基于 Top- K 项频繁模式挖掘的研究及实现*

胡 燕 韩瑞雪
(武汉理工大学计算机科学与技术学院 武汉 430070)

摘 要 频繁模式挖掘是关联规则、序列分析等数据挖掘任务的关键步骤,我们知道,当给定的最小支持度阈值非常小,将产生大量的频繁模式,反之,可能产生很少的模式或根本没有结果。用户有时仅对其中的部分项的频繁度感兴趣,这属于部分频繁模式挖掘问题。文章通过有效设置挖掘区间,讨论一种 top - k 项频繁模式挖掘问题,进而扩展到连续区间上的情况,最后将给出实验结果。

关键词 频繁模式 频繁项集 数据挖掘
中图分类号 TP311.13

Research and Implementation of Top-k Frequent Patterns Mining

Hu Yan Han Ruixue
(School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430070)

Abstract Frequent pattern mining is a key step for many tasks of data mining such as association rules mining, series analysis and so on. As we know, when the minimum support threshold is low, we may acquire large numbers of frequent patterns, Whereas we may get few or even no frequent pattern. Sometimes users are only interested in frequency about part of all items, which falls into the problem of mining part of frequent patterns. In this paper, we discuss a kind of top k frequent patterns mining and then extend it to the situation in a sequential range. Finally, we present our experiment result.

Key words frequent patterns, frequent items, data mining, Top - k
Class Number TP311.13

1 引言

数据挖掘(Data Mining)是数据库技术与人工智能技术相结合的产物,是一门新兴的数据分析技术,频繁模式挖掘是数据挖掘中的一项基础性的工作,是近十几年来数据挖掘领域的热点问题。自 Agrawal 等人提出关联规则发现算法以来,该问题得到了众多学者的广泛研究,人们尝试着从各种方法出发来解决频繁模式挖掘的效率问题,提出了许多算法,并且不断地有新的研究成果产生。本文首先简要介绍频繁模式挖掘的基本概念,在此基础上讨论一些目前常见频繁模式挖掘算法,本文重点介绍对频繁模式挖掘的一种新的应用 - 对 I 中的 top k 项进行频繁模式挖掘,最后给出实验结果分析。

2 频繁模式挖掘研究基础

设 $I=\{I_1, I_2, \cdots, I_m\}$ 为 m 个项的集合,事务数据库 $DB=\langle T_1, T_2, \cdots, T_n\rangle$ 为 n 个事务组成的事务集,其中事务 T_i 是 I 的子集,即 $T_i \subseteq I$,事务可以由事务 ID 或事务本身所包含的项集来表示,如 T_i 或 $\{I_1, I_2, I_3\}$。不同项目组成的集合称为项集或模式。项集的长度是指它所包含的项的个数,长度为 k 的项集称为 k - 项集。

项集 X 的支持度 support(X) 是指 DB 中包含 X 的事务所占的比例。对于给定的最小支持度阈值 ξ,如果 support$(X) \geq \xi$,那么 X 是频繁的,否则 X 是非频繁的。所谓的频繁模式挖掘问题就是要在 DB 中找出所有的频繁项集。

* 收稿日期:2009 年 1 月 6 日,修回日期:2009 年 1 月 14 日
作者简介:胡燕,女,教授,研究方向:人工智能,数据挖掘。韩瑞雪,女,硕士研究生,研究方向:数据挖掘。

频繁模式的是关联规则挖掘的一个关键步骤,也可以应用到分类、聚类等数据挖掘任务中,为了发现大型超市中顾客购买行为之间隐含关系,Agrawal 等人于 1994 年创造性地提出了关联规则挖掘问题,并给出了著名的 Apriori[1] 算法,由挖掘出的频繁项集产生关联规则,从而发现顾客购买的不同商品之间的关联,分析顾客的购买习惯,以便于商家做出有价值的营销策略。

我们知道在频繁模式挖掘过程中,如果最小支持度阈值非常小,那么模式挖掘算法将产生大量的模式,反之,当最小支持度阈值过大时,可能产生很少的模式或根本没有结果。我们很难根据自定义的最小支持度阈值预测输出模式的多少,这时提出了 top-k 模式挖掘问题。

第一个 top-k 模式挖掘算法是 A,W-C,Fu 等人提出的 Itemset-Loop[2],该算法可以挖掘出最多 k 个长度小于用户给定的值 m。文献[3]中提到的 LOOPBACKT 和 BOMO 是基于 FP-tree 的 top-k 模式挖掘算法,它使用与 Itemset-Loop 一样的估计机制,并且实验显示 LOOPBACKT 和 BOMO 的性能优于 Itemset-Loop[3]。TFP 算法[4] 也是基于 FP-tree 的,它用来挖掘 top-k 个长度大于用户给定值 min_l 的频繁闭合项集。TSP[5] 是第一个用来挖掘 top-k 个闭合连续模式,其长度不小于用户给定的最小长度 min_l。本文在现有研究的基础上提出关于 top-k 模式挖掘的一种新的应用,即找出 Top-k 项相关的频繁项集。例如,从图 1 所示的事务数据库 DB 的数据表中可以看出,对于项集 $I=\{I_1, I_2, \cdots, I_m\}$ 来说,每条事务中的项大部分分布在 I 的前 9 项中,对于 I_9 之后的项则很少出现,仅 I_{14} 在第 5 个事务中出现过一次,这时我们可能更关心前 9 项的频繁信息,那么我们可以仅需要完成前 9 项相关的挖掘任务。下面将具体介绍这一应用。

3 基于 Top-K 项频繁模式挖掘改进算法

经典 Apriori 算法[1] 采用层次搜索策略进行频繁模式挖掘:令 F_k 为频繁 k 项集,C_k 为候选 k 项集,算法对数据库进行多次扫描,每次扫描进行如下两步操作:1) 通过第 $k-1$ 次遍历得到的 F_{k-1} 生成 C_k。2) 对于每个事务,确定它包含 C_k 中的哪些候选模式,并计算支持数,扫描结束后,检查候选项集 C_k 中哪些是频繁的,从而构成 F_k。该算法执行直到 F_k 为空时为止。

图 1 事务数据库中的数据表

近年来,研究人员还提出了许多频繁模式挖掘的改进方法,如 Zaki 等人提出的 Eclat[6] 算法采用了纵向(vertical)数据表示方法,并通过网格(lattice)和项集聚簇技术来挖掘频繁项集。J. Han 提出了一种基于频繁模式树(FP-tree)且不产生候选项集的频繁模式挖掘算法 FP-growth[7]。与 Apriori 算法相比,该算法具有以下特点:1) 使用 FP-tree 存储数据库中的所有频繁项信息。FP-tree 的构建只需扫描两次数据库,避免了大量的 I/O 操作。2) 不需要产生候选项集,从而减少了由于产生和检测候选项集所带来的开销。3) 采用分而治之的方式将查找长频繁模式的问题转化为先找出较短的频繁模式后将其合并。在 FP-growth 算法中,绝大部分时间都消耗在 FP-tree 及条件 FP-tree 的构造与遍历上,后来的研究人员已针对这一方面的问题做了大量的改进工作,提高了挖掘效率。

本文所讨论的找出 Top k 项相关的频繁项集,它主要针对 $I=\{I_1, I_2, \cdots, I_m\}$ 的前 k 项进行频繁模式挖掘,对于区间 $(k, m]$ 中的各项则不再考虑。这个应用可扩展为当 m 的值非常大的时,对 I 的任意区间上的频繁项集进行挖掘。此问题研究的价值在于,当给定的最小支持阈值 ξ 非常小时,执行挖掘算法后产生不同长度的频繁项集可能非常多,或者当 m 值非常大时,我们可能仅对其中的前几项或某一区间段上的项集是否频繁感兴趣,例如,超市中的商品 I 按其类别食品-a、服饰-b、工具-c 等有序,同时每种商品也有其唯一的 ID,$I=\{a_1, a_2, \cdots, a_x, b_1, b_2, \cdots, b_y, c_1, c_2, \cdots, c_z\}$,此时商家关注 b 类产品的频繁度,就可以将挖掘范围设置为 $[x+1, x+y]$。如果商家对于前 k 个商品感兴趣,可以将范围设为 $[1, k]$。分析以上两种情况后,我们可以看出本应用可以解决 I 中连续项相关的频繁模式挖掘问题,而对于间断的项,尤其是常用的具有应用价值的间断的类别之间的频繁项的挖掘问题该如何解决呢?解决这个问题,一种方法是改变相关类别商品在 I 中的位置,它涉及到大量的移动操作,开销较大,不建议使用。另一种是为 I 建立类别索引,这样关注间断类别中的频繁模式,就可以通过调整类别索引表中相关的位置,得到新的所关注类别连续了的 I 用于接下来的挖掘过程。

结合以上讨论,我们知道,对 I 中的所有项进行完全的挖掘有时不利于对挖掘结果的分析与利用,本文依据 Apriori 算法的挖掘思想,来解决 Top k 项相关的频繁项集挖掘问题,获取用户感兴趣的频繁信息。

由于我们关注于 I 中某一连续区间内各项之间的频繁性,所以可以直接使用它们来对长度为 1 的频繁项集进行初始化。这样将区间之外的所有项排除,挖掘过程中不考虑他们是否频繁。

```
// 初始化候选项目集
    for (int nInitCount = 0; nInitCount < nItemCount;
nInitCount + +)
    {
        CandLargeItem[0][nInitCount] = I[nItemInit
+ strInit];
    }
```

这里的 nItemCount 表示 I 的某个连续区间内各种不同项目个数,取值为 nItemCount = nItemEnd-nItemInit + 1,其中 nItemInit, nItemEnd 分别存放"参数设置"(见图 2)弹出对话框中所设置的范围边界值,接下来的挖掘过程采用 Apriori 算法的设计思想来实现。

输入:事务数据库 D,最小支持数 minsup(minsup = ξ * | D| ,| D| 为 D 中所包含的事务数)。

输出:D 中 Top k 项相关的频繁项集 F

```
F1 = {frequent 1-itemsets} //根据候选 1 - 项集,得到频繁
1 - 项集
for(k = 2; Fk-1 ≠Φ;k + +){
    Ck = Candidate_gen(Fk-1);
    for each transaction t ∈D {
        Ct = subset(Ck,t);// Ct 为 t 中所包含的存在于
Ck 中的候选项集
        for each candidate c ∈Ct
            c.count + +;
    }
    Fk = { c ∈Ck| c.count ≥minsup}
 }
Return F = ∪k Fk

Procedure Candidate_gen(Fk-1){
    For each itemset f1 ∈Fk-1
    For each itemset f2 ∈Fk-1{
        if((f1[1] = f2[1]) and (f1[2] = f2[2]) and …(f1
[k - 2] = f2[k - 2]) and (f1[k - 1] = <f2[k - 1])){
            c = f1[1] f1[2] …f1[k - 1] f2[k - 1];
            add c to Ck;
    }
    }
    For each c ∈Ck
    For each (k - 1) - subset s of c
        If (s ∉Fk-1)
            Delete c;
    Return Ck;
}
```

4 实验分析

本实验环境为 Inter Pentium 4 CPU 2.8GHz,内存 1G, Windows XP 操作系统, 程序用 C + + 编写,在 Visual C + + 6.0 中编译执行。本程序的主界面如图 2 所示。通过"参数设置"来确定挖掘区间及最小支持度阈值,因为挖掘过程在指定的区间上执行,则相对于完整的频繁模式挖掘过程来说,它的搜索空间减小,所以能在较短的时间内获得用户感兴趣的频繁信息,同时从视觉上方便用户对挖掘结果的分析与利用。

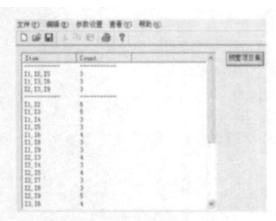

图 2　程序主界面

5 结语

本文以现有的频繁模式挖掘算法为基础,提出了一种新的关于频繁模式挖掘的应用,实验表明将更加灵活地运用挖掘算法满足常用需求。本应用还可以采用更加高效的频繁模式挖掘算法来实现,如 FP - growth 及其改进算法,以提高它在大量数据环境下的执行效率。对于现实世界中大量不同类型的数据,目前的频繁模式挖掘算法的执行效率也存在或多或少的差异,是否可以进一步优化,如何应用于实践,将作为进一步的研究内容。

参 考 文 献

[1]R Agrawal and R Srikant. Fast algorithms for mining association rules[C]. Proceedings of the 20th International Conference on Very Large Data Bases, Morgan Kaufmann, 1994:487～499

[2]A. W. - C. Fu, R. W. - W. Kwong, J. Tang. Mining N - most Interesting Itemsets[C]. in Proc. of 12th International Symposium on Methodologies for Intelligent Systems (ISMIS00), 2000,10

[3]Y.L. Cheung, A. W. - C. Fu. Mining Association Rules without Support Threshold: with and without Item Constraints[J]. IEEE Transactions on Knowledge and Data Engineering,2004,9,16(9):1052～1069

[4]O. Nasraoui , C. Cardona , C. Rojas , and F. Gonzalez. TECNO - STREAMS: Tracking Evolving Clusters in Noisy Data Streams with a Scalable Immune System Learning Model[C]. in Proc. of 3rd IEEE International Conference on Data Mining (ICKM'03) ,2003 :235～242

[5]O. Nasraoui , C. Cardona , C. Rojas , F. Gonzalez. Mining Evolving User Profiles in Noisy Web Clickstream Data with a Scalable Immune System Clustering Algorithm [C]. in Proc. of WebKDD 2003 - KDD Workshop on Web mining as a Premise to Effective and Intelligent Web Applications ,2003 :71～81

[6]Zaki M , Parthasarathy S , Ogihara M , Li W. New algorithms for fast discovery of association rules[C]. In: D Heckerman , et al eds. Proc of the Third Intl. Conf. on Knowledge Discovery and Data Mining (KDD'97). AAAI Press ,1997 :283

[7]J. Han , J. Pei , Y. Yin. Mining frequent patterns without candidate generation[C]. In Proceeding of Special Insterest Group on Management of Date , Dallas , TX , 2000 ,5 :1～12

(上接第 9 页)

6 个隐节点和 1 个输出节点。为了实现基于粒子群神经网络的移动机器人门牌号码识别,验证本文所提出方法的有效性,利用轮式机器人平台在实际室内环境中完成了实验,结果如下图所示。

图 3 位于门牌左侧位置的识别结果

图 4 位于门牌右侧位置的识别结果

图 5 光亮的环境下的识别结果

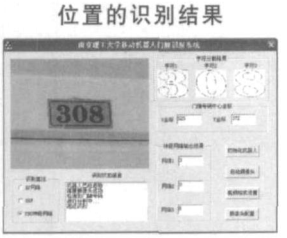

图 6 昏暗的环境下的识别结果

以上四次实验耗时分别为: 114. 811ms、123. 345ms、120. 780ms、121. 493ms , 平均耗时 121. 286 ms。由实验结果可知,移动机器人在不同的位置与不同的光照环境下均能准确地完成字符的识别,并满足实时性要求,验证了本文所提算法的有效性。

5 结语

本文所提出的基于粗分类与细分类相结合的门牌号码识别方法,充分利用了粗分类的正确分类率、分类稳定性和分类快速性与细分类的鲁棒性,大大提高了门牌号码识别的识别率,而且识别速度快。实验表明,该方法有效地应用于移动机器人门牌号码识别具有宽广的应用前景。

参考文献

[1]王兴玲. 最大类间方差车牌字符分割的模板匹配算法[J]. 计算机工程 ,2006 ,32(19) :193～195

[2]焦娜 , 迟呈英 , 苗夺谦. 基于软 K 段主曲线算法的字符特征提取研究及实现[J], 计算机科学 , 2006 ,33(1) : 229～231

[3]郭招球 ,赵跃龙 ,高敬欣. 基于小波和神经网络的车牌字符识别新方法[J]. 计算机测量与控制 , 2006 ,14 (9) :1257～1259

[4]Kennedy J , Eberhart R C. Particle swarm optimization[C]. IEEE International Conference on neural networks , Piscataway , NJ , 1995 ,4 : 1942～1948

[5]Y. Shi , R. C. Eberhart. A modified particle swarm optimizer[C]. Proceedings of the International Joint Conference on Evolutionary Computation , 1998 :69～73

[6]江涛 ,张玉芳. 一种改进的粒子群算法在 BP 网络中的应用研究[J]. 计算机科学 , 2006 ,33(9) :164～165

[7]Liang Peng , Haiyun Liu. Decision-making and simulation in multi-agent robot system based on PSO-neural network[C]. 2007 IEEE International Conference on Robotics and Bionomics , 2008 : 1763～1768

[8]Hu M K. Visual pattern recognition by moment invariants[J]. IEEE Transaction on Information Theory , 1962 ,11(8) :179～187