TRARM-RelSup: Targeted Rare Association Rule Mining Using Itemset Trees and the Relative Support Measure

Jennifer Lavergne, Ryan Benton, and Vijay V. Raghavan

University of Louisiana at Lafayette: The Center for Advanced Computer Studies, Lafayette LA 70503, USA
jjslavernge686@hotmail.com, rbenton@cacs.louisiana.edu, raghavan@louisiana.edu

Abstract. The goal of association mining is to find potentially interesting rules in large repositories of data. Unfortunately using a minimum support threshold, a standard practice to improve the association mining processing complexity, can allow some of these rules to remain hidden. This occurs because not all rules which have high confidence have a high support count. Various methods have been proposed to find these low support rules, but the resulting increase in complexity can be prohibitively expensive. In this paper, we propose a novel targeted association mining approach to rare rule mining using the itemset tree data structure (aka TRARM-RelSup). This algorithm combines the efficiency of targeted association mining querying with the capabilities of rare rule mining; this results in discovering a more focused, standard and rare rules for the user, while keeping the complexity manageable.

Keywords: data mining, association mining, targeted association mining, itemset tree, rare association rule mining.

1 Introduction

For years, organizations have sought to make sense of the large repositories of data collected from their day to day workings. Hidden within these datasets are potentially interesting correlations. An example is a department store that wishes to know which articles of clothing, jewelry, and/or shoes are selling well together. They utilize this information and advertise slower moving items with more desirable ones in magazines and stores. Prediction of new items that may sell well together can also be drawn from this information.

The process by which these gems of information are discovered is referred to as association mining [11,2,12]. Association mining processes large quantities of transactional data and returns rules based upon strong co-occurrences found. This process can be broken up into two separate subtasks: finding frequent co-occurrences and generating implications. An itemset is considered frequent if the frequency of its occurrence in a dataset is greater than some user-defined threshold. Examples of itemsets from a department store transactional dataset

L. Chen et al. (Eds.): ISMIS 2012, LNAI 7661, pp. 61–70, 2012.

would be I = {shoes, shirt, tie} and I = {necklace, earrings}. Unfortunately, using brute force association mining to find frequent co-occurences can be highly complex and require significant amount of processing time. Agrawal et al. [2] observed that if an itemset (a set of items from the dataset) is infrequent, then any superset containing the itemset will also be infrequent. This is known as the Apriori principle. Using the Apriori principle, we are able to prune large amounts of candidate itemsets from the mining process, reducing complexity and processing time [2].

The first association mining subtask scans the dataset for itemsets I that occur frequently together. Support is found using the formula support = count(I)/ number of transactions in the dataset [2]. The Apriori principle uses a user-defined threshold minimum support, or minsup, to determine if an itemset is considered frequent. Once the set of all frequent itemsets is generated, the second subtask uses this set to generate association rules. An association rule is an implication $X \Rightarrow Y$, where $I = X \cup Y$, X and Y are disjoint itemsets. In order to determine if an association rule is potentially interesting, Apriori uses the conditional probability, i.e. how confident are we that Y will occur given X. Confidence is found using the formula confidence = Support($X \cup Y$)/Support(X) [2]. Like minsup, we use a user-defined threshold minimum confidence, or minconf, to determine a rule's potential interestingness. If a rule's confidence $\geq$ minconf and it is derived from a frequent itemset, then the rule is considered strong and is returned to the user as output.

Association mining is designed to return all strong rules. This can result in prohibitively large rule sets returned to the user. A user may not be interested in most of the rules returned. For example, a department store owner may not be interested in correlations containing only their last season inventory, in favor of correlations concerning their newest products.

In 1999, Hafez et al. [6] proposed the concept of targeted association mining. Targeted association mining discovers rules based upon the interests of users and, therefore, can generate more focused results. The itemset tree data structure was developed to facilitate the targeted rule mining process. The use of the tree structure reduces the complexity of rule mining when compared to the Apriori style algorithms and generates more focused results [6,8]. Additional work has been completed showing that the itemset tree structure can also be used for efficiently finding all possible frequent association rules in a dataset [9].

Association mining and targeted association mining both focus on finding frequent itemsets based upon the minsup threshold. Unfortunately, not all high-confidence rules have a high support. By using the Apriori principle, these high confidence/low support rules are lost. This is referred to as the rare rule problem [1,13]. One can lower the minsup value in order to find more rare rules, but this greatly increases complexity and enlarges the rule set. Another study proposed the use of more than one minimum support value in order to separate itemsets into different frequency zones and then mined the rules therein. This method can potentially lose rules that may occur outside the frequency zones [3].

Thus, on one side, the rare rule mining permits the discovery of high confidence/low support rules, but at great computational cost. On the other, targeted mining currently allows for the efficient discovery of rules containing items of interest to the user, but it lacks the ability to find high-confidence/low-support rules. By merging the targeted mining with rare rule mining, the ability to mine targeted standard and rare rules, in a computationally efficient manner could be achieved. This is the premise of TRARM-RelSup algorithm that is being proposed in this paper. TRARM-RelSup combines the efficiency of the itemset tree for targeted association mining with the strategy of using relative support in order to efficiently find targeted rare association rules.

This paper is organized in the following manner. Section 2 discusses work related to the TRARM-RelSup algorithm. Section 3 will cover the proposed approach for combining targeted and rare association mining in the TRARM-RelSup algorithm. In Section 4, we discuss the experimental setup and the queries and comparisons used therein. Section 5 discusses experimental results. Finally, Section 6 presents our conclusions and planned future work.

2 Related Work

This section covers background information that will be used in the TRARM-RelSup algorithm. First the rule mining process implemented by the itemset tree algorithm and data structure will be covered [6,8]. Then, a discussion on the relative support measure used for finding rare rules within a dataset will be presented. Relative support finds all rules with a potentially high confidence value using a user-defined threshold, minRelSup, for pruning low confidence candidate itemsets. This measure will be adopted by the TRARM-RelSup algorithm for incorporating rare rule mining capabilities to the basic itemset tree querying method.

2.1 Targeted Association Mining

The itemset tree data structure was developed by Hafez et al. [6] and facilitates the targeted association mining process. Users specify query itemsets or areas of interest and the itemset tree generates all rules containing those specific items of interest to the user. All itemsets within the transaction set are mapped to integer values to expedite the mining process. The rule mining process has also been extended such that an itemset tree can be searched efficiently to find all rules in a dataset [9].

Definition 1: An *itemset tree*, T, consists of a root pair [I, f(I)], where I is an itemset and f(I) is I's frequency in the dataset, and a possibly empty set $\{T_1, T_2, \ldots, T_k\}$, each element of which is an itemset tree. If a query itemset Q is in the root itemset I, then it will also be in one or more children itemset trees of the root and if Q is not in the root, then there is a possibility that it may be in the children of the root if: first_item(I) $\leq$ first_item(Q) and last_item(I)

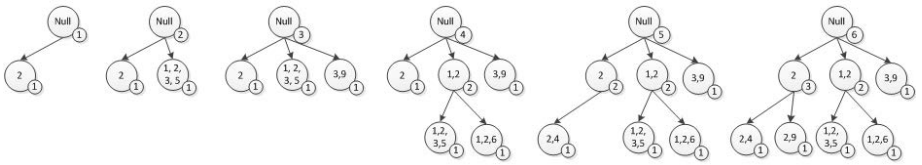

Fig. 1. Building an itemset tree

< last_item(Q) [6]. Note that in the latter case, notation I refers to the itemset associated with the root of one of the children itemset trees, recursively. The items in both sets I and Q are stored in increasing order of item IDs.

Example: The following steps detail the tree generation process seen in Fig. 1 and using the transactions: {2}, {1,2,3,5}, {3,9}, {1,2,6}, {2,4}, {2,9}.

- Step 1: Add transaction = {2} as the null root's first child.
- Step 2: {1,2,3,5} has no lexical overlap with {2}, so it becomes the root's second child.
- Step 3: {3,9} has no lexical overlap with {2}, so it becomes the root's third child.
- Step 4: {1,2,3,5} and {1,2,6} have a common lexical overlap of {1,2}. A node is created for the overlap with {1,2,3,5} and {1,2,6} as its children.
- Step 5: {2,4} becomes the first child of {2}.
- Step 6: {2,9} becomes the second child of {2}.

There are two ways to query an itemset tree. The first query type finds the support of a given query itemset Q. This algorithm takes advantage of the knowledge that if a query itemset Q is in the root itemset I, then it will also be in the children itemset trees of the root. Also if Q is not in the root, then there is a possibility that it may be in the children of the root if: first_item(I) $\leq$ first_item(Q) and last_item(I) < last_item(Q). Using this logic, one can traverse the tree and find all nodes that contribute support to Q and skip all subtrees that fail both conditions [6,8].

The second query type finds all itemsets in the tree that contain an itemset Q and is used as a preprocessing step before generating association rules. This algorithm is similar to query type 1. However, when a node whose itemset contains Q is found, not only is that itemset returned, but also all the itemsets contained in the subtree below. Only those itemsets whose support > minsup will be added to the results set [6,8].

In order to generate rules using this algorithm, the first step is to combine all itemsets into L = {l_1, ... ,l_n} where no l_i is a proper subset of and l_j, for i $\neq$ j, and n = $|L|$. The condition that l_i should not be a subset of l_j is applied only in the case that the support of l_i is equal to the support of l_j. Second, for each l_j the confidence is for the rules with LHS equal to Q and RHS equal to all subsets of l_j - Q is confidence = support((subset of (l_j - Q)) $\cup$ Q) / Support(Q). If confidence $\geq$ minconf, the rule is considered potentially interesting.

2.2 Rare Rule Mining

Recent work in rare rule mining has explored the ever present issue of trading accuracy for speed and vice versa. Reducing minsup to include all possible high confidence rules causes an undesirable increase in complexity and results in many duplicate and useless rules [1]. The MSApriori algorithm uses multiple minsups for creating frequent zones reduces complexity when compared to the brute force rule mining. Bu,t overall, it still generates large result sets. In addition this method also loses the itemsets outside of the frequency zones [3]. In 2010, Gedikli and Jannach [5] proposed adapting MSApriori [3] for creating weighted association rules for recommenders. This new recommendation system generates proposed personalized rule sets based upon the rules sets of the user's surrounding neighbors. This method works best when mined upon "very sparse" datasets [5]. The concept of a support difference (SD) for assigning a minimum supports to an itemset was proposed by Kiren et al [7]. This difference considers the maximum support deviation allowable when comparing the itemset's support to that of its corresponding supersets. If a superset's support is within this bound, it can be considered "frequent". This method also dynamically applies an additional minsup for each itemset in the transaction set based upon their current support [7].

A confidence-like measure, relative support, attempts to narrow the search field by reducing the number of candidate itemsets. This measure uses knowledge of rule confidence to prune itemsets that do not have the potential for a high confidence value. A method for finding rare and frequent rules with high support proposes the use of both minsup and the user-defined minRelSup threshold for finding candidate itemsets. The relative support of an itemset I can be found using the formula RelSup = supp(I)/supp(X), where X is a subset of I with the highest support. This method finds all candidate itemsets with the potential for high confidence rules. All rules generated from this method will have confidence $\geq$ minRelSup. This reduces complexity for finding candidate itemsets, but increases complexity for the rule generation process due to the increased number of candidate itemsets [14]. Also the resulting rules sets are still the same as an Apriori-like algorithm and contain all the same redundant and uninteresting rules as before.

3 Combining Itemset Trees and Rare Rule Mining: TRARM-RelSup Algorithm

The previous sections have covered three different types of association mining: frequent, targeted, and rare. Frequent association mining has complexity issues and fails to find all high confidence rules [1,13,10]. Also, not all rules generated are of interest to the user, and the result set it large and unfocused. Targeted association mining reduces some of the complexity of the mining process, but still fails to find all high confidence rules. Rare rule mining is said to have higher complexity than frequent association mining due to the addition of frequent and rare candidate itemsets. This also results in large and unfocused rule sets.

This paper proposes a rare rule mining algorithm, TRARM-RelSup, which combines the efficiency of targeted association mining with the capabilities of rare rule mining. By combining these concepts, we can allow users the ability to choose the areas they want to search in for a more focused results set and we can not only find strong rules, but also rare rules that contain items of user interest. We will begin by determining the user-defined interest itemsets, the UDII query set $\mathcal{Q}$.

3.1 TRARM-RelSup

TRARM-RelSup finds both frequent rules as well as rare rules above a certain confidence level. This algorithm combines the efficiency of the itemset tree with that of the rare rule mining relative support method. Rules will be generated using this method from itemsets, I, that are above either minsup and minrelsup.

TRARM-RelSup algorithm: In order to find strong association rules containing UDII set = $\mathcal{Q}$ using itemset trees and relative support:

- Step 1: Build an itemset tree based upon the transactional dataset T.
- Step 2: For each query Q_i contained in the UDII set $\mathcal{Q}$, find all nodes $\mathcal{N}$ which contain Q_i in the tree, their supports, and the support of Q_i.
- Step 3: For each N_i in $\mathcal{N}$, return the set $\mathcal{C}$ of all itemsets whose support $\geq$ minsup OR whose relSup $\geq$ minRelSup.
- Step 4: For each C_i a subset of C_j in $\mathcal{C}$, remove C_i if $\text{support}(C_i) = \text{support}(C_j)$.
- Step 5: For each C_i in $\mathcal{C}$, generate rules with LHS = Q_i and RHS = all subsets of C_i - Q_i whose confidence $\geq$ minconf.

Using this algorithm, TRARM-RelSup is capable of discovering high support rules as well as rare rules with high confidence that would be missed by the original itemset tree algorithm. Using the transactional dataset in Example 1 and minsup = 40%, the rule $\{1\} \Rightarrow \{2\}$ with supp = 33% and confidence = 100% would not appear in the results set. TRARM-RelSup would find this rule, and return it as a part of its high confidence rare rules set.

4 Experimental Setup

This section outlines the experiments carried out for testing the TRARM-RelSup algorithm. First, there will be a discussion of the IBM Quest synthetic data generator [4] and the datasets created for the experiments. Second, this section will cover the UDII query sets generated to test various user interest patterns. Finally, the experiments carried out for testing the TRARM-RelSup algorithm will be outlined and comparisons will be discussed.

4.1 IBM Quest Data Generator

IBM QUEST was created by IBM's Quest research group and creates two sets of data D and T. The former is the database of transactions itself and the latter

is a set of n candidate frequent itemsets. T populates D by using a Poisson distribution to determine a size for the transactions. Each transaction is created using elements existing in T and are either already existing in the database, or are randomly chosen from T's unused elements. This models market basket style data which contains many co-occurrences [2]. The system is able to create realistic data in large quantities but predictions on algorithm behavior cannot be made. Due to its wide range of variation, it is hard to generalize via empirical analysis [2,4].

The datasets generated for the following experiments can be seen in Table 1.

Table 1. Experimental dataset description

	50 1-item datasets		100 1-item datasets	
Dataset Transactions	Number of nodes in the tree	Width of the itemset tree	Number of nodes in the tree	Width of the itemset tree
25,000	31,993	47	31,383	56
50,000	62,265	47	61,483	58
100,000	62,265	47	61,483	58
200,000	62,265	47	61,483	58

Each of these datasets has an average transaction size of 10 items and contains 50 or 100 unique 1-items. The number of transactions was varied in order to discern whether a larger tree will result in less rare rules due to its denser nature or vice versa. By varying the number of unique itemsets we are able to mimic the diversity found in a transactional dataset but also increase potential for both high and low support rules.

4.2 UDII Query Set

The UDII elements of $\mathcal{Q}$ used for the following experiments range between $1 \leq |Q_i| \leq 3$ and mimic a user's interest in specific items. $\mathcal{Q}$ was populated with itemsets such that their supports ranged between 0.005% and 0.15%. Approximately 50% of the Q_i were selected to contain 1 - 2 itemsets in common with other Q_j, where $Q_i \neq Q_j$. The other 50% are mutually exclusive.

4.3 Experiments

The following experiments are designed to test the ability of the proposed algorithm to discover rare rules that may potentially be interesting for the user.

Experiments for testing the TRARM-RelSup algorithm:

- Build itemset trees on all 8 IBM Quest datasets.
- Query all trees with the 8 UDII query sets using the original itemset tree and TRARM-RelSup algorithm.
- Generate rules from the resulting set $\mathcal{C}$ using the rule generation algorithm.
- Compare the Itemset Tree and TRARM-RelSup results sets.

The goal of these experiments is to demonstrate that TRARM-RelSup is capable of finding not only the strong rules found by the original itemset tree, but also a significant number of the rare rules the original algorithm would miss. Comparisons of the number of rules generated will be made for each of the 8 datasets with emphasis on strong rules discovered vs rare rules discovered. For each experiment, the minSup measure is varied in order to discern the effect of that parameter on rare rules set size and the strong rules set. For both experiment sets, the minRelSup and minConf remain constant as minsup varies. The variations of minsup as well as the values of minRelSup and miconf for the two experiment sets can be seen below:

- Experiment set 1: (minSup = 0.15%, minSup = 0.1%, minSup = 0.05%), minRelSup = 0.005%, minConf = 0.005%.
- Experiment set 2: (minSup = 0.15%, minSup = 0.1%, minSup = 0.05%), minRelSup = 0.02%, minConf = 0.02%.

All parameters were selected based upon the characteristics of the generated datasets.

5 Results

For each tree the number of rules generated ranged between 3,000 - 24,000 depending upon tree size and the number of unique 1-items. Tables 2 and 3 show the number of rules generated by the original itemset tree algorithm and TRARM-RelSup. The strong rules listed were found by both the original itemset tree algorithm and TRARM-RelSup. The rare rules listed were only found by the TRARM-RelSup algorithm.

Table 2. Experiment set 1: number of rules found with minRelSup = 0.005% and minconf = 0.005%

		minSup = 0.15%		minSup = 0.1%		minSup = 0.05%	
Transactions	1-items	strong	rare	strong	rare	strong	rare
25,000	50	293	23,745	478	23,560	1,198	22,840
50,000	50	260	19,096	440	18,916	987	18,369
100,000	50	122	1,846	133	1,835	152	1,816
200,000	50	101	1,024	104	1,021	106	1,019
25,000	100	111	14,457	179	14,389	545	14,023
50,000	100	110	13,085	177	13,018	463	12,732
100,000	100	110	11,324	177	11,257	458	10,976
200,000	100	110	13,085	177	13,018	463	12,732

The number of rules generated by each experiment can be seen in Tables 2 and 3. Several observations can be made based upon these results. First, as minRelSup and minconf decreases, the total number of rules discovered increases

Table 3. Experiment set 2: number of rules found with minRelSup = 0.02% and minconf = 0.02%

		minSup = 0.15%		minSup = 0.1%		minSup = 0.05%	
Transactions	1-items	strong	rare	strong	rare	strong	rare
25,000	50	293	4,786	478	4,601	1,198	3,881
50,000	50	260	3,492	440	3,312	987	2,765
100,000	50	122	1,123	133	1,112	152	1,193
200,000	0	101	1,016	104	1,013	106	1,011
25,000	100	111	2,888	179	2,820	545	2,454
50,000	100	110	2,158	177	2,091	463	1,805
100,000	100	110	1,991	177	1,924	458	1,643
200,000	100	110	2,158	177	2,091	463	1,805

dramatically. Second, decreasing minsup displays a correspondingly steady increases in the number of strong rules, but not as dramatic as the decrease in the number of rare rules using minRelSup. Third, as the number of 1-items in a transaction set increases from 25 to 50, the number of rules decreases marginally. This occurs because as the number of 1-items increases, the tree becomes more varied and the overall supports are reduced to compensate. Finally, as the numbers of transactions increase but the number of 1-items remain constant, the dataset repeats the same items resulting in a more concentrated smaller tree and vice versa. This results in a decrease in the number of rules discovered by the algorithms.

6 Conclusions and Future Work

Based upon the experiments performed for TRARM-RelSup, several observations can be made. First, for all experiments, TRARM-RelSup demonstrated a drastic increase in the number of rare rules discovered. Second, as minsup increases, the size of the strong rules set decreases as rules from the stronger are rule set are transferred to the rare rules set. Finally, given the large number of rare rules discovered by TRARM-RelSup when compared to just the number of strong rules discovered by both the original itemset algorithms, it is apparent that there are many rules missed using only minsup to discover potentially interesting itemsets. Even with an increase in the number of discovered itemsets, the linear nature of querying an itemset tree (where n is the number of transactions) [8] results in a manageable increase in rule mining complexity. Also the resulting rules sets are tailored to the user's specific interests.

Future work involving TRARM-RelSup will include exploration of the possibility of reducing a dataset T to T', where T' has only transactions containing the UDIIs. This is based upon the ARM-PDI-RT method proposed by Sha and Chen in 2011 for finding rare rules using Apriori [10]. Also explorations of removing minsup and only using minRelSup when querying an itemset tree are being contemplated.

References

1. Adda, M., Wu, L., Feng, Y.: Rare Itemset Mining Machine Learning and Applications. In: ICMLA 2007, pp. 73–80 (2007)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487–499 (1994)
3. Bing, L., Wynne, H., Yiming, M.: Mining association rules with multiple minimum supports. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 337–341 (1999)
4. Cooper, C., Zito, M.: Realistic synthetic data for testing association rule mining algorithms for market basket databases. In: Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, pp. 398–405 (2007)
5. Gedikli, F., Jannach, D.: Neighborhood-Restricted Mining and Weighted Application of Association Rules for Recommenders. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 157–165. Springer, Heidelberg (2010)
6. Hafez, A., Deogun, J., Raghavan, V.V.: The Item-Set Tree: A Data Structure for Data Mining. In: Mohania, M., Tjoa, A.M. (eds.) DaWaK 1999. LNCS, vol. 1676, pp. 183–192. Springer, Heidelberg (1999)
7. Kiran, R., Reddy, P.: An improved multiple minimum support based approach to mine rare association rules. In: Computational Intelligence and Data Mining, CIDM 2009, pp. 340–347 (2009)
8. Kubat, M., Hafez, A., Raghavan, V., Lekkala, J., Chen, W.: Itemset trees for targeted association querying. IEEE Transactions on Knowledge and Data Engineering, 1522–1534 (2003)
9. Li, Y., Kubat, M.: Searching for high-support itemsets in itemset trees. Intell. Data Anal., 105–120 (2006)
10. Sha, Z., Chen, J.: Mining association rules from dataset containing predetermined decision itemset and rare transactions. In: Seventh International Conference on Natural Computation, vol. 1, pp. 166–170 (2011)
11. Srikant, R., Agrawal, R.: Mining generalized association rules. In: Proceedings of the 21st International Conference on Very Large Data Bases, VLDB 1995, pp. 407–419 (1995)
12. Srikant, R., Agrawal, R.: Mining quantitative association rules in large relational tables. SIGMOD Rec., 1–12 (1994)
13. Szathmary, L., Napoli, A., Valtchev, P.: Towards rare itemset mining. Tools with Artificial Intelligence, 305–312 (2007)
14. Yun, H., Ha, D., Hwang, B., Ryu, K.: Mining association rules on significant rare data using relative support. Journal of Systems and Software 67, 181–191 (2003)