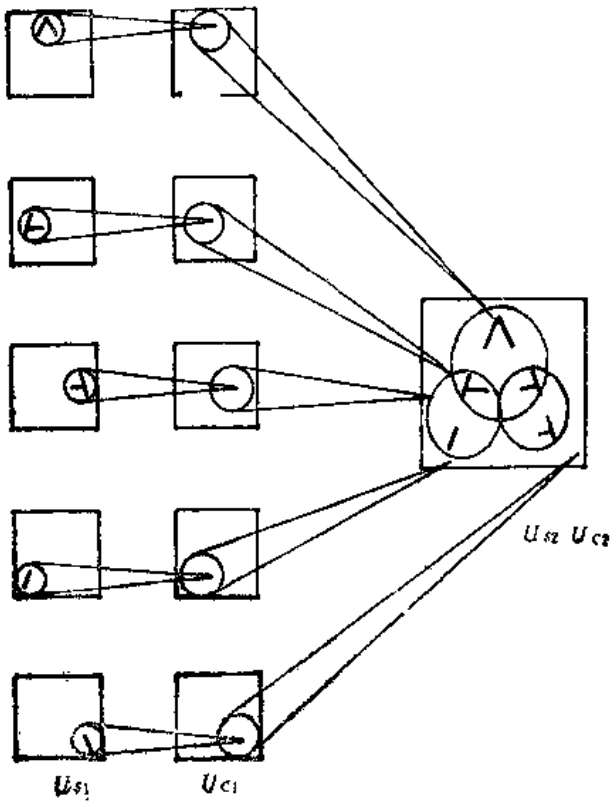


图 4-27　U_{c1} 的作用

图 4-27 所示。U_{s2} 和 U_{c2} 对 U_{s1} 各个子平面进行组合，形成了组合后的特征为"A"、"A"、"A"三个，由于 U_{s2} 上的 s 神经元对 U_{c1} 所有子平面都相联，同时联接的范围随着模块向前推进而扩大，因此当到达第三个模块时就是 U_{s2} U_{c2} 的整个子平面的组合，因而在 U_{c3} 上得到了决策。图 4-26 就表示了这种多模块的组合过程，从局部到整体，从低级到综合的进程。图 4-27 表示 U_c 平面的作用，因此这种网络对位置和畸变敏感性不强，可以识别变异的文字。

在进行计算机模拟时，建议采用分层训练，即先对低层的 s 神经元进行训练，在其输入联接权足够大后再训练高一层的 s 神经元。因为只有在低层的 s 神经元都获得了特征检测能力后，这模块中的 s 神经元(c 神经元)的输出才代表了学习样本经过这一层次的特征映照后在特征空间的表示，而这时高层 s 神经元摘录的才是学习样本的完整的特征表示。在这以

前，由于一些 s 神经元尚未获得特征检测能力，其输出是由初始权造成的，如果在这时允许高层神经元进行学习，则其摘录的则是学习样本的一个不完整的特征表示。当学习全部结束后，这一组不完整的特征表示将不会再出现，因此，摘录这些不完整的特征表示的高层 s 神经元就没有发挥作用。

图 4-28 是计算机模拟人工认知机的例子，图中同一行数字最左边一列为学习样本，其余为网络能识别的有位移、畸变和噪声干扰的认别样本，学习后对干扰平移样本都可识别。

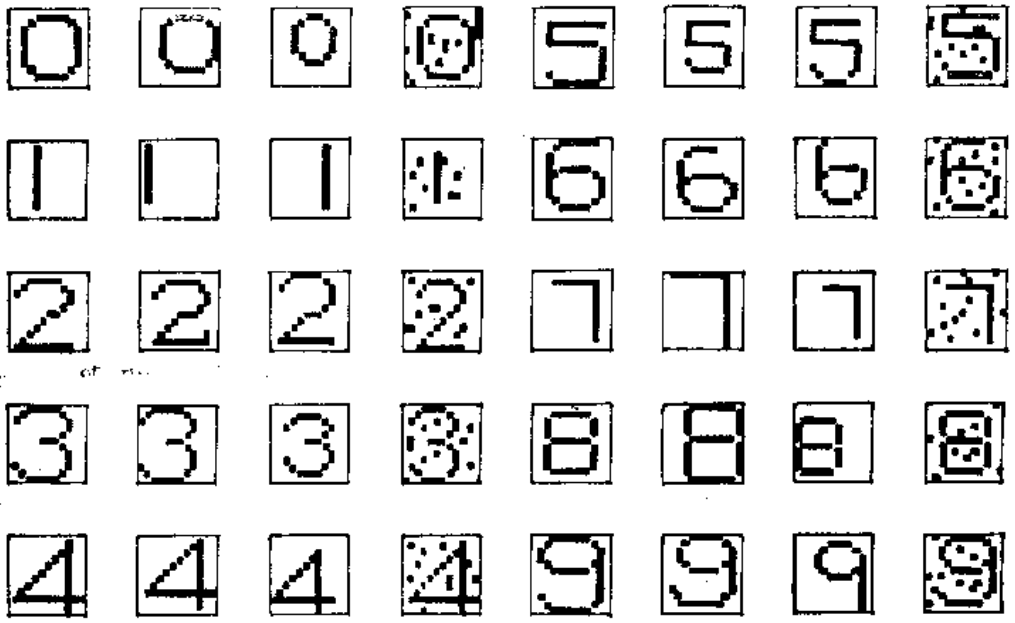


图 4-28　神经认知机的学习样本和识别样本

二、具有选择注意力的 Fukushima 神经网络

具有选择注意力的 Fukushima 网络模型是基于神经认知机模型的一个发展。选择注意力具有两方面的含义：其一，在识别样本由两个或多个学习样本构成时，网络能分时将它们识出。这就是说，网络能够有选择地先识别出样本中的一个，然后转换注意力，识别余下的样本；其二，在输入的识别样本存在噪声干扰或有缺损时，网络的注意力将集中在样本图形上，不受干扰和缺损的影响，并且通过联想记忆产生一个无噪声、无缺损的联想输出。

和人工神经感知机一样，网络采用重复模块结构，输入样本的信息在每个模块(层)上被加工处理，并通过传入联结向高层传送。同时，与神经认知机不同的是，高层神经元的响应信息将通过传出联接向低层神经元进行反馈。而传入和传出信号间还有一定的相互作用，传出信号对传入信号有保持作用，传入信号对传出信号有选通作用。

如果我们只考虑网络中的传入联接，则网络与人工神经认知机非常相似。它同样具有自组织学习功能，在重复输入学习样本集一定次数后，网络就能自动对学习样本进行分类。这表现在学习结束后，对每个不同的学习样本，最高层有而且只有一个神经元能对其作出响应。这样的神经元被称为"人工感悟细胞"，即它的状态代表了网络的识别结果。在网络的学习完成后，识别结果只与识别样本和学习样本的外形上的相似性有关，基本上不受位移、畸变和噪声的影响。

在一个识别样本是由两个学习样本组成的时候，假设最高层只有一个"人工感悟细胞"被激活，即网络识出了其中的一个样本，那么从该"人工感悟细胞"发出的传出信息将使传入通道中与被识出样本有关的神经元保持兴奋，从而保持该样本的传入信息，而使另一个样本的传入信息逐步衰减。这意味着网络的注意力只集中在一个样本上。

如果输入样本缺损了一些部分，网络会自动降低缺损区域附近进行特征检测的神经元的选择性，仔细检查缺损区域内是否有已摘录特征的细微痕迹。这样，在缺损区域不是很大时，网络能联想出一个完整的样本。与之相反，噪声则被抑制，因为网络不会将注意力选择在噪声上。

在网络的动态响应过程达到稳定后，如果我们将网络的传出信号中断一段时间，则那些由传出信号保持的神经元的响应能力将被减弱，而其他神经元由于"习惯化"而失去的响应能力将得到恢复。这时输入样本中的另一学习样本的信息就比较容易传入到高层，并激发与之相应的、新的"人工感悟细胞"，然后，新的传出信息将保持该样本的传入信息。这样，网络便实现了注意力的转移。

1. 网络的结构

网络由多层神经元组成，和人工神经感知机一样，网络中各种神经元的输出为非负的模拟量。在相邻的神经元层间，既有传入联结，又有传出联结。

网络的结构如图 4-29 所示，网络的输入层为 U_0，第 L 个模块为 U_L，将最高层记为 U_3。神经元之间的突触联接用单线或双线表示，单线表示在不同神经元层所对应的两个神经元平面间是一对一的联接，双线则表示不同层的神经元平面间是多对一的联接。在每个模块中，各种神经元如：u_s、u_{sv}、u_{sw}、w_s、w_c、w_{sv}、w_{cx} 等神经元都分布在一个二维阵列中，标记"○"就代表了一个这样的神经元层。我们用 U_{cL} 代表第 L 个模块中的 U_c 神经元层；U_{sL} 代表第 L 个模块中 u_s 神经元层($L=0,1,2,3$)，U_{c0} 为输入样本平面，样本为 P，在图 4-29 中，从 U_{c0} 到 U_{c3} 的过程同神经感知机一样，神经元子平面的大小随着 L 的增大而减小，在最高层为 U_{c3}，每个子平面只包含一个神经元。我们把这个与人工神经感知机一样的通道称为传入通道，具体由下式表示：

$$U_{c0}\rightarrow U_{s1}\rightarrow U_{c1}\rightarrow U_{s2}\rightarrow U_{c2}\rightarrow U_{s3}\rightarrow U_{c3}$$

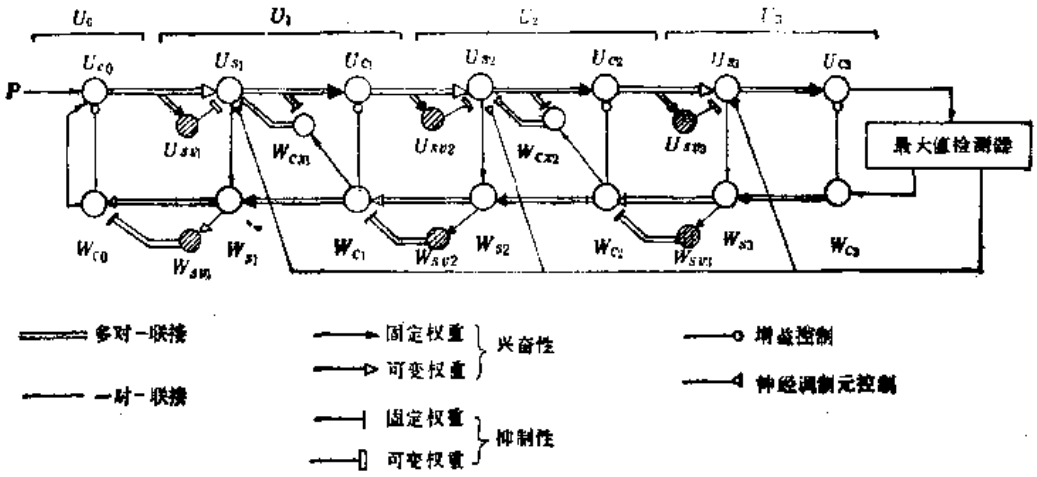


图 4-29　具有选择注意力的自组织网络结构

图 4-29 中最大值检测器是检测 U_{c3} 层中具有最大值的神经元，并对 U_{cL} 中神经元进行控制。从最大值检测器的输出还有一个返回通道，它是从 W_{c3} 开始到 W_{c0}，并回到输入层 U_{c0}：$W_{c3}\rightarrow W_{s3}\rightarrow W_{c2}\rightarrow W_{s2}\rightarrow W_{c1}\rightarrow W_{s1}\rightarrow W_{c0}$，$W_{cL}$ 与 U_{cL} 的联接结构相似，W_{sL} 与 U_{sL} 相似，W_{svL} 与 V_{svL} 相似，不同的是方向相反，其公式也不完全一样，我们称作为转出通道。在图中 W_{cxL} 是控制输出通道对输入通道效率的神经元层，为了引入图 4-29 中各层的神经元的计算公式，用标记 $u^t_{cL}c(n,k)$ 表示第 L 神经元层中第 k 个神经元子平面上坐标为 n 的 u_c 神经元在时刻 t 时的输出。其他神经元的表示与此类似。如 U_{cs}，W_{cs} 分别用 $u^t_{cs}(n,k)$，$W^t_{cs}(n,k)$ 表示。

2. 传入联接与传出联接

(1) 传入联接：如前所述，若只考虑网络的传入联接，则网络的结构与神经认知机非常相似，u_s、u_c、u_{sv} 神经元分别与 s、c、V_s 神经元对应。

在学习过程结束后，负责特征摘录的 u_s 神经元在 u_{sv} 神经元的协同作用下可以在输入中检测其摘录的特征。这就是说，每个 u_s 神经元都只有在一个特定的输入、在一个特定的位置上出现时才会兴奋，而这个特定的输入就是其摘录的特征。特征的选定是网络在自组织学习时自动完成的。在低层模块，u_s 神经元摘录的是一些局部特征，如线段、拐角等等，在高层，摘录的则是局部特征的组合。

此外，一个 u_c 神经元以一组 u_s 神经元的输出为输入，若这组 u_s 神经元中有一个兴奋，则 u_c 神经元就兴奋。所以，输入样本位置的改变对 u_c 神经元的影响较小。

这样，在传入通道中，随着层次的增高，局部特征被综合成全局特征，最高层的 u_s 神经元通过对全局特征的匹配进行样本识别；同时，由于每个 u_c 神经元层都能消除一些位移的影响，所以网络的识别效果基本上不受输入样本的位移和畸变的影响。

第 L 个模块的 u_s 神经元的输出由下式表示：

$$u^t_{sL}(n,k)=r^t_L(n,k)\varphi\left[\frac{\sigma_L+\sum\limits_{k_{L-1}=1}^{k_{L-1}}\sum\limits_{V\in AL}a_L(V,k',k)\cdot u^t_{cL-1}(n+V,k')}{\sigma_L+\dfrac{r^t_L(n,k)}{1+r^t_L(n,k)}\cdot b_L(k)\,u^t_{svL}(n)}-1\right] \tag{4-66}$$

式中 $\varphi(x)$ 由式(4-46)给出。

σ_L 是参变量，为一正常数，可决定 u_s 神经元的输入，输出的饱和特征；

$r^t_L(n,k)$ 是正变量，用以控制抑制输入信号对 u_s 神经元输出的影响；

$a_L(V,k',k)$ 为可变兴奋联接权；

$b_L(k)$ 为可变抑制联接权；

k' 为上一层的子平面序号。

抑制性神经元 u_{sv} 给 u_s 神经元一个抑制输入信号：

$$u^t_{svL}(n)=\sqrt{\sum_{k'=1}^{k_{L-1}}\sum_{V\in AL}c_L(V)\left[u^2_{cL-1}(n+V,k')\right]} \tag{4-67}$$

固定联接权 $c_L(V)$ 的取法与神经认知机相同。

u_c 神经元的输出可表示为

$$u^t_{cL}(n,k)=g^t_L(n,k)\psi\left[\sum_{V\in DL}d_L(V)u^t_{sL}(n+V,k)\right] \tag{4-68}$$

$$\psi(x)=\frac{\varphi(x)}{1+\varphi(x)} \tag{4-69}$$

固定联接权 $d_L(V)$ 的取法与神经认知机相同；变量 $g_L^k(n,k)$ 代表了 u_c 神经元的增益，其值受传出通道中相应的 w_c 神经元的控制。

(2) 传出联接：网络传入通道中的每个 u_s、u_c 和 u_{sv} 神经元，都一一对应于传出通道中位置相同的 w_s、w_c 和 w_{sv} 神经元。网络的设计使得传出信号的流动途径与传入信号相同，但方向相反。

(i) w_s 神经元到 w_c 神经元的传出联接：由于传出信号的流动方向与传入信号相反，所以我们考虑第 L 层 w_s 神经元到 $L-1$ 层 w_c 神经元间的传出联接。网络模型假设在学习完成后，第 L 层的任一个 w_s 神经元到 $L-1$ 层的任一个 w_c 神经元间的联接权等于传入通道中位置与这两个神经元相同的 u_s 神经元和 u_c 神经元间的联接权。所以 w_c 神经元的输出为

$$w_{cL}^k(n,k)=\varphi\left\{\alpha_L\left[\sum_{k'=1}^{K_{L+1}}\sum_{V\in A_{L+1}}a_{L+1}(V,k,k')\cdot w_{sL+1}^k(n-V,k')-\sum_{V\in A_{L+1}}c_{L+1}(V)w_{sV_{L+1}}^k(n-V)\right]\right\} \tag{4-70}$$

上式中 α_L 为正常数。

比较上式和式(4-66)，可见，从 w_s 神经元传出的信号的路径和传入到同一位置 u_s 神经元的信号路径相同，但信号流动的方向相反。

抑制神经元 w_{sv} 的输出为

$$w_{sV_{L+1}}^k(n)=\frac{r_{L+1}^0}{1+r_{L+1}^0}\sum_{k'=1}^{K_{L+1}}b_{L+1}(k')\cdot w_{sL+1}^k(n,k') \tag{4-71}$$

式中 r_{L+1}^0 为变量 $r_{L+1}^k(n,k)$ 的初值。这说明，从 w_{sL+1} 神经元到 w_{svL+1} 神经元的联接权和传入通道中 u_{svL+1} 神经元到 u_{sL+1} 神经元的联接权是相同的，但信号传送的方向相反。

(ii) 从 w_c 神经元到 w_s 神经元的传出联接：相应于传入通道中一个 u_s 神经元平面上的一组 u_s 神经元联接到对应 u_c 神经元平面中的一个 u_c 神经元，每个 w_c 神经元也都和一组 w_s 神经元相连。但是我们并不希望在 w_c 神经元兴奋时，所有与该 w_c 神经元相连的 w_s 神经元都兴奋。所以网络模型假设每个 w_s 神经元不仅从 w_c 神经元处接受传出信息，而且还从相同位置的 u_s 神经元处获得一个选通信号，只有两个信号都有效时，w_s 神经元才会被激活，其输出表示式为

$$w_{sL}^k(n,k)=\min\left[u_{sL}^k(n,k),\sum_{V\in D_L}d_L(V)w_{cL}^k(n-V,k)\right] \tag{4-72}$$

这样，在 w_c 神经元到 w_s 神经元的传出通道中，信号流动的路径与传入通道相同，而方向相反。

(3) 传出信号对传入信号的作用：由上述分析可见，网络的传出信号受到传入信号的影响，但是传出信号也能对传入信号产生影响，这主要表现在对传入信号有保持作用和控制 u_s 神经元抑制信号的效率。

(i) 保持作用：我们考虑输入样本中包含两个学习样本时的情况。假设最高层的 u_c 神经元已有一个兴奋，就是说网络已识别出其中的一个样本。在传入通道中，不仅相应于被识别出样本特征的神经元是兴奋的，相应于另一样本特征的神经元也是兴奋的。为了保持前一类神经元的兴奋状态，而抑制后一类兴奋的神经元，网络模型假设每个 u_c 神经元有"习惯化"特性，即 u_c 神经元的输入、输出增益是随时间递减的。同时 u_c 神经元又从传出通道相同位置的 w_c 神经元处获得一个保持信号，该信号可以使其对应的前一类的 u_c 神经元增益不衰减，增益的表达式如下：

$$g_L^t(\boldsymbol{n},k)=\nu_L g_L^{t-1}(\boldsymbol{n},k)+(1-\nu_L)\psi[\nu'_L,w_{cL}^{t-1}(\boldsymbol{n},k)] \tag{4-73}$$

式中 ν_L 为正常数，决定 u_c 神经元"习惯化"的速度，$1\geqslant\nu_L\geqslant 0$；$\nu'_L$ 也为正常数，决定保持信号的饱和特性。网络的初始增益 $g_L^0(\boldsymbol{n},k)=1$。

(ii) 控制 u_s 神经元抑制信号输入的效率：考虑 u_c 神经元被抑制，而与它位置相同的 w_c 神经元兴奋的实际情况：这意味着网络在传出通道中，联想出来的一个特征在传入通道中没有被检测到。当输入样本有缺损时，会出现这种情况。W_{cx} 神经元专门用于检查这种情况，其输出为

$$w_{cxL}^t(\boldsymbol{n},k)=\psi\Big[w_{cL}^t(\boldsymbol{n},k)-\sum_{V\in D'_L}d'_L(V)u_{sL}^t(\boldsymbol{n}+V,k)\Big] \tag{4-74}$$

因为 u_c 神经元的"习惯化"特性，所以在上式中用 u_s 神经元的输出作累加，而不直接用 u_c 神经元的输出。联接区域 D'_L 稍大于 D_L，以克服输入样本可能的位移所造成的影响。

如果一个 w_{cx} 神经元兴奋，它将送出一个神经调制信号 x_s 给相应的 u_s 神经元，x_s 用下式表示：

$$x_s^t(\boldsymbol{n},k)=\beta_L x_s^{t-1}(\boldsymbol{n},k)+\beta'_L\sum_{V\in DL}d_L(V)w_{cxL}^{t-1}(\boldsymbol{n}-V,k) \tag{4-75}$$

式中 β_L 为神经调制信号 x_s 的衰减常数，$0<\beta_L\leqslant 1$，β'_L 为另一正常数。

x_s 可以减弱 u_{sv} 神经元送到 u_{sv} 神经元的抑制输入信号对 u_s 神经元输出的影响，从而降低 u_s 神经元特征检测的灵敏度。这是通过减小式(4-66)中的 r_L^t 来实现的：

$$r_L^t(\boldsymbol{n},k)=\frac{r_L^0}{1+x_{sL}^t(\boldsymbol{n},k)+x_{xL}^t} \tag{4-76}$$

其中：r_L^0 为 r_L^t 的初始值；变量 x_{xL}^t 是受最大值检测器控制的另一类神经调制信号。在最高层模块中，不设置 w_{cx} 神经元，可以认为 $x_s=0$。

这样，w_{cx} 神经元送出的神经调制信号可以促使 u_s 神经元对有缺损的特征输入作出响应。一旦 u_s 神经元被这样激活后，传出信号就可以通过与 u_c 神经元相应的 w_c 神经元完整地向低层模块传送。

3. 输入层 U_0 上的联想与分割

我们已经知道，在识别样本包含多个学习样本时，只有一个样本的信息能从最高层反馈到 w_{c_0} 神经元层，且传出信号经过的路径与传入信号相同。即使输入的样本与学习样本相比有一些缺损，在 w_{c_0} 神经元层上出现的将是一个经过联想记忆后的样本，其外形与输入的样本相同，但缺损部分已被联想出来。因此，w_{c_0} 神经元层的输出可以看成是一个自联想记忆的结果。从另一种观点来看，w_{c_0} 神经元层的输出是对输入样本进行分割后的结果，这个输出只和输入样本中的一个样本有关，最高层 u_{cL} 神经元的输出指示了该样本的类别。

为了使网络具有较好的联想功能，网络模型假设输入样本 P 和网络的联想记忆输出 w_{c_0} 在输入层 U_0 上进行迭加：

$$u_{c_0}^t(\boldsymbol{n},k)=g_0^t(\boldsymbol{n})\cdot\max[P^t(\boldsymbol{n})\cdot w_{c_0}^t(\boldsymbol{n})] \tag{4-77}$$

增益 $g_0^t(\boldsymbol{n})$ 的计算方法和中间层 u_c 神经元的增益计算相同。

由于上述正反馈回路的引入，网络的第一次联想记忆输出将反馈给输入样本，形成新的输入，并在此基础上进行下一轮的识别。这样，即使输入样本有较大的缺损，网络仍能通过多次联想记忆逐步产生一个完整的记忆输出。

4. 最大值检测器

在两个或两个以上学习样本同时输入网络时，都可能同时激活最高层的多个 u_c 神经元，这时，最大值检测器负责检测出具有最大输出值的 u_c 神经元，并激活与该 u_c 神经元位置相同的 w_c 神经元，抑制其他 w_c 神经元。所以，最高层 w_c 神经元的输出与其他层不同，由下式给出：

$$w_{cL}^{t}(k)=\begin{cases}1 & \text{当 } u_{cL}^{t}(k)=\max\limits_{k'}[u_{cL}^{t}(k')]>0\\ 0 & \text{其他}\end{cases} \tag{4-78}$$

在传入通道中，只有与最大输出 u_{cL} 神经元有关的信息才会被传出信号保持。

多样本输入时，另一种情况是最高层的 u_c 神经元都不兴奋，在单样本输入时，如果识别样本与学习样本差别太大或干扰太大时也会出现上述情况。这时，在网络输出接入的最大值检测器就增大神经调制信号 x_{xL} 的值：

$$x_{xL}^{t}=\begin{cases}x_{xL}^{t-1}+\beta_{xL} & \text{若所有 } u_{cL}^{t-1}(k)=0\\ \beta'_{xL}x_{xL}^{t-1} & \text{其他}\end{cases} \tag{4-79}$$

其中 β_{xL}，β'_{xL} 为正常数，$0<\beta'_{xL}\leqslant 1$。从式(4-66)、(4-76)中可以看到，神经调制信号 x_{xL} 和 x_{sL} 一样，可以减弱 u_{sL} 神经元的抑制输入信号，降低其对特征的灵敏度，使 u_{sL} 神经元容易作出响应。

在最大值检测器增大 x_{xL} 值期间，网络模型假设所有 u_c 神经元的增益 g 保持不变，而不按式(4-73)变化。

5. 转换注意力

假设在多样本输入时，有一个样本已被识别，为了转换注意力到其他样本上，需要将传出信号中断一段时间，这时，各个 u_c 神经元的增益由下式给出：

$$g_L^t(n,k)=\frac{1}{1+\nu''g_L^{t-1}(n,k)} \tag{4-80}$$

ν'' 为正常数。可以看出，原来有较大增益的 u_c 神经元在转换注意力后增益减小，而原来增益被衰减的 u_c 神经元将恢复响应能力。同时，为了防止同一 u_{cL} 神经元再次被最大值检测器选中，将原来有最大输出的 u_{cL} 神经元的增益置为零。神经调制信号 x_{sL} 和 x_{xL} 的值也复位为零。经过这样处理后，原来被识别的样本信息就很难再经过传入通道进到最高层，而另一样本的信息则比较容易传入并激活一个新的人工"感悟细胞"。

6. 计算机模拟

考虑一个 $L=3$ 的网络，5 个学习样本如图 4-30 所示。在学习时，传出通道被断开，并且假设 u_c 神经元的增益不随时间变化而变化。这时，网络与神经认知机非常相似，其学习算法可以参考神经认知机模型的自组织学习过程。在网络学习结束后，所有可修改的联接权不再发生变化。

图 4-30　网络的五个学习样本

图 4-31 和图 4-32 显示了网络的一些特性。图中给出了网络 U_{c_0} 层和 w_{c_0} 层上的输出图形，每个图形左上角的数代表时间 t，标记"▼"代表该时刻中断传出信号，以转换注意力。

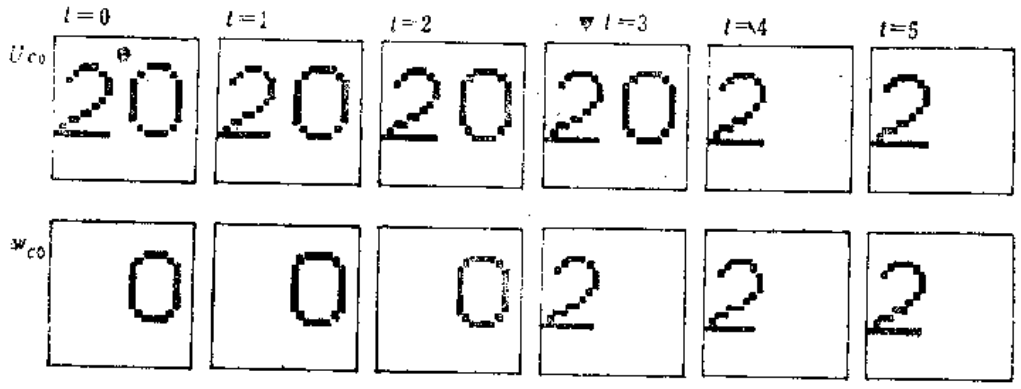


图 4-31　网络对于两个学习样本组成的识别样本的响应

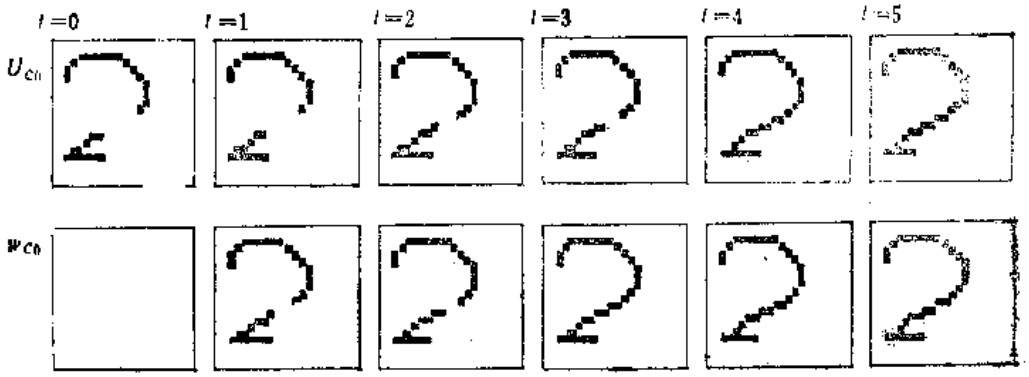


图 4-32　网络对应缺损样本的响应

图 4-31 中，输入样本为"20"，在最高层的"感悟"细胞中，相应于学习样本"2"和"0"的μ神经元同时兴奋，但对应于"0"的 u_c 神经元输出值更大，故与该 u_c 神经元对应的 w_c 神经元被激活，并反馈出传出信息，在 w_{c_0} 神经元层上产生"0"的联想记忆输出。在时刻 $t=3$ 时，传出信息被中断，相应于"2"的感悟细胞被激活，实现了注意力转移。

图 4-32 则显示了网络对一个有缺损的样本的响应过程。在时刻 $t=0$，由于识别样本与学习样本差别较大，最高层 u_c 神经元没有一个兴奋，故 w_{c_0} 上没有输出。最大值检测器送出神经调制信号 x_s，降低各 u_i 神经元的选择性。在 $t=1$ 时，对应于"2"的人工感悟细胞被激活，w_{c_0} 上也产生识别样本的联想记忆输出，该输出被反馈给输入，这样，经过一段时间后，识别样本中的缺损部分便逐渐被恢复出来。