Text Categorization based on Boosting Association Rules

Yongwook Yoon
Dept. of Computer Sci. & Eng.
POSTECH
Pohang 790-784, South Korea
ywyoon@postech.ac.kr

Gary G. Lee
Dept. of Computer Sci. & Eng.
POSTECH
Pohang 790-784, South Korea
gblee@postech.ac.kr

Abstract

Associative classification is a novel and powerful method originating from association rule mining. In the previous studies, a relatively small number of high-quality association rules were used in the prediction. We propose a new approach in which a large number of association rules are generated. Then, the rules are filtered using a new method which is equivalent to a deterministic Boosting algorithm. Through this equivalence, our approach effectively adapts to large-scale classification tasks such as text categorization. Experiments with various text collections show that our method achieves one of the best prediction performance compared with the state-of-the-arts of this field.

1 Introduction

The associative classifier uses the association rules produced by a frequent pattern mining algorithm [1], [7]. Since the associative classifier is a rule-based classifier, humans can easily understand its operation, and the prediction result provides a simple and direct interpretation. Moreover, it can exploit the combined information of multiple features as well as a single feature, while Naïve Bayes and Support Vector Machine(SVM) classifiers only consider a single feature. This means that text categorization can use phrase occurrence information as well as word occurrence information.

In text classification problems, *large-scale* is inevitable due to the large number of word features, class labels and example documents in the text corpus. Many high-order[1] association rules are generated in the induction procedure for the associative classifier. Generally, high-order rules are more informative; hence the classification with those rules has a better performance [15], [22]. However, as the order of the rules grows, the number of generated rules increases very rapidly so that the computational complexity is unbearable.

[1]In associative rules, *order* means the number of words occurring in the rule.

The paradigm of the previous methods of associative classification was to use a small number of *high-quality* rules regardless of the order of rules (or preferring high-order rules) [12], [21]. To reduce the time of rule mining, they limited the number of word features by eliminating the words that were estimated as not useful in the classification. In contrast, our method uses most words in the vocabulary except for stop-words as features. To avoid generating an excessive number of rules, the order of the rules is limited to less than a prescribed threshold.

Since the number of those generated rules can be several millions, a small number of rules should be selected for prediction in real situations. When predicting for test instances, the selected rules votes as much as their individually weighted scores. We propose a new rule selection algorithm based on validation by training examples. In addition, we show that this rule selection process is equivalent to a deterministic AdaBoost algorithm [6]. This analogy is utilized to filter the generated rules, which greatly improves the training and generalization errors. Our method fits well to text corpora because a large number of low-order rules can cover the high dimensional feature space of test instances.

This paper is structured as follows. Section 2 introduces the associative text classification and provides the problem formulation. Section 3 describes the classification rule mining. Section 4 explains the algorithm of boosting association rules and presents an analytical justification. Section 5 proposes our method of handling multi-label predictions. Section 6 shows experimental results and the analyses. Finally, the paper concludes in Section 7.

2 Associative Classification

2.1 Application to Text Categorization

The overall system architecture of the associative classification for text documents is shown in Fig. 1. The left-hand side of the figure denotes a training process and the right-hand side a testing process.

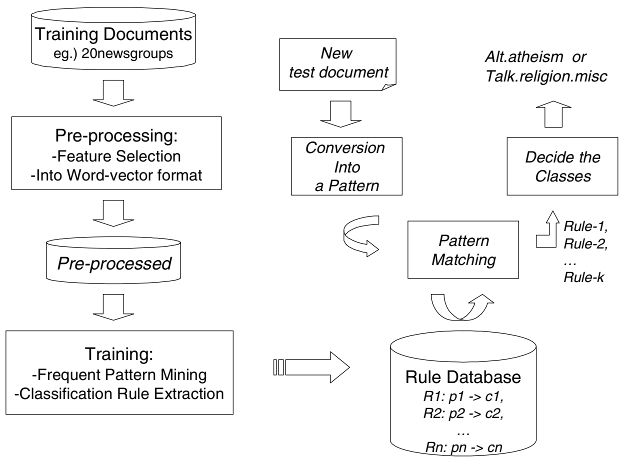


Figure 1. Associative Classification – Training and Testing

First, raw text data is pre-processed for rule mining. Each document is converted into a transactional record format. From this pre-processed database, we mine frequent patterns, i.e. classification rules. Because the number of mined rules is very large, we filter out some useless or redundant rules and select a small number of well-qualified rules. This process is called *Pruning*. Finally, we construct a classification-rule database with these selected rules.

When a new document comes in to be classified, we convert it into a pattern of words and search the database for matching rules. With matched rules, we decide which class the test document is assigned to.

2.2 Formal Definition

The database for a classification task consists of attributes and their instances. The instances of attribute values constitute training (or testing) examples. If we denote the set of features as $A = \{A_1, A_2, ..., A_d\}$, where d is the dimension of the feature space, then a set called *domain* can be represented as $X = A_1 \times A_2 \times ... \times A_d$. Let $Y = \{c_1, c_2, ..., c_{|Y|}\}$ be the set of class labels, then the set of training examples D is

$$D = \{(x_j, y_j) \mid x_j \in X,\ y_j \in Y\}.$$

An *item* is defined as the instance value of an attribute. *Itemset*(or *pattern*) is a term denoting the instance of single or combined features. Association rule mining extracts frequent itemsets from a transactional database. We define the *support* of a pattern as the number of training examples in which the pattern occurs. The pattern $p_j \in A_{i_1} \times A_{i_2} \times ... \times A_{i_k}$ is called *frequent* if the support of the pattern exceeds some given threshold value *min_sup*.

A classification rule r_k is a mapping from a set of features to the set of class labels:

$$A_{i_1} \times A_{i_2} \times ... \times A_{i_k} \rightarrow Y. \tag{1}$$

If the rule (1) is frequent, then it is called a *class association rule(CAR)* [12]. We define the *confidence* of a rule as the support of the rule divided by the support of the pattern of the left-hand side of the rule. From application to application, the extent of CAR may be confined to those of which confidence values exceed a given threshold *min_conf*.

For example, assume that in the document collection about sports games, the following word phrase and the associated game was mined using a rule mining algorithm,

$$run, diamond \rightarrow baseball\ (5, 0.71),$$

where the numbers within the parenthesis denote the support and confidence of the rule, respectively. This rule says that the co-occurrence of *run* and *diamond* in a document implies the theme of the document to be the *baseball*.

Let $R = \{r_1, r_2, ..., r_{|R|}\}$ be the final classification rule set. When we have a test example x, we apply the rules to x. Let s_{ij} be the score which rule r_i produces supporting that the class label of x is c_j. Then, S_j, the total score for c_j when the entire rule set is considered can be written as:

$$S_j = \sum_{r_i \in R \text{ and } c_j \in Y} s_{ij}. \tag{2}$$

Then, the final prediction label $\hat{c}$ for x is determined such that

$$\hat{c} = \arg\max_{c_j \in Y} S_j.$$

3 Generating Class Association Rules

3.1 Transactional Representation for Document Examples

Most text corpora have a high-dimensional feature space. Since they are not relational databases, there is no predetermined length for an example record. The average number of words in a document is far less than the vocabulary size, which shows the sparsity of word distribution in text collections. A document can be modeled statistically as the

Dataset	msup / mconf	# rules
Sick *from* UCI	1 / 50	71,828
e-mail	5 / 0	42,182
20 newsgroups	0.02 / 5	8,185,780

Table 1. The number of generated rules

events of the words that constitute the document. There are two different document models: the *Multi-variate Bernoulli Model* and the *Multinomial Model* [13]. The former ignores the count information of words and uses only binary information (*present* or *not-present*), while the latter includes the information of word count.

In the multinomial model, a document d which is a sequence of words, $\langle w_1, w_2, w_4, w_2 \rangle$, is modeled as the set $\{w_1, w_2, w_4\}$ and their occurrence counts. But it is known that information about the multiple occurrence of words does not yield much additional assistance for an exact classification [4], [19]. Thus, as the input format of a document to the mining process for association rules, we choose a "transactional" format which has no occurrence count information.

3.2 Mining Classification Rules

The frequent pattern mining was done with the method of Han et al. [7]. CBA [12] and CMAR [11] took 1% and 50% as threshold values of *min_sup* and *min_conf* respectively. Such high threshold values cause a relatively small number of generated rules, which could raise the classifying precision. But this may miss useful information contained in the rules which have lower confidence or support than the thresholds.

Our approach adopts the opposite direction to the above heuristic. We lower the values of *min_sup* and *min_conf* to the lowest possible. Hence, we collect many rules that are considered slightly better than *a random guess*. Table 1 lists the thresholds in % and the numbers of rules mined from raw texts. The first data is used in CBA, and the second is an e-mail collection by Itskevitch [8]. With 20 newsgroups we produced twenty times more rules compared to Itskevitch.

Lowering association mining thresholds can make it impossible to generate rules within a reasonable time unless another constraint is applied. We limit the order of generated rules to less than a small number. In the case of text corpus, the order is limited as less than or equal to 2 (or 3). Despite this limitation, the number of generated rules is still too large. The next section considers the rule selection process for better classification performance.

4 Boosting Association Rules

4.1 Analogy to Boosting

According to PAC-learning theory [9], a strong PAC-learning algorithm is an algorithm that, given $\varepsilon, \delta > 0$ and access to random examples, outputs with probability $1-\delta$ a hypothesis with error at most ε. A weak PAC-learning algorithm satisfies the same conditions but only for $\varepsilon \leq 1/2 - \gamma$ where $\gamma > 0$.

Schapire [17] proved that any weak learning algorithm can be efficiently transformed or *boosted* into a strong learning algorithm. Generally, *Boosting* refers to producing a very accurate prediction rule by combining rough and moderately inaccurate rules-of-thumb. AdaBoost [6] is a very effective and efficient boosting algorithm, where the weak learner produces the hypotheses with any $\varepsilon_t \in [0, 1]$ and the hypotheses are boosted adaptively.

A class association rule with a low confidence value may fall under the class of weak classifiers. The pruning procedure of our classification method can be thought as the boosting procedure where the weak classifiers are not retrained but *selected* from the set of association rules according to the modified distribution of training examples.

4.2 A New Algorithm based on Boosting

Fig. 2 represents our new algorithm to boost weak association rules. This algorithm is a modification of *the database coverage pruning* [12] based on the principle of Boosting. To simplify our analysis, we begin with a binary classification problem where $Y = \{0, 1\}$. A class association rule in Fig. 2 corresponds to a weak hypothesis of **AdaBoost**.

In Fig. 2, the main loop iterates for the rule index t. For each iteration, r_t is applied to all of the training examples remained so far. If the prediction is correct, then the cover count v_i is increased as much as the confidence value of r_t at Line 2-b-ii. This is equivalent to the weight update step of **AdaBoost** [6]:

$$w_i^{t+1} = w_i^t \beta^{l_i^t} = w_i^t \exp(l_i^t \ln \beta),$$

where w_i^t is the weight of example x_i at t round and β is a weight update factor. l_i^t, the loss of x_i for r_t, corresponds to the confidence value which is added to v_i by r_t. **BoostCARs** is *not* an adaptive algorithm since β is fixed to some value less than 1 for all $t = 1, ..., T$.

Next, if the updated v_i exceeds the threshold *cvth*, then the example (x_i, y_i) is removed from the database. This corresponds to the modification in the distribution of the examples in **AdaBoost**. Finally, after all T iterations, **BoostCARs** yields a final hypothesis h_f combined with the selected hypotheses r_t's.

```
Algorithm BoostCARs
Input Class Association Rules: $\{r_1, r_2, ..., r_T\}$,
      Database: $\{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$,
      cvth (database coverage threshold)
Initialize the cover count: $v_i^1 = 0$ for $i = 1, ..., N$,
      sort the rules $\{r_j\}$ in the descending order of confidence
For $t = 1, 2, ..., T$
  1. $CorrectPred = false$
  2. For $i = 1, 2, ..., N$
     (a) Apply $r_t$ to $x_i$.
     (b) If $r_t$ predicts $y_i$, then
         i. $CorrectPred = true$
         ii. $v_i = v_i + \mathrm{conf}(r_t)$
         iii. If $v_i >$ cvth
              then delete $(x_i, y_i)$ from the database.
  3. If $CorrectPred = false$
     then $\mathrm{conf}(r_t) = 0$
Output the final hypothesis
     $h_f(x) = \arg\max_{y \in Y} \sum_{t: h_t(x) = y} \mathrm{conf}(r_t)$
```

Figure 2. Algorithm of Boosting weak association rules

4.3 Training Error

It can be shown that the training error of **BoostCARs** approaches to zero exponentially fast as the training progresses.

Theorem 1. *Suppose **BoostCARs** chooses the class association rule whose error $\varepsilon_t \leq 1/2 - \gamma$ for some $\gamma > 0$ at each round t (t $= 1, ..., T$). Then, the error ε of the final hypothesis h_f output by **BoostCARs** is bounded above by*

$$\varepsilon = \frac{|\{i : h_f(x_i) \neq y_i\}|}{N} \leq \exp(-T\gamma^2/2). \qquad (3)$$

Proof Sketch. First, we convert **BoostCARs** into an equivalent form of the deterministic version of **AdaBoost**, where the weight update factor β_t is the same for all $t = 1, ..., T$. We choose β to be $1-\gamma$. Also, we change r_t in **BoostCARs** into $h' : X \rightarrow [0, 1]$ such that

$$h'_t(x_i) = \begin{cases} \mathrm{conf}(r_t) & \text{if } y_i = 1 \\ 1 - \mathrm{conf}(r_t) & \text{if } y_i = 0 \end{cases}.$$

Then, the remining proof procedure becomes equivalent to that of **Hedge**(β) algorithm of the *on-line allocation model* [6]. If we apply the analysis to **BoostCARs**, then the error bound (3) can be directly derived. □

The theorem says that, if we collect many weak association rules all of which performances are slightly better than random guessing by γ, then the error will decrease exponentially fast. This analysis can be easily extended to the case of multi-class problems if we adopt the techniques in [6].

4.4 Generalization Error

In our approach, since the *min_sup* and *min_conf* parameters are set to the lowest, the number of the omitted features is minimized. Thus, these features can cover the attributes of the test domain better. In addition, since BoostCARs has abundant low-order classification rules, it can cover the word patterns of test documents better than the approach which adopts high-order rules. This property enables BoostCARs to show minimized generalization errors.

Schaprie et al. [18] defined the classification *margin* of a training example to be the difference between the number of correct votes and the maximum number of votes received by any incorrect label. They proved that maximizing the margin can improve the generalization error of a classifier and Boosting tends to increase the margins of training examples when the final classifier is combined from weak hypotheses. Although the final classifier becomes larger, its test error constantly decreases.

Based on the margin theory, we adjust parameter *coverage threshold* (*cvth*) in Fig. 2 so that BoostCARs can achieve the minimum generalization error. It can be achieved if the whole range of generated rules are selected evenly according to the principle of Boosting and there are no remaining rules and examples when the rule selection process is completed. Then, the margin of the examples is maximized and the value of *cvth* is selected as the value of that case.

5 Multi-Label Classification

Our associative classifier yields the prediction scores for all class labels at once. Thus, it is necessary to set a threshold for the scores to determine whether the prediction is right or wrong [20]. We propose a novel thresholding scheme that is similar to "RCut" in [24]. RCut always predicts k class labels with the highest scores for each test document.

We assume that, regardless of the number of answer labels, the prediction score of an answer label occupies larger than some ratio in the total prediction score of a document. Let ρ be such ratio threshold. If we denote S_j as the prediction score of class c_j as in (2), then the estimated class labels $\{\hat{c}\}$ is determined by this relation:

$$\hat{c} \in \{c_j \,|\, c_j \in Y \text{ and } \frac{S_j}{\sum_{c_i \in Y} S_i} \geq \rho\}. \qquad (4)$$

The uneven distribution in the number of training examples of different class labels invokes another problem. In such circumstance, the class label which has a large number of training examples will have much more classification rules than the class label which has a small number of training examples. Thus, the prediction score of the latter would always be less than the score of the former even when the latter is an correct label. The sum of prediction scores S_j for class c_j in (2) would no longer denote a correct prediction score.

We introduce *a normalized prediction score model* to compensate for such uneven distribution of training examples. First, we define a weight function w of a class label c_j as

$$w(c_j) = \frac{\frac{1}{|Y|}\sum_{i=1}^{|R|} \mathrm{conf}(r_i)}{\sum_{k \in R_j} \mathrm{conf}(r_k)}, \quad (5)$$

where $\mathrm{conf}(r_i)$ is the confidence value of r_i and R_j is the set of the rules with the label c_j as their consequents. When $w(c_j)$ is multiplied to the final prediction score S_j, the score $\tilde{S_j}$ is effectively normalized between the class labels:

$$\tilde{S_j} = w(c_j)^{1/2} \cdot S_j, \quad (6)$$

where the square root of $w(c_j)$ smoothes further the effect of w. When we predict on a severely imbalanced corpus, we replace S_j in (4) with this $\tilde{S_j}$ before judging correct labels.

6 Experiments

6.1 Test Collections

Reuters-21578 and *20 newsgroups* [10] are multi-class and slightly multi-label text collections. Reuters-21578 is a collection of articles from Reuters newswire. We used the *ModApte* split version [2] from which we further selected the documents with the top 10 TOPICS categories, which are *earn, acq, money-fx, grain, crude*, etc. This final set is the same as the one used in HARMONY [21]. 20 newsgroups is a collection of USENET mail postings whose category set includes the names of the 20 discussion groups, for example, *comp.os.ms-windows.misc*. Some statistics on the collections are listed in Table 2, where we can find that 20 newsgroups is larger and more complex than Reuters-21578.

OHSUMED [16] is the collection of citations to medical journals from the year 1987 to 1991. Instead of the original OHSUMED collection, many researches have used "Heart Diseases (HD)" subset of MeSH(Medical Subject Headings). It is also referred to as "HD-119". HD-119 contains 16,595 documents in 107 distinct categories. We divided this sub collection according to the publication year: from 1987 to 1990 for training, 1991 for testing.

We used BOW-toolkit [14] for pre-processing of the documents. The header part except for the title was removed

	Reuters-21578	20 newsgroups
# documents	9,979	19,997
# classes	10	20
vocabulary size	22,424	90,833
# words /doc	49.2	77.3
# labels /doc	1.10	1.05

Table 2. Statistics on the text databases in our experiment

from the training documents. We filtered out general stop words and conducted no stemming. We prepared the training input according to the document model in Section 3.1. We implemented our associative classification system with C++. Our codes were executed The program was run on a Linux machine with 4GB memory and 2.8GHz CPU speed.

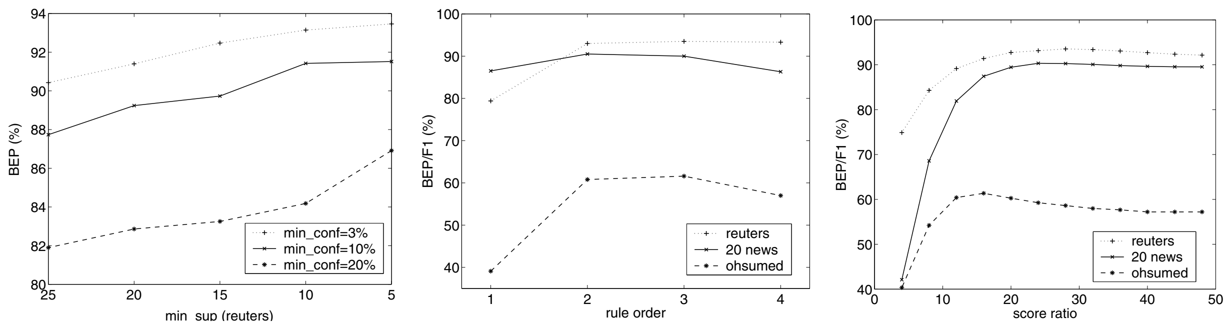
6.2 Parameter Selection

Fig. 3(a) represents the prediction accuracy of our associative classifier for various *min_sup* and *min_conf* thresholds when applied to Reuters-21578 collection. In all results shown in this paper, *min_sup* is represented with absolute numbers not with ratios. The accuracy is represented with Breakeven point (BEP) between the recall and the precision measures from the Information Retrieval community. As the *min_sup* and *min_conf* thresholds become lower, the number of rules grows larger and the performance improves accordingly. These results agree with our intuitions in the previous sections.

In Fig. 3(b), the x-axis represents the highest order of rules included in the rule set, and the y-axis BEP or F1 score. F1 is the harmonic mean of the precision and the recall. BEP is used in the Reuters and 20 newsgroups, and F1 in the OHSUMED collection. As the order exceeds some number, the generalization performance starts to drop due to overfitting. The order at which the performance decreases differs to text corpora. Most large-scale text collections show the best performance at the order of *two*. The reason why such high-order rules do not assist in raising the prediction accuracy is that high-order word phrases in the rules have lower probability to occur in the test set than low-order phrases.

Fig. 3(c) shows the selection process of optimal score ratio thresholds in the multi-label associative classification. As the average number of class labels per document becomes larger, the ratio decreases. We select optimal values through validation using training examples.

Fig. 4 shows a detailed process of **BoostCARs** algorithm for 20 newsgroups data. The x-axis represents the id of the generated rules which are sorted in the descending order of confidence values. The y-axis represents the number of re-



(a) minimum support and confidence (b) rule order (c) score ratio threshold

Figure 3. Selection of Classification Performance Parameters

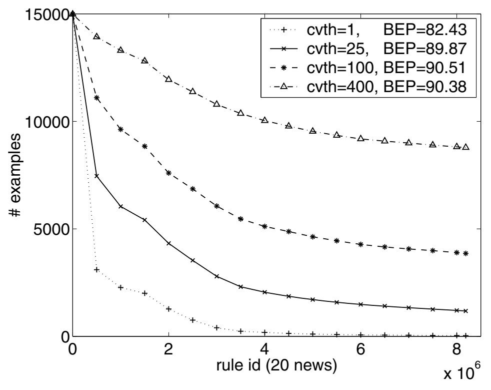


Figure 4. Selection of coverage threshold

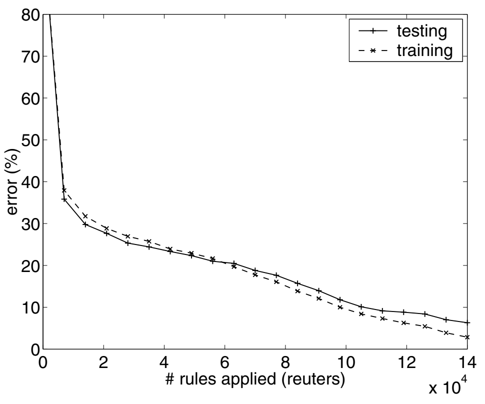


Figure 5. Learning curve of Associative Classifier

maining training examples when rule r_t predicts for the examples remaining at round $t-1$. If the coverage threshold is low, then the training examples are exhausted prematurely. If the coverage threshold is high, then the probability for incorrect rules to be selected increases because inappropriate training examples still remain. The coverage threshold should be selected so that the training examples can be effectively used in the process. The selection of 100 as *cvth* value shows the best performance in this case.

Fig. 5 represents the learning curve of our associative classifier for Reuters. The x-axis represents the number of rules included in the prediction. As the size of the combined classifier grows, the training error decreases constantly. At the same time, the test error continues to decrease without overfitting to the training examples. From this, we find that the margin theory on the generalization error also applies well to our boosting algorithm.

6.3 Performance Comparison

Table 3 lists the classification accuracies of HARMONY and SVM for Reuters-21578. The result of linear SVM classifier is from [5]. The performance for each class is measured with BEP, and finally averaged for all the test instances. BCAR with *min_sup* threshold 5 shows the best BEP, 93.5%. This performance is obtained when the order of rules is set to three and the *min_conf* threshold set to 20, which is slightly better than random guessing.

In addition to such excellent classification accuracy, our approach shows good efficiency in computation. Table 4 shows the time spent from mining association rules up to predicting test instances for Reuters-21578. The computing time of BCAR grows as *min_sup* value is lowered. Although the time of BCAR with *min_sup* of 5 is several times of that of HARMONY, BCAR with *min_sup* of 20 shows comparable execution time to HARMONY while its accuracy is kept better than HARMONY.

Table 5 lists the prediction performances of previous studies with 20 newsgroups. The classification accuracy is measured with micro-averaged BEP. The BEPs are obtained by 4-fold cross validation. The computing time of all the classifiers is a single-fold time elapsed during training and testing phase. The result of Naive Bayes is obtained using the Rainbow tool [14]. Recently, the classifiers based

	Harmony (msup=60)	SVM (linear)	BCAR (msup=5)
acq	95.3	93.6	**97.8**
corn	78.2	**90.3**	82.2
crude	85.7	**88.9**	88.1
earn	**98.1**	98.0	97.4
grain	91.8	**94.6**	86.5
interest	77.3	77.7	**83.5**
money-fx	80.5	74.5	**84.4**
ship	86.9	85.6	**92.6**
trade	88.4	75.9	**89.8**
wheat	62.8	**91.8**	79.9
Total	92.0	92.0	**93.5**

Table 3. Classification performance of Reuters-21578

	Harmony (msup=60)	BCAR (msup=20)	BCAR (msup=5)
BEP	92.0	92.8	93.5
time(sec)	73	99	333

Table 4. Computing time of Reuters-21578

on SVM have shown *state-of-the-art* results for text categorizations. The result of SVM-1 in the third column is from [3]. They conducted a feature selection based on clustering of the words appeared in the corpus. SVM-2 [25] is a hierarchical model of base SVM classifiers constructed with the 3-level hierarchy of the 20 categories. Except for SVM-1, the rest three classifies used the same feature set for the training and testing procedures.

BCAR of Table 5 is trained with very low support and confidence thresholds: 3 and 5% respectively. We limit the order of the rules k to 2. Let us consider the complexity of generating association rules. It grows exponentially to the number of words in the vocabulary. Thus, if we generated high-order association rules, then the computational complexity would be unbearable. Fortunately, the classification performance did not improve anymore for the orders higher than two. The value of BEP 90.5 is the best ever reported among the results for 20 newsgroups. In addition, the computing time has also decreased compared with that of SVM-2.

In Table 6, we compare the performances of the classifiers which have been tested on OHSUMED. The performance is measured by the F1 averaged for all the class labels. Yang [23] conducted several classification experiments with various kinds of classifiers on OHSUMED. Among them, we have put the result of Linear Least Square Fit (LLSF) classifier in the first column. Yoon et al. [25] reported a better classification result using the hierarchical SVM classifier. The weighing on prediction scores in (6) is very important to OHSUMED where the number of training examples is unevenly distributed with respect to the categories; the third column that applies no weighting shows a poorer performance than the weighted one, the fourth column.

	Naive Bayes	SVM-1 *clustering*	SVM-2 *hier.*	BCAR
BEP	83.2	88.6	89.0	90.54
time	8.3 mns	4 hrs	5.3 hrs	4.9 hrs

Table 5. Classification performance of 20 newsgroups

LLSF	SVM *hierarchical*	BCAR	
		w/o weight	*w/ weight*
55	58.7	53.2	61.6

Table 6. Classification performance of OHSUMED

7 Conclusion

Our approach is different from others in that it generates as many rules as possible including the rules whose prediction accuracy is moderate or even worse than random guessing. We proposed a new selection method that filters rules on the principle of Boosting. In addition, the new scheme for multi-label classification was provided based on score-ratio thresholding. By many experiments with representative test collections, we showed that our approach achieves excellent classification performance in the large-scale sparse data such as text corpora.

We need to decrease the number of class association rules to lessen the learning time while keeping the classification performance the same as the original one. In addition, the parameter setting in the associative classification depends deeply on the distribution of the training database. For further study it is worth investigating which aspects of the distribution affect the performance.

Acknowledgments

This research was supported by the MKE(Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute for Information Technology Advancement) [IITA-2008-C1090-0801-0045]

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[2] C. Apté, F. Damerau, and S. M. Weiss. Towards language independent automated learning of text categorisation models. In *SIGIR*, pages 23–30, 1994.

[3] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. On feature distributional clustering for text categorization. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 146–153, New Orleans, US, 2001. ACM Press, New York, US.

[4] K. Church and W. Gale. Poisson mixtures, 1995.

[5] S. T. Dumais, J. C. Platt, D. Hecherman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM*, pages 148–155, 1998.

[6] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

[7] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In W. Chen, J. Naughton, and P. A. Bernstein, editors, *2000 ACM SIGMOD Intl. Conference on Management of Data*, pages 1–12. ACM Press, 05 2000.

[8] J. Itskevitch. Automatic hierarchical e-mail classification using association rules, 2001.

[9] M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. The MIT Press, August 1994.

[10] K. Lang. NewsWeeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995.

[11] W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *ICDM*, pages 369–376, 2001.

[12] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[13] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification, 1998.

[14] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

[15] D. Meretakis, D. Fragoudis, H. Lu, and S. Likothanassis. Scalable association-based text classification. In A. Agah, J. Callan, and E. Rundensteiner, editors, *Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management*, pages 373–374, McLean, US, 2000. ACM Press, New York, US.

[16] M. E. Ruiz and P. Srinivasan. Hierarchical text categorization using neural networks. *Information Retrieval*, 5(1):87–118, 2002.

[17] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5:197–227, 1990.

[18] R. E. Schapire, Y. Freund, P. Barlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In *ICML*, pages 322–330, 1997.

[19] K.-M. Schneider. On word frequency information and negative evidence in naive bayes text classification. In J. L. V. González, P. Martínez-Barco, R. Muñoz, and M. Saiz-Noeda, editors, *EsTAL*, volume 3230 of *Lecture Notes in Computer Science*, pages 474–486. Springer, 2004.

[20] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[21] J. Wang and G. Karypis. Harmony: Efficiently mining the best rules for classification. In *SDM*, 2005.

[22] A. K. C. Wong and Y. Wang. High-order pattern discovery from discrete-valued data. *IEEE Transactions on Knowledge and Data Engineering*, 9(6):877–893, 1997.

[23] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2):69–90, 1999.

[24] Y. Yang. A study of thresholding strategies for text categorization. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 137–145, New York, NY, USA, 2001. ACM Press.

[25] Y. Yoon, C. Lee, and G. G. Lee. An effective procedure for constructing a hierarchical text classification system. *JASIST*, 57(3):431–442, 2006.