A Survey-Paper on Filtering Datasets Using Frequent Patterns to Obtain Representative Pattern Sets

Mrs.Soumya.N.G, Mr.Rajashekar S A, Mr. Mahesha H.S

Abstract: Frequent pattern mining is an important problem in the data mining area. Frequent pattern mining has been a focused theme in data mining research for over a decade. Frequent pattern mining often produces an enormous number of frequent patterns, which imposes a great challenge on visualizing, understanding and further analysis of the generated patterns. Finding a small number of representative patterns to best approximate all other patterns becomes an important task. This paper gives a relative study on different techniques that can be used in order to find minimum number of representative pattern sets where closed sets play an important role.

Index Terms— Closed pattern set, Data mining, Delta free sets, Disjunction free sets, Free sets, Frequent Pattern mining.

1. Introduction:

Data mining is the process of discovering interesting and useful patterns and relationships in large volumes of data. Frequent pattern mining is an important problem in the data mining area. It was first introduced by Agrawal et al. in 1993. It has been observed that a lot of redundancy can appear in the complete set of frequent patterns , as any frequent patterns have similar items and supporting transactions. It is desirable to group similar patterns together and represent them using one single pattern.

2. Literature Survey:

A) Closed Sets

The application of this theory to frequent itemset mining has been proposed independently by Pasquier et al. and by Zaki and Ogihara . In this context, an itemset I is said to be *closed* in D if and only if no proper superset of I has the same support than I in D. The *closure* of an itemset I in D, denoted $cl(I)$, is the unique maximal superset of I having the same support than I and a closed itemset is equal to its closure. An alternative definition is to consider the equivalence classes of the itemsets appearing in the same sets of transactions, i.e., the equivalence classes of the relation has the same closure. Closed itemsets are the unique maximal elements of each equivalence class . For a given support threshold, it is thus sufficient to know the collection of all frequent closed itemsets (denoted *FreqClosed*) and their supports, to be able to generate all the frequent itemsets and their supports, i.e., F. For example, consider an itemset X, if X has no superset in *FreqClosed*, this means that $cl(X)$ is not frequent, and thus X cannot be frequent. If X has at least one superset in *FreqClosed*, then $supp(X) = supp(Y)$ where $Y = cl(X)$ is the smallest superset of X in *FreqClosed*. Let us consider the database containing the following transactions: two transactions *{a , b}*, two transactions *{a , b , c ,d}* two transactions *{a , b , c , d , e}* and one transaction *{a , b , c, d, e , f }* (see Table 1).

Items Trans. *a b c d e f*

	a	b	c	D	e	f
T1	1	1	0	0	0	0
T2	1	1	0	0	0	0
T3	1	1	1	1	0	0
T4	1	1	1	1	0	0
T5	1	1	1	1	1	0
T6	1	1	1	1	1	0
T7	1	1	1	1	1	1

Table 1. A toy database

In such a database, for example, the itemset *abc* is not closed, since it has the same support (i.e., 5 transactions) than *abcd*, one of its proper supersets. The itemset *abcd* is the maximal superset of *abc* having the same support, and thus is the closure of *abc*. If we choose a support threshold of 2 transactions, then the frequent closed sets are *ab*, *abcd*, *abcde* and their respective supports are 7, 5 and 3. Having only at hand these frequent closed sets, to generate the support of *abc* we consider the smallest frequent closed set that is a superset of *abc*. This frequent closed set is *abcd* and its support (i.e., 5 transactions) gives us the support of *abc*.

B) Free Sets

The *free sets* (also termed *δ-free sets*) have been introduced in [12, 13] and are based on the notion of *δ strong rule*3. Informally, a δ strong rule is an association rule of the form $X \Rightarrow a$, where $X \subseteq I$, $a \in I \setminus X$, and δ is natural number. This rule is valid in a database if $supp(X) - supp(X \cup \{a\}) \leq \delta$, i.e., the rule is violated in no more than δ transactions. Since δ is

- *Mrs.Soumya.N.G, Mtech Student in East west Institute of Technology, Dept of CSE, Bangalore, India*
- *Mr. Rajshekar S A, Assistant Professor in East West Institute of Technology, Dept of CSE, Bangalore, India*
- *Mr. Mahesha H.S, Mtech Student in East west Institute of Technology, Dept of CSE, Bangalore, India*

supposed to be small w.r.t. |D|, δ-strong rules have a high confidence (in particular confidence 1 when $\delta = 0$). An itemset $Y \subseteq I$ is a δ -free set if and only if there is no valid δ-strong rule $X \Rightarrow^{\delta} a$ such that $X \subseteq Y$, $a \in Y$ and where by definition $a \notin X$. The set of all frequent δ -free sets, denoted $FreqFree^{\delta}$,and their supports enables to approximate the support of the frequent non- δ free sets. Let us consider Y a frequent non- δ-free set. Then, there exists a valid δ-strong rule $X \Rightarrow^{\delta} a$ such that $X \subset Y$ and $a \in Y$. Moreover, $Y \setminus \{a\} \Rightarrow^{\delta} a$ is also valid. Thus the support of Y can be approximated by the support of the frequent set $Y \setminus \{a\}$. If $Y \setminus \{a\}$ is a free-set then we have its support, if not, it can be in turn approximated by the support of a smaller itemset. This recursive process gives an approximation of the support of Y. Using this principle, the best approximation is the lowest upper bound. Thus, in practice, the support of Y is approximated by the minimal support value of the frequent δ -free sets that are subsets of Y. The error made has been formalized using the framework of an ε-adequate representation, and is small on common real datasets.

When δ = 0, the support of all frequent non-δ-free sets can be determined exactly. In fact, the 0-free sets corresponds to the *key patterns* (also called *generators*) and used in other works.

The following property mentioned by several authors establishes a direct link between 0-free sets and closed sets: any frequent closed sets is the closure of at least one frequent 0-free sets. As a result, when considering each (frequent) 0-free set X, $cl(X)$ is a (frequent) closed set but also $X \Rightarrow cl(X) \setminus X$ is an association rule

with confidence 1. In fact, 0-free sets are the minimal elements of the already mentioned equivalence classes. Since several minimal elements are possible, collections of 0-free sets are generally larger than collections of closed sets. In the toy example from Table 1, the 2-frequent 0-free sets are ∅, *c*, *d* and *e*. Even though the frequent δ-free sets are sufficient to approximate the support of all frequent non-δ-free sets (or to determine this support exactly when δ = 0), they are not sufficient to decide whether an itemset is frequent or not. For this purpose, the collection of frequent δ -free sets is completed by the collection of minimal infrequent δ -free itemsets, Now, given any itemset Y , if there exists $Z \subseteq Y$, such that Z is a minimal infrequent δ -free itemsets, then we know that Y is not frequent. In the other case, the support of Y can be approximated as described above. There are many algorithm proposed on closed itemsets namely.

i) A-CLOSE**:** The A-Close(Apriori based closed frequent itemset) algorithm is works based on the apriori algorithm along with the concept of the closed itemset lattices concept.

ii) CHARM: stands for Closed Association Rule Mining. The algorithm is used to mine the closed frequent patterns. It explores patternset and didset(Document idset) space simultaneously which skips many levels quickly to identify the closed frequent patterns.

iii) CLOSET: algorithm used to mine the closed frequent patterns with the help of three techniques such as compression frequent pattern tree structure without candidate generation, Single Path compression technique and partition based projection mechanism.

iv) CLOSET+: algorithm is used to mine closed frequent pattern. Initially, it scans the database only once to find the global frequent patterns and sort the database in support descending order and forms the frequent pattern list, scans the document and builds the FP-Tree using the pattern list, using divide and conquer technique and depth first searching paradigm it finds the closed frequent patterns. Finally, stop the process until all the patterns in the global header are mined. The frequent closed patterns are obtained either from result tree or from the output file.

v) CARPENTER: stands for Closed Pattern Discovery. Transposing Tables that are Extremely Long used to mine long biological dataset. It consists of two main steps: Transpose the data into table and row enumeration tree search. In the First step, it transpose the patterns into the table named as transpose table, in that each tuple lists the feature along with the row ides feature occurs in the original table. In the second step, according to the transpose table, construct the row enumeration tree which enumerates row ids with predefined order and search the tree in depth first order without any pruning strategies.

vi) TD-CLOSE: The algorithm uses the Top-Down strategy and closeness checking method to mine the frequent closed patterns .Initially, it performs the transposition operation to transform original table to the transposed table and initialize the frequent closed patterns as an empty set and size of the rowset as zero. Finally, the TopDownMine is used to find the frequent closed patterns.

vii) PGMiner: The PGMiner [17, 23] is the Prefix Graph Miner which mines the frequent closed patterns. This algorithm integrates two methods such as projected database and bit vectors. Initially it projecting the document containing into nodes of a graph as similar to FP-Tree but different in the cost of traversing

multiple branches of the tree to collect frequency information are low compared to the FP-Tree .Then, the projection of the nodes are encode into the bit vectors with the shorter length compare to the existing approaches. The efficiency of mining algorithm is improved by using two phase such as intra node itemset mining and inter node pruning mechanism. In the intra node itemset mining, it finds the frequent closed patterns for each node and form local closed patterns. In the inter node pruning mechanism, it checks whether the local closed patterns are also globally closed or not and finally obtain the frequent closed patterns.

viii) PTclose: The PTclose stands for the Patricia Tree used to mine the closed frequent patterns. In this algorithm, it uses the PTArray Technique to reduce the scanning of the patricia tree. It has two inputs such as Patricia tree and Closed Frequent Pattern tree. The Patricia tree is a compact tree, used to characterize all relevant frequency information within the document, each branch in patricia tree represents a frequent

patterns and the nodes along the branches are in frequency decreasing order from root through leaves. Initially it verifies whether the patricia tree is a single path tree, if so all candidate closed Frequent patterns (CFP) are obtained from the patricia tree and candidate patterns is then compared with all the CFPs within patricia tree. If it is closed patterns it will be inserted into patricia and all CFP – tree existing in memory will be updated until Frequent closed pattern are obtained.

C) Disjunction-Free Sets

Simple Disjunction-Free Sets

This representation has been proposed in [17, 18] as a generalization of 0-free sets. It is based on *disjunctive rules* of the form $X \Rightarrow a \vee b$, where $X \subseteq I$ and $a, b \in I \setminus X$. Such a rule is said to be valid if any transaction containing X contains also a or b (maybe both). Thus the support of X is equal to the sum of $supp(X \cup \{a\})$ and $supp(X \cup \{b\})$ minus $supp(X \cup \{a , b\})$ since the transactions containing $X \cup \{a , b\}$ have been counted both in $supp(X \cup \{a\})$ and $supp(X \cup \{b\})$. So, we have the relation $supp(X \cup \{a , b\}) = supp(X \cup \{a\}) + supp(X \cup \{b\}) - supp(X)$ and the satisfaction of this relation is equivalent to the validity of the rule $X \Rightarrow a \vee b$. Similar to δ -free sets, an itemset $Y \subseteq I$ is a *disjunction-free set* if and only if there is no valid disjunctive rule $X \Rightarrow a \vee b$, such that $X \subset Y$, $a, b \in Y$ and where by definition $a \notin X$ and $b \notin X$. In the following, the collection of all frequent disjunction-free sets is denoted *FreqDFree*. Knowing all elements in *FreqDFree* and their supports is not sufficient to determine the support of all frequent itemsets. For that purpose the representation can be completed in different ways. Intuitively, *FreqDFree* must be completed with the collection of all the valid rules of the form $X \Rightarrow a \vee b$, where $X \in FreqDFree$ and $X \cup \{a , b\}$ is frequent. This can be illustrated inductively as follows. Suppose that using *FreqDFree* (and the supports of its elements) and the collection of rules defined above, we are able to compute the support of any itemset having a size lesser or equal to k. Let us consider a frequent itemset Y such that $|Y| = k + 1$. If Y is disjunction-free then $Y \in FreqDFree$ and we know its support. If Y is not disjunction-free, then there exists a valid rule $X \Rightarrow a \vee b$ such that $X \subset Y$ and $a, b \in Y$. By definition of a valid rule, $Y \setminus \{ a , b \} \Rightarrow a \vee b$ is also valid. Hence the relation $supp(Y) = supp(Y \setminus \{b\}) + supp(Y \setminus \{a\}) - supp(Y \setminus \{a , b\})$ holds. Since Y is frequent, the itemsets $Y \setminus \{b\}$, $Y \setminus \{a\}$ and $Y \setminus \{a , b\}$ are also frequent. Moreover, these three sets have a size strictly lesser than $k + 1$. Thus, by hypothesis, we can determine their supports, and then compute $supp(Y)$.

3. Existing System

This existing system is based on closed pattern set which are unique maximal elements of equivalence class. The following algorithm gives a brief description of constructing a closed frequent pattern tree to mine representative patterns.

THE MINRPSET Algorithm

A straightforward algorithm for finding a minimum representative pattern set works as follows. First we mine all patterns in F^, and then we generate C(X)—the set of frequent patterns that X covers—for every pattern $X \in F^$. We get $|F^|$ sets. The elements of these sets are frequent patterns in F. Let $S = \{C(X) | X \in F^\}$. Finding a minimum representative pattern set is now equivalent to finding a minimum number of sets in S that can cover all the frequent patterns in F. This is a set cover problem, and it is NP-hard. We use the well-known greedy algorithm [5] to solve the problem, which achieves an approximation ratio of $\Sigma^k_{i=1} = 1/i$, where k is the maximal size of the sets in S. This algorithm is called MinRPset. The greedy algorithm is essentially the best-possible polynomial time approximation algorithm for the set cover problem. Our experiment results have shown that it usually takes little time to finish. Generating C(X)s is the main bottleneck of the MinRPset algorithm when F and F^ are large because we need to find C(X)s over a large F for a large number of patterns in F^. Let F be the set of frequent patterns in a dataset D with respect to threshold minsup, and F^ be the set of patterns with support no less than $minsup \cdot (1- \epsilon)$ in D. Obviously, $F \subseteq F^$. Given a pattern $X \in F^$, we use C(X) to denote the set of frequent patterns that can be ϵ -covered by X. We have $C(X) \subseteq F$. If X is frequent, we have $X \in C(X)$. We use the following techniques to improve the efficiency of MinRPset: 1) consider closed patterns only; 2) use a structure called CFPtree to find C(X)s efficiently and 3) use a light-weight compression technique to compress C(X)s.

Considering closed patterns only:

A pattern is closed if it is more frequent than all of its supersets. If a pattern X1 is non-closed, then there exists another pattern X2 such that $X1 \subset X2$ and $supp(X2) = supp(X1)$.

Lemma 1: Given two patterns X1 and X2 such that $X1 \subseteq X2$ and $supp(X1) = supp(X2)$, if X2 is ϵ -covered by a pattern X, then X1 must be ϵ -covered by X too. It implies that instead of covering all frequent patterns, we can cover frequent closed patterns only, which leads to the following lemma.

Lemma 2: Let F be the set of frequent patterns in a dataset D with respect to a threshold minsup. If a set of patterns R ϵ -covers all the frequent closed patterns in F, then R ϵ -covers all the frequent patterns in F.

Lemma 3: Given two patterns X1 and X2 such that $X1 \subseteq X2$ and $supp(X1) = supp(X2)$, if a pattern X is ϵ-covered by X1, then X must be ϵ -covered by X2 too.

It suggests that we can use closed patterns only to cover all frequent patterns. The number of frequent closed patterns can be orders of magnitude smaller than the total number of frequent patterns. Considering only closed patterns improve the efficiency of the MinRPset algorithm in two aspects. First, it reduces the size of individual C(X)s since now they contain only frequent closed patterns. Second, it reduces the number of patterns whose C(X) needs to be generated as now we need to generate C(X)s for closed patterns only. The CFP-tree structure allows different patterns to share the storage of their prefixes as well as suffixes. Prefix sharing is easy to understand. Only the items after an entry E can appear in the subtree pointed by E, and these items are called *candidate extensions* of E. Suffix

sharing occurs when a candidate extension of an entry E occurs in the same set of transactions as the pattern represented by E. Let i be a candidate extension of E and X be a pattern represented by E. If supp(X) = supp(X ∪ {i}), then for any pattern Z, we must have supp(X ∪ Z) =supp(X ∪ {i} ∪ Z). In other words, X and X ∪ {i} have the same extensions. In CFP-tree, a singleton node containing item i is created to enable the sharing between X and X ∪{i}. In the root node of Figure 1, item f is a candidate extension of item p. Every entry in a CFP-tree represents one or more patterns with the same support, and these patterns contain the items on the path from the root to the entry. Items contained in singleton nodes are optional. CFP-tree a very compact structure for storing frequent patterns. The number of entries in a CFP-tree is much smaller than the total number of patterns stored in the tree. The CFP-tree structure has the following property.

Property 1: In a multiple-entry node, the item of an entry E can appear in the subtrees pointed by entries before E, but it cannot appear in the subtrees pointed by entries after E. In a CFP-tree, the supersets of a pattern cannot appear on the right of the pattern. They appear either on the left of the pattern or in the subtree pointed by the pattern.

4. Conclusion

The notion of condensed representation has been identified as a core concept for inductive query optimization and its interest goes far beyond simple KDD processes based on itemsets. The existing algorithm first mines frequent Patterns and then find representative patterns in a post-processing step, Due to the use of the post-processing strategy, MinRPset has the following additional benefits besides producing fewer representative patterns:1) The post-processing strategy allows users to try different ϵ values without mining frequent patterns multiple times. This is especially beneficial on very large datasets. 2) In MinRPset it is easy to keep record of the set of patterns covered by each representative pattern. This information is useful for users to inspect individual representative patterns in more details.3) We can relax the conditions on ϵ-covered to further reduce the number of representative patterns

References:

1] A. Bykowski and C. Rigotti, "A condensed representation to find frequent patterns," in *PODS*, 2001.

2] J.-F. Boulicaut, A. Bykowski, and C. Rigotti, "Free-sets: A condensed representation of boolean data for the approximation of frequency queries," *Data Mining and Knowledge Discovery*,vol. 7, no. 1, pp. 5–22, 2003.

3]Bruno Cremilleux,Jean-Francois Boulicaut, "Simplest Rules characterizing Classes Generated by δ-Free Sets".

4] Guimei Liu, Haojun Zhang, and Limsoon Wong , "A Flexible Approach to Finding Representative Pattern Sets".

5] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233–235, 1979.

6] R. Jin, M. Abu-Ata, Y. Xiang, and N. Ruan, "Effective and efficient itemset pattern summarization: regression-based approaches," in *KDD*, 2008, pp. 399–407.

7] A. Giacometti, D. Laurent, and C. T. Diop. Condensed representations for sets of mining queries. In *Database Technologies for Data Mining - Discovering Knowledge with Inductive Queries*, volume 2682 of *LNCS*, pages 250{269. Springer-Verlag,2004.

8] El Far M. Moumoun L., Gadi, T., and Benslimane R., 2010, "An efficient CHARM algorithm for indexation 2D/3D and selection of characteristic views" , Proceeding of 5th International Symposium on I/V Communications and Mobile Network (ISVC), pp.1-4

9] Feng Pan, Gao Cong, Anthony K. H. Tung, Jiong Yang and Mohammed J. Zaki, 2003, "CARPENTER: Finding Closed Patterns in Long Biological Datasets, in Proceedings of the SIGKDD '03.