带混合属性的神经网络规则提取方法

何晓琴[1] 张向华[2] 白 勇[1]

(重庆电力高等专科学校计算机科学系 重庆 400044)[1] (重庆教育学院 重庆 400067)[2]

摘 要 人工神经网络应用中最大的弊端是缺乏可理解性,而对结果的解释是任何一个完善的智能系统必备的基本特征。从神经网络中提取规则被公认为是解决该问题最有效的手段之一。因此,所提取规则的可理解程度成为衡量规则提取算法质量的重要指标。目前该领域的研究主要集中在分类规则的提取上。对于分类问题,待测模式的属性的取值可能是离散的,也可能是连续的。现有的算法针对全连续或者全离散的问题已取得较好的效果。但对既包含连续属性也包含离散属性的问题,已有算法未取得理想的结果。本文针对带混合属性的分类问题,提出了一种规则提取算法,在提取规则的可理解性上同时照顾了连续属性和离散属性。

关键词 人工神经网络,规则提取,混合属性

Method for the Identification of Rules in the Nerve Net with Complex Property

HE Xiao-qin[1] ZHANG Xiang-hua[2] BAI Yong[1]

(Department of Computer Science, Chongqing Electric Power College, Chongqing 400044, China)[1]

(Chongqing Education College, Chongqing 400067, China)[2]

Abstract In the application of manual nerve net, the lacking of understandability is the most serious shortcoming, so the ability to explain the result is the essential characteristic for all the capable intelligent systems. The identification of rules from the nerve net is recognized as one of the most efficient methods to solve such problem. Therefore, the understandability of the identified rules becomes the important index to evaluate the arithmetic quality of the identified rules. Presently, the main study of this field is concentrated on the identification of rules in classification, in which the property number of the model to be measured is either dispersed or consistent. The present arithmetic has already achieved a lot as dealing with the pure consistent or dispersed number. However, there is still a large gap as dealing with the result containing both consistent and dispersed number. Hence, this thesis will put forward an arithmetic applied in the identification of rules for such complex property in classification, which takes care of both the consistent property and the dispersed property for the understandability of identified rules.

Keywords Manual nerve net, Identification of rules, Complex property

1 引言

作为机器学习的工具,神经网络已经成功地应用到许多领域。由于神经网络的优越性,例如较好的预测精度、鲁棒性、并行性等,许多研究者投入到这个领域,并且进行了深入的研究。但是将神经网络作为预测工具最大的不足,即是缺乏可理解性,这已经影响到神经网络的应用。知识以连接、权值和偏置的形式隐式地嵌入在神经网络中,造成知识不易理解。神经网络不能使用户理解它是怎么做决定的,这将降低神经网络所供的推理和建议的可信度。

解决这个问题办法之一是从神经网络中提取规则,这种方法是基于这样一个思想——某个规则集可以逼近一个确定的神经网络。近十多年,国际上提出了许多从神经网络中抽取规则的方法。

Fu[4,5]提出的KT算法是这一领域早期算法。它假定激活单元的值是0或者1,主要处理过程是搜索使得输出单元激活的输入层单元的子集。该算法将这个过程看成一个组合问题。

Towell和Shavlik[6]将网络的权值聚成几个等价的类,删掉对节点激活值影响不明显的类。规则提取过程与KT算法类似,但是规则的形式更容易理解。此后,许多研究者都努力提取出这种形式的规则。

上述方法都是适合带离散值输入的问题,因为这些方法抽取出的规则的前件是离散布尔表达式的组合。对具有连续值输入的问题,它们首先离散化连续输入,这样,把输入连续的问题转化成为所有输入都是离散的问题。Rudy Setiono和Huan Liu[8]提出的NeuroLinear算法不用离散化连续输入,故适合于输入是全连续的问题。这种算法抽取的规则前件是一个线性表达式,可以视作超平面。如果所有的输入都是连续的,这种规则就非常容易理解,并且很容易在几何上解释它。但是,如果问题带有一些离散的输入,这种形式的规则是不易理解的,因为对于离散的变量,布尔表达式更适合。

综上,现有的方法可以处理输入都是离散或者都是连续的问题,因为这些方法提取的规则,或者对连续值易于理解,或者对离散值易于理解。本论文提出了解决同时带有连续值输入和离散值输入的问题的方法。本文第2节叙述了此类问题的规则提取过程,包括训练和剪枝神经网络,离散化隐层节点的值,提取超平面规则,分离规则中的连续属性和离散属性。第3节以一个经典的数据集为例说明规则提取的过程。最后是总结。

2 规则提取

在神经网络规则提取方法研究的初期,对于连续属性的处理,通常是将某个连续属性的取值分成若干个区间,分别给

予编号，样本在该属性上的取值落在哪个区间内，则该样本在这个属性上就赋予相应区间的编号。这样，把一个连续问题离散化了。例如，每个样本有两个连续的属性(x_1，x_2)，每个属性的取值都分布在区间[0，1]，提取出的规则类似形式如下：

IF $0.2<x_1<0.7$ AND $0.3<x_2<0.8$ THEN

$t=1$，ELSE $t=0$ (1)

上述规则在几何上有非常直观的解释，所表示的分类边界与坐标轴平行，如图(a)Rudy Setiono 在他们的文章中所提出的方法、提取出的规则类似形式如下：

IF $(-x_2+1.5x_1)<-0.258$ AND $x_2 \geqslant x_1$ THEN

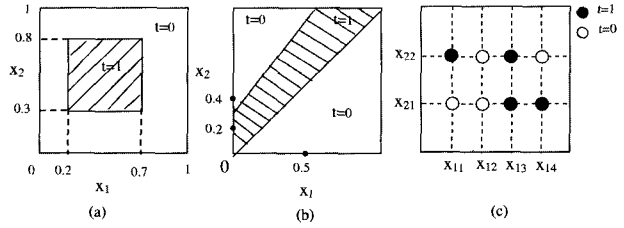
$t=1$，ElSE $t=0$ (2)

上述规则在几何上也有直观的几何解释，所表示的分类表界是倾斜的超平面如图(b)。

对于离散的属性，规则的前件通常被表示成布尔函数的形式。例如样本有两个离散的属性(x_1，x_2)，x_1 可能取值 x_{11}，x_{12}，x_{13}，x_{14}；x_2 可能取值 x_{21}，x_{22}。总的可能情况有 2×4 种，提取出的规则类似形式如下：

IF $(x_1=x_{12}) \vee (x_1=x_{11} \wedge x_2=x_{22}) (\vee (x_1=x_{13} \wedge x_1=x_{14}) \wedge (x_2=x_{21}))$ THEN $t=1$，ELSE $t=0$ (3)

规则(9)在几何上也有直观的解释，描述的是空间中的点集，如图(c)。



从上面的分析，可以得出结论：对于神经网络规则提取方法所提取的分类决策区域的边界，超平面能更好地描述对连续属性的限定，而对于离散属性，点集的形式更能说明问题。应该注意到，上面的讨论都是基于全连续或全离散的问题，而对于既有连续属性又有离散属性的问题，目前尚无较好的解决方法。

神经网络规则提取方法最根本的目的是在保证一定精度的前提下提取出尽可能容易理解的符号规则。基于此本文力图对带混合类型(连续和离散)的问题从神经网络中提取出易于理解的符号规则。

本文所提出的规则提取过程分为如下步骤：

Step1　建立并训练一个神经网络，通过剪枝，去掉冗余的连接。

Step2　隐层结点的激活值离散化。

Step3　提取出描述隐层与输出层关系的规则集 1，描述输入层与隐层关系的规则集 2，整合规则集 1 和 2，得到规则集 3(倾斜超平面)。

Step4　从规则集 3 中分离连续属性和离散属性，得到规则集 4(混合类型)。

2.1　训练和剪枝神经网络

我们同时用连续值和离散值集中讨论一个问题。简而言之，我们总是假设论文中所论述的问题具有连续属性值 c_1，c_2，…，c_r 和离散属性值 d_1，d_2，…，d_t，(其中 $r+t=n$)，并且为了简化，本文只讨论二分类问题，这个问题可以很容易地扩展到多类别问题。

创建一个标准的三层前馈全连接神经网络，用 0 与 1 之间的随机数初始化它的权值。ω_{ij} ($i=1,2,\cdots,n$；$j=1,2,\cdots,h$)表示神经网络输入层单元 i 和隐层单元 j 连接的权值。v_{jk} ($j=1,2,\cdots,h$；$k=1,2,\cdots,m$,)表示隐层单元 j 和输出层单元 k 连接的权值，其中 n，h 和 m 分别是输入层、隐层和输出层单元数目。隐层神经元和输出层神经元的激发函数都选取 sigmoid 函数，即

$$s(x)=1/(1+e^{-x}) \tag{4}$$

这样，对于给定的样本 p(期望输出 Y_{kp})，神经网络的隐层单元 j 的输出 H_{jp} 和网络的实际输出 O_{kp} 分别计算如下：

$$H_{js}=s\left(\sum_{i=1}^{n}\omega_{ij}x_i-\theta_j\right) \tag{5}$$

$$O_{ps}=s\left(\sum_{j=1}^{h}v_{jp}H_{js}-\xi_p\right) \tag{6}$$

其中，θ_j，ξ_k 分别表示隐层单元 j 和输出单元 k 的偏置。

网络的学习算法选用标准的后向学习方法，这种方法在许多应用中被广泛采用[1,2,11,12]。

由于我们一开始建立的是全连接的网络，因此训练后的网络结构比较复杂，主次不明确，不利于提取规则。在保证网络具有较高预测精度的前提下，对网络进行剪枝，去掉相对不太重要的神经单元或者连接。本文采用类似于 NeuroRule[7] 中的剪枝算法，算法的证明过程详见文献[7]，以下仅给出剪枝算法的具体步骤：

(1)设 η_1 和 η_2 为两个正数，满足 $\eta_1+\eta_2<0.5$ (η_1 是网络的容错标准，η_2 是权重剔除的阈值)；

(2)选择一个全连接的神经网络，训练网络满足式(4)的准确率，(ω,v)为网络的权重集

$$|e_{kp}|=|Y_{kp}-O_{kp}| \leqslant \eta_1, k=1,2,\cdots,m, \eta_1 \in [0,0.5); \tag{7}$$

(3)对于网络中的每一个 ω_{ij}，如果满足

$$\max_p |\omega_{ij}\times v_{jp}| \leqslant 4\eta_2, \tag{8}$$

将 ω_{ij} 从网络中剔除；

(4)对于网络中的每一个 v_{jp}，如果满足

$$|v_{jp}| \leqslant 4\eta_2, \tag{9}$$

则将 v_{jp} 从网络中剔除；

(5)如果没有权重满足(4)或者(5)，则计算 $w'=\max_p |v_{jp}\times \omega_{ij}|$，去掉具有最小值的 $\omega_{ij'}$；

(6)重新训练网络，如果分类的准确率低于预定标准则用前一次的网络权重代替训练后的网络权重，否则转步骤(3)。

2.2　离散化隐层节点的值

在神经网络的训练过程中，使用硬限幅阈值函数，隐层单元的激活值要么是 1，要么是 0。而使用平滑的激活函数(如(1))，隐层单元激活值的取值范围是(0，1)，且在该区间上连续变化。将隐层单元的激活值进行聚类，被分成若干等级，再提取规则[6]。

本文所使用的聚类算法如下：

(1)$\{a_i, i=1,\cdots,N\}$是某隐层单元激活值的集合，N 是该隐层单元激活值的数目。令 $C_1=a_{i1}$，$C_2=a_{i2}$，$C_3=a_{i3}$，a_{i1}，a_{i2}，a_{i3}是从集合$\{a_i\}$中随机选取的。$\mathrm{sum}(j)=a_{ij}$，$\mathrm{count}(j)=1$，$D_j=0$，$j=1,2,3$

(2)令 $C(j)=\mathrm{sum}(j)/\mathrm{count}(j)$，$j=1,2,3$

对所有的 a_i，$i=1,2,\cdots,N$，如果存在 $\bar{j}$，使得

$$d_{\bar{j}}=|a_i-C_{\bar{j}}|=\min_{j=1,2,3}|a_i-C_i| \text{ and } |a_i-C_{\bar{j}}| \leqslant \varepsilon \ (\text{let } \varepsilon \in (0,1))$$

那么，令 $\mathrm{count}(\bar{j})=\mathrm{count}(\bar{j})+1$，$\mathrm{sum}(\bar{j})=\mathrm{sum}(\bar{j})+1$，

$D_{\bar{j}}=D_{\bar{j}}+d_{\bar{j}}$, and $C(\bar{j})=\mathrm{sum}(\bar{j})/\mathrm{count}(\bar{j})$

(3) 如果 $\sum_{j=1}^{3}D_j>\zeta$(根据实际情况选定) 那么,$D_j=0$, $\mathrm{sum}(j)=0$, $\mathrm{count}(j)=0$, $j=1,2,3$,并且重复(2)。否则,结束。

2.3 提取超平面规则

这个过程提取的规则前件是线性不等式 $\tau_1<\sum_i\omega_i x_i<\tau_2$,其中 ω_i 是系数,x_i 是属性的取值,τ_1,τ_2 是阈值,我们称它为超平面。这个过程分为两个阶段。第一,根据离散化的隐层激活值提取描述分类的规则。第二,根据输入属性值提取描述隐层节点激活值的规则。

由于隐层单元的值已被离散化,我们可以选取现有的任何一种处理离散输入问题的规则提取方法([ ][ ][ ])。实际上,由于我们之前已经对网络进行了剪枝,去掉了相当大一部分冗余的神经元和连接。这样,只有少数神经元剩下。我们完全可以不在计算机程序的帮助下,人工地完成规则提取的过程。

在第二阶段中,我们注意到剪枝后剩下的每个隐层单元的输出都是由下式决定的:

$$f(Net_j)=\frac{1}{(1+e^{-Net_j})}$$

其中,$Net_j=\sum_i w_{ij}x_i$ (10)

在 Step2 中,已经将分布在(0,1)区间的隐层单元值聚成了若干个子区间$[\tau_{j-1},\tau_j]$,由(10),又 sigmoid 函数是严格单调递增的,所以

$$f^{-1}(\tau_{j-1})\leqslant\sum_i w_{ij}x_i\leqslant f^{-1}(\tau_j), \quad (11)$$

其中 $f^{-1}(x)$是 sigmoid 函数的逆函数。

$f^{-1}(x)=-\ln(1/x-1), x\in(0,+\infty)$

此时,将(11)带入第一阶段所提取的规则的前件中,则得到规则集,描述了数据集原始属性与所属类别的关系。

2.4 分离连续属性和离散属性

由 2.3 节,得到形如(1)式的规则,正如前文所述,这样的规则形式对于混合型属性是不易理解的。因此,需要将所涉及的离散属性从中分离出来。例如,对于规则

IF $\omega_1x_1+\omega_2x_2+\omega_3x_3+\omega_4x_4\geqslant\phi$ THEN $t=1$,
ELSE $t=0$ (12)

其中,ω_i 是一组实形的系数,属性 x_1,x_2 具有二元离散值,x_3,x_4 具有连续值,并且 $x_3\in[\mu_1,\mu_2]$, $x_4\in[\nu_1,\nu_2]$。x_1,x_2 可能的组合共 2^2 种(离散属性的个数为 2)。

对这条规则的分离过程的流程如下:

(1) 对 x_1,x_2 某种可能组合,代入(12)的前件中,得到关于 x_3,x_4 的线性组合不等式。

(2) 将上一步得到的不等式与 $x_3\in[\mu_1,\mu_2],x_4\in[\nu_1,\nu_2]$ 取交集,若交集为空,且 x_1,x_2 尚有未进行搜索的组合,则转(1),若交集不空,则得到 x_3,x_4 所组成输入子空间的一个超平面划分。

(3) 合并步骤(1),(2)所得到的关于 x_1,x_2,x_3,x_4 的限定,取代(12)的规则前件,如果 x_1,x_2 尚有未进行搜索的组合,则转(1),否则,结束。

例如,将 $x_1=1,x_2=0$ 代入,则 $\omega_3x_3+\omega_4x_4\geqslant\varphi-\omega_1$,令其与 $x_3\in[\mu_1,\mu_2],x_4\in[\nu_1,\nu_2]$取交集,得规则如下:

IF$(x_1=1)\wedge(x_2=0)\wedge(\omega_3x_3+\omega_4x_4\geqslant\varphi-\omega_1)\wedge(x_3\in[\mu_1,\mu_2])\wedge(x_4\in[\nu_1,\nu_2])$ THEN $t=1$,ELSE $t=0$。

通过上面的步骤,我们成功地将连续属性和离散属性分别用各自更适合的形式表示出来。

3 实验

这一节,我们用一个实际的例子来说明和验证本文所提出的规则提取方法。选择测试规则提取算法的典型数据集 Cleveland heart disease dataset。这个数据集共包含 303 个样本,除去属性信息不全的样本,剩余 297 个有效样本。将数据集中全部有效样本随机地分成两部分,2/3 的样本作为训练数据集,1/3 的样本作为测试集。数据集的描述及其与神经网络输入的对应关系见表 1

表 1

Attribute	Type	Possible values/Range	Network inputs
A_1	Continuous	[29,77]	x_1
A_2	Discrete	0 or 1	x_2
A_3	Discrete	1,2,3 or 4	x_3,x_4,x_5,x_6
A_4	Continuous	[94,200]	x_7
A_5	Continuous	[126,564]	x_8
A_6	Discrete	0 or 1	x_9
A_7	Discrete	0,1 or 2	x_{10},x_{11},x_{12}
A_8	Continuous	[71,202]	x_{13}
A_9	Discrete	0 or 1	x_{14}
A_{10}	Continuous	[0,6.2]	x_{15}
A_{11}	Discrete	1,2 or 3	x_{16},x_{17},x_{18}
A_{12}	Discrete	0,1,2 or 3	$x_{19},x_{20},x_{21},x_{22}$
A_{13}	Discrete	3,6 or 7	x_{23},x_{24},x_{25}

每个样本由 13 个属性描述($A_1,A_2,\cdots,A_{13}$),其中 5 个是连续的,8 个是离散的。所有样本被分作两类,要不健康(class label=0),要不有心脏病(class label=1)。在建立神经网络时,一个有连续值的属性对应一个网络输入,一个具有 n 个离散取值的属性对应 n 个网络输入。例如,$n=2$,则两个离散值分别对应(0,1)和(1,0)。具体对应关系见表 1。

建立一个标准的三层全连接前馈神经网络,25 个输入单元,7 个隐层单元,1 个输出。采用前述的方法,对该神经网络进行训练,剪枝后,剩下第 4 和第 7,两个隐层节点,分别对这两个隐层单元的激活值进行聚类:

1. Hidden unit 1. 两类:[0.02,0.09],[0.56,0.85]
2. Hidden unit 2. 三类:[0.09,0.23],[0.3,0.45],
[0.88,0.97]

这样,对这 6 个区间分别编号为 $h_{41},h_{42},h_{71},h_{72},h_{73}$。每个样本在这两个神经元处的激活值分别落入其中的两个子区间,一共有 6 种组合的可能,其中有 2 种导致输出目标为 0,4 种为 1。提取出反映隐层和输出层关系的规则如下:

If $(h_{42}$ and $h_{72})$ or h_{73} then t=1
Default rule:$t=0$ (13)

其中,h_{ij} 表示样本在第 i 个隐层单元的激活值落入其第 j 个子区间。

根据 2.3 节中叙述,可以提取出描述输入层与隐层之间关系的超平面规则。

例如对第 7 个隐层神经元,我们有:

If $f^{-1}(0.88)\leqslant0.20x_7-0.49x_8-1.27x_{19}+0.19x_{20}+0.74x_{21}-0.96x_{23}+0.68x_{25}\leqslant f^{-1}(0.97)$ then h_{73} (14)

根据规则集 1 与规则集 2 整合起来,得规则集 3,例如将上述规则(13),(14)合并,得:

If $f^{-1}(0.88)(0.20x_7-0.49x_8-1.27x_{19}+0.19x_{20}+0.74x_{21}-0.96x_{23}+0.68x_{25}\leqslant f^{-1}(0.97)$ then $t=1$ (15)

(下转第 272 页)

如果该循环体没有执行，就出现了图 8 消息 16、17 所示的调用关系；如果循环体的判断条件被满足，而条件语句的判断条件不满足，调用关系就如图 8 中消息 84 至 88 所示。从修改后的序列图中很容易理解到 setCarDistance 函数的作用是将停车场中所有的车遍历一遍，以不冲突为前提放置场内汽车。IsVerticalClashing(acar, another) 等函数和函数 setCarDistance()的关系完全体现出来，逻辑关系清楚。

需要注意的是，在图 9 中的选择项中，有的判断条件本身就是函数。为了避免重复显示造成混淆，在生成序列图时将这种函数视为条件，不视为消息，也不进行消息编号。

从实验分析中可以看出，基于动静态信息相结合逆向生成的序列图最小粒度为对象，并且深入函数内部，表现出了函数间的深层调用关系，而不仅仅是函数的调用顺序。所以，基于动态信息和静态的程序依赖关系生成的序列图更为合理和全面，有助于用户对目标程序的理解。

结束语 在 UML2.0 序列图中，不但要表现出对象间消息传递的执行序列，还要体现出消息间的控制流，这给序列图的逆向生成带来了难度。本文结合本项目组开发的逆向工程工具集 XDRE 的功能特点，提出一种新的动静态信息相结合的 UML2.0 序列图逆向生成方法。该方法将程序执行所收集到的动态信息与静态程序依赖图相结合，动态信息保证序列图的有效性，静态信息保证序列图的完备性。实验表明，该方法生成的序列图易于理解、符合实际模型、主线清晰、信息全面，适合于实际应用。

参 考 文 献

[1] Rountev S R. A Interactive Exploration of UML Sequence Diagrams. Visualizing Software for Understanding and Analysis, 2005

[2] Albir S S. UML in a NutShell. O'Reilly, 1998

[3] Rountev A, Connell B H. Object naming analysis for reverse-engineered sequence diagrams // International Conference on Software Engineering. 2005: 254-263

[4] Ghezzi C, Jazayeri M, Mandrioli D. Fundamentals of Software Engineering. Prentice Hall International Ed. 1991

[5] Walker R J, Murphy G C, Freeman-Benson B, et al. Visualizing Dynamic Software System Information through High-level Models // Proceedings of the Conference on Object-oriented Programming, Systems, Languages, and Applications. 1998: 271-283

[6] DePauw W, Jensen E, Mitchell N, et al. Visualizing the execution of Java programs. Software Visualization // LNCS 2269. 2002: 151-162

[7] Oechsle R, Schmitt T. JAVAVIS: Automatic program visualization with object and sequence diagrams using JDI. Software Visualization // LNCS 2269. 2002: 176-190

[8] Tonella P, Potrich A. Reverse engineering of the interaction diagrams from C++ code // International Conference on Software Maintenance. 2003: 159-168

[9] Rountev A, Volgin O, Reddoch M. Static control-flow analysis for reverse engineering of UML sequence diagrams // The 6th ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering. 2005: 96-102

[10] Systa T, Koskimies K, Muller H. Shimba an environment for reverse engineering Java software systems. Software Practice & Experience, 2001, 31(4): 371-394

[11] Richner T, Ducasse S. Using Dynamic Information for the Iterative Recovery of Collaborations and Roles // 18th IEEE International Conference on Software Maintenance (ICSM'02). 2002: 34-43

[12] Kollmann R, Gogolla M, Capturing Dynamic Program Behavior with UML Collaboration Diagrams // The 5th European Conf. on Software Maintenance and Reengineering. Los Alamitos, 2001: 58-67

(上接第 246 页)

其中，x_7，x_8 连续属性，其它均是离散的。根据 2.4 分离连续和离散的属性。得到规则集 4，例如，上面规则(15)，将 x_{19} x_{20} x_{21} x_{23} x_{25} 代入，得

If $(0.57 \leqslant 0.2x_7 - 0.49x_8 \leqslant 2.06) \wedge x_{19}=0 \wedge x_{20}=0 \wedge x_{21}=1 \wedge x_{23}=0 \wedge x_{25}=1$

then $t=1$

结合问题的原始属性输入，上面的规则很容易地转换成：

If $(0.57 \leqslant 0.2 * \text{Trestbps} - 0.49 * \text{chol} \leqslant 2.06)$ ∧ ca is 2 ∧ thal is rever

then $t=1$.

其中，Trestbps 代表 resting blood pressure，chol 代表 cholesteral，ca 代表 the number of vessels colored.

结束语 针对带混合类型属性的问题，目前的规则提取方法所提取的规则都没有区分连续属性与离散属性。本文所提出的规则提取方法，在规则的表示形式上，将连续属性与离散属性区分，使得所提取的规则更易于理解。对于连续属性的相关条件，用相应子空间的划分即子空间的超平面来表示，对离散属性，说明其离散取值。然而，一般情况下，这种方法会使得所提取出的规则数比一般方法的多。因为，我们在分离的时候，对离散属性采取了组合试探的方法。例如，对规则(15)进行分离时，对 5 个变量的取值组合和一个线性不等式所组成的空间进行搜索，可能会有多个搜索点满足期望(∽即 t=1)∽，则一条规则变成了多条。在规则集规模和预测精度上都有待进一步研究和改进。

参 考 文 献

[1] Andrews R, Diederich J, Tickle A. Survey and critique of techniques for extracting rules from trained artificial neural networks[J]. Knowledge Based Systems, 1995, 8 (6): 373-389

[2] Alexander J A, Mozer M C. Template-based procedures for neural network interpretation. Neural Networks, 1999, 12(3): 479-498

[3] Gallant S I. Connectionist expert systems. Communications of the ACM, 1988, 31(2): 152-169

[4] Fu L M. Rule generation from neural networks[J]. IEEE Transactions on Systems, Man and Cybernetics, 1994, 28 (8): 1114-1124

[5] Fu L M. Rule learning by searching on adapted nets // Proc. 9th National Conf. on Artificial Intelligence. 1991: 590-595

[6] Towell G G, Shavlik J W. Extracting refined rules from knowledge-based neural networks. Machine Learning, 1993, 13(1): 71-101

[7] Lu H, Setiono R, Liu H. NeuroRule: A Connectionist Approach to Data Mining // Proceedings of the 21th International Conference on Very Large Data Bases. September 1995: 478-489

[8] Setiono R, Liu H. Neurolinear: From neural networks to oblique decision rules. Neurocomputing, 1997, 17(1): 1-24