

一种有效的多关系聚类算法

邓左祥^{1,2}, 李春贵¹

(1 广西科技大学 计算机科学与通信工程学院, 广西 柳州 545006;

2 上海交通大学 电子信息与电气工程学院, 上海 200240)

摘 要: 研究多关系数据挖掘的聚类问题, 提出一种有效的多关系聚类算法 EMC. EMC 算法的目标是提高聚类的准确率, 并且降低运行时间. EMC 算法首先利用元组 ID 传播的思想, 计算两个对象之间的相似度, 接着利用 K 中心点聚类算法, 将对象划分成簇. 实验表明, EMC 算法显著降低运行时间, 并且提高聚类的准确率.

关键词: 多关系数据挖掘; 聚类; 元组 ID 传播; 相似度; K 中心点聚类算法

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2016)04-0133-05

Efficient Multi-Relational Clustering Algorithm

DENG Zuo-xiang^{1, 2}, LI Chun-gui¹

(1 School of Computer Science and Communication Engineering, Guangxi University of Science and Technology, Liuzhou 545006, China; 2 School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: The problem of clustering in multi-relation data mining was investigated, and an efficient multi-relational clustering algorithm called EMC was proposed. EMC aims at increasing the accuracy of clustering, and decreasing running time. First, EMC computed the similarity between two objects by taking advantage of tuple ID propagation approach. Then, EMC clustered the objects by K-medoids clustering algorithm. Performance results demonstrate that, EMC significantly decreases running time, and increases the accuracy of clustering.

Key words: multi-relational data mining; clustering; tuple ID propagation; similarity; K-medoids clustering algorithm

1 引言

多关系数据挖掘, 是数据挖掘中崭新的研究热点之一^[1]. 传统的数据挖掘算法只是考虑在单表中进行挖掘, 而多关系数据挖掘研究如何直接在多张表中挖掘知识. 在现实生活中, 多关系数据挖掘有非常多的应用, 例如信用卡欺诈检测、生物学和医学数据分析、销售管理等^[2].

然而, 在多关系中进行数据挖掘, 是一个比较具有挑战性的问题. 传统的数据挖掘算法, 只适用于单表. 然而, 在实际应用中, 数据库通常以多关系的形式存在. 当面对多表时, 传统数据挖掘算法不得不把

多张表通过物理连接的方法, 合成到一张表中, 但是物理连接多表的方法一般存在效率比较低、容易造成数据冗余等缺点. 因此, 研究避免物理连接的多关系数据挖掘算法^[3], 是一个非常有意义的问题.

多关系聚类是多关系数据挖掘的研究任务之一. 多关系聚类, 就是在多关系数据库中, 基于多关系的信息, 根据对象之间的相似度, 将多关系中某个关系所代表的实体对象划分成簇的过程.

Crossclus^[4]和 LinkClus^[5]是关于多关系聚类算法的两个研究成果^[6]和传统关联规则挖掘一样, 在多关系关联规则挖掘前, 同样首先需要进行多关系频繁模式挖掘. WARMR^[7]和 FARMER^[8]是基

于 ILP 思想^[9]的两种多关系频繁模式挖掘算法.

2 元组 ID 传播

元组 ID 传播,也叫元组标识传播. 在多关系数据库中,如果关系 R_2 包含关系 R_1 的主码 k ,也就是 R_1 的主码 k 在 R_2 中作为外码,称关系 R_1 和 R_2 之间存在链接. 假如 R_1 中的元组 t_1 与 R_2 中的元组 t_2 在属性 k 上具有相同的值,称元组 t_1 与 t_2 之间存在链接. 利用多关系数据库中的链接,在多关系数据库中建立虚连接的这种思想,被元组 ID 传播所采纳. 元组 ID 传播的定义如下.

定义 1 元组 ID 传播(tuple ID propagation): 假设在多关系数据库中,每个关系的主码都是整数型,这些整数代表着每个元组的 ID(如果没有这样的主码,就人工创建一个). 假设两个关系 R_1 和 R_2 之间存在链接(假设 R_1 包含 R_2 的主码 k ,而 R_1 的主码用 $R_1.id$ 来表示),那么在 R_2 中增加一列属性 $IDset(R_1)$. 对于 R_2 的每一个元组 t ,它在 $IDset(R_1)$ 属性上的值即为 R_1 中所有与 t 存在链接的元组在 $R_1.id$ 上的值的并集,也就是元组 t 的 $IDset(R_1) = \bigcup_{u \in R_1, u.k=t.k} R_1.id(u)$, $R_1.id(u)$ 代表元组 u 在 $R_1.id$ 上的值. 也就是说, R_1 中所有元组的 ID 都传播到 R_2 与其存在链接的元组的属性 $IDset(R_1)$ 上.

下面是元组 ID 传播的一个例子,如表 1 所示. 表 1 中的多关系数据库包含有两个表:学生表和教师表. “学号”是学生表的主码,“教师号”是外码. “教师号”是教师表的主码. 学生表和教师表之间存在链接. 元组 ID 传播就是,在教师表中增加一列属性 “ $IDset(学生表)$ ”,那么学生表中与这个元组存在链接的所有元组在“学号”上的值的并集,就是教师表的某个元组在属性“ $IDset(学生表)$ ”上的值.

表 1 元组 ID 传播的例子
学生表(省略一些属性)

学号	教师号	所在系
101	001	计算机系
102	001	计算机系
103	002	计算机系
104	002	计算机系
105	003	计算机系

教师表(省略一些属性)

教师号	教师姓名	年龄	$IDset(学生表)$
001	赵老师	40	101,102
002	王老师	45	103,104
003	刘老师	50	105

3 多关系聚类算法 EMC

本节提出一种多关系聚类算法 EMC. 它对多关系中其中一个关系所代表的实体进行聚类,进行聚类的实体称为目标实体. 在 EMC 算法中,首先利用元组 ID 传播的思想,计算出目标实体中对象之间的相似度,接着根据计算出来的相似度,采用 K 中心点聚类算法,将目标实体中的对象划分成簇. 本节在最后给出 EMC 算法的伪代码.

3.1 相似度计算

本小节重点介绍在多关系中,计算目标实体中两个对象之间相似度的方法.

对于只包含一个关系的单关系数据库,它只有一个实体,在计算实体对象之间的相似度时,可以通过实体自身的属性来计算. 例如,一个单关系“学生表”,“学生”是实体,包括性别、年龄、身高、体重、成绩、专业等属性. 通过这些属性,就可以计算学生之间的相似度. 在多关系数据库中,存在着多种实体,以及实体之间的联系. 在计算目标实体对象之间的相似度时,除了借助于实体自身的属性,还可以利用存在于多关系上的实体之间的联系. 通过实体之间的联系,找到相关元组,借助于相关元组的属性,来计算相似度. 实体之间的联系是很有用的信息,如果充分利用它,不但可以提高相似度计算的准确率,而且还可以进一步提高聚类的质量.

下面描述借助元组 ID 传播,寻找相关元组的方法. 假设目标实体是 R_1 ,要找到与实体关系 R_1 的元组 t_1 相关的元组. 与 t_1 相关的元组,或者在与 R_1 存在链接的实体关系上,或者在与 R_1 存在链接的联系关系所产生的另一个实体关系上.

假如与 R_1 存在链接的关系是实体关系(实体之间的联系是 1:1 或 1:N 的情况),假设这个实体关系为 R_2 ,目的是在 R_2 上找出与 t_1 相关的元组. 在 R_1 与 R_2 之间进行元组 ID 传播. 元组 ID 传播可以分为两种情况:如果是 R_1 的元组 ID 传播到 R_2 ,那么可以通过 R_1 上 t_1 元组的外码,找到 R_2 上与 t_1 相关的元组;如果是 R_2 的元组 ID 传播到 R_1 ,那么可以通过 R_1 上 t_1 元组的 $IDset(R_2)$,找到 R_2 上与 t_1 相关的元组.

假如与 R_1 存在链接的关系是联系关系(实体之间的联系是 M:N 的情况),假设这个联系关系为 R_2 ,由这个联系所转换成的另一个实体关系为 R_3 . 现在要在 R_3 上找出与 t_1 相关的元组. 首先,在 R_1 与 R_2 ,以及 R_2 与 R_3 之间进行元组 ID 传播. 接着,提取 R_1 上 t_1 元组的 $IDset(R_2)$ 值. 最后,根据 R_3 上的

IDset(R_2),找出拥有对应值的元组,这些元组就是与 t_1 相关的元组.

例如,在如表 2 所示的包含“学生表”,“选课表”,“课程表”的多关系数据库中,“学生表”和“课程表”是实体转化成的关系,代表实体“学生”和“课程”,它们之间的联系是 $M:N$ 的情况,其中学号和课程号分别是主码,“选课表”是联系转化成的关系.如果要找出与“学生表”中学号为“001”的元组相关的元组.那么,首先通过“学生表”中的“IDset(选课表)”,得到“001”元组的“IDset(选课表)”上的值为“1”和“2”,接着在“课程表”中寻找在“IDset(选课表)”上拥有“1”和“2”的元组.因为在“课程表”中,课程号为“1”的元组在“IDset(选课表)”上拥有“1”,课程号为“2”的元组在“IDset(选课表)”上拥有“2”,所以“课程表”中的课程号为“1”和“2”的元组,就是与“学生表”中学号为“001”的元组相关的元组.

表 2 在多关系中寻找相关元组
学生表(省略一些属性)

学号	IDset(选课表)
001	1, 2
002	3, 4

选课表(选课表. id 作为主码,人工创建)			
选课表. id	学号	课程号	成绩
1	001	1	80
2	001	2	75
3	002	2	85
4	002	3	80

课程表(省略一些属性)	
课程号	IDset(选课表)
1	1
2	2, 3
3	4

找出相关元组后,就可以计算元组之间的相似度.下面描述计算相似度的方法.假设现在要计算的是 R_1 关系上两个元组 t_1 和 t_2 之间的相似度 $s(t_1, t_2)$.

计算元组 t_1 和 t_2 之间相似度的步骤可以分为三步.第一步,根据 t_1 和 t_2 自身的属性,计算出它们之间的相似度 s_1 .第二步,假设在关系 $R_2, R_3, \dots, R_n$

上,都存在分别与 t_1 和 t_2 相关的元组.在 $R_2, R_3, \dots, R_n$ 中的每个关系上,都分别找出与 t_1 和 t_2 相关的元组,并且分别对每个关系上找出的与 t_1 相关的元组 v_1 ,以及与 t_2 相关的元组 v_2 ,根据它们自身属性,计算出 v_1 和 v_2 之间的相似度.假设关系 $R_2, R_3, \dots, R_n$ 上对应计算出来的相似度分别为 $s_2, s_3, \dots, s_n$.第三步,取 $s_1, s_2, \dots, s_n$ 的平均值,就可以得到元组 t_1 和 t_2 之间的相似度 $s(t_1, t_2)$.

在一个关系中,比如 R_2 ,有可能存在多个元组与 t_1 相关,多个元组与 t_2 相关.在这种情况下, s_2 的值就是, R_2 上与 t_1 相关的所有元组,以及与 t_2 相关的所有元组,它们之间相似度的平均值.例如,假设在关系 R_2 上,与 t_1 相关的元组有 m 个,分别是 $a_1, a_2, \dots, a_m$,与 t_2 相关的元组有 n 个,分别是 $b_1, b_2, \dots, b_n$,则 s_2 按照下式来计算:

$$s_2 = \frac{\sum_{i=1}^m \sum_{j=1}^n s(a_i, b_j)}{m \times n}.$$

(1)

接下来重点要解决的一个问题,就是如何根据元组自身的属性,计算两个元组之间的相似度.属性类型有很多种.针对不同类型的属性,计算相似度的方法是不一样的.区间标度属性、二元属性、分类属性、序数型属性等是常见的属性类型.它们计算相似度的方法由文献[4]给出.

3.2 根据相似度聚类

计算出目标实体的对象之间的相似度后,就可以根据相似度,对目标实体的对象进行聚类.

本文提出的 EMC 算法,采用 K 中心点聚类算法,将对象划分成簇.假设需要将对象划分成 k 个簇. K 中心点聚类算法的过程可以分为以下两步:第一步,随机地选择 k 个代表对象,每个代表对象代表一个簇的参照点. 剩余的对象根据其 与每个代表对象的相似度,分配给相似度最小的代表对象所代表的簇;第二步,反复用非代表对象来代替代表对象,以优化聚类质量,直到聚类质量收敛. 在一个代表对象被某个非代表对象代替后,除了未被代替的代表对象外,其余各点根据相似度,重新分配给相似度最小的代表对象所代表的簇. 重新分配各点之后,如果聚类质量变好,就用这个非代表对象替换掉代表对象;如果聚类质量没有变好,就仍然保持原来的代表对象不变.

在算法的第二步中,聚类质量用一个代价函数来表示.代价函数的定义是,所有对象与其对应代表对象之间的相似度之和.最小化代价函数是优化聚

类质量的目标. 特别的, 用 $C_1, C_2, \dots, C_k$ 代表 k 个簇, 用 $r_1, r_2, \dots, r_k$ 分别代表这 k 个簇的其中一个簇的代表对象, 用 o 代表一个对象, 则代价函数 P 按照下式来计算:

$$P = \sum_{i=1}^k \sum_{o \in C_i} s(o, r_i). \tag{2}$$

3.3 EMC 算法的伪代码

算法: EMC, 一种有效的多关系聚类算法.

输入: 一个多关系数据库 D , 并指定一个目标实体, 簇的数目 k .

输出: 将目标实体划分成的 k 个簇的集合.

方法:

- (1) 为目标实体的每个元组寻找相关元组;
- (2) 计算目标实体对象之间的相似度;
- (3) 从目标实体中, 随机选择 k 个代表对象, 分别代表一个簇的参照点;
- (4) repeat
- (5) 根据相似度, 将每个剩余对象分配给相似度最小的代表对象所代表的簇;
- (6) 随机选择一个非代表对象 O_{random} ;
- (7) 用 O_{random} 代替一个代表对象 O_j , 分别计算代替前的代价函数 P_j , 以及代替后的代价函数 P_{random} ;
- (8) if $P_{\text{random}} < P_j$, then 用 O_{random} 替换掉 O_j , 形成新的 k 个代表对象;
- (9) until k 个代表对象不再发生变化

4 实验

4.1 实验方法和参数设置

在实验中, 将本文提出的 EMC 算法与传统算法、LinkClus 进行比较. 在传统算法中, 首先将多关系通过物理连接合成到一张表中, 接着计算元组之间的相似度, 最后采用 K 中心点聚类算法进行聚类. 选择聚类的准确率, 以及运行时间, 也就是聚类所花的时间, 作为两个性能评价指标.

选择 DBLP 数据库^[10]来作为实验的数据集. 提取 DBLP 中的 XML 数据, 形成一个多关系数据库. 这个多关系数据库总共有 7 个关系, 其中 5 个是实体关系, 它们分别是 authors、conferences、keywords、proceedings、publications, 2 个是联系关系, 它们分别是 has-keyword、publishers. 在实验中, 选择 publications 关系作为聚类的目标实体, 将它划分成 10 个簇. 另外, 表的个数从 4 个变化到 7 个.

4.2 实验结果与分析

图 1 显示本文提出的 EMC 算法和其他两种算

法关于聚类准确率的比较结果. 从图 1 可以看出, 传统算法的准确率最高, 这是由于传统算法通过物理连接各个表, 可以更加精确地计算出对象之间的相似度, 进而提高聚类准确率. 此外, 从图 1 还可以看出, EMC 算法和传统算法之间的准确率相差不大, 要远远优于 LinkClus 算法. 这是因为在计算对象之间的相似度时, EMC 算法不但利用实体之间的联系, 而且还利用对象自身的属性, 而 LinkClus 算法仅仅利用实体之间的联系, 因此跟 LinkClus 算法相比较, EMC 算法可以得到对象之间更加精确的相似度.

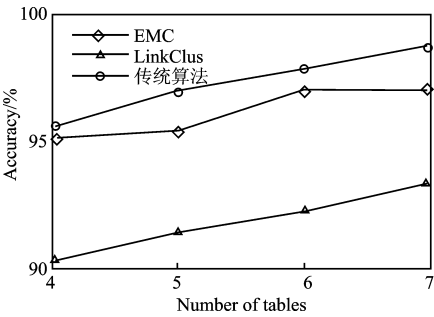


图 1 准确率比较结果

图 2 显示本文提出的 EMC 算法和其他两种算法关于运行时间的比较结果. 从图 2 可以看出, LinkClus 算法的运行时间最少, EMC 算法的运行时间次之, 和 LinkClus 算法的运行时间相差不大, 而传统算法的运行时间非常长, 远远高于 LinkClus 算法和 EMC 算法. 这是因为传统算法在聚类前, 需要物理连接各个表, 这是非常耗时的.

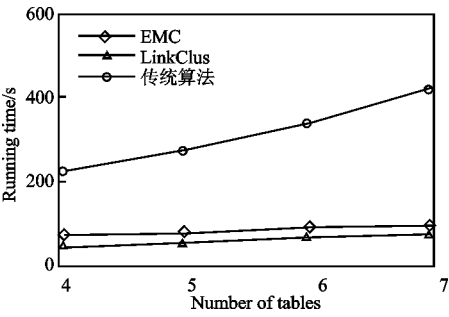


图 2 运行时间比较结果

当表的个数为 7 时, 让 EMC 算法运行 30 次, 得出各次的运行时间. 图 3 显示各次运行时间之间差异的累积分布函数. 从图 3 可以看出, 大部分运行时间的差异小于 1.5 s, 所有运行时间之间的差异都小于 2.5 s. 这可以说明 EMC 算法的运行时间是稳定的.

根据实验结果, 可以做出以下两点总结: (1) 虽然本文提出的 EMC 算法在运行时间上不如 Link-

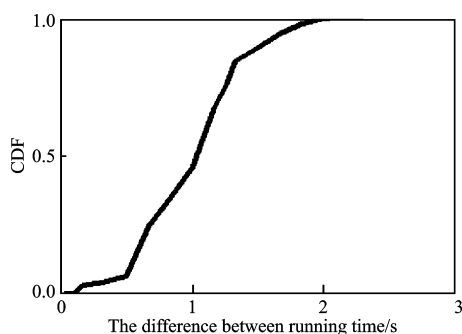


图3 不同运行时间之间差异的累积分布函数

Clus 算法,但是 EMC 算法的聚类准确率要远远优于 LinkClus 算法,已经接近传统算法的聚类准确率;(2)跟传统算法相比较,EMC 算法极大减少运行时间。综合这两点总结,可以得到,EMC 算法实现较高的聚类准确率,并且极大地减少聚类所花的时间,是一种比较有效的多关系聚类算法。

5 结束语

本文关注于数据挖掘中的多关系聚类问题,提出一种有效的多关系聚类算法 EMC。在聚类时,EMC 首先计算目标实体的对象之间的相似度,接着根据相似度,采用 K 中心点聚类算法,将对象划分成簇。为了提高聚类质量,在计算相似度时,EMC 利用元组 ID 传播的思想,不但根据对象自身的属性,而且根据不同实体对象之间的联系。基于真实数据的实验表明,EMC 实现较高的聚类准确率,并且极大地减少运行时间。

参考文献:

- [1] Ling P, Rong X. Double-phase locality sensitive hashing of neighborhood development for multi-relational[C]// Proceedings of 2014 UK Workshop on Computational Intelligence, Hong Kong; UKCI, 2014: 206-213.
- [2] Houshmand M, Alishahi M. Improve the classification and Sales management of products using multi-relational data mining[C]// IEEE 3rd International Con-

ference on Communication Software and Networks. Xi'an: IEEE, 2011: 329-337.

- [3] Vaghela V, Vandra K, Moni N. MR-MNBC: maxrel based feature selection for the multi-relational naïve bayesian classifier[C]// Nirma University International Conference on Engineering. Ahmedabad: NUICONE, 2013: 1-9.
- [4] Yin X, Han J, Yu P. Cross-relational clustering with user's guidance[C]// Proceedings of the 11th ACM SIGKDD Conference on Knowledge Discovery in Data Mining. Chicago: ACM, 2005:344-353.
- [5] Yin X, Han J, Yu P. LinkClus: efficient clustering via heterogeneous semantic links[C]// Proceedings of International Conference on Very Large Data Bases. Seoul: ACM, 2006: 427-438.
- [6] Han J, Kamber M. Data mining: concepts and techniques [M]. FAN M, MENG X, translated. 2nd ed. Beijing: China Machine Press, 2007: 373-382.
- [7] Dehaspe L, Raedt L. Mining association rules in multiple relations[C]// Proceedings of the 7th International Workshop on Inductive Logic Programming. Berlin: Springer, 1997: 125-132.
- [8] Nijssen S, Kok J. Faster association rules for multiple relations[C]// Proceedings of 17th International Joint Conference on Artificial Intelligence. Adelaide: ACM, 2001: 891-896.
- [9] Lavrac N, Dzeroski S. Inductive logic programming: techniques and applications [M]. New York: Ellis Horwood, 1994.
- [10] Ley M. The DBLP computer science bibliography[DB/OL]. [2015-04-20](2014-07-22). <http://www.informatik.uni-trier.de/~ley/db/>.

作者简介:

邓左祥 男,(1983-),博士.研究方向为数据挖掘、自组织网络. E-mail:dengzuoxiang@126.com.

李春贵 男,(1968-),博士,教授.研究方向为数据挖掘、人工智能.