Rule Pruning Methods in Associative Classification Text Mining

Fadi Thabtah
Wael Hadi
MIS Department, Philadelphia University, Amman, Jordan
{ffayez, whadi}@philadelphia.edu.jo

Hussein Abdel-jaber
CIS Department, WISE University, Amman, Jordan
habdelja@wise.edu.jo

Mofleh Aldiabat
Computer Science Department, Al-Albayt University
maldiabat@al-bayt.edu.jo

ABSTRACT: *Associative classification is the integration of classification and association rule discovery in data mining. This approach often derives higher quality classifiers with reference to classification accuracy than traditional classification approaches such as decision trees and rule induction. In this paper, the problem of rule pruning in associative text categorisation is investigated. We propose five rule pruning methods within an existing associative classification algorithm called MCAR. Experimental results against large text collection (Reuters-21578) using the developed pruning methods as well as other known existing methods (Database coverage, lazy pruning) are conducted. The bases of the experiments are the classification accuracy and the number of generated rules. The results derived show that the proposed rule pruning methods derive higher quality and more scalable classifiers than those produced by lazy and database coverage pruning approaches. In addition, the numbers of rules generated by the developed pruning methods are usually less than those of lazy and database coverage. This makes the associative classifiers generated when using the proposed methods more maintainable by end users.*

Keywords: Artificial Intelligence, Learning Algorithms, Text Mining, Rule pruning

Received: 12 September 2009, Revised 09 October 2009, Accepted 11 November 2009

1. Introduction

Associative Classification (AC) also known as classification based on association rule, is an integration of two important data mining tasks, association rule discovery and classification. Association rule and classification are analogous tasks, with the exception that classification's main aim is the prediction of class labels, while association rule describes correlations among items in a database. Several studies (Liu et al., 1998; Li et al., 2001; Yin and Han, 2003; Thabtah, et al., 2004; Thabtah, et al., 2005; Li et al., 2008; Niu et al., 2009) provide evidence that AC approach is able to derive more accurate classifiers than traditional classification techniques such as decision trees (Quinlan, 1993), and rule induction (Cohen, 1995). However, this approach suffers from the exponential growth of rules since it employs association rule during the learning step where all the correlations among the items and the class are discovered in a form of if-then rules. Thus, pruning duplicate and misleading rules becomes essential.

The AC mining approach works as follow. First, all rules satisfying some user-specified parameters (minimum support (minsupp), minimum confidence (minconf)), are generated using an association rule mining algorithm. The number of rules

generated might be of the order of several thousands, and many of them are either redundant, i.e. never used, or noisy, i.e. not discriminative among the classes. Second, the redundant and noisy rules are removed using pruning procedure(s). A significant number of rules can be discarded at this stage and the rules left are the interesting ones that form a model (classifier) used later to classify test cases. Each classifier must have a default rule which is applied when no other rule in the classifier is used.

AC algorithms normally derive large sets of rules (Topor and Shen, 2001; Li et al., 2001; Kundu et al., 2008; Li et al. , 2008) since (1) classification data sets are typically highly correlated and (2) association rule mining is utilised in the rule discovery step. As a result, there have been many attempts to reduce the size of classifiers produced by AC approaches, mainly focused on preventing rules that are either redundant or misleading from taking any role in the prediction process of test data cases. The removal of such rules can make the classification process more effective and accurate since it is not recommended to have a large number of rules when we classify test cases. There is a great chance of having more than one rule contradicting each other in the answer class, therefore one wants to have a small number of highly predictive rules.

The aim of this paper is to investigate the rule pruning step in AC mining in order to reduce the size of the resulting classifiers. Particularly, we develop five rule pruning methods and implement them within a known AC algorithm called MCAR (Thabtah et al., 2005). These methods are tested against large text categorisation collection called Reuters-21578 Mod Split (Lewis, 1998), and compared with other existing pruning methods in AC such as the database coverage, and greedy pruning. The bases of the comparison are the number of rules produced and the classification accuracy. To the best of the authors' knowledge, rule pruning AC on text categorization has not been explored yet in classification data mining research.

This paper is structured as follows: AC approach and known rule pruning methods are discussed in Section 2. In Section 3, the proposed rule pruning techniques are presented. Further, the results that show the impact of pruning on the size of the classifiers and the prediction accuracy are demonstrated in Section 4. Finally, conclusions are given in Section 5.

2. Associative Classification

2.1 Associative Classification Problem

AC is a special case of association rule in which only the class attribute is considered in the rule's right-hand-side (consequent) (Liu et al., 1998), for example in a rule such as $X \rightarrow Y$, Y must be a class attribute. We follow (Thabtah, et al., 2004) for the definition of the AC problem where a training data set T has m distinct attributes $A_1, A_2, \ldots, A_m$ and C is a list of class labels. The number of rows (cases) in T is denoted $|T|$.

Definition 1: A row or a training case in T can be described as a combination of attributes A_i and values a_{ij}, plus a class denoted by c_j.

Definition 2: An attribute value can be described as a term name A_i and a value a_i, denoted $<(A_i, a_i)>$.

Definition 3: An *AttributeValueSet* can be described as a set of disjoint attribute values contained in a training case, denoted $< (A_{i1}, a_{i1}), \ldots, (A_{ik}, a_{ik})>$.

Definition 4: A *ruleitem r* is of the form $<$ *AttributeValueSet*, $c>$, where $c \quad C$ is the class.

Definition 5: The actual occurrence (*actoccr*) of a *ruleitem r* in T is the number of rows (cases) in T that match the *AttributeValueSet* of *r*.

Definition 6: The support count (*suppcount*) of *ruleitem r* is the number of rows in T that match *r's AttributeValueSet*, and belong to the class *c* of *r*.

Definition 7: The occurrence of an *AttributeValueSet i* (*occatt*) in T is the number of rows in T that match *i*.

Definition 8: A *ruleitem r* passes the *minsupp* threshold if $(suppcount(r)/|T|) \geq minsupp$,

Definition 9: A *ruleitem r* passes the *minconf* threshold if $(suppcount(r)/actoccr(r)) \geq minconf$.

Definition 10: Any *ruleitem r* that passes the *minsupp* threshold is said to be a *frequent ruleitem*.

Definition 11: A class association rule (CAR) is represented in the form: $(A_{i1}, a_{i1}) \wedge \ldots \wedge (A_{ik}, a_{ik}) \rightarrow c$, where the antecedent (rule body/LHS) of the rule is an *AttributeValueSet* and the consequent (RHS) is a class.

Definition 12: We say that a CAR *R* partially matches a training case *t* during building the classifier if R contains at least an item in its antecedent that exists in *t*.

Definition 13: We say that a CAR *R* is fully matches a test training *t* during building the classifier if all *R* items are in *t*.

A classifier is a mapping form $H: A \rightarrow Y$, where *A* is the set of *AttributeValueSet* and *Y* is the set of class labels. The main task of AC is to construct a set of rules (model) that is able to predict the classes of previously unseen data, known as the test data set, as accurately as possible. In other words, the goal is to find a classifier h H that maximises the probability that h (a) = y for each test case.

2.2 Current Pruning Methods in AC

Several pruning methods have been used effectively to cut down the number of rules in AC, some of which have been adopted from decision trees, like pessimistic estimation (Quinlan, 1987), others from statistics such as Chi-square testing (÷2) (Snedecor and Cochran, 1989). These pruning methods are utilised during the construction of the classifier. Throughout this section, we summarise pruning methods that are solely developed within AC mining from (Thabtah, 2006).

The database coverage is a post pruning technique used in AC (Liu et al., 1998), which is usually invoked after rules have been created. If at least one case among all the cases in training data set is fully matched by the rule, the rule is inserted into the classifier and all cases covered are removed from the training data set. The rule insertion stops when either all of the rules are used or no cases are left in the training data set. The majority class among all cases left in the training is selected as default class. The database coverage method was used first by CBA (Liu et al., 1998) and then latterly by other associative algorithms, including CBA (2) (Liu et al., 2003), CMAR (Li et al., 2001), ARC-BC (Antonie and Zaïane, 2002), CAAR (Xu et al., 2004), ACN (Kundu et al., 2008), and ACCF (Li et al. , 2008).

A pruning method that discards specific rules with less confidence values than general rules called redundant rule pruning, has been proposed in (Li et al., 2001). This method works as follows: Once the rule generation process is finished and rules are ranked, an evaluation step is performed to prune all rules such as $I' \rightarrow c$ from the set of generated rules, where there is some general rule $I \rightarrow c$ of a higher rank and $I \subseteq I'$. Algorithms, including (Li et al., 2001), have used redundant rule pruning. They perform such pruning immediately after a rule is inserted into the compact data structure, the CR-tree. When a rule is added to the CR-tree, a query is issued to check if the inserted rule can be pruned or some other already inserted rules in the tree can be removed.

Some AC techniques (Baralis, et al., 2004; Baralis et al. 2008) argue that pruning classification rules should be limited to only "negative" rules (those that lead to incorrect classification). In addition, they claim that database coverage pruning often discards some useful knowledge, as the ideal support threshold is not known in advance. Due to this, these algorithms have used a late database coverage-like approach, called lazy pruning, which keeps the rules that incorrectly classify training cases and keeps all others. The main difference between lazy pruning and database coverage is that rules which don't cover a training case and held in the memory by the lazy pruning are completely removed by the database coverage during rule discovery step.

Experimental tests reported in (Baralis, et al., 2004; Baralis, et al., 2008) using 26 different data sets from (Merz and Murphy, 1996) showed that methods that employ lazy pruning such as L3 and L3G algorithms may improve classification accuracy on average by 1.63% over other techniques that use database coverage pruning (Liu et al., 1998). However, lazy pruning may lead to very large classifiers, which makes it difficult for a human to understand or be able to interpret. In addition, the experimental tests indicate that lazy pruning algorithms consume more memory than other AC techniques and, more importantly, they may fail if the support threshold is set to a very low value due to the very large number of potential rules.

3. The Proposed Rule Pruning Methods

The main goals for proposing the new pruning methods in this paper are first to minimise the number of rules generated, and second to measure the impact of pruning on the derived classifiers with respect to classification accuracy. Specifically, a combination of rule evaluation criteria such as the correctness of the rule class when compared to the training cases that are

covered by such rule, and partial matching (Definition 12- Section 2), are considered. The main reason for utilising partial matching during construction of the classifier step is to give a chance for more rules to contribute to the decision of assigning the correct class to the test case during the classification step, which consequently may improve the classification accuracy since there could be multiple rules used to make the prediction decision. We show the impact of using partial matching on the classification accuracy of different data sets in Section 4. In this regard, some of the research questions in which this paper attempt to answer are:

- During building the classifier, if a rule is considered significant when its body fully matches the training data in the classification, does this impact the accuracy? or
- Does the accuracy of the classifier get improved if partial matching is considered? or
- Does utilising a hybrid pruning approach (partial matching + full matching) produces good results with respect to number of rules and accuracy.

The widely used database coverage pruning inputs the rule into the classifier during rule evaluation if the rule body is identical to at least one training case, and the class of such rule is similar to that of the training case. Unlike the database converge, we would like to explore more criteria during building the classifier. For instance the proposed HCP method takes into account partial matching between the candidate rule body and the training cases, whereas, the HP methods adds upon the HCP and considers a combination of full and partial matching. In other words, HCP checks first whether there is an existed rule that its body fully matches the training case, if this condition is false, it then looks for the highest ranked rule which partially matches the training case. This indeed reduces the use of the default class which usually increases error during the classification of test cases step. Further, we would like to investigate the relationship between the training case class and that of the rule during building the classifier, and answer the question "is it necessary that the training case class must equal the candidate rule class in order for that rule to be inserted into the classifier?".

Rule-id	Rule	conf	sup	Class count	Rank
1	real=>sport	0.99	0.286	2	2
2	madrid=>sport	0.67	0.285	2	10
3	stock=>acq	0.75	0.428	3	8
4	share=>acq	0.67	0.285	3	9
5	group=>acq	1.00	0.285	3	1
6	amman=>general	0.99	0.285	2	3
7	madrid & real=>sport	0.99	0.285	2	6
8	share & stock=>acq	0.67	0.285	3	11
9	group & stock=>acq	0.99	0.285	3	4
10	group & share=>acq	0.99	0.285	3	5
11	group&share&stock=>acq	0.99	0.285	3	7

Table 1. Example of a rule-based model (potential rules)

In this section, we discuss the proposed rule pruning methods along with an example to illustrate each of them. Assume that the eleven rules shown in Table 1 are produced by an AC algorithm called MCAR (Thabtah et al., 2005) using minsupp and minconf of 20% and 40%, respectively, from the training data set shown in Table 2. Before pruning kicks in the rules must be ranked according to confidence, support, and rule size values in descending manner. According to Table 1, Rule-5 is the highest ranked rule since its confidence is the largest among the rest of the rules. Though Rule-1, Rule-6, Rule-7, Rule-9, Rule-

id	Document	Class	Random Rank
1	real,madrid,stock,loss,share	sport	1
2	real,madrid	sport	5
3	stock,share,group	acq	2
4	stock,share,buy,group	acq	3
5	stock	acq	4
6	iraq,amman,jordan	general	7
7	amman,madrid,real	general	6

Table 2. Example of a training data

10, and Rule-11 have the same confidence; Rule-1 is ranked higher due to its larger support. Still Rule-6, Rule-7, Rule- 9, Rule-10, and Rule-11 have the same confidence and support values; but Rule-6 is ranked higher due to the fact that it has less number of values in its antecedent, and so forth.

Generally, our rule pruning methods can be categorised into two main groups, one that makes the pruning based on one criteria (Single criteria Pruning) and one that makes the pruning based on two criteria (Hybrid Pruning). In the following subsections, these methods are presented.

3.1 Single Criteria Pruning

We discuss here the three new rule pruning methods, which consider a single condition for pruning during building the classifier. The first method is called (HCP), which signifies the rule if the rule antecedent (body) is partially matching any of the training document keywords and the rule class is identical to that of the training document. The second method is called (HP) that considers the rule significant if its antecedent partially matches any of the training document keywords without checking the class labels. Lastly, the FMP considers the rule significant if its antecedent fully matches any of the training document keywords regardless the class label.

3.1.1 High Precedence Classify Correctly Pruning Method (HCP)

Input: Given a set of generated rules *R*, and training data set *T*
Output: classifier (*Cl*)

```
1    R' = sort(R);
2       For each rule r_i in R' Do
3     Find all applicable training cases in T that match r_i's condition
4    If r_i correctly classifies a training case in T
5       Insert the rule at the end of Cl
6       Remove all training cases in T covered by r_i
7    end if
8    If r_i cannot correctly cover any training case in T
9     Remove r_i from R
10   end if
11      end for
```

Figure 1. HCP pruning method

Not all of the rules derived during the learning phase can be used in the prediction step, and therefore some rules will be discarded. This pruning method should remove many redundant rules from the set of produced rules as shown in Figure 1. To describe this pruning method, let's use the example of Section 3 in which the first ranked rule Rule-5: group=>acq is

applied on the training data shown in Table 2, we find that this rule partially matches documents 3 and 4, and the class of this rule is identical to that of these documents. So, we insert this rule into the classifier, and we remove documents 3 and 4 from the training data set. We proceed to the second ranked rule i.e. Rule-1: real=>sport, and we check the applicability of this rule with the remaining training documents. We find that Rule-1 partially matches documents 1, 2, and 7, but the class of document 7 is not consistent with that of Rule-1. So, we insert Rule-1 since it covers correctly at least one document at the end of the classifier, and we remove documents 1, and 2. The third ranked rule Rule-6: amman=>general covers correctly two documents 6 and 7, so we insert it into the classifier, and we remove the documents 6 and 7 from the training documents set, and we repeat the same steps for the rest of the rules until the training set becomes empty. Finally we only consider the rules within the classifier for the prediction phase. In this example, the classifier contains just four significant rules, and the remaining candidate rules will be deleted.

The main difference between this pruning method and the database coverage (Liu et al., 1998) is that a rule gets inserted into the classifier if it partially covers at least one training case. On the other hand, in the database coverage of CBA algorithm, a rule must fully match the training case antecedent in order to be inserted.

3.1.2 High Precedence Pruning Method (HP)

In this pruning method (Figure 2), a rule is considered if its antecedent partially matches the training documents words regardless the class label. To describe this pruning method, let's use the example of Section 3 in which the first ranked rule Rule-5: group=>acq is evaluated on the training data shown in Table 2, we find that this rule partially matches documents 3 and 4. So, we insert this rule into the classifier, and we remove documents 3 and 4 from the training data set. We proceed to the second ranked rule i.e. Rule-1: real=>sport, and we check its applicability with the remaining training documents. We find that Rule-1 partially matches documents 1, 2, and 7. So, we insert Rule-1 at the end of the classifier, and we remove documents 1, 2 and 7. It should be noted that unlike the HCP method which checks the correctness of class label between the candidate rule and the training documents, the HP method removed document 7 even though its class is different than the candidate rule Rule-1. Further, The third ranked rule Rule-6: amman=>general covers one document 6 so we insert it into the classifier, and we remove the document 6 from the training documents set, and we repeat the same steps for the rest of the rules until the training documents set becomes empty.

```
Input: Given a set of generated rules R, and training data set T
Output: classifier (Cl)

1  R' = sort(R);
2  For each rule r_i in R' Do
3        Find all applicable training cases in T that match r_i's condition
4          Insert the rule at the end of Cl
5          Remove all training cases in T covered by r_i
6       If r_i cannot correctly cover any training case in T
7          Remove r_i from R
8     end if
9 end for
```

Figure 2. HP pruning method

The difference between HP and HCP methods is that in the HP, a rule gets inserted into the classifier if it partially covers at least one training case regardless if it classifies that case correctly or not. On the other hand, in the HCP, a rule must classify a training case correctly in order to be considered in the classifier.

3.1.3 Full Match Pruning Method (FMP)

In this pruning method (Figure 3), a rule is considered if its antecedent fully matches the training documents words. To describe this method, we apply it on the above example, the first ranked rule is Rule-5: group=>acq, is applied on the training documents, only documents 3 and 4 are considered, therefore we insert Rule-5 into the classifier, and we remove documents 3, and 4 from the training data set. We proceed with Rule-1: real=>sport, and apply it on the remaining training documents, we find that Rule-1 is associated with documents 1, 2, and 7, so we insert Rule-1 into the classifier, and documents 1, 2 and 7 will be removed from the training data set. The same steps are repeated for the rest of the candidate rules.

Input: Given a set of generated rules *R*, and training data set *T*
Output: classifier (*Cl*)

1 *R'* = sort(*R*);
2 For each rule r_i in *R'* Do
3 Find all applicable training cases in *T* that match r_i's conditions
4 Insert the rule at the end of Cl
5 Remove all training cases in *T* covered by r_i
6 If r_i cannot correctly cover any training case in *T*
7 Remove r_i from *R*
8 end if
9 end for

Figure 3. FMP pruning method

3.2 Hybrid Pruning

In this section two rule pruning methods called FHP and FCHCP that combine the criteria of the single pruning methods are proposed.

3.2.1 Full Match then High Pruning Method (FHP)

In this pruning method (Figure 4), a rule can be part of the classifier if it fully matches at least one training document, in a case where the selected rule does not match any training document then the highest sorted rule that partially matches the training document is considered. To describe this method, we utilise it on the example described in Section 3, the first ranked rule is Rule-5: group=>acq, is considered on the training documents we find that this rule fully matches documents 3 and 4, so we insert it into the classifier, and we remove documents 3 and 4 from the training set. We proceed with the second ranked rule, Rule-1: real=>sport, and apply it on the remaining training documents, we find that the rule fully matches documents 1, 2 and 7, so we insert this rule into the classifier, and we remove documents 1, 2 and 7 from the training set. The third ranked rule Rule 6: amman=>general matches document 6, so we insert this rule into the classifier, and we remove document 6 from the remaining training documents set. The fourth ranked rule is Rule-9: group & stock=>acq, Rule-9 does not fully match document 5 because the term "group" in the rule body does not exist in document 5, so the "High Precedence" pruning method kicks in. We find that Rule-9 partially matches document 5, so we insert the rule into the classifier, and we remove document 5 from the training documents set. Now the training documents set is empty so we stop.

Please note that unlike existing methods such as the database coverage which takes into account the default class if no rules are applicable to the training case, the FHP method instead considers rules that partially applicable to the training case, which eventually minimises randomisation in making decision, and therefore may improve predictive power..

```
1  Input: Given a set of generated rules R, and the training data set T
2  Output: classifier (Cl)
3  R' = sort(R);
4  for each rule r ∈ R' in sequence do
5  if r fully matches  at least a single training case
6          insert the rule at the end of Cl
7          remove all training cases in T covered by r_i
8          else if r partially match at least a single training case
9          insert the rule at the end of Cl
10         remove all training cases in T covered by r_i
11 end if
12 If r_i cannot match any training case in T
13         Remove r_i from R
14 end if
15 end for
```

Figure 4. FHP Pruning Method

3.2.2 Full Match that Classify Correctly then High Classify Correctly (FCHCP)

In this pruning method (Figure 5), a rule can be part of the classifier if it fully matches at least one training document, and if it classifies these documents correctly. If the rule does not match any training document then we apply partial matching method. To describe this method, we pertain it on the example described in Section 3, the first ranked rule Rule-5: group=>acq is applied on the training documents, only documents 3 and 4 are consistent with Rule-5, and the class labels of these documents are identical to Rule-5 class, therefore we insert Rule-5 into the classifier, and we remove documents 3, and 4 from the training data set. We proceed with Rule-1: real=>sport, and consider it on the remaining training documents, we find that Rule-1 is associated with documents 1, 2, and 7, but the class of document 7 "general" is not consistent with that of Rule-1, so we insert Rule-1 into the classifier, and only documents 1 and 2 are removed from the training data set. The third ranked rule Rule-6: Amman=>general covers two documents 6 and 7 and classifies these documents correctly, so we insert the rule into the classifier, and we delete documents 6 and 7 from the training set. The fourth ranked rule Rule-9: group & stock=>acq does not fully match document 5, so we use the next technique on this rule (HCP), and we find out that this rule partially matches document 5, so we insert the rule into the classifier, and we discard document 5 from training documents. Now the training set becomes empty, so we stop. Finally, FCHCP returns only four rules in the classifier.

It should be noted that this method is different than the FHP in that it looks at the class labels of both the candidate rule and the training case. If they are identical then a rule is considered significant and inserted into the classifier, otherwise it invokes the HCP method which considers partial matching.

4. Data and Results

4.1 Text Data Collection

The benchmark used in the experiments is the Reuters-21578 (Lewis, 1998). The Reuters-21578 is the most widely used text data set in the text categorisation research. We used the ModApte version of Reuters-21578, which contains a corpus of 9,174 documents consisting of 6,603 training cases and 2,571 testing cases, respectively. We tested our pruning procedures within the MCAR algorithm (Thabtah, et al., 2005) on the seven most populated categories with the largest number of documents assigned to them in the training data set. The experiments are conducted on 2.8 Pentium IV machine with 1GB RAM, and the proposed methods and MCAR are implemented using VB.Net programming language with a minsupp and minconf of 2%, and 40%, respectively. The minsupp has been set to 2% since more extensive experiments reported in (Liu, et al., 1998; Li, et al., 2001, Thabtah, et al., 2004, Thabtah, et al., 2005) suggested that it is one

Input: Given a set of generated rules R, and training data set T
Output: classifier (Cl)

1 R' = sort(R);
2 For each rule r_i in R' Do
3 if r fully match at least a single training case and (the r classify a training case correctly)
4 Insert the rule at the end of Cl
5 Remove all training cases in T covered by r_i
6 else if r partially match at least a single training case and (the r classify a training case correctly)
7 Insert the rule at the end of Cl
8 Remove all training cases in T covered by r_i
9 end if
10 If r_i cannot correctly cover any training case in T
11 Remove r_i from R
12 end if
13 end for

Figure 5. FCHCP pruning method

of the rates that achieve a good balance between the accuracy and the size of the classifiers. The confidence threshold, on the other hand, has a smaller impact on the behaviour of any AC method and it has been set to 40%.

Table 3 represents the number of documents for each category in the Reuters-21578 data set. On these documents we performed stop word elimination but not stemming, and we select the top 1000 features using Chi Square (Snedecor and Cochran, 1989). In the following section, we show the results of the proposed pruning procedures, the database coverage method, lazy method, and the case of no pruning.

Category Name	Training set	Testing Set
Acq	1650	719
Crude	389	189
Earn	2877	1087
Grain	433	149
Interest	347	131
Money-FX	538	179
Trade	369	117
Total	6603	2571

Table 3. Number of documents per category (Reuters-21578)

4.2 Experimental Results

In the association rule discovery, a document can be used to generate many rules; therefore, there are tremendous numbers of potential rules. Without adding constraints on the rule discovery and generation phases or imposing appropriate pruning, the very large numbers of rules, often in the order of tens of thousands, make humans unable to understand or maintain the classifier. Pruning noisy and redundant rules in classifiers becomes an important task. We performed extensive experiments on the seven most populated categories of the Reuters-21578 test collection to compare our proposed methods, the database coverage (Liu et al., 1998), lazy pruning (Baralis, et al., 2004), and the case of no pruning with reference to the number of rules generated and the predictive accuracy.

Class label	no Pruning	HP	FMP	HCP	FCHCP	FHP	Database Coverage	Lazy
Acq	80	32	28	28	26	27	27	40
Crude	8	5	5	5	4	4	4	6
Earn	172	29	18	15	16	17	17	55
Grain	5	5	5	5	5	5	5	5
Interest	4	3	2	1	1	2	2	4
Money-FX	23	20	12	12	11	12	12	15
Trade	9	8	6	4	6	6	6	8
Total	**301**	**102**	**76**	**70**	**69**	**73**	**73**	**133**

Table 4. MCAR number of rules produced when different pruning approaches are used against Reuter data set

large number of rules by HP is due to storing rules that cover at least one training document regardless if the rules classify training document correctly or not.

Moreover, the number of rules generated without pruning method on "Acq" class collection is 80, whereas the number of rules derived using HCP pruning procedure is 28. The additional fifty two rules produced in the case of no pruning may increase the prediction time. It is obvious from the numbers shown in Table 4 that algorithms, which use lazy pruning approach, often generate many more rules than those that employ other approaches. In particular, for all classification data sets we considered, MCAR using lazy pruning method produced more rules than the other considered approaches.

One of the principle reasons for generating large number of rules by lazy pruning algorithms is due to storing rules that do even cover a single training data case in the classifier. For example, while constructing the classifier by the MCAR algorithm using lazy pruning method, every rule is evaluated to validate whether or not it covers a training data case. If a rule covers correctly a training case, it then will be added into the classifier, otherwise it will be removed. However, rules, which have been never tested, are also added as spare rules into the classifier. These spare rules are may raise many problems such as user understandability and maintenance. Furthermore, the classification time may increase since the classification system must iterate through the rule set, and consequently these problems limit the use of lazy associative algorithms.

Unlike lazy pruning, the database coverage and the proposed methods eliminate the spare rules and that explains its moderate size classifiers. Specifically, MCAR using our proposed methods and MCAR using database coverage algorithms generate reasonable size classifiers if compared with MCAR using lazy pruning method. This enables domain users to benefit from.

Table 5 depicts the classification accuracy (%) derived by MCAR algorithm using the different rule pruning methods on the seven most populated categories of Reuters-21578. We also reported the accuracy derived without pruning in column 2. The accuracy numbers have been generated using a minsupp of 2% and a minconf of 40%. Table 5 indicates that when the HCP pruning is employed, the classification accuracies derived are better than those of HP, FMP, FCHCP, FHP, database coverage and the lazy pruning methods. Particularly, the won-tied-loss records of HCP records against no pruning, HP, FMP, database coverage and lazy pruning are 7-0-0, 1-5-1, 6-0-1, 6-0-1, 7-0-0, 7-0-0 and 7-0-0, respectively. In addition, HP and FMP methods outperformed the database converge, no pruning and lazy pruning approaches with respect to prediction accuracy. In fact, we achieved on average +19.6%, +18.5%, +16.9% higher prediction rates within MCAR algorithm than no pruning, the database coverage and lazy pruning, respectively on the Reuter text collection.

5. Conclusions

The number of rules that can be generated in the rule discovery step in associative classification could be very large. There are two issues that must be addressed in this case. One of them is that such a huge numbers of rules could contain noisy information which would mislead the classification process. Another is that a huge set of rules would make the classification time longer. This could be an important problem in applications where fast responses are required. In this paper, we proposed different rule pruning methods within associative classification mining. We conducted experiments on seven categories

Class label	NoPruning	HP	FMP	FCHCP	FHP	HCP	DatabaseCoverage	Lazy pruning
Acq	80.6	99.9	98.5	80.6	80.6	99.9	82.6	82.4
Crude	76.2	96.3	94.7	76.2	76.2	96.3	80.4	88.4
Earn	99.6	99.7	99.3	99.6	99.6	99.7	99.6	99.6
Grain	90.6	96	94	90.6	90.6	96	91.8	94.8
Interest	3.1	3.8	3.1	3.1	3.1	69.5	3.1	3.1
Money-FX	49.7	88.3	49.7	49.7	49.7	73.7	49.7	49.7
Trade	93.2	94.9	96.6	93.2	93.2	94.9	93.2	93.4
Average	**70.4**	**82.7**	**76.6**	**0.704**	**0.704**	**90.0**	**71.5**	**73.1**

Table 5. Accuracy per class of the Reuter data derived by the MCAR algorithms using the different rule pruning methods

selected from the Reuters-21578 text collection using the developed pruning procedures and existing pruning methods in associative classification. The bases of the comparison are the number of produced rules and the accuracy, and we implemented all pruning methods within a known associative algorithm called MCAR. The experimental results revealed that the HCP method outperformed all other pruning techniques with reference to predictive accuracy and number of rules generated. Particularly, on average the combined proposed methods achieved on average +6.52%, +4.92%, +7.62% higher prediction rates within MCAR algorithm than database coverage, lazy pruning, and no pruning, respectively. In near future, we intend to expand our research to include other rule pruning heuristics in the areas of decision trees, statistics, and rule induction.

References

[1] Antonie M. and Zaiane O. (2002). Text Document Categorization by Term Association, Proceedings of the IEEE International Conference on Data Mining (ICDM '2002). p.19-26, Maebashi City, Japan.

[2] Baralis E., Chiusano S., Garza P. (2008). A Lazy Approach to Associative Classification, *IEEE Trans. Knowl. Data Eng*. 20 (2) 156-171.

[3] Baralis E., Chiusan, S., Garza P. (2004). On support thresholds in associative classification, *In*: Proceedings of the 2004 ACM Symposium on Applied Computing, p. 553-558. Nicosia, Cyprus.

[4] Cohen W. (1995). Fast effective rule induction, *In*: Proceedings of the 12th International Conference on Machine Learning, p. 115-123. CA, USA.

[5] Kundu, G., Islam M., Munir, S., Bari M. (2008). ACN: An Associative Classifier with Negative Rules, Computational Science and Engineering, p. 369-375, *In*: 11th IEEE International Conference on Computational Science and Engineering.

[6] Lewis D. (1998). Reuters 21578 text categorisation test collection.http://www.daviddlewis.com/resources/testcollections/reuters21578.

[7] Li B., Li H., Wu, M., Li P. (2008). Multi-label Classification based on Association Rules with Application to Scene Classification, icycs, p.36-41, *In:* 2008 The 9th International Conference for Young Computer Scientists.

[8] Li W., Han J., Pei J. (2001). CMAR: Accurate and efficient classification based on multiple-class association rule, *In:* Proceedings of the ICDM'01, p. 369-376. San Jose, CA.

[9] Liu B., Hsu, W., Ma Y. (1998). Integrating classification and association rule mining, *In*: Proceedings of the KDD, p. 80-86. New York, NY.

[10] Liu B., Ma Y., Wong, C-K., Yu. P. (2003). Scoring the data using association rules, *Applied Intelligence*, 18. 119-135.

[11] Merz, C., Murphy P. (1996). UCI repository of machine learning databases. Irvine, CA, University of California, Department of Information and Computer Science.

[12] Niu Q., Xia S., Zhang L. (2009). Association Classification Based on Compactness of Rules, wkdd, p.245-247, Second International Workshop on Knowledge Discovery and Data Mining.

[13] Quinlan, J. (1993). C4.5: Programs for machine learning. San Mateo, CA: Morgan Kaufmann.

[14] Quinlan, J. (1987). Simplifying decision trees, *International Journal of Man-Machine Studies*, 27. 221-248.

[15] Snedecor, W., Cochran W. (1989). Statistical Methods, Eighth Edition, Iowa State University Press.

[16] Thabtah, F., Cowling, P., Peng, Y. (2005) MCAR: Multi-class classification based on association rule approach, *In*: Proceeding of the 3rd IEEE International Conference on Computer Systems and Applications, p. 1-7.Cairo, Egypt.

[17] Thabtah, F., Cowling, P., Peng, Y. (2004). MMAC: A new multi-class, multi-label associative classification approach, *In*: Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM '04), p. 217-224. Brighton, UK. (Nominated for the Best paper award).

[18] Topor, R., Shen H. (2001). Construct robust rule sets for classification, *In:* Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 564-569. Edmonton, Alberta, Canada.

[19] Xu X., Han G., Min H. (2004). A novel algorithm for associative classification of images blocks, *In*: Proceedings of the fourth IEEE International Conference on Computer and Information Technology, p. 46-51. Lian, Shiguo, China.

[20] Yin, X., Han J. (2003). CPAR: Classification based on predictive association rule, *In*: Proceedings of the SDM. p. 369-376. San Francisco, CA.