Journal of Computer Applications
计算机应用,2013,33(11):3045-3048,3089
ISSN 1001-9081
CODEN JYIIDU
2013-11-01
http://www.joca.cn

文章编号:1001-9081(2013)11-3045-04
doi:10.11772/j.issn.1001-9081.2013.11.3045

基于散列的频繁项集分组算法

王红梅[1,2],胡　明[2*]
(1.吉林大学 计算机科学与技术学院,长春 130012;　2.长春工业大学 计算机科学与工程学院,长春 130012)
(*通信作者电子邮箱 huming@mail.ccut.edu.cn)

摘　要:Apriori 算法是频繁项集挖掘的经典算法。针对 Apriori 算法的剪枝操作和多次扫描数据集的缺点,提出了基于散列的频繁项集分组(HFG)算法。证明了2-项集剪枝性质,采用散列技术存储频繁2-项集,将 Apriori 算法剪枝操作的时间复杂度从 $O(k\times|L_k|)$ 降低到 $O(1)$;定义了首项的子项集概念,将数据集划分为以 I_i 为首项的数据子集并采用分组索引表存储,在求以 I_i 为首项的频繁项集时,只扫描以 I_i 为首项的数据子集,减少了对数据集扫描的时间代价。实验结果表明,由于 HFG 算法的剪枝操作产生了累积效益,以及分组扫描排除了无效的项集和元组,使得 HFG 算法在时间性能方面与 Apriori 算法相比有较大提高。

关键词:频繁项集;2-项集剪枝;散列表;首项分组;索引表
中图分类号:TP311　　**文献标志码**:A

Frequent itemsets grouping algorithm based on Hash list

WANG Hongmei[1,2], HU Ming[2*]
(1. *College of Computer Science and Technology, Jilin University, Changchun Jilin* 130012, *China*;
2. *School of Computer Science and Engineering, Changchun University of Technology, Changchun Jilin* 130012, *China*)

Abstract: Apriori algorithm is a classic algorithm in frequent itemsets mining. In view of the shortcomings of pruning operations and multiply scanning data set in Apriori, a Hash-based Frequent itemsets Grouping algorithm, named HFG was put forward. In this paper, 2-length itemsets pruning property was proved, frequent 2-length itemsets were stored based on Hash list, the time complexity of Apriori algorithm in pruning operation was dropped from $O(k\times|L_k|)$ to $O(1)$; the concept of sub-itemset of first term was defined, dataset was divided into subsets with I_i as first item and stored by grouping index list. Therefore, only the sub data set with I_i as the first item was scanned to find the frequent itemsets, and the time cost of scanning dataset was reduced. The experimental results show that the HFG algorithm is much more efficient than Apriori algorithm in time performance because of the cumulative benefits of pruning operations and skipping the invalid itemsets and records in HFG algorithm.

Key words: frequent itemset; 2-length itemset pruning; Hash list; first term grouping; index list

0　引言

随着频繁项集挖掘应用领域的扩展,吸引了很多学者加入研究,提出了许多频繁项集挖掘算法,其中,美国学者 Agrawal 等[1]提出的 Apriori 算法是一个里程碑,其基本原理是用频繁 k-项集找出候选频繁 $(k+1)$-项集,再扫描数据集得到频繁 $(k+1)$-项集及其支持度。Apriori 算法的缺点是产生大量候选频繁项集,并且需要多次扫描数据集。针对 Apriori 算法的缺点,很多学者对 Apriori 算法进行了改进研究,如采用 FP-树(Frequent-Pattern tree)存储数据集[2-4]、采用垂直格式存储数据集[5-6]、采用散列表存储候选频繁项集[7]、分段计算支持度[6,8-10]、不产生候选项集[2-3]等。近年来,有学者提出基于概念格的挖掘算法[11]、基于滑动窗口的挖掘算法[12]等。

本文针对 Apriori 算法的剪枝操作和多次扫描数据集的缺点,证明了非频繁 2-项集剪枝性质,采用散列表存储频繁 2-项集,在 $O(1)$ 时间完成了与 Apriori 算法同样的剪枝操作;定义了首项的子项集概念,将数据集按首项进行分组,在求以 I_i 为首项的频繁项集时,只扫描以 I_i 为首项的数据子集,从而减小了对数据集扫描的时间代价;在此基础上,提出了基于散列的频繁项集分组(Hash-based Frequent itemsets Grouping, HFG)算法。相比其他频繁项集挖掘算法,HFG 算法没有复杂的理论推导,容易理解和实现。实验结果表明,HFG 算法在时间性能方面与 Apriori 算法相比有较大提高。

1　相关工作

1.1　问题描述

问题描述[1,13]如下:设 $I=\{I_1,I_2,\cdots,I_m\}$ 是所有项的集合,非空项集中包含项的个数称为项集 A 的长度,长度为 k 的项集记为 k-项集。设 $T=\{T_1,T_2,\cdots,T_n\}$ 是相关事务的数据集合,其中每个事务由 TID 标识,且 $1\leqslant i\leqslant n$。

设 A 是一个项集,数据集 T 包含项集 A 当且仅当 $1\leqslant i\leqslant n$,数据集 T 中包含项集 A 的事务的个数,称为项集 A 在数据集 T 中的支持度,记为 $sup\_count(A)$。给定支持度阈值 min_sup,如果项集 A 满足 $sup\_count(A)\geqslant min\_sup$,则称项集 A 为数据集 T 的一个频繁项集。频繁项集挖掘的任务就是在数据集 T

收稿日期:2013-05-28;**修回日期**:2013-05-19。
基金项目:国家自然科学基金资助项目(61133011);吉林省自然科学基金资助项目(20101525)。
作者简介:王红梅(1968-),女,吉林长春人,教授,博士研究生,主要研究方向:数据挖掘、智能算法;　胡明(1963-),男,江苏句容人,教授,主要研究方向:数据挖掘、分布式计算。

《计算机应用》Journal of Computer Applications

中找出所有支持度不小于给定阈值 min_sup 的频繁项集。

1.2 Apriori 算法

Apriori 算法采用逐层搜索的迭代方法,具体过程是:扫描数据集找出所有频繁 1- 项集(记为 L_1);将 L_1 进行自身连接,产生所有候选频繁 2- 项集(记为 C_2),再扫描数据集得到 L_2;将 L_2 进行连接,产生 C_3,再扫描数据集得到 L_3;依此下去,直至不能找到频繁项集为止。

在 Apriori 算法的连接步,对于两个频繁 k- 项集 A 和 B,假定数据集中的项集按字典排序,如果 $A[1]=B[1]\wedge A[2]=B[2]\wedge\cdots\wedge A[k-1]=B[k-1]\wedge A[k]<B[k]$,则称 A 和 B 是可连接的,连接后产生 $(k+1)$- 项集 $\{A[1],A[2],\cdots,A[k-1],A[k],B[k]\}$。为减小候选频繁项集 C_{k+1} 的规模,Apriori 算法利用 Apriori 性质进行剪枝:如果候选频繁 $(k+1)$- 项集的某个 k- 项子集非频繁,则该候选项集一定非频繁,从而可以从 C_{k+1} 中删除。

1.3 Apriori 算法的改进研究

为了避免对数据集进行重复扫描,文献[2]提出 FP-Grow(Frequent-Pattern-Grow)算法将数据集压缩存储到 FP-树中,然后递归地对 FP-树进行挖掘。FP-Grow 算法的缺点是遍历树的开销仍然很大,而且当数据集规模很大时,构造基于内存的 FP-树遇到困难。文献[3]提出 OP(Opportune Project)算法集成了自顶向下和自底向上的方法遍历 FP-树;文献[4]使用基于投影进行超集检测的机制,减少了遍历树的开销;文献[5]使用了空间代价较大的垂直格式存储数据集,通过取频繁 k-项集的 TID 集的交得到频繁 $(k+1)$-项集的 TID 集。

为了减少对整个数据集的扫描,文献[6]提出 Partition 算法将数据集从逻辑上划分为互不相交的块,对每个分块生成局部频繁项集,再扫描数据集得到全局频繁项集;文献[8]将数据集划分为用开始点标记的块,然后动态评估已计数的所有项集的支持度;文献[9]提取候选模式的紧密上阶,从而减少对数据集的扫描;文献[10]提出分段计算支持度的思想,并对数据集进行约简,缩减了潜在项集的规模。上述改进算法的共同缺点是需要设定合理的局部支持度阈值或划分点,才能不丢失全局频繁项集,而且由于局部频繁项集之间可能互不相同,导致全局候选频繁项集非常庞大。

为了加快对候选频繁项集的支持度计算,文献[7]采用散列表存储候选频繁项集并增加桶计数,如果某散列表项的桶计数小于支持度阈值,则该项集一定非频繁。该算法的缺点是存放项集计数的散列树与存放候选项集的散列表之间存在内存争用问题,而且由于无法避免冲突,当桶计数大于等于支持度阈值时,还须判断其中每个候选项集是否满足最小支持度。

2 基于散列的频繁项集分组算法

2.1 基本概念

定义1 设 $I=\{I_1,I_2,\cdots,I_m\}$,非空项集的第一项称为项集 A 的首项。设 $T=\{T_1,T_2,\cdots,T_n\}$,对于 $I_i\in I(1\leqslant i\leqslant m)$,$T_j\in T(1\leqslant j\leqslant n)$,若 $I_i\in T_j$,则事务 T_j 中 I_i 之后的所有项构成以 I_i 为首项的子项集,所有以 I_i 为首项的子项集构成的集合称为以 I_i 为首项的数据子集,记为 ST_i。

例如,对表 1 所示事务数据集 T,以 I_3 为首项的数据子集 ST_3 如表 2 所示。显然,$sup\_count(I_i)=|ST_i|\ (1\leqslant i\leqslant m)$。

表 1 某分店的事务数据集

TID	list of item-IDs	TID	list of item-IDs
1	I_1,I_2,I_3	6	I_1,I_3,I_7,I_8,I_9
2	I_1,I_2,I_5,I_8	7	I_1,I_3,I_5,I_7,I_9
3	I_2,I_6	8	I_1,I_2,I_3,I_5,I_9
4	I_4,I_5,I_6	9	I_4,I_5
5	I_1,I_2,I_3,I_5	10	I_2,I_3,I_5

表 2 数据子集 ST_3

TID	list of item-IDs	TID	list of item-IDs
1	I_3	7	I_3,I_5,I_7,I_9
5	I_3,I_5	8	I_3,I_5,I_9
6	I_3,I_7,I_8,I_9	10	I_3,I_5

定理1 设 $C_{i,k}$ 表示所有以 I_i 为首项的候选频繁 k-项集,则 $C_{i,k}$ 中的所有项集只包含于数据子集 ST_i 中。

证明 设 $c\in C_{i,k}$,则 I_i 为项集 c 的首项,设 $T_j\notin ST_i$,若 T_j 包含项集 c,则一定有 $I_i\in T_j$,与 $T_j\notin ST_i$ 产生矛盾,因此,结论成立。 证毕。

定理2 连接频繁 k- 项集 A 和 B 得到 $(k+1)$- 项集 c,项集 c 存在非频繁 k- 项子集当且仅当 c 存在非频繁 2- 项子集 $\{A[k],B[k]\}$。

证明 设 $c=\{A[1],A[2],\cdots,A[k-1],A[k],B[k]\}$,则项集 c 的 k- 项子集 $\{A[1],A[2],\cdots,A[k-1],A[k]\}$ 和 $\{A[1],A[2],\cdots,A[k-1],B[k]\}$ 一定是频繁的。设 $a[1]a[2]\cdots a[k-2]\subset A[1]A[2]\cdots A[k-1]$,在项集 c 的 k- 项子集 $\{a[1],a[2],\cdots,a[k-2],A[k],B[k]\}$ 中,根据 Apriori 性质,项集 $\{a[1],a[2],\cdots,a[k-2],A[k]\}$ 一定是频繁的,如果 2- 项子集 $\{A[k],B[k]\}$ 非频繁,则 k- 项子集 $\{a[1],a[2],\cdots,a[k-2],A[k],B[k]\}$ 一定非频繁,反之亦然,从而结论成立。 证毕。

定理3 设项 $l\in T$,若 $sup\_count(l)<min\_sup$,则 $l\notin U_kL_k$。

证明 任取 $c\in U_kL_k$,则 $sup\_count(c)\geqslant min\_sup$,设项 $l'\in c$,则 $sup\_count(l')\geqslant min\_sup$,因此,一定有 $l'\neq l$,即项 l 一定不会出现在任意频繁项集中。 证毕。

定理4 在计算 $(k+1)$- 项集的支持度时,可以忽略数据集中项集长度小于等于 k 的事务。

证明 若数据集中某项集的长度小于等于 k,则该项集不存在长度为 $k+1$ 的子集,因此,该项集对计算 $(k+1)$- 项集的支持度没有贡献,可以忽略。 证毕。

2.2 改进的着眼点

针对 Apriori 算法的剪枝操作和多次扫描数据集的缺点,HFG 算法提出了以下改进:

1) Apriori 算法应用 Apriori 性质剪掉存在非频繁 k- 项子集的候选频繁 $(k+1)$- 项集,从而得到精简的候选频繁 $(k+1)$- 项集,最坏情况下需要查找 k 个 k- 项子集,其时间复杂度是 $O(k\times|L_k|)$。在长模式挖掘以及候选项集的规模增大时,剪枝操作非常耗时。本文采用散列表存储频繁 2- 项集,利用定理2只须执行一次判断,即可在 $O(1)$ 时间完成与 Apriori 算法同样的剪枝操作,这个改进由于频繁执行剪枝操作将获得较大的累积效益。

2) Apriori 算法在计算 C_k 的支持度时需要对数据集进行全扫描,当数据集的规模很大时,扫描一次所花费的代价也相当大。根据定理1,本文对数据集进行首项分组,在求以 I_i 为首

Journal of Computer Applications 计算机应用

项的频繁项集时,只须扫描数据子集 ST_i,从而减小了扫描的数据集规模。

3) 数据结构的设计。

采用分组索引表[14] 存储以 I_i 为首项的数据子集,节点结构如图 1 所示,其中:TID 表示元组的标识号,first 表示项 I_i 在元组中的序号,length 表示从项 I_i 开始的项集长度。

TID	first	length

图 1　分组索引表的节点结构

分组索引表设置 length 域的作用是,在求频繁 k-项集时,可以排除长度小于 k 的事务。根据定理 3,在计算项集的长度时排除了所有非频繁 1-项集。

2.3　算法的运行实例

设 $min\_sup = 3$,对表 1 所示事务数据集 T,HFG 算法的执行过程如下。

1) 扫描数据集 T,对每项出现的次数进行计数,得到候选频繁 1- 项集 $C_1 = \{\{I_1\}:6, \{I_2\}:6, \{I_3\}:6, \{I_4\}:2, \{I_5\}:7, \{I_6\}:2, \{I_7\}:2, \{I_8\}:2, \{I_9\}:3\}$,删去支持度小于 min_sup 的项集,得到频繁 1- 项集 $L_1 = \{\{I_1\}:6, \{I_2\}:6, \{I_3\}:6, \{I_5\}:7, \{I_9\}:3\}$。

2) 将 L_1 进行自身连接产生 C_2,扫描数据集对候选频繁 2- 项集进行计数,删去支持度小于 min_sup 的项集,得到 L_2。设散列函数为:

$$H(x,y) = ((\text{order of } x) \times 10 + (\text{order of } y)) \bmod 11$$

采用链地址法[14] 处理冲突,将 L_2 存储到散列表 HB 中,如图 2 所示。

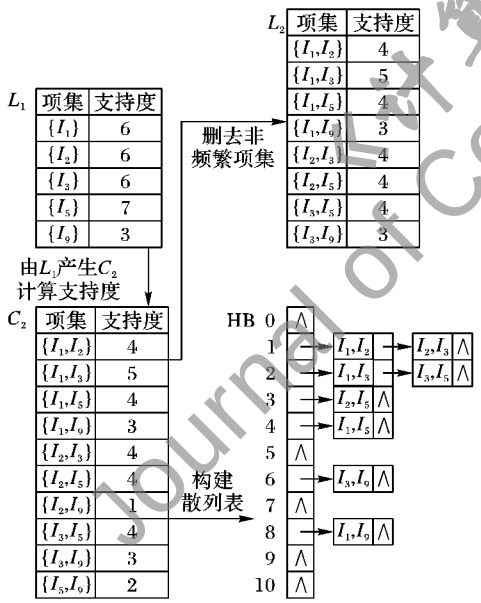

图 2　计算频繁 2-项集并散列存储

3) 取 L_2 中的首项在数据集 T 中进行筛选,得到分组索引表如图 3 所示。

ST_1

TID	first	length
1	1	3
2	1	3
5	1	4
6	1	3
7	1	4
8	1	5

ST_2

TID	first	length
1	2	2
2	2	2
3	1	1
5	2	3
8	2	4
10	1	3

ST_3

TID	first	length
1	3	1
5	3	2
6	2	2
7	2	3
8	3	3
10	2	2

图 3　首项分组结果

4) 将 L_2 中首项为 I_1 的频繁 2- 项集进行连接,利用定理 2 进行剪枝,例如在散列表中没有 2- 项子集 $\{I_2, I_9\}$,因此将 $\{I_1, I_2, I_9\}$ 进行剪枝,得到以 I_1 为首项的候选频繁 3- 项集 C_{13},扫描 ST_1 中长度大于等于 3 的元组,得到以 I_1 为首项的频繁 3- 项集 L_{13}。依此下去,直至求得所有以 I_1 为首项的频繁项集,如图 4 所示。

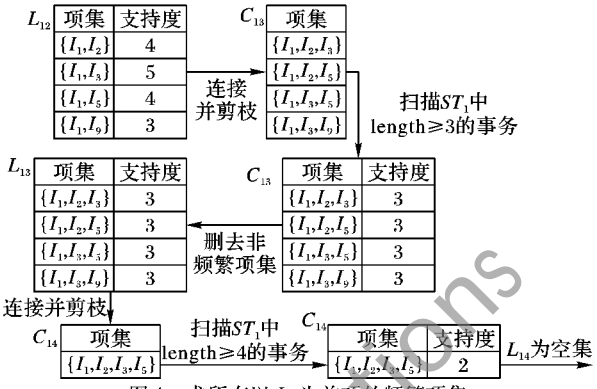

图 4　求所有以 I_1 为首项的频繁项集

5) 将 L_2 中首项为 I_2 的频繁 2- 项集进行连接,利用定理 2 进行剪枝,得到 C_{23},扫描 ST_2 中长度大于等于 3 的元组,得到以 I_2 为首项的频繁 3- 项集 L_{23}。依此下去,直至求得所有以 I_2 为首项的频繁项集,如图 5 所示。

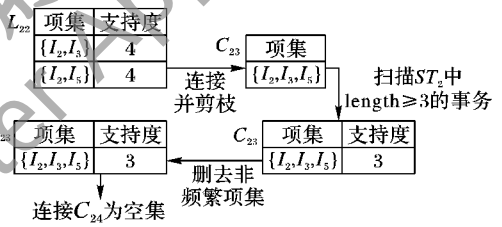

图 5　求所有以 I_2 为首项的频繁项集

6) 将 L_2 中首项为 I_3 的频繁 2- 项集进行连接,得到 $\{I_3, I_5, I_9\}$,由于散列表中没有 2- 项子集 $\{I_5, I_9\}$,剪枝后得到 C_{33} 为空集,算法结束。

2.4　算法描述

设 $C_{i,k}$ 表示所有以 I_i 为首项的候选频繁 k- 项集,$L_{i,k}$ 表示所有以 I_i 为首项的频繁 k- 项集,$c = l_1 \rhd\lhd l_2$ 表示将项集 l_1 和 l_2 进行连接,HFG 算法具体描述如下。

算法　HFG。

输入:数据集 T,项集 I,支持度阈值 min_sup。

输出:所有频繁项集 $\bigcup_k L_k$。

```
1)//求 L1
  for each ti ∈ T
    for each l ∈ ti
      l.count ++ ;
  L1 = {l ∈ I | l.count ≥ min_sup};
2)//求 L2
  for each (l1 ∈ L1 and l2 ∈ L1 and l1 ≠ l2)
  {
    c = l1 ▷◁ l2; add c to C2;
  }
  for each ti ∈ T
    for each c ∈ C2
      if (c[1] ∈ ti and c[2] ∈ ti)
        c.count ++ ;
  L2 = { c ∈ C2 | c.count ≥ min_sup};
3)//划分首项不同的频繁项集,散列存储 L2
  for each c ∈ L2
  {
```

Journal of Computer Applications 计算机应用

```
    i = c[1]; add c to L_{i,2};
    d = H(c[1], c[2]);
    add c to HB[d]
  }
4)//建立分组索引表
  for each t_j ∈ T
    for each I_i ∈ t_j
      if (I_i ∈ L_2 的首项)
        add (t_j.TID, t_j.first, t_j.length) to ST_i;
5)// 依次在 ST_i 中求以 I_i 为首项的频繁项集
  for (k = 2; L_{i,k+1} ≠ ∅; k ++)
  {
    for each (l_1 ∈ L_{i,k} and l_2 ∈ L_{i,k})
    {
      if (l_1[2] = l_2[2] ∧ … ∧ l_1[k-1] = l_2[k-1] ∧
        l_1[k] < l_2[k])
      {
        c = l_1 ▷◁ l_2;
        if (Search(HB, c[1], c[2]) == 1)
          add c to C_{i,k+1};
      }
    }
    for each (t_j ∈ ST_i and t_j.length > k)
    {
      C_t = Subset(C_{i,k+1}, t_j);
      for each c ∈ C_t;
      c.count ++;
    }
    L_{i,k+1} = {c ∈ C_{i,k+1} | c.count ≥ min_sup};
  }
6) return ⋃_k L_k;
```

3 实验及结论

实验环境为 CPU Intel Core 2、主频 1.8 GHz、2 GB 内存，操作系统为 Windows XP 2002，编程环境为Visual C++ 6.0，分别实现了 Apriori 算法和 HFG 算法，实验数据采用文献[1]的人工合成方法，具体参数如表 3 所示，实验中设定项的总个数是 1 000，最大潜在频繁项集的个数是 500。

表 3 生成实验数据的参数说明

参数	说明
D	事务数据集的元组个数
T	元组中项集的平均长度
I	潜在的最大频繁项集的平均长度

图 6 给出了在 T10I4D10k 数据集上，支持度阈值在 0.2% ~2% 变化，Apriori 算法和 HFG 算法的执行时间对比。从图 6 可以看出：在支持度阈值较小时 HFG 算法的优越性非常明显，随着支持度阈值的增加，Apriori 算法和 HFG 算法的差距基本不变。这是因为 HFG 算法的剪枝操作只消耗常数时间，支持度阈值较小时候选频繁 k-项集的个数较多，出现了剪枝操作的积累效应；当支持度阈值达到一定值后，Apriori 算法的剪枝操作需要查找非频繁 k-项子集的个数较少，HFG 算法的剪枝操作获得较少的累积效益。

图 7 给出了在 T10D10k 数据集上，支持度阈值设为 1%，潜在最大频繁项集的平均长度在 I2 ~ I10 变化，元组的项集长度呈正倾斜分布（图 8 所示是均值为 10 的正倾斜数据集），Apriori 算法和 HFG 算法的执行时间对比。从图 7 可以看出，在正倾斜分布的元组上执行 HFG 算法，随着潜在最大频繁项集长度的增加，由于在计算候选项集的支持度时排除了较多的元组，HFG 算法的优势非常明显。

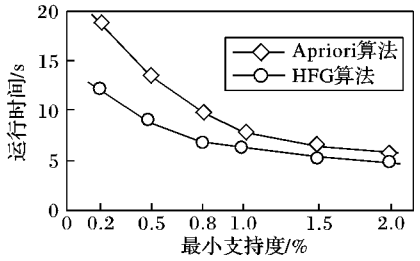

图 6 不同支持度阈值的效率对比

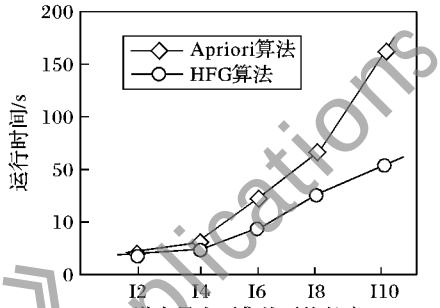

图 7 不同长度频繁项集的效率对比

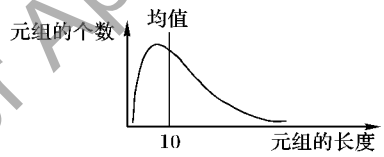

图 8 正倾斜数据

4 结语

HFG 算法采用散列技术，应用非频繁 2-项集性质进行剪枝；采用分治技术，通过将数据集进行首项分组，在每个数据子集中分别进行频繁项集挖掘，提高了 Apriori 算法的时间性能。HFG 算法采用分组索引表存储数据子集，用索引来表示首项的子项集，避免了数据子集的重复存储。HFG 算法没有复杂的理论推导，易于理解和实现，其基本思想可以移植到最大频繁项集挖掘、序列模式挖掘等领域。

参考文献：

[1] AGRAWAL R, SRIKANT R. Fast algorithms for mining association rules [C]// VLDB'94: Proceedings of the 1994 International Conference on Very Large Data Bases. San Francisco: Morgan Kaufmann, 1994: 487 - 499.

[2] HAN J, PEI J, YIN Y. Mining frequent patterns without candidate generation [C]// SIGMOD'00: Proceedings of the 2000 International Conference on Management of Data. New York: ACM Press, 2000: 1 - 12.

[3] LIU J Q, PAN Y H, WANG K, *et al*. Mining frequent item sets by opportunistic projection [C]// KDD'02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2002: 229 - 238.

[4] 颜跃进，李舟军，陈火旺. 基于 FP-Tree 有效挖掘最大频繁项集[J]. 软件学报，2005，16(2)：215 - 222.

[5] ZAKI M J. Scalable algorithms for association mining [J]. IEEE Transactions on Knowledge and Data Engineering, 2000, 12(3): 372 - 390.

（下转第 3089 页）

Journal of Computer Applications 《计算机应用》

时,本文方法构建的迁移回归系统可以利用历史数据较准确地还原出酒样光谱信息,以供鉴别真伪、评定品质所用。

表 4　各种方法在酒样数据集上的泛化性能比较

回归系统	预测性能
SVR(基于历史数据 Dh)	2.4639E-002
SVR(基于当前数据 Dc)	7.8712E-003
SVR(基于所有数据 Dh + Dc)	1.7765E-002
HiRBF	1.4685E-002
T-SVR	4.4337E-004

4　结语

本文通过采用迁移学习的思想,针对当前场景信息缺失带来的回归系统建模之挑战,提出了具有迁移学习能力的 T-SVR建模方法。本文方法利用利用历史知识对当前知识的增益作用,给信息缺失区间带来了较好的学习效果。通过模拟数据实验以及酒样光谱的仿真实验,结果表明了本文方法较之于传统方法的更好适应性。本文方法尚存在一些不足之处:本文引入了参数 λ 用于权衡历史与当前场景的知识相似性,但目前参数 λ 的取值是通过交叉验证的方法获取的。在今后的工作中将致力于研究两个场景之间的相似度与两个数据集概率分布之间的关联,希望能利用数据的概率分布特征来估计 λ 的合理区间乃至直接求得 λ 的值。

参考文献:

[1] PAN S J, YANG Q. A survey on transfer learning [J]. IEEE Transactions on Knowledge and Data Engineering, 2010, 22(10): 1345-1359.

[2] QUAN Z B, HUAN J. Large margin transductive transfer learning [C]// Proceedings of the 18th ACM Conference on Information and Knowledge Management. New York: ACM Press, 2009: 1327-1336.

[3] SHI Y, LAN Z Z, LIU W, *et al*. Extending semi-supervised learning methods for inductive transfer learning [C]// Proceedings of the 9th IEEE International Conference on Data Mining. Washington, DC: IEEE Computer Society, 2009: 483-492.

[4] PAN S J, TANG I W, KWOK J T, *et al*. Domain adaption via transfer component analysis [J]. IEEE Transactions on Neural Networks, 2011, 22(2): 199-210.

[5] TAO J W, CHUNG F L, WANG S T. On minimum distribution discrepancy support vector machine for domain adaptation [J]. Pattern Recognition, 2012, 45(11): 3962-3984.

[6] DUAN L X, TSANG I W, XU D. Domain transfer multiple kernel learning [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(3): 465-479.

[7] BRUZZONE L, MARCONCINI M. Domain adaptation problems: a DASVM classification technique and a circular validation strategy [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 32(5): 770-787.

[8] YU K, TRESP V, SCHWAIGHOFER A. Learning Gaussian processes from multiple tasks [C]// Proceedings of the 22nd International Conference on Machine Learning. New York: ACM Press, 2005: 1012-1019.

[9] ARGYRIOU A, EVGENIOU T, PONTIL M. Multi-task feature learning [C]// Proceedings of the 2006 Conference on Neural Information Processing Systems 19. Cambridge, MA: MIT Press, 2006. 41-48.

[10] BAKKER B, HESKES T. Task clustering and gating for Bayesian multitask learning [J]. The Journal of Machine Learning Research, 2003, 4: 83-89.

[11] GELMAN A, CARLIN J B, STERN H S, *et al*. Bayesian data analysis [M]. 2nd ed. London: Chapman & Hall, 2000.

[12] YANG P, TAN Q, DING Y. Bayesian task-level transfer learning for non-linear regression [C]// Proceedings of the 2008 International Conference on Computer Science and Software Engineering. Piscataway: IEEE Press, 2008: 62-65.

[13] 蒋亦樟, 邓赵红, 王士同. ML 型迁移学习模糊系统[J]. 自动化学报, 2012, 38(9): 1393-1409.

[14] VAPNIK V. Statistical learning theory [M]. New York: Wiley, 1998.

[15] SCHÖLKOPF B, SMOLA A, WILLIAMSON R C, *et al*. New support vector algorithms [J]. Neural Computation, 2000, 12(5): 1207-1245.

[16] PLATT J. Fast training of support vector machines using sequential minimal optimization [M]// Advances in Kernel Methods-Support Vector Learning. Cambridge: MIT Press, 2000: 185-208.

[17] GAN M T, HANMANDLU M, TAN A H. From a Gaussian mixture model to additive fuzzy systems [J]. IEEE Transactions on Fuzzy Systems, 2005, 13(3): 303-316.

[18] 张英锋, 马彪, 张金乐, 等. 基于光谱分析和 SVM 的综合传动故障诊断研究[J]. 光谱学与光谱分析, 2010, 30(6): 1586-1590.

(上接第 3048 页)

[6] SAVASERE A, OMIECINSKI E, NAVATHE S B. An efficient algorithm for mining association rules in large databases [C]// VLDB'95: Proceedings of the 21th International Conference on Very Large Data Bases. San Francisco: Morgan Kaufmann, 1995: 432-443.

[7] PARK J S, CHEN M S, YU P S. An effective Hash-based algorithm for mining association rules [C]// SIGMOD'95: Proceedings of the 1995 International Conference on Management of Data. New York: ACM Press, 1995: 175-186.

[8] BRIN S, MOTWANI R, ULLMAN J D, *et al*. Dynamic itemset counting and implication rules for market basket analysis [C]// SIGMOD'97: Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 1997: 255-264.

[9] GEERTS F, GOETHALS B, BUSSCHE J. A tight upper bound on the number of candidate patterns [C]// ICDM'01: Proceedings of the 2001 International Conference on Data Mining. New York: ACM Press, 2001: 155-162.

[10] 李雄飞, 苑森淼, 董立岩, 等. 多段支持度数据挖掘算法研究[J]. 计算机学报, 2001, 24(6): 661-665.

[11] 柴玉梅, 张卓, 王黎明. 基于频繁概念直乘分布的全局闭频繁项集挖掘算法[J]. 计算机学报, 2012, 35(5): 990-1001.

[12] 李海峰, 章宁, 朱建明, 等. 时间敏感数据流上的频繁项集挖掘算法[J]. 计算机学报, 2012, 35(11): 2280-2293.

[13] HAN J, KAMBER M. 数据挖掘: 概念与技术[M]. 2 版. 范明, 孟小峰, 译. 北京: 机械工业出版社, 2007.

[14] 王红梅, 胡明, 王涛. 数据结构 C++版[M]. 2 版. 北京: 清华大学出版社, 2011.

《计算机应用》Journal of Computer Applications