第 19 卷 第 4 期 华侨大学学报(自然科学版) Vol. 19 No. 4
1998 年 10 月 Journal of Huaqiao University (Natural Science) Oct. 1998

知识分解与组合探索*

洪国彬 吴桂珍

(华侨大学计算机科学系, 泉州 362011)

摘要 知识获取一直被认为是专家系统的瓶颈问题. 文中撇开传统上通过归纳、解析等方式进行机器自学习的方法, 提出对知识进行分解, 根据语义, 实现知识的第二次分解, 最后对知识重新组合, 从而实现机器自学习的基本过程.

关键词 机器自学习, 知识表达, 知识分解, 知识匹配

分类号 TP 18

目前专家系统的研究处于停滞不前的状态, 在于知识的获取手段上没有本质上的突破[~3]. 传统专家系统的获取手段大多建立在认为机器已智能的前提上——他们力图使机器具有归纳能力. 而机器具有的最大优势在于匹配、演绎和推断能力. 因此, 我们尝试通过对知识的分解与组合, 从而从语义上达到对知识的理解和进行机器自学习. 分解与组合以及根据语义的推导, 其主要用到的是匹配、演绎和推断, 这恰是传统系统面临的问题.

1 背景

1.1 本专家系统的知识表示形式

if (前提), then(结论), cond (置信度), 其中 前提以 $a_1 \wedge a_2 \wedge \cdots \wedge a_m$ 的形式表示, 结论以 $b_1 \vee b_2 \vee \cdots \vee b_n$ 形式表示, 置信度 $cf \in \{0, 1\}$.

1.2 本专家系统的定义

ES(D, C, Z, F, S), 其中 D 为知识库, 它存储着本领域的专家知识(大量推导规则); C 为常识库, 它由Ⅰ类常识库和Ⅱ类常识库组成. Ⅰ类常识库即传统系统具有的常识库, 用于特征系统的外域, 保证系统的坚固性. Ⅱ类常识库则存储了对本领域内特定知识的语义理解, 它的数据结构形式如表 1 所示. 这里的成分集是{主体, 谓词, 修饰语}. Z 为组成基, 它表示规则前提最基本的组成部分, 用于分解后的语义检测. F 为分类基, 它表示规则结论最基本的组成部分, 用于对知识的分类. S 为算子基. 因为前提、结论用一阶谓词描述, 所以前提命题可用{¬ ∧}将原子前提命题复合成一个命题, 结论命题可用{¬ ∨}来复合. {¬ ∧}, {¬ ∨}都是全功能集. 前提用{¬ ∧}来复合容易理解, 而结论用{¬ ∨}复合是因为

表 1 数据结构形式表

论域	根词	词性与成分	解析
成分 1	用根词解析	置信度	量词
成分 n	用根词解析	置信度	量词

* 本文 1998-04-30 收到

同一个前提可能导致多个不同结论.

1.3 本专家系统的结构

系统结构如图1所示.

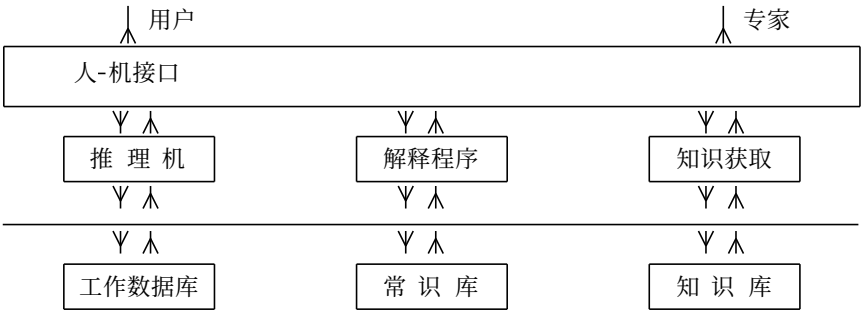

图1 专家系统结构图

2 模型

2.1 知识的分解

机器本身不具有智能,只能通过匹配来进行知识分解[4,5]. 首先,机器必须有一个识别知识成分的匹配机制,编译的研究提供了丰富的理论,我们用到的是自动机理论. 其次,因为知识按予定格式表达,所以可通过某个预处理程序,将知识分解为一个个原子命题,我们称这次分解为第一次分解;然后对各原子命题进行分解,分解出该命子命题的"个体"、"谓词"和"修饰语",我们称这次分解为第二次分解. 我们将郑重讨论第二次分解.

(Ⅰ) 讨论由于增加新的常识而引起识别自动机变化的情况. 设增加的词 $A=c_1c_2c_3\cdots c_i\cdots c_n$,则A与库中的词之间有三种关系. (1) 库中存在词 $B=Ab_1b_2\cdots b_m$,即以A为前缀. (2) 库中存在词 $B=c_1c_2\cdots c_ib_1b_2\cdots b_m$,即A与B有相同前缀 $c_1c_2\cdots c_i$,且该前缀为最大前缀. (3) 库中没有词与A有相同前缀,可根据三种情况分别进行处理. (a) 照原来的自动机进行状态转变直到该词所有符号处理完,这时到达一个中间状态结点,将该中间状态结点置为终状态结点,并将词指针赋给该状态结点. (b) $c_1c_2\cdots c_i$ 通过状态转换到达某状态结点,产生一棵状态结点的子树,用以识别后面的 $b_1b_2\cdots b_m$. (c) 从树根直接产生一棵子树以判别 $c_1c_2\cdots c_n$.

算法1. 增加的新词加入旧的词识别自动机,成为新的词识别自动机. (1) 从首状态出发,根据词符号顺序进行状态变化直到状态无法继续改变或词符号完. 记下最后一个状态结点. (2) IF 词符号完,THEN 将该词指针赋给保留下的状态结点的词指针. (3) IF 保留的状态结点为根结点,THEN 产生根结点的一棵子树. (4) IF 保留的状态结点既不为根结点且词符号又没完,THEN 产生一棵保留的状态结点的一棵子树. (5) 结束.

(Ⅱ) 讨论由于删除库中的词引起自动机的变化的情况. 仍设删除的词 $A=c_1c_2\cdots c_n$,该词与库中其它词的关系同上. 对三种情况也分别处理:(1) 将该词的终状态结点的该词指针为空,表明此结点为中间结点;(2) 删除能识别前缀 $c_1c_2\cdots c_n$ 的状态 $c_i\cdots c_n$ 子树;(3) 删除根结点下的 $c_1c_2\cdots c_n$ 子树.

算法2. 删除词引起词识别自动机的改变:(1) 记录识别该词路径上顺序经过的状态结点;

(2) IF 终态结点不是叶子结点:

THEN 将该终态结点的词指针清空(置 Ni1)

ELSE WHILE 该终点的词指针不为空, 且该结点为叶子结点 DO 删除该结点;

取被删除结点的父结点;

ENDWHILE;

(3) 结束.

(Ⅲ) 有了算法 1, 可以容易地建立各识别自动机.

算法 3. 建立词识别自动机: (1) 输入一个词到库中; (2) 根据算法 1, 修改自动机; (3) IF 未输入完 THEN GOTO step1; (4) 结束.

(Ⅳ) 个体、谓词、修饰语之间的差别是明显的. 现采用最大正向匹配法进行分解. 由于谓词贯穿于整个命题中, 个体、修饰可能有多个 . 为了识别多元谓词, 将多元谓词中的个体用一个特殊的符号(设为@)代替. 分解时, 对要分解的串先判断是否有前缀的个体或修饰语, 以成为个体或修饰语的最大前缀为主; 若都不存在这种前缀, 才接着判别是否谓词的组成部分. 已知输入命题字符串 $P= c_1c_2\cdots c_n$; 分解出的个体、谓词和修饰语输出存入工作数据库. 算法将使用到 C—— 字符变量; FstCP, CP —— 字符指针, FstCP 指向未识别的字符串; ActionI, ActionA —— 分别为个体识别自动机、修饰语识别自动机的状态标志, 其值域= {ACTIVE, UNACTIVE}. ACTIVE 表示自动机活跃; UNACTIVE 表示自动机封闭; I, A(串指针), 分别用来保留识别出的个体和修饰语 . lenI, lenA 分别为识别出的个体和修饰语的长度; SI, SU, SA 分别为个体识别自动机、谓词识别自动机的修饰语和识别自动机的状态结点指针; SIO, SUO, SAO 分别指向相应自动机的树根.

2. 2 知识分解过程

算法 4. 知识分解为

(1) 初始

FstCP= 1; ActiveI= ActiveA= ACTIVE; I= A= NIL; lenI= lenA= 0

SI= SIO= SUO; SA= SAO

(2) C= P[CP]

(3) IF ActiveI= ACTIVE THEN

IF SI 结点不是叶子结点且 SI 结点与其孩子结点间的权值有字符 C

THEN { SI 根据字符 C 指向孩子结点;

IF SI 结点为某个体的状态结点且字符 P[CP+ 1] 是分隔符

THEN { 将 SI 结点中的词指针保留在 I 中; 记下该词长度: lenI= CP- FstCP+ 1

} }

ELSE ActiveI= UNActive

(4) IF ActiveA= Active THEN

IF SA 结点不是叶子结点且 SA 结点与其孩子结点间权值中有字符 C

THEN { SA 根据字符 C 指向孩子结点;

IF SA 结点为某个体的终态结点且字符 P[CP+ 1] 是分隔符

THEN { 将 SA 结点中的词针保留在 A 中;

记下该词的长度: lenA = CP- FstCP+ 1;
}
}
ELSE ActiveA= UNACTIVE

(5) IF ActiveI= ACTIVE∨ActiveA= ACTIVE
THEN { CP= CP+ 1;
IF 字符 P[CP]是结束符
THEN GOTO step1;
ELSE GOTO step2;
}

(6) IF lenA= lenI= 0
then
IF SU 结点不是叶子结点与其孩子结点之间的权值有字符 C
THEN SU 根据字符 C 指向孩子结点;
ELSE 发现语法错误; GOTO step11

(7) IF lenA< lenI
then { 将个体 I 存入工作数据库; 以@替代个体 I;
IF SU 结点不是叶子结点, 且 SU 结点与其孩子结点间的权值有字符@
THEN SU 根据字符@ 指向孩子结点;
ELSE 发现语法错误; GOTO step11; }

(8) IF lenA> lenI THEN 将修饰语 A 存入工作数据库

(9) CP= CP+ 1; C= P[CP]

(10) IF C 不是串结束符 THEN
IF SU 不是叶子结点与其孩子结点间的权值有字符 C
THEN { SU 根据字符 C 指向孩子结点: GOTO step9; }
ELSE { FstCP= CP; lenI= lenA= 0; ACTIVEI= ACTIVEA= ACTIVE;
I= A= NIL; SI= SIO; SA= SAO; GOTO step2 }

(11) 结束.

2.3 副词的产生

首先简述副词产生的原理. 两个模糊概念 A, B 之间存在推导关系 A→B, 这时若 A 有副词 Q 修饰, 只要 Q 修饰不超过一定的界限, 根据 A→B, 可得由相应的一个副词 Q 修饰结论 B, 即 QA→Q'B, Q 是副词的产生的预期效果. 副词相当于一种语言修正因子, 理论上每个修正因素的作用都有一个与它们等价的运算表示, 我们定义系统中与各副词作用等价的运算 $f_{副词}$. 例如, "非常 t_i ", 我们定义为 $f_{非常}[t_i]= t_i^2$. 有了隶属函数, 定义了 $f_{副词}$, 可根据 Zadeh 提出的 "合成推理规则"进行推导. 这种规则的基本原理, 首先求示出一个前提中两个模糊概念的模糊关系, 最后再求另一个前提中的模糊概念与该模糊概念的模糊关系, 然后再求另一个前提中的模糊概念与该模糊关系的 max- min 合成 "。", 即可得推理的结论. 得到结论后, 判断该结论最有可能由哪个副词产生作用的原理, 这样即可产生副词.

在系统论域上取 m 个值代表一个模糊概念的隶属函数. 任意一个模糊概念都表示成 $A = U_A(u_1)/u_1, \cdots, U_A(u_m)/u_m$, 我们采用 Mizumoto 等人[6]提出的 Rs 方法, 求两模糊概念之间的模糊关系. Rs 定义的模糊关系为

$Rs = A \times V \Rightarrow U \times B = \int_{U\times V} [U_A(u) \rightarrow U_s(v)]/(u,v)$, 其中

$$U_A(u), U_s(v) = \begin{cases} 1 & U_A(u) \leq U_B(v), \\ 0 & U_A(u) > U_B(v), \end{cases}$$

$A[u_1, \cdots u_m] = \{U_A(u_1)/u_1, \cdots, U_A(u_m)/u_m\}$,

$B[v_1, \cdots v_n] = \{U_B(v_1)/v_1, \cdots, U_B(v_n)/v_n\}$,

设空间 $Rs[u_1, \cdots u_m, v_1, \cdots v_n]$, $QA[u_1, \cdots u_m]$, $Q'[v_1, \cdots v_n]$ 置初值为0.

算法5. 由 $A \rightarrow B \wedge QA \Rightarrow QA \rightarrow Q'B$, 求 Q'.

(1) 建立模糊关系 Rs:

```
FOR i= u1 TO Um DO
  FOR j= v1 TO Vm DO
    IF UA[i] ≤ UB[j] THEN Rs[i,j]= 1; ELSE Rs[i,j]= 0;
```

(2) 建立 QA 一元关系:

```
FOR i= u1 TO um DO QA[i]= fQ(UA[i]);
```

(3) QA^0Rs(max→min 合成):

```
FOR i= v1 TO vm DO
    FOR j= u1 TO um DO
        IF min(QA[j], Rs[j,i])> Q'B[i]
        THEN Q'B[i]= min(QA[j], Rs[j,i]);
```

(4) 对 B 进行各种修正等价运算, 求得各 $Q'B$ 把各个 $Q'B$ 与 $Q'B$ 比较, 取最近似的 $Q'B$;

(5) 结束, 返回 Q''.

2.4　置信度的产生

设存在规则 "if x is A then y is B", A 和 B 是两个模糊概念, 分别用各自认域 U 和 V 上的模糊集合表示. $U \times V$ 是两者的模糊关系. 定义 $u_1 \in U$, $v_1 \in V$, 若 $U_A(u_1) = 0$ 或 $U_B(v_1) = 0$ 或两者都为0, 则这时的规则不可信. $U_A(u_1) \neq 0$ 且 $U_B(v_1) \neq 0$, 原来的模糊规则就变成了具体规则, 于是就有一个具体的置信度. 值得注意, 这里的置信度与原规则的置信度不同, 原规则中的置信度表明的是一种趋向. 假设 $U \times V$ 中, 存在 m 个 $\langle u_i, v_i \rangle$, 其中 $U_A(u_i) \neq 0$, $U_B(v_i) \neq 0$. 由此得知相应的 m 个 cf_i 值(置信度), 这 m 个知识, 即同类知识, 由这 m 个 cf_i 可模拟一条 cf 函数曲线. 利用牛顿插值法求近似函数曲线, 求出函数为 $P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)\cdots(x - x_n - 1)$, 求出 $a_0, a_1, \cdots a_n$ 等待定系数后, 函数即为可知.

算法6. 求 $a_0, a_1, \cdots a_n$ 等待定系数.

(1) FOR i= 1 TO n DO $a[i] = cf_i$;

(2)
```
FOR j = 1 TO n DO
    FOR i= 1 TO n DO
      a[i]= (a[i] - a[i- 1])/Ui- Ui- 1);
```

(3) 结束.

利用插值函数可求得cf值. 当然还可以用贴近度的方法求cf值, 即若A与B为论域U上的模糊集, A与B的贴近度记为(A, B), 即

$$(A, B) = 1/[A \in B + (1 - A \int B)]$$

其中的 $A \in B = V(A(u) \wedge B(u))$.

$A \int B = \wedge(A(u) \vee B(u))$ 即A与B的内积与外积, "$\vee$" = "max", "$\wedge$" = "min".

3 结论

本文中, 我们提出了知识分解与组合的一些原理, 它在自学习方面更合乎机器的特性. 我们定义了一个具备分解与组合功能, 拥有Ⅱ类常识库的专家系统. 在此基础上, 研究说明了知识分解及分解后如何根据语义, 利用副词自学习, 以及置信度改变的原理, 并提供几个算法. 文中, 我们对某些方面还未做深入研究, 比如当超过某个极限时, 知识发生质变的情况及置信度产生中尚存在某些问题. 限于篇幅, 我们也未能对分解后如何利用量词进行自学习作说明.

文中算法的实现, 是用Borland C++ 3.1结合Paradox(数据库管理语言)编写的.

参 考 文 献

1 洪国彬. 专家系统开发环境的思想及其设计实现. 计算机应用与软件, 1998, 7: 13～16
2 林尧瑞, 张 钹, 石纯一. 专家系统原理与实践. 北京: 清华大学出版社, 1988. 91～103
3 范慧琳. 知识概念表达与处理. 华侨大学学报(自然科学版), 1995, 16(3): 349～352
4 陆汝钤, 庄庆丽, 吴建敏. 推进知识获取方法学的研究. 见: 孙怀民主编. 知识工程进展. 武汉: 中国地质大学出版社, 1988. 215～219
5 周 青, 王树林, 迟忠先. 自动书本知识获取系统TKAS. 见: 许卓群主编. 知识工程进展. 武汉: 中国地质大学出版社, 1990. 189～194
6 杨雪南, 李德毅. 关系数据库中的模糊知识发现. 软件学报, 1995, 6(1): 61～64

A Probe into Decompostion and Composition of Knowledge

Hong Guobin Wu Guizhen

(Dept. of Comput. Sci., Huaqiao Univ., 362011, Quanzhou)

Abstract Knowledge acquisition is considered as bottleneck of expert system. Leaving aside traditional method of machine self-learning by way of induction, analysis, etc., the authors describe the process of realizing machine self-learning. The process starts with decomposition of knowledge, and passes through the second decomposition of knowledge in the light of semantic; and completes in the recomposition of knowledge.

Keywords machine self-learning, knowledge acquisition, decomposition of knowledge, match of knowledge