The Weekly Mining of Fuzzy Patterns from Temporal Datasets

Md Husamuddin
College of Computer Science and IT
AlBaha University
AlBaha, KSA

ABSTRACT
The process of extracting fuzzy patterns from temporal datasets is a well known data mining problem. Weekly pattern is one such example where it reflects a pattern with some fuzzy time interval every week. This process involves two steps. Firstly, it finds frequent sets and secondly, it finds the association rules that occur in certain time intervals weekly. Most of the fuzzy patterns are concentrated as user defined. However, the probability of user not having prior knowledge of datasets being used in some applications is more. Thus, resulting in the loss of fuzziness related to the problem. The limitation of the natural language also bounds the user in specifying the same. This paper, proposes a method of extracting patterns that occur weekly in a particular fuzzy time frame and the fuzzy time frame is generated by the method itself. The efficacy of the method is backed by the experimental results.

Keywords
Temporal Patterns, Temporal Association rules, Superimposed intervals, Fuzzy set, Right reference functions, left reference functions, Membership functions.

1. INTRODUCTION
The analysis of transactional data is given significant importance among the various types of data mining applications. The most common example of such transactional data is ‘market basket’ data. In a market basket data set, each transaction is a collection of items bought by a customer at one time. The notion proposed in [1] is to capture the co-occurrence of items in transactions, given minimum support and minimum confidence threshold. Adding the time attribute to the above problem is one of the extensions. Whenever a transaction occurs, the time of transaction is automatically recorded. Ale et al [2] has proposed a method of extracting association rules that occur within a life span of a corresponding itemset.

The concept of locally frequent itemset has been proposed by Mahanta et al [3], where patterns occur frequently in a particular time frame and may or may not occur frequently throughout the life-span of the itemset. In [3] an algorithm has been proposed to extract such itemsets along with a list of sequences of time frames. Here each frequent itemset is linked with a sequence of time frames where it is frequent. In [4], the time stamp is considered as calendar dates and a method is discussed which can extract yearly, monthly and daily periodic or partially periodic patterns. If these periodic patterns are in a compact manner using the method discussed. In [5], a superimposition method is used for overlapping the frequency of intervals. Considering the time-stamp as year_month_day_hour_minute_second, methods were proposed in [6, 7, 8, 9], for extracting yearly, monthly, and daily fuzzy frequent itemsets. This paper discusses weekly fuzzy patterns and devised an algorithm for extracting such patterns. The algorithm discussed in this similar to that in [6, 7, 8, 9]. The paper is organized as follows. In section-II related works are discussed. Section-III discusses terms, definitions and notations used in the algorithm. In section-IV, the algorithm is discussed. Section-V discusses about results and analysis. Finally a summary and lines for future works are discussed in section-VI.

2. RELATED WORKS
Agrawal et al [1] first formulated the problem of discovering of association rules. Given a set ‘S’ of items and a large collection ‘C’ of transactions containing the items, the query is to find the association among the presence of various items in the transactions. An important extension of conventional data mining is Temporal Data Mining [10]. More interesting patterns can be extracted, if time aspect is taken into account. The association rule discovering process is also extended to incorporate temporal aspects. The query associated is to extract valid time frames during which association rules occur and the discovery of possible periodicities that association rules consists of. In [2], the authors proposed an algorithm for the discovery of temporal association rules. For each item (or itemset), a life-span is defined which is the time gap between the first occurrence and the last occurrence of the item in the transaction in the dataset. Thus each rule is associated with a time frame. In [3], the works done in [2] has been extended by considering time gap between two consecutive transactions containing an item set into account.

Ozden [11] proposed a method, considering the periodic nature of patterns which is able to extract patterns having periodic nature where the time period is specified by the user. In [12], Li et al discussed a method to find temporal association rules corresponding to fuzzy match i.e. association rule holding during “enough” number of intervals given by the corresponding calendar pattern. Similar works were done in [13] with multiple granularities of time intervals (e.g. first working day of every month) where both cyclic and user defined calendar patterns can be achieved.

Many researchers have studied about extracting fuzzy patterns from datasets. In [14], the authors proposed a method for extracting fuzzy temporal patterns from a given process instance. Similar work is done in [15]. In [16], a method of mining fuzzy periodic association rules is discussed. In [6], authors have discussed a method for finding yearly fuzzy patterns. In [7], methods for mining monthly fuzzy patterns are discussed. In [8], methods are discussed for finding daily fuzzy patterns. In [9], similar methods are discussed for extracting hourly fuzzy patterns.
3. TERMS, DEFINITIONS AND NOTATIONS

Let us review some definitions and notations used in this paper.

Let $U$ be the universe of discourse. A fuzzy set $X$ in $U$ is characterized by a membership function $X(a)$ lying in $[0, 1]$. Thus a fuzzy set $X$ is defined as

$$ X= \{(a, X(a)), a \in U\} $$

A fuzzy set $X$ is said to be normal if $X(1)=1$ for at least one $a \in U$. An $a$-cut of a fuzzy set is an ordinary set of elements with membership grade greater than or equal to a threshold $a$, $0 \leq a \leq 1$. Thus an $a$-cut $X_a$ of a fuzzy set $X$ is characterized by $X_a=\{a \in U; X(a) \geq a\}$ [see e.g. [17]].

A fuzzy set is said to be convex if all its $a$-cuts are convex sets.

A fuzzy number is a convex normalized fuzzy set $X$ defined on the real line $R$ such that

1. there exists an $a_0 \in R$ such that $X(a_0)=1$, and
2. $X(1)$ is piecewise continuous.

Thus a fuzzy number can be thought of as containing the real numbers within some interval to varying degrees.

Fuzzy intervals are special fuzzy numbers satisfying the following:

1. there exists an interval $[x, y] \subset R$ such that $X(a)=1$ for all $a \in [x, y]$, and
2. $X(1)$ is piecewise continuous.

A fuzzy interval can be thought of as a fuzzy number with a flat region. A fuzzy interval is denoted by $X=[x, y, z, w]$ with $x < y < z < w$ where $X(x)=X(y)=0$ and $X(a)=1$ for all $a \in [y, z]$. $X(a)$ for all $a \in [x, y]$ is known as a left reference function and $X(1)$ for all $a \in [z, w]$ is known as the right reference function. The left reference function is non-decreasing and the right reference function is non-increasing.

The support of a fuzzy set $X$ within a universal set $U$ is the crisp set that contains all the elements of $U$ that have non-zero membership grades in $X$ and is denoted by $S(X)$. Thus

$$ S(X)=\{a \in U; X(a)>0\} $$

The core of a fuzzy set $X$ within a universal set $U$ is the crisp set that contains all the elements of $U$ having membership grades 1 in $X$.

3.1 Set Superimposition

In [18] an operation called superimposition denoted by $(S)$ was proposed. If $X$ is superimposed over $Y$ or $Y$ is superimposed over $X$, then

$$ (X \cup Y) = (X \cup Y)^{(2)} = \left(\begin{array}{c} X \cup Y \\ Y \cup X \end{array}\right) $$

Where $(X \cup Y)^{(2)}$ are the elements of $(X \cup Y)$ represented twice, and $(\cup)$ represents union of disjoint sets.

To explain this, an example has been taken.

If $X=\{x_1, y_1\}$ and $Y=\{x_2, y_2\}$ are two real intervals such that $X \cap Y \neq \phi$, it would get a superimposed portion. It can be seen from (1)

$$ x_1 \leq \min(x_1, x_2), \quad x_2 \geq \max(x_1, x_2) $$

$$ y_1 \leq \min(y_1, y_2), \quad y_2 \geq \max(y_1, y_2) $$

(2) Explains why if two line segments are superimposed, the common portion looks doubly dark [5]. The identity (2) is called fundamental identity of superimposition of intervals.

Let now $[x_1, y_1]^{(1/2)}$ and $[x_2, y_2]^{(1/2)}$ be two fuzzy sets with constant membership value $1/2$ everywhere (i.e. equi-fuzzy intervals with membership value $1/2$). If $[x_1, y_1] \cap [x_2, y_2] \neq \phi$ then applying (2) on the two equi-fuzzy intervals it can be written as

$$ [x_1,y_1]^{(1/2)} \cap [x_2,y_2]^{(1/2)} \Leftarrow [x_1,y_1]^{(1/2)} + [x_2,y_2]^{(1/2)} $$

$$ \cdots (3) $$

Let $[a_i, b_i], i=1,2,\ldots,n$, be $n$ real intervals such that

$$ \bigcap_{i=1}^{n} [a_i, b_i] \neq \phi $$

Generalizing (3) gives

$$ [a_1,b_1]^{(1/n)} \cap [a_2,b_2]^{(1/n)} \cap \cdots \cap [a_n,b_n]^{(1/n)} \subseteq [a_1,a_2]^{(1/n)} \cup [a_2,a_3]^{(1/n)} \cup \cdots \cup [a_n,a_1]^{(1/n)} $$

$$ \cdots (4) $$

In (4), the sequence $[a_0]_i$ is formed by sorting the sequence $[a_i]$ in ascending order of magnitude for $i=1,2,\ldots,n$ and similarly $[b_0]_i$ is formed by sorting the sequence $[b_i]$ in ascending order.

Although the set superimposition is operated on the closed intervals, it can be extended to operate on the open and the half-open intervals in the trivial way.

3.1 Lemma 1. (The Glivenko-Cantelli Lemma Of Order Statistics)

Let $A=(A_1, A_2, \ldots, A_n)$ and $B=(B_1, B_2, \ldots, B_n)$ be two random vectors, and $(a_1, a_2, \ldots, a_n)$ and $(b_1, b_2, \ldots, b_n)$ be two particular realizations of $A$ and $B$ respectively. Assume that the sub-fields induced by $A_k$, $k=1, 2, \ldots, n$ are identical and independent. Similarly assume that the sub-fields induced by $B_k$, $k=1, 2, \ldots, n$ are also identical and independent. Let $a_{(1)}, a_{(2)}, \ldots, a_{(n)}$ be the values of $a_1, a_2, \ldots, a_n$ and $b_{(1)}, b_{(2)}, \ldots, b_{(n)}$ be the values of $b_1, b_2, \ldots, b_n$ arranged in ascending order.

For $A$ and $B$, if the empirical probability distribution functions $\phi_i(a)$ and $\phi_i(b)$ are defined as in (5) and (6) respectively. Then, the Glivenko-Cantelli Lemma of order statistics states that the mathematical expectation of the empirical probability distributions would be given by the respective theoretical probability distributions.
intervals is represented in a compact manner discussed in section-III.

For representing each superimposed interval of the form

\[ [i^{(1)}, i^{(2)}], [i^{(2)}, i^{(3)}], [i^{(3)}, i^{(4)}], \ldots \]

and so on.

Let's take two arrays of real numbers, one for storing the values \( i^{(1)}, i^{(2)}, i^{(3)}, \ldots, i^{(n)} \) and the other for storing the values \( i^{(1)}, i^{(2)}, \ldots, i^{(n)} \). Each of which is a sorted array. Now if a new interval \([\tilde{i}, \tilde{i}']\) is to be superimposed on this interval it adds \( t \) to the first array by finding its position (using binary search) in the first array so that it remains sorted. Similarly \( \tilde{i} \) is added to the second array.

Data structure used for representing a superimposed interval is

```plaintext
struct superinterval {
    int arysize, count;
    short *p, *q;
}
```

Here \( \text{arsize} \) represents the maximum size of the array used, \( \text{count} \) represents the number of intervals superimposed, and \( p \) and \( q \) are two pointer pointing to the two associated arrays.

**Algorithm 4.1**

for each locally frequent item set \( s \) do

1. \( T \leftarrow \) sequence of time intervals associated with \( s \)
2. \( Ts \leftarrow \) set of superimposed intervals initially set to null
3. \( pi = T \).get();
   // 'pi' is now pointing to the first interval in \( T \)
4. \( Ts.append(pi); \)
5. Do while ((pi = T.get()) != null)
6.   \( \text{flag} = 0; \)
7.   Do while ((psi = Ts.get()) != null)
8.      if (compsuperimp(pi, psi))
9.         \( \text{flag} = 1; \)
10.    if (\( \text{flag} == 0 \)) Ts.append(pi);
11. }
12. }
13. if \( \text{flag} == 0 \)
14.   { superimp(pi, psi); return 1; }
15. return 0;

The function \( \text{compsuperimp}(pi, psi) \) first computes the intersection of \( pi \) with the core of \( psi \). If the intersection non-
empty it superimposes \( pi \) by calling the function \( \text{superimp}(pi, psi) \) which actually carries on the superimposition process by updating the two lists associated as described earlier. The function returns 1 if \( pi \) has been superimposed on the \( psi \) otherwise returns 0. \( \text{get} \) and \( \text{append} \) are functions operating on lists to get a pointer to the next element in a list and to append an element into a list.

5. RESULTS OBTAINED

For experimental purpose, a synthetic dataset T10I4D100K is used, available from FIMI\(^1\) website. The number of items here are 942 with the transactions involved are 100000. The minimum number of items in a transaction is 4. The maximum number of items in a transaction is 77. The average number of items in a transaction is 39. The results obtained are presented in table 1 and figure 1.

Table 1. Weekly fuzzy frequent itemsets for different set of transactions for itemset \{5\}

<table>
<thead>
<tr>
<th>Data Size (No of Transactions)</th>
<th>10000</th>
<th>20000</th>
<th>30000</th>
<th>40000</th>
<th>50000</th>
<th>60000</th>
<th>70000</th>
<th>80000</th>
<th>90000</th>
<th>100000</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. fuzzy time intervals</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Since dataset is non-temporal, it incorporates temporal features into it. Here it keeps the life-span of the datasets as 1 year. Firstly, it takes only 10,000 transactions and found that the itemset \{5\} has a superimposed intervals superimposed on one place and hence it has one fuzzy time interval where it is frequent. For 20,000 and 30,000 transactions the same itemset has two superimposed intervals and so two fuzzy intervals, finally from 60,000-100,000 transactions, it gets \{5\} is frequent in four fuzzy time intervals.

6. CONCLUSION AND LINES FOR FUTURE WORK

This paper discusses a method for extracting weekly fuzzy patterns. The input is a list of time intervals associated with a frequent itemset generated using a method discussed [4]. Here it does not consider the year, month, week in the time hierarchy and only consider day, minute, second. The sequence of time intervals of the form \[\text{day_minute_second, day_minute_second}\] will be associated with each frequent itemset where it is frequent. Each interval in the sequence is visited by the algorithm one by one and stores the intervals in the superimposed form. In this way each frequent itemset is associated with one or more superimposed time intervals. Each superimposed interval delivers fuzzy time intervals. So, each frequent itemset is associated with one or more fuzzy time intervals. The best thing about the method is that the algorithm is not user-dependent i.e. fuzzy time intervals are extracted by algorithm automatically. Future work may be on the extraction of fuzzy patterns namely quarterly, half yearly patterns and bi-yearly.
7. REFERENCES


