Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Sciences 165 (2004) 1–20
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

An efficient strategy for mining exceptions in multi-databases

Shichao Zhang [a,b,*], Chengqi Zhang [a], Jeffrey Xu Yu [b]

[a] *Faculty of Information Technology, University of Technology, Sydney, P.O. Box 123, Broadway, NSW 2007, Australia*

[b] *Department of Systems Engineering and Engineering Management, Chinese University of Hong Kong, Shatin, New Territories, Hong Kong*

Received 27 July 2003; received in revised form 6 October 2003; accepted 16 October 2003

Abstract

This paper proposes a new strategy, referred to as local instance analysis, for multi-database mining. While many interstate organizations have an imperative need to analyze their data in multi-databases distributed throughout their branches, traditional multi-database mining utilizes the strategies for mono-database mining: pooling all the data from relevant databases into a single dataset for discovery. This leads to the destruction of useful information, for instance, '70% of branches within a company agreed that a married customer usually has at least 2 cars if his/her age is between 45 and 65'. This information assists in global decision-making within the company. Our new strategy is developed for discovering this useful information. Using the local instance analysis, we design an algorithm for identifying exceptions from multi-databases. Exceptional pattern reflects the 'individuality' of, say, branches of an interstate company.

* Corresponding author. Address: Faculty of Information Technology, University of Technology, Sydney, P.O. Box 123, Broadway, NSW 2007, Australia. Fax: +61-2-9514-1807.

E-mail addresses: zhangsc@it.uts.edu.au (S. Zhang), chengqi@it.uts.edu.au (C. Zhang), yu@se.cuhk.edu.hk (J.X. Yu).

0020-0255/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2003.10.008

1. Introduction

The advance of multi-database technology, such as computer communication networks and distributed database systems leads to many multi-database systems having been developed in real-world applications. For example, the French Teletel system has 1500 separate databases [6]. So, many organizations need to mine their multi-databases distributed in branches for the purpose of decision-making.

Traditional multi-database mining puts all the data together from relevant databases to amass a huge dataset for discovery upon which mono-database mining strategies can be used. However, this can destroy important information that reflects certain distributions of patterns. This information is useful in decision-making within a company. Therefore, using mono-database mining techniques are inadequate for multi-database mining. This is because there are essential differences between mono- and multi-database mining. So the efforts in tackling multi-database mining problems will greatly impact on both industry and academia.

A pattern is called a *high-vote pattern* if the pattern is supported or voted for by most of the branches within a company. While high-voting patterns are useful when a company is reaching common decisions, headquarters are also interested in viewing *exceptional patterns* used when special decisions are made at only a few of the branches, perhaps for predicting the sales of a new product. Exceptional patterns reflect the individuality of branches. This paper focuses on identifying exceptional patterns in multi-databases by analyzing local instances.

Unlike high-voting patterns, however, exceptional patterns can be hidden in local instances. Exceptional patterns reflect the 'individuality' of, say, branches of an interstate company. In real-world applications, exceptional patterns are often presented as more glamorous than high-vote patterns in such areas as marketing, science discovery, and information safety.

This paper proposes a new strategy, referred to as the local instance analysis, for identifying exceptional patterns from multi-databases. We begin with presenting the problem statement in Section 2. Section 3 proposes a local instance analysis. In Section 4, a model for identifying exceptional patterns is proposed. In Section 5, an algorithm is designed for searching exceptional patterns. Section 6 evaluates the proposed approach by the use of experiments. Finally, this paper is summarized in Section 7.

2. Problem statement

For description, this section states the multi-database mining problem in a simple way.

2.1. Related work

Data mining techniques (see [1,12]) have been successfully used in many diverse applications. These include medical diagnosis and risk prediction, credit-card fraud detection, computer security break-in and misuse detection, computer user identity verification, aluminum and steel smelting control, pollution control in power plants and fraudulent income tax return detection. Developed techniques are oriented towards mono-databases.

Multi-database mining has been recently recognized as an important research topic in the KDD community. One article [16] proposed a means of searching for interesting knowledge in multiple databases according to a user query. The process involves selecting all interesting information from many databases by retrieval. Mining only works on the selected data.

Liu et al. [7] proposed another mining technique in which relevant databases are identified. Their work has focused on the first step in multi-database mining, which is the identification of databases that are most relevant to an application. A relevance measure was proposed to identify relevant databases for mining with an objective to find patterns or regularity within certain attributes. This can overcome the drawbacks that are the result of forcedly joining all databases into a single very large database upon which existing data mining techniques or tools are applied. However, this database classification is typically database-dependent. Therefore, Zhang and Zhang have proposed a database-independent database classification in [15], which is useful for general-purpose multi-database mining.

Zhong et al. [17] proposed a method of mining peculiarity rules from multiple statistical and transaction databases based on previous work. A peculiarity rule is discovered from peculiar data by searching the relevance among the peculiar data. Roughly speaking, data is peculiar if it represents a peculiar case described by a relatively small number of objects and is very different from other objects in a data set. Although it appears to be similar to the exception rule from the viewpoint of describing a relatively small number of objects, the peculiarity rule represents the well-known fact with common sense, which is a feature of the general rule.

Other related research projects are now briefly described. Wu and Zhang advocated an approach for identifying patterns in multi-database by weighting [14]. Ref. [10] described a way of extending the INLEN system for multi-database mining by incorporating primary and foreign keys as well as developing and processing knowledge segments. Wrobel [13] extended the concept of foreign keys to include foreign links since multi-database mining also involves accessing non-key attributes. Aronis et al. [3] introduced a system called WoRLD that uses spreading activation to enable inductive learning from multiple tables in multiple databases spread across the network. Existing parallel mining techniques can also be used to deal with multi-databases [2,4,5,8,9,11].

The above efforts provide a good insight into multi-database mining. However, there are still some limitations in traditional multi-database mining that are discussed in next subsection.

2.2. Limitations of previous multi-database mining

As we have seen traditional multi-database mining is fascinated with mono-database mining techniques. It consists of a two-step approach. The first step is to select the databases most relevant to an application. All the data is then pooled together from these databases to amass a huge dataset for discovery upon which mono-database mining techniques can be used. However, there are still some limitations discussed below.

1. Putting all the data from relevant databases into a single database can destroy some important information that reflect the distributions of patterns. The statement "85% of the branches within a company agree that a customer usually purchases sugar if he/she purchases coffee" is an example of such a piece of information. These patterns may be more important than the patterns present in the mono-database in terms of global decision-making within a company. Hence, existing techniques for multi-databases mining are inadequate for applications.

   In some contexts, each branch of an interstate company, large or small, has equal power in voting patterns for global decisions. For global applications, it is natural for the company headquarters to be interested in the patterns voted for by most of the branches. It is therefore inadequate in multi-database mining to utilize existing techniques for mono-databases mining.
2. Collecting all data from multi-databases can amass a huge database for centralized processing using parallel mining techniques.

   It may be an unrealistic proposition to collect data from different branches for centralized processing because of the huge data volume. For example, different branches of Wal-Mart receive 20 million transactions a day. This is more than the rate at which data can be feasibly collected and analyzed using today's computing power.
3. Because of data privacy and related issues, it is possible that some databases of an organization may share their patterns but not their original databases.

   Privacy is a very sensitive issue, and safeguarding its protection in a multi-database is of extreme importance. Most multi-database designers take privacy very seriously, and allow some protection facility. For source sharing in real-world applications, sharing patterns is a feasible way of achieving this.

From the above observations, it is clear that traditional multi-database mining is inadequate to serve two-level applications of an interstate company. This prompts the need to develop new techniques for multi-database mining.

2.3. Our approach

This paper develops a *local instance analysis*, inspired by competition in sport, for identifying exceptional patterns of interest in multi-databases. Exceptional patterns are of the form such as '15% of 40 supermarket branches strongly support that sales increased 9% when chili is frequently purchased'. The number '15%' is the ratio of branches that support the pattern 'sales increased 9% when chili is frequently purchased'. The number '40' is the number of relevant branches that are supermarkets.

To avoid re-mining multiple databases, local instance analysis will firstly be advocated in this paper (see Section 3). To search useful exceptional patterns, two metrics are then advocated for measuring the interestingness of patterns (see Section 4). One of the interestingness measures considers the deviation of the voting rate of a pattern from the average voting rate. Another measure considers the supporting strength of the pattern in branches that support or vote for the patterns.

Based on the above analysis, the problem for our research can be formulated as follows.

> Let $D_1, D_2, \ldots, D_m$ be m databases in the m branches $B_1, B_2, \ldots, B_m$ of a company, respectively; and LI_i be the set of patterns (local instances) from D_i $(i = 1, 2, \ldots, m)$. We are interested in the development of new techniques for identifying exceptional patterns of interest in the local instances.

Our model in this paper is the first research effort in this direction because traditional multi-database mining, which puts all the data together from relevant databases to amass a huge dataset for discovery, cannot identify exceptional patterns in multi-databases.

3. Local instance analysis

A *local instance* is a *local pattern* that is identified in the local database of a branch. Therefore, a local pattern may be a frequent itemset, an association rule, causal rule, dependency, or some other expression.

Local instance analysis is to explore techniques and methods for identifying laws, rules, and useful patterns from a set of local instances in an interstate company. To recognize patterns in multi-databases, we now demonstrate, in a simple way, the structure of a pattern.

In a multi-database environment, a pattern has attributes: the name of the pattern, the rate voted for by branches, and supports (and confidences for a rule) in branches that vote for the pattern. In other words, a pattern is a *super-point* of the form

$$P(name, vote, vsupp, vconf)$$

where,

- *name* is the dimension of the name of the pattern;
- *vote* is the dimension of the voted rate of the pattern;
- *vsupp* is a vector that indicates the m dimensions of supports in m branches, referred to as the support dimensions, and
- *vconf* is a vector that indicates the m dimensions of confidences in m branches, referred to as the confidence dimensions.

Without loss of generality, patterns are taken as itemsets in this subsection. Consequently, a pattern is of the form $P(name, vote, vsupp)$, which is the projection of $P(name, vote, vsupp, vconf)$ on *name*, *vote* and *vsupp*. Patterns in multi-dimension space are depicted in Fig. 1.

Fig. 1 illustrates the properties of a pattern in multi-dimensional space, where *Pattern* stands for the dimension of *name*; *vote* stands for the dimension of the voting rate of a pattern; and $Branch_i$ $(1 \leqslant i \leqslant m)$ stands for the m dimensions of supports in m local instance sets. For a global pattern P_1, there is a super-point describing the pattern, where the projection of the super-point on

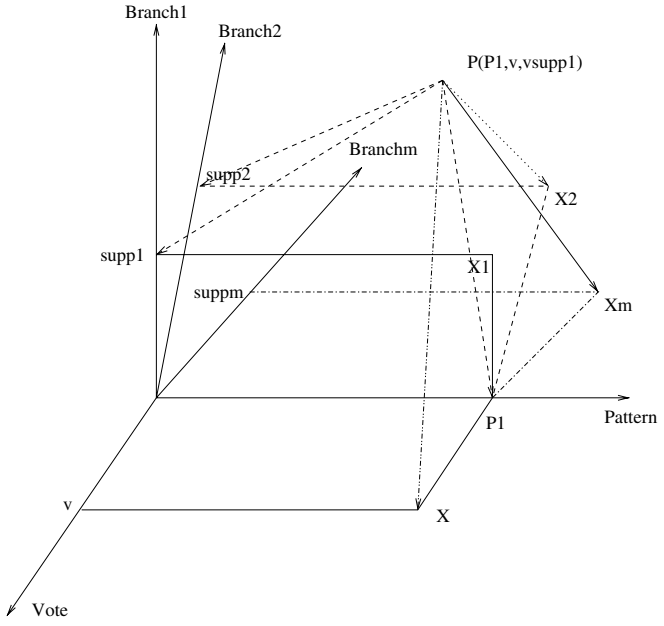

Fig. 1. Representation of patterns in multi-dimension space.

Pattern-Vote plane is X, indicating the voting rate v; the projection of the super-point on *Pattern-Branch*$_1$ plane is X_1, indicating the support $supp_1$ in the first branch; the projection of the super-point on *Pattern-Branch*$_2$ plane is X_2, indicating the support $supp_2$ in the second branch; ...; the projection of the super-point on *Pattern-Branch*$_m$ plane is X_m, indicating the support $supp_m$ in the mth branch.

Consider a company that has five branches with five databases $D_1, D_2, \ldots, D_5$ as follows. (Without loss of generality, let each branch have a database and the database be a relation or table.)

$$D_1 = \{(A, B, C, D); (B, C); (A, B, C); (A, C)\}$$
$$D_2 = \{(A, B); (A, C); (A, B, C); (B, C); (A, B, D); (A, C, D)\}$$
$$D_3 = \{(B, C, D); (A, B, D); (B, C); (A, B, D); (A, B)\}$$
$$D_4 = \{(A, C, D); (A, B, C); (A, C); (A, D); (D, C)\}$$
$$D_5 = \{(A, B, C); (A, B); (A, C); (A, D)\}$$

where each database has several transactions, separated by a semicolon; each transaction contains several items, separated by commas.

For databases $D_1, D_2, \ldots, D_5$, patterns A, B, C, D, and AB are represented as follows

$$P(A, 1.0, (0.75, 0.833, 0.6, 0.8, 1.0))$$
$$P(B, 0.8, (0.75, 0.667, 1.0, 0.2, 0.5))$$
$$P(C, 0.8, (1.0, 0.667, 0.4, 0.8, 0.5))$$
$$P(D, 0.4, (0.25, 0.333, 0.6, 0.6, 0.25))$$
$$P(AB, 0.8, (0.5, 0.5, 0.6, 0.2, 0.5))$$

Let $minsupp = 0.5$. In local instance analysis, patterns A, B, C, D, and AB are really represented as

$$P(A, 1.0, (0.75, 0.833, 0.6, 0.8, 1.0))$$
$$P(B, 0.8, (0.75, 0.667, 1.0, 0.0, 0.5))$$
$$P(C, 0.8, (1.0, 0.667, 0.0, 0.8, 0.5))$$
$$P(D, 0.4, (0.0, 0.0, 0.6, 0.6, 0.0))$$
$$P(AB, 0.8, (0.5, 0.5, 0.6, 0.0, 0.5))$$

That is, the ith support of a pattern is 0.0, if the ith branch does not vote for the pattern.

To identify patterns from local instances, the projection of $P(name, vote, vsupp, vconf)$ on *name*, *vote* and *vsupp* is considered, i.e., the projection

$$P(name, vote, vsupp)$$

is used to search exceptional patterns of interest in the rest of this paper.

4. Identifying exceptional patterns of interest

To avoid re-mining multiple databases, local instance analysis was advocated in Section 2. This section proposes a model for identifying, from local instances, another kind of new pattern, referred to as exceptional patterns. Exceptional patterns reflect the individuality of branches of an interstate company. They are useful in making decisions for individual branches.

Indeed, exceptional patterns are often presented as more glamorous than high-voting patterns in such areas as marketing, science discovery, and information safety. For example, '20% of 10 toy branches strongly supported the new toy "Mulan" which was purchased with rather high frequency'. These local instances can be used to analyze the possible purchasing trends, although 'Mulan" has a low-voting rate.

As we have said, exceptional patterns can grasp the individuality of branches. This subsection presents models for measuring the interestingness of such patterns.

To identify exceptional patterns of interest from local instances, the projection of $P(name, vote, vsupp, vconf)$ on *name*, *vote* and *vsupp* is considered, i.e., the projection

$$P(name, vote, vsupp)$$

is considered. The table of frequencies of patterns voted for by the branches of an interstate company for local instances is constructed in Table 1.

In Table 1, B_i is the ith branch of an interstate company ($1 \leqslant i \leqslant m$); $a_{i,j} = 1$ if branch B_i votes for pattern r_j (it says, r_j is a valid pattern in branch B_i), $a_{i,j} = 0$ if branch B_i does not vote for pattern r_j (it says, r_j is not a valid pattern in branch B_i) ($1 \leqslant i \leqslant m$ and $1 \leqslant j \leqslant n$); and $voting_i$ is the number of branches that vote for the ith patterns ($1 \leqslant i \leqslant m$).

From Table 1, the average voting rate can be obtained as

$$AverageVR = \frac{voting(r_1) + voting(r_2) + \cdots + voting(r_n)}{n}$$

where $voting(r_j) = voting_j/m$, which is the voting ratio of r_j.

If the voting rate of a pattern is less than *AverageVR*, the pattern might be an exceptional pattern. This means that interesting exceptional patterns are hidden in low-voting patterns. To measure the interestingness of an exceptional pattern, r_j, its voting rate and its supports in branches must be considered. In

Table 1
Frequencies of patterns voted for by branches of an interstate company

	r_1	r_2	...	r_n
B_1	$a_{1,1}$	$a_{1,2}$	...	$a_{1,n}$
B_2	$a_{2,1}$	$a_{2,2}$	...	$a_{2,n}$
...	...	...		...
B_m	$a_{m,1}$	$a_{m,2}$	...	$a_{m,n}$
Voted Number	$voting_1$	$voting_2$	...	$voting_n$

order to reflect the factors, two metrics for the interestingness are constructed below.

The first metric considers the relationship between the voting rate $voting(r_j)$, and the average voting rate $AverageVR$. If $voting(r_j) < AverageVR$, the pattern r_j refers to a low-voting pattern. In this case, $voting(r_j) - AverageVR$ satisfies:

$$-AverageVR \leqslant voting(r_j) - AverageVR < 0$$

In particular, we have

$$0 < \frac{voting(r_j) - AverageVR}{-AverageVR} \leqslant 1$$

Certainly, the bigger the ratio $(voting(r_j) - AverageVR)/(-AverageVR)$, the more interesting the pattern.

Consequently, one of the interest measures, $EPI(r_j)$, of a pattern, r_j, is defined as the deviation of the voting rate, $voting(r_j)$, from the average voting rate, $AverageVR$.

$$EPI(r_j) = \frac{voting(r_j) - AverageVR}{-AverageVR}$$

for $AverageVR \neq 0$, while $EPI(r_j)$ is referred to as the interestingness of r_j, given $AverageVR$.

From the above interest measure, $EPI(r_j)$ is negatively related to the voting ratio of the pattern r_j. It is highest if the voting ratio is 0. A pattern r_j in local instances is an exceptional pattern if its interestingness measure $EPI(r_j)$ is greater than or equal to the threshold minimum interest degree ($minEP$) given by users or experts.

Because an exceptional pattern is a low-voting pattern, it cannot be a valid pattern in the whole dataset that is the union of a class of databases. However, exceptional patterns are those that are strongly supported by a few of branches. That is, exceptional patterns reflect the individuality of such branches. Consequently, the second metric considers the support dimensions of a pattern.

From local instances, the supports of patterns in branches of an interstate company are listed in Table 2.

In Table 2, B_i is the ith branch of the interstate company ($1 \leqslant i \leqslant m$), and $supp_{i,j} \neq 0$ stands for the support of pattern r_j in branch B_i, when r_j is a valid pattern in B_i. If $supp_{i,j} = 0$, then r_j is not a valid pattern in branch B_i ($1 \leqslant i \leqslant m$ and $1 \leqslant j \leqslant n$). Meanwhile, $minsupp_i$ is the minimum support used to identify local patterns within the ith branch.

A pattern, r_j, in local instances is exceptional if its supports in a few of branches are pretty high. How large can a support be referred to as *high*? In the case of 0.5 and 0.2, 0.5 is obviously higher than 0.2. However, the supports of a pattern are collected from different branches, and local databases in branches may differ in size. For example, let 100 and 100000 be the number of transactions in databases D_1 and D_2 respectively, and 0.5 and 0.2 be the supports of r_j in D_1 and D_2 respectively. For these supports of r_j, the support 0.5 of r_j in D_1 cannot be simply regarded as high, and the support 0.2 of r_j in D_2 cannot be simply regarded as low.

Consequently, a reasonable measure for high support should be constructed. Though it is difficult to deal with raw data in local databases, the minimum supports in branches are suitable references for determining which of the supports of a pattern are high.

Generally, the higher the support of a pattern in a branch, the more interesting the pattern. Referenced with the minimum support $minsupp_i$ in branch B_i, the interestingness of a pattern r_j in B_i is defined as

$$RI_i(r_j) = \frac{supp_{i,j} - minsupp_i}{minsupp_i}$$

where $supp_{i,j}$ is the support of r_j in branch B_i.

From the above measure, $RI_i(r_j)$ is positively related to the support of r_j in branch B_i. It is highest if the support is 1. The pattern r_j in B_i is of interest if RI_i is greater than or equal to a threshold: the minimum interest degree (*minEPsup*) given by users or experts.

Example 1. Consider the supports of patterns in four branches given in Table 3.

Table 2
Supports of patterns in branches of an interstate company

	r_1	r_2	...	r_n	*minsupp*
B_1	$supp_{1,1}$	$supp_{1,2}$	...	$supp_{1,n}$	$minsupp_1$
B_2	$supp_{2,1}$	$supp_{2,2}$	...	$supp_{2,n}$	$minsupp_2$
...	...	...		...	...
B_m	$supp_{m,1}$	$supp_{m,2}$	...	$supp_{m,n}$	$minsupp_m$

Table 3
Supports of patterns in four branches of an interstate company

	r_1	r_2	r_3	r_4	*minsupp*
B_1	0.021	0.012	0.018	0	0.01
B_2	0	0.6	0	0.3	0.25
B_3	0	0	0.8	0	0.4
B_4	0.7	1	0	0	0.5

For pattern r_1, it is a valid pattern in branches B_1 and B_4 and not a valid pattern in branches B_2 and B_3. The interesting degrees $RI_1(r_1)$ and $RI_4(r_1)$ of r_1 in B_1 and B_4 are as follows

$$RI_1(r_1) = \frac{supp_{1,1} - minsupp_1}{minsupp_1} = \frac{0.021 - 0.01}{0.01} = 1.1$$

$$RI_4(r_1) = \frac{supp_{4,1} - minsupp_4}{minsupp_4} = \frac{0.7 - 0.5}{0.5} = 0.4$$

This means that the interestingness of r_1 in B_1 is higher than that in B_4 though $supp_{1,1} = 0.021 < supp_{4,1} = 0.7$.

For other patterns, their interesting degrees are shown in Table 4.

In Table 4, '–' stands for a pattern not being voted for by a branch.

From the above discussion, an exceptional pattern r is of interest if

(C1) $EPI(r) \geqslant minEP$; and
(C2) $RI_i(r) \geqslant minEPsup$ for branches that vote for r.

The problem of finding exceptional patterns can now be stated as follows: given a set of local instance sets, *LIset*, find all patterns that satisfy both the conditions (C1) and (C2).

Table 4
Interestingness of patterns in branches

	r_1	r_2	r_3	r_4	*minsupp*
B_1	1.1	0.2	0.8	–	0.01
B_2	–	1.4	–	0.2	0.25
B_3	–	–	1	–	0.4
B_4	0.4	1	–	–	0.5

5. Algorithm designing

Below, we design an algorithm, *ExceptionalPatterns*, for identifying interesting exceptional patterns from local instances. Then, in the next section, an example will be used to illustrate how to search exceptional patterns from local instances by way of algorithms.

Algorithm 1 *ExceptionalPatterns*
begin
Input: LI_i $(1 \leqslant i \leqslant M)$: *sets of local instances*,
minEP: *threshold value that is the minimal interest degree for the voting ratios of exceptional patterns;*
minEPsupp: *threshold value that is the minimal interest degree for the supports of exceptional patterns in branches;*
Output: *EPattern*: *set of exceptional patterns of interest;*
(1) **generate** a classification *class* for M local instances LI_i;
(2) **let** $EPattern \leftarrow \{\ \}$;
(3) **for** *each* $class_i$ in *class* **do**
begin
(3.1) **let** $EPattern_i \leftarrow \{\ \}$; $giveup_i \leftarrow \{\ \}$;
let pattern set $E \leftarrow \emptyset$;
(3.2) **for** a local instance set LI in $class_i$ **do**
for each pattern r in LI **do**
if $(r \notin giveup_i)$ and $(RI_{LI}(r) < minEPsup)$ **then**
if $r \in E$ **then**
let $E \leftarrow E - \{r\}$; $giveup_i \leftarrow giveup_i \cup \{r\}$;
else
if $r \in E$ **then**
//$r.count$ is the counter of r.
let $r.count \leftarrow r.count + 1$;
else
begin
let $E \leftarrow E \cup \{r\}$;
let $r.count \leftarrow 1$;
end
(3.3) **for** each pattern r in E **do**
if $EPI(r) \geqslant minEP$ **then**
let $EPattern_i \leftarrow EPattern_i \cup \{r\}$;
(3.4) **let** $EPattern \leftarrow EPattern \cup \{EPattern_i\}$;
end
(4) **output** the exceptional patterns in *EPattern*;
end;

The algorithm *ExceptionalPatterns* is to search all interesting exceptional patterns from given M local instance sets.

Step (1) finds a classification *class* for the M local instance sets (from M databases) using our database clustering in [15]. For the above *class*, we need to search exceptional patterns for classes one by one. The set of the interesting exceptional patterns found in a class is taken as an element of *EPattern*. Step (2) initializes the set *EPattern* to be an empty set.

Step (3) is to handle the instances for each class $class_i$ in *class*. Step (3.1) initializes the set variables $EPattern_i$, $giveup_i$ and E. $EPattern_i$ is used to save all exceptional patterns in $class_i$, and $giveup_i$ is used to save the patterns that are given up. E is used to save all patterns in $class_i$. Step (3.2) is to sum up the frequency for a pattern r that satisfies both $r \notin giveup_i$ and $RI_{LI}(r) \geqslant minEPsup$ in the local instance set of $class_i$. Step (3.3) generates all interesting exceptional patterns from the set E and saves them in $EPattern_i$, where each pattern r in $EPattern_i$ is with $EPI(r) \geqslant minEP$ in the local instance set of $class_i$. Step (3.4) is to append $EPattern_i$ into *EPattern*.

Step (4) outputs all exceptional patterns in the given local instance sets class by class.

5.1. An example

Consider the five local instance sets $LI_1, LI_2, \ldots, LI_5$ below identified from branches $B_1, B_2, \ldots, B_5$, respectively.

$$LI_1 = \{(A, 0.34); (B, 0.45); (C, 0.47); (D, 0.56); (F, 0.8)\}$$
$$LI_2 = \{(A, 0.35); (B, 0.45); (C, 0.46)\}$$
$$LI_3 = \{(A, 0.05); (B, 0.02); (AB, 0.02)\}$$
$$LI_4 = \{(A, 0.38); (B, 0.45); (C, 0.65); (F, 0.68)\}$$
$$LI_5 = \{(A, 0.22); (C, 0.3); (D, 0.61)\}$$

where each local instance set has several patterns, separated by a semicolon, and each pattern consists of its name and its support in a branch, separated by a comma.

Let $minsupp_1 = 0.3$, $minsupp_2 = 0.3$, $minsupp_3 = 0.008$, $minsupp_4 = 0.25$, $minsupp_5 = 0.2$, $minEP = 0.5$, and $minEPsup = 0.75$.

We now use the algorithm *ExceptionalPatterns* to search all exceptional patterns from the five local instance sets. There is one class $class_1 = \{LI_1, LI_2, LI_3, LI_4, LI_5\}$ for the given local instance sets.

For $class_1$, the interesting degrees of patterns are shown in Table 5.

Because $minEPsup = 0.75$, patterns D, F, and AB are selected according to Step (3.2) in the algorithm *ExceptionalPatterns*. For $class_1$, $AverageVR = 0.6$, and for D, F, and AB

Table 5
Interestingness of patterns in $class_1$

	A	B	C	D	F	AB	$minsupp$
LI_1	0.133	0.5	0.567	0.867	1.667	–	0.3
LI_2	0.167	0.5	0.533	–	–	–	0.3
LI_3	5.25	1.5	–	–	–	1.5	0.008
LI_4	0.52	0.8	1.6	–	1.72	–	0.25
LI_5	0.1	–	0.5	2.05	–	–	0.2

$$EPI(D) = \frac{voting(D) - AverageVR}{-AverageVR} = \frac{0.4 - 0.6}{-0.6} \approx 0.333$$

$$EPI(F) = \frac{voting(F) - AverageVR}{-AverageVR} = \frac{0.4 - 0.6}{-0.6} \approx 0.333$$

$$EPI(AB) = \frac{voting(AB) - AverageVR}{-AverageVR} = \frac{0.2 - 0.6}{-0.6} \approx 0.667$$

According to $minEP = 0.5$, we can obtain all exceptional patterns in the set *EPattern*. And *EPattern* consists of one element: $EPattern_1 = \{(AB, 0.2, (0, 0.25, 0, 0, 0))\}$. There is only one interesting exceptional pattern $P(AB, 0.2, (0, 0.25, 0, 0, 0))$ in the given local instance sets.

5.2. Algorithm analysis

The algorithm *ExceptionalPatterns* identifies all interesting exceptional patterns from local patterns. Therefore, we have a theorem for the procedure as follows.

Theorem 1. *The Algorithm ExceptionalPatterns works correctly.*

Proof. Clearly, in Steps (3) and (4), a set *EPattern* of interesting exceptional patterns is generated, as it is output for given local pattern sets. We need to show that all interesting exceptional patterns are identified, and all uninteresting exceptional patterns are given up.

For any local pattern r_j in a class, all patterns that satisfy the condition (C2) are selected in the loop in Step (3.2) and, if a pattern does not satisfy the condition (C2), the pattern is not appended to E.

Again, if $EPI(r_j) \geqslant minEP$ in the loop in Step (3.3), it is an interesting exceptional pattern, and the pattern is appended to *EPattern*. Otherwise, it is not appended to *EPattern*. This means that all interesting exceptional patterns are identified, and all uninteresting exceptional patterns are given up. □

In the algorithm *ExceptionalPatterns*, Step (1) generates database clusters. The complexities have been discussed in [15]. So, the complexity of *ExceptionalPatterns* can be regarded as consisting of Steps (3) and (4).

In Step (3), $m = |class^{\alpha}|$ classes are searched for. Assume that n_i is the number of local patterns in the class, $class_i$. We need

$$n_1 * |class_1| + n_2 * |class_2| + \cdots + n_m|class_m|$$

units to save all the local patterns, which are less than, or equal to, *mnl*, where $n = \{n_1, n_2, \ldots, n_m\}$, l is the maximum among $|class_1|, |class_2|, \ldots, |class_m|$, and $|class_i|$ stands for the number of local patterns in the class, $class_i$. Also, we need N units to save all the interesting exceptional patterns in $EPattern_1, \ldots, EPattern_m$, and *EPattern*. Obviously, $N \leqslant mn$. Consequently, the space complexity of *ExceptionalPatterns* is $O(mnl)$.

Apparently, the time complexity of *ExceptionalPatterns* is dominated by the loop in Step (3). Therefore, we have the following theorem.

Theorem 2. *The time complexity of ExceptionalPatterns is* $O(m^2nl)$.

Proof. For Step (3), m classes need to be processed to identify interesting exceptional patterns. For a class, $class_i$, there are $|class_i|$ local pattern sets and n_i local patterns. Each pattern needs to be checked as to whether the pattern can be extracted as an interesting exceptional pattern. Therefore, Step (3.2) needs $O(n_i * |class_i|)$ comparisons, and Step (3.3) needs $O(n_i)$ comparisons. Consequently, the time complexity of Step (4) is $m * (n_1 * |class_1| + n_2 * |class_2| + \cdots + n_m * |class_m|) \leqslant m^2nl$. This means that the time complexity of *ExceptionalPatterns* is $O(m^2nl)$. □

6. Experiments

To study the effectiveness of the approach proposed in this paper, a set of experiments was carried out on Dell, using Java. The experiments were designed to check whether the proposed approach can effectively identify interesting exceptional patterns.

6.1. Behavior of interest measurements

Before introducing the experiments, this subsection illustrates the behaviors of measurements $EPI(r)$ and $RI_i(r)$. As we have explained previously, an exceptional pattern r is of interest if $EPI(r) \geqslant minEP$ and $RI_i(r) \geqslant minEPsup$ for branches that vote for r. Figs. 2 and 3 demonstrate the behaviors of measurements $EPI(r)$ and $RI_i(r)$.

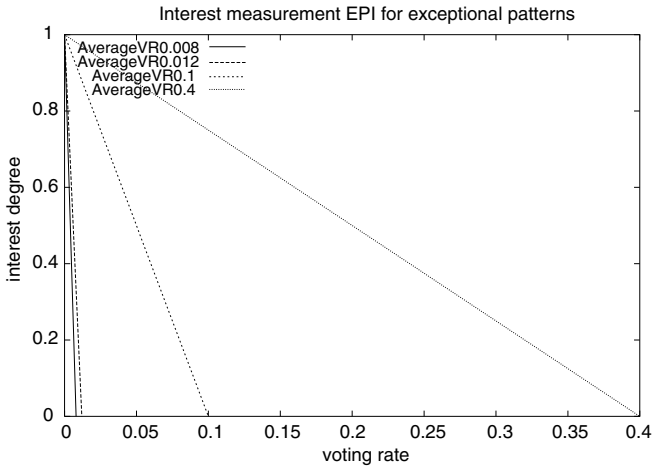

Fig. 2. Interest measurement $EPI(r_j)$ of exceptional pattern r_j.

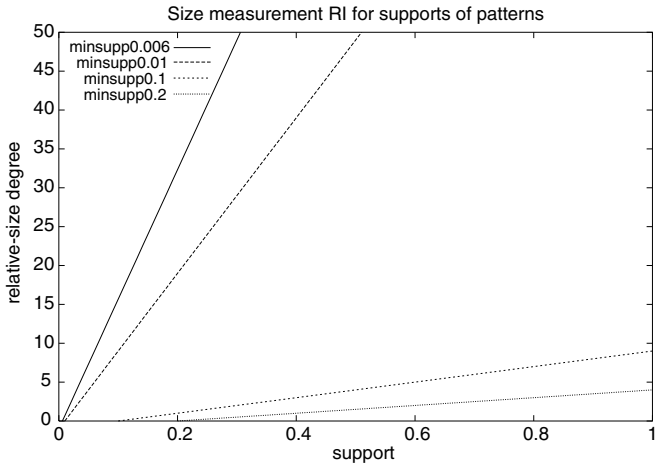

Fig. 3. Relative-size $RI(r_j)$ of the support pattern r_j in a branch.

6.2. Effectiveness of identifying interesting exceptions

In this subsection, we check whether the proposed approach can effectively identify interesting exceptional patterns.

From dataset selection, it is hoped that some interesting exceptional patterns will be found to exist in classes of databases. This can be done by changing the

names of some high-voting patterns in some local instance sets. For example, let A be a high-voting pattern that is voted for by 28 of 30 local instance sets. A is changed to X in 25 of the 28 local instance sets such that the supports of A in the remaining 3 local instance sets are very high.

To obtain multiple, possibly relevant, databases, the techniques in [7,16] were adopted. That is, a database was vertically partitioned into a number of subsets, each containing a certain number of attributes. It was hoped that some databases obtained are relevant to each other in classes. In this set of experiments, the databases are generated by databases from the Synthetic Classification Data Sets on the Internet (http://www.kdnuggets.com/). One of the experiments is demonstrated in the following, in which four databases were generated from the web site. The main properties of the four databases are as follows. There are $|R| = 1000$ attributes, and the average number T of attributes per row is 6. The number $|r|$ of rows is approximately 300 000. The average size I of maximal frequent sets is 4. The databases are vertically partitioned into 30 datasets. Table 6 summarizes the parameters for the databases.

To evaluate the proposed approach, the algorithm *ExceptionalPatterns* is used to generated all exceptional frequent-itemsets (patterns). The effectiveness and efficiency of the algorithm *ExceptionalPatterns* for identifying exceptional patterns are as follows. Table 7 shows the results for generating exceptional patterns.

In Table 7, '0.0015' is the minimum support used to measure frequent-itemsets when the datasets are mined. 'Ni' is the name of the ith group of datasets that are obtained from the ith database T6I4Ni. '5' is the number of

Table 6
Synthetic data set characteristics

Data set name	$\|R\|$	T	$\|r\|$	Number of subset
T6I4N1	1000	6	300256	30
T6I4N2	1000	6	299080	30
T6I4N3	1000	6	298904	30
T6I4N4	1000	6	301005	30

Table 7
Exceptional frequent-itemsets ($minEP = 0.40$)

Group name	$minsupp = 0.0015$	$minsupp = 0.001$
N1	5	5
N2	8	8
N3	15	15
N4	10	10

exceptional frequent-itemsets in 'N1' when $minsupp = 0.0015$ and $minEP = 0.4$, and '5' is the number of exceptional frequent-itemsets in 'N1' when $minsupp = 0.001$ and $minEP = 0.4$.

Fig. 4 depicts the distribution of exceptional frequent-itemsets in the four groups of datasets.

Table 8 shows the running time of *ExceptionalPatterns* when $minEP = 0.4$ for $minsupp = 0.0015$ and $minsupp = 0.001$.

Fig. 5 illustrates the running time for searching exceptional frequent-itemsets in the four groups of datasets when $minEP = 0.4$.

The results from the proposed approach for identifying exceptional patterns are promising. It is also evident from the experimental results that the proposed approach is very efficient due to the fact that it focuses on only local instances.

As we have argued previously, traditional multi-database mining techniques puts all the data from relevant databases into a single database for pattern discovery. This can destroy some important information such as the above

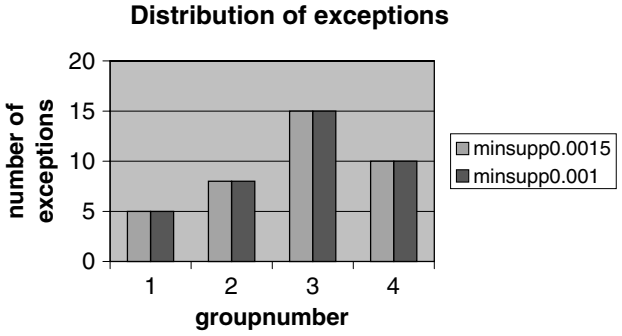

Fig. 4. Distribution of exceptional frequent-itemsets in the four groups of datasets when $minEP = 0.4$.

Table 8
Running time ($minEP = 0.4$)

Group name	$minsupp = 0.0015$	$minsupp = 0.001$
N1	13	15
N2	19	22
N3	15	18
N4	16	17

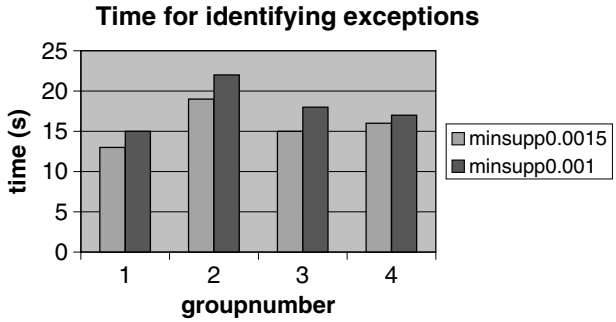

Fig. 5. Running time for identifying exceptional frequent-itemsets when $minEP = 0.4$.

exceptions. In other words, our multi-database mining technique can identify a new kind of patterns: exceptions.

7. Summary

Because putting all the data from (a class of) multi-databases into a single dataset can destroy important information, such as '10% of branches strongly support that the fact that a customer is likely to purchase pork if he/she purchase chili', this paper develops techniques for identifying novel patterns, known as exceptional patterns, in multi-databases by analyzing local instances. The main contributions of this paper are as follows.

(1) Constructed two measurements for the interestingness of exceptional patterns.
(2) Presented an approach for finding exceptional patterns in local instances of an interstate company.
(3) Designed an algorithm for searching interesting exceptional patterns from local instances.
(4) Evaluated the proposed techniques by experiments.

Unlike high-voting patterns, exceptional patterns are hidden in local instances. In real-world applications, exceptional patterns are often presented as more glamorous than high-voting patterns in such areas as marketing, science discovery, and information safety. However, traditional multi-database mining puts all the data together from relevant databases to amass a huge dataset for discovery upon which mono-database mining techniques can be used. This means that traditional multi-database mining techniques cannot identify exceptional patterns in multi-databases. Therefore, our model in this paper is the first research effort in this direction.

Acknowledgements

The authors would like to thank the anonymous reviewers for their detailed comments on the first version of this paper, which have helped improve the paper vastly. This research is partially supported by a large grant from the Australian Research Council (DP0343109) and partially supported by large grant from the Guangxi Natural Science Funds.

References

[1] R. Agrawal, T. Imielinski, A. Swami, Database mining: a performance perspective, IEEE Trans. Knowledge Data Eng. 5 (6) (1993) 914–925.
[2] R. Agrawal, J. Shafer, Parallel mining of association rules, IEEE Trans. Knowledge Data Eng. 8 (6) (1996) 962–969.
[3] J. Aronis et al., The WoRLD: Knowledge discovery from multiple distributed databases, Proceedings of 10th International Florida AI Research Symposium, 1997, pp. 337–341.
[4] J. Chattratichat et al., Large scale data mining: challenges and responses, in: Proceedings of KDD, 1997, pp. 143–146.
[5] D. Cheung, V. Ng, A. Fu, Y. Fu, Efficient mining of association rules in distributed databases, IEEE Trans. Knowledge Data Eng. 8 (6) (1996) 911–922.
[6] A. Hurson, M. Bright, S. Pakzad, Multidatabase systems: an advanced solution for global information sharing, IEEE Computer Society Press, 1994.
[7] H. Liu, H. Lu, J. Yao, Identifying relevant databases for multidatabase mining, in: Proceedings of Pacific–Asia Conference on Knowledge Discovery and Data Mining, 1998, pp. 210–221.
[8] A. Prodromidis, S. Stolfo, Pruning meta-classifiers in a distributed data mining system, in: Proceedings of the First National Conference on New Information Technologies, 1998, pp. 151–160.
[9] A. Prodromidis, P. Chan, S. Stolfo, Meta-learning in distributed data mining systems: issues and approaches, in: H. Kargupta, P. Chan (Eds.), Advances in Distributed and Parallel Knowledge Discovery, AAAI/MIT Press, 2000.
[10] J. Ribeiro, K. Kaufman, L. Kerschberg, Knowledge discovery from multiple databases, in: Proceedings of KDD95, 1995, pp. 240–245.
[11] T. Shintani, M. Kitsuregawa, Parallel mining algorithms for generalized association rules with classification hierarchy, in: Proceedings of ACM SIGMOD, 1998, pp. 25–36.
[12] G. Webb, Efficient search for association rules, in: Proceedings of ACM SIGKDD, 2000, pp. 99–107.
[13] S. Wrobel, An algorithm for multi-relational discovery of subgroups, in: J. Komorowski, J. Zytkow (Eds.), Principles of Data Mining and Knowledge Discovery, 1997, pp. 367–375.
[14] X. Wu, S. Zhang, Synthesizing high-frequency rules from different data sources, IEEE Trans. Knowledge Data Eng. 15 (2) (2003) 353–367.
[15] C. Zhang, S. Zhang, Database clustering for mining multi-databases, in: Proceedings of the 11th IEEE International Conference on Fuzzy Systems, Honolulu, Hawaii, USA, May 2002.
[16] J. Yao, H. Liu, Searching multiple databases for interesting complexes, in: Proceedings of the PAKDD, 1997, pp. 198–210.
[17] N. Zhong, Y. Yao, S. Ohsuga, Peculiarity oriented multi-database mining, in: Proceedings of PKDD, 1999, pp. 136–146.