# Mining Associative Classification without Candidate Rules

Panida Songram

***Abstract*—Associative classification is a combination of association rule mining and classification for prediction. For mining associative classification, traditional algorithms generate the complete set of association rules, and then use a minimum confidence threshold to select interesting rules for classification. If the number of association rules is very large, it is time consuming to select only the interesting rules. In this paper, a new algorithm, called TOPAC (Top Associative Classification), is proposed to solve the problem. The TOPAC algorithm directly produces the interesting rules without the generation of candidate rules. Moreover, it discovers the interesting rules based on frequent closed itemsets to reduce the redundancy rules.**

***Index Terms*—Associative classification, association rule, closed itemset, high confidence.**

## I. Introduction

CBA was introduced to integrate association rule mining and classification for prediction. It first generates all frequent itemsets based on the Apriori algorithm [1] to find candidate rules. Then the most effective rules are selected from the candidate rules to form a classifier. Since the Apriori algorithm finds frequent itemsets from all possible candidate itemsets, CBA consumes time to find actual frequent itemsets. Like CBA, CMAR generates association rules from frequent itemsets and then selects the most effective rules from them to form a classifier. It adopts the FP-Growth algorithm [9] to minimize candidate generation in CBA and proposes multiple rules for prediction. CMAR has been shown to outperform CBA in accuracy. CPAR (Classification based on Predictive Association Rules) adopts FOIL [10] to avoid the generation of a large number of candidate rules and uses the PNArray data structure [4] to reduce time consumed by FOIL. MMAC was proposed to combine conflicting rules into one multi-class label rule in the form $X \rightarrow c_1, c_2, ..., c_k$ where $c_1, c_2, ..., c_k$ is a list of ranked class labels associated with the frequent itemset $X$. MMAC scans datasets to generate a complete set of association rules and discovers more rules having no lower than the minimum support and minimum confidence threshold from the remaining unclassified until no further frequent items can be found. Then the rules will be merged to form multi-class label rules to be used for prediction. MCAR improves the efficiency of the rule discovery by using the transaction IDs intersection method [11]. It stores items along with their transaction ids in arrays and then intersects the transaction ids to find frequent itemsets. Using this method is an efficient approach to obtaining frequent items that involve more than one attribute. Furthermore, the support and confidence values are easily found. In addition, MCAR uses a rule ranking method to ensure that the rules with high confidence are part of the classifier. ACAC was proposed to mine association rules based on Apriori, but it adds all-confidence threshold in the mining process. Association rules satisfying a minimum support and all-confidence threshold are selected as candidate rules, then the candidate rules will be selected to be in a classifier if they pass a given confidence threshold. Unlike the previously mentioned algorithms, ACCF produces association rules based on closed frequent itemsets by adopting the CHARM algorithm [12]. Therefore, it can reduce a large number of generated frequent itemsets and association rules. In addition, ACCF gives a small set of predictive rules with high quality and no redundancy. However, it has to generate candidate rules and then prune them by using a minimum confidence threshold.

The above mentioned algorithms mine frequent itemsets or closed frequent itemsets first and then match them with classes to generate association rules as candidate rules, then the interesting rules for prediction are selected from the candidate rules by using a given minimum confidence threshold. Unfortunately, if a large number of candidate rules are generated, it will take significant effort to select the interesting rules from the large number of candidate rules. Furthermore, they may miss interesting rules whose supports are below the minimum support threshold but have high confidence [13]. Such rules have a good classification power, but are missing.

In this paper, a new efficient algorithm, called TOPAC, is proposed to mine classification rules without candidate rules generated. TOPAC produces only classification rules whose confidence pass a given minimum confidence threshold. Therefore, it does not use effort to generate unnecessary rules. Moreover, TOPAC produces the classification rules based on closed itemsets to give a small number of high quality predictive rules with no redundancy.

The remainder of the paper is organized as follows. Some basic definitions are given in Section 2. The TOPAC algorithm is presented in Section 3. The performance of TOPAC is analyzed in Section 4. Finally, the paper is concluded in Section 5.

Manuscript received April 6, 2012; revised May 4, 2012.

Panida Songram is with Department of Computer Science, Faculty of Informatics, Mahasarakham University, Mahasarakham, 44150, Thailand.

## II. Basic Definitions

Let *D* be a set of transactions, *I* be a set of items and *C* be a set of classes in the database, each transaction $d \in D$ follows the scheme $(i_1, i_2, ..., i_k, c)$ where $i \in I$ and $c \in C$. An itemset *X* is a set of items. The support of itemset *X* is a number of transactions containing *X*, denoted as $supp(X)$. Length of itemset *X* is a number of items, denoted as *l-itemset.* Given a minimum support threshold *min_supp* , an itemset *X* is a frequent itemset if $supp(X) \geq min\_supp$ . An itemset *X* is closed if and only if $\varsigma(X) = f(g(X)) = fog(X) = X$ , where the composite function $\varsigma = fog$ is called a Galois operator or closure operator and the function *g* returns a set of transactions supporting a given itemset *X* , and the function *f* returns the largest itemset included in $g(X)$ [14, 15]. $|g(X)|$ is the number of transactions supporting *X* that is the support of *X*. A classification rule is formed $r: X \to c$ . The support of rule *r* is a number of transactions containing *X* and *c*. The confidence of *r* is $supp(Xc)/supp(X)$ [2, 7] that is $|g(Xc)|/|g(X)|$ , denoted as $conf(r)$ . Given a minimum confidence threshold *min_conf* , the rule is confident or interesting if $conf(r) \geq min\_conf$ .

## III. The TOPAC Algorithm

### A. Generation Closed Itemsets

TOPAC is proposed to generate rules from closed itemsets, because the number of rules is smaller than the rules generated from frequent itemsets. Moreover, the rule can express more information than those generated from frequent itemsets. TOPAC discovers closed itemsets by adopting the transaction IDs intersection method to find a closed itemset based on lemma 1 [2, 3].

**Lemma 1** Given an itemset *X* and an item $i \in I, g(X) \subseteq g(\{i\})$ if and only if $i \in \varsigma(X)$ .

From Lemma 1, the same closure may be calculated twice from two different itemsets. Therefore, an itemset has to be checked for duplication first before producing a closed itemsets. An itemset *X* is checked for duplication with post-items. If $\exists l \in post-items(X)$ such that $g(X) \subseteq g(\{l\})$ , then itemset *X* is discarded. Otherwise, it is computed to find a closed itemsets with pre-items [16]. If $j \in pre-items(X)$ and $g(X) \subseteq g(j)$ , then $j \in \varsigma(X)$ . After a closed itemset discovery, the closed itemset will be extended with pre-items which are not included in the closed itemset. Then the extended itemsets are used to find other closed itemsets.

**Example 1** The table I shows transaction ids of each item and class in the dataset. The distinct items are sorted in descending order of the number of transactions. The closed itemset *be* is extended with all items that are not included in *be*, i.e., items *a*, c and *d*. Therefore itemset *be* can be extended to *bea*, *bec* and *bed*. All extended itemsets will be checked for duplication first. For example, *bea* is checked for duplication with its post-items that are *c* and *d*, items appear after item *a* in the sorted items. Itemset *bea* is not duplicate because *g*(*bea*) ={1,3,4,5}, $g(bea) \not\subset g(c)$ and $g(bea) \not\subset g(d)$ . Therefore, it is used to generate a closed itemset by included pre-items whose transactions containing transactions of *bea*. There is no any pre-items hold the transaction, so *bea* is a closed itemset.

TABLE I: Transactions of each Item and Class

| Item | Transactions |
|---|---|
| *b* | 1, 2, 3, 4, 5, 6 |
| *e* | 1, 2, 3, 4, 5 |
| *a* | 1, 3, 4, 5 |
| *c* | 2, 4, 5, 6 |
| *d* | 1, 3, 5, 6 |
| *Y* | 1, 2, 3 |
| *N* | 4, 5, 6 |

### B. Generating Rules with 100% Confidence

To find the high quality rules for prediction, the rules with high confidence should be found first. In the first step, TOPAC divides datasets according to classes into sub-datasets and then mines the sub-datasets as shown in Fig.1. At the first level, TOPAC discovers the largest itemsets contained in the transactions of *c*. Then the largest itemsets of all sub-transactions of the class are generated in descending order of the number of transactions. The largest itemsets at the first level are generated by using the items union method. For example, the largest itemset contained in transactions of class *Y* can be found by including items contained in transactions 1, 2 and 3 that is *be*. If the largest itemset *X* holds $g(X) \subseteq g(c)$, then $conf(X \to c) = 100\%$ based on theorem 1. If the largest itemset *X* is produced and got the same as ready-produced itemset *Y*. It will be pruned.

**Theorem 1** Given $X \to c$ , *X* is a closed itemset and *c* is a class, if $g(X) \subseteq g(c)$ then $conf(X \to c) = 100\%$

**Proof** if $g(X) \subseteq g(c)$, then $g(Xc) = g(X)$ .

Therefore, $conf(X \to c) = |g(Xc)|/|g(X)| = 1$ .

In conclusion, $conf(X \to c) = 100\%$

**Example 2** From Fig. 1, $g(beac) \subseteq g(N)$ . Therefore, $conf(beac \to N) = 100\%$ .

The largest itemsets are all closed itemsets because there is no itemsets that is larger than them with the same support. If a closed itemset giving the rule having lower than 100% confidence is found, it is extended to find other rules based on the generation of closed itemsets in section 3.1. If itemset *X* gives a rule with lower than minimum confidence, *X* will be stopped from extending because $conf(X \cup \{i\} \to c) < conf(X \to c)$ as shown in theorem 2.

**Theorem 2** Given $X \to c$ , *X* is a closed itemset and *c* is a class, if $g(Xc) \subseteq g(X)$ , then

$conf(X \cup \{i\} \to c) < conf(X \to c)$

**Proof** $g(X \cup \{i\}) \subseteq g(X)$ , so there is a set number of transactions *T* contained in $g(X)$ , but it is not contained in $g(X \cup \{i\})$ , $g(X \cup \{i\}) = g(X) - T$ .

$conf(X \cup \{i\} \to c) = |g(X \cup \{i\}) \cap g(c)|/|g(X \cup \{i\})|$ .

Since $g(Xc) \subseteq g(X)$ and the numerator is not more than the denominator, $conf(X \cup \{i\} \to c) < conf(X \to c)$ .

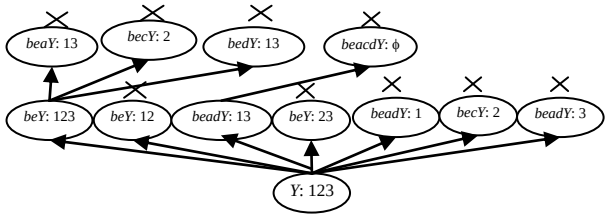
**Example 3** $conf(bc \to N) = |g(bcN)|/|g(bc)|$

$= |\{3,4,5,6\}| / |\{2,4,5,6\}| = 3/4$ . When item *e* is added,

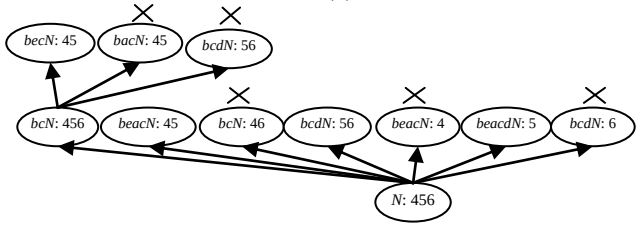
$$conf(bce \rightarrow N) = |g(bceN)|/|g(bce)|$$
$$= |\{4,5,6\} - \{6\}|/|\{2,4,5,6\} - \{6\}|$$
$$= |\{4,5\}|/|\{2,4,5\}| = 2/3$$

In conclusion, $conf(bc \rightarrow N) < conf(bec \rightarrow N)$.

The mining process step of TOPAC is displayed in Fig.2. After the mining process is stopped, there are only two interesting rules generated belonging to class *Y* as shown in Fig.1, i.e., *bead*→*Y*: 2/3 and *be*→*Y*: 3/5 (rule:confidence). The interesting rules generated belonging to class *N* are beac→*N* : 2/2, *bcd*→*N*: 2/2, *beacd*→*N*: 1/1, *bec*→*N*: 2/3 and *bc*→*N*: 3/4.




(a)




(b)

Fig. 1. Interesting rules belonging (a) class *Y* and (b) class *N*

1. **Algorithm**: TOPAC (*min_conf*)
2. **Input:** A database *D*, A minimum confident *min_conf*.
3. **Output:** A set of top rules *TR*.
4. Method:
5. scan database and find a set of transactions of each item and class
6. sort distinct items in descending order of their supports
7. divide dataset according to class $c_1, c_2, \ldots c_k$
8. for each $c_i$
9. find the largest itemsets $X_j$ containing in $g(c_i)$ and all subset of $g(c_i)$
10. for each $X_j$
11. if $g(X_j) \subseteq g(c_i)$ then $conf(r: X_j \rightarrow c_i)$=100%
12. $TR = TR \cup r$
13. else if $conf(r: X_j \rightarrow c) = |g(X_j c)|/|g(X_j)| \geq min\_conf$ then
14. $TR = TR \cup r$
15. extend $X_j$ with all items $i \in$ pre-items($X_j$) ; $Y_h = X_j \cup i$
16. for each $Y_h$
17. if $conf(r: Y_h \rightarrow c) = |g(Y_h c)|/|g(Y_h)| \geq min\_conf$ and !duplicate($Y_h$) then
18. find $\varsigma(Y_h)$
19. $r: \varsigma(Y_h) \rightarrow c$
20. $TR = TR \cup r$
21. extend $\varsigma(Y_h)$ with all $i \in$ pre-items($\varsigma(Y_h)$)
22. for each extended itemsets
23. do step 17-23

Fig. 2. The TOPAC Algorithm

## IV. Performance Analysis

### A. Space Analysis

The traditional algorithms produce all frequent itemsets and then combine them with classes to produce the candidate rules. Given *FI* is a set of frequent itemsets and *C* is a set of classes, the number of candidate rules is $|FI|.|C|$. Therefore, the memory allocation of the traditional algorithms is $O(|FI|.|C|)$. The ACCF algorithm generates the candidate rules from all closed itemsets combined with classes. Given *CI* is a set of closed itemsets, the memory allocation of ACCF is $O(|CI|.|C|)$. Since $|CI| \leq |FI|$, the number of candidate rules generated by the ACCF algorithm is always smaller than the number of candidate rules generated by the traditional algorithms. For TOPAC algorithm, it produces only a set of closed itemsets (*CF*) which lead to the rules having confidence not lower than *min_conf*. Therefore, the worst case memory allocation of TOPAC is $O(|CF|.|C|)$. TOPAC allocates only the interesting rules, which is smaller than the number of candidate rules generated by the ACCF algorithm. The TOPAC algorithm is therefore more efficient than the ACCF algorithm in term of space allocation.

### B. Time Analysis

The traditional algorithms are time consuming to produce frequent itemsets for producing the candidate rules. If the task of finding all frequent itemsets is essentially linear in the database size, the time complexity for finding the frequent itemsets is $O(|FI|.|D|.2^{|I|})$ [12]. The frequent itemsets are matched with classes to get the candidate rules using $O(|FI|.|C|)$ and then the interesting rules are selected from the candidates by using $O(|FI|.|C|)$. Therefore, the overall cost of the traditional algorithm is $O((|FI|.|D|.2^{|I|}) + (|FI|.|C|)^2)$. Unlike the tradition algorithms, ACCF produces closed itemsets to generate the candidate rules based on the CHARM algorithm. It uses $O(|CI|.|D|))$ to produce the closed itemsets [12]. Therefore, ACCF costs $O((|CI|.|D|) + (|CI|.|C|)^2)$. The TOPAC algorithm produces only closed itemsets leading to the rules having confidence not lower than *min_conf*. In addition, it does not need to match closed itemsets with classes. Therefore, TOPAC costs $O((|CF|.|D|))$ that is more efficient than the traditional algorithms and ACCF in term of computation time.

## V. Conclusion

This paper proposes a new algorithm, called TOPAC, for mining associative classification. The TOPAC algorithm produces interesting rules for prediction without candidate rules generated. Firstly, the rules with high confidence are produced and then the closed itemsets having lower than 100% confidence will be extended to find other interesting rules. If the closed itemset has a lower minimum confidence, it will be stopped from extending. The rules with lower

minimum confidence are not produced. Therefore, TOPAC is efficient in terms of time and memory space.

## REFERENCES


[1] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," in *Proceedings of the 1993 ACM SIGMOD Internatinal Conference on Mangament of Data*, USA, 1993, pp. 207-216.

[2] B. Liu, "Integrating Classification and Association Rule Mining," in *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, New York, 1998, pp. 80-86.

[3] W. Li, J. Han, and J. Pei, "CMAR: Accurate and Efficient Classification based on Multiple Class-associatin rules.," in *Proceedings of the International Conference on Data Mining*, San Jose, CA, Nov. 2001, pp. 369-376.

[4] X. Yin and J. Han, "CPAR: Classification based on Predictive Association Rules," in *Preceedings of the SIAM Internatinal Conference on Data Mining*, San Francisco, CA, 2003, pp. 369-376.

[5] F. A. Thabtah, P. Cowling, and Y. Peng, "MMAC: A New Multi-class, Multi-label Associative Classificatin Approach," in *Proceeding of the 4th IEEE International Conference on Data Mining ( ICDM'04)*, Brighton, UK, 2004, pp. 217-224.

[6] F. A. Thabtah, P. Cowling, and Y. Peng, "MCAR: Multi-class classificatin based on association rule approach," in *Proceedings of the 3rd IEEE Internatinal Conference on Computer Systems and Applicatins*, Cario, Ebypt, 2005, pp. 1-7.

[7] X. Li, D. Qin, and C. Yu, "ACCF: Associative Classification Based on Closed Frequent Itemsets," in *Proceedings of the Fith International Conference on Fuzzy Systems and Knowledge Discovery*, Shandong, 2008, pp. 380-384.

[8] Z. Huang, Z. Zhou, T. He, and X. Wang, "ACAC: Associative Classification Based on All-Confidence," in *Proceedings of IEEE International Conference on Granular Computing*, 2011, pp. 289-293.

[9] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of DATA*, Dallas, Tx, 2000, pp. 1-12.

[10] J. R. Quinlan and R. M. Cameron-Jones, "FOIL: A midterm report," in *Proceedings of 1993 European Conference Machine Learning*, Vienna, Austria, 1993, pp. 3-20.

[11] M. J. Zaki, S. Parthasarathy, M. Ogihar, and W. Li, "New algorithms for fast discovery of association rules " in *Proceedings of 3rd International Conference on Knowledge Discovery and Data Mining*, USA, 1997, pp. 283-286.

[12] M. J. Zaki and C.-J. hsiao, "CHARM: An Efficient Algorithm for Closed Itemset Mining," in *Proceedings of the 2nd SIAM International Conference on Data Mining*, Arlington, Virginia, USA, April 2002, pp. 34-43.

[13] J. Li and X. Zhang, "Efficient Mining of High Confidence Association Rules without Support Thresholds," in *Proceedings of 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases*, Prague, 1999, pp. 406-411.

[14] J. Wang, J. Han, Y. Lu, and P. Tzvetkov, "TFP: An Efficient Algorithm for Mining Top-k Frequent Closed Itemsets," in *Proceedings of the 2002 IEEE Internatinal Conference on Data Mining*, Washington, DC, USA, 2002, pp. 211-218.

[15] J. Wang, J. Han, Y. Lu, and P. Tzvetkov, "TFP: An Efficient Algorithm for Mining Top-k Frequent Closed Itemsets," *Journal of IEEE Transaction on Knowledge and data mining*, vol. 17, pp. 652-664, 2005.

[16] P. Songram and V. Boonjing, "N-Most Interesting Closed itemset Mining," in *Proceedings of 3rd International Conference on Convergence and Hybrid Information Technology*, South Korea, 2008, pp. 619-624.