基于遗传与粒子群算法的Markov逻辑网学习研究

于 鹏, 刘大有, 欧阳 丹彤
(吉林大学计算机科学与技术学院, 吉林长春 130012)

摘 要: Markov逻辑网(MLN)是一种重要的统计关系模型, 目前其学习问题主要采用确定性的优化方法, 所求的解不够简洁、易陷入局部极值. 针对这些问题, 本文定义谓词模板、子句模板以缩小搜索空间, 提出一种采用子句模板编码的遗传算法来学习MLN结构, 并用粒子群算法学习MLN的权参数. 文中设计了适应度函数和相应的遗传算子, 保证算法不断向好的逻辑子句结构进化. 理论分析与实验结果都表明本文的算法可以学习到较优解.

关键词: 统计关系学习; Markov逻辑网; 谓词模板; 子句模板; 遗传算法; 粒子群算法
中图分类号: TP183 文献标识码: A 文章编号: 0372-2112 (2006) 12A-2551-05

Research on Learning Markov Logic Networks Based on GA and PSO

YU Peng, LIU Da-you, OUYANG Dan-tong
(*College of Computer Science and Technology, Jilin University, Changchun, Jilin 130012, China*)

Abstract: Markov Logic Networks (MLN) is an important model in Statistical Relational Learning. Nowadays deterministic search methods are the main methods to learn MLNs. However, the result was not compact and the algorithms can easily get into the local optima. Aiming at solving these problems, we defined the predicate template and the clause template to reduce search space and put forward a learning algorithm using Genetic Algorithm (GA), which code is clause template, to learn MLNs' s structure, using Particle Swarm Optimization (PSO) to learn MLN' s weight. We gave the fitness function which makes the algorithm convergence, design the genetic operators. We testify our algorithm can get better result by theoretical analysis and experiment comparison.

Key words: statistical relational learning; markov logic networks; predicate template; clause template; genetic algorithm, particle swarm optimization

1 引言

统计关系学习, 简称SRL(Statistical Relational Learning)或PLL(Probabilistic Logic Learning), 主要研究复杂的关系数据形式, 是一个新兴发展的研究领域[1], 对它的研究方法主要集中于对基于概率理论的似然表示与推理机制, 如Bayesian网, Markov网等的扩展, Markov逻辑网(MLN)[2]是其中一类重要的模型, 可看作是一种统一的SRL框架[3]. MLN是一个二元组集合$\{(F_i, w_i)\}$, 其中F_i为一阶逻辑表示的公式, w_i为正实数. MLN可看成是生成Markov网的模板, 给定一组常量MLN对应一个Markov网$M_{L,C}$, 它具有两种性质: (1) MLN中每个基谓词对应$M_{L,C}$中的一个节点, 若基谓词为真, 则节点值为1, 否则为0; (2) MLN中每个公式F_i的基公式对应$M_{L,C}$中一个特征函数, 若基公式为真, 则特征函数值为1, 否则为0. 特征函数的权值就是L中F_i所对应的w_i, 其中C为常量的有限集合$\{c_1, c_2, \cdots, c_{|C|}\}$. 两个节点之间有边当且仅当这两个基原子同时出现在同一公式中.

学习MLN包括结构学习与参数学习, 结构学习指获得逻辑公式[2]; 参数学习指获得MLN的权值, 它通常在结构学习后才进行. 当前MLN的结构学习问题主要采用确定性的搜索方法来得到较优的解[4], 因此存在着搜索得到的解不够简洁易陷入局部极值的情况. 而MLN的参数学习主要采用对联合分布函数取极大似然值的方法[5], 因此计算复杂, 尤其对于较大的数据集不利于学得较优解. 研究一种避免陷入局部极值, 具有快速搜索能力的MLN学习算法是有意义的. 本文对此进行了研究, 定义谓词模板和子句模板来表示谓词与子句的结构, 采用遗传算法来学习子句模板从而缩小了搜索空间, 之后再将其转化为子句; 采用粒子群算法(PSO)学习权值, 进而提出了一种MLN学习方法, 实验结果显示该算法可有效的学习MLN.

2 MLN学习算法MGAPSO

逻辑公式的学习可简化为子句的学习[6], 常规的确定性优化算法搜索子句易陷入局部极值的情况, 而遗传算法适宜

收稿日期: 2006-09-08; 修回日期: 2006-11-26
基金项目: 国家自然科学基金重大项目(No. 60496321): 非规范知识处理的基本理论和核心技术; 国家自然科学基金(No. 60373098, No. 60573073); 吉林省科技发展计划(No. 20030523)

解决这类问题. 遗传算法学习子句已有一些研究[7~11], 但这些方法以直接搜索子句为目标, 面对的搜索空间很大. 本文搜索满足条件的子句模板, 再将其转化为子句, 缩小了搜索空间. MLN 的权值反映了一个领域违背该条公式与满足该条公式的可能性之间的差别. 我们直接使用粒子群算法由联合分布函数中学习权值, 避免了确定性方法求梯度等复杂的运算. 为方便下面的讨论先定义超变量、谓词模板、子句模板.

定义 **1** 设在一领域中有集合 H_c 和 C, 集合 C 由一些常量构成, 集合 H_c 由该领域中所有值域为 C 的变量与函数构成, 如果变量 v 取值为集合 $H_c \cup C$, 则称变量 v 为关于常量集合 C 的超变量.

定义 **2** 若 $p(x_1, x_2, \cdots, x_n)$ 是 n 元谓词符号, $v_1, v_2, \cdots, v_n$ 是超变量, 则 $p(v_1, v_2, \cdots, v_n)$ 称为谓词模板, 规定谓词模板具有真值标记.

定义 **3** 多个谓词模板由析取符号连接组成的表达式称为子句模板.

2.1 结构学习

2.1.1 学习子句模板

本文将子句模板直接作为基因编码, 用遗传算法进行学习. 对于一个子句模板从中随机选择一谓词模板作为其头部, 其它谓词模板作为子句模板的体. 衡量子句模板好坏的适应度函数定义为:

$$k\left(\frac{M_a}{n}+\frac{p^h}{p^g}+\frac{1}{N}\left(\sum_{i=1}^{n}(h_i-1)-m\right)\right) \qquad (1)$$

该式中 N 表示子句中出现的谓词个数即子句长度, M_a 表示该子句模板的子句体中出现了多少类超变量. p^h 表示子句模板头部覆盖的正实例个数, p^g 表示子句模板头部覆盖的所有实例的个数, $k>0$ 是调节系数可根据子句的长度来取值, m 表示子句中有多少类谓词模板的正负真值同时出现了. n 表示该子句模板中共出现了多少类超变量, h_i 表示子句中出现的第 i 类超变量在多少个谓词模板中同时出现.

初始群体采用随机生成的方法, 即随机生成一个小于等于 a(子句最大长度)的常数 h 作为子句的长度, 从待学习的数据中随机选取 h 个谓词模板, 每个谓词模板的真值随机给定. 按上述方法生成多个子句模板, 组成初始群体.

选择操作我们采用"余数随机选择"的方法[12]. 交叉操作是对待交叉的两个子句模板, 从中选择长度较大的一个, 随机选取其中的几个位置与另一子句模板的相同位置进行交叉, 经过交叉操作后, 子句头可能改变了, 因此需要重新选择子句模板的头部. 变异操作我们设计了五种:

(1) 向基因串中随机加入一个谓词模板;

(2) 从基因串中随机删除一个谓词模板;

(3) 对基因串中的某一谓词模板用其它的谓词模板加以取代;

(4) 对基因串中的某一谓词模板的真值取反;

(5) 重新选择基因串所对应的子句模板的头部;

每次对发生变异的个体随机选择上述的一种变异操作作用于基因串, (1)~(4)操作之后要重新选择子句模板的头部. 在进行交叉、变异操作之前对当前种群要进行复制保留, 新个体由这一复制的种群和遗传操作后的种群按适应度大小产生, 使最优个体得以保存.

定理 **3** 学习子句模板的遗传算法在进化中较优个体呈指数增长, 算法是收敛的.

证明 将所有谓词模板统一编码为二进制代码, 则该遗传算法等价为二进制编码的遗传算法. 由文献[13]的定理 2.6 得证.

2.1.2 子句模板转化为子句

本文仅对谓词中的项是变量的情况加以讨论, 因此只需将超变量用相应的变量取代即可完成子句模板的转化. 我们先对头部的谓词模板进行转化, 再将体中的超变量与头部的变量进行对应. 该方法依赖统计量确定变量的对应关系. 转换算法描述为算法 1:

算法 1:

```
SuperVariableConvert(DataBase db) /* db 指用来学习的关系数据库*/
Begin
  获取 db 中被子句模板头部覆盖的正实例集 P;
  For 正实例 e∈P do
    Begin
      e 中的第一项标记为 0, 其后的第 i 项都与前面的 i-1 项比较, 如与第 m(m<=i-1)项相同则标记为第 m 项的标记, 否则标记为 i-1, 这些标记构成矩阵 M 中的一行;
    End
  For M 中每列 i do
    Begin
      计算第 i 列中各标记的概率 p_ij;
      If(最大 p_ij 唯一)
        最大 p_ij 的标记作为该项的变量名;
      Else
        该项的变量名取 i-1 标记;
    End
  获得子句模板中各项在 db 中出现的常量集;
    For 头部中每类超变量 v do
    Begin
    将子句模板体中对应 v 的各项加入集合 PM;
    For pm∈PM do
    Begin
    pm 与子句头部各 v 项的常量集比较, 记录共同出现的常量个数;
    If(出现共同常量最多的头部项唯一)
      pm 项取该头部项变量值;
    Else
        随机选择一共同常量最多的头部项的变量值作为 pm 项的变量值;
    End
  End
  For 子句模板体中的每个谓词模板 t do
```

If (*t* 中有超变量多次出现)
Begin
构建 ***M*** 矩阵获得各变量名;
If(项曾用头部中的变量命名且与 ***M*** 矩阵得到的变量名不符)
重新从头部选变量;
End
对其它不出现在头部的超变量取变量名;
End

对构建矩阵 ***M*** 举例来说明, 例如谓词模板 $P(t, t, t, t)$ 对应的实例集为: $P(a_1, a_2, a_1, a_5)$, $P(a_2, a_1, a_1, a_3)$, $P(a_2, a_3, a_3, a_1)$, $P(a_1, a_1, a_1, a_3)$; 则构建的矩阵 ***M*** 为

$$\begin{vmatrix} 0 & 1 & 0 & 3 \\ 0 & 1 & 1 & 3 \\ 0 & 1 & 1 & 3 \\ 0 & 0 & 0 & 3 \end{vmatrix}$$

第一列变量标记为 0; 第二列 1 的概率最大, 变量标记为 1; 第三列的标记 0 与 1 概率相同则令其变量标记为 2, 第四列变量标记为 3.

称 ***M*** 矩阵的每行为变量的一种对应形式, ***M*** 矩阵可反映实例集所有变量的对应形式.

定理 **2**　谓词 P 的某一有限基谓词集合 C 所对应的关于超变量 X 的 ***M*** 标记矩阵, 可表示集合 C 中所反映的 X 类型变量的所有对应形式.

证明　我们采用反证法, 假设有一种变量的对应形式没有在 ***M*** 标记矩阵中描述出来, 由于对应形式至少存在于集合 C 中的一个基谓词里, 而该基谓词对应 ***M*** 标记矩阵中的一行, 这说明该行中必有两个变量的对应关系标记错误, 由于两个变量的对应关系只有相同和不同两种情况, 我们分别讨论: (1)若真实的对应关系为两个变量相同, 而在 ***M*** 矩阵中却被标记为不同的两个非负整数, 这必然是对应的基谓词中的两个常量不同所致, 可见这与集合 C 反映的对应关系矛盾; (2)若真实的对应关系为两个变量不同, 而在 ***M*** 矩阵中却被标记为相同的, 这必然是每一包含这一对应关系的基谓词中的两个常量都相同所致, 这说明集合 C 中反映不出这种对应关系, 因此是不可能的. 综上定理 2 得证.

2.2　参数学习

本文使用 PSO 算法[14]从 pseudo- likelihood 函数中学习权值, pseudo- likelihood 函数为:

$$p(X = x) = \prod_{l=1}^{n} p(X_l = x_l \mid MB_x(X_l)) \qquad (2)$$

其中 $MB_x(X_l)$ 是 X_l 的 Markov 篮子(blanket)的状态, 其详细定义见文献[4]. 在实验中 PSO 采用全局最佳搜索方法.

2.3　MGAPSO 算法

由于算法运行后学得的子句可能存在一些冗余结构. 为此在加入 MLN 之前要对子句化简, 以便获得简洁的子句结构. MGAPSO 算法描述如下:

(1) 数据库中存在正例, 转(2), 否则转(8);

(2) 遗传算法学习子句模板;

(3) 选取适应度最大的子句模板, 用算法 1 将其转化为子句;

(4) 若子句体中同一原子多次出现则只保留其中的一个;

(5) 若子句体中的某一原子与其它原子没有公共项则删除该原子;

(6) 若该子句未覆盖任一负例且 MLN 中不包括该子句则将其加入 MLN 中, 转(7), 否则转(1);

(7) 删除数据库中被子句覆盖的正例, 转(1);

(8) 对学得的 Markov 逻辑网用粒子群算法学得权值;

对于遗传算法结束条件的选择, 可采用连续若干代最佳适应度值不变或指定学习多少代结束. 本文试验中采用指定学习多少代作为停止条件.

3　试验对比

采用的实验数据是文献[4]中使用的 UW- CSE 数据库, 该数据库包括 23 种谓词和 1125 个常量, 常量分为 10 种类型. 谓词中包括 10 个带有 *Same* 字样的等价谓词. 抽取三个数据集, 数据集一是整个 UW- CSE 数据库; 数据集二是不包括 10 个等价谓词的 UW- CSE 数据库; 数据集三是同时满足表 1 中三条子句的 UW- CSE 数据库中的 441 个基谓词, 三条子句中 *level_500*、*faculty* 是常量. 试验平台选用的计算机配置为 2.6GHz Pentium 4 CPU, 内存为 512M.

表 **1**　UW- CSE 中的三条子句

$TaughtBy(c, p, q) \wedge CourseLevel(c, level\_500) \xrightarrow{} Prfessor(p)$
$TempAdvisedBy(p, s) \xrightarrow{} Position(s, faculty)$
$TA(c, p, q) \wedge AdvisedBy(p, s) \xrightarrow{} Student(p)$

实验一　我们将本文的结构学习算法与文献[7]中的 GILP 子句学习算法进行了对比, GILP 算法是将子句进行了二进制编码, 之后再应用遗传算法进行学习. 两种算法以覆盖子句头部的全部正实例而不覆盖任一负实例为目标, 子句头预先指定, 从数据集中学习满足条件的全部子句. 两算法在同一试验平台下, 各运行 6 次, 取最好的结果, 使用的参数与运行时间列在表 2 中.

表 **2**　GILP 与 MGAPSO 结构学习算法参数比较

算法	MGAPSO		GILP	
数据集	据数集三	数据集二	数据集三	数据集二
初始群体	16	10	16	10
选择概率	余数随机选择	余数随机选择	0.4	0.4
交叉概率	0.6	0.6	0.6	0.6
变异概率	0.3	0.3	0.1	0.1
运行时间(s)	25	1084	10	1928

实验一表明在数据集较小时 GILP 学习算法所用时间较少, 这主要是搜索的编码空间与 MGAPSO 的子句模板编码空间几乎相当, 而 MGAPSO 算法的时间主要花费在适应度与子句模板转化为子句的计算上. 当数据集加大时 GILP 的搜索空间成指数递增, 而且个体编码生长现象的存在[9]使得适应度计算量加大, 因此运行时间也随之增加, 此时本文算法的效率较高.

实验二　对数据集二进行学习. 将结构学习算法初始群

体设为 10, 子句长度最大为 4, 变异概率取 0.3, 交叉概率为 0.6, GA 停止条件设为连续学习 30 代, 适应度函数中的 k 取 3; 权值学习方法连续学习 5000 代停止. 采用文献[4]中使用的 CLL 与 AUC 度量[4]方法.

CLL 定义为公式(2)中的所有 x_l 的 $p(X_l = x_l \mid MB_x(X_l))$ 取对数的平均值. 对于一基谓词集合中的每一基谓词 i, 分配一概率 p_i, 并且规定一个类标识 c_i, 如果 i 为真 $c_i = 1$, 否则 $c_i = 0$, 在 0~1 之间选取 T 个不同的实数作为采样点, 则

$$AUC = \frac{1}{n_p}\sum_{j=1}^{n_p} f_j \qquad (3)$$

其中 $f_j = \frac{1}{T}\sum_{t=1}^{T} w_t k_j$, $w_t = \frac{1}{2}(prec_{t+1} + prec_{t-1})$, $prec_t = \frac{m_{ij}}{m_i}$; n_p 表示为真的基谓词的个数; k_j 当 $p_i \geqslant t$ 时取值为 1(t 是一采样点), 否则为 0; m_{ij} 表示所有 $p_i \geqslant t$, 并且 $c_i = 1$ 的基谓词个数; m_i 表示所有 $p_i \geqslant t$ 的基谓词个数.

算法运行 6 次取最好值, 将实验结果标记在表 3 中的 MGAPSO(1)行. 在试验中整个算法平均运行时间为 25 分钟左右.

表 3 MLN 学习方法的 CLL 与 AUC 比较结果

算法	CLL	AUC
MLN(SLS)	-0.061±0.004	0.533±0.003
MLN(SLB)	-0.088±0.005	0.472±0.004
MGAPSO(1)	-0.446±0.012	0.301±0.005
MGAPSO(2)	-0.125±0.009	0.446±0.003

实验三 采用数据集一进行学习. 将结构学习的 GA 算法初始群体设为 20, 子句长度最大为 4, 变异概率 0.3, 交叉概率 0.7, 算法连续学习 80 代结束, 适应度函数中 k 取 3; 权值学习方法连续学习 5000 代停止. 使用的 CLL 与 AUC 度量方法, 算法运行 6 次取最好值, 将实验结果标记在表 3 的 MGAPSO(2)行. 试验中整个算法平均运行时间为 1 小时 20 分钟左右.

表 3 中 MLN(SLS) 及 MLN(SLB) 是文献[4]中提出的 MLN 学习算法, MLN(SLS) 算法获得的结果精度最好. 本文算法在求解的精度上与 MLN(SLS) 接近, 实验表明 MGAPSO 的结构学习算法在加大种群和提高指定的进化代数以及适当的调整概率参数, 都有利于学得较好的结果, MLN 的精度会有所提高, 但计算时间也会相应的增加.

4 总结

本文提出谓词模板、子句模板等概念并将它们作为遗传算法的编码来学习, 设计了将子句模板转化为子句的转换算法, 以此作为一种 MLN 的结构学习算法; 对于 MLN 的权值使用粒子群算法学习. 理论分析与实验对比表明该算法可有效的学习 MLN 结构与权值. 然而该算法在转换子句模板时对应的变量标记选择比较盲目, 有待于改进. 此外, 结构学习算法存在与文献[7]相似的个体编码生长现象, 对适应度函数的改进可改善这种现象. 下一步我们将对 MGAPSO 算法进行扩展, 希望能将其作为 SRL 中自动构建模型的一种统一算法框架, 无疑这将拓展 SRL 的研究领域, 有利于进一步的应用.

参考文献:

[1] L De Raedt, K Kersting. Probabilistic logic learning[A]. ACM-SIGKDD Explorations: Special issue on Multi-Relational Data Mining[C]. New York: ACM Press, 2003. 31-48.

[2] M Richardson, P Domingos. Markov logic networks[J]. Machine Learning, 2006, 62: 107-136.

[3] P Domingos, M Richardson. Markov logic: A unifying framework for statistical relational learning[A]. In: Proceedings of the ICML-2004 Workshop on Statistical Relational Learning and its Connections to Other Fields[C]. Banff, Canada: IMLS, 2004. 49-54.

[4] S Kok, P Domingos. Learning the structure of Markov Logic Networks[A]. In: Proceedings of the Twenty-Second International Conference on Machine Learning[C]. Bonn, Germany: ACM Press, 2005, 119: 441-448.

[5] P Singla and P Domingos. Discriminative training of Markov logic networks[A]. In: Proc. AAAI-05[C]. Washington: AAAI Press, 2005. 868-873.

[6] Nils J Nilsson. Artificial Intelligence: A New Synthesis[M]. San Francisco: Morgan Kaufmann, 1998. 257-259.

[7] 杨新武, 刘椿年. 遗传归纳逻辑程序设计中规则的位串表示法[J]. 北京工业大学学报, 2001, 27(3): 297-302.
Yang Xinwu, Liu Chunnian. Bits string representation of rules in design of inducing logic programs by genetic algorithm[J]. Journal of Beijing Polytechnic University, 2001, 27(3): 297-302. (in Chinese)

[8] 杨新武, 刘椿年. 遗传归纳逻辑程序设计的个体编码生长现象[J]. 计算机研究与发展. 2003, 40(8): 1238-1243.
Yang Xinwu, Liu Chunnian. Growth phenomenon of individuals' code length in genetic inductive logic programming[J]. Journal of Computer Research and Development, 2003, 40(8): 1238-1243. (in Chinese)

[9] Po Shun Ngan, Man Leung Wong, Kwong Sak Leung, et al. Using grammar based genetic programming for data mining of medical knowledge[A]. In: John Koza eds. Proc of the 3rd Annual Genetic Programming Conf[C]. San Francisco, CA: Morgan Kaufmann, 1998. 254-259.

[10] Amund Tveit. Genetic inductive logic programming[D]. Trondheim, Norway: Dept of Computer and Information Science, Norwegian University of Science and Technology, 1997.

[11] A Tamaddoni-Nezhad, S H Muggleton. Searching the subsumption lattice by a genetic algorithm[A]. In J Cussens and A Frisch, editors, Proceedings of the 10th International Conference on Inductive Logic Programming[C]. London: Springer-Verlag, 2000. 243-252.

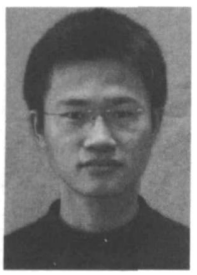
[12] 史忠植. 知识发现[M]. 北京: 清华大学出版社, 2002. 280

– 282.

[13] 陈国良, 王煦法, 庄镇泉, 王东生. 遗传算法及其应用[M]. 北京: 人民邮电出版社, 1996. 92– 97.

[14] R C Eberhart, Shi Y. Particle swarm optimization: developments, applications and resources[A]. Proceedings of the IEEE Congress on Evolutionary Computation[C]. Piscataway, NJ: IEEE Service Center, 2001. 81– 86.

作者简介:



于 鹏　男, 1979 年 5 月出生于吉林省柳河县, 吉林大学计算机科学与技术学院博士研究生, 主要研究方向: 进化算法、机器学习等. E-mail: yu_peng79@126.com

刘大有　男, 1942 年 7 月出生于河北省乐亭县, 现为吉林大学计算机科学与技术学院教授、博士生导师, 主要研究方向: 知识工程与专家系统、分布式 AI 与多 Agent 系统、不确定性推理、空间推理与 GIS 应用等.
E-mail: dyliu@jlu.edu.cn

欧阳丹彤　女, 1968 年出生于吉林省长春市, 吉林大学计算机科学与技术学院博士, 教授, 从事基于模型诊断和定理机器证明的研究. E-mail: ouyangdantong@163.com