基于关联规则的分布式通信网告警相关性研究

吴 简 李兴明
(电子科技大学通信与信息工程学院 成都 610054)

摘 要 描述了基于数据挖掘的通信网告警相关性分析。在分布式数据库中直接运用序列算法效率很低,因为这需要大量的额外通信。为此提出了一种有效的分布式关联规则挖掘算法——EDMA,它通过局部剪枝与全局剪枝来最小化候选项集数目和通信量。在局部站点上运用先进的压缩关联矩阵CMatrix统计局部项集支持数。此外还利用项目剪枝与交易剪枝共同来减少扫描时间。最后仿真验证了EDMA比其他经典分布式算法有更高的运算效率、更低的通信开销以及更好的可扩展性。

关键词 网络差错管理,分布式关联规则挖掘,频繁项集,压缩关联矩阵

Efficient Distributed Mining Algorithm for Alarm Correlation in Communication Networks

WU Jian LI Xing-ming
(School of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China)

Abstract This paper described the alarm correlation in communication networks based on data mining. A direct application of sequential algorithms to distributed databases is not effective, because it requires a large amount of communication overhead. An efficient algorithm-EDMA was proposed. It minimized the number of candidate sets and exchanged messages by local and global pruning. In local sites, it runs the application based on the improved algorithm-CMatrix, which is used to calculate local support counts. Our solution also reduced the size of average transactions and datasets that leads to reduction of scan time. The performance study shows that EDMA has superior running efficiency, lower communication cost and stronger scalability than direct application of a sequential algorithm in distributed databases.

Keywords Network fault management, Association rules distributed mining, Frequent itemsets, Compressed association matrix

1 引言

1.1 背景介绍

数据挖掘(data mining)是一个从数据中析取潜在有用的、先前未知的和最终可理解的知识的过程。挖掘关联规则是其研究的一个重要方面[1],而生成频繁项目集是关联规则挖掘的关键技术和步骤。自R. Agrawal于1993年首次提出布尔型关联规则问题及相应的Apriori算法[2]以来,数据挖掘领域的研究者在挖掘关联规则上做了大量的工作,主要为频繁项目集挖掘与更新。但研究工作大多局限在单机环境,针对分布式环境的全局频繁项目集挖掘与更新尚不多见。而来自不同站点的数据融合需要大量的通信开销,直接将Apriori算法用于分布式环境是不现实的。通信瓶颈成为分布式关联规则挖掘首要面对的问题。

Apriori算法是挖掘关联规则频繁项集的基本算法,该算法利用一个层次顺序搜索的循环和对候选项集的剪枝方法来完成频繁项集的挖掘。然而,由于Apriori算法是依赖于候选项集产生频繁项集的理论,它主要存在两个不足:①它可能需要产生大量候选项集;②它需要通过多次扫描数据库来计算频繁项目集,大量的时间消耗在内存与数据库中的数据交换上。尤其是将该算法应用于分布式多层关联规则数据挖掘时,扫描次数多,通过传送局部频繁项集来求全局频繁项集时,站点间的数据通信量大,影响了挖掘性能的提高。

1.2 关联规则的描述

设 $I=\{i_1,i_2,\cdots,i_m\}$ 是所有数据项的集合,设 $D=\{T_1,T_2,\cdots,T_n\}$ 为所有事务的集合,即一个事务数据库,其中的每个事务 T 是一个数据项子集,即 $T\subseteq I$。每个事务可以用惟一的标识符TID来标识。设 X 为一个数据项集合,称为项集,包含 k 个数据的项集称为 k 项集。当且仅当 $X\subseteq T$ 时,称为事务 T 包含 X。D 中项集 X 的支持度为:$\text{support}(X)=|T_X|/|D|$,其中 $T_X=\{T\in D|X\subseteq T\}$。在用户给定的支持度阈值minsup下,若 $\text{support}(X)\geqslant\text{minsup}$,则称 X 为频繁项集,否则 X 为非频繁项集。为方便描述,本文中 X 的支持度都是用 D 中包含 X 的事务数量来表示,又称绝对支持度。

一个关联规则是"$X=>Y$"形式的蕴含式,其中 $X\subseteq I$,$Y\subseteq I$且 $X\cap Y=\emptyset$。如果 D 中包含事务 $X\cup Y$ 的百分比为 s,则称 s 为关联规则 $X=>Y$ 的支持度,它是概率 $P(X\cup Y)$。如果 D 中包含 X 的事务同时也包含 Y 的百分比为 c,则称 c

到稿日期:2008-12-09 返修日期:2009-02-25 本文受国家自然科学基金(60572091)资助。

吴 简 女,博士生,主要研究方向为数据挖掘与知识发现、计算机网络结构;李兴明 男,博士,教授,博士生导师,主要研究方向为现代通信网理论、光传送网及智能光网络、自动交换光网络等。

为关联规则 $X=>Y$ 的置信度，它是条件概率 $P(Y|X)$。即：

$$\text{support}(X=>Y)=\text{support}(X\cup Y) \quad (1)$$

$$\text{confidence}(X=>Y)=P(Y|X)=\text{support}(X\cup Y)/\text{support}(X) \quad (2)$$

挖掘关联规则的问题就是要生成所有满足 $\text{support}(X=>Y)\geqslant\text{minsup}$ 和 $\text{confidence}(X=>Y)\geqslant\text{minconf}$ 的关联规则，其中 minsup 和 minconf 分别为用户给定的最小支持度阈值和最小置信度阈值。同时满足这两个条件的关联规则称为强关联规则。挖掘关联规则主要包含两个子问题：

①从事务数据库 D 中生成所有的频繁项集。

②根据获得的频繁项集生成强关联规则。

事实上，挖掘关联规则的整个执行过程中第一个子问题是核心。在找到所有的频繁项集后，相应的关联规则将很容易生成，Apriori 算法主要是处理第一个子问题。

1.3 定义与定理

本文讨论的分布式数据库由地理空间相对分离的通信子网组成，分布在 n 个不同的站点 $S^1,S^2,\cdots,S^n$，且各部分的数据库模式逻辑同构，它们之间除了通过网络传递信息外，其他资源全部独立。各站点只处理自身的局部数据库 $DB^i(i=1,2,\cdots,n)$，站点间仅通过网络传递有限的统计信息，最终在整个事务数据库中挖掘出分布关联规则。

某一项目集 X，$X.\sup$ 及 $X.\sup^i$ 分别表示在 DB 及 DB^i 中含 X 的交易数；按照之前定义也表示了 X 在 DB 及 DB^i 中的支持度，称 $X.\sup$ 为 X 的全局支持度，$X.\sup^i$ 为 X 在站点 S^i 的局部支持度。若 $X.\sup\geqslant\text{minsup}*|D|$，则 X 为全局频繁项目集；若 $X.\sup^i\geqslant\text{minsup}*|D^i|$，则 X 为站点 S^i 的局部频繁项目集。称长度为 k 的频繁项集为频繁 k 项集，用 L_k 表示从 DB 中挖掘出的全局频繁 k 项集组成的集合。

引理 1 若 X 为站点 S^i 上的局部频繁项目集，则 X 的所有子集均为站点 S^i 上的局部频繁项目集。

推论 1 若项集 X 不是局部频繁项目集，则 X 的超集一定不是局部频繁项目集。类似地，若 X 为全局频繁项目集，则 X 的所有非空子集均为全局频繁项目集。

以上结论由频繁项集的向下封闭性易知。

定理 1 若 X 为全局频繁项目集，则至少存在一站点 S^i $(1\leqslant i\leqslant n)$ 使得 X 及 X 的所有子集在该站点上均为局部频繁项集。

证明：反证法。假设 X 在所有站点 S^i 上均不频繁，则 $X.\sup=\sum_{i=1}^{n}X.\sup^i<(|D^1|+|D^2|+\cdots+|D^n|)\times\text{minsup}$，与 X 为全局频繁项集矛盾，故 X 至少在某一站点 $S^i(1\leqslant i\leqslant n)$ 上局部频繁。再由引理 1 得 X 的所有子集在站点 S^i 上均为局部频繁项集。证毕。

可见，由于数据库固有的非均匀性，许多项集只在某些站点上频繁而在另一些上则不频繁，而全局频繁项集又由各局部频繁项集产生，故减少局部频繁项集引起的通讯代价是提高效率的关键。

2 相关工作

分布式关联规则挖掘算法是从不同地理位置站点上的不同数据集中产生规则，因此外部通信贯穿了整个挖掘过程。算法核心是减少通信量，使生成全局频繁项集的开销小于将各站点局部数据库合并为一个集中数据库挖掘的开销。下面介绍已有的两种典型的分布式挖掘算法。

2.1 CD 算法

CD 算法是在平行环境中运用 Apriori 算法，并假设不同站点的数据集是水平分割的。每轮迭代中，它根据上一轮得到的全局频繁项集由 Apriori-gen 函数在每个站点产生候选项集，各站点计算这些候选项集的局部支持数并把它们广播给其它站点。如此，所有站点经过叠加都能得到本轮迭代的全局频繁项集，并进行下一轮迭代。为实现支持数交换，算法采用了简单通信机制。然而，随着候选项集增多将产生大量额外通信开销，导致无法有效利用系统内存。每轮的通信复杂度为 $O(|C|*n^2)$，$|C|$ 与 n 分别为候选项集大小和站点数。

2.2 FDM 算法

相对 CD 算法，FDM 算法产生较少的候选项并运用有效剪枝技术最小化支持度交换需要的通信量。当每一站点找出局部频繁项集后并不进行广播，而是将其发送到该项集对应的投票站点上，由投票站点负责项集的全局计数统计。投票站点的计算通常由散列函数决定，此外，所有投票站点需要向除发送站点以外的其余站点发送支持数询问请求。这些都将增加算法通信量。FDM 比 CD 最大的优势是将通信复杂度降低为 $O(|C|*n)$。当各站点数据分布不均匀时，它们的局部频繁项集差别较大，因此 FDM 生成的候选项集比 CD 少。

3 有效分布式关联规则算法——EDMA

3.1 压缩关联矩阵——CMatrix

CMatrix 算法也采用逐层迭代的候选产生验证方法找出事务数据库中的所有频繁项集。但是 CMatrix 算法在第一次迭代中将事务数据库映射为位于主存中的关联矩阵，以便在以后的迭代过程中直接对矩阵进行运算，整个过程中只需扫描一遍数据库。矩阵中的元素 A_{ij} 表示项目 I_j 在第 i 个事务中的发生频度，且 $A_{ij}\in\{0,1\}$。因为告警关联规则是一种布尔型关联规则，所以用 1 表示告警事务中存在该告警，用 0 表示不存在。显然，矩阵中第 j 列向量包含的 1 的个数就是项目 I_j 的支持数，称该列向量为 I_j 的元向量。一个 $m*n$ 的压缩关联矩阵表示有 m 条事务与 n 条告警的事务数据库。

当 $k=1$ 时，项集的元向量是该项目在压缩关联矩阵 $C_{m\times n}$ 中对应的列向量；$k>1$ 时，设 k 项集的元向量为 V_k，组成该 k 项集的各 1 项集的元向量依次为：$V_i,V_{i+1},\cdots,V_{i+k-1}$，则有 $V_k=V_i\&V_{i+1}\&\cdots\&V_{i+k-1}$，其中 & 代表按位与运算。因为 k 项集是由两个频繁 $k-1$ 项集生成的，根据元向量含义与布尔运算特性，可以得到 k 项集的元向量 $V_k=V_{k-1,1}\&V_{k-1,2}$，其中 $V_{k-1,1}$ 和 $V_{k-1,2}$ 分别代表产生它的 $k-1$ 项子集。基于 CMatrix 迭代算法，逐层得到任意长度项集的元向量，从而计算出它们的支持数，直至算法结束。用向量运算代替了数据库的重复扫描，节约了大量的内存和时间。

下面举例说明压缩矩阵的构造与应用。事务数据库如表 1 所列。

表 1 事务数据库 D

(TID)	(Items)
1	ABD
2	ACDE
3	ACE
4	BD
5	BDE

6	ABCD
7	BC
8	BCD

根据前面的描述，很容易得出事务数据库 D 的关联压缩矩阵 C_{8*5}，如式(3)所示。以项目 A 为例，通过计算矩阵第一列向量中'1'的个数可以得到其相应的支持数。此后由层次迭代原理，任意项集的元向量都可以由构造它的两个长度小于 1 的子项集的元向量相与而成。如项集 AC 由项目 A 和 C 连接构成，其元向量如式(4)所示。再计算 V_{AC} 中'1'的个数，得出 AC 支持数为 3。由此可见，运用压缩矩阵计算项集支持度方便有效。

$$C_{8*5}=\begin{pmatrix}1&1&0&1&0\\1&0&1&1&1\\1&0&1&0&1\\0&1&0&1&0\\0&1&0&1&1\\1&1&1&1&0\\0&1&1&0&0\\0&1&1&1&0\end{pmatrix} \tag{3}$$

$$V_{AC}=V_A\&V_c=\begin{pmatrix}1\\1\\1\\0\\0\\1\\0\\0\end{pmatrix}\&\begin{pmatrix}0\\1\\1\\0\\0\\1\\1\\1\end{pmatrix}=\begin{pmatrix}0\\1\\1\\0\\0\\1\\0\\0\end{pmatrix} \tag{4}$$

3.2 EDMA 算法描述

通过前面的定理可知，全局频繁项集都由局部频繁项集产生。因此各站点首先产生自身的局部候选项集，待验证频繁后发往一个固定的中央站点统计全局支持数。如项集 X 在站点 S^i，同时局部频繁与全局频繁，称 X 是 S^i 中的重频繁项集。用 HL^i 表示站点 S^i 上重频繁项集的集合。在 EDMA 中，重频繁项集是产生候选项集的关键因素。

定理 2 若 X 为全局频繁项目集，则至少存在一站点 S^i $(1\leqslant i\leqslant n)$ 使得 X 及 X 的所有子集在该站点上均为重频繁项集。

证明：反证法。假设 X 在所有站点 S^i 上均不频繁，则 $X.\sup=\sum_{i=1}^{n}X.\sup^i<(|D^1|+|D^2|+\cdots+|D^n|)\times\text{minsup}$，与 X 为全局频繁项集矛盾，故 X 至少在某一站点 S^i $(1\leqslant i\leqslant n)$ 上局部频繁，即 X 在 S^i 上重频繁。再由引理 1 及频繁项集的向下封闭性得 X 的所有子集在站点 S^i 上为重频繁项集。证毕。

用重频繁项集来生成局部候选项集。第 k 轮迭代时，在站点 $S\cap LL_{k-1}^i$ 上产生的候选项集 CH_k^i 应为：

$$CH_k^i=\text{Apriori\_gen}(HL_{k-1}^i) \tag{5}$$

显然，本轮中所有的全局频繁项集都包含于各局部站点的候选项集集合中，即：

$$L_k\subseteq\bigcup_{i=1}^{n}CH_k^i=\bigcup_{i=1}^{n}\text{Apriori\_gen}(HL_{k-1}^i) \tag{6}$$

因为各站点的重频繁项集集合 HL_{k-1}^i 只是全局频繁项集集合的一个子集，故由它们产生的总候选项集集合 CH_k 将大大小于直接由整个 L_{k-1} 产生的候选项集集合 C_k。这即是所谓的局部剪枝。

产生候选项集 CH_k 后，中央站点需要对各局部站点进行支持数交换来发现全局频繁项集。由定理 1 可知，k 项集 X 若全局频繁则必在某站点局部频繁。因此在将候选项集发送到中央站点前，局部站点先剪枝掉那些局部非频繁项集以减小通信与计算量。用 LL_k^i 表示第 k 轮迭代站点 S^i 的局部频繁项集集合。

值得注意的是，尽管在站点 S^i 项集 X 非局部频繁，X 却可能成为其它站点的候选项集。则中央站点将会对 S^i 发送 X 的支持度请求，势必造成对 DB^i 进行重复扫描。为避免这样的二次扫描，EDMA 做了如下改进。因为在迭代初站点 S^i 都可以获得 HL_{k-1}^i 与 CH_k^j $(j=1,\cdots,n)$，故 S^i 也可以计算它们的局部支持度。换句话说，每个站点只需扫描本地数据库一次就可得到本轮迭代中的所有候选项集 CH_k 的支持数。如此，局部剪枝和支持度交换所要求的计算可以通过一次数据库扫描完成，使局部数据库扫描次数最小化。

用 $\text{maxsup}(X)^i$ 表示 k 项集 X 所有 $(k-1)$ 项子集的最小局部支持数，即 $\text{maxsup}(X)^i=\min\{Y.\text{count}^i\mid Y\subset X\text{ and }|Y|=k-1\}$。根据子集的关系易知 $\text{maxsup}(X)^i$ 是 $X.\sup^i$ 局部支持度的上限；而所有局部上限和 $\text{maxsup}(X)$ 构成全局支持数 $X.\sup$ 的上限。由于 $\text{maxsup}(X)$ 在 k 次迭代的开始在每个站点就能得到，因此可以利用它来剪枝。如果 $\text{maxsup}(X)<s*D$，X 将不能构成候选项集，则称其为全局剪枝。

综上所述，第 k 次迭代时 EDMA 在局部站点 S^i 上进行的程序描述如下：

① 设置标志 $flag$ 与 $flag^i$ 分别代表全局与局部进程的结束，初始值均为 $true$。当 $flag^i=true$ 时，由上一轮迭代结果得到站点 S^i 的重频繁 $k-1$ 项集 $HL_{k-1}^i=L_{k-1}\cap LL_{k-1}^i$，再由它生成候选 k 项集 $CH_k^i=\text{Apriori\_gen}(HL_{k-1}^i)$。

②对每个 $X\in CH_k^i$，局部计算其支持度上限 $X.\text{maxcountl}=X.\sup^i+\sum_{j=1,j\neq i}^{n}\text{maxsup}(X)^j$，剪枝掉那些支持数小于 $s*D$ 的项集 X。

③访问压缩矩阵 CMatrix^i 计算剩余 $X\in\sum_{i=1}^{n}CH_k^i$ 的局部支持数。当 X 在 S^i 上局部频繁，将其存储为 $CH_{k1}^i=LL_k^i$；余下的候选项存储在 CH_{k2}^i 中。当 $CH_{k1}^i=\varnothing$ 时，置 $flag^i=false$。

④发送 LL_k^i 到中央站点收集它们的全局支持数。

⑤一旦 $flag$ 变为 $false$，置 $flag^i=false$。

⑥当 $flag=true$，如果 S^i 收到来自中央站点的项集 X 的计数请求，在 CH_{k2}^i 中获得 X 的支持数并返回给中央站点；否则将接收到本轮迭代的全局频繁项集和它们的支持数。

在 Apriori 算法中，每个站点向其它站点广播其候选项集需要 $O((n-1)*|C|)$ 量级的信息，$|C|$ 与 n 分别代表候选项集与站点数目。而 EDMA 中 S^i 只需要交换量级 $O(|C|)$ 的消息就可以决定自己的重频繁项集。

第 k 次迭代时 EDMA 在中央站点上进行的程序描述如下：

①接收来自各局部站点的 LL_k^i 及它们的支持数。当 $\bigcup_{i=1}^{n}LL_k^i=\varnothing$ 时，置 $flag=false$。对每个候选项集 $X\in\bigcup_{i=1}^{n}LL_k^i$，存

储其原始发送站点列表。

②如果所有局部站点都在属于 X 的站点列表中，即 X 在所有站点均频繁，将其存储为 L_k 并计算其支持度为 $X.\sup=\sum_{i=1}^{n}X.\sup^i$。

③对其余 $X\in\bigcup_{i=1}^{n}LL_k^i$，全局计算其支持度上限 $X.\max\text{-countg}=\sum_{m\in X.\text{large\_sites}}X.\sup_m+\sum_{n\notin X.\text{large\_sites}}\min(\max\sup(X)^n, s*D^n-1)$，剪枝掉那些 $X.\max\text{countg}<s*D$ 的项集，对剩余项集向不属于它们原始站点列表的站点广播支持度请求。

④因为之前所有 $X\in\sum_{i=1}^{n}CH_k^i$ 都在各局部站点扫描过，可直接响应中央站点的支持数请求。接收各局部站点的支持数并相加，如果 $X.\text{count}=\sum_{i=1}^{n}X.\text{count}^i\geqslant s*D$，将其存储在 L_k 中。

⑤由频繁项集的向下封闭性知，k 项集 X 频繁当且仅当它的 k 个 $(k-1)$ 项子集频繁，若 $|L_k|<k+1$，置 $flag=false$。这是第二次全局剪枝。

⑥当 $flag=true$ 时，向局部站点广播全局频繁项集及它们的全局支持数。

4 仿真实验和算法分析

4.1 告警数据库的产生

本文中，用人工产生数据库来评估不同算法的性能。图 1 描述了网络拓扑结构，告警数据按以下原则生成：

• 8,9 号节点为原始告警节点，它们随机产生一定级别与类型的告警数据。

• 原始节点告警后在所有相邻节点引发同级别的相应告警并不再进行下一级传播。

产生告警后，利用时间窗口和滑动步长进行同步化处理生成告警数据库。再经过数据压缩与数据预处理，将其转化为告警事务数据库，方便关联规则的挖掘。

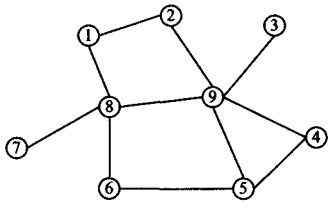

图 1 告警产生的通信网络拓扑图

4.2 性能分析

我们对本文算法性能进行了广泛研究。通过 EDMA 与 CD 和 FDM 算法的比较，体现了前者的优越性。算法源代码由 Java 编写。

第一个实验中固定局部站点数，改变支持度门限与数据库大小来进行比较。假设有 4 个分布式站点，各自的数据库大小相同但相互独立。图 2 显示了数据库为 10k 时，3 种算法在不同支持度下的执行时间。可以看出 EDMA 比另两种算法快 4%到 67% 左右，且支持度越小差别越明显。图 3 是当支持度为 0.3，数据库从 1k 增长到 10k 时，3 种算法执行时间的比较。结果显示，由于候选项集与扫描时间的减少，EDMA 的可扩展性最优，十分稳定。

第二个实验固定了支持度与数据库大小，而将局部站点数从 3 增加到 7 进行比较。图 4 与图 5 分别为 EDMA 与 CD 和 FDM 算法局部站点候选项集数目以及总信息传输量的比率。从图易知，EDMA 候选项集的数目比 CD 减少 75%到 95%，比 FDM 减少了 40%到 78%；而 EDMA 传输的信息量比 CD 小 83%至 92%，比 FDM 小 47%至 66% 。这说明随着站点数的改变，EDMA 都能用较少的时间发现频繁项集，生成关联规则，从而有着更高的效率与更好的扩展性。

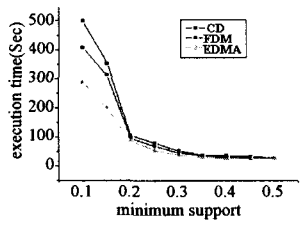

图 2 不同支持度下的执行时间

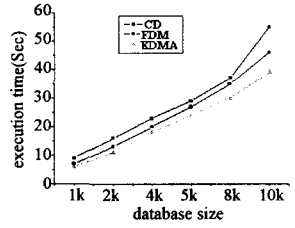

图 3 不同数据库大小下的执行时间

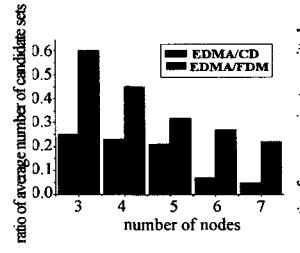

图 4 EDMA/CD 与 EDMA/FDM 的候选项集比率

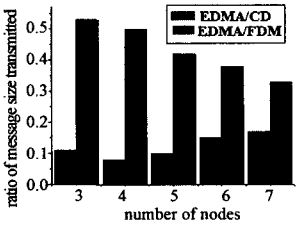

图 5 EDMA/CD 与 EDMA/FDM 的传输信息量比率

结束语 本文针对分布式数据库环境提出了一种新型的关联规则挖掘算法——EDMA。相对以往的分布式算法，其创新性主要体现在以下几点：

(1)引入一个新的全局站点，负责完成候选项集的收集、全局频繁项集的计算和广播，将通信复杂度降低为 $O(n)$，同时负责各局部站点间挖掘过程的同步，避免了各站点之间交叉通讯带来的时间延迟较大和负载不均衡的问题。

(2)将事务数据库映射为压缩关联矩阵，实现了存储内容的有效压缩，节约了内存空间，极大地减少了数据库扫描次数。

(3)每轮迭代中采用的项目剪枝与事务剪枝，进一步减小了压缩关联矩阵，加快了读取与计算时间，提高了算法的整体性能。

(4)在全局站点和局部站点上，分别使用全局剪枝技术与局部剪枝技术进行候选项集的剪枝，快速有效地压缩了候选项集的规模，减小了通信开销。

参 考 文 献

[1] 王能斌. 数据库系统原理[M]. 北京：电子工业出版社，2000

[2] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases[A]// Proceedings of the ACM SIGMOD International Conference on Management of Data[C]. Washington, USA, 1993: 207-216

[3] Cheung D W, et al. A Fast Distributed Algorithm for Mining Association Rules[C]// Proc. Parallel and Distributed Information Systems. IEEE CS Press, 1996: 31-42

[4] Zaki M J, Pin Y. Introduction: Recent Developments in Parallel and Distributed Data Mining[J]. J. Distributed and Parallel Databases, 2002, 11(2): 123-127

(下转第 212 页)

采用多项式核函数的实验结果如图3所示。根据图3可知，在USPS数据集上，采用多项式核函数的AVM算法取得了和采用高斯核函数类似的结果。相较于CVM算法只能采用各向同性核函数（例如高斯核函数），AVM算法可以采用任意的核函数，因此具有更宽广的适用范围。

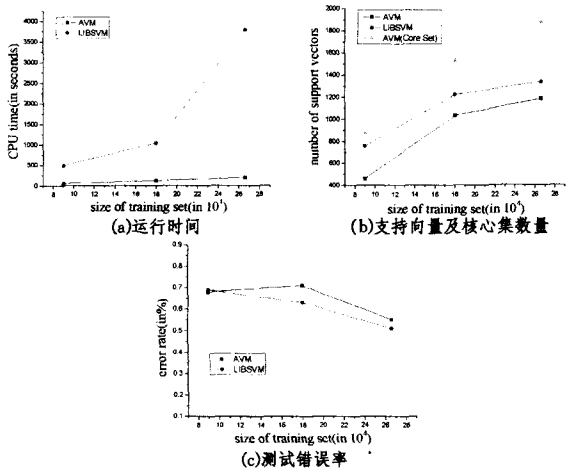

图3 AVM和LIBSVM在扩展USPS数据集上的运行结果比较（采用多项式核函数）

结束语 本文提出了一种适用于超大规模数据集的SVM学习算法：AVM，它采用增量学习和近似最优解相结合的方法来解决SVM在超大规模数据集上的训练问题。理论分析和实验结果表明，该算法的计算复杂度与训练样本的数量无关，因此具有良好的时间与空间扩展性。和目前存在的适用于超大规模数据集的SVM训练算法相比，AVM具有如下优势：

1) AVM算法的设计思路并不是单纯依靠经验观察，而是具有坚实的理论基础。

2) AVM算法具有更快的训练速度，并且训练完毕具有更少的支持向量，因此相应的结果分类器具有更快的分类速度。

3) AVM算法对采用何种核函数和损失函数没有限制，因此更易获得稳定的分类性能和具有更宽广的适用范围。

目前的AVM算法只适用于分类问题，将其推广到回归问题是今后研究工作的一部分。

参考文献

[1] Vapnik V. Statistical Learning Theory[M]. New York: Wiley, 1998

[2] 边肇祺，张学工，等. 模式识别（第二版）[M]. 北京：清华大学出版社，2000：284-303

[3] Yu H, Yang J, Han J. Classifying large data sets using SVM with hierarchical clusters[C]// Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Washington DC, USA, 2003: 306-315

[4] Erdem Z, Polikar R, Gurgen F S, et al. Ensemble of SVMs for Incremental Learning. In Multiple Classifier Systems, 2005: 246-256

[5] Tsang I W, Kwok J T, Cheung P. Core vector machines : Fast svm training on very large data sets[J]. JMLR, 2005, 6: 363-392

[6] 朱永生，王成栋，张优云. 二次损失函数支持向量机性能的研究[J]. 计算机学报，2003，26(8)：982-989

[7] Badoiu M, Clarkson K. Optimal core - sets for balls [C]// DIMACS Workshop on Computational Geometry. 2002

[8] Har-Peled S, Roth D, Zimak D. Maximum Margin Coresets for Active and Noise Tolerant Learning[C]// Proceedings of the twentieth International joint Conference on Artificial Intelligence. Hyderabad, India, 2007

[9] Li Xuchun, Zhu Yan, Sung E. Sequential bootstrapped support vector machines[J]. IEEE Trans. Neural Netw, 2005, 10(5): 1000-1017

[10] Chang C-C, Lin C-J. LIBSVM: a library for support vector machines[OL]. 2001. http://www.csie.ntu.edu.tw/?cjlin/libsvm

[11] Murphy P M, Aha D W. UCI repository of machine learning databases. Irvine, CA[OL]. http://www.ics.uci.edu/~mlearn/MLRepository.html), 2004

[12] Agarwal P, Har - Peled S, Varadajan K. Geometric approximations via coresets. Manuscript[OL]. http://valis.cs.uiuc.edu/~sariel/papers/04/survey/, 2004

（上接第207页）

[5] Cheung D W, et al. Efficient Mining of Association Rules in Distributed Databases[J]. IEEE Trans. Knowledge and Data Eng., 1996, 8(6): 911-922

[6] Schuster A, Wolff R. Communication-efficient Distributed Mining of Association Rules[C]// Proc. ACM SIGMOD Int'l Conf. Management of Data. ACM Press, 2001: 473-484

[7] Ashrafi M Z. Monash University ODAM: An Optimized Distributed Association Rule Mining Algorithm, IEEE DISTRIBUTED SYSTEMS ONLINE 1541-4922 © 2004[J]. IEEE Computer Society, 2004, 5(3)

[8] Ma Y, Liu B, Wong C K. Web for Data Mining: Organizing and Interpreting the Discovered Rules Using the Web[J]. SIGKDD Explorations, New York: ACM Press, 2000, 2(1): 16-23

[9] Kimball R, Ross M. The Data Warehouse Toolkit, The Complete Guide to Dimensional Modeling. 2nd edn[M]. John Wiley & Sons, New York, 2002

[10] Nestorov S, Jukic N. Ad - Hoc Association - Rule Mining within the Data Warehouse[C]// Abstract Proc. Hawaii Int. Conf. on System Sciences (HICSS'03). IEEE Computer Society Press (2003) 232. http://people.cs.uchicago.edu/~evtimov/pubs/hicss03.pdf

[11] Zaky M J, Parthasarathy S, Ogihara M, et al. New Algorithm for Fast Discovery of Association Rules[R]. No. 261. University of Rochester, 1997

[12] Hand D, Manilla H, Smyth P. Principles of Data Mining[M]. Cambridge-London: MIT Press, 2001