

清华大学毕业设计论文

摘 要

本文分析了一种新的神经网络构造算法,并将这种算法应用于模式分类领域.文章首先分析了使用现有误差反传算法 (BP) 训练神经网络的缺点,并针对模式分类的特定领域,设计了前馈网络的构造算法,这种算法能保证对于训练数据的识别率达到 100%. 在将特征空间转换到同维的超球面后,我们分析了神经元分类功能与球面上圆域覆盖间的关系,并由此将网络的构造与寻找球面上的特定覆盖等同起来.我们给出了一种可用于多维多类的逐类覆盖法. 对于平面双螺旋、三螺旋、空间三螺旋以及胸癌数据的分类表明这一网络学习速度很快,使用的神经元数目少,并能够获得很高的识别率.最后,我们总结了这一阶段的工作,并对下一阶段的研究提出了几个富有挑战性的方向.

文中的附录给出了文中所有分类问题的求解结果,同时也介绍了我们所编制的分类、测试程序.

Abstract

In this paper, we proposed a novel construction-algorithm (FP-algorithm) for neural networks and applied it to the field of pattern recognition. After pointing out some defects of the common used BP-algorithm which tunes the network little by little, we stated that the network can be constructed by our FP-algorithm, so long as the problem is concerned with pattern recognition, and a recognition rate of 100% to the training patterns is ensured by the algorithm.

We mapped the characteristics space of patterns to a super-sphere and found out that one neuron functionally equals to one circle-cover on the sphere. So we translated the work of network construction to the work of circle-covering. A covering algorithm — C^3 (Cover one Class by one Class) — is proposed to solve problems of multi-dimension and multi-class.

The computer experiments on classification problems ‘twin-spire in plane’, ‘tri-spire in plane’, ‘tri-spire in space’, and ‘breast cancer data’ showed that very high construction speed, small amount of neurons, and high recognition rate are three major advantages of the FP-algorithm. In the last part of the paper, we summarized the recent work, and gave out some suggestions on the future work.

清华大学毕业设计论文

目 录

第一章	序言	1
§1.1	模式分类的基本概念	1
§1.2	特征空间的划分	2
§1.3	神经网络与模式分类	3
第二章	前馈网络的构造方法	5
§2.1	前馈网的网络结构	5
§2.1.1	前馈网第一层	6
§2.1.2	前馈网第二层	7
§2.2	前馈网络的学习算法	8
第三章	球面上的圆域覆盖	10
§3.1	神经元的几何分析	10
§3.1.1	超平面表示	10
§3.1.2	球面上的圆域覆盖	11
§3.2	圆域的分类能力	13
§3.3	前馈网络与圆域覆盖	15
第四章	前馈网络分类过程	18
§4.1	平面双螺旋问题	18
§4.2	逐类覆盖法	19
§4.3	网络第二层权值的设置	21
§4.4	平面双螺旋问题的分类结果	23
§4.5	小结	25
第五章	其它分类例子	27
§5.1	文件格式及测试程序	27
§5.2	平面三螺旋问题	28
§5.3	空间三螺旋问题	31
§5.4	Wisconsin 胸癌数据	32
第六章	初步结论及进一步的工作	34

清华大学毕业设计论文

致谢	35
参考文献	36
附录A 分类结果	37
§A.1 文中所有分类问题的基本数据	37
§A.2 平面双螺旋的分类结果	38
附录B 文件格式与程序	40
§B.1 文件格式	40
§B.2 第一套程序的结果	40
§B.3 第二套程序的结果	42

清华大学毕业设计论文

插图目录

1.1 采用决策论的模式识别系统	1
3.1 空间球面上的圆域覆盖	11
3.2 将 \mathcal{D} 向“上”投影到球面 S^n	12
3.3 $w_{n+1} > 0$ 时的覆盖情况 ($n = 2$)	15
3.4 $w_{n+1} < 0$ 时的覆盖情况 ($n = 2$)	16
4.1 平面双螺旋分类问题	19
4.2 采用逐类覆盖法对平面双螺旋分类问题的求解结果	24
5.1 平面三螺旋分类问题	28
5.2 采用逐类覆盖法对平面三螺旋分类问题的求解结果	29
5.3 加入边界阈值控制后逐类覆盖法对平面螺旋问题的求解结果	30
5.4 空间三螺旋分类问题	31
B.1 第一套程序的一次运行输出	42
B.2 第二套程序界面一	43
B.3 第二套程序界面二	43

表格目录

A.1 用逐类覆盖法构造网络所得的分类结果	37
A.2 加入阈值控制后逐类覆盖法所得的分类结果	37
A.3 逐类覆盖法对训练数据 1.1 的覆盖结果及网络第二层参数	38
A.4 逐类覆盖法对训练数据 1.2 的覆盖结果及网络第二层参数	39
A.5 阈值控制的逐类覆盖法对训练数据 1.2 的覆盖结果	39
B.1 数据文件格式一	41
B.2 数据文件格式二	41
B.3 覆盖数据文件格式	41

清华大学毕业设计论文

第一章 序 言

在这一章中，我们简单地阐明了模式分类问题的内涵，列举了历史上使用过的一些分类方法，以及神经网络在模式分类上的应用。

§1.1 模式分类的基本概念

一般说来，模式分类问题指的是对一系列过程或事件的分类与描述^[1]。要加以分类的一系列过程或事件可以是一系列物理的对象，也可以是一些比较抽象的如心理状态等。具有某些相类似的性质的过程或事件就分为一类。模式类别的数目总是由特定的问题或应用所决定的。例如，考虑国标一、二级汉字的识别问题时，就应该是个 6763 类的问题；但如果我们只想将汉字和英文字母区分开，则要解决的就只是一个两类问题。用来解决模式识别问题的许多不同的数学技巧可以综合成两种一般的方法，即决策论方法 (或统计法) 及句法方法 (或语言结构法)^[1]。在本论文中，我们只讨论决策论的方法。

决策论方法中，作为模式类特性度量的特征首先被抽取出来；而把每一个模式指定到某个模式类的识别通常是用划分特征空间的办法来实现的^[2]。这一方法通常包括两个问题。第一个问题是应该从输入模式中提取什么样的特征，目前在这方面的理论并不多，现有的研究主要是关于如何从一个给定的度量集合中选择一个子集或是确定特征的次序的。第二个问题是根据所选择的特征度量进行分类的问题，这也是这篇论文主要讨论的问题。图 1.1 给出了一个识别系统的简单方框图，虽然并不是必须分成两部分。

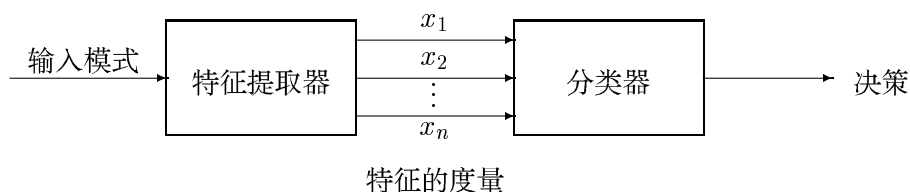


图 1.1: 采用决策论的模式识别系统

清华大学毕业设计论文

§1.2 特征空间的划分

模式分类的概念可以通过把特征空间加以划分 (或认为是从特征空间到决策空间的映射) 来说明. 假定从每个输入模式中测得 n 个特征, 或者我们说, 输入的样本是 n 维的向量 $x = (x_1, x_2, \dots, x_n)^T$, 其中 x_t 表示第 t 个特征的度量, $t = 1, 2, \dots, n$. 用 $\mathcal{J}_i(x)$ 表示与第 i 类模式 ($i = 1, 2, \dots, k$) 相联系的判别函数, 则对于所有的属于类 i 的样本 x , 通用的判定依据是:

$$\mathcal{J}_i(x) > \mathcal{J}_j(x), \quad i, j = 1, 2, \dots, k, \quad j \neq i.$$

而相应地, 在特征空间 \mathcal{D} 中模式类 i 与 j 的边界就可由下述方程表示:

$$\mathcal{J}_i(x) - \mathcal{J}_j(x) = 0.$$

在以往的研究中, 基本的有以下几种判别函数:

1. 线性判别函数: 设 $w^i = (w_1^i, w_2^i, \dots, w_n^i)^T$ 是类 i 的线性函数的权重, θ_i 是类 i 线性函数的阈值, 则

$$\mathcal{J}_i(x) = (w^i)^T x - \theta_i = \sum_{t=1}^n w_t^i x_t - \theta_i. \quad (1.1)$$

这事实上是用超平面来划分特征空间.

2. 最小距离法: 设在每一类中设置参考向量 R_i , 若 x 与向量 R_i 距离最小, 则 x 属于类 i . 即

$$\mathcal{J}_i(x) = -|x - R_i|.$$

如果采用欧式距离, 则判别函数也可改写成:

$$\mathcal{J}_i(x) = R_i^T R_i - x^T R_i - R_i^T x.$$

即最小距离分类器也是一个线性分类器.

3. 最近邻法: 可以将最小距离法的概念推广到一些参考向量集合上去. 令 p_i 表示类 i 的参考向量集的容量, 而 $R_{i,j}$ ($j = 1, \dots, p_i$) 是具体的参考向量, 则

$$\mathcal{J}_i(x) = \max_{j=1, \dots, p_i} \{R_{i,j}^T R_{i,j} - x^T R_{i,j} - R_{i,j}^T x\}$$

这是一个分段线性的判别函数.

4. 其它还有诸如多项式等判别函数.

这些较为传统的分类法, 将特征空间以刚性边界划分成了许多小块. 由于空间划分上的复杂性, 以及真实世界所必定存在的模糊和不确定性, 各种新兴的技术如模糊逻辑、遗传算法、竞争学习等也被引入了分类领域. 但这些算法最终都还是要涉及到特征空间的划分这一问题, 只是在具体细节如划分边界的具体形式上有不同的差别.

§1.3 神经网络与模式分类

从上一节可以看出, 模式分类与特征空间的划分有着很大的关系. 由于神经网络在函数映射领域中的突出表现, 有不少学者将其用于模式识别、分类领域, 籍神经网络对于任意函数的任意精度的逼近能力^[3]来产生模式类之间的决策面, 而取得了不少成果. 如 [4] 所述, 对于一般的使用 BP 算法的多层前馈网络来说, 有一个隐层的神经网络就可以形成任意的开凸区域或闭凸区域, 而有着两个隐层的神经网络则可以形成任意形状的划分区域.

但是, 一般依靠 BP 算法进行修正学习的网络有着以下几个主要的缺点: 一是学习收敛速度慢, 耗时长; 二是网络的容错能力差; 三是可能产生假吸引中心. 这三个缺点, 很大地影响了多层神经网络在分类, 尤其是大样本分类中的应用.

本文就是试图克服神经网络在模式分类问题中的上述缺点, 通过抛弃其误差修正的学习算法, 而采用变换、覆盖、构造的方法来构造分类网络. 这种网络, 正如我们在文中将会看到的那样, 具有学习速度快、使用神经元少、识别率高 (对训练数据的识别率总可以达到 100%)、泛化能力好且可以较容易地利用先验知识的特点. 而且, 通过变换、覆盖的思想, 我们还有可能真正实现分类网络的数学分析.

论文第二章先对张铃、张钊教授所提出的前馈网构造方法做了简单的介绍, 同时证明了其中的一些结论; 第三章扩展了前馈网内在的构造思想, 将之与神经网络的空间分析以及空间球面上的圆域覆盖相结合, 同时通过数学分析及一个简单的例子, 证明了用圆域覆盖来划分特征空间并不会降低分类器的分类性能. 论文的第四章通过一个完整的分类例子, 不仅说明了我们的这种分类网络的三个相对独立的部分, 即空间变换、圆域覆盖、网络构造, 同时也说明了这一网络对于一般而言相当难

清华大学毕业设计论文

以解决的问题能够达到很高的识别率，初步展示了这一分类网络强大的分类能力。接下去的一章，再通过几个典型的例子，测试了网络对于多维特征以及多类模式的分类能力，并将某些结果与国际上已有的部分结果做比较，进一步分析了网络的特点和分类能力。最后，我们总结了这一初步的工作，并给出了今后对于前馈网络的研究展望。

第二章 前馈网络的构造方法

本章讨论如何通过构造的方法将神经网络应用于静态分类问题. 本章的第一部分给出了网络的结构, 相应地亦给出了其构造算法; 第二部分则简单介绍了前馈网中增量学习的概念及算法. 最后, 我们用一个简单的例子说明了网络的分类能力.

当考虑神经网络用于静态分类问题时, 一般使用的是前向网络. 模式数据从网络的第一层流入, 经过各层的运算处理, 在最后一层得到分类结果. 这时, 我们并不一定要使用误差反传 (BP) 算法来训练网络, 相反, 我们同样可以考虑也用前向传播 (FP) 的方法来确定网络各层的权值. 即先从第一层开始, 确定神经元的个数及各神经元的权、阈值, 然后以第一层网络的输出作为第二层的输入, 再确定第二层神经元的个数、权值及阈值; 以此类推, 直到确定最后一层神经元的权值和阈值.

张铃、张钊等在文献 [5] 中所提出的前馈神经网络及其构造方法, 使用了三层 (即输入层、第一层、第二层) 神经元, 能够将取值于 $\{-1, 1\}$ 的 n 维输入向量任意地映射到同样取值于 $\{-1, 1\}$ 的 m 维输出空间. 下面是其具体的网络结构.

§2.1 前馈网的网络结构

当输入向量为 x , 输出为 y 时, 前馈网所采用的基本神经元的功能如下 (类似于线性分类器中的 (1.1) 式):

$$y = \text{sgn}(w^T x - \theta) \quad (2.1)$$

其中 w 与 θ 分别为神经元的权值与阈值, 而 $\text{sgn}(\cdot)$ 为符号函数. 神经元的作用是用超平面 $w^T x - \theta = 0$ 将空间划为两半, 一半 (称为正半空间) 输出为 $+1$, 另一半 (负半空间) 输出为 -1 .

当给定 p 个样本时, 网络的输入层将样本传到第一层网络的每个神经元. 第一层网络使用了 $(p-1)$ 个神经元, 每个神经元有 n 个输入, 其功能是将 p 个样本的输入变换成 $(p-1)$ 维空间中正交单纯形的 p 个顶点. 第二层网络的作用是将正交单纯形的顶点用非线性的方法变换为所要求的输出, 其有 m 个神经元.

清华大学毕业设计论文

§2.1.1 前馈网第一层

这一层的作用是试图以不同的模式向量为中心, 并以尽可能大的半径来划分模式空间.

对于 p 个互不相同的 n 维向量 x^0, x^1, \dots, x^{p-1} , 我们可以设定网络第一层中各个神经元的权值 w^i 及阈值 θ_i 为 ($i = 1, 2, \dots, p-1$):

$$w^i = x^i, \quad \theta_i = \begin{cases} n - d_i + 1, & \text{当 } d_i \text{ 是偶数时;} \\ n - d_i, & \text{当 } d_i \text{ 是奇数时.} \end{cases} \quad (2.2)$$

式 (2.2) 中

$$d_i = \min_{j \neq i} d(x^i, x^j) > 0, \quad j = 0, 1, \dots, p-1. \quad (2.3)$$

而 $d(x, y)$ 表示 x, y 间的 Hamming 距离, 即两向量间对应分量相异的个数. 设网络的输入、输出关系为 F , x 为输入向量, 参考 (2.1) 式, 则第一层的输出 z 为

$$\begin{aligned} z &= F(x) = (z_1, z_2, \dots, z_{p-1})^T, \\ z_j &= \text{sgn} [(w^j)^T x - \theta_j] \end{aligned} \quad (2.4)$$

对于第一层神经元权值及阈值的设置, 我们有如下引理:

引理 2.1 若前馈网络第一层的权、阈值如式 (2.2) 设置, 则在 (2.4) 的输入、输出关系下, 对应于输入向量 x^0, x^1, \dots, x^{p-1} 的输出 $z^i = F(x^i)$ 满足:

$$\begin{aligned} z^0 &= (-1, -1, \dots, -1)^T, \\ z^1 &= (+1, -1, \dots, -1)^T, \\ z^2 &= (-1, +1, \dots, -1)^T, \\ &\vdots \\ z^{p-1} &= (-1, -1, \dots, +1)^T. \end{aligned}$$

这样我们就达到了将不同的模式输入映射到 $(p-1)$ 维空间中的正交单纯形的顶点的目的. 引理的证明如下: (参见 [5])

清华大学毕业设计论文

证明 (引理 2.1) 只需证: 当 $i \neq 0$ 且 $j \neq i$ 时, 有

$$(w^i)^T x^i - \theta_i > 0 \quad (2.5)$$

$$(w^i)^T x^j - \theta_i < 0 \quad (2.6)$$

注意到对任意的分量取值于 $\{-1, 1\}$ 的向量 x, y , 有

$$x^T y = n - 2d(x, y)$$

故有当 $i \neq 0$ 时

$$(w^i)^T x^i - \theta_i = \begin{cases} n - (n - d_i + 1) = d_i - 1 > 0, & d_i \text{ 为偶数;} \\ n - (n - d_i) = d_i > 0, & d_i \text{ 为奇数.} \end{cases}$$

而 $j \neq i$ 时, 有

$$\begin{aligned} (w^i)^T x^j - \theta_i &= (x^i)^T x^j - \theta_i \\ &= [n - 2d(x^i, x^j)] - (n - d_i) \leq d_i - 2d_i \\ &= -d_i < 0 \end{aligned}$$

这样 (2.5) 与 (2.6) 都已得证. □

§2.1.2 前馈网第二层

类似于第一层中的输入输出关系 F , 网络第二层的输出为:

$$\begin{aligned} y &= G(z) = (y_1, y_2, \dots, y_m)^T, \\ y_j &= \text{sgn}[(u^j)^T z - \xi_j], \quad j = 1, 2, \dots, m. \end{aligned} \quad (2.7)$$

对于任意给定的 p 个取值于 $\{-1, 1\}$ 的 m 维输出向量 $\{y^0, y^1, \dots, y^{p-1}\}$, 为了满足网络的构造要求, 即符合映射关系

$$y^i = G(z^i) = G(F(x^i)), \quad i = 0, 1, \dots, p-1 \quad (2.8)$$

我们必须寻找合适的权值 u^j 与阈值 ξ_j . 为此, 有如下引理:

清华大学毕业设计论文

引理 2.2 如果我们依 (2.9) 与 (2.10) 式设定前馈网络第二层的权值 u^j 与阈值 ξ_j , ($j = 1, 2, \dots, m$), 则网络的输出必满足 (2.8) 式.

$$u_i^j = \frac{y_j^i - y_j^0}{2} = \begin{cases} 0, & \text{当 } y_j^0 = y_j^i \text{ 时;} \\ y_j^i, & \text{其它.} \end{cases} \quad (2.9)$$

$$\xi_j = -(y_j^0 + u_1^j + \dots + u_{p-1}^j). \quad (2.10)$$

证明 (引理 2.2) 对于 $i = 0$ 的情形, 我们有 $z^0 = (-1, -1, \dots, -1)^T$, 故有

$$\begin{aligned} (u^j)^T z^0 - \xi_j &= -(u_1^j + u_2^j + \dots + u_{p-1}^j) + (y_j^0 + u_1^j + \dots + u_{p-1}^j) \\ &= y_j^0 \end{aligned}$$

而当 $i > 0$ 时, 注意到式 (2.9) 中 u_i^j 的定义, 我们有

$$\begin{aligned} (u^j)^T z^i - \xi_j &= -u_1^j - \dots - u_{i-1}^j + u_i^j - u_{i+1}^j - \dots - u_{p-1}^j + (y_j^0 + u_1^j + \dots + u_{p-1}^j) \\ &= y_j^0 + 2u_i^j = y_j^i \end{aligned}$$

故对所有的 $i = 0, \dots, p-1$, 我们都有 $(u^j)^T z^i - \xi_j = y_j^i$, 即有式 (2.8) 成立. \square

综合引理 2.1 与 2.2, 我们可以得到关于前馈网的一个简单但又重要的定理:

定理 2.1 当给定训练样本集 $\{x^0, x^1, \dots, x^{p-1}\}$, 按 (2.2) 式定义 w^i 和 θ_i , 按 (2.9), (2.10) 式定义 u^j 和 ξ_j , 则前馈网络将输入 x^i 变换为 y^i , $i = 0, 1, \dots, p-1$, 即每个样本输入向量都是吸引中心.

证明 (定理 2.1) 显然. \square

另外, 对于网络吸引域的分析也表明, 不仅每个输入的模式样本是某一吸引域的中心, 而且网络除输入样本外不存在别的吸引中心, 即不存在假吸引中心^[5]. 这是非常难得的优点.

§2.2 前馈网络的学习算法

在网络的结构中, 我们已经给出了前馈网络各层神经元的参数设置算法, 即网络学习算法. 这种学习是一次性的, 不仅其计算量小 (就计算量的阶而言可谓最小),

清华大学毕业设计论文

不会产生假吸引子, 而且还能够很容易地在获取新知识 (对分类来说是新的模式) 后扩充网络, 容纳新内容.

当网络已学习了 p 个样本后, 再学习第 $(p+1)$ 个样本, 这种情况我们称之为知识的扩充. 设新增的第 $(p+1)$ 个样本为 x^p , 则扩充的算法为:

1. 在第一层增加一个神经元, 取其权值 $w^p = x^p$.
2. 令 $d_i(t)$ 是根据前 $(t+1)$ 个样本计算得到的 d_i 值. 我们可令

$$d_i(p) = \min \{d_i(p-1), d(x^i, x^p)\}, \quad i = 1, 2, \dots, p-1,$$

同时我们也令 $d_p(p) = \min d(x^p, x^j)$, $j = 0, 1, \dots, p-1$. 在此基础上, 再依据 (2.2) 式更新所有 θ_i ($i = 1, \dots, p$) 的值. 特别地,

$$\theta_p = \begin{cases} n - d_p + 1, & \text{当 } d_p \text{ 是偶数时;} \\ n - d_p, & \text{当 } d_p \text{ 是奇数时.} \end{cases}$$

3. 将网络第二层的各神经元都增加一个输入, 接收第一层中新增神经元的输出. 其对应的权向量分量 u_p^i ($i = 1, 2, \dots, m$) 为:

$$u_p^i = \frac{y_i^p - y_i^0}{2} = \begin{cases} 0, & \text{当 } y_i^0 = y_i^p \text{ 时;} \\ y_i^p, & \text{其它.} \end{cases}$$

4. 将 ξ_i 改为 $\xi_i - u_{p-1}^i$, $i = 1, 2, \dots, m$.

经过这样的修改, 网络就将第 $(p+1)$ 个样本也学习进去了. 这实际给出了前馈网络的增量式的学习算法. 这种算法的时间复杂度很低, 容易看出其复杂度和网络已有的规模 (n, m, p) 仅成线性的关系. 这种算法的另一个优点是其在学习新知识的同时, 对已学习的旧知识基本保持原来状态, 即能够保持知识的稳定性. 而传统的 BP 算法往往无法做到这一点.

第三章 球面上的圆域覆盖

在这一章中，我们将进一步从特征空间划分的角度来讨论前面所提出的前馈网络；通过一种较为清晰直观的几何表示方法，我们讨论了将分类问题转化为球面上的覆盖问题的可能途径，并深入讨论了两者的关系。本章的内容主要偏重于数学上的分析，而下一章则通过一个完整的分类例子从应用的角度展示了分类、网络构造以及覆盖间的关系。

容易看出，前述的前馈网络将任意的映射分成了两步：第一步，先将所有 p 个输入重新编排，成为 $(p-1)$ 维空间中 p 个坐标十分有规律的向量——它们的长度是一样的，且分量的设置类似于模式样本的特征函数。之后的第二步，则是将前面得到的向量根据分类器对于模式的期望输出做进一步的变换，而这类变换，由于建立在第一步的基础上，成为纯粹的数字或是编码变换，已和最初具体的分类问题无关了。

从对模式样本聚类、或是对模式类分簇的角度来看，前述的前馈网络是先将属于同一类的不同样本分开，即在同一类中每一样本都单分为“簇”，之后在第二层中再将属于同一类的模式向量映射到一起。其实，我们可以考虑在第一步中就将充分照顾到属于同一类的样本点在特征空间中的分布，将几个相互距离比较近的样本点合在一起而使用同一个神经元。这样做，不仅可以减少第一层中的神经元个数，而且从直观上来说，也能增加网络的泛化能力。

我们在本章中的任务，就是要在几何上说明“将一些样本点合起来使用同一个神经元”是如何与覆盖问题结合起来，并对前馈网的构造产生影响的。在本章的稍后部分，等到我们分析了前馈网在特征空间划分中所生成的基本图象后，我们能够更清楚地认识前馈网络的本质，也能够更好地设计网络的学习算法。

§3.1 神经元的几何分析

§3.1.1 超平面表示

考虑 (2.1) 式所给出的神经元模型，一般认为 $w^T x - \theta = 0$ 代表了 n 维空间中的一个超平面 P 。当 $w^T x - \theta > 0$ 时，向量 x 位于平面 P 所划分的正半空间中，此时神

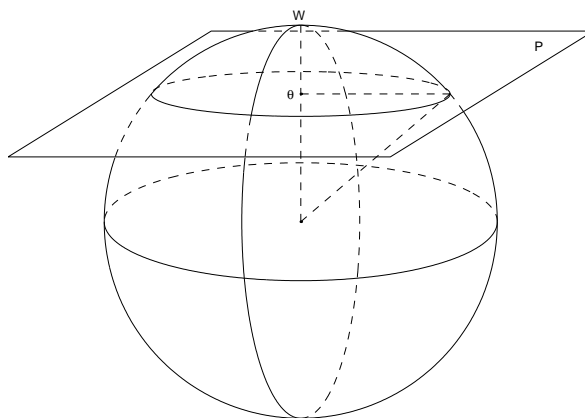


图 3.1: 空间球面上的圆域覆盖

经元输出 $y = 1$; 同样, 当 $w^T x - \theta < 0$ 时, 向量 x 位于平面 P 所划分的负半空间, 且输出 $y = -1$. 这样, (2.1) 所给出的神经元, 就具备了判断输入究竟位于由平面 P 所划分的空间中的哪一个半空间的作用. 一些研究者曾利用了神经元的这一几何特性来分析神经网络的行为^[6, 7], 但可惜的是当空间的维数 n 与神经元的数目 (与超平面 P 的个数有关) 较大时, 超平面间的相互切割就会使空间的分析十分地困难. 因此, 直到现在, 这种空间表示依然无法有效地用于复杂神经网络的性能分析.

§3.1.2 球面上的圆域覆盖

为了克服采用超平面分析空间划分的困难, 我们尝试采用神经元对于空间划分的另一种几何表示, 即球面上的圆域覆盖. 为了更形象地说明问题, 我们采用了三维空间中的图形与术语, 但这一概念却是可以用于多维的.

首先, 我们假定每个输入向量 x 都有着相同的长度 (长度不同的一般情形在后面将讨论). 这样, 所有的输入向量就被限制在一个球面 S^n 上 (图 3.1). 这时, $w^T x - \theta = 0$ 所代表的超平面 P 与 S^n 的交线 — 三维空间中是一个在 S^n 上的圆 (如果不考虑极端情况的话) — 就成了分类的界限. 类似于上一小节中关于超平面划分空间的讨论, 当输入 x 落在圆内¹时, 输出 $y = 1$; 当输入 x 在圆外时, 输出 $y = -1$. 而 $w^T x - \theta = 0$ 也成了球面上圆域的边界方程. 如果权重 w 的长度也和输入向量的长度相同, 则其就是球邻域在圆内的中心.

¹由于球面的特点, 我们所说的“圆内”与“圆外”是相对的. 这里我们姑且假定“圆内”处于 P 所划分的正半空间, 而“圆外”则处于负半空间.

清华大学毕业设计论文

这样，我们就将神经元的功能与球面上的圆域联系起来，多个神经元间的组合就可以用球面上多个圆域间的交或并来表示。这种比较清楚的空间上的关系，有利于对神经网络总体分类性能的评价。

一般说来，输入向量长度并不会正好相同。这时我们可以做某种变换操作将所有的输入向量变换为长度相同的向量。最常用的方法是归一化 (normalization)，用

$$x \mapsto \frac{x}{|x|}$$

将输入向量的长度均变为 1。但这样使得原先 n 维的向量事实上变成了 $n-1$ 的向量，丢失了部分信息。另一种类似但不丢失信息的方法是将 n 维空间的输入向量集 $\mathcal{D} (\subseteq \mathcal{R}^n)$ 变换到 $(n+1)$ 维空间中的球面 \mathcal{S}^n 上去 (球面上的点实际可认为是 n 维的)：

$$T: \mathcal{D} \rightarrow \mathcal{S}^n, \quad x \mapsto \left(x^T, \sqrt{d^2 - |x|^2} \right)^T. \quad (3.1)$$

其中 d 为输入向量长度的上限，即 $d \geq \max_{x \in \mathcal{D}} |x|$ 。这样，变换 T 就将 \mathcal{D} 中的点向“上”投影到球面 \mathcal{S}^n 上 (如图 3.2)。

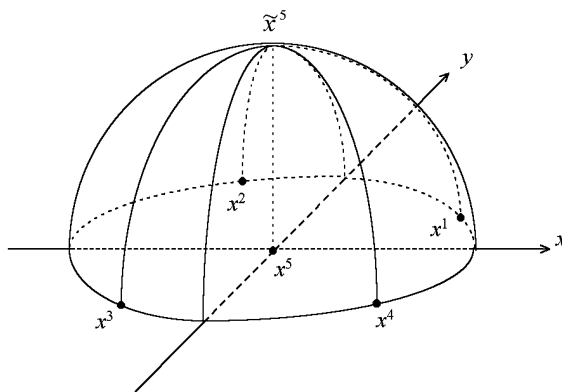


图 3.2: 将 \mathcal{D} 向“上”投影到球面 \mathcal{S}^n 。

一个简单的例子可以说明这种变换以及由此带来的空间划分上的变化：

例 3.1 给定平面上的 5 个点：

$$\mathcal{D} = \{x^1, x^2, x^3, x^4, x^5\} = \{(1, 1), (-1, 1), (-1, -1), (1, -1), (0, 0)\}.$$

如图 3.2，我们要用一个神经元将 \mathcal{D} 分为两类 $\mathcal{D}^1 = \{x^1, x^2, x^3, x^4\}$ 与 $\mathcal{D}^2 = \{x^5\}$ 。

清华大学毕业设计论文

解: 我们先取 $d = \sqrt{2}$, 用 (3.1) 将 \mathcal{D} 转换成三维空间中向量长度均为 $\sqrt{2}$ 的点集, 即

$$\begin{aligned}\tilde{\mathcal{D}} &= \{\tilde{x}^1, \tilde{x}^2, \tilde{x}^3, \tilde{x}^4, \tilde{x}^5\} \\ &= \left\{ (1, 1, 0)^T, (-1, 1, 0)^T, (-1, -1, 0)^T, (1, -1, 0)^T, (0, 0, \sqrt{2})^T \right\}.\end{aligned}$$

从图 3.2 中容易看出, 确实只需要一个神经元 (w, θ) 就可以将 $\tilde{\mathcal{D}}$ 分成两类, 即 $\tilde{\mathcal{D}}^1$ 与 $\tilde{\mathcal{D}}^2$. 例如, 可取 $w = (0, 0, \sqrt{2})^T$, $\theta = 1$. \square

§3.2 圆域的分类能力

上面的例子用一个神经元将 \mathcal{D} 分成了 \mathcal{D}^1 与 \mathcal{D}^2 两类. 而我们知道, 如果在原来的二维空间 (平面) 中实现这一分类, 至少要用两条直线; 反映在网络的第一层上, 就要多用一个神经元. 虽然例子很简单, 但也暗示了在实现相同分类任务的条件下, 采用圆域覆盖构造的网络使用的神经元很有可能比采用超平面的要少. 同样地, 我们还可以得出这么一个最初步的结论, 就是在神经元数相同的情况下, 圆域覆盖的分类能力至少和超平面的相当. 事实上, 我们有如下引理:

引理 3.1 设 \mathcal{D} 是空间中的点集 (模式样本集), 而 P_R 是用空间超平面对 \mathcal{D} 所做的一个划分, 则经过 (3.1) 中的变换 T 后, 存在一个相应于 P_R 的球面上的圆域划分 P_S , 它对样本集的划分结果和 P_R 一样, 使用的划分单元数 (分别为圆域数与超平面数) 也一样.

证明 (引理 3.1) 对于 P_R 中任一超平面, 设其平面方程为 $w^T x - \theta = 0$ ($|w| > 0$). 我们可以根据球面的半径 d 做如下变换:

$$\tilde{w} = \left(\frac{d}{|w|} \cdot w^T, 0 \right)^T, \quad \tilde{\theta} = \frac{d}{|w|} \cdot \theta.$$

之后我们就可以用方程 $(\tilde{w})^T \tilde{x} - \tilde{\theta} = 0$ 作为球面上圆域的方程, 其中 \tilde{x} 是 x 在球面上的对应点. 容易看出, 这一圆域与方程为 $w^T x - \theta = 0$ 的超平面的分类作用完全相同.

将 P_R 中每一个超平面均相应地转化为 P_S 中与之分类作用等价的圆域, 我们就可以证明引理 3.1. \square

清华大学毕业设计论文

引理证明中所构造的圆域，其最后一维分量，即由变换 (3.1) 所引入的那一维分量，总是 0. 对于圆域覆盖来说，这是一种很特殊的情况. 考虑到一般的圆域覆盖，我们还有下面这个引理：

引理 3.2 球面 S^n 上的任意圆域的边界，经反变换 T^{-1} 之后在空间 \mathcal{R}^n 中为一超椭圆面（或退化为超平面）.

证明 (引理 3.2) 考虑半径为 d 的球面 S^n 上的圆域，中心为 $w = (w_1, w_2, \dots, w_{n+1})^T$ ，阈值为 θ . 由 (2.1) 式知球面上点 $x = (x_1, x_2, \dots, x_{n+1})^T$ 处于圆域内的条件是：

$$\sum_{i=1}^{n+1} w_i x_i > \theta$$

利用 x 是由 n 维空间中点 $(x_1, x_2, \dots, x_n)^T$ 向“上”投到球面上这一前提，我们可将上式改写为：

$$w_{n+1} \sqrt{d^2 - x_1^2 - \dots - x_n^2} > \theta - \sum_{i=1}^n w_i x_i \quad (3.2)$$

这其实给出了球面上圆域经过反变换 T^{-1} 即对 n 维空间的投影后的区域方程. 下面对 w_{n+1} 的符号进行讨论，来分析区域的形状. 注意 \mathcal{D} 中所有样本在原先的 n 维空间中也都是处于一个半径为 d 的球内的.

1. $w_{n+1} = 0$. 此时圆退化为超平面，区域退化为半空间. 即引理 3.1 中的情形.
2. $w_{n+1} > 0$. 此时 (3.2) 可进一步化为：

$$\sum_{i=1}^n w_i x_i > \theta \quad \text{或} \quad w_{n+1}^2 (d^2 - x_1^2 - \dots - x_n^2) > \left(\theta - \sum_{i=1}^n w_i x_i \right)^2$$

也即区域为半空间与超椭球的并（再与 \mathcal{R}^n 中半径为 d 的球相交）. 其中半空间 $\sum_{i=1}^n w_i x_i > \theta$ 与半径为 d 的球的交集是否为空应视球面上圆域的半径而定；当圆域的半径大到使圆域中部分点的第 $(n+1)$ 维分量为负时，交集不为空. $n=2$ 时的两种可能的覆盖情况如图 3.3.

椭圆边界的方程可进一步写为：（经相似变换及平移变换后可望转成标准的椭圆方程）

$$\sum_{i=1}^n (w_{n+1}^2 + w_i^2) x_i^2 + 2 \sum_{i=2}^n \sum_{j=1}^i w_i w_j x_i x_j - 2\theta \sum_{i=1}^n w_i x_i + \theta^2 - w_{n+1}^2 d^2 = 0 \quad (3.3)$$

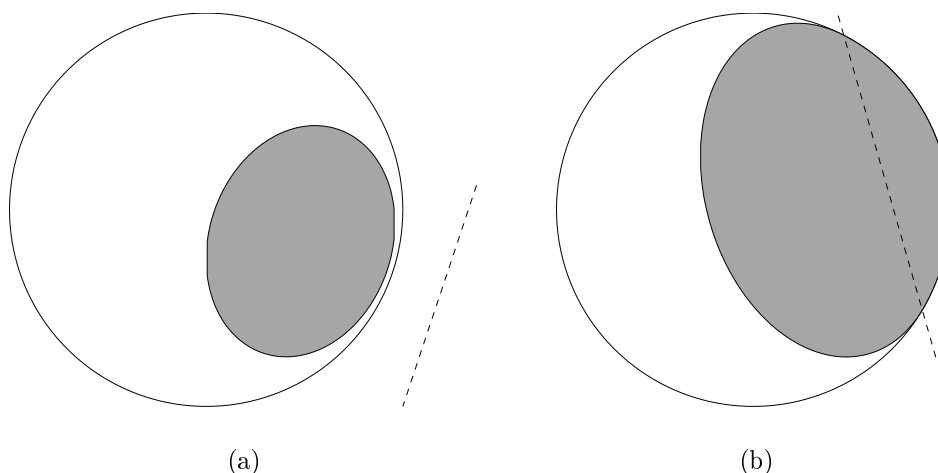


图 3.3: $w_{n+1} > 0$ 时的覆盖情况 ($n = 2, d = 1$), 图中虚线方程为 $\sum_{i=1}^n w_i x_i = \theta$, 灰色区域为圆域覆盖的投影. (a) $w = (0.6, -0.2, \sqrt{0.6})^T$, $\theta = 0.8$, 圆域半径较小, 半空间 $\sum_{i=1}^n w_i x_i > \theta$ 与半径为 d 的圆不相交; (b) $w = (0.7, 0.2, \sqrt{0.47})^T$, $\theta = 0.5$, 圆域半径较大, 区域为椭圆边及圆边共同所围.

3. $w_{n+1} < 0$. 这时 (3.2) 式又可写为

$$\sum_{i=1}^n w_i x_i < \theta \quad \text{且} \quad w_{n+1}^2 (d^2 - x_1^2 - \cdots - x_n^2) < \left(\theta - \sum_{i=1}^n w_i x_i \right)^2$$

类似于 $w_{n+1} > 0$ 时的分析, 我们知道根据超平面 $\sum_{i=1}^n w_i x_i = \theta$ 与原空间中球的相交情况, 我们又有两种可能的覆盖情况, 如图 3.4. 而区域中椭球的边界方程仍为 (3.3).

综合上面三种情况, 圆域经反变换 T^{-1} 后边界要么退化为超平面, 要么是一椭球, 因此引理成立. □

上面的引理说明我们所用的变换以及圆域覆盖并没有减少对于模式样本的分类能力; 相反地, 经变换后我们用圆域覆盖可以做到比以前用超平面分类更多的事.

§3.3 前馈网络与圆域覆盖

现在再回过头来看上一章所说的前馈网络. 仔细观察式 (2.2) 与 (2.3), 可以看出: (1) 所有模式均在半径为 \sqrt{n} 的球面上; (2) 第一层网络中的每个神经元都定义

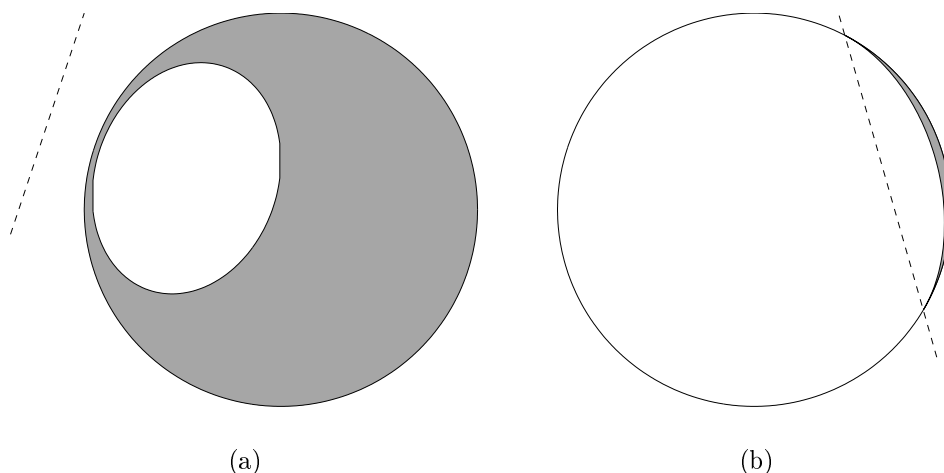


图 3.4: $w_{n+1} < 0$ 时的覆盖情况 ($n = 2, d = 1$), 图中虚线方程为 $\sum_{i=1}^n w_i x_i = \theta$, 灰色区域为圆域覆盖的投影. (a) $w = (0.6, -0.2, -\sqrt{0.6})^T, \theta = -0.8$, 圆域半径较小, 半空间 $\sum_{i=1}^n w_i x_i < \theta$ 覆盖了整个半径为 d 的圆; (b) $w = (0.7, 0.2, -\sqrt{0.47})^T, \theta = 0.5$, 区域很小, 为一窄条.

了一个以训练模式为中心, 以其到其它模式的最小距离的一半为半径的圆域; (3) 凡是落在某圆域内的点, 其对应神经元的输出就是 1, 反之则为 -1. 这其实是用一个样本定义一个圆域的方法, 来对样本空间 \mathcal{D} 进行划分.

在明确了前馈网中神经元的这一功能后, 我们就有可能自觉地利用圆域覆盖与神经元间的关系, 用尽可能少的圆域 (即对应尽可能少的神经元) 来覆盖所有的训练样本, 同时又尽可能地将输入空间划分得“合理”一些, 使多个圆域覆盖组合的形状更贴近样本自然的分布. 之后再将覆盖的结果转化为神经元的参数, 完成分类网络的构造. 下面, 我们将把这一过程说得更为详细一些, 而在下一章, 则会通过一个具体例子来说明分类过程.

我们将模式样本映射到球面 S^n 上之后, 就可以用多个圆域来覆盖球面上的样本. 简单一点, 我们可以令同一个圆域盖住的样本都是属于同一类的, 这可称之为简单覆盖; 而复杂一些, 则可以利用多个圆域间的交、并、差等操作来产生多个由圆域的弧所围成的不规则小区域, 同时保证每个小区域中的样本总是属于同一模式类的, 这可称为复杂覆盖. 无论是简单覆盖中的圆域, 还是复杂覆盖中的不规则小区域, 都形成了特征空间在映射到球面 S^n 之后的空间划分.

在得到了对于所给样本在 S^n 上的一种覆盖方案后，我们就可以设计前馈网的第一层，使其同样起到空间划分的作用。对于每一个覆盖 C_i ，我们可以为第一层相应地设置一个神经元，取其权值为覆盖的中心，而阈值为覆盖的半径，则神经元的输出就表明了样本点是否落在覆盖内。对一简单覆盖来说，对单个样本，最多只可能有一个神经元输出为 +1，而其余的输出应是 -1。这正好是第二章中前馈网第一层输出的结果。而对于复杂覆盖来说，则有可能多个神经元都输出 +1，这时这些 ± 1 的组合就反映了球面上圆域覆盖间的组合，即反映了不规则小区域的信息。这样，若我们用 m 个覆盖盖住了所有的训练样本，则我们所构造的前馈网的第一层也就有 m 个神经元，同时第一层的输出反映了空间划分的信息。

而网络的第二层，则可以通过对划分信息的再编码，而最后起到分类的作用。对于简单覆盖，由前可知我们已经可以直接构造网络的第二层；而对于复杂覆盖，一般的方法是将第二章所述的前馈网络“接”在现在构造的第一层之后，一共用四层网络完成分类。对于某些特殊的复杂覆盖，正如我们将在下一章中看到的那样，还是可以只用三层就完成分类的。

通过清晰直观的空间几何上的分析，我们将模式样本的分类问题转化并分解成了三个部分：一是特征的抽取以及到球面 S^n 的映射，这一步需要分类问题的特殊（先验）知识，也可以通过参数试凑的方法来解决；二是对映射到球面 S^n 之后的样本，采用某种覆盖算法将其用圆域划分开来，这一步与前一步有关，也需要一些先验知识，但是我们可以预先设计一些各具特点的覆盖算法以供实际应用时选用；第三步，就是由覆盖算法所得的划分结果构造前馈网络，这一步是固定且高效的。由于特征抽取（第一步）与我们研究的课题有一些距离，因此论文中没有过多地涉及这一问题，而仅仅简单地采用了式 (3.1) 的变换方法。

第四章 前馈网络分类过程

前几章我们从数学的角度,给出了前馈网的模型及几何意义,分析了其与空间球面上圆域覆盖的关系,并给出了将覆盖结果用于网络构造的三个步骤.在这一章中,我们将针对平面双螺旋分类问题,将使用前馈网进行分类的全过程展现给读者.我们首先介绍本章中作为例子的平面双螺旋分类问题,说明我们使用的训练数据及测试数据;之后,给出用于这一例子的逐类覆盖算法以及网络第二层的参数构造算法;最后再对网络的分类能力进行测试.当然,这里的前馈网,是指经覆盖之后构造的网络,并不是指在第二章中我们所述的最初的前馈网.

§4.1 平面双螺旋问题

参考 [8], 我们定义平面双螺旋问题如下:

分类问题 1 在平面极坐标系中, 我们定义如下两条曲线 K_1 与 K_2 (即两类数据)

$$K_1: \rho = \theta, \quad K_2: \rho = \theta + \pi.$$

如图 4.1 所示, 这是两条在平面上交错的螺旋线, 其中 $\frac{\pi}{2} \leq \rho \leq 6\pi$. 这一问题要求根据输入的平面坐标值将这两条曲线区分开来.

由于曲线的交错及螺旋性, 使得这一问题对许多传统的分类方法, 以及以往依赖于 BP 学习的神经网络来说都是很难的.

我们将使用如下的训练数据以及测试数据来训练和测试网络:

训练数据 1.1 在 K_1, K_2 两条曲线上各取 78 个样本点, 按 ρ 从小到大排列, 即取

$$\rho = \left(i + \frac{j}{i+1}\right) \cdot \frac{\pi}{2}, \text{ 及 } \rho = 6\pi. \quad (4.1)$$

其中 $i = 1, \dots, 11; j = 0, \dots, i$.

测试数据 1.1 曲线 K_1 及 K_2 上随机各取 10000 个点作为测试样本.

在使用变换 $T(3.1)$ 式时, 我们取变换的半径 $d = 19.3496$ (这个值是平面上样本的最小半径加 0.5). 网络的输入, 就采用变换后点在球面上的笛卡尔坐标.

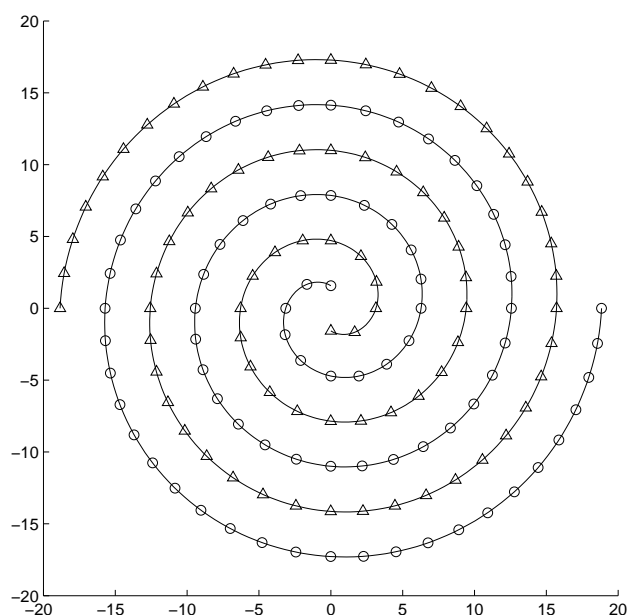


图 4.1: 平面双螺旋分类问题. 图中曲线为双螺旋线, \circ 及 \triangle 表示两类的训练样本点.

§4.2 逐类覆盖法

文献 [8] 中所提出的交叉覆盖法, 是针对两类问题而设计的. 我们针对多类问题, 扩展并改进了交叉覆盖法, 使其不仅适于两类样本的分类问题, 而且还可用于多类分类问题. 这就是本节将要叙述的逐类覆盖法. 算法的思路是: 先对第 1 类模式寻找一个圆域覆盖, 它只盖住类 1 中的样本; 之后将类 1 中被覆盖的样本删去, 再对第 2 类模式寻找一个覆盖, 同样它也只盖住类 2 中的样本; 删去类 2 中被盖住的样本; $\dots\dots$ 对每一类模式都进行同样的覆盖、删除操作, 直到每类非空样本都已形成一个覆盖, 这样就结束了一轮的覆盖操作. 之后再从第 1 类开始, 不断循环覆盖, 直到所有的样本都已被删去.

算法 1 逐类覆盖法: 对第 i 类样本 K_i 执行如下算法:

- S1 寻找属于 K_i 的一个未覆盖住的样本点. 如果没有找到, 则转下一类 K_{i+1} .
- S2 初始化: 设置 S1 中所找到的样本为当前覆盖 C_k 的中心 a_k , 当前覆盖样本集只含 a_k , 而设置覆盖样本数 $m = 0$, 覆盖半径 $d_k = 0$.
- S3 对当前覆盖内的样本点求重心 \bar{a}_k .

清华大学毕业设计论文

S4 暂以 \bar{a}_k 为新覆盖中心, 寻找最大的覆盖半径 d'_k . d'_k 可按如下方法确定:

$$\begin{aligned} d'_1 &= \max_{x \notin K_i} \{(\bar{a}_k)^T x\} \\ d'_2 &= \min_{x \in K_i} \{(\bar{a}_k)^T x | (\bar{a}_k)^T x > d'_1\} \\ d'_k &= \frac{d'_1 + d'_2}{2} \end{aligned} \quad (4.2)$$

S5 计算在以 \bar{a}_k 为中心, d'_k 为覆盖半径的区域中样本点的个数 m' . 如果 $m' > m$, 则将新的覆盖设为当前覆盖, 即 $a_k \leftarrow \bar{a}_k$, $d_k \leftarrow d'_k$, $m \leftarrow m'$, 更新当前覆盖样本集合, 转 S3. 反之, 则保持当前覆盖不变, 转 S6.

S6 (平移) 求当前覆盖中心 a_k 的平移点 \bar{a}_k

S6.1 令 $\alpha = a_k$, β 是不属于类 K_i 的离 α 距离最近的一个样本点, 即

$$\beta \notin K_i, d(\alpha, \beta) = \min_{x \notin K_i} d(\alpha, x);$$

并令 $k = 0$, $c_0 = \beta$, 集合 $P_0 = \{c_0\}$.

S6.2 令 $E_k = \left\{ x \mid x \notin K_i \cup P_k \text{ 且 } (\alpha - \beta)^T(x - \beta) > 0 \right\}$. 若 E_k 为空, 则取 $\bar{a}_k \leftarrow \alpha - \beta$ 并做规范化的操作, 即

$$\bar{a}_k \leftarrow \frac{d}{|\bar{a}_k|} \bar{a}_k, \quad (4.3)$$

然后转 S7; 反之转 S6.3.

S6.3 求

$$d = \min_{x \in E_k} \left\{ \frac{|x - \beta|^2}{2(\alpha - \beta)^T(x - \beta)} \right\}$$

且找出 $x' \in E_k$, 使得 $\frac{|x' - \beta|^2}{2(\alpha - \beta)^T(x' - \beta)} = d$, 则可令 $\bar{a}_k = \beta + d(\alpha - \beta)$ 并做 (4.3) 规范化的操作, 再令 $c_{k+1} = x'$, $P_{k+1} = P_k \cup \{c_{k+1}\}$.

S6.4 $k \leftarrow k + 1$. 若 $k = n$, 则 \bar{a}_k 即为所求, 转 S7. 反之求 \bar{a}_k 到 P_k 的投影 β' , 令 $\beta \leftarrow \beta'$, $\alpha \leftarrow \bar{a}_k$, 转 S6.2.

S7 类似 S4 与 S5, 对 \bar{a}_k 计算新覆盖半径并判断新覆盖是否可取代当前覆盖. 若能, 则以新覆盖取代当前覆盖, 转 S3; 否则转 S8.

S8 记录当前覆盖 C_k , $k \leftarrow k + 1$, 并删去当前覆盖样本集中的样本; $i \leftarrow i + 1$, 转下一类. 若已是最后一类, 则从第一类 $i = 1$ 重新开始. 若所有类的样本点均已被覆盖, 则算法结束.

清华大学毕业设计论文

在算法中, 寻找一个未覆盖点是采用在样本数据集中顺序查找的方法, 找到第一个属于类 K_i 的未覆盖数据点就作为算法中所初始时的覆盖中心. 而算法中所提的投影算法如下:

算法 2 投影算法: 设 $P_k = \{c_0, c_1, \dots, c_k\}$ 为算法 1 中所定义, a 不属于由 P_k 所张成的空间 (仍记为 P_k), 求 a 到 P_k 的投影 a' .

S1 令 $d_i = c_i - c_0$, $g_i = (a - c_0)^T d_i$, $h_{ij} = (d_i)^T d_j$, 其中 $i, j = 1, \dots, k$.

S2 $H = (h_{ij})$ 为一 $k \times k$ 的矩阵, $g = (g_i)$ 为一 $k \times 1$ 的列向量. 由 c_i 的定义, 知 $\{d_i\}$ 是线性无关的¹, 故 H^{-1} 存在. 令 $(f_1, f_2, \dots, f_k)^t = H^{-1}g$, 则 $a' = c_0 + \sum_{i=1}^k f_i d_i$ 即为 a 在 P_k 上的投影.

其实, 上面的运算是可以通过先求归一化正交基再投影的方法来方便求解.

求得了一个覆盖 C_k 之后, 我们就可以为第一层相应地设置一个神经元, 取其权值为覆盖的中心, 而阈值为覆盖的半径, 即:

$$w^k = a_k, \quad \theta_k = d_k$$

这样, 如前所述, 若我们用 m 个覆盖盖住了所有的训练样本, 则我们所构造的前馈网的第一层也就有 m 个神经元.

逐类覆盖法所生成的圆域覆盖是有顺序的, 这个顺序也带来了覆盖“优先级”上的差别. 若某个模式样本落在第一个覆盖 C_1 中, 则其必被归类到 C_1 所对应的模式类, 而不管它是否还被其它圆域所盖住. 推广一下, 设盖住某个样本的覆盖中序号最小的一个是 C_k , 则该样本就被归类到 C_k 所对应的模式类中.

§4.3 网络第二层权值的设置

在前一章中, 我们已说明了在得到复杂覆盖并构造第一层网络之后, 可以用第二章中所说的前馈网将不规则小区域的划分信息映射到分类结果, 从而完成整个分类过程. 但逐类覆盖法所生成覆盖间“优先级”的差别, 又使得我们完全可以用一层网络 (共三层) 来实现划分信息到模式类别的映射.

¹简单地讲, 由于算法 1 中的 $c_{k+1} \in E_k$, 而当 $k > 0$ 时 β 是 α 到空间 P_k 的投影, 故集合 E_k 的条件 $(\alpha - \beta)^T(x - \beta) > 0$ 保证了所找到的 c_{k+1} 不在空间 P_k 中, 从而由数学归纳可知 P_{k+1} 内向量线性无关.

清华大学毕业设计论文

由于类别总可以用一位或多位二进制来表示, 我们在本节中就只讨论如何设置第二层中的单个神经元, 使其将第一层的输出按需要映射到集合 $\{-1, 1\}$ 上; 而多个神经元相配合, 即可完成从第一层输出到多位的指定映射. 虽然这一方法是针对逐类覆盖法设计的, 但只要生成的覆盖是有着优先顺序的, 就可以使用下面的权值设置方法.

设用逐类覆盖法顺序生成的圆域覆盖为 $C_i, i = 1, 2, \dots, m$, 其中 m 为覆盖数, 也是第一层中的神经元的数目. 当盖住当前输入样本的最小序号覆盖为 C_i 时, 该样本就被归类到与 C_i 相对应的模式类. 设此时网络第二层某个神经元 j 的输出被指定为 $y_j^i \in \{-1, 1\}$, 则该神经元权值 u^j 与阈值 ξ_j 的设置必须满足: (即 (2.7) 式)

$$y_j^i = \text{sgn} [(u^j)^T z^i - \xi_j] \quad (4.4)$$

其中 $z^i = (\underbrace{-1, \dots, -1}_{(i-1)\text{个}}, 1, z_{i+1}^i, \dots, z_m^i)^T$. 这时, 我们有以下引理:

引理 4.1 m 个圆域覆盖的参数已知, 且其优先级随下标的增加而降低, 则可如下设置前馈网第二层的权值 u^j 及 ξ_j , 以满足 (4.4).

$$u_m^j = y_j^m, u_i^j = \begin{cases} 1 - \sum_{\substack{t=i+1 \\ u_t^j < 0}}^m u_t^j, & \text{若 } y_j^i = +1; \\ -1 - \sum_{\substack{t=i+1 \\ u_t^j > 0}}^m u_t^j, & \text{若 } y_j^i = -1. \end{cases} \quad (4.5)$$

$$\xi_j = - \sum_{t=1}^m u_t^j \quad (4.6)$$

证明 (引理 4.1) 考虑将 z_k^i 改写为 $\frac{z_k^i + 1}{2}$, 则样本被覆盖 C_i 盖住时网络第一层的相应神经元 k 输出为 1, 反之为 0. 再设当前盖住样本的覆盖中序号最小的一个为 C_i , 我们希望下式

$$(u^j)^T \left(\frac{z^i + 1}{2} \right) = u_i^j + \sum_{t=i+1}^m u_t^j \cdot \frac{z_t^i + 1}{2} \quad (4.7)$$

与所需的输出 y_j^i 同号. 若所需输出为 +1, 则考虑序号大于 i 的所有权值 u_t^j 为负的

清华大学毕业设计论文

覆盖亦都盖住了输入样本这一最坏情况，我们要求

$$w_i^j > - \sum_{\substack{t=i+1 \\ u_t^j < 0}}^m u_t^j,$$

故可设 $u_i^j = 1 - \sum_{\substack{t=i+1 \\ u_t^j < 0}}^m u_t^j$. 而类似地，当所需输出为 $y_j^i = -1$ 时，我们要求

$$w_i^j < - \sum_{\substack{t=i+1 \\ u_t^j > 0}}^m u_t^j,$$

故可设 $u_i^j = -1 - \sum_{\substack{t=i+1 \\ u_t^j > 0}}^m u_t^j$. 通过这样的设置，我们可以保证 $(u^j)^T \left(\frac{z^i + 1}{2} \right)$ 总是与 y_j^i 同号.

最后，我们将 $\frac{z_i^j + 1}{2}$ 展开，此时 (4.7) 式变为

$$\sum_{t=1}^m u_t^j \cdot \frac{z_t^i + 1}{2} = \frac{1}{2} \left(\sum_{t=1}^m u_t^j z_t^i + \sum_{t=1}^m u_t^j \right)$$

$\frac{1}{2}$ 不影响结果的符号. 由上面的分析，易知依 (4.5) 设置权值 u^j 、依 (4.6) 式设置 ξ_j ，是能够使 (4.4) 成立的. \square

§4.4 平面双螺旋问题的分类结果

对于训练数据 1.1，用逐类覆盖法得到了 10 个覆盖，即在网络的第一层有 10 个神经元. 图 4.2(a) 给出了将三维空间中球面上的圆域覆盖投影到二维平面后的结果. 对于所给的训练数据 1.1，网络分类的识别率为 100%，这是在预料之中的. 而对于测试数据 1.1，网络分类的识别率为 98.245%.

虽然图 4.2(a) 是由空间中的圆域覆盖投影到二维平面上形成的，但从中依然可以容易地看出前述逐类覆盖法乃至一般的覆盖法的几何意义. 这里我们可以再将与第二章中所给的最初的前馈网做一个比较. 在最初的前馈网的学习中，我们仅仅是根据样本之间的距离，对每一个所给的样本构造一个合适大小的邻域覆盖，而

不管这一样本所在的数据类中其它样本的分布情况 (同样也没有考虑异类样本的分布). 但从实际的样本分布来看, 同类的样本往往聚集成簇, 否则, 从常理上来说, 也与“同类”相矛盾了. 我们所讨论的神经元的几何意义, 使我们明白分类其实在某种程度上与覆盖是等价的. 那么, 既然同类的点倾向于聚集在一起, 我们为什么还要用小而多的覆盖, 而不是用几个较大的覆盖去盖住它们呢? 在这种思路指导下, 结合清晰的几何描述, 我们就可以用球面上的圆域覆盖来完成样本点的分类. 而且, 可以想象, 这种用几个较大的覆盖来代替多个小覆盖的作法, 不仅仅引起了覆盖数 (相应地, 网络所用神经元的数目) 的减少, 而且它还从整体上把握了样本的分布, 增加了网络的泛化能力.

观察图 4.2(a), 我们还可以看出, 虽然 10 个神经元所对应的圆域已将所有的训练样本覆盖住, 但还有部分曲线是被不正确的圆域所覆盖的. 这显然和样本集中的样本数目较少有关. 通过增多训练样本的数目, 就可能更好地反映曲线 K_1 与 K_2 的特征. 我们在实验中又使用了另一训练数据集:

训练数据 1.2 曲线 K_1 及 K_2 的方程不变, 但将范围扩大到 $0.45\pi < \rho < 6.05\pi$, 在这

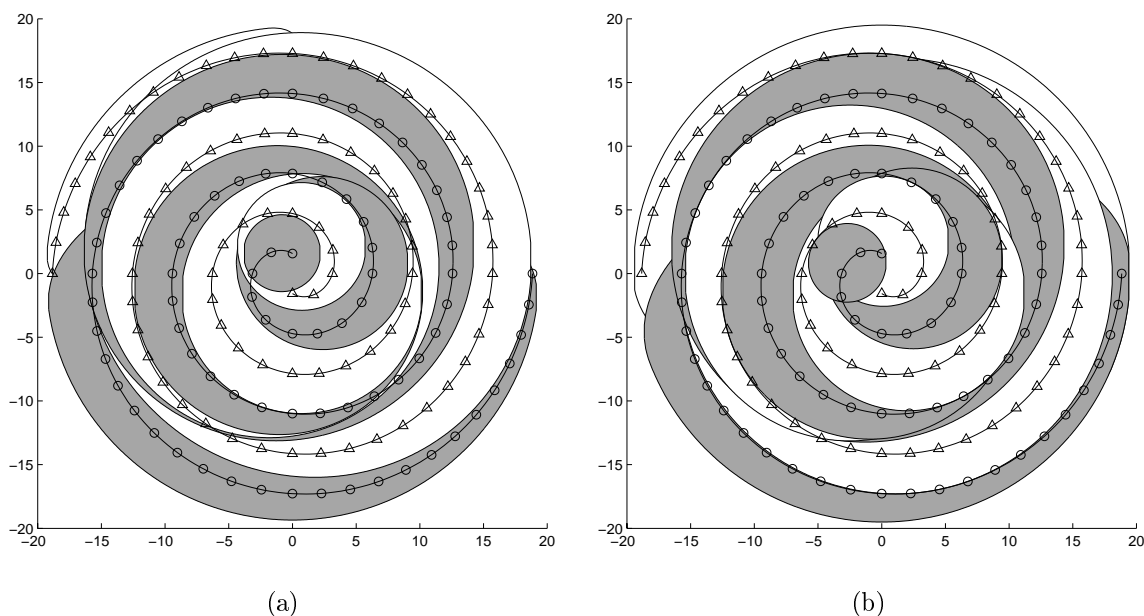


图 4.2: 采用逐类覆盖法对平面双螺旋分类问题的求解结果. (a) 训练数据为双螺旋上 156 个样本点; (b) 训练数据为 20000 个数据点. 图中灰色表示网络所划分的第一类样本的覆盖区域, 白色表示第二类样本的覆盖区域, \circ 及 Δ 表示两类的训练样本点.

清华大学毕业设计论文

一范围中随机各取 10000 个点, 经对 ρ 从小到大排序后作为学习样本.

这是一个有着很多样本的数据集. 我们不仅可以用它来测试网络的学习速度, 而且也可以将由训练数据 1.2 所得的覆盖结果和前面用训练数据 1.1 的覆盖结果做比较, 来分析覆盖算法. 这一次, 我们同样也得到了 10 个覆盖, 如图 4.2(b). 但由于这次的学习样本几乎覆盖了曲线上所有部分, 因此对于测试数据 1.1, 其识别准确率大大提高, 为 99.995%, 而所用的学习时间仍很短, 在 486/66 上只用了 30 多秒.

需要说明的一点是, 由于逐类覆盖法在步骤 S1 时是将顺序查找所得的第一个属于类 K_i 的未覆盖点作为初始的覆盖中心, 因此学习数据的排列顺序对于算法的结果是有很影响的. 我们在使用大数据训练集进行分类实验时, 最初使用的是曲线上随机产生的 20000 个点, 但即没有排序. 由于样本间的无序, 使得覆盖算法所得到的覆盖顺序也比较无序, 因此所得的识别效果并不太好, 用的覆盖也较多 (在某次实验中为 26 个覆盖). 在认真考虑了双螺旋的特点之后, 我们决定将 20000 个随机样本按 ρ 做升序排序. 这时的覆盖数与学习时间已大大减少. 但在将训练数据 1.1 作为测试数据测试网络分类效果时, 总发现有 1 个数据被误分类. 对于有着 20000 个随机分布的数据的学习样本, 这样的错误不应该发生. 仔细的分析表明误分类总是发生在 $\rho = 6\pi$ 的点上. 这是因为范围为 $\frac{\pi}{2} \leq \rho \leq 6\pi$ 内的随机数往往无法包括 $\rho = 6\pi$ 的点在内. 这就造成了分类覆盖上的偏差. 在将训练样本的范围稍许扩大到 $0.45\pi < \rho < 6.05\pi$ 后, 就不再发生类似的误分类了.

附录中的表 A.1 给出了实验的一些结果, 而覆盖以及网络第二层的具体参数可见附录中表 A.3 和表 A.4.

§4.5 小结

通过这个并不是很简单的例子, 我们大致了解了用覆盖算法及前馈网络进行分类的全过程, 同时, 我们也可以暂时得出以下结论:

1. 对于训练数据, 算法保证了其识别率必为 100%.
2. 对于网络的训练, 我们将其转变成了数据样本点的覆盖算法. 从实际的一种算法的运行结果来看, 所需的时间很短, 而网络构造所用的神经元的数目却非常少. 一次 20000 个数据点的学习只使用了 32 秒多的时间, 而只用了 10 个神经元.

3. 网络所用的神经元个数基本与训练数据的数目无关，只要这些训练数据反映了待分类数据的特征。在所给的例子中，无论训练数据是一两百还是几万，所得的覆盖数都是 10。一般而言，这些数目仅与待分类数据的分布特征、分类算法以及所选的空间变换方法 T 有关。

第五章 其它分类例子

在本章中, 我们利用自己编制的程序, 对另外三个例子进行了分析, 进一步展示了覆盖算法以及前馈网的特点. 部分例子由于所用的数据维数较高, 我们无法给象平面双螺旋分类中那样给出直观的覆盖图, 因此这里以算法的识别率作为主要的评价依据.

§5.1 文件格式及测试程序

在分析了前馈网及覆盖的基本要求之后, 我首先设计了程序所需使用的数据结构与文件结构. 数据结构用于程序内部, 主要描述了覆盖 (簇)、类 (含多个簇) 以及样本集的概念, 同时还模拟多层神经网络的结构, 设计了描述神经元及层的数据结构.

实验中需要使用大量的不同的数据集. 这就需要从外部以文件的方式向分类及测试程序提供数据. 而在得到了圆域覆盖之后, 我们也需要将覆盖结果 (包括网络第二层的权值) 永久地保存在外部文件中. 因此, 我设计了两种用于存放样本数据的文件格式, 以及一种用于存放覆盖结果的文件格式. 由于文件结构涉及到程序的使用, 因此在附录 B.1 中列出了这三种文件的具体格式.

在整个实现过程中, 一共完成了两套程序.

第一套程序的用户界面使用了命令行方式. 我们提供了一系列的子功能, 用于将样本数据、覆盖结果等从文件中读取或写入文件, 同时我们也提供了进行空间变换、覆盖生成以及由覆盖数据得到网络第二层的权值及阈值的子程序. 为了方便地将网络训练所得的结果转送给别的应用程序, 我还编制了一些接口, 实现了诸如将三维覆盖结果转化成 (经映射后) 二维覆盖的 Matlab 显示子程序、将覆盖结果及第二层网络权值信息转化成 L^AT_EX 表格文件等功能. 这些子程序和接口可以自由地在用户的程序中使用, 非常方便. 本文中的所有例子, 均由此套程序测试.

第二套程序是基于 Windows 的分类、测试程序. 该程序主要的目的是提供一个用户界面友好, 且较为通用的分类程序, 以便应用研究人员将前馈网络试用于不同的领域. 这一程序采用了 Wizard 的方式, 将分类、测试分为三步, 一步步地根据用

清华大学毕业设计论文

户的需要进行计算、给出结果. 这一程序还远未完善, 但其基本框架已完成 (参见附录 B.3).

§5.2 平面三螺旋问题

作为对平面双螺旋较为近似的一个扩展, 同时也为了检验网络对于两类以上数据的分类能力, 我们定义平面三螺旋问题如下:

分类问题 2 在平面极坐标系中, 我们定义如下三条曲线 K_1 , K_2 与 K_3 :

$$K_1: \rho = \theta, \quad K_2: \rho = \theta + \frac{2\pi}{3}, \quad K_3: \rho = \theta + \frac{4\pi}{3}.$$

其图可见 5.1, 这是三条在平面上交错的螺旋线, 其中 $\frac{\pi}{2} \leq \rho \leq 6\pi$. 这一问题要求根据输入的平面坐标值将这三条曲线区分开来.

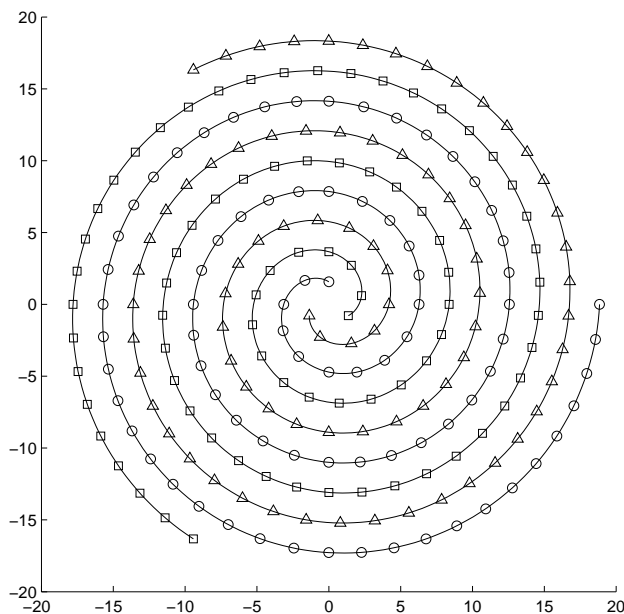


图 5.1: 平面三螺旋分类问题. 图中曲线为三螺旋线, \circ 、 Δ 及 \square 分别表示三类训练样本点.

类似地, 我们也设计了如下的训练数据与测试数据:

训练数据 2.1 在 K_1, K_2, K_3 三条曲线上各取 78 个样本点, 即取

$$\rho = \left(i + \frac{j}{i+1}\right) \cdot \frac{\pi}{2}, \text{ 及 } \rho = 6\pi. \quad (5.1)$$

共 234 个点, 其中 $i = 1, 2, \dots, 11$; $j = 0, 1, \dots, i$.

训练数据 2.2 曲线 K_1 , K_2 及 K_3 的方程不变, 但将范围扩大到 $0.45\pi < \rho < 6.05\pi$, 在这一范围中随机各取 10000 个点, 经对 ρ 从小到大排序后作为学习样本 (共 30000 个点).

测试数据 2.1 曲线 K_1 , K_2 及 K_3 上随机各取 10,000 个点, 共 30,000 个点作为测试样本.

在使用 (3.1) 式时, 我们取变换区域的半径 $d = 19.5056$ (这个值是平面上数据点的最小半径加 0.5). 网络的输入, 就采用变换后点在球面上的笛卡尔空间坐标值.

对于训练数据 2.1, 用逐类覆盖法得到了 24 个覆盖, 覆盖的投影图可见图 5.2(a). 相应地对于训练数据 2.2, 其 26 个覆盖的图见图 5.2(b). 对于测试数据 2.1, 两种情况

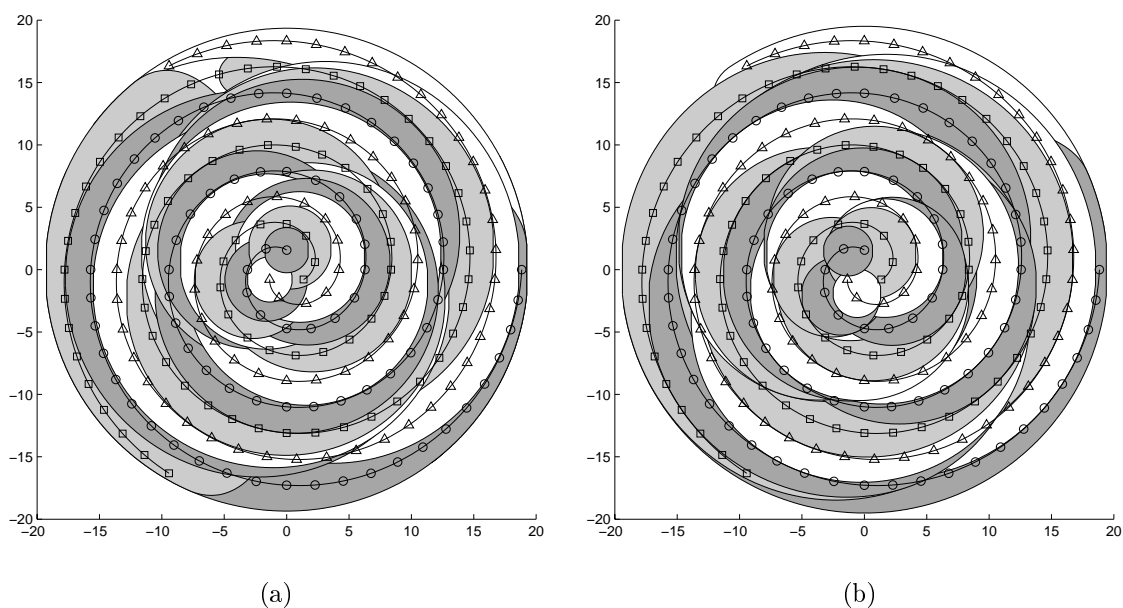


图 5.2: 采用逐类覆盖法对平面三螺旋分类问题的求解结果. (a) 训练数据为三螺旋上 234 个样本点; (b) 训练数据为 30000 个数据点. 图中黑、白、灰分别表示网络所划分的第一类、第二类、第三类样本的覆盖区域, \circ 、 \triangle 及 \square 分别表示三类训练样本点.

下训练的网络所达的识别率为 91.11% 和 99.067%. 相比起平面双螺旋问题来说, 识别率有所下降, 但依然很高.

仔细观察图 5.2 与平面双螺旋分类时的图 4.2, 我们会发现两个图中的覆盖都存在同样的问题: 即部分样本与覆盖的边缘距离太近了. 这不仅使分类网络在那些样

本附近非常敏感，而且容易因为计算精度不高而引起误分类。我们能够回忆起在覆盖算法的步骤 S4 中 (页 19) 中，覆盖的“半径”由 (4.2) 式定为类内点到中心的最大距离与类外点到中心的最小距离的算术平均。这其实是希望用一种“中庸”的办法，在覆盖住尽可能多样本的同时，保证一定的分类余地。事实上就是希望能保证网络有较好的泛化能力。但这两个矛盾的目标，即覆盖住尽可能多的样本与保证一定的分类余地，在我们所给的逐类覆盖法中协调得并不是很好。从算法的角度来说，这一目标已达到，因为从图上也可看出，确实有这么两个样本，分别在覆盖内与外，而且离边界都很近；但从实际来说，我们希望覆盖的边界与样本点间有一个阈值。

在将阈值的思想加入到逐类覆盖法中后 (我们将覆盖外与覆盖内的点到覆盖中心的点积阈值均取为 0.1)，算法的识别率有了较大的提高。对于平面二螺旋问题来说，无论使用训练数据 1.1 还是 1.2，所得的覆盖都增加到了 13 个，而识别率分别上升到 98.355% 与 100%。而对于平面三螺旋来说，则所用的覆盖数都没有改变，而使用训练数据 2.1 与 2.2 所得的识别率却上升到了 91.41% 与 99.987%。可见阈值主要在样本分布比较密集时起作用。而对于训练数据 1.1 或 2.1，由于其本身的样本数就不足以完全反映模式类的分布，因此在算法中加入阈值所得到的识别率改善也是有限的。对于两个大样本集的覆盖结果如图 5.3。

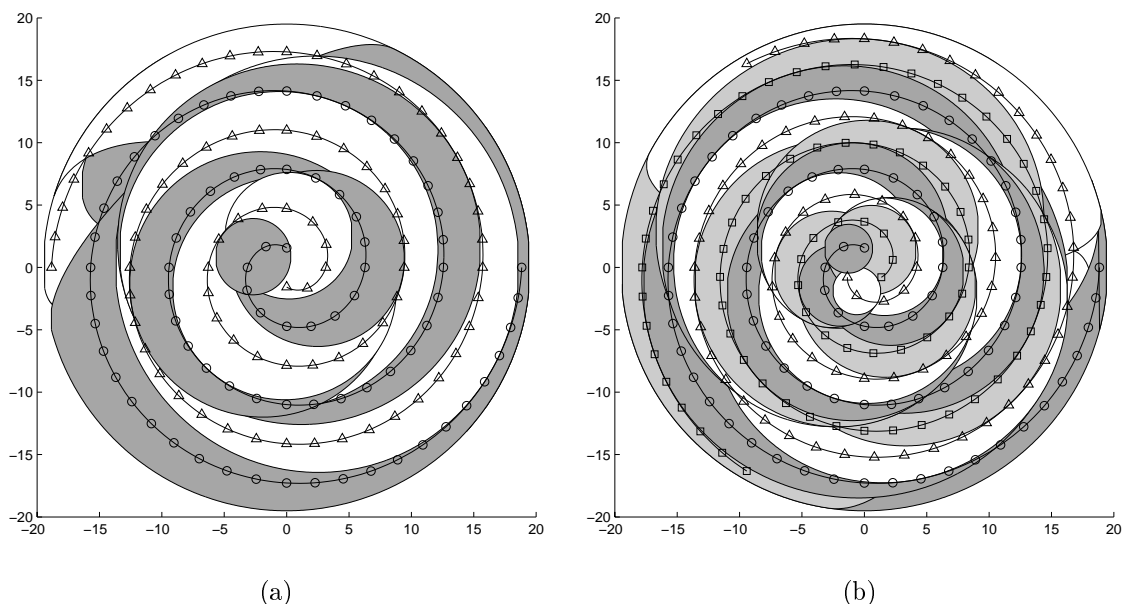


图 5.3: 加入边界阈值控制后逐类覆盖法对平面螺旋问题的求解结果。(a) 训练数据为二螺旋上 20000 个样本点 (1.2); (b) 训练数据为三螺旋上 30000 个数据点 (2.2)。

清华大学毕业设计论文

§5.3 空间三螺旋问题

在用前馈网络尝试进行了二维平面上两类乃至多类问题的分类求解之后，我们再一次，以类似的方式，来检验这种方法对于更高维问题的求解能力。这里我们定义三维的三螺旋问题如下：

分类问题 3 在空间柱坐标系中，我们定义如下三条曲线 K_1, K_2 与 K_3 ：

$$K_1: z = \theta, \rho = 1;$$

$$K_2: z = \theta + \frac{2\pi}{3}, \rho = 1;$$

$$K_3: z = \theta + \frac{4\pi}{3}, \rho = 1.$$

其图可见 5.4，这是三条在空间中交错盘旋的螺旋线，其中 $0 \leq z \leq 6\pi$ 。这一问题要求根据输入的空间笛卡尔坐标值将这三条曲线区分开来。

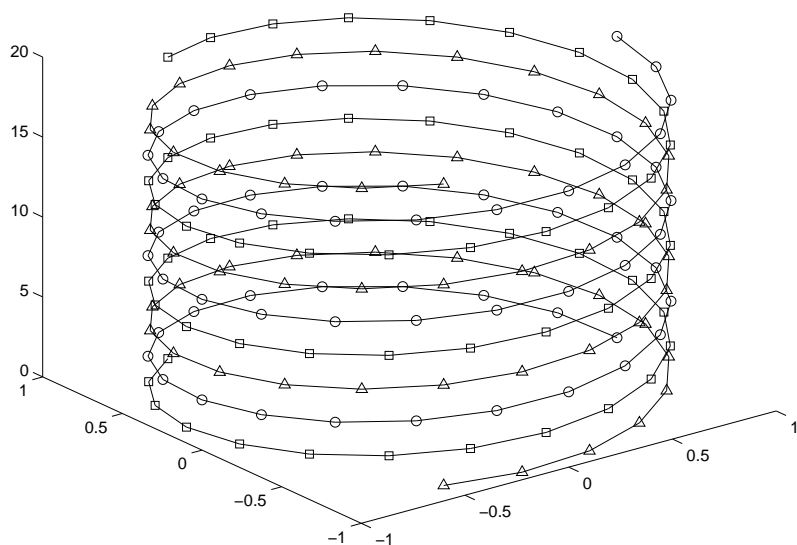


图 5.4: 空间三螺旋分类问题。图中曲线为三螺旋线， \circ 、 Δ 及 \square 分别表示三类训练样本点。

我们使用了如下的训练数据与测试数据：

训练数据 3.1 在 K_1, K_2, K_3 三条曲线上各取 61 个样本点，即取

$$z = \frac{i\pi}{10}, i = 0, 1, \dots, 60.$$

清华大学毕业设计论文

训练数据 3.2 曲线 K_1 , K_2 及 K_3 的方程不变, 但将范围扩大到 $-0.1\pi < z < 6.1\pi$, 在这一范围中随机各取 10000 个点, 经对 z 从小到大排序后作为学习样本 (共 30000 个点).

测试数据 3.1 曲线 K_1 , K_2 及 K_3 上随机各取 10,000 个点作为测试样本.

对于训练数据 3.1, 用逐类覆盖法得到了 28 个覆盖, 而对于训练数据 3.2 则得到了 37 个覆盖. 对于测试数据 3.1, 两种情况下训练的网络所达的识别率为 96.293% 和 99.653%. 而采用加上阈值后的逐类覆盖法, 则覆盖数分别为 31 与 42, 而识别率为 96.56% 与 100%.

§5.4 胸癌数据 (Wisconsin Breast Cancer Database)

从 [9] 处我们得到了文献 [10] 所使用的胸癌数据. 其中的数据共分为两类, 即良性的 (benign) 和恶性的 (malignant). 除数据的编号和类型外, 每个样本含有 9 个特征, 有部分数据的特性是不全的. 数据库在不断地修正和扩充之中, 现在的数据库含有 699 个样本点, 其中有 16 个数据是特征不全的.

在参考 [9] 中文件 `breast-cancer-wisconsin.names` 后, 我恢复了最先在文献 [10, 11] 使用的 369 个数据点的胸癌数据库. 剔去其中含有未知数据的 14 个样本, 我们得到了 355 个样本, 为了方便起见, 下面我们将这 355 个样本形成的数据称为胸癌数据集 1. 由于没有得到 [10, 11] 的原文, 仅知道其使用的分类数据即为前述的 369 个胸癌数据, 我们可以假定这一分类问题为:

分类问题 4 利用从 [9] 得到的数据的一部分作为训练样本, 构造分类器, 以便将良性的数据和恶性的数据区分开.

同时, 依据文献 [10, 11] 中的方法, 我们将胸癌数据集 1 的前一半以及前 2/3 作为训练数据, 即建立以下两个训练数据集:

训练数据 4.1 将胸癌数据集 1 的前 177 个数据作为训练数据.

训练数据 4.2 将胸癌数据集 1 的前 236 个数据作为训练数据.

并以整个胸癌数据集 1 作为测试集.

对于训练数据 4.1, [10] 使用了两对平行的超平面将之分开, 并且这两对超平面对胸癌数据集 1 中剩余的一半的识别率为 93.5%. 而用我们前面所述的前馈网络及

清华大学毕业设计论文

逐类覆盖方法, 我们使用 5 个覆盖, 对于整个乳腺癌数据集 1 的识别率为 97.183%, 即对剩余一半的识别率为 94.382%. 这要高于 [10] 中的结果. 在使用训练数据 4.2 时, 生成了 7 个覆盖, 但识别率并没有显著提高, 只为 98.028%, 对于剩余 1/3 数据的识别率只有 94.118%, 比 [10] 中的 95.9% 要低. 不过考虑到对于训练数据的组成上可能有些差别, 因此这些优劣的比较只能作为参考.

进一步分析乳腺癌数据可发现, 两类乳腺癌数据的分布并不是大致相同的, 相对而言良性数据分布较为集中. 对覆盖数据的分析也表明, 逐类覆盖法只用一个覆盖就完成了对大部分良性数据的分类, 而之后对良性数据的覆盖就只是“拾遗补漏”了. 这说明了逐类覆盖法存在一个较大的缺陷, 它只会机械地“逐类”进行覆盖, 而无法判断究竟哪一类需要较多的覆盖, 哪一类不需要. 这也和该算法缺乏全局知识有关.

第六章 初步结论及进一步的工作

本文是近半年来对于张铃、张钹所提出的前馈网络以及覆盖算法进行学习、研究的一个初步的小结，同时也是下一步工作开始前的准备。

经过理论上的初步研究，以及对于一些典型问题的求解，我们可以看到，前馈网络(加上相应的覆盖算法)具有如下一些特点：

1. 网络的构造过程总是能够保证对于训练数据的识别率是 100%。对于许多其它算法来说，这是一个十分难得的优点。
2. 神经网络的训练被转变成了数据样本点的覆盖过程。由于对训练样本的识别率已能够保证，对于覆盖算法的考虑就应该主要从效率以及泛化能力这两方面出发。如果覆盖能够尽可能实际地反映样本在空间中的分布，就能得到较好的识别率。因此，可以利用前面所述的网络构造和覆盖算法的相对无关性，针对具体问题设计相应的含有先验知识的覆盖算法，就可能得到性能优良的分类器。
3. 现有算法(包括覆盖算法在内)的速度很快。从已有的几次实验可看出，即使在 486/66 上，几万个数据点的学习也没有超过 2 分钟。而且基本不含任何先验知识的逐类覆盖法(它利用了样本点间的顺序)对大部分问题所得到的覆盖数，即第一层的神经元数，是很少的。
4. 在覆盖能合理反映样本点空间分布的理想状况下，网络所用的神经元个数应基本与训练数据的数目无关。在我们所做的四个例子中，基本反映了这一规律，但也有因为使用的数据集不同而生成的覆盖数相差很多的。

总的来说，将分类问题转化为覆盖问题，同时针对不同领域的分类问题相应地设计特殊的覆盖算法，这对于研究分类问题是一个很好的思想。下一步的研究，应该从以下几方面来进一步探索或是完善这一思想：

1. 针对一些特定的分类领域，开发出更多的覆盖算法；
2. 在数学上进一步研究可能的空间变换映射。在有些分类问题中，不同特征的范围有可能相差很大。这时再使用 (3.1) 式的变换公式，显然是不合适的。因此，研究空间变换对于样本点分布的影响、进而研究空间变换对于后续覆盖分类的影响，是很必要的。

清华大学毕业设计论文

3. 现有的覆盖方法，主要考虑了刚性的边界；但对于许多本身带有模糊不确定性的分类问题，刚性的边界并不是很适合。因此，如何在现有的前馈网络及覆盖框架中引入模糊逻辑的概念，可以做进一步的探索。
4. 在将分类问题延展到语音识别等需要时序知识的领域时，现有的网络结构由于不具备递归或是反馈的功能而显得不是很适合。如何在保持网络现有优点的同时，使其适合时序识别的要求，也值得研究。
5. 如何处理部分特征缺失的数据是分类问题中的一个老问题，但也是难问题。一般的作法是将之丢弃或是简单地用某种方法对缺失的特征赋值。这一问题如何在覆盖的背景下解决，还没有任何人研究过。

总之，在新的思路和背景下，还有许许多多的工作要做。

致 谢

在论文的结尾，我想感谢我的导师张钹老师和邓志东老师，他们指导我接触了神经网络。虽然这一领域对我来说是全新的，但他们耐心地引导我了解了神经网络的两大应用领域，即分类和控制，并为我制定了合适的学习和研究进度。在近两年的学习中，两位老师不厌其烦地教会了我做研究和撰写论文的基本要领，使我能够顺利地完成毕业设计以及论文的写作。同时，张老师和邓老师严谨的治学与平易的为人，也让我受益非浅。

还要感谢吴鸣锐、张朝晖、晏雄伟、朱仲涛、覃振兴等师兄，他们平时在学术、生活上都给了我很大帮助。我与他们的讨论，往往给我以启发；而对于我的一些问题或是求助，他们也总是能热心相助。我还要感谢同组的葛子辉和张奕同学。在写论文期间，他们的鼓励是我不断改进我的论文的动力之一。

最后，我想感谢我的父母。是父母的希望，使我不敢懈怠；是父母的勉励，令我始终都有着勇气与动力。我只希望，四年来本科优异的成绩能让父母高兴。

清华大学毕业设计论文

参考文献

- [1] 傅京孙主编, 程民德等译, 模式识别应用, 北京大学出版社, 1990.2 第一版.
- [2] Fu, K. S., Sequential Methods in Pattern Recognition and Machine Learning, Academic Press, New York, 1968.
- [3] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks* v 4, pp. 251–257, 1991.
- [4] 阎平凡、黄端旭, 人工神经网络 — 模型·分析与应用, 安徽教育出版社, 1993.5 第一版.
- [5] 张铃、张钊等, "多层前馈神经网络的学习和综合算法", 软件学报, v 6 n 7, pp. 440–448, 1995.
- [6] Rujan, P. and Marchand, M., "A geometric approach to learning in neural networks," Proc. of IJCNN'89, vol. II, pp. 105–110.
- [7] Ramacher, U. and Wesseling M., "A geometric approach to neural network design," Proc. of IJCNN'89, vol. II, pp. 147–154.
- [8] Zhang, L. and Zhang, B., "A Geometrical Representation of M-P Neural Model and Its Applications," posted.
- [9] Murphy, P. M. and Aha, D. W., *UCI Repository of machine learning databases* [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science, 1994.
- [10] Wolberg, W. H. and Mangasarian, O. L., "Multisurface method of pattern separation for medical diagnosis applied to breast cytology," Proc. of the National Academy of Sciences, U.S.A., v 87, pp. 9193–9196, Dec. 1990.
- [11] Zhang, J., "Selecting typical instances in instance-based learning," In Proceedings of the Ninth International Machine Learning Conference, pp. 470–479. Aberdeen, Scotland: Morgan Kaufmann. 1992.

清华大学毕业设计论文

附录A 分类结果

§A.1 文中所有分类问题的基本数据

为了便于比较, 这里所有的分类结果均是在一台 486/66 上、Windows 95 的 DOS 环境里运行第一套程序得到的. 时间的度量, 以那台机器上的系统时间为准, 但由于 Windows 下系统自身的进程调度需要时间, 故这里列出的时间只是一个大致的参考, 但至少是一个上界. 表 A.1 和 A.2 分别给出了文中的分类问题用逐类覆盖法和加入阈值控制后的逐类覆盖法进行覆盖分类的结果. 虽然针对具体问题相应地调整阈值会更有利于分类, 我们将这里的阈值控制始终设为 0.1.

表 A.1: 用逐类覆盖法构造网络所得的分类结果

问题	训练样本 (样本数)	覆盖半径 (d)	学习时间 (486/66)	覆盖数	测试样本	识别率 (%)
平面双螺旋	1.1 (156)	19.3496	0.05s	10	1.1	98.245
	1.2 (20000)	19.5056	32.95s	10	1.1	99.995
平面三螺旋	2.1 (234)	19.3496	0.22s	24	2.1	91.11
	2.2 (30000)	19.5066	90.46s	26	2.1	99.067
空间三螺旋	3.1 (183)	19.3761	0.22s	28	3.1	96.293
	3.2 (30000)	19.6874	81.56s	37	3.1	99.653
胸癌数据	4.1 (177)	30.0	0.06s	5	数据集 1	97.183
	4.2 (236)	30.0	0.11s	7	数据集 1	98.028

表 A.2: 加入阈值控制后逐类覆盖法所得的分类结果

问题	训练样本 (样本数)	覆盖半径 (d)	学习时间 (486/66)	覆盖数	测试样本	识别率 (%)
平面双螺旋	1.1 (156)	19.3496	0.06s	13	1.1	98.355
	1.2 (20000)	19.5056	26.70s	13	1.1	100.0
平面三螺旋	2.1 (234)	19.3496	0.27s	24	2.1	91.41
	2.2 (30000)	19.5066	89.47s	26	2.1	99.987
空间三螺旋	3.1 (183)	19.3761	0.17s	31	3.1	96.56
	3.2 (30000)	19.6874	104.41s	42	3.1	100.0

清华大学毕业设计论文

§A.2 平面双螺旋的分类结果

在附录中，我们只列出平面双螺旋问题用逐类覆盖法及加入阈值控制的逐类覆盖法所得到的覆盖的参数，以及网络第二层的权、阈值。由于另外几个分类问题所得的覆盖数较多，因此那些问题的覆盖参数就不再列出。

采用训练数据 1.1 时的结果

采用训练数据 1.1, 取半径 $d = 19.3496$, 用逐类覆盖法在 0.05 秒内我们得到了 10 个覆盖。覆盖数据（即第一层的参数）以及在第二层的输入权值如下：

表 A.3: 逐类覆盖法对训练数据 1.1 的覆盖结果及网络第二层参数

类别	向量坐标			阈值	第二层权重
Line1	-0.833817	1.61995	19.2636	369.786	-89
Line2	0.741275	2.20568	19.2092	361.454	55
Line1	2.49036	0.87857	19.1685	350.546	-34
Line2	0.854883	-1.79928	19.2468	326.607	21
Line1	-1.37766	-1.60496	19.2336	302.81	-13
Line2	-2.41764	0.663283	19.1865	270.652	8
Line1	-1.72617	3.3631	18.9767	226.739	-5
Line2	3.08558	3.70253	18.7397	147.614	3
Line1	0.216973	-16.0665	10.781	47.1161	-2
Line2	-12.7733	12.7733	6.93489	270.979	1
第二层阈值					55

采用训练数据 1.2 时的结果

采用训练数据 1.2, 取半径 $d = 19.5056$, 逐类覆盖法在 35.49 秒内我们得到了 10 个覆盖。覆盖数据（即第一层的参数）以及在第二层的输入权值如表 A.4. 而使用加入阈值控制的逐类覆盖法，则在 26.70 秒内，得到了 13 个覆盖，其数据如下页表 A.5.

表 A.4: 逐类覆盖法对训练数据 1.2 的覆盖结果及网络第二层参数

类别	向量坐标			阈值	第二层权重
Line1	-2.73165	0.85193	19.2946	375.662	-89
Line2	0.144662	2.63364	19.3265	366.931	55
Line1	2.60267	1.27594	19.289	354.318	-34
Line2	2.40376	-1.82811	19.2704	335.504	21
Line1	-1.2062	-1.81774	19.3832	306.084	-13
Line2	-3.38489	0.060542	19.2096	280.483	8
Line1	-1.78212	3.34594	19.1337	231.197	-5
Line2	2.35404	-0.400784	19.3589	182.918	3
Line1	3.1382	-15.456	11.4775	50.8325	-2
Line2	3.11978	16.8082	9.39256	-67.0917	1
第二层阈值					55

表 A.5: 阈值控制的逐类覆盖法对训练数据 1.2 的覆盖结果

类别	向量坐标			阈值	第二层权重
Line1	-2.7146	0.871225	19.2961	375.752	-322
Line2	0.193643	2.63549	19.3258	367.134	199
Line1	2.6272	0.767916	19.3126	354.685	-123
Line2	-0.184677	-1.88541	19.4134	337.805	76
Line1	-2.37029	-1.62647	19.2926	318.441	-47
Line2	-0.843264	1.80346	19.4037	287.278	29
Line1	1.50755	2.80277	19.2442	252.268	-18
Line2	4.71658	0.494249	18.9203	196.874	11
Line1	11.6483	-14.1453	6.6855	-147.487	-7
Line2	8.10603	14.4214	-10.3337	-73.7646	4
Line1	-11.0828	10.0644	12.5039	350.449	-3
Line2	-16.1951	8.6473	6.58876	307.337	1
Line2	-18.7432	1.23392	5.25725	371.226	1
第二层阈值					199

附录B 文件格式与程序

本附录列出了我们所编制的分类及测试程序使用的数据文件及覆盖结果文件的格式，同时也给出了两套程序典型运行过程的图示。

§B.1 文件格式

所有的文件均为文本文件，基本数据由换行符隔开。文件中可以有注释，也可以有空行；注释以分号‘;’开始，一直延续到本行末；注释可以在一行的中间开始。文件格式中注明在一行中的数据，可以隔行输入，只要中间只有空行或是注释；但已注明不在同一行中的数据，不能放在一行中，否则部分数据将被读取文件的子程序忽略。

数据格式一(表 B.1) 将所有类别的数据混杂排放，对于事先并未做任何分检工作的原始数据，用这一格式来形成我们所需的数据文件是很方便的。同时，它也利于数据的添加，尤其是自动添加。

数据格式二(表 B.2) 将数据按类存放，同一类的数据有一个相对固定的位置，便于数据的查找和修改。同时，也便于研究者观察同一类中数据的情况。

覆盖数据文件说明了所有的覆盖的总体信息，及所有覆盖的中心及阈值。其不仅用于分类程序将覆盖结果输出到文件，而且测试程序还可依据这一格式将覆盖文件中已有的覆盖(分类)信息读入。其格式如表 B.3。

§B.2 第一套程序的结果

第一套程序使用的是字符界面，图 B.1 列出了一次运行(省略与用户的交互过程)的程序输出。

这是加入阈值控制后的逐类覆盖法对训练数据 2.2 (文件为‘train13s.dat’)进行覆盖的运行输出。在读入文件之后，程序计算了进行空间变换所需的最小半径，并将其加上 0.5 作为当前空间变换的半径，即 19.5066。之后，程序用了 89.47 秒得到了 26 个覆盖，并将覆盖信息与自动生成的 Matlab 绘图(画出平面覆盖)命令分

清华大学毕业设计论文

表 B.1: 数据文件格式一

行序	内 容	说 明
1	Type_1	固定字符串, 表明文件类型.
2	k_p	数据的类别数 k 及数据个数 p .
3	n	数据的原始维数 $n(> 0)$.
4	类名 1_..._类名 k	用空格分开的所有类的名字.
5 ~ ($p + 4$)	特征 1_... _特征 n _类名	p 行数据, 每行包括 n 个以空格为间隔的实数, 作为数据的坐标或是特征值; 行末为该数据所属类的类名.

表 B.2: 数据文件格式二

行序	内 容	说 明
1	Type_2	固定字符串, 表明文件类型.
2	k	数据的类别数 k .
3	n	数据的原始维数 $n(> 0)$
4	类名 1_ p_1	第一类的名字, 及该类中数据个数 p_1 .
5 ~ ($p_1 + 4$)	特征 1_..._特征 n	p_1 行数据, 每行包括 n 个以空格为间隔的实数, 作为数据的坐标或是特征值.
⋮	类似行 4 ~ ($p_1 + 4$)	依次为其它类的名字、个数及具体数据构成的小块数据.

表 B.3: 覆盖数据文件格式

行序	内 容	说 明
1	k	数据的类别数 k .
2	类名 1_..._类名 k	用空格分开的所有类的名字.
3	n_d	数据的原始维数 $n(> 0)$ 及空间变换操作所使用的半径 d .
4	m	覆盖数, 也即分簇数 m .
5 ~ ($m + 4$)	特征 0_特征 1_... _特征 n _阈值_类名	m 行数据, 每行包括 $n + 1$ 个以空格为间隔的实数, 作为覆盖在球面 S^n 上的中心; 之后为覆盖的阈值; 而行末为该数据所属类的类名. 其中特征 0 为由空间变换所产生的那一维数据. 覆盖间有优先级的差别, 级别高的行序较小.

```
Reading data file 'train13s.dat' ... ok
Setting radius to 19.5066 ...
Making clusters ... Using 89470 milliseconds. 26 cluster(s) covered.
Writing cluster file 'clus13st.cov' ... ok
Writing Matlab file 'clus13st.m' ... ok
Reading cluster file 'clus13st.cov' ... ok
Layer name: Layer 3, neurons: 2
  neuron 0:
    weight: -70226 51409 -18817 -18817 13775 -5042 -5042 3691 -1351
-1351 989 -362 -362 265 -97 -97 71 -26 -26 19 -7 -7 5 -2 -2 1
    threshold: 51409
  neuron 1:
    weight: -40545 -40545 29681 -10864 -10864 7953 -2911 -2911 2131
-780 -780 571 -209 -209 153 -56 -56 41 -15 -15 11 -4 -4 3 -1 -1
    threshold: 70226
Reading data file 'test13.dat' ... ok
Setting radius to 19.5066 ...
Testing ...
class 1: correct 9998, wrong 2, reject 0
class 2: correct 9998, wrong 2, reject 0
class 3: correct 10000, wrong 0, reject 0
class 4: correct 0, wrong 0, reject 0
Writing TeX file 'clus13st.ctx' ... ok
```

图 B.1: 第一套程序的一次运行输出

别写入了文件 'clus13st.cov' 与 'clus13st.m'。接下来，程序根据覆盖信息和类别信息生成了网络第二层神经元的权值和阈值。最后，程序用测试数据 2.1 (文件为 'test13.dat') 测试了网络的分类效果，并将分类的结果以 L^AT_EX 格式写入了文件 'clus13st.ctx'。

§B.3 第二套程序的结果

下一页的图 B.2 与图 B.3 分别给出了第二套程序在用训练数据 3.1 进行分类构造，并对 3.1 进行测试的结果。这是在 Pentium-II/233 上用逐类覆盖算法得到的。

清华大学毕业设计论文

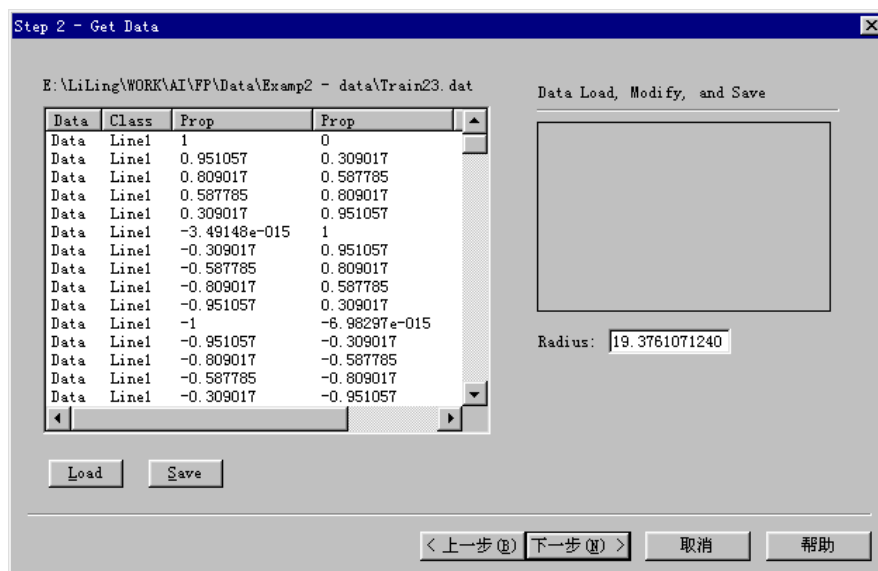


图 B.2: 第二套程序在读入模式数据进行覆盖时的界面，所用的训练数据为 3.1

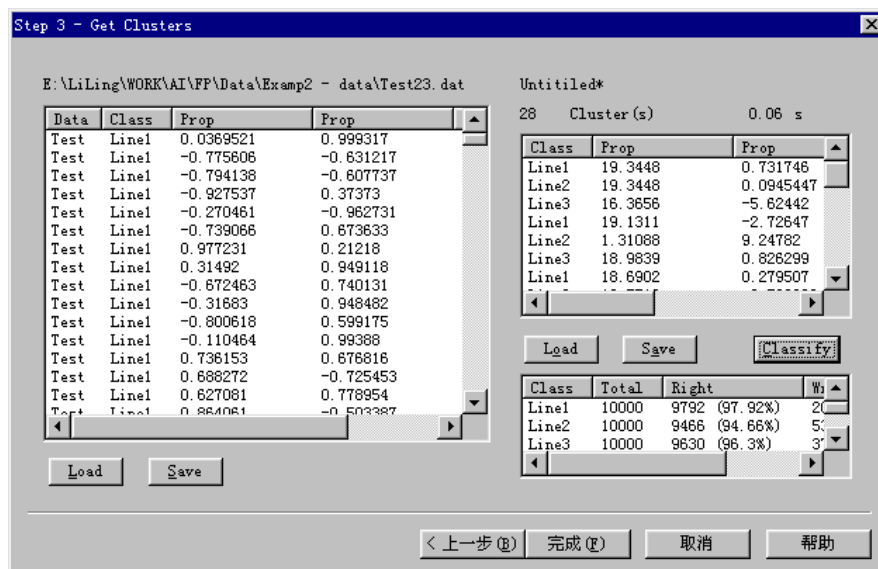


图 B.3: 第二套程序在进行分类测试时的界面，所用的测试数据为 3.1