Text Document Categorization by Term Association

Maria-Luiza Antonie
University of Alberta, Canada
luiza@cs.ualberta.ca

Osmar R. Zaïane
University of Alberta, Canada
zaiane@cs.ualberta.ca

Abstract

A good text classifier is a classifier that efficiently categorizes large sets of text documents in a reasonable time frame and with an acceptable accuracy, and that provides classification rules that are human readable for possible fine-tuning. If the training of the classifier is also quick, this could become in some application domains a good asset for the classifier. Many techniques and algorithms for automatic text categorization have been devised. According to published literature, some are more accurate than others, and some provide more interpretable classification models than others. However, none can combine all the beneficial properties enumerated above. In this paper, we present a novel approach for automatic text categorization that borrows from market basket analysis techniques using association rule mining in the data-mining field. We focus on two major problems: (1) finding the best term association rules in a textual database by generating and pruning; and (2) using the rules to build a text classifier. Our text categorization method proves to be efficient and effective, and experiments on well-known collections show that the classifier performs well. In addition, training as well as classification are both fast and the generated rules are human readable.

1 Introduction

Automatic text categorization has always been an important application and research topic since the inception of digital documents. Today, text categorization is a necessity due to the very large amount of text documents that we have to deal with daily. A text categorization system can be used in indexing documents to assist information retrieval tasks as well as in classifying e-mails, memos or web pages in a yahoo-like manner. Needless to say, automatic text categorization is essential.

The text classification task can be defined as assigning category labels to new documents based on the knowledge gained in a classification system at the training stage. In the training phase we are given a set of documents with class labels attached, and a classification system is built using a learning method. Classification is an important task in both data mining and machine learning communities, however, most of the learning approaches in text categorization are coming from machine learning research.

Recent studies in the data mining community proposed new methods for classification employing association rule mining [12, 13]. These associative classifiers have proven to be powerful and achieve high accuracy. However, they were only implemented and tested on small numerical datasets from the UCI archives [19].

In this work we present a new classification method for text that takes advantage of association rule mining in the learning phase and makes the following contributions: First, *a new technique for text categorization that makes no assumption of term independence is proposed. This method proves to perform as well as other methods in the literature*. Second, *it is fast during both training and categorization phases*. Third, *the classifier that is built using our approach can be read, understood and modified by humans.* Experiments show that the effectiveness of the classifier can be improved by manually fine tuning the classification rules generated during the training phase. The resulting classifier is able to perform both single-class classification, by which each document is assigned a unique class label, and multiple-class classification, by which a document could be classified in many classes simultaneously. Our experiments are performed on text databases, however, this doesn't limit the use of our classifier to text documents. It can be applied in addition to images or any other database that can be modelled as a transactional database [2].

The remainder of the paper is organized as follows: Section 2 gives an overview of related work in automatic text categorization and in association rule mining. In Section 3 we introduce our new text categorization approach. Experimental results are described in Section 4 along with the performance of our system compared to known systems. We summarize our research and discuss some future work directions in Section 5.

2 Related work

Many text classifiers have been proposed in the literature using machine learning techniques, probabilistic models, etc. They often differ in the approach adopted: decision trees, naïve-Bayes, rule induction, neural networks, nearest neighbors, and lately, support vector machines. Although many approaches have been proposed, automated text categorization is still a major area of research primarily because the effectiveness of current automated text classifiers is not faultless and still needs improvement.

A classifier is built by applying a learning method to a training set of objects. This model is further used to predict the labels to new incoming objects. With all the effort in this domain there is still place for improvement and a great deal of attention is paid to developing highly accurate classifiers.

The use of association rule mining for building classification models is very new. Recent studies have proposed the use of association rules in building effective classifiers for numerical data. These classification systems discover the strongest association rules in the database and use them to build a categorizer.

In the following subsections a more detailed overview of the related work is presented from both domains that merge in our research: text categorization and association rule mining.

2.1 Text categorization

The automatic text classification problem can be defined as building a classification model to assign one or more predefined classes to new documents. Text categorization research has a long history, starting in the early 1960s. Nowadays, with all the textual information on the Web or in companies' intranets, text categorization has revived and there is more demand for effective and efficient classification models.

Most of the research in text categorization comes from the machine learning and information retrieval communities. Rocchio's algorithm [8] is the classical method in information retrieval, being used in routing and filtering documents. Researchers tackled the text categorization problem in many ways. Classifiers based on probabilistic models have been proposed starting with the first presented in literature by Maron in 1961 and continuing with naïve-Bayes [10] that proved to perform well. ID3 and C4.5 are well-known packages whose cores are making use of decision trees to build automatic classifiers [5, 6, 9]. K-nearest neighbor (k-NN) is another technique used in text categorization [20]. Another method to construct a text categorization system is by an inductive rule learning method. This type of classifiers is represented by a set of rules in disjunctive normal form that best cover the training set [14, 11, 3]. As reported in [18] the use of bigrams improved text categorization accuracy as opposed to unigrams use. In addition, in the last decade neural networks and support vector machines (SVM) were used in text categorization and they proved to be powerful tools [16, 21, 9].

2.2 Association Rule Mining

2.2.1 Association Rules Generation

Association rule mining is a data mining task that discovers relationships among items in a transactional database. Association rules have been extensively studied in the literature. The efficient discovery of such rules has been a major focus in the data mining research community. From the original *apriori* algorithm [1] there has been a remarkable number of variants and improvements culminated by the publication the FP-Tree growth algorithm [7]. However, most popular algorithms designed for the discovery of all types of association rules, are apriori-based.

Formally, association rules are defined as follows: Let $\mathcal{I} = \{i_1, i_2, ...i_m\}$ be a set of items. Let $\mathcal{D}$ be a set of transactions, where each transaction T is a set of items such that $T \subseteq \mathcal{I}$. Each transaction is associated with a unique identifier TID. A transaction T is said to contain X, a set of items in $\mathcal{I}$, if $X \subseteq T$. An *association rule* is an implication of the form "$X \Rightarrow Y$", where $X \subseteq \mathcal{I}, Y \subseteq \mathcal{I}$, and $X \cap Y = \emptyset$. The rule $X \Rightarrow Y$ has a *support* s in the transaction set $\mathcal{D}$ if $s\%$ of the transactions in $\mathcal{D}$ contain $X \cup Y$. In other words, the support of the rule is the probability that X and Y hold together among all the possible presented cases. It is said that the rule $X \Rightarrow Y$ holds in the transaction set $\mathcal{D}$ with *confidence* c if $c\%$ of transactions in $\mathcal{D}$ that contain X also contain Y. In other words, the confidence of the rule is the conditional probability that the consequent Y is true under the condition of the antecedent X. The problem of discovering all association rules from a set of transactions $\mathcal{D}$ consists of generating the rules that have a *support* and *confidence* greater than given thresholds. These rules are called *strong rules*.

The main idea behind apriori algorithm is to scan the transactional database searching for k-itemsets (k items belonging to the set of items I). As the name of the algorithm suggests, it uses prior knowledge for discovering frequent itemsets in the database. The algorithm employs an iterative search and uses k-itemsets discovered to find (k+1)-itemsets. The frequent itemsets are those that have the support higher than a minimum threshold.

2.2.2 Associative classifiers

Besides the classification methods described above, recently, and parallel to our work on associative text categorization, a new method that builds associative general clas-

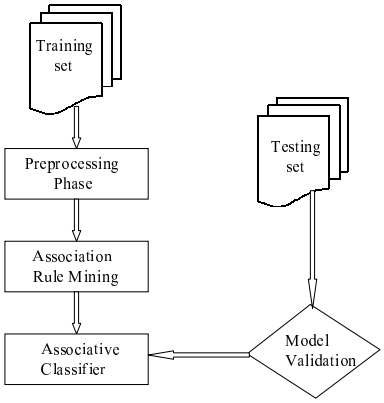

Figure 1. Construction phases for an association-rule-based text categorizer

sifiers has been proposed. In this case the learning method is represented by the association rule mining. The main idea behind this approach is to discover strong patterns that are associated with the class labels. The next step is to take advantage of these patterns such that a classifier is built and new objects are categorized in the proper classes.

Two such models were presented in the literature: CMAR [12] and CBA [13]. Although both of them proved to be effective and achieve high accuracy on relatively small UCI datasets [19], they have some limitations. Both models perform only single-class classification and were not implemented for text categorization. In many applications, however, and in text categorization in particular, multiple-class classification is required. In our paper we try to overcome this limitation and construct an associative classification model that allows single and multiple-class categorizations of text documents based on term co-frequency counts (i.e. a probabilistic technique that doesn't assume term independence).

3 Building an Associative Text Classifier

In this paper we present a method to build a categorization system that merges association rule mining task with the classification problem. This model is graphically presented in Figure 1.

Given a data collection, a number of steps are followed until the classification model is found. Data preprocessing represents the first step. At this stage cleaning techniques can be applied such as stopwords removal, stemming or term pruning according to the TF/IDF values (term frequency/inverse document frequency). The next step in building the associative classifier is the generation of association rules using an apriori-based algorithm. Once the entire set of rules has been generated an important step is to apply some pruning techniques for reducing the set of association rules found in the text corpora. The last stage in this process is represented by the use of the association rules set in the prediction of classes for new documents. The first three steps belong to the training process while the last one represents the testing (or classification) phase. More details on the process are given in the subsections below. If a document D_i is assigned to a set of categories $C = \{c_1, c_2, ...c_m\}$ and after word pruning the set of terms $T = \{t_1, t_2, ...t_n\}$ is retained, the following transaction is used to model the document: $D_i : \{c_1, c_2, ...c_m, t_1, t_2, ...t_n\}$ and the association rules are discovered from such transactions representing all documents in the collection. The association rules are, however, constrained in that the antecedent has to be a conjunction of terms from T, while the consequent of the rule has to a member of C.

3.1 Data Collection Preprocessing

In our approach, we model text documents as transactions where items are words or phrases from the document as well as the categories to which the document belongs, as described above. A data cleaning phase is required to weed out those words that are of no interest in building the associative classifier. We consider stopwording and term pruning as well as the transformation of documents into transactions as a pre-processing phase. Stopword removal and term pruning is done according to the TF/IDF values and a given list of stopwords. We have opted to selectively turn on and off stopwording depending upon the data set to categorize. It is only after the terms are selected from the cleansed documents that the transactions are formed. The subsequent phase consists of discovering association rules from the set of cleansed transactions.

3.2 Association Rule Generation

In our algorithm, as we shall see in this section, we take advantage of the apriori algorithm to discover frequent term-sets in documents. Eventually, these frequent itemsets associated with text categories represent the discriminate features among the documents in the collection. The association rules discovered in this stage of the process are further processed to build the associative classifier.

Using the apriori algorithm on our transactions representing the documents would generate a very large number of association rules, most of them irrelevant for classification. We use an apriori-based algorithm that is guided by the constraints on the rules we want to discover. Since we are building a classifier, we are interested in rules that indicate a category label, rules with a consequent being a category label. In other words, given the document model described

above, we are interested in rules of the form $T \Rightarrow c_i$ where $T \subseteq T$ and $c_i \subseteq C$. To discover these interesting rules efficiently we push the rule shape constraint in the candidate generation phase of the apriori algorithm in order to retain only the suitable candidate itemsets. Moreover, at the phase for rule generation from all the frequent k-itemsets, we use the rule shape constraint again to prune those rules that are of no use in our classification.

There are two approaches that we have considered in building an associative text classifier. The first one ARC-AC (Association Rule-based Classifier with All Categories) [22] is to extract association rules from the entire training set following the constraints discussed above. As a result of discrepancies among the categories in a text collection of a real-world application, we discovered that it is difficult to handle some categories that have different characteristics (small categories, overlapping categories or some categories having documents that are more correlated than others). As a result we propose a second solution ARC-BC (Associative Rule-based Classifier By Category) that solves such problems. In this approach we consider each set of documents belonging to one category as a separate text collection to generate association rules from. If a document belongs to more than one category this document will be present in each set associated with the categories that the document falls into. The ARC-BC algorithm is described in more detail below.

Algorithm ARC-BC Find association rules on the training set of the text collection when the text corpora is divided in subsets by category

Input A set of documents (D) of the form D_i : $\{c_i, t_1, t_2, ...t_n\}$ where c_i is the category attached to the document and t_j are the selected terms for the document; A minimum support threshold; A minimum confidence threshold;

Output A set of association rules of the form $t_1 \wedge t_2 \wedge ... \wedge t_n \Rightarrow c_i$ where c_i is the category and t_j is a term;

Method:

(1) $C_1 \leftarrow \{$Candidate 1 term-sets and their $support\}$
(2) $F_1 \leftarrow \{Frequent$ 1 term-sets and their $support\}$
(3) for $(i \leftarrow 2; F_{i-1} \neq \emptyset; i \leftarrow i+1)$ *do*{
(4) $C_i \leftarrow (F_{i-1} \bowtie F_{i-1})$
(5) $C_i \leftarrow C_i - \{c \mid (i-1)$ item-set of $c \notin F_{i-1}\}$
(6) $\mathcal{D}_i \leftarrow$ *FilterTable*$(\mathcal{D}_{i-1},\ F_{i-1})$
(7) foreach document d in $\mathcal{D}_i$ do {
(8) foreach c in C_i do {
(9) $c.support \leftarrow c.support$ + *Count*$(c,\ d)$
(10) }
(11) }
(12) $F_i \leftarrow \{c \in C_i \mid c.support > \sigma\}$
(13) }
(14) Sets $\leftarrow \bigcup_i \{c \in F_i \mid i > 1\}$
(15) R= $\emptyset$
(16) foreach itemset I in Sets do {
(17) $R \leftarrow R + \{I \Rightarrow Cat\}$
(18) }

In ARC-BC algorithm step (2) generates the frequent 1-itemset. In steps (3-13) all the k-frequent itemsets are generated and merged with the category in C_1. Steps (16-18) generate the association rules. The document space is reduced in each itereation by eliminating the transactions that do not contain any of the frequent itemsets. This step is done by *FilterTable*$(\mathcal{D}_{i-1},\ F_{i-1})$ function.

Table 1 presents a set of rules that are discovered in the text collection. Such rules are composing the classifier. Although the rules are human readable and understandable if the amount of rules generated is too large it is time consuming to read the set of rules for further tuning of the system. This problem leads us to the next subsection where pruning methods are presented. Although the rules are similar to those produced using a rule-based induced system, the approach is different. In addition, the number of words belonging to the antecedent could be large (in our experiments up to 10 words), while in some studies with rule-based induced systems, the rules generated have only one or a pair of words as antecedent [3].

3.3 Pruning the Set of Association Rules

The number of rules that can be generated in the association rule mining phase could be very large. There are two issues that must be addressed in this case. One of them is that such a huge amount 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.

The pruning methods that we study in this paper are the following: eliminate the specific rules and keep only those that are more general and with high confidence, and prune unnecessary rules by database coverage. Let us introduce the notions used in this subsection by the following definitions:

Definition 1 Being given two rules $T_1 \Rightarrow C$ and $T_2 \Rightarrow C$ we say that the first rule is more general if $T_1 \subseteq T_2$.

The first step of this process is to order the set of rules. This is done accordingly to the following ordering definition.

Definition 2 Being given two rules R_1 and R_2, R_1 is higher ranked than R_2 if:

(1) R_1 has higher confidence than R_2

(2) if the confidences are equal, supp(R_1) must exceed supp(R_2)

(3) both confidences and support are equal, but R_1 has less attributes in left hand side than R_2

With the set of association rules sorted, the goal is to select a subset that will build an efficient and effective classifier. In our approach we attempt to select a high quality subset of rules by selecting those rules that are general and

net $\wedge$ profit $\Rightarrow$ earn
agriculture $\wedge$ department $\wedge$ grain $\Rightarrow$ corn
assistance $\wedge$ bank $\wedge$ england $\wedge$ market $\wedge$ money $\Rightarrow$ interest
acute $\wedge$ coronary $\wedge$ function $\wedge$ left $\wedge$ ventricular $\Rightarrow$ myocardial-infarction
ambulatory $\wedge$ ischemia $\wedge$ myocardial $\Rightarrow$ coronary-disease
antiarrhythmic $\wedge$ effects $\Rightarrow$ arrhythmia

Table 1. Examples of association rules composing the classifier.

have high confidence. The most significant subset of rules is finally selected by applying the database coverage. The algorithm for building this set of rules is described below.

Algorithm Pruning the set of association rules

Input The set of association rules that were found in the association rule mining phase (S) and the training text collection (D)

Output A set of rules used in the classification process

Method:

(1) sort the rules according to **Definition 1**
(2) **foreach** rule in the set S
(3) find all those rules that are more specific according to (**Definition 2**)
(4) prune those that have lower confidence
(5) a new set of rules S' is generated
(6) **foreach** rule R in the set S'
(7) go over D and find those transactions that are covered by the rule R
(8) **if** R classifies correctly at least one transaction
(9) select R
(10) remove those cases that were covered by R

3.4 Prediction of Classes Associated with New Documents

The set of rules that were selected after the pruning phase represent the actual classifier. This categorizer will be used to predict to which classes new documents are attached. Given a new document, the classification process searches in this set of rules for finding those classes that are the closest to be attached with the document presented for categorization. This subsection discusses the approach for labelling new documents based on the set of association rules that forms the classifier.

A trivial solution would be to attach to the new document the class that has the most rules matching this new document or the class associated with the first rule that apply to the new object. However, in the text categorization domain, multi-class categorization is an important and challenging problem that needs to be solved. In our approach we give a solution to this problem by introducing the *dominance factor*. By employing this variable we allow our system to assign more than one category. The dominance factor δ is the proportion of rules of the most dominant category in the applicable rules for a document to classify. Given a document to classify, the terms in the document would yield a list of applicable rules. If the applicable rules are grouped by category in their consequent part and the groups are ordered by the sum of rules' confidences, the ordered groups would indicate the most significant categories that should be attached to the document to be classified. We call this order category dominance, hence the dominance factor δ. The dominance factor allows us to select among the candidate categories only the most significant. When δ is set to a certain percentage a threshold is computed as the sum of rules' confidences for the most dominat category times the value of the *dominance factor*. Then, only those categories that exceed this threshold are selected. TakeKClasses(S,δ) function selects the most k significant classes in the classification algorithm.

The next algorithm describes the classification of a new document.

Algorithm Classification of a new object

Input A new object to be classified o; The associative classifier (ARC); The dominance factor δ; The confidence threshold τ;

Output Categories attached to the new object

Method:

(1) $S \leftarrow \emptyset$ /*set of rules that match o*/
(2) **foreach** rule r in ARC (the sorted set of rules)
(3) **if** ($r \subset o$) { count++ }
(4) **if** (count == 1)
(5) fr.conf $\leftarrow$ r.conf /*keep the first rule confidence*/
(6) $S \leftarrow S \cup r$
(7) **else if** (r.conf $>$ fr.conf-τ)
(8) $S \leftarrow S \cup r$
(9) **else** exit
(10) divide S in subsets by category: $S_1, S_2 ... S_n$
(11) **foreach** subset $S_1, S_2 ... S_n$
(12) sum the confidences of rules and divide by the number of rules in S_k
(13) **if** it is single class classification
(14) put the new document in the class that has the highest confidence sum
(15) **else** /*multi-class classification*/
(16) TakeKClasses(S,δ)
(17) assign these k classes to the new document

4 Experimental Results and Performance Study

4.1 Text Corpora

In order to be able to objectively evaluate our algorithm vis-a-vis other approaches, like other researchers in the field of automatic text categorization, we used the Reuters-21578 text collection [15] as benchmarks. This text database is described below. Text collections for experiments are usually split into two parts: one part for training or building the classifier and a second part for testing the effectiveness of the system.

There are many splits of the Reuters collection; we chose to use the *ModApte* version. This split leads to a corpus of 12,202 documents consisting of 9,603 training documents and 3,299 testing documents. There are 135 topics to which documents are assigned. However, only 93 of them have more than one document in the training set and 82 of the categories have less than 100 documents [22]. Obviously, the performances in the categories with just a few documents would be very low, especially for those that do not even have a document in the training set. Among the documents there are some that have no topic assigned to them. We chose to ignore such documents since no knowledge can be derived from them. Finally we decided to test our classifiers on the ten most populated categories with the largest number of documents assigned to them in the training set. Other researchers have used the same strategy [17], which constrained us to do the same for the sake of comparison. By retaining only the ten most populated categories we have 6488 training documents and 2545 testing documents. On these documents we performed stopword elimination but no stemming.

4.2 Experimental Results

On this data set we tested our classification system ARC-BC on a Pentium III 700MHz dual processor machine running Linux. Several measurements have been used in previous studies for evaluation. Some measures, as well as those used in our evaluation, can be defined in terms of precision and recall. The formulae for precision and recall are given below: $R = \frac{a}{a+c}$ and $P = \frac{a}{a+b}$. The terms used to express precision and recall are given in the contingency table Table 2.

For evaluating the effectiveness of our system, we used the breakeven points. The breakeven point is the point at which precision equals recall and it is obtained as reported in [4].

When dealing with multiple classes there are two possible ways of averaging these measures, namely, macro-average and micro-average. In the macro-averaging, one contingency table per class is used, the performance measures are computed on each of them and then averaged. In micro-averaging only one contingency table is used for all classes, an average of all the classes is computed for each cell and the performance measures are obtained therein. The macro-average weights equally all the classes, regardless of how many documents belong to it. The micro-average weights equally all the documents, thus favoring the performance on common classes.

Category *cat*		human assignments	
		Yes	No
classifier assignments	Yes	a	b
	No	c	d

Table 2. Contingency table for category *cat*

In Table 3 we report the micro-averages for ARC-BC when both support and dominance factor were varied. The results are computed on the ten most populated categories in Reuters dataset. As described in Section 3.3 we applied some pruning techniques on the discovered association rule set. Table 3 reports micro-averages when the classifier is built by employing the pruning methods (i.e. no pruning at all (without pruning), removing specific rules (rm-s) and removing specific rules plus database coverage applied (rm-s + db-cov).

micro BEP	supp=10%			supp=15%			supp=20%		
	δ=50	δ=70	δ=90	δ=50	δ=70	δ=90	δ=50	δ=70	δ=90
without pruning	82.0	84.6	85.6	81.8	84.4	85.8	81.0	86.3	84.0
rm-s	65.5	72.9	76.3	68.2	75.0	78.8	68.2	76.1	79.7
rm-s + db-cov	71.0	79.0	82.3	71.4	79.6	73.1	70.1	70.4	84.1

Table 3. Precision/Recall-breakeven point micro-averages for ARC-BC

Table 4 (the results for the other classification systems are reported as given in [9]) shows a comparison between our ARC-BC classifier and other well-known methods. The measures used are precision/recall-breakeven point, micro-average and macro-average on ten most populated Reuters categories. Our system proves to perform well as compared to the other methods. It outperforms most of the conventional methods, but it does not perform better than SVM. In addition to these results, our system has two more features. First, it is very fast in both training and testing phases (see Table 6). The times reported are for all training and testing documents. Second, it produces readable and understandable rules that can be easily modified by humans (see Table 1). Table 5 reports the improvements in the response of the system when human tuning was applied. The support was set to 20% which made *corn* and *wheat* categories to perform very poor. By reading the rules we noticed that by adding 4 more rules for each of these categories the perfor-

BEP	ARC-BC with δ=50			Bayes	Rocchio	C4.5	k-NN	bigrams	SVM (poly)	SVM (rbf)
	10%	15%	20%							
acq	**90.9**	89.9	87.8	91.5	92.1	85.3	92.0	73.2	94.5	95.2
corn	69.6	**82.3**	70.9	47.3	62.2	87.7	77.9	60.1	85.4	85.2
crude	77.9	77.0	**80.7**	81.0	81.5	75.5	85.7	79.6	87.7	88.7
earn	**92.8**	89.2	86.6	95.9	96.1	96.1	97.3	83.7	98.3	98.4
grain	68.8	72.1	**73.1**	72.5	79.5	89.1	82.2	78.2	91.6	91.8
interest	70.5	70.1	**75.3**	58.0	72.5	49.1	74.0	69.6	70.0	75.4
money-fx	70.5	**72.4**	70.5	62.9	67.6	69.4	78.2	64.2	73.1	75.4
ship	**73.6**	73.2	63.0	78.7	83.1	80.9	79.2	69.2	85.1	86.6
trade	68.0	69.7	**69.8**	50.0	77.4	59.2	77.4	51.9	75.1	77.3
wheat	84.8	**86.5**	85.3	60.6	79.4	85.5	76.6	69.9	84.5	85.7
micro-avg	**82.1**	81.8	81.1	72.0	79.9	79.4	82.3	73.3	85.4	86.3
macro-avg	76.74	**78.24**	76.32	65.21	79.14	77.78	82.05	67.07	84.58	86.01

Table 4. Precision/Recall-breakeven point on ten most populated Reuters categories for ARC-BC and most known classifiers

ARC-BC supp=20% δ=90 rm-s+db-cov		
BEP	initial set of rules	manual tuned set of rules
acq	89.6	89.6
corn	2.0	63.6
crude	80.0	80.0
earn	92.7	92.7
grain	92.5	81.9
interest	57.7	57.7
money-fx	77.9	77.9
ship	61.3	61.3
trade	75.5	75.5
wheat	6.0	63.5
micro-avg	84.14	84.62
macro-avg	63.55	74.41

Table 5. Micro-average Precision/Recall-breakeven point for ten most populated Reuters categories - manual tuning of the classifier

mances improved as presented in Table 5.

A comparison between the pruning methods is given in Table 7. By applying the pruning methods the accuracy of the classifier is not improved. However, the reduction in number of rules represents a step further in manually or automatically tuning of the system.

5 Conclusion and Future Work

This paper introduced a new technique for text categorization. It employs the use of association rules. Our study provides evidence that association rule mining can be used for the construction of fast and effective classifiers for automatic text categorization. We have presented an association rule-based algorithm for building the classifier: **ARC-BC** that considers categories one at a time. The algorithm assume a transaction-based model for the training document set.

support	training	testing
10%	18	3
15%	9	2
20%	8	2

Table 6. Training and testing time (in seconds) with respect to the support threshold for Reuters-21578 dataset

The experimental results show that the association rule-based classifier performs well and its effectiveness is comparable to most well-known text classifiers. One major advantage of the association rule-based classifier is its relatively fast training time. Moreover, the rules generated are understandable and can easily be manually updated or adjusted if necessary. The maintenance of the classifier is straight forward. In the case of ARC-BC, when new documents are presented for retraining, only the concerned categories are adjusted and the rules could be incrementally updated.

The introduction of the dominance factor δ allowed multi-class categorization. However, other feature selection techniques, such as latent semantic analysis could improve the results by giving an insight on the discriminative feature among classes. We are working on reducing the number of features, thus better discrimination among classes is expected. Currently the discovered rules consider the presence of terms in documents to categorize. We are studying possibilities to take into account the absence of terms in the classification rules as well.

BEP	ARC-BC with $\delta = 50$ and supp=15%		
	w/o pruning	rm-s	rm-s + db-cov
	3072 rules	383 rules	127 rules
acq	89.9	84.2	76.6
corn	82.3	62.7	2.8
crude	77.0	58.4	64.8
earn	89.2	85.6	78.0
grain	72.1	56.4	71.8
interest	70.1	60.8	63.6
money-fx	72.4	62.3	68.7
ship	73.2	67.6	59.0
trade	69.7	59.2	73.5
wheat	86.5	46.9	24.7
micro-avg	81.8	68.2	71.4
macro-avg	78.24	64.40	58.53

Table 7. Precision/Recall-breakeven point for ten most populated Reuters categories with different pruning methods

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. 1993 ACM-SIGMOD Int. Conf. Management of Data*, pages 207–216, Washington, D.C., May 1993.

[2] M.-L. Antonie, O. R. Zaïane, and A. Coman. Application of data mining techniques for medical image classification. In *Second International ACM SIGKDD Workshop on Multimedia Data Mining*, pages 94–101, San Francisco, USA, August 2001.

[3] C. Apte, F. Damerau, and S. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1994.

[4] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. On feature distributional clustering for text categorization. In *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 146–153, New Orleans, US, 2001.

[5] W. Cohen and H. Hirsch. Joins that generalize: text classification using whirl. In *4th International Conference on Knowledge Discovery and Data Mining (SigKDD'98)*, pages 169–173, New York City,USA, 1998.

[6] W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems*, 17(2):141 –173, 1999.

[7] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *ACM-SIGMOD*, Dallas, 2000.

[8] D. A. Hull. Improving text retrieval for the routing problem using latent semantic indexing. In *17th ACM International Conference on Research and Development in Information Retrieval (SIGIR-94)*, pages 282–289, 1994.

[9] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *10th European Conference on Machine Learning (ECML-98)*, pages 137–142, 1998.

[10] D. Lewis. Naïve (bayes) at forty: The independence assumption in information retrieval. In *10th European Conference on Machine Learning (ECML-98)*, pages 4–15, 1998.

[11] H. Li and K. Yamanishi. Text classification using esc-based stochastic decision lists. In *8th ACM International Conference on Information and Knowledge Management(CIKM-99)*, pages 122 –130, Kansas City,USA, 1999.

[12] W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *IEEE International Conference on Data Mining (ICDM'01)*, San Jose, California, Novermber 29-December 2 2001.

[13] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *ACM Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD'98)*, pages 80–86, New York City, NY, August 1998.

[14] I. Moulinier and J.-G. Ganascia. Applying an existing machine learning algorithm to text categorization. In S.Wermter, E.Riloff, and G.Scheler, editors, *Connectionist statistical,and symbolic approaches to learning for natural language processing*. Springer Verlag, Heidelberg, Germany, 1996. Lecture Notes for Computer Science series, number 1040.

[15] The reuters-21578 text categorization test collection. http://www.research.att.com/˜lewis/reuters21578.html.

[16] M. Ruiz and P. Srinivasan. Neural networks for text categorization. In *22nd ACM SIGIR International Conference on Information Retrieval*, pages 281–282, Berkeley, CA, USA, August 1999.

[17] F. Sebastiani. Machine learning in automated text categorization. Technical Report IEI-B4-31-1999, Consiglio Nazionale delle Ricerche, Pisa, Italy, 1999.

[18] C. M. Tan, Y. F. Wang, and C. D. Lee. The use of bigrams to enhance text categorization. *Journal of Information Processing and Management*, 2002. http://www.cs.ucsb.edu/ yfwang/papers/ig&m.pdf.

[19] University of california irvine knowledge discovery in databases archive. http://kdd.ics.uci.edu/.

[20] Y. Yang. An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Carnegie mellon University, April 1997.

[21] Y.Yang and X.Liu. A re-examination of text categorization methods. In *22nd ACM International Conference on Research and Development in Information Retrieval (SIGIR-99)*, pages 42–49, Berkeley,US, 1999.

[22] O. R. Zaïane and M.-L. Antonie. Classifying text documents by associating terms with text categories. In *Thirteenth Australasian Database Conference (ADC'02)*, pages 215–222, Melbourne, Australia, January 2002.