Clustering Association Rules on Proximity-based Fuzzy Data

Been-Chian Chien
Department of Information Engineering
I-Shou University, Kaohsiung, Taiwan, R.O.C.
E-mail: cbc@csa500.isu.edu.tw

Shyue-Liang Wang
Department of Information Management
I-Shou University, Kaohsiung, Taiwan, R.O.C.
E-mail: slwang@csa500.isu.edu.tw

Abstract

Mining information or discovering knowledge from large databases has been identified as one of important research topics in database analysis and machine learning. The association rules are useful for determining the correlation between attributes in a relation. The problem of mining generalized association rules is to find all association rules from a set of items and a given taxonomy of items. The algorithm for finding generalized association rules is fixed for a taxonomy and does not suit the applications with different requirements. This paper examines the fuzzy proximity relations between the sets of attributes in databases, and shows that the extensional rules with proximity relations can be derived from the primitive association rules.

1: Introduction

Data mining, or the efficient knowledge discovery in large databases, has been recognized as an important area for database research. Discovering association rules is one of the most useful techniques in data mining. The problem of mining association rules was introduced in [1]. The association rules embedded in large databases can provide some useful knowledge among the underlying data and have been applied to the analyses of marketing and financial. An association rule has a general form of expression \( X \Rightarrow Y \), where \( X \) and \( Y \) are sets of items within the set of transactions in a database. The meaning of an association rule is that the transactions containing the items in \( X \) tend to contain the items in \( Y \), too. Each association rule has support and confidence. For rule \( X \Rightarrow Y \), the support is defined as the ratio of the number of transactions containing both \( X \) and \( Y \) and the total number of transactions in a database. The confidence is the ratio of the number of transactions containing both \( X \) and \( Y \) and the number of transactions that contain \( X \) only. The problem of mining association rules is to compute all association rules that satisfy a user-specified minimum support and minimum confidence.

Since the mining process for finding all association rules needs huge computation time, it is important for users to develop efficient mining methods. There are many approaches proposed for finding association rules efficiently, such as [1], [2], [9]. Other related problems of data mining include mining generalized association rules [10],[11], mining association rules over interval data [8] and discovering functional dependencies [4].

Generalized association rules are the association rules discovered from in a large database of transactions, which consist of a set of items and taxonomy (is-a hierarchy) on the items. In general cases, there are some relationships among the items in transactions. Taxonomy is the simplest one that considers hierarchy classes in a set of items. In the real world, relationships among the items in transactions are various from the different viewpoints and are uncertain in many applications. It is more complicated than the situation of hierarchical taxonomy usually. In this paper, we consider the fuzzy proximity relations on the sets of items in transactions and propose an algorithm for clustering association rules.
In next section, the generalized association rules and the fuzzy proximity relation are described in detail. In Section 3, we define the association rules that are appropriate for fuzzy proximity relation. The algorithm for discovering association rules is also presented. The final section concludes the results and outlines the future work.

2: The proximity relation in fuzzy data

For a large amount of transactions in a database, it is usually exists some relations between items in the transactions. For instance, there are two items, "milk" and "juice", in the database. Both of "milk" and "juice" are "drink." It is clear that the relation above is a hierarchical classification. The hierarchical class created by "is a kind of," is a taxonomy. The problem of mining generalized association rules proposed by Srikant and Agrawal [10] is to discover all rules that satisfy the user-specified minimum support and minimum confidence for given a set of transactions and a set of taxonomies. In [10], they proposed two efficient algorithms to prune the redundant rules using taxonomies. In real applications, the items are sometimes uncertain and difficult to classify clearly. For instance, a tomato is a kind of fruit and is a kind of vegetable. This situation comes up multiple taxonomy. Multiple taxonomy may be modeled as a DAG (directed acyclic graph). Here, we extend the relation of taxonomy to the fuzzy proximity relations.

We consider a fuzzy relation of compatibility on categorical data. A binary relation $R(S, S)$ on a set of items $S$ that is reflexive and symmetric fuzzy relations is called a proximity relation [5]. The proximity relation $R$ defines some compatibility classes with a specified membership degree $\alpha$. A $\alpha$-compatibility class is a subset $A$ of $S$ such that $R(x, y) \leq \alpha$ for all $x, y \in A$. For example, considering a fuzzy relation $R(S, S)$, where $S = \{A, B, C, D, E, F, G, H, I\}$ and the fuzzy membership relations are shown as the matrix in Fig. 1. All $\alpha$-compatibility classes can be found in Fig. 2. There are varied classes in different membership degree $\alpha$. As we see, there is only one category $T_1$ when $\alpha = 1$, and three categories $T_1, T_2$ and $T_3$ when $\alpha \geq 0.8$.

For transactions in a database, we can apply the similar approach of mining generalized association rules in [10] to find all association rules from a set of items $S$ in transactions and a given proximity relation $R(S, S)$. A straight idea is to compute the compatibility classes for a specified membership degree $\alpha$, and then find the association rules for the class hierarchy and item sets. This approach is time-consuming since it needs to be executed several times if users requested a few different membership degrees at the same time. To satisfy the requirements of various users for distinct membership degree $\alpha$ or even a new fuzzy proximity relation, the redundant computation for the set of items should be reduced. The improvements are developed in the next two sections.

$$
\begin{array}{cccccccc}
A & B & C & D & E & F & G & H & I \\
\hline
A & 1 & .8 & 0 & 0 & 0 & 0 & 0 & 0 \\
B & .8 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
C & 0 & 0 & 1 & 1 & .8 & 0 & 0 & 0 \\
D & 0 & 0 & 1 & 1 & .8 & .7 & .5 & 0 \\
E & 0 & 0 & .8 & .8 & 1 & .7 & .5 & .7 \\
F & 0 & 0 & 0 & .7 & .7 & 1 & .4 & 0 \\
G & 0 & 0 & 0 & .5 & .5 & .4 & 1 & 0 \\
H & 0 & 0 & 0 & .7 & 0 & 0 & 1 & 0 \\
I & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{array}
$$

Fig. 1 A proximity relation membership matrix

![Fig. 2 All $\alpha$-compatibility classes for $R$](image)

3: Native algorithm

The first improvement transforms the fuzzy relation membership matrix into the $\alpha$-compatibility classes and models the class hierarchy as taxonomy. Hence, the modified problem can be thought as to mine generalized association rules in [10]. Given a set of items $S$, a database of transactions $D$ produced by $S$, and a fuzzy proximity relation $R(S, S)$, the native algorithm finds the hierarchy with all membership degrees first. Then the Cumulate algorithm proposed in [10] is used to find all association rules for all possible $\alpha$. However, some users may not be interested in all possible membership degrees. It is waste of time to compute the whole relations in the hierarchy. Moreover, once users are concerned with other
relations, they must rebuild a new taxonomy and run the algorithm on the same transactions again for obtaining new association rules. The native algorithm is described as follows:

**INPUT:** $S$, $R(S, S)$, $D$, minimum support $s$ and minimum confidence $c$  
**OUTPUT:** All association rules on all possible of users, many redundant works will be performed during applications. As mentioned above, to fit the requirements knowledge for users in different levels and viewpoints of from other classes or primitive item sets. For example, in supports of some compatibility classes can be derived Observation: from the following observation.

The reason is that $T_4$ is union of items $D$, $E$ and $F$. Similarly, items $D$, $E$ or $F$ satisfies the minimum support. The and $G$. If there are association rules including $T_4$ or $T_6$, the primitive and $G$. If there are association rules including $T_4$ or $T_6$, the rules are also true for $T_7$. The computation of rules with $T_7$ is omitted.

The above observation motivates our algorithm. The $S$, $D$, and $R(S, S)$ are defined as in Section 2. Let $T = \{T_1$, $T_2$, $…$, $T_k\}$ be $k$ compatibility classes of all possible membership degree $\alpha$ for $R(S, S)$, $T_i \in 2^S$, $1 \leq i \leq k$. Assume that a large item set containing only primitive $S$ is called primitive large item set $L^p$, and class large item set $L^L$ is a large item set with items in $T$. We know that the class large item can be derived from primitive large item set or the other class items sets, but all primitive large item sets have to be computed from $D$. Since we can generate association rules from the corresponding large item sets directly, it is the critical problem to find the large item sets.

In our algorithm, the primitive large item sets are found by existing algorithm first. Generally, the mining procedure produces a set of candidate items $C^p$, and find the items that satisfies minimum support to form $L^p$. Let $C^p_i$ be the set of candidate with $i$ items, and $L^p_i$ be the primitive $i$ large item set. After finding the valid primitive large item sets, we produce the class large item set $L^L_i$ from $C^p_i$ and $L^p_i$. The first step of process is to determine the set of compatibility classes $T$ for given $\alpha$. Then, the candidate class item set $C^c$ is generated. Assume that $T_i$ is the union of primitive items $X_i$, $1 \leq j \leq t_i$, the number of transactions satisfying $X_i$ is denoted as $|X_i|$. An item set $T_iY$ will be in $C^c$ if and only if the summation of $|X_iY|$ is not less than the minimum support $s$, $1 \leq j \leq t_i$. From $C^c_i$, we calculate $L^L_i$ which should reach one of the following two conditions:

1. One of $|X_iY| \geq s$, $1 \leq j \leq t_i$;
2. $|X_i\cup Y| \geq s$.

We have $|X_i\cup Y| = |X_i| + |Y| - |X_iY|$. If the number of $|X_i|$, $|Y|$ and $|X_iY|$ are found in $C^p$, $|X_i\cup Y|$ will be determined. If not, $X_iY$ should be added into $C^p_i$ and count $|X_iY|$ for obtaining $|X_i\cup Y|$. After complete $L^L_i$, we derive the next candidate item set $C^c_i$ from $L^L_i$, $L^p_i$ and the values in $C^p_i$. This procedure will be repeated until $C^c_i = \{\}$. $L^L_i$ are the new large item sets.

When users request a specified $\alpha$, it can be solved by above process. While $\alpha$ being in a range, e.g. $\alpha \geq 0.8$, we must find the low level classes first before computing the high level classes. The algorithm is shown in the following.

**4: Algorithm for Rules Clustering**

A database provides many information and knowledge for users in different levels and viewpoints of applications. As mentioned above, to fit the requirements of users, many redundant works will be performed during the processing. We can avoid the redundant computations from the following observation.

**Observation:** For a proximity-based fuzzy relation, the supports of some compatibility classes can be derived from other classes or primitive item sets. For example, in Fig. 2, $T_4$ will satisfy the minimum support if one of the items $D$, $E$ or $F$ satisfies the minimum support. The reason is that $T_4$ is union of items $D$, $E$ and $F$. Similarly, $T_7$ can be derived from $T_4$ and $T_6$ or from items $D$, $E$, $F$ and $G$. If there are association rules including $T_4$ or $T_6$, the rules are also true for $T_7$. The computation of rules with $T_7$ is omitted.

The above observation motivates our algorithm. The $S$, $D$, and $R(S, S)$ are defined as in Section 2. Let $T = \{T_1$, $T_2$, $…$, $T_k\}$ be $k$ compatibility classes of all possible membership degree $\alpha$ for $R(S, S)$, $T_i \in 2^S$, $1 \leq i \leq k$. Assume that a large item set containing only primitive $S$ is called primitive large item set $L^p_i$, and class large item set $L^L_i$ is a large item set with items in $T$. We know that the class large item can be derived from primitive large item set or the other class items sets, but all primitive large item sets have to be computed from $D$. Since we can generate association rules from the corresponding large item sets directly, it is the critical problem to find the large item sets.

In our algorithm, the primitive large item sets are found by existing algorithm first. Generally, the mining procedure produces a set of candidate items $C^p_i$, and find the items that satisfies minimum support to form $L^p_i$. Let $C^p_i$ be the set of candidate with $i$ items, and $L^p_i$ be the primitive $i$ large item set. After finding the valid primitive large item sets, we produce the class large item set $L^L_i$ from $C^p_i$ and $L^p_i$. The first step of process is to determine the set of compatibility classes $T$ for given $\alpha$. Then, the candidate class item set $C^c$ is generated. Assume that $T_i$ is the union of primitive items $X_i$, $1 \leq j \leq t_i$, the number of transactions satisfying $X_i$ is denoted as $|X_i|$. An item set $T_iY$ will be in $C^c$ if and only if the summation of $|X_iY|$ is not less than the minimum support $s$, $1 \leq j \leq t_i$. From $C^c_i$, we calculate $L^L_i$ which should reach one of the following two conditions:

1. One of $|X_iY| \geq s$, $1 \leq j \leq t_i$;
2. $|X_i\cup Y| \geq s$.

We have $|X_i\cup Y| = |X_i| + |Y| - |X_iY|$. If the number of $|X_i|$, $|Y|$ and $|X_iY|$ are found in $C^p$, $|X_i\cup Y|$ will be determined. If not, $X_iY$ should be added into $C^p_i$ and count $|X_iY|$ for obtaining $|X_i\cup Y|$. After complete $L^L_i$, we derive the next candidate item set $C^c_i$ from $L^L_i$, $L^p_i$ and the values in $C^p_i$. This procedure will be repeated until $C^c_i = \{\}$. $L^L_i$ are the new large item sets.

When users request a specified $\alpha$, it can be solved by above process. While $\alpha$ being in a range, e.g. $\alpha \geq 0.8$, we must find the low level classes first before computing the high level classes. The algorithm is shown in the following.

**INPUT:** $S$, $R(S, S)$, $D$, minimum support $s$ and minimum confidence $c$, a specified membership degree $\alpha$  
**OUTPUT:** The large item sets with $\alpha$  
**STEP 1:** Find all possible $\alpha$ and compute the class with $R(S, S) \geq \alpha$ to model the taxonomy $T$.  
**STEP 2:** Call algorithm Cumulate($T$, $S$, $D$, $s$, $c$) to find $I$, the set of all association rules on the taxonomy.  
**STEP 3:** For all possible $\alpha$, classifying the association rules $R$ to $I_{\alpha}$ by different membership degrees.

<table>
<thead>
<tr>
<th>Transactions</th>
<th>set of Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A B</td>
</tr>
<tr>
<td>2</td>
<td>A C D E</td>
</tr>
<tr>
<td>3</td>
<td>C E</td>
</tr>
<tr>
<td>4</td>
<td>B C E</td>
</tr>
<tr>
<td>5</td>
<td>A C E</td>
</tr>
<tr>
<td>6</td>
<td>B E</td>
</tr>
<tr>
<td>7</td>
<td>C D</td>
</tr>
<tr>
<td>8</td>
<td>C D E</td>
</tr>
</tbody>
</table>
Table 1 A database $D$

Example: Consider only the items A, B, C, D, E of the fuzzy relation in Fig. 2. A database of transactions $D$ is given in Table 1. The minimum support is 3, and $\alpha \geq 0.8$. That is, $S = \{A, B, C, D, E\}$, $T = \{T_1, T_2, T_3\}$. Because the level of $T_3$ is higher than $T_1$ and $T_2$, we find $T_1$ and $T_2$ firstly. The following example is the details of our algorithm.

$C^p_1 = \{A, B, C, D, E\}$;
$C^r_1 = \{T_1, T_2\}$;
$|A| = 3, |B| = 3, |C| = 6, |D| = 3, |E| = 6$;
$L^p_1 = \{A, B, C, D, E\}$.

$C^p_2 = \{AB, AC, AD, AE, BC, BD, BE, CD, CE, DE\}$;
$|AB| = 1, |AC| = 2, |AD| = 1, |AE| = 2, |BC| = 1,$
$|BD| = 0, |BE| = 2, |CD| = 3, |CE| = 5, |DE| = 2$;
$L^p_2 = \{CD, CE\}$.

$|T_1| = |C| + |D| - |CD| = 6 + 3 - 3 = 6$
$|T_2| = |A| + |B| - |AB| = 3 + 3 - 1 = 5$
$L^c_1 = \{T_1, T_2\}$.

$C^p_3 = \{\}$;
The primitive large item sets are complete here.

Generating candidate class item set by the following calculating:

$|T_1A| \leq |AC| + |AD| = 2 + 1 = 3$,
$|T_1B| \leq |BC| + |BD| = 1 + 0 = 1 < s$,
$|T_1E| \leq |AE| + |BE| = 2 + 2 = 4$;

$|T_1T_2| \leq |AC| + |AD| + |BC| + |BD| - |ACD| - |BCD| + |ABCD| = 3$;

$C^c_2 = \{T_1A, T_1E, T_2C, T_2E, T_1T_2\}$.

The new candidate primitive item set is

$C^p_3 = \{ABC, ACD, BCD, CDE, ABE\}$ and
$C^p_4 = \{ABCD\}$.

$|ABC| = 0, |ACD| = 1, |BCD| = 0, |CDE| = 2, |ABE| = 0$,
$|ABCD| = 0$;

$|T_1A| = |AC| + |AD| - |ACD| = 2$,
$|T_1E| = |AE| + |BE| - |ABE| = 4$;

$|T_1T_2| = |AC| + |AD| + |BC| + |BD| + |ACD| + |ABCD| + |ABE| = 3$;

$C^c_3 = \{T_1T_2E, T_2CE\}$.

$C^p_5 = \{T_1T_2E, T_2CE\}$.

$C^p_4 = \{ABCD, BCDE\}$.

$C^p_5 = \{ABCDE\}$.

$|T_1T_2E| = |ACE| + |ADE| + |BCE| + |BDE| - |ACDE| - |BCDE| + |ABCDE| = 3$,
$|T_2CE| = |ACE| + |BCE| - |ABCE| = 3$;

$L^c_3 = \{T_1T_2E, T_2CE\}$.

$C^p_4 = \{\}$.

Fig. 3 shows the data cube of the algorithm's processing.

The higher level $T_3$ is added to $T$ after finishing the $T_1$ and $T_2$, and execute the procedure again. However, most of the computing works for the number of item sets are reduced, since the item sets are redundant. For another membership degree or even a new fuzzy relation, the information in $C^p_i$ and $C^r_i$ are useful for the computation of these requirements.

5: Conclusions

The association rules discovered from a large amount of transactions can support many applications, such as business analysis and decision making. The requirements are distinct for people in various positions. For the same set of transactions in a database, it is one of important considerations to provide valuable information for different users efficiently. This paper proposes an algorithm to find the association rules from a set of items and a given fuzzy proximity relation by clustering the association rules of primitive items. Our method is more efficient than the native algorithm of finding generalized association rules repeatedly. The proposed method can be also extended to other fuzzy relations easily.

References


[8] R. J. Miller and Y. Yang, Association Rules over Interval Data, in *Proc. of the ACM SIGMOD, Int'l Conf. on*
Fig. 2 The data cube in our Example