出层的元素产生输出值 $O_i \geqslant 0.95$ 和 $O_q \leqslant 0.05$ 时($q = 1,2,\cdots,N_Q$ 且 $q \neq i$),我们说网络从所有的 4 个类学到了这些形状。换句话说,对类 $\omega_i$ 的所有模式,与所求类一致的输出节点必须为高($\geqslant 0.95$),而同时,所有其他节点必须为低($\leqslant 0.05$)。

训练的第二部分要在带有噪声样本的情况下训练。在无噪声图形中,每个处在轮廓线上的像素被给予一个保留在图形平面上的原始坐标的概率 $V$,并且对该像素 8 个相邻像素其中一个的坐标随机赋予一个概率 $R = 1 - V$。噪声的等级随 $V$ 的减小而增加(即增加 $R$)。两个噪声数据集合就产生了。第一个集合通过在 0.1 到 0.6 之间的不同 $R$ 值对每个类生成 100 个噪声模式,总共为 400 个模式。这个集合称为检测集合,用于确定系统在训练后的性能。

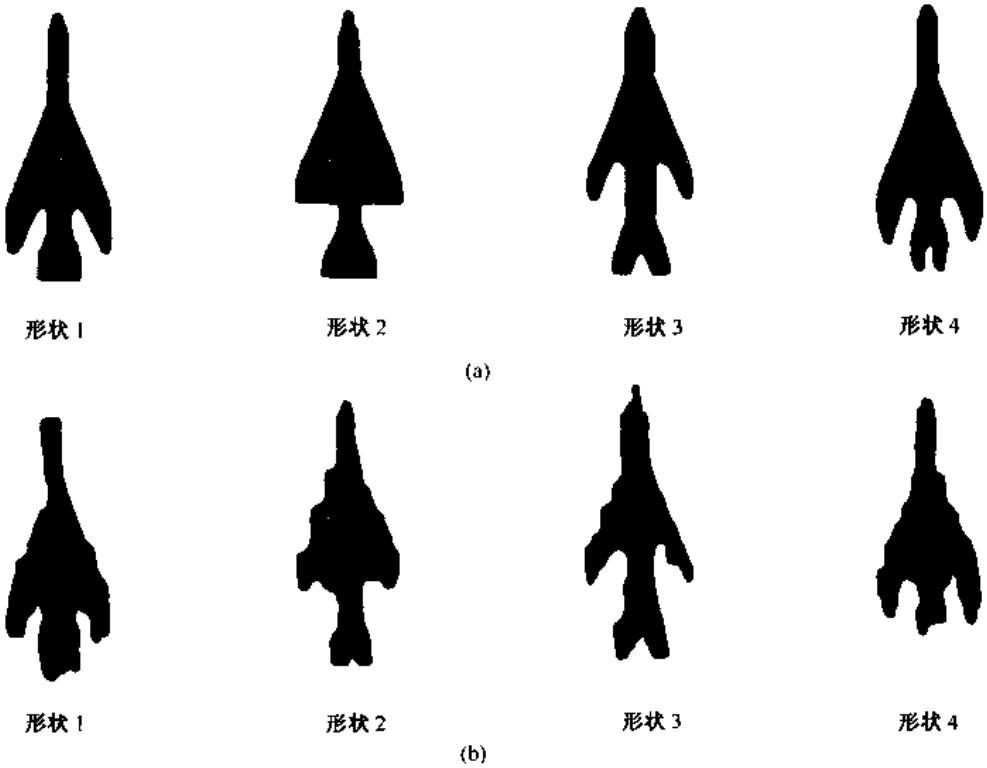



图 12.18 (a)参考图形和(b)用于训练图 12.19 所示神经网络的典型噪声图形(由南伊利诺伊大学ECE系的Lalit Gupta博士提供)

用噪声数据训练系统要产生几个噪声集合。第一个集合由每类的 10 个样本组成,通过令 $R_t = 0$ 生成,这里 $R_t$ 代表用于生成训练数据的 $R$ 值。从第一部分(无噪声)训练得到的权值矢量开始,系统允许经过一系列有序的带有新数据集合的学习过程。因为 $R_t = 0$ 暗示无噪声,这种再训练是以前无噪声训练的延伸。以这种方式产生学习过的权重,网络受测试数据集的影响,数据集生成的结果如图 12.20 中标有 $R_t = 0$ 的曲线所示。错误分类的模式数目除以检测过的模式总数得到错误分类的概率,一般是衡量确立的神经网络性能的度量。

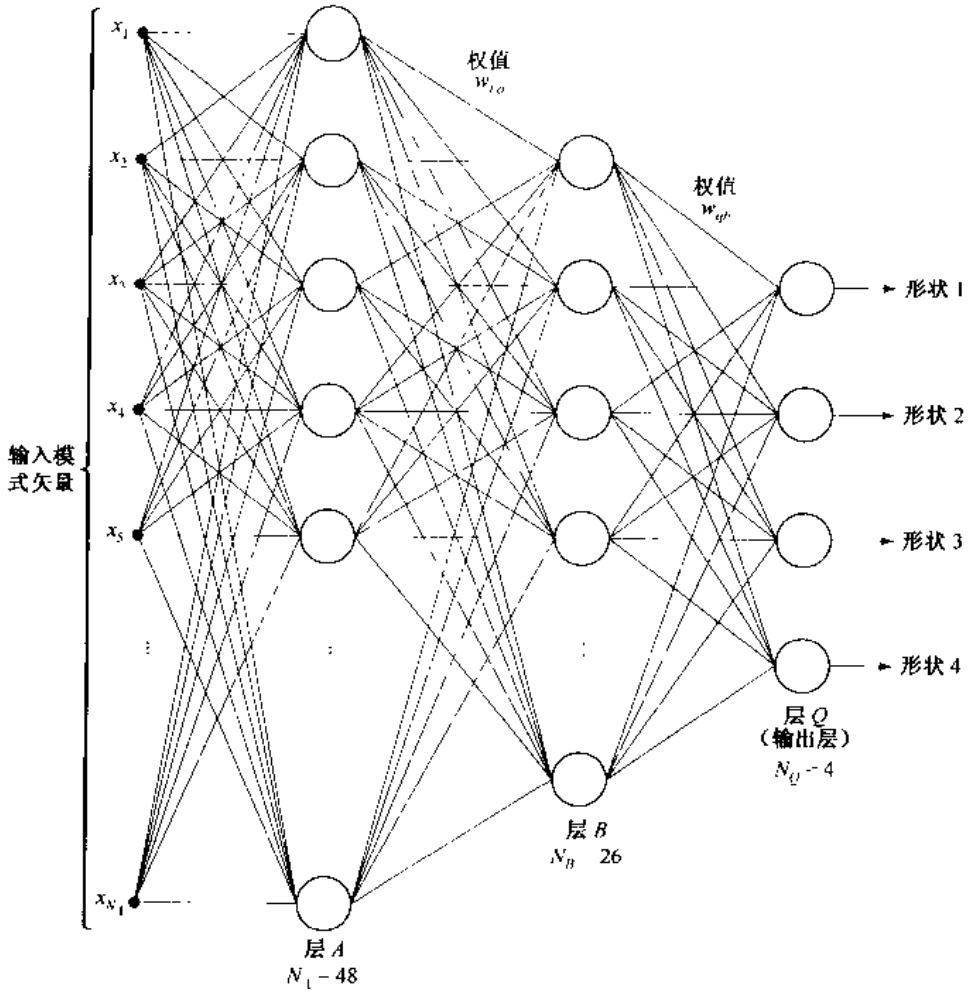



图 12.19　用于识别图 12.18 中图形的三层神经网络(由南伊利诺伊大学ECE系的Lalit Gupta博士提供)

接下来,从使用 $R_t = 0$ 生成的噪声数据权值矢量开始,用由 $R_t = 0.1$ 生成的噪声数据集合对系统进行重新训练。识别性能再用新权值矢量通过系统运行检测样本来确立。注意,这次系统在性能上的显著改进。图 12.20 显示了通过持续这种再训练和令 $R_t = 0.2, 0.3$ 和 0.4 后进行的再检测过程得到的结果。如所希望的那样,如果系统进行了适当的学习,由检测集合测试出的模式错误分类的概率会在 $R_t$ 增加时降低,因为系统使用了有更高 $R_t$ 值的噪声数据进行训练。图 12.20 中的一个例外是 $R_t = 0.4$ 时的结果。原因是训练系统的样本数目少,即网络无法在使用这样少的样本情况下使自己充分适应具有更高噪声级别的图形的更大变化。这种假设在图 12.21 中得到证实,在此图中显示出,当样本数目增加时得到更低的错误分类概率。图 12.21 也显示了图 12.20 中的曲线在 $R_t = 0.3$ 时的形状作为参考。

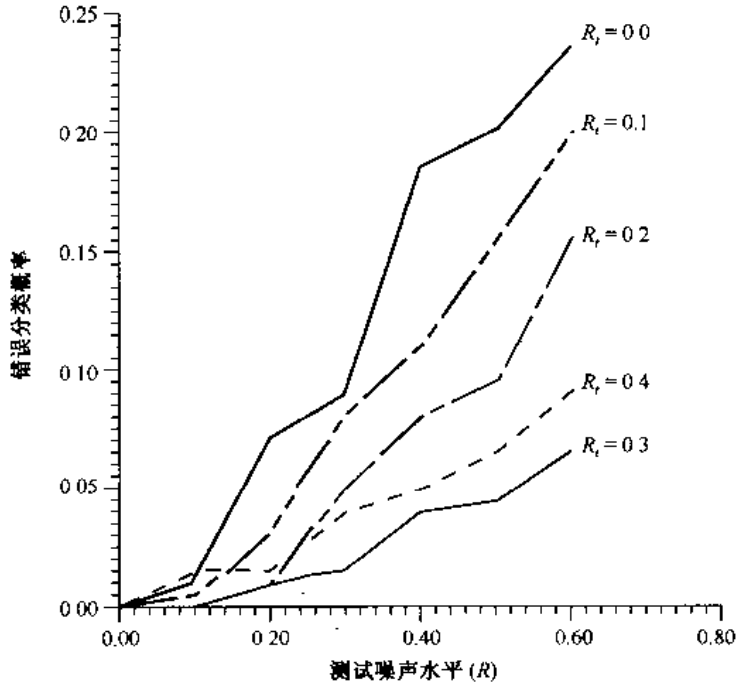



图 12.20 作为噪声水平函数的神经网络性能(由南伊利诺伊大学ECE系的Lalit Gupta博士提供)

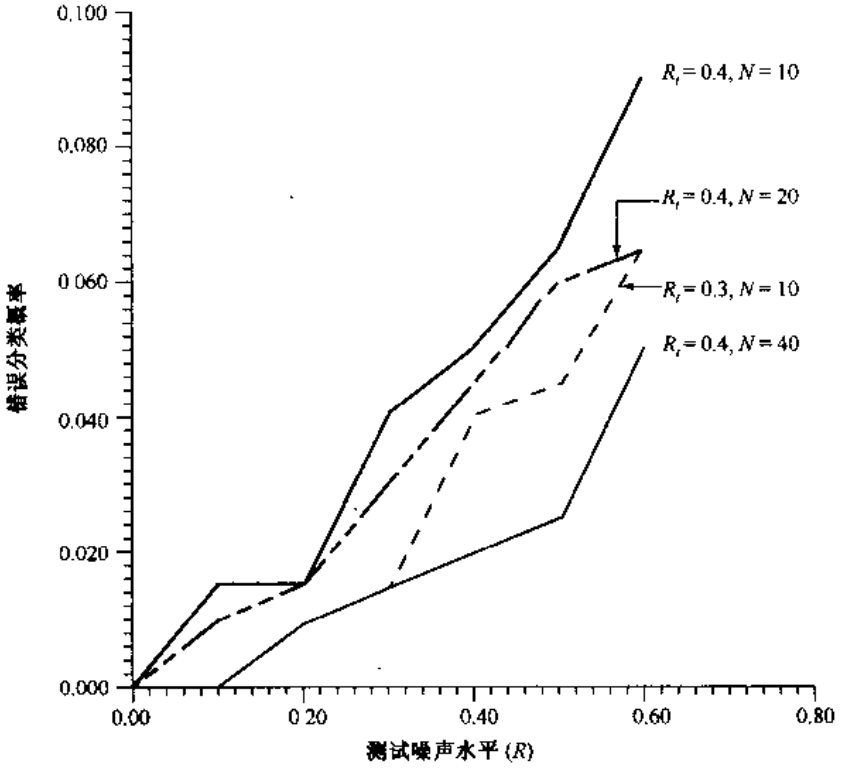



图 12.21 通过增加训练模式数目改进 $R_t = 0.4$ 时的效果($R_t = 0.3$ 的曲线作为参考)(由南伊利诺伊大学ECE系的Lalit Gupta博士提供)

前述结果显示出一个三层神经网络在经过适当训练后有能力识别被噪声干扰的图形。甚至使用无噪声数据(图 12.20 中 $R_r=0$)训练的系统当使用被较高噪声(图 12.20 中 $R=0.6$)干扰的数据测试时,其正确识别的水平依然可以接近 77%。当系统使用噪声水平更高的数据($R_r=0.3$ 和 $R_r=0.4$)训练后,对同样数据的识别率增加到 99%。应该注意到通过系统性的、噪声水平小量增加的训练来增强系统的分类能力是很重要的。当噪声性质已知时,这种方法对改进神经网络在学习过程中的收敛和稳定性是很理想的。

**决策面的复杂性**

我们已经确定一个单层感知器执行一个超平面的决策面。关于这一点一个自然而然的问题是,像图 12.16 中的模型那样,由一个多层网络实现的决策面性质是什么? 这个问题将在下面的讨论中论证,下面的讨论是一个三层网络有能力实现由相交超平面组成的任意复杂的决策面。

作为讨论的起点,考虑图 12.22(a)所示的两个输入并有两层的网络。具有两个输入,模式是两维的,所以网络第一层的每个节点执行的是二维空间中的一行。用 1 和 0 分别代表这两个节点的高和低输出。假设 1 输出表示,对第一层一个节点的相应输入向量位于直线为正的一边。然后,有可能输出给第二层中单个节点的组合是(1,1),(1,0),(0,1)和(0,0)。如果定义两个区域,一个对应类 $\omega_1$,位于两条直线正的一侧,面另一个对应于类 $\omega_2$,位于直线的另一侧,输出节点可以把任何输入模式分类,只需要对属于这两个区域之一的输入模式进行简单的逻辑"与"操作。换句话说,输出节点响应为 1,仅当第一层的两个输出都为 1 时,表示类 $\omega_1$。如果 $\theta_j$ 在半开半闭区间(1,2]中设值,"与"操作可以由早些时候讨论的形式的神经节点来执行。如果假设为 0 和为 1 的响应超出第一层,只有当来自第一层的神经节点的两个输出执行的求和大于 1 时,输出节点的响应才会为高,表示类 $\omega_1$。图 12.22(b)和(c)显示了图 12.22(a)所示网络如何成功地等分两个模式类,而使用一个单一的线性面则做不到这一点。

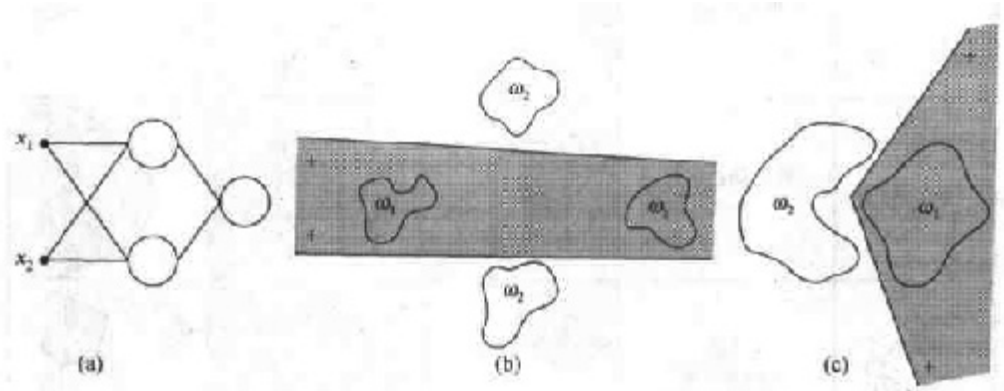



图 12.22　(a)一个两输入、两层的前馈神经网络。(b)和(c)可以由这个网络实现的决策边界的例子

如果第一层中的节点数目增加到 3 个,图 12.22(a)中的网络将由 3 条直线相交组成的判别边界来执行。类 $\omega_1$ 位于所有 3 条直线正的一侧的要求产生一个由这 3 条直线作为边界线的凸区域。实际上,任何开放的或闭合的凸区域都可以简单地通过增加两层神经网络第一层的节点数来构造。

下一个逻辑步骤是将层数增加到3层。在这种情况下,和以前一样,第一层的节点执行一行。为了从各个行形成一个区域,第二层的节点进行"与"操作。第三层的节点将不同区域划归不同的类。例如,设想类 $\omega_1$ 由两个不同区域组成,每个区域由不同的直线集合构成边界。第二层节点中的两个对应于相同模式类的区域。当第二层的两个节点中任一个变为高时,输出节点中应有一个能以信号表示该类的存在。假设第二层中由1和0分别代表高和低状态,这可以通过令网络的输出节点执行逻辑"或"操作得到。根据前面讨论过的神经节点的形式,给 $\theta_j$ 在半开区间内设置一个值。无论什么时候,在第二层中至少有一个节点与变为高状态的输出节点(输出为1)相联系,输出层的相应节点变为高状态就表示被处理的模式属于与那个节点相联系的类。

图12.23总结了前面的讨论。注意在第三行中,由三层网络实现的判别区域的复杂性在原理上是任意的。实际上,通常一系列的困难主要是由构造第二层来正确响应与各种特殊类相关的组合产生的。原因是直线不会正好在与其他直线的交点处截止,结果在模式空间中,属于相同类的模式也许会在直线的两边出现。从实际方面讲,在第二层中很难画出哪一条线应该包括在给定模式类的"与"操作中,或许这根本就是不可能的。参考图12.23的第三列中异或问题处理的情况,即,当输入模式为二元形式时,以两个维度只能构造四个不同的模式。如果模式被排列成如下形式:类 $\omega_1$ 由模式{(0,1),(1,0)}构成,类 $\omega_2$ 由模式{(0,0),(1,1)}构成,这两个类中,模式的类归属由异或(XOR)逻辑函数给出。这个函数只有在两个变量中的一个为1时才为1,否则就为0。因此,XOR函数值为1表示模式属于类 $\omega_1$,XOR函数值为0表示模式属于类 $\omega_2$。

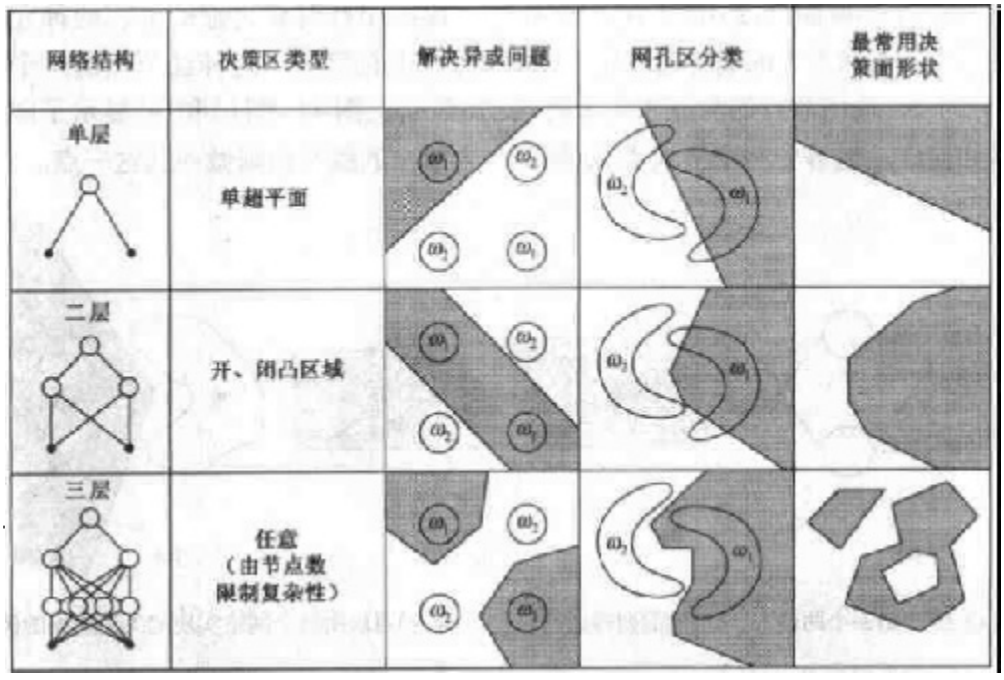



图12.23 可以由单层和多层前馈网络组成的判别区域类型,这个网络有一层或两层隐藏单元和两个输入(Lippman)

前面的讨论以直接的方式推广到 $n$ 维情况:这种情况中处理的是超平面而不是直线。一个单层网络执行一个单一的超平面。一个两层网络执行由超平面交集组成的任意凸区域。一

个三层网络执行任意复杂的判别面。每层使用的节点数决定了后两种情况的复杂性。第一种情况中，限制类的数目为两个。在另两种情况下，类的数目是任意的，因为可以根据要处理的问题决定输出节点的数目。

考虑到前述的评论，一个合乎逻辑的问题是，为什么人们对研究三层以上的神经网络有兴趣？毕竟，一个三层网络可以执行任意复杂的决策面。答案就在于仅使用三个层次对网络进行训练时所使用的方法上。图 12.16 中网络的训练规则将误差度量降至了最小，但却并未谈到如何把多个超平面和以前讨论过的三层网络第二层中特殊的节点联系起来。实际上，如何在层次数和每层上的节点数之间进行折中的分析的问题仍没有解决。在实现过程中，一般是通过反复实验和检验错误或凭借对给定问题领域的经验解决折中分析难题的。

## 12.3　结构性方法

在 12.2 节里讨论过在数量上处理模式的技术。这种技术在很大程度上忽略了模式图形的任何结构上的关系。在这一节中，将讨论结构性方法，通过对这些类型的关系进行适当的估计来实现模式识别。

### 12.3.1　匹配形状数目

在 12.2.1 节中，为了将模式矢量公式化，介绍了最小距离的概念。模式矢量公式化是为了对比依据图形数目描绘的区域边界。以 11.2.2 节的论述为参考，两个区域边界间(形状)的相似度 $k$ 定义为它们相一致的图形数目的最大量级。例如，令 $a$ 和 $b$ 代表有闭合边界的图形数目。闭合边界通过 4 方向的链码表示。如果：

$$\begin{aligned} s_j(a) &= s_j(b) \qquad j = 4,6,8,\cdots,k \\ s_j(a) &\neq s_j(b) \qquad j = k+2,k+4,\cdots \end{aligned} \tag{12.3.1}$$

则这两个图形有一个相似程度 $k$。这里 $s$ 代表图形数，下标代表次序。两个图形 $a$ 和 $b$ 间的距离定义为它们相似程度的倒数：

$$D(a,b) = \frac{1}{k} \tag{12.3.2}$$

$a$ 和 $b$ 间的距离满足下列性质：

$$\begin{aligned} &D(a,b) \geqslant 0 \\ &D(a,b) = 0, \text{当且仅当 } a = b \\ &D(a,c) \leqslant \max[D(a,b),D(b,c)] \end{aligned} \tag{12.3.3}$$

无论 $k$ 或 $D$ 都可以用于两个图形的比较。如果应用了相似度，则 $k$ 越大，图形越相似(注意 $k$ 对同样的图形是无穷大的)。当使用距离度量时与使用 $k$ 时相反。

**例 12.7**　*应用图形数目对比图形*

假设有一个图形 $f$，并且希望在有 5 个其他图形($a,b,c,d$ 和 $e$)的集合中找到与它最为接近的匹配，如图 12.24(a)所示。这个问题同已有 5 个原型图形而试图寻找给定未知图形的最佳匹配相似。有图 12.24(b)中显示的相似树的辅助，这个搜索有可能被可视化。树根节点对应相似度最低的可能性，在这个例子中是 4。假设图形相似程度为 8，除了图形

$a$(它与其他所有图形的相似程度为6)。沿着树向下,发现图形 $d$ 与其他图形的相似程度为8,等等。

图形 $f$ 和 $c$ 惟一地匹配,有高于其他任何两个图形的相似度。在其他极端情况下,如果 $a$ 是一个未知图形,那么可以说,使用这种方法使 $a$ 同其他5个图形以相似度为6的程度相似。使用相似性矩阵可以得到同样的结论。相似性矩阵如图 12.24(c)所示。

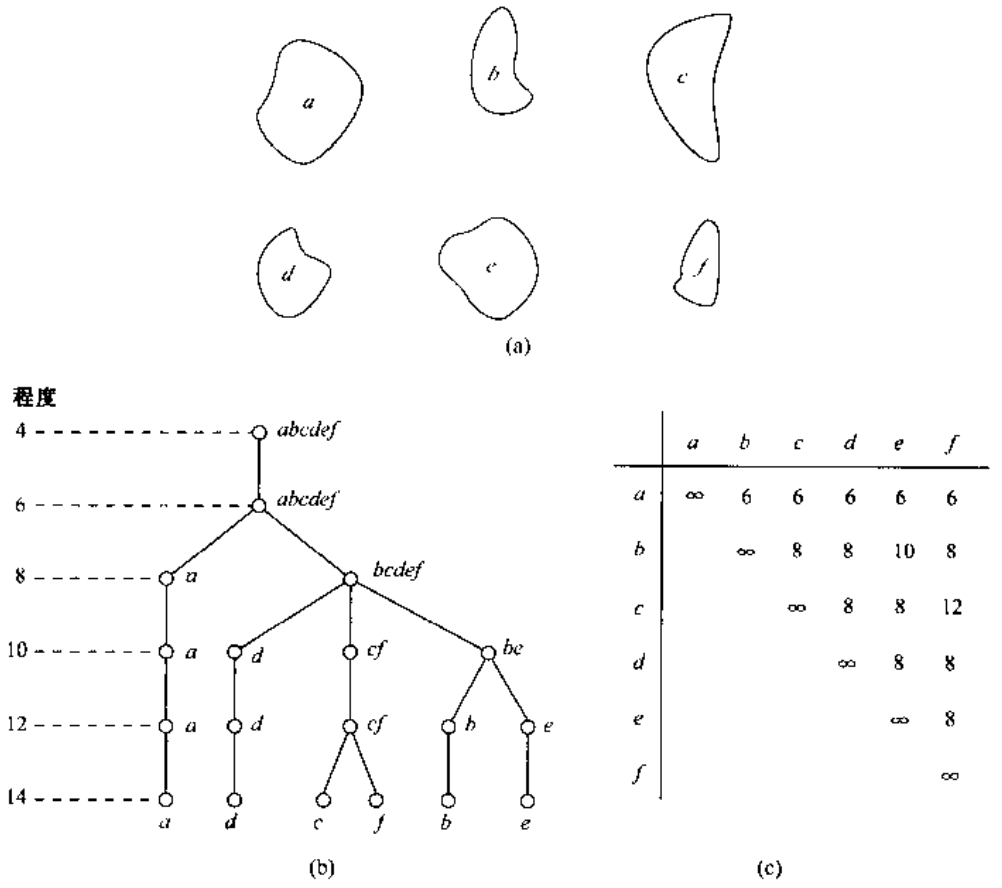



图 12.24 (a)图形,(b)假定的相似树,(c)相似矩阵(Bribiesca 和 Guzman)

## 12.3.2 串匹配

假设有两个区域边界 $a$ 和 $b$,被编码后写成串(见 11.5 节),分别用 $a_1a_2,\cdots,a_n$ 和 $b_1b_2,\cdots,b_m$ 表示。令 $\alpha$ 代表两个串之间的匹配数目,如果 $a_k = b_k$,则匹配发生在第 $k$ 个位置上。无法匹配的字符数目为:

$$\beta = \max(|a|,|b|) - \alpha \tag{12.3.4}$$

这里|arg|是串码变量的长度(字符数目)。当且仅当 $a$ 和 $b$ 相同时 $\beta=0$(见习题 12.21)。

一种判断 $a$ 和 $b$ 间相似性的简单方法是使用如下所示的比率:

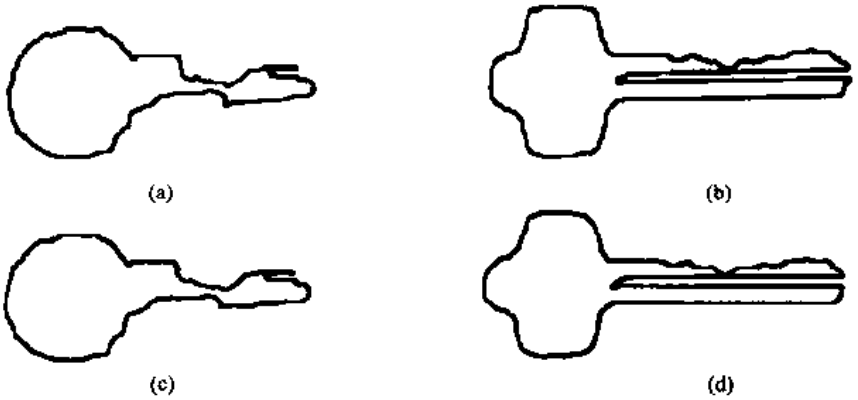
$$R = \frac{\alpha}{\beta} = \frac{\alpha}{\max(|a|,|b|) - \alpha} \tag{12.3.5}$$

对完全匹配 $R$ 为无限大,$a$ 和 $b$ 没有任何字符匹配时 $R$ 为0(此时 $\alpha=0$)。因为匹配是逐个字符

进行的，对于减少计算量来说，每条边界的起始点是很重要的。对任何标准化的或近似标准化的方法，同样的起点是很有帮助的，只要这种方法比强制匹配[从包括在每一串的任意点开始，移动串的一点，并对每一次移动计算式(12.3.5)]有计算上的优点。$R$ 的最大值给出最佳匹配。

**例 12.8**　串匹配的说明

图 12.25(a)和(b)显示了来自两个对象类的样本边界，它们使用多边形拟合近似给出(见 11.1.2 节)。图 12.25(c)和(d)分别显示了对应于图 12.25(a)和(b)中边界的多边形近似。串是由计算每一个多边形被顺时针追踪时各段间的内角形成的。将角度编成用 8 个字符之一代表的码，对应于 45°的增量，即 $\alpha_1: 0° < \theta \leqslant 45°$；$\alpha_2: 45° < \theta \leqslant 90°$；…；$\alpha_8: 315° < \theta \leqslant 360°$。



(a)　(b)　(c)　(d)

(e)

| $R$ | 1.a | 1.b | 1.c | 1.d | 1.e | 1.f |
|---|---|---|---|---|---|---|
| 1.a | ∞ | | | | | |
| 1.b | 16.0 | ∞ | | | | |
| 1.c | 9.6 | 26.3 | ∞ | | | |
| 1.d | 5.1 | 8.1 | 10.3 | ∞ | | |
| 1.e | 4.7 | 7.2 | 10.3 | 14.2 | ∞ | |
| 1.f | 4.7 | 7.2 | 10.3 | 8.4 | 23.7 | ∞ |

(f)

| $R$ | 2.a | 2.b | 2.c | 2.d | 2.e | 2.f |
|---|---|---|---|---|---|---|
| 2.a | ∞ | | | | | |
| 2.b | 33.5 | ∞ | | | | |
| 2.c | 4.8 | 5.8 | ∞ | | | |
| 2.d | 3.6 | 4.2 | 19.3 | ∞ | | |
| 2.e | 2.8 | 3.3 | 9.2 | 18.3 | ∞ | |
| 2.f | 2.6 | 3.0 | 7.7 | 13.5 | 27.0 | ∞ |

(g)

| $R$ | 1.a | 1.b | 1.c | 1.d | 1.e | 1.f |
|---|---|---|---|---|---|---|
| 2.a | 1.24 | 1.50 | 1.32 | 1.47 | 1.55 | 1.48 |
| 2.b | 1.18 | 1.43 | 1.32 | 1.47 | 1.55 | 1.48 |
| 2.c | 1.02 | 1.18 | 1.19 | 1.32 | 1.39 | 1.48 |
| 2.d | 1.02 | 1.18 | 1.19 | 1.32 | 1.29 | 1.40 |
| 2.e | 0.93 | 1.07 | 1.08 | 1.19 | 1.24 | 1.25 |
| 2.f | 0.89 | 1.02 | 1.02 | 1.24 | 1.22 | 1.18 |

图 12.25　(a)和(b)是两个不同对象类的边界，(c)和(d)是它们对应的多边形近似，(e)到(g)是 $R$ 值表(Sze 和 Yang)

图 12.25(e)显示了对象 1 的 5 个样本的度量值 $R$ 与它们自身之比的值。例如，相应于 $R$ 的全部值，符号 1.c 来自类 1 的第 3 个串。图 12.25(f)显示的结果是第 2 个对象类的串和

它们自身之比。最后,图 12.25(g)显示的是通过比较一个类和其他类的串得到的 $R$ 值表。这里请注意,所有的 $R$ 值比前两个表中的任何项都小。这表示 $R$ 度量方法在两类对象间达到了很高的辨别级别。例如,如果串 1.$a$ 的类别成员未知,将本串和类 1 的样本(原型)串相比较产生的最小 $R$ 值将是 4.7[图 12.25(e)]。相比之下,将本串和类 2 的串相比较的最大值将会是 1.24[图 12.25(g)]。从这个结果可以得出结论:串 1.$a$ 是对象类 1 的成员。这种分类方法和 12.2.1 节介绍的最小距离分类器相似。

### 12.3.3 串的语法识别

语法方法为处理结构识别问题提供了一种统一的方法。基本上,语法模式识别的主要思想是规定一个模式基元集合(见 11.5 节)、一个规定模式基元相互间关联的规则集合(采用语法的形式)和一个识别器(称为自动机),识别器的结构由语法规则集合决定。首先考虑串文法及其自动机,在下一节中将把这种思想扩展到树文法及其相应的自动机。

#### 串文法

假设有两个类 $\omega_1$ 和 $\omega_2$,它们的模式是 11.5 节所讨论方法生成的元素组成的串。可以把每个元素作为某种文法的字母表中可接受的字符,这里文法是一个句法规则的集合(因此有语法识别的名称)。这个句法集合规范着由字符生成的句子。文法 $G$ 生成的句子集合叫语言,用 $L(G)$表示。这里,句子是符号组成的串(依次代表模式),语言与模式类相对应。

考虑两个文法 $G_1$ 和 $G_2$,它们的句法规则是:$G_1$ 只允许生成与来自类 $\omega_1$ 的模式相对应的句子,$G_2$ 只允许生成与来自类 $\omega_2$ 的模式相对应的句子。带有上述特性的文法一经设定,句法模式识别过程在原理上就简单了。对于一个表示未知模式的句子,目的是判别在哪种语言下模式表示一个有效的句子。如果句子属于语言 $L(G_1)$,认为模式属于类 $\omega_1$。同样,如果句子在语言 $L(G_2)$中是正确的,则认为此句子属子类 $\omega_2$。如果句子同时属于这两种语言,就无法得到惟一的判别。对两种语言来说都不能正确识别的句子将被拒绝。

在存在两种以上的模式类时,句法分类方法与前面段落中描述的一样,只是过程中涉及的文法更多而已(每个类至少有一条对应的文法)。对多个类别的分类,如果它仅对于语言 $L(G_i)$是可用的句子,则模式属于类 $\omega_i$。与前面一样,如果一个句子属于不止一种语言,就不能做出惟一的判别。所有语言都不可用的句子会被拒绝。

在处理串时,用四元组定义一个文法:

$$G = (N, \Sigma, P, S) \tag{12.3.6}$$

这里

$N$ 是有限变元集合,称为非终端符,

$\Sigma$是有限常元集合,称为终端符,

$P$ 是重写规则集合,称为产生式,

$S$ 属于 N 集合,称为起始符。

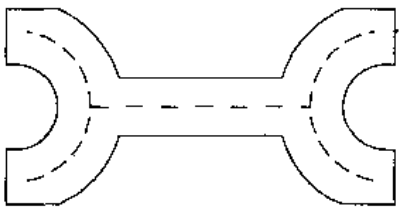
这里要求 $N$ 和$\Sigma$是不相交的集合。在下面的论述中,大写字母 $A, B, \cdots, S, \cdots$代表非终端符号。字母表中开头的小写字母 $a, b, c, \cdots$代表终端符号。字母表中末尾的小写字母 $v, w, x, y, z$ 代表终端符号串。小写希腊字母 $\alpha, \beta, \theta, \cdots$代表终端符号和非终端符号混合组成的串。

空语句(没有符号的句子)用 $\lambda$ 表示。最后,对于一个符号集合 $V$,符号 $V^*$ 表示由来自 $V$ 的元素组成的句子集合。

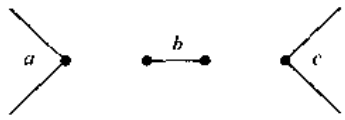
串文法由它的产生式描述其性质。在句法模式识别中,特别感兴趣的是正则文法和上下文无关文法。正则文法的产生式只有 $A \rightarrow aB$ 或 $A \rightarrow a$ 的形式。$A$ 和 $B$ 属于 $N$,$a$ 属于 $\Sigma$。上下文无关文法的产生式形式为 $A \rightarrow \alpha$,$A$ 属于 $N$,$\alpha$ 属于 $(N \cup \Sigma)^*$,即 $\alpha$ 是除空字符以外的终端符号和非终端符号构成的串。

**例 12.9**　使用正则串文法生成对象类

在开始之前,考虑一下文法生成对象类的机理是有用的。假设图 12.26(a)中显示的对象由它的(已删除的)骨架来描述,并且定义示于图 12.26(b)中的基元来描述这一(和相似的)骨架的结构。考虑文法 $G=(N,\Sigma,P,S)$,$N=\{A,B,S\}$,$\Sigma=\{a,b,c\}$ 且 $P=\{S\rightarrow aA, A\rightarrow bA, A\rightarrow bB, B\rightarrow c\}$,这里终端符号 $a,b,c$ 对应于图 12.26(b)中所示的基元。如前边所指出的那样,$S$ 是生成语言 $L(G)$ 的字串的起始符。例如,使用第一个产生式,并接下来使用两次第二个产生式得到:$S \Rightarrow aA \Rightarrow abA \Rightarrow abbA$,这里($\Rightarrow$)表示一个串从 $S$ 开始推导,使用 $P$ 集合中的产生式得到串。第一个产生式将 $S$ 重写为 $aA$,第二个产生式将 $A$ 重写为 $bA$。串 $abbA$ 中有一个非终端符,可以继续推导。例如,再使用两次第二个产生式,使用一次第三个产生式,并使用一次第四个产生式,生成的串 $abbbbbc$,对应图 12.26(c)所示的结构。使用第四个产生式后,串中就不存在非终端符了。因此,使用第四个产生式后推导就结束了。由此文法规则产生的语言是 $L(G)=\{ab^nc \mid n \geqslant 1\}$,这里 $b^n$ 表示符号 $b$ 重复 $n$ 次。换句话说,$G$ 仅能产生形如图 12.26(c)所示的骨架形式,但长度不限。



(a)




(b)

(c)

图 12.26　(a)由骨架代表的对象,(b)原始基元,(c)使用正则串文法生成的结构