第十九章 神经网络模型

§1 神经网络简介 人工神经网络是在现代神经科学的基础上提出和发展起来的，旨在反映人脑结构及

功能的一种抽象数学模型。自 1943 年美国心理学家 W. McCulloch 和数学家 W. Pitts 提 出形式神经元的抽象数学模型—MP 模型以来，人工神经网络理论技术经过了 50 多年 曲折的发展。特别是 20 世纪 80 年代，人工神经网络的研究取得了重大进展，有关的理 论和方法已经发展成一门界于物理学、数学、计算机科学和神经生物学之间的交叉学科。 它在模式识别，图像处理，智能控制，组合优化，金融预测与管理，通信，机器人以及 专家系统等领域得到广泛的应用，提出了 40 多种神经网络模型，其中比较著名的有感

知机，Hopfield 网络，Boltzman 机，自适应共振理论及反向传播网络（BP）等。在这

里我们仅讨论最基本的网络模型及其学习算法。

1.1 人工神经元模型

图 1 表示出了作为人工神经网络（artificial neural network，以下简称 NN）的基本 单元的神经元模型，它有三个基本要素：

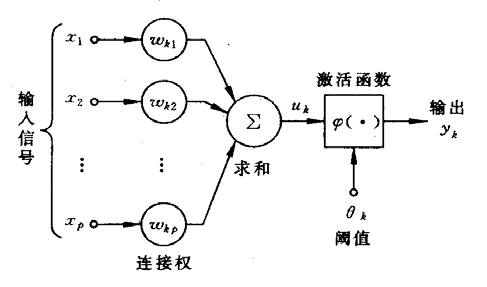


图 1 神经元模型

（i）一组连接（对应于生物神经元的突触），连接强度由各连接上的权值表示，权 值为正表示激活，为负表示抑制。

（ii）一个求和单元，用于求取各输入信号的加权和（线性组合）。

（iii）一个非线性激活函数，起非线性映射作用并将神经元输出幅度限制在一定范 围内（一般限制在 (0,1) 或 (−1,1) 之间）。

此外还有一个阈值θ*k* （或偏置 *bk* = −θ*k* ）。

以上作用可分别以数学式表达出来：

*p*

*uk* = ∑ *wkj x j* ，

*j* =1

*vk* = *uk* −θ*k* ，

*yk* = ϕ (*vk* )

式中 *x*1 , *x*2 ,L, *x p* 为输入信号， *wk*1 , *wk* 2 ,L, *wkp* 为神经元 *k* 之权值， *uk* 为线性组合结 果，θ*k* 为阈值，ϕ (⋅) 为激活函数， *yk* 为神经元 *k* 的输出。

若把输入的维数增加一维，则可把阈值θ*k* 包括进去。例如

*p*

*vk* = ∑ *wkj x j* ， *yk* = ϕ (*uk* )

*j* =0

此处增加了一个新的连接，其输入为 *x*0 = −1（或 + 1 ），权值为 *wk* 0 = θ*k* （或 *bk* ），如

图 2 所示。

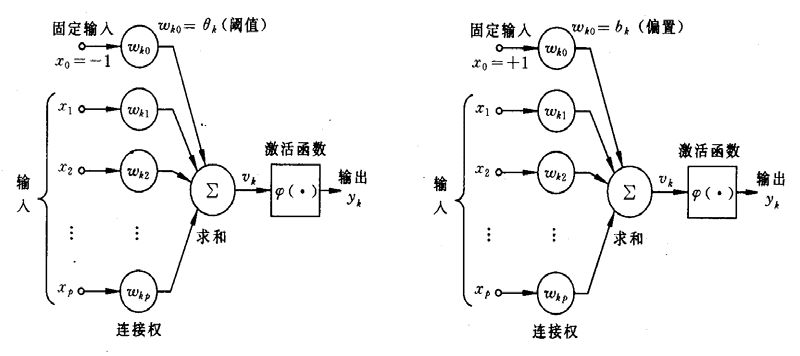


图 2 神经元模型

激活函数ϕ (⋅) 可以有以下几种:

（i）阈值函数

ϕ (*v*) = ⎧1,

⎨

⎩0,

*v* ≥ 0

*v* < 0

（1）

即阶梯函数。这时相应的输出 *yk* 为

⎧1,

*yk* = ⎨

⎩0,

*p*

*vk* ≥ 0

*vk* < 0

其中 *vk* = ∑*wkj x j* − θ*k* ，常称此种神经元为 *M* − *P* 模型。

*j* =1

（ii）分段线性函数

⎧1,

⎪ 1

ϕ (*v*) = ⎨

⎪ 2

⎪⎩0,

(1 + *v*),

*v* ≥ 1

−1 < *v* < 1

*v* ≤ −1

（2）

它类似于一个放大系数为 1 的非线性放大器，当工作于线性区时它是一个线性组合器， 放大系数趋于无穷大时变成一个阈值单元。

（iii）sigmoid 函数 最常用的函数形式为

ϕ (*v*) =

1

1 + exp(−α*v*)

（3）

参数α > 0 可控制其斜率。另一种常用的是双曲正切函数

(*v*)

tanh⎜ *v* ⎟ = 1 − exp(−*v*)

ϕ = ⎛ ⎞

⎝ 2 ⎠

1 + exp(−*v*)

（4）

这类函数具有平滑和渐近性，并保持单调性。

Matlab 中的激活（传递）函数如表 1 所示。

表 1 Matlab 工具箱中的传递函数

|  |  |
| --- | --- |
| 函数名 | 功 能 |
| purelin | 线性传递函数 |
| hardlim | 硬限幅传递函数 |
| hardlims | 对称硬限幅传递函数 |
| satlin | 饱和线性传递函数 |
| satlins | 对称饱和线性传递函数 |
| logsig | 对数 S 形传递函数 |
| tansig | 正切 S 形传递函数 |
| radbas | 径向基传递函数 |
| compet | 竞争层传递函数 |

各个函数的定义及使用方法，可以参看 Matlab 的帮助（如在 Matlab 命令窗口运行

help tansig，可以看到 tantig 的使用方法，及 tansig 的定义为ϕ (*v*) =

1.2 网络结构及工作方式

2

1 + *e* −2 *v*

− 1 ）。

除单元特性外，网络的拓扑结构也是 NN 的一个重要特性。从连接方式看 NN 主要 有两种。

（i）前馈型网络 各神经元接受前一层的输入，并输出给下一层，没有反馈。结点分为两类，即输入

单元和计算单元，每一计算单元可有任意个输入，但只有一个输出（它可耦合到任意多 个其它结点作为其输入）。通常前馈网络可分为不同的层，第 *i* 层的输入只与第 *i* − 1 层 输出相连，输入和输出结点与外界相连，而其它中间层则称为隐层。

（ii）反馈型网络 所有结点都是计算单元，同时也可接受输入，并向外界输出。

NN 的工作过程主要分为两个阶段：第一个阶段是学习期，此时各计算单元状态不 变，各连线上的权值可通过学习来修改；第二阶段是工作期，此时各连接权固定，计算 单元状态变化，以达到某种稳定状态。

从作用效果看，前馈网络主要是函数映射，可用于模式识别和函数逼近。反馈网络 按对能量函数的极小点的利用来分类有两种：第一类是能量函数的所有极小点都起作 用，这一类主要用作各种联想存储器；第二类只利用全局极小点，它主要用于求解最优 化问题。

§2 蠓虫分类问题与多层前馈网络

2.1 蠓虫分类问题

蠓虫分类问题可概括叙述如下：生物学家试图对两种蠓虫（Af 与 Apf）进行鉴别， 依据的资料是触角和翅膀的长度，已经测得了 9 支 Af 和 6 支 Apf 的数据如下：

Af: (1.24,1.27) ，(1.36,1.74) ，(1.38,1.64) ，(1.38,1.82) ，(1.38,1.90) ，(1.40,1.70) ，

(1.48,1.82)，(1.54,1.82)，(1.56,2.08).

Apf: (1.14,1.82)，(1.18,1.96)，(1.20,1.86)，(1.26,2.00)，(1.28,2.00)，(1.30,1.96).

现在的问题是：

（i）根据如上资料，如何制定一种方法，正确地区分两类蠓虫。

（ii）对触角和翼长分别为(1.24,1.80)，(1.28,1.84)与(1.40,2.04)的 3 个标本，用所得 到的方法加以识别。

（iii）设 Af 是宝贵的传粉益虫，Apf 是某疾病的载体，是否应该修改分类方法。 如上的问题是有代表性的，它的特点是要求依据已知资料（9 支 Af 的数据和 6 支

Apf 的数据）制定一种分类方法，类别是已经给定的（Af 或 Apf）。今后，我们将 9 支

Af 及 6 支 Apf 的数据集合称之为学习样本。

2.2 多层前馈网络

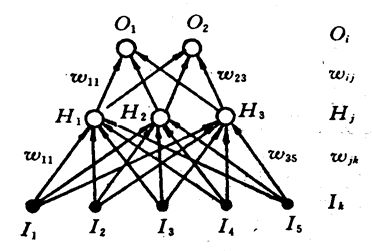


图 3 多层前馈网络

为解决上述问题，考虑一个其结构如图 3 所示的人工神经网络，激活函数由

ϕ (*v*) =

1

1 + exp(−α*v*)

来决定。图中最下面单元，即由 • 所示的一层称为输入层，用以输入已知测量值。在

我们的例子中，它只需包括两个单元，一个用以输入触角长度，一个用以输入翅膀长度。

中间一层称为处理层或隐单元层，单元个数适当选取，对于它的选取方法，有一些文献 进行了讨论，但通过试验来决定，或许是最好的途径。在我们的例子中，取三个就足够 了。最上面一层称为输出层，在我们的例子中只包含二个单元，用以输出与每一组输入 数据相对应的分类信息．任何一个中间层单元接受所有输入单元传来的信号，并把处理 后的结果传向每一个输出单元，供输出层再次加工，同层的神经元彼此不相联接，输入 与输出单元之间也没有直接联接。这样，除了神经元的形式定义外，我们又给出了网络 结构。有些文献将这样的网络称为两层前传网络，称为两层的理由是，只有中间层及输 出层的单元才对信号进行处理；输入层的单元对输入数据没有任何加工，故不计算在层 数之内。

为了叙述上的方便，此处引入如下记号上的约定：令 *s* 表示一个确定的已知样品标 号，在蠓虫问题中， *s* = 1,2,L,15 ，分别表示学习样本中的 15 个样品；当将第 *s* 个样 品的原始数据输入网络时，相应的输出单元状态记为 *O s* (*i* = 1,2) ，隐单元状态记为

*i*

*H s* ( *j* = 1,2,3) ，输入单元取值记为 *I s* (*k* = 1,2) 。在这一约定下，从中间层到输出层的

*j k*

权记为 *wij* ，从输入层到中间层的权记为 *w jk* 。如果 *wij* ， *w jk* 均已给定，那么，对应于

*s s*

任何一组确定的输入 ( *I*1 , *I* 2 ) ，网络中所有单元的取值不难确定。事实上，对样品 *s* 而

言，隐单元 *j* 的输入是

2

*s* = ∑ *s*

相应的输出状态是

*h j*

*k* =1

*w jk I k*

（5）

2

*H s* = ϕ (*h s* ) = ϕ (∑*w*

*I s* )

（6）

*j j*

*k* =1

*jk k*

由此，输出单元 *i* 所接收到的迭加信号是

3 3 2

*h s* = ∑ *w H s* = ∑ *w* ϕ (∑ *w*

*I s* )

（7）

*i*

网络的最终输出是

*ij*

*j* =1

*j ij*

*j* =1

3

*k* =1

*jk k*

3 2

*O s* = ϕ (*h s* ) = ϕ (∑ *w*

*H s* ) = ϕ (∑ *w* ϕ (∑ *w*

*I s* ))

（8）

*i i ij j*

*j* =1

*ij*

*j* =1

*k* =1

*jk k*

这里，没有考虑阈值，正如前面已经说明的那样，这一点是无关紧要的。还应指出的是， 对于任何一组确定的输入，输出是所有权{*wij* , *w jk* } 的函数。

如果我们能够选定一组适当的权值{*wij* , *w jk* } ，使得对应于学习样本中任何一组 Af

*s s s s*

样品的输入 ( *I*1 , *I* 2 ) ，输出 (*O*1 , *O*2 ) = (1,0) ，对应于 Apf 的输入数据，输出为 (0,1) ，

那么蠓虫分类问题实际上就解决了。因为，对于任何一个未知类别的样品，只要将其触

角及翅膀长度输入网络，视其输出模式靠近 (1,0) 亦或 (0,1) ，就可能判断其归属。当然， 有可能出现介于中间无法判断的情况。现在的问题是，如何找到一组适当的权值，实现 上面所设想的网络功能。

2.3 向后传播算法 对于一个多层网络，如何求得一组恰当的权值，使网络具有特定的功能，在很长一

段时间内，曾经是使研究工作者感到困难的一个问题，直到 1985 年，美国加州大学的 一个研究小组提出了所谓向后传播算法（Back-Propagation），使问题有了重大进展，这 一算法也是促成人工神经网络研究迅猛发展的一个原因。下面就来介绍这一算法。

如前所述，我们希望对应于学习样本中 Af 样品的输出是 (1,0) ，对应于 Apf 的输出

是 (0,1) ，这样的输出称之为理想输出。实际上要精确地作到这一点是不可能的，只能 希望实际输出尽可能地接近理想输出。为清楚起见，把对应于样品 *s* 的理想输出记为

{*T s* } ，那么

*i*

= 1 ∑

*s* − *s* 2

*E* (*W* )

2 *i* , *s*

(*Ti Oi* )

（9）

度量了在一组给定的权下，实际输出与理想输出的差异，由此，寻找一组恰当的权的问 题，自然地归结为求适当*W* 的值，使 *E* (*W* ) 达到极小的问题。将式（8）代入（9），有

3 2

= 1 ∑ *s* − ∑ ∑ *s* 2

*E* (*W* )

2 *s* ,*i*

[*Ti*

ϕ (

*j* =1

*wij*ϕ (

*k* =1

*w jk I k* ))]

（10）

易知，对每一个变量 *wij* 或 *wij* 而言，这是一个连续可微的非线性函数，为了求得其极 小点与极小值，最为方便的就是使用最速下降法。最速下降法是一种迭代算法，为求出 *E* (*W* ) 的（局部）极小，它从一个任取的初始点*W*0 出发，计算在*W*0 点的负梯度方向

— ∇*E*(*W*0 ) ，这是函数在该点下降最快的方向；只要 ∇*E*(*W*0 ) ≠ 0 ，就可沿该方向移动 一小段距离，达到一个新的点*W*1 = *W*0 −η∇*E*(*W*0 ) ，η 是一个参数，只要η 足够小， 定能保证 *E* (*W*1 ) < *E* (*W*0 ) 。不断重复这一过程，一定能达到 *E* 的一个（局部）极小点。

就本质而言，这就是 BP 算法的全部内容，然而，对人工神经网络问题而言，这一算法 的具体形式是非常重要的，下面我们就来给出这一形式表达。

对于隐单元到输出单元的权 *wij* 而言，最速下降法给出的每一步的修正量是

∂*E s s s s s s*

Δ*wij* = −η

∂*w*

= η∑[*Ti*

− *Oi* ]ϕ ' (*hi* )*H j*

= η∑δ *i H j*

（11）

此处令

*ij s s*

δ *s* = ϕ ' (*h s* )[*T s* − *O s* ]

（12）

*i i i i*

对输入单元到隐单元的权 *w jk*

∂*E*

*s s s s s*

Δ*w jk* = −η

∂*w jk*

= η∑[*Ti*

*s* , *i*

− *Oi* ]ϕ ' (*hi* )*wij*ϕ ' (*h j* ) *I j*

= ∑ *s*

*s s* = ∑ *s s*

η

*s* , *i*

此处

*w* δ

δ *i wij*ϕ ' (*h j* ) *I k*

η δ *j I k*

*s*

（13）

δ *s* =

*j*

ϕ ' (*h s* )∑

*i*

*j*

*s ij i*

从（11）和（13）式可以看出，所有权的修正量都有如下形式，即

Δ = ∑ *s s*

*w pq*

η δ *p vq*

*s*

（14）

指标 *p* 对应于两个单元中输出信号的一端，*q* 对应于输入信号的一端，*s* 或者代表 *H* 或

者代表 *I* 。形式上看来，这一修正是“局部”的，可以看作是 Hebb 律的一种表现形式。

还应注意，δ *s* 由实际输出与理想输出的差及 *hs* 决定，而δ *s* 则需依赖δ *s* 算出，因此，

*i i j i*

这一算法才称为向后传播算法。稍加分析还可知道，利用由（11）~（13）式所给出的 计算安排，较之不考虑δ *s* 的向后传播，直接计算所有含ϕ ' 的原表达式，极大地降低了 计算工作量。这组关系式称作广义δ − 法则，它们不难推广到一般的多层网络上去。

*p*

利用这一迭代算法，最终生成在一定精度内满足要求的{*wij* , *w jk* } 的过程，称为人

工神经网络的学习过程。可以看出，这里所提供的学习机制是元与元之间权的不断调整， 学习样本中任何一个样品所提供的信息，最终将包含在网络的每一个权之中。参数η 的

大小则反映了学习效率。

为了更有效地应用 BP 算法，我们做出如下一些补充说明。

（i）在式（11）与（13）中， Δ*wij* , Δ*w jk* 表示为与所有样品 *s* 有关的求和计算。

实际上，我们还可以每次仅考虑输入一个样品所造成的修正，然后，按照随机选取的顺 序，将所有样品逐个输入，不断重复这一手续，直至收敛到一个满意的解为止。

（ii）在如上的算法中，利用实际输出与理想输出差的平方和作为度量{*wij* , *w jk* } 优

劣的标准，这并不是唯一的度量方式，完全可以从其它的函数形式出发，例如从相对熵 出发，导出相应的算法。

（iii）在如上的讨论中使用的是最速下降法，显然，这也不是唯一的选择，其它的 非线性优化方法，诸如共轭梯度法，拟牛顿法等，都可用于计算。为了加速算法的收敛 速度，还可以考虑各种不同的修正方式。

（iv）BP 算法的出现，虽然对人工神经网络的发展起了重大推动作用，但是这一 算法仍有很多问题．对于一个大的网络系统，BP 算法的工作量仍然是十分可观的，这 主要在于算法的收敛速度很慢。更为严重的是，此处所讨论的是非线性函数的优化，那 么它就无法逃脱该类问题的共同困难：BP 算法所求得的解，只能保证是依赖于初值选 取的局部极小点。为克服这一缺陷，可以考虑改进方法，例如模拟退火算法，或从多个

随机选定的初值点出发，进行多次计算，但这些方法都不可避免地加大了工作量。

2.4 蠓虫分类问题的求解 下面利用上文所叙述的网络结构及方法，对蠓虫分类问题求解。编写 Matlab 程序

如下：

clear p1=[1.24,1.27;1.36,1.74;1.38,1.64;1.38,1.82;1.38,1.90;

1.40,1.70;1.48,1.82;1.54,1.82;1.56,2.08];

p2=[1.14,1.82;1.18,1.96;1.20,1.86;1.26,2.00

1.28,2.00;1.30,1.96]; p=[p1;p2]'; pr=minmax(p);

goal=[ones(1,9),zeros(1,6);zeros(1,9),ones(1,6)]; plot(p1(:,1),p1(:,2),'h',p2(:,1),p2(:,2),'o') net=newff(pr,[3,2],{'logsig','logsig'}); net.trainParam.show = 10;

net.trainParam.lr = 0.05;

net.trainParam.goal = 1e-10; net.trainParam.epochs = 50000; net = train(net,p,goal);

x=[1.24 1.80;1.28 1.84;1.40 2.04]';

y0=sim(net,p)

y=sim(net,x)

§3 处理蠓虫分类的另一种网络方法

3.1 几个有关概念 在介绍本节主要内容之前，首先说明几个不同的概念。在上一节中，我们把利用

BP 算法确定联接强度，即权值的过程称为“学习过程”，这种学习的特点是，对任何一 个输入样品，其类别事先是已知的，理想输出也已事先规定，因而从它所产生的实际输 出与理想输出的异同，我们清楚地知道网络判断正确与否，故此把这一类学习称为在教 师监督下的学习；与它不同的是，有些情况下学习是无监督的，例如，我们试图把一组 样品按其本身特点分类，所要划分的类别是事先未知的，需要网络自身通过学习来决定， 因而，在学习过程中，对每一输入所产生的输出也就无所谓对错，对于这样的情况，显 然 BP 算法是不适用的。

另一个有关概念是所谓有竞争的学习。在上节所讨论的蠓虫分类网络中，尽管我们

所希望的理想输出是 (1,0) 或 (0,1) ，但实际输出并不如此，一般而言，两个输出单元均 同时不为 0。与此不同，我们完全可以设想另外一种输出模式：对应任何一组输入，所 有输出单元中，只允许有一个处于激发态，即取值为 1，其它输出单元均被抑制，即取 值为 0。一种形象的说法是，对应任何一组输入，要求所有的输出单元彼此竞争，唯一 的胜利者赢得一切，失败者一无所获，形成这样一种输出机制的网络学习过程，称为有 竞争的学习。

3.2 最简单的无监督有竞争的学习 本节叙述一种无监督有竞争的网络学习方法，由此产生的网络可用来将一组输入样

品自动划分类别，相似的样品归于同一类别，因而激发同一输出单元，这一分类方式， 是网络自身通过学习，从输入数据的关系中得出的。

蠓虫分类问题对应有教师的网络学习过程，显然不能由如上的方法来解决。但在这 种无监督有竞争的学习阐明之后，很容易从中导出一种适用于有监督情况的网络方法； 此外，本节所介绍的网络，在数据压缩等多种领域，都有其重要应用。

考虑一个仅由输入层与输出层组成的网络系统，输入单元数目与每一样品的测量值 数目相等，输出单元数目适当选取。每一个输入单元与所有输出单元联接，第 *j* 个输入 元到第 *i* 个输出元的权记为 *wij* ，同层单元间无横向联接。不妨假设所有输入数值均已

规化到[−1,1] 之间，又因为是有竞争的学习，输出单元只取 0 或 1 两个值，且对应每一

组输入，只有一个输出元取 1。

取 1 的输出元记为 *i*\* ，称之为优胜者。对于任何一组输入 *s* ，规定优胜者是有最大 净输入的输出元，即对输入 *I* = ( *I*1 ,L, *I n* ) 而言，

*hi* = ∑*wij I j* ≡ *Wi* ⋅ *I*

*j*

取最大值的单元，其中*Wi* 是输出元 *i* 所有权系数组成的向量，也就是说

(15)

*W* \* ⋅ *I* ≥ *Wi* ⋅ *I* ，

*i*

(∀*i*)

（16）

如果权向量是按照 ∑ *wij* = 1 的方式标准化的，（16）式等价于

2

*j*

| *W* \* − *I* |≤| *Wi* − *I* | ， (∀*i*)

*i*

（17）

即优胜者是其标准化权向量最靠近输入向量的输出元。令 *O* \* = 1 ，其余的输出

*i*

*Oi* = 0 。这样的输出规定了输入向量的类别，但为了使这种分类方式有意义，问题化

为如何将学习样本中的所有样品，自然地划分为聚类，并对每一聚类找出适当的权向量。 为此，采用如下的算法：随机取定一组不大的初始权向量，注意不使它们有任何对称性。

然后，将已知样品按照随机顺序输入网络。对输入样品 *s* ，按上文所述确定优胜者 *i*\* ， 对所有与 *i*\* 有关的权作如下修正

Δ*w* = η( *I s*

\*

*j*

*i j*

− *w* )

*i j*

\*

\*

（18）

所有其它输出单元的权保持不变。注意到*Oi* \* = 1 ，*Oi* = 0(*i* ≠ *i*

) ，所有权的修正公式

可统一表示为

Δ*w* \*

*i j*

= η*Oi* ( *I j*

*s*

− *w* \* )

*i j*

这一形式也可视为 Hebb 律的一种表现。（18）式的几何意义是清楚的，每次修正将优 胜者的权向量向输入向量移近一小段距离，这使得同一样品再次输入时， *i*\* 有更大的 获胜可能。可以合理地预期，反复重复以上步骤，使得每个输出单元对应了输入向量的 一个聚类，相应的权向量落在了该聚类样品的重心附近。当然，这只是一个极不严密的

说明。

特别应当指出，上述算法，对于事先按照 ∑ *I j* = 1 标准化了的输入数据更为适用，

整个过程不难由计算机模拟实现。

为了更有效地使用如上算法，下面对实际计算时可能产生的问题，作一些简要说明。 首先，如果初始权选择不当，那么可能出现这样的输出单元，它的权远离任何输入

向量，因此，永远不会成为优胜者，相应的权也就永远不会得到修正，这样的单元称之 为死单元。为避免出现死单元，可以有多种方法。一种办法是初始权从学习样本中抽样 选取，这就保证了它们都落在正确范围内；另一种办法是修正上述的学习算法，使得每

一步不仅调整优胜者的权，同时也以一个小得多的η 值，修正所有其它的权。这样，对

于总是失败的单元，其权逐渐地朝着平均输入方向运动，最终也会在某一次竞争中取胜。

此外，还存在有多种处理死单元的方法，感兴趣的读者可从文献中找到更多的方法。

另外一个问题是这一算法的收敛性。如果式（18）或（19）中反映学习效率的参数 η 取为一个固定常数，那么权向量永远不会真正在某一有限点集上稳定下来。因此，应 当考虑在公式中引进随学习时间而变化的收敛因子。例如，取η = η(*t*) = η *t* −*a* ，

0

0 < *a* ≤ 1 。这一因子的适当选取是极为重要的，η 下降太慢，无疑增加了不必要工作 量，η 下降太快，则会使学习变得无效。

3.3 LVQ 方法

上述有竞争学习的一个最重要应用是数据压缩中的向量量子化方法（Vector

Quantization）。它的基本想法是，把一个给定的输入向量集合 *I s* 分成 *M* 个类别，然后 用类别指标来代表所有属于该类的向量。向量分量通常取连续值，一旦一组适当的类别 确定之后，代替传输或存储输入向量本身，可以只传输或存储它的类别指标。所有的类 别由 *M* 个所谓“原型向量”来表示，我们可以利用一般的欧氏距离，对每一个输入向 量找到最靠近的原型向量，作为它的类别。显然，这种分类方法可以通过有竞争的学习 直接得到。一旦学习过程结束，所有权向量的集合，便构成了一个“电码本”。

一般而言，上述无监督有竞争的学习，实际提供了一种聚类分析方法，对如蠓虫分 类这种有监督的问题并不适用。1989 年，Kohonen 对向量量子化方法加以修改，提出 了一种适用 于有监督情 况的学习方 法，称为学 习向量量子 化（ Learning Vector Quantization），该方法可用于蠓虫分类问题。在有监督的情况下，学习样品的类别是事 先已知的，与此相应，每个输出单元所对应的类别也事先作了规定，但是，代表同一类 别的输出单元可以不止一个。

在 LVQ 中，对于任一输入向量，仍按无监督有竞争的方式选出优胜者 *i*\* ，但权的 修正规则则依输入向量的类别与 *i*\* 所代表的是否一致而不同，确切地说，令

Δ*w* \*

*j*

⎪⎧η(

= ⎨

*I s* −

*w* \* )

*i j*

一致情况

*i j* ⎪ η(

−

⎩

*I s* −

*w* \* )

*i j*

不一致情况

前一种情况，修正和无监督的学习一致，权朝向样品方向移动一小段距离；后一种 则相反，权向离开样品方向移动，这样就减少了错误分类的机会。

*j*

对于上述的蠓虫分类问题，我们编写 Matlab 程序如下：

clear p1=[1.24,1.27;1.36,1.74;1.38,1.64;1.38,1.82;1.38,1.90;

1.40,1.70;1.48,1.82;1.54,1.82;1.56,2.08];

p2=[1.14,1.82;1.18,1.96;1.20,1.86;1.26,2.00

1.28,2.00;1.30,1.96];

p=[p1;p2]' pr=minmax(p) goal=[ones(1,9),zeros(1,6);zeros(1,9),ones(1,6)] net = newlvq(pr,4,[0.6,0.4])

net = train(net,p,goal) Y = sim(net,p)

x=[1.24 1.80;1.28 1.84;1.40 2.04]' sim(net,x)

习 题 十 九

1. 利用 BP 算法及 sigmoid 函数，研究以下各函数的逼近问题

（i） *f* ( *x*) = 1 ,

*x*

1 ≤ *x* ≤ 100

（ii） *f* ( *x*) = sin *x*,

0 ≤ *x* ≤ π

2

对每一函数要完成如下工作：

① 获取两组数据，一组作为训练集，一组作为测试集；

② 利用训练集训练一个单隐层的网络；用测试集检验训练结果，改变隐层单元数， 研究它对逼近效果的影响。

2. 给定待拟合的曲线形式为

*f* ( *x*) = 0.5 + 0.4 sin(2π*x*)

在 *f* ( *x*) 上等间隔取 11 个点的数据，在此数据的输出值上加均值为 0，均方差σ

= 0.05

的正态分布噪声作为给定训练数据，用多项式拟合此函数，分别取多项式的阶次为 1，

3 和 11 阶，图示出拟合结果，并讨论多项式阶次对拟合结果的影响。