Considering Re-occurring Features in Associative Classifiers

Rafal Rak, Wojciech Stach, Osmar R. Zaïane, and Maria-Luiza Antonie

University of Alberta, Edmonton, Canada
{rrak, wstach}@ece.ualberta.ca
{zaiane, luiza}@cs.ualberta.ca

Abstract. There are numerous different classification methods; among the many we can cite associative classifiers. This newly suggested model uses association rule mining to generate classification rules associating observed features with class labels. Given the binary nature of association rules, these classification models do not take into account repetition of features when categorizing. In this paper, we enhance the idea of associative classifiers with associations with re-occurring items and show that this mixture produces a good model for classification when repetition of observed features is relevant in the data mining application at hand.

1 Introduction

Classification is one of the most common tasks in data mining and machine learning. By and large, it consists of extracting relevant features from labelled training data to build a model that discriminates between classes for unlabelled observed objects. Myriad techniques have been proposed and while there are, in general, better approaches than others, there is no clear winner in terms of correctness and usability given a particular problem application.

Associative classification is a relatively new method. The main objective is to discover strong patterns that are associated with the class labels in the training set. The training set is modeled into transactions with items being the observed features. As a final classification model, one obtains a set of association rules associating features with class labels. In the literature, there are few known classifiers based on the above-mentioned idea, i.e. CBA [4], CMAR [3], and ARC-AC/ARC-BC [9].

One considerable limitation of all these algorithms is that they do not handle the observations with repeated features. In other words, if a data object is described with repeated features, only the presence of the feature is considered, but not its repetition. However, in many applications such as medical image categorization or other multimedia classification problems, the repetition of the feature may carry more information than the existence of the feature itself [10]. Also in text mining and information retrieval, it is widely recognized that the repetition of words is significant and symptomatic, hence the common use of TF/IDF (i.e. the frequency of a term in a document relative to the frequency of the term in a collection).

Associative classifiers use association rule mining to build a classification model. However, association rule mining typically considers binary transactions; transactions that indicate presence or absence of items. Binary transactions simply do not model repetitions. A few approaches to mining association rules with re-occurring items have been proposed, such as MaxOccur [10], FP'-tree [5] and WAR [8]. The main goal of our research is to devise a classifier that combines the idea of associative classification and association rules with reoccurring items. Our contributions presented in this paper exploit, combine, and extend the ideas mentioned above, especially ARC-BC and MaxOccur algorithms. We also suggest new strategies to select rules for classification from the set of discovered association rules. Our hypothesis is that associative classifiers with recurrent items have more discriminatory power since they maintain and exploit more information about both objects and rules.

A delicate issue with associative classifiers is the use of a subtle parameter: support. Support is a difficult threshold to set, inherited from association rule mining. It indicates the proportion of the database transactions that support the presence of an item (or object). It is known in the association rule mining field that the support threshold is not obvious to tune in practice. In the associative classification literature it has been commonly and arbitrarily set to 0.1%. However, the accuracy of the classifier can be very sensitive to this parameter. In the case of re-occurring items, there are two ways of calculating support: transaction-based support and object-based support [10] (i.e. either the proportion of transactions or the proportion of objects that support the existence of an object in the database). Our experiments show that an associative classifier that considers re-occurrence of features is considerably less sensitive to the variation of support. This leads to more practical applications and eventually the possibility to automatically determine and tune this parameter.

The remainder of the paper is organized as follows: Section 2 presents the problem statement: the model of an associative classifier and the consideration in the model of recurrent items. Related work on associative classification and mining association rules with repetitions is also presented in Section 2. We present our new approach ACRI in Section 3. The experiments showing the performance of our approach are presented in Section 4. Section 5 offers some conclusions.

2 Problem Statement and Related Work

The first known classifier using association rules was introduced in [4]. The main idea was to modify the form of transactions known from the traditional approach to the form of $< \{i_1, i_2, ...i_n\}, c >$, where i_k is an item in a transaction and c is a class label. In other words, objects in a training set are represented by sets of features appended with the observed class label. All the rules generated from frequent itemsets are restricted to those with a class label as a consequent.

Our task is to combine the associative classification with the problem of recurrent items. More formally, it can be stated that our goal is to modify the original approach using transactions to the form of $< \{o_1i_1, o_2i_2, ...o_ni_n\}, c >$, where o_k is the number of the occurrences of the item i_k in the transaction.

Association rules have been recognized as a useful tool for finding interesting hidden patterns in transactional databases. Several different techniques have been introduced. However less research has been done considering transactions with reoccurrence of items. In [8], the authors assign weights to items in transactions and introduce the WAR algorithm to mine the rules. This method is two fold: in the first step frequent itemsets are generated without considering weights and then weighted association rules (WARs) are derived from each of these itemsets. MaxOccur algorithm [10] is an efficient Apriori-based method for discovering association rules with recurrent items. It reduces the search space by effective usage of joining and pruning techniques. The FP'-tree approach presented in [5] extends the FP-tree design [2] with a combination from the MaxOccur idea. For every distinct number of occurrences of given item, the separated node is created. In case when a new transaction is inserted into the tree, it might increase support count for the different path(s) of the tree as well. This is based on the intersection between these two itemsets. Given the complete tree, the enumeration process to find frequent patterns is similar to that from the FP-tree approach. None of the existing associative classifier uses reoccurrence.

3 The Proposed Approach

Our approach, ACRI (Associative Classifier with Reoccurring Items), consists of two modules: Rule generator and classifier. We decided to base our algorithm for mining associations with reoccurring items on Apriori-based MaxOccur. The building of the classification model follows our previous ARC-BC approach. The rational is based on the efficiency of this method in the case of non-evenly distributed class labels. Indeed other associative classification methods are biased towards dominant classes in the case when rare classes exist. Rare classes are classes with very few representatives in the training set. MaxOccur run on transactions from each known class separately makes the core of our rule generator module. It mines the set of rules with reoccurring items from the training set. These rules associate a condition set with a class label such that the condition set may contain items preceded by a repetition counter. The classification process might be considered as plain matching of the rules in the model to the features of an object to classify. Different classification rules may match, thus the classifier module applies diverse strategies to select the appropriate rules to use. In addition, simple matching is sometimes not possible because there is no rule that has the antecedent contained in the feature set extracted from the object to classify. With other associative classifiers, a default rule is applied, either the rule with the highest confidence in the model or simply assigning the label of the dominant class. Our ACRI approach has a different strategy allowing partial matching or closest matching by modeling antecedents of rules and new objects in a vector space. The following elaborates on both modules.

Rule Generator: This module is designed for finding all frequent rules in the form of $< \{o_1 i_1, o_2 i_2, \ldots, o_n i_n\}, c >$ from a given set of transactions. The modules's general framework is based on ARC-BC [9]: transactions are divided into N subsets - each for one given class (N is equal to the number of classes); Once rules are generated for each individual class, the rules are merged to form a classification model. The rule generator for each class C_x is an Apriori-based algorithm for mining frequent itemsets that extends the original method by taking into account reoccurrences of items in a single transaction à la MaxOccur [10]. In order to deal with this problem, the support count was redefined. Typically, a support count is the number of transactions that contain an item. In our approach, the main difference is that single transactions may increase the support of a given itemset by more than one. The formal definition of this approach is as follows. A transaction $T =< \{o_1 i_1, o_2 i_2, \ldots, o_n i_n\}, c >$ supports itemset $I = \{l_1 i_1, l_2 i_2, \ldots, l_n i_n\}$ if and only if $\forall i = 1..n l_1 \leq o_1 \wedge l_2 \leq o_2 \wedge \ldots \wedge l_n \leq o_n$. The number t by which T supports I is calculated according to the formula: $t = min[\frac{o_i}{l_i}] \forall i = 1..n, l_i \neq 0 \wedge o_i \neq 0$.

The Classifier: This module labels new objects based on the set of mined rules obtained from the rule generator. An associative classifier is a rule-based classification system, which means that an object is labelled on the basis of a matched rule (or set of rules in case of multi-class classification). This task is simple if there is an exact match between a rule and an object. The model, however, often does not include any rule that matches a given object exactly. In such a case, in order to make the classification, all rules are ranked according to a given scenario and the best one (or several) is matched to a given object. Rule ranking might be performed following different strategies, which associate each rule to a number that reflects its similarity to a given object. These strategies may be used either separately or in different combinations. We have tested *cosine measure, coverage, dominant matching class, support* and *confidence*. Let us consider the rule $< \{o_1 i_1, o_2 i_2, \ldots, o_n i_n\}, c >$ and the object to be classified $< l_1 i_1, l_2 i_2, \ldots, l_n i_n >$. The corresponding n-dimensional vectors can be denoted as $\overrightarrow{0} = [o_1, o_2, \ldots, o_n]$ and $\overrightarrow{l} = [l_1, l_2, \ldots, l_n]$. The **Cosine measure (CM)** assigns a value that is equal to the angle between these two vectors, i.e. The smaller the CM value is, the smaller the angle, and the closer these vectors are in the n-dimensional space. **Coverage (CV)** assigns a value that is equal to the ratio of the number of common items in the object and rule to the number of items in the rule (ignoring reoccurrences). In this case, the larger the CV ratio is, the more items are common for the rule and the object. CV=1 means that the rule is entirely contained in the object. With **Dominant matching class**, the class label is assigned to the object by choosing the one being the most frequent from the set of rules matching the new object. Notice that dominance can be counted by simply enumerating the matching rules per class or a weighted count using the respective confidences of the matching rules. The **support** and **confidence** are used to rank rules. They refer to the rule property only and do not depend on the classified object. Thus, they have to be used with other measures that prune the rule set.

4 Experiments

We tested ACRI on different datasets to evaluate the best rule selection strategy as well as compare ACRI with an associative classifier like ARC-BC. As an example, we report here an experiment with the mushroom dataset from the UCI repository [7]. It appears that the rule selection strategies have roughly similar performance in terms of accuracy. However, this accuracy varies with the support threshold. The lower the support, the more rules are discovered allowing a better result using selection based on cosine measure for example. Using the dominant matching class was also doing well, confirming the benefit of the dominance factor introduced in [9]. The selection based on best rule support was not satisfactory in general and is not reported here. We also observed that coverage (CV) gave better results when set to 1. Thus all results reported herein have CV set to 1. The other measures are comparable in performance and trend, except for best confidence. When the support threshold is high, fewer rules are discovered and confidence tends to provide better results while the cosine measure returns matches that have big angles separating them from the object to classify, hence the lower accuracy. Figure 1 on the left shows the superiority of the rule selection strategy dominant matching class up to a support threshold of 25%, beyond which best confidence becomes a winning strategy. Figure 1 on the right shows how the more rules are discovered the more effective in terms of accuracy the strategies dominant matching class and cosine measure becomes in comparison to best confidence approach. The number of rules is correlated with support.

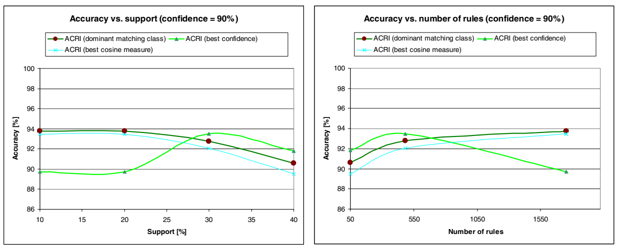

Fig. 1. Accuracy of rule selection strategies vis-à-vis support and number of rules

We compare ACRI with ARC-BC using the Reuters-21578 text collection as in the paper presenting ARC-BC for text categorization using the top 10 topics [6]. The total of 9980 documents is split into two sets: 7193 and 2787 for a training and test set respectively. At first, we tested both approachesACRI and ARC-BC using relatively high support. We produced several different sets of rules to be used in the classifier. For ARC-BC we chose the support threshold range from 10 to 30% with the step of 5%; and 15 to 65% with the same step for our approach. The difference between the support thresholds lies in the definition of support for mining rules with recurrent items. A single document can support a set of words more than once. Therefore, if we consider support as the ratio of support count to the total number of transactions, as it was introduced in [5], we may encounter support more than 100% for some itemsets. On the other hand, if we choose the definition presented in [10], i.e., the ratio of support count to the number of distinct items (words), the support will never reach 100%. Actually, in practice, the latter support definition requires for setting very small thresholds to obtain reasonable results. Hence, we decided to use the first one as it is more similar to the "classical" definition of support. It is important to notice that no matter which definition we choose, it eventually leads to setting the same support count with ARC-BC. For each support threshold we set three different confidence thresholds: 0, 35, and 70%. The latter threshold was used in [9] as minimum reasonable threshold for producing rules; the first one (no threshold) was introduced to observe the reaction of the classifier for dealing with a large number of rules; and the threshold of 35% is simply the middle value between the two others. For each single experiment we tried to keep the level of more then 98% of classified objects, which resulted in manipulating the coverage CV from 0.3 to 1. We discarded cases for which it was not possible to set CV to satisfy the minimum number of classified objects. More than 90% of the remaining results had CV = 1. We also performed experiments without specifying CV (using different methods of choosing applicable rules); however, they eventually produced lower accuracy than those with specified CV ¿ 0.3. We used different classification techniques for choosing the most applicable rule matching the object. Best confidence and dominant matching class matching methods were utilized for both ARC-BC and ACRI approaches. Additionally, ACRI was tested with the cosine measure technique. So for all experiments herein reported the coverage (CV) is set to 1. In other words, for a rule to be selected for classification, all features expressed in the antecedent of the rule have to be observed in the new object to classify. We also performed tests with combination of matching techniques with different tolerance factors for each test. An example scenario in Figure 2 A, combines cosine measure, dominant matching class and best confidence: (1) choose top 20% of rules with the best cosine measure, then (2) choose 50% of the remaining rules with the highest confidence, and then (3) choose the rule based on the dominant class technique. We also did a battery of tests using relatively low supports. This significantly increases the number of classification rules. We varied the support between 0 and 0.1% and compared the harmonic average of precision and recall (F1 measure) for the same cases as before: Best confidence and dominant matching class for both ARC-BC and ACRI approaches, and the cosine measure technique for ACRI (Figures 2 E-F).

Categorizing documents from the Reuters dataset was best performed when the confidence level of the rules was at the 35% threshold for both the ACRI and ARC-BC approaches. For ARC-BC classifier, the best strategy was to use dominant factor, whereas in case of ACRI combination of cosine measure and confidence factors worked best. Figure 2 A shows the relationship between support and accuracy for these approaches. Comparing the best-found results, ARC-BC slightly outperforms the ACRI using the dominant matching class strategy at the 20% support level. However, ARC-BC seems to be more sensitive to changes of the support threshold. The accuracy of ACRI virtually does not depend on the

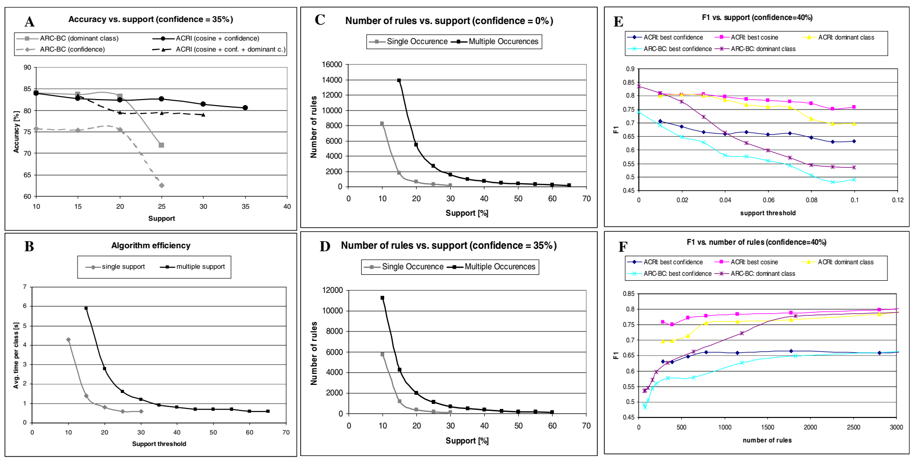

Fig. 2. A: Accuracy of ACRI and ARC-BC with high support; B: Algorithms CPU time efficiency; C: Number of rules with confidence $> 0\%$; D: Number of rules with confidence $> 35\%$; E: Effectiveness at low support; F: Effectiveness versus size of model

support threshold and is stable. In the case of ARC-BC the accuracy decreases significantly when this support is greater than 20%.

Figures 2 C and D show the number of generated rules with and without recurrent items. Considering recurrences results in having more rules, this has its origin in different support definition. The other interesting relationship is that by increasing the confidence threshold from 0% to 35%, the difference between number of rules decreases more rapidly for ACRI.

Experiments using low support thresholds confirm the stability of ACRI with regard to support. When varying the support from 0% to 0.1% ARC-BC loses in precision and recall while ACRI remains relatively consistent or looses effectiveness on a slower pace. Figure 2 E also shows that ACRI outperforms ARC-BC at these lower support thresholds. Using the cosine measure for selecting rules appears to be the best strategy. The cosine measure is also the best rule selection strategy when considering the number of rules discovered. In addition, the more rules are available the more effective the cosine measure becomes at selecting the right discriminant rules.

Figure 2 B shows the relationship between running time for rule generator with and without considering recurrent items. The algorithm with recurrences is slower, since it has to search a larger space, yet the differences become smaller when increasing the support threshold.

The best results for ACR-BC were found in [9] for confidence threshold greater than 70%. However, our experiments show that effectiveness is better on lower confidence for both ARC-BC and ACRI approaches. In other words, some classification rules with low confidence have more discriminant power and are selected by our rule selection strategies. This discrepancy with previous results may be explained by the use of the different method of counting support and confidence or/and by the fact that our classifier ACRI with re-occurring items and without re-occurrence consideration to simulate ARC-BC is using a different setup for rule selections.

5 Conclusion and Future Work

In this paper we introduced the idea of combining associative classification and mining frequent itemsets with recurrent items. We combined these two and presented ACRI, a new approach of associative classification with recurrent items. We also suggest new strategies to select classification rules during the classification phase. In particular, using the cosine measure to estimate the similarity between objects to classify and available rules is found very effective for associative classifiers that consider re-occurrence. When comparing our ACRI approach with other associative classifiers represented by ARC-BC we found that considering repetitions of observed features is beneficial. In particular in the case of text categorization, repetition of words has discriminant power and taking these repetitions in consideration can generated good classification rules. Our experiments also show that ACRI becomes more effective as the number of rules increases in particular with our cosine measure for rule selection. Moreover, ACRI seems to be less sensitive, with respect to accuracy, to the support threshold, while other associative classifiers are typically very sensitive to the support threshold which is very difficult to determine effectively in practice. This research is still preliminary. We intend to investigate the possibility to eliminate the need for the support threshold by automatically selecting an optimal support based on available data. This is in part possible because ACRI is not substantially sensitive to the variation of the support. We are also investigating other rule selection strategies since selecting the right rules has a paramount effect on the precision of a classifier. Moreover, pruning the large set of classification rules can improve the accuracy and speed of the classifier.

References

1. Antonie, M.-L., Zaïane, O. R. Text document categorization by term association. *IEEE International Conference on Data Mining*, (2002) 19–26
2. Han, J., Pei J., Y., Yin: Mining Frequent Patterns Without Candidate Generation, *ACM Intl' Conf. on Management of Data*, (2000)
3. Li, W., Han, J., Pei J.: CMAR: Accurate and efficient classification based on multiple class-association rules. *IEEE International Conference on Data Mining*, (2001)
4. Liu, B., Hsu, H., Ma, Y.: Integrating classification and association rule mining. In *4th Intl. Conf. on Knowledge Discovery and Data Mining*, (1998) 80–86.
5. Ong K.-L., Ng, W.-K., Lim, E.-P., Mining Multi-Level Rules with Recurrent Items Using FP'-Tree, ICICS (2001)
6. Reuters-21578 Top 10 Topics Collection, http://www.jihe.net/datasets.htm
7. UCI repository, http://www.ics.uci.edu/~mlearn/MLRepository.html

8. Wang, W., Yang, J., Yu, P., WAR: Weighted Association Rules for Item Intensities, Knowledge and Information Systems, vol. 6, (2204) 203-229
9. Zaïane, O. R., and Antonie, M.-L. Classifying text documents by associating terms with text categories. In *Proc. of the Thirteenth Australasian Database Conference (ADC'02)*, (2002) 215–222.
10. Zaïane, O. R., Han, J., Zhu, H. Mining recurrent items in multimedia with progressive resolution refinement. In *Int. Conf. on Data Engineering*, (2000) 461–470