

Using Clustering to Boost Text Classification

Y.C. Fang, S. Parthasarathy, F. Schwartz
Ohio State University
Contact: srini@cis.ohio-state.edu

Abstract

In recent years we have seen a tremendous growth in the number of text document collections available on the Internet. Automatic text categorization, the process of assigning unseen documents to user-defined categories, is an important task that can help in the organization and querying of such collections. In this article we consider the problem of classifying online papers from a specific journal in the geological sciences, over a set of expert defined categories. We evaluate two general strategies and several variants thereof. The first strategy is based on Naïve Bayes, a popular text classification algorithm. The second strategy is based on Principle Direction Divisive Partitioning, an unsupervised document clustering algorithm. While the performance of both approaches is quite good, some of the new variants that we propose including one, which involves a combination of these two approaches yield even better overall performance..

1. Introduction

In recent years, the amount of online text data has grown tremendously due to the popularity of the Internet and the World Wide Web. As a result, there is an overriding need to provide effective content-based text retrieval, search and querying capabilities. In this paper we consider the problem of automatic text categorization, a means by which one can rapidly enable such retrieval and querying capabilities on a large scale.

Automated text categorization [17, 18, 5] is a supervised learning task, involving the assigning of category labels (pre-defined) to new documents based on the information learned from a labeled training data. The labeled training data involves manual categorization of the data by subject experts. A large number of learning approaches have been brought to bear on this problem domain including, statistical methods [1], Bayesian learning [2, 4, 10, 11, 5, 12, 13], decision trees [3, 14, 4, 10, 5], neural networks [15, 16] and support vector machines [5].

In this paper we evaluate the performance of two such methods for the automatic categorization of articles published by an international journal in the geological sciences. The first method we consider is the Naïve Bayes (NB) method, a popular categorization method. Not surprisingly, we observe that when the domain expert, in addition to labeling duties also provides a list of keywords and associated weighting the performance of this algorithm is greatly enhanced. The second method we consider is Principle Direction Divisive Partitioning (PDDP), an unsupervised text clustering approach based on principal components analysis. Basically in this approach we cluster both the labeled and unlabeled data together and let the labeled documents within each cluster determine the labels of the unlabeled documents. For the dataset we considered

the PDDP-based approaches outperformed the NB-based approaches in terms of the overall quality of performance (as measured by classification or categorization error). However, when taking into account execution time the NB-based approaches were much faster. We then considered a method that combines both the NB-based approach and the PDDP approach. The new approach uses PDDP to cluster the training data to help the method identify pure nodes (representing pure clusters). The text documents that are clustered in the pure nodes are then used to seed the Naïve Bayes approach. We find that the overall quality of this hybrid approach improves the quality of the results as compared with the baseline NB-based approaches. This is attributed to the minimization of noise in the training data by using only the documents from the pure nodes to train the classifier. Also since the PDDP clustering is applied only once and that to on a small portion of the entire dataset (the training data) the hybrid approach also does not suffer from poor performance and is therefore much more scalable than the baseline PDDP-based approaches. We also evaluate the performance of these methods and their variants on papers that are considered to be a mixture of categories.

The rest of this article is organized as follows. Sections 2 and 3 describe the Naïve Bayes and the PDDP algorithms respectively. In Section 4 we describe the different classifiers we evaluated along with the experimental results. In section 5 we discuss the results obtained and some additional features of our work (visualization etc.). We also outline directions for future work in this section.

2. Naïve Bayes Classifier

The basic idea in Naïve Bayes methods is to use the joint probabilities of words and categories to estimate the probabilities of categories given a document. The naïve aspect of the method has to do with the fact that the dependencies between words are ignored, i.e. the conditional probability of a word given a category is assumed to be independent from the conditional probabilities of other words given that category. The assumption is necessary for efficiency reasons. We next describe this approach more formally using the notation from [6].

Assume that each text document is described by a conjunction of the English words (a_i) it contains. Let v_j be the set of user-defined categories. A set of labeled (training) documents in every category is provided to the classifier. After the training phase (described below) when a new document is presented, described by the tuple of its words $\langle a_1, a_2 \dots a_n \rangle$, the learner is asked to predict the category for this new document. The Bayesian approach to classifying the new document is to assign the most probable category it fits in, v_{MAP} -- the maximum a posteriori (MAP) hypothesis, given the words $\langle a_1, a_2 \dots a_n \rangle$ contained in the document.

$$v_{MAP} = \operatorname{argmax} P(v_j | a_1, a_2 \dots a_n)$$

We can use Bayes theorem to rewrite this expression as

$$v_{MAP} = \operatorname{argmax} P(a_1, a_2 \dots a_n | v_j) P(v_j) \tag{1}$$

Estimating the different $P(a_1, a_2 \dots a_n | v_j)$ terms in this fashion is not feasible unless there is an exceptionally large set of training data. The problem is that the number of these terms is equal to the number of possible documents times the number of possible categories. Therefore, we need to see every document in the training set many times in order to obtain reliable estimates. The Naïve Bayes classifier is based on the simplifying assumption that the contained words are conditionally independent in the category. This assumption reduces equation 1 to equation 2.

$$v_{NB} = \operatorname{argmax} P(v_j) \prod_i P(a_i | v_j) \quad (2)$$

Here v_{NB} denotes the probability output by naïve Bayes classifier. Notice that with the naïve Bayes classifier the number of distinct $P(a_i | v_j)$ terms that must be estimated from the training data is just the number of distinct words times the number of distinct categories – a much smaller number than if we were to estimate the $P(a_1, a_2 \dots a_n | v_j)$ terms first contemplated.

Whenever the naïve Bayes assumption of conditional independence is satisfied, this naïve Bayes classification v_{NB} is identical to the MAP classification v_{MAP} obtained from equation (1). Even when this assumption is not met, as in the case of learning to classify text, the naïve Bayes classifier is often quite effective [7]. Completing the design of the learning algorithm requires choosing a method for estimating the probability $P(a_i | v_j)$ terms. We adopt the m -estimate [8] with uniform priors and with m equal to the size of the word vocabulary. Thus, the estimate for $P(a_i | v_j)$ will be $(n_k + 1) / (n + |\text{Vocabulary}|)$, where n is the total number of word positions in all training examples whose category is v_j , n_k is the number of times word a_i is found among these n word positions, and $|\text{Vocabulary}|$ is the total number of distinct words (and other tokens) found within training data [6].

We adopt the implementation of the algorithm proposed in [9]. In addition to the base implementation we also implement a variant that allows us to embed domain specific knowledge within the classifier. This domain knowledge is embedded in terms of keywords associated with a particular category and weights associated with each keyword. Essentially, instead of using the pre-classified papers as training data, we build our own collections of words with different weights based on the domain expert's experience with the data. Note, that the Naïve Bayes classifier needs several training papers for each topic to classify an unknown paper. The classifier is dependent on the quality of papers in the training set. Instead of using the entire paper if one can instead train the classifier based on a bag of words deemed important by an expert, the classifier is less likely to be affected by noise effects.

3. PDDP (Principal Direction Divisive Partitioning)

PDDP is an unsupervised technique that clusters together related documents. It constructs a binary tree, in which each node is a data structure holding the documents associated with that node, the various quantities computed from that set of documents, and the pointers to the two children nodes. The information stored in each node consists of the documents in the cluster associated with that node, the centroid (mean vector), the

leading singular value and associated singular vector, and the pointers to the children nodes. The “scatter” value is also stored in the node, which will be discussed later.

The PDDP tree starts with the root cluster representing all documents. The algorithm then recursively splits each leaf cluster into two children until some criterion is satisfied. This partitioning constructs a binary tree, the PDDP tree. Each document is represented by an n -vector d of word counts. Each n -vector is scaled so that $\|d\| = 1$ to ensure that the values are independent of document length. The vectors of all the m documents to be clustered are assembled into term frequency matrix $M(n \times m) = (d_1 \dots d_m)$. This term frequency matrix along with the mean vector (centroid) is used to obtain the principle direction and the hyper-plane partition that is used to split the documents within a given node into two partitions. To decide at each stage which node should be split next, one choice is to try to keep the binary tree balanced by splitting all the nodes at a given level before proceeding to the next level. However, by doing so the resulting clusters are often imbalanced with a few large clusters and many small clusters, even singletons. PDDP uses the notion of “scatter” to determine the best node to split. The scatter value is simply a measure of how cohesive the documents within a cluster are [19, 20, 21].

The algorithm chooses the cluster with the largest scatter value to split. The total scatter value represents the distance between each document in the cluster and the overall mean of the cluster. Using the total scatter value to choose the next cluster to be split usually results in clusters all having more or less similar numbers of documents. This scatter value is the only component of this algorithm based on “distance” measure. It would be possible to use other measures not based on a “distance” measure and appropriate for particular data sets [19]. At each pass through the main loop, a node is selected based on total scatter value, and the mean vector and principal direction are obtained for the documents associated with that node. Then the collected mean vector and principal vector are used to split the node into two child nodes. In a geometric sense, the process is splitting the documents using hyperplane normal to the principal direction passing through the mean vector.

The execution time of the algorithm depends on the size of the term frequency matrix. Much like we did for the Naïve Bayes approach we have implemented a variant of the base PDDP algorithm where instead of using all the distinct words in the documents to build the term frequency matrix we limit the words to the set of words chosen by domain experts for the problem on hand. This variant reduces the execution time of the algorithm significantly (since the size of the term frequency matrix is reduced) without affecting the quality of results obtained by much as we shall see in the next section.

4. Experiments and Results

The data we used is from a key journal in the water sciences called “*Water Resources Research*”. We evaluated articles over 7 years, i.e., from 1990-1996. There were basically five major categories of water sciences research (“precipitation”, “unsaturated”, “groundwater”, “river-lake”, and “estuary-ocean”) amongst these articles along with several articles that involve a mixture of research topics. Unless otherwise stated we used a training set of 116 papers (roughly an equal number from each category) and a testing set of 233 papers to evaluate classifier performance. Of the 233 papers in our testing set

the domain experts identified 51 papers as hard to classify (this is also borne out in our classification results). We report classification accuracies over the entire test dataset as well as over this hard subset of test papers. We evaluated the following classifiers:

1. NB0 (Naïve Bayes): This classifier served as a baseline against which we evaluate the other NB variants. This is the classical approach that uses a training set of documents and evaluates the classifier on an independent testing set. While the overall performance of this classifier was not bad (81% accuracy), only 13 out of 51 papers (25% accuracy) from “hard” set were correctly classified.

2. NB1 (Naïve Bayes with experts’ chosen keywords and weighting)

The second classifier we evaluated is the variant described in Section 2. Instead of sending in the pre-labeled chosen papers as training set, we used experts’ chosen keywords and associated weights to seed the classifier (see Figure 1). In this experiment, the memory needed for storing vocabularies (128 words) and running time is much smaller and more efficient than using training papers (116 pre-labeled training papers created more than 13,000 words). The overall accuracy of this classifier was 87% and the accuracy on the hard subset was 39% (20/51).

3. NB2 (Naïve Bayes with experts’ chosen keywords without weights)

Essentially the same as NB1 except that no weight was assigned to each word (Figure 1). The primary purpose of evaluating this classifier is to quantify the extra benefit (if any) of weighting and to evaluate its importance within the context of the NB approaches. The results showed that the overall accuracy was around the same as the baseline (NB0) and the accuracy on the hard subset dropped from 39% (NB1) to 20% (10 out of 51 papers from testing set). This result underscores the importance of weighting on classification performance.

groundwater X 10	groundwater
saturated X 10	saturated
phreatic X 10	phreatic
aquifer X 5	aquifer
head X 1	head
heads X 1	heads
transmissivity X 5	transmissivity
storage X 1	storage
net X 1	net
pathline X 1	pathline
equipotential X 1	equipotential
potentiometric X 1	potentiometric
table X 1	table
line X 1	line
piezometer X 1	piezometer
well X 1	well
subsurface X 1	subsurface
recharge X 1	recharge
discharge X 1	discharge
confining X 1	confining
confined X 1	confined
unconfined X 1	unconfined
bed X 1	bed
darcy X 1	darcy
darcy's X 1	darcy's

Figure 1. Bag of words (for “groundwater” category) selected by experts. Weights assigned (X #) (left); no weights (right)

4. NB3 (Naïve Bayes with experts' chosen keywords with distribution weighting)

In this variant we used the distribution of the expert-provided keywords in the various categories to determine the weights of the words. This variant was motivated by the fact that some words such as “precipitation”, and “modeling” seemed relevant for more than one category but they also seemed to have largely different frequency distributions for different categories. The frequency of occurrence also seems to depend on the contextual interpretation of these words. For instance, “precipitation”, meaning rainfall in the “precipitation category”, and having a chemical connotation in the “groundwater category” had a greater frequency of occurrence in the “precipitation category” when compared with the “groundwater category”. There was a marginal improvement in the classification accuracy when evaluated on the hard subset (increased from 39% (NB1) to 41% (21 out of 51 papers from testing set)).

5. PDDP1 (with experts' chosen keywords)

Experts' chosen keywords were used to build the term frequency matrix. The PDDP tree was built based on this term frequency matrix. As mentioned earlier, the storage requirements and execution time of this algorithm is much less than the storage requirements and execution time of an algorithm that used all the words from the training set. Since this essentially a clustering algorithm we grouped the test and training data together and grew the PDDP tree. To compute the classification accuracy for each leaf cluster we looked at the labeled documents that were in that cluster and classified all the documents in that cluster based on the dominant class label. The overall accuracy of this algorithm compares with the best of the NB approaches at around 87%. What is very interesting is the fact that the accuracy on the hard subset (57%) is significantly better than the accuracy reported by the best of the NB approaches (39%). The generated tree can be viewed at <http://128.146.67.134/~fang/Tree17.html>.

6. PDDP2 (with all words from the training set on testing set of 223 papers)

In this experiment, we used all words collected from training set (116 pre-labeled documents) except words like, “I”, “and”, “the”, etc., in order to build the term-frequency matrix. As can be seen here in some of the leaf nodes, the results are different than PDDP1; some papers clustered together in PDDP1 were not clustered together in PDDP2. In this variant no supervision was required in word selection. The overall accuracy of this algorithm is the best of all the classifiers we evaluated at 90%. Like PDDP1 the accuracy on the hard subset is significantly better than the accuracy reported by the NB approaches. The generated tree can be viewed at <http://128.146.67.134/~fang/Tree18.html>

7. NB4 (Naïve Bayes with PDDP words and weighting)

In this experiment, we wanted to evaluate if we could improve the performance of Naïve Bayes by bootstrapping information derived from running PDDP on the *training data*.

Essentially by clustering the training data we hoped to minimize noise effects in the training procedure by only using those documents that were part of pure clusters. Basically 12 distinctive words were chosen from the principal direction vectors of *pure* nodes in a PDDP tree generated from only the training data. A *pure* node is one which is dominated by one of the five categories. Weights were assigned to these distinctive words according to their relative frequencies (already calculated by PDDP). Using these bags of collected words from PDDP tree and associated weights we embedded this information into the Naïve Bayes classifier (as in NB1). The overall performance of the classifier was 85% a clear improvement on the baseline (NB0). Its accuracy on the hard subset was 33% (17/51) again an improvement over NB0. This classifier did slightly worse compared to NB1 but one has to remember that there was no supervision required in terms of providing keywords and weights associated with categories. This experiment highlights the fact that integrating PDDP with NB boosts performance without any additional supervision being required.

	NB0	NB1	NB2	NB3	NB4
Training Words	Training set (116 papers)	Expert's words (with weights)	Expert's words (no weights)	Expert's words (w/dist. weight)	PDDP words (w/weights)
Accuracy	13/51 = 25%	20/51 = 39%	10/51 = 20%	21/51 = 41%	17/51 = 33%
Testing set	233 papers	233 papers	233 papers	233 papers	233 papers
Overall Perform.	82%	87%	82%	87%	85%

Table 1. Summary of the Naïve Bayes classifiers.

	PDDP1	PDDP2
Words	Expert's words (128)	Words from training set (13372)
Documents	116+233 papers	116+233 papers
Term.Freq.Matrix	349 X 128	349 X 13372
Accuracy	29/51 = 57%	31/51 = 61%
Overall Perform.	87%	90%

Table 2. Summary of the PDDP tree Experiments.

5. Discussion

Tables 1 and 2 summarize the results presented in the previous section. While the results presented in the previous section were over one training and testing set, these results have been cross validated. Of the Naïve Bayes variants NB1 and NB3 performed the best. Clearly if one can embed the necessary domain knowledge (keywords and associated weights) classification performance improves. Note, however that our results (performance of NB2) showed that just identifying keywords alone is not enough, the associated weights are crucial to this performance gain. Another crucial advantage of NB1 and NB3 over NB0 is that these classifiers are space and time-efficient to build.

However, one drawback of NB1 and NB3 is the extra supervision required to identify keywords and associated weights. If either this domain knowledge is unavailable or difficult to quantify and characterize then the best option seems to be NB4 which requires no additional supervision (when compared with NB0). *NB4 combines the best of both worlds, it has the efficiency of NB1 and NB3, closely approximates the accuracy of NB1 and NB3 and does not require any more supervision than NB0.* Again as we pointed out earlier this gain in accuracy is due to minimization of noise effects (a result of clustering the training data and using only information from documents that are a part of pure nodes).

In the PDDP tree experiments we evaluated the impact of using the keywords suggested by the domain expert to generate the word frequency matrix. We compared this against the baseline approach where words are collected from all root documents. In the former case the running time and storage requirements are reduced due to the size of the word frequency matrix. While the execution times are significantly lower, the overall performance (classifier accuracy) is also lower when using the reduced word set. Of all the classifiers evaluated the classification accuracy of PDDP2 was the best at 90%. However, one should be sensitive to the crucial issue of *scalability*. As more and more papers are added to the problem domain, the classification efficiency of the PDDP-based methods is going to deteriorate much more rapidly than the NB methods (this is true even for NB4 since PDDP is applied only on the training data). Another problem with using the PDDP-based methods in their current form, is that after running the above experiment if one is to classify another paper, one has to re-run the algorithm from scratch. Of course one may be able to modify the PDDP algorithm to identify which hyper-plane partition a paper belongs to and therefore be able to classify it. This would require some modifications to the existing PDDP implementation. *Because of these limitations NB4 might still be preferred when compared with PDDP-based approaches for large scale text categorization problems.*

Since the training set was pre-labeled and the PDDP tree was generated in HTML format, we added a coloring scheme to the labeled data better visualize and evaluate the performance of the tree. Different colors were used for different categories. On viewing the resulting PDDP tree one can immediately identify the cohesive or pure nodes by the uniformity in color of the labeled data (Figure 2).

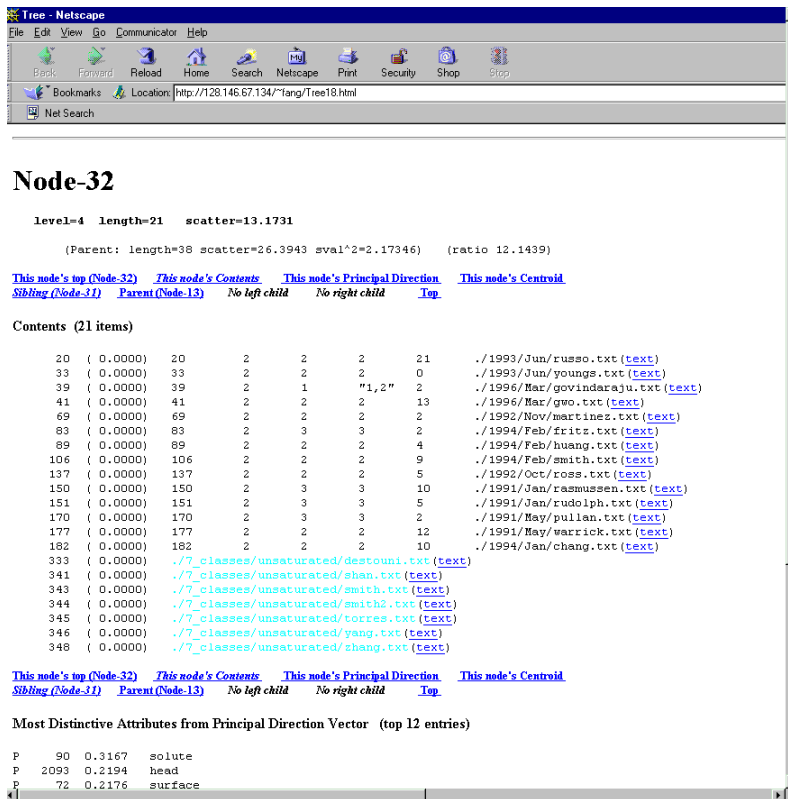


Figure 2: Screen shot of Node 32 in Tree18 (PDDP2). Use of color to highlight pure nodes.

We also evaluated the performance of the classifiers on papers that were considered to involve a mixture of research areas. 14 such combined-topic papers were selected and evaluated. The NB classifiers basically identified one of the categories but could not identify the paper as a mixture paper (the probabilities for the second paper was infinitesimal). The PDDP results were slightly better in the sense that the mixture papers were often categorized in nodes which involve papers from both categories. However, the notion of mixture is hard to capture within the PDDP context. This is certainly an area that needs further investigation.

As part of ongoing work we are currently carrying out scalability measurements to fully characterize and understand the various tradeoffs between the different methods. We plan to incorporate these results in a future version of this document. We are also interested in exploring other text categorization methods for our domain. In addition we also plan to explore the impact of word associations (i.e. moving away from the Naïve Bayes assumption), perhaps via association rule mining [22], on classifier performance. Another interesting avenue of future research that we may consider is the use of semantic information (say from WordNet) to improve classifier performance.

References

- [1] Y. Yang. An evaluation of statistical approaches to text categorization. *Journal of information Retrieval*, 1(1/2):67—88, 1999.
- [2] K. Tzeras and S. Hartman. Automatic indexing based on bayesian inference networks. In *Proc 16th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'93)*, pages 22—34, 1993.
- [3] C. Apte, F. Damerau, and S. Weiss. Text mining with decision rules and decision trees. In *Proceedings of the Conference on Automated Learning and Discovery, Workshop 6: Learning from Text and the Web*, 1998.
- [4] D. D. Lewis and M. Ringuette. Comparison of two learning algorithms for text categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, 1994.
- [5] Thorsten Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *European Conference on Machine Learning (ECML)*, pages 137— 142, Berlin, 1998. Springer.
- [6] Tom Mitchell. *Machine Learning*. McGraw Hill, 1996.
- [7] Domingos, P., and Pazzani, M. 1996. Beyond independence: Conditions for optimality of the simple Bayesian classifier. *Proceedings of the 13th International Conference on Machine Learning* (pp. 105-112).
- [8] Cestnik, B. 1990. Estimating probabilities: A crucial task in machine learning. *Proceedings of the Ninth European Conference on Artificial Intelligence* (pp. 147-149). London: Pitman.
- [9] McCallum, A., 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. <http://www.cs.cmu.edu/~mccallum/bow>. 1996.
- [10] I. Moulinier. Is learning bias an issue on the text categorization problem? In *Technical report, LAFORIA-LIP6, Universite Paris VI*, 1997.
- [11] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *The Fourteenth International Conference on Machine Learning (ICML'97)*, pages 170—178, 1997.
- [12] A. McCallum and K. Nigan. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

- [13] L. Douglas Baker and Andrew K. McCallum. Distributional clustering of words for text categorization. In *Proceedings of the 21th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)* pages 96—103, 1998.
- [14] N. Fuhr, S. Hartmann, G. Lustig, M. Schwantner, and K. Tzeras. Air/x - a rule-based multistage indexing systems for large subject fields. In 606-623, editor, *Proceedings of RIAQ'91*, 1991.
- [15] E. Wiener, J.O. Pedersen, and A.S. Weigend. A neural network approach to topic spotting. In *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retriever (SDAIR'95)*, pages 317—332, Nevada, Las Vegas, 1995. University of Nevada, Las Vegas.
- [16] H.T. Ng, W.B. Gob, and K.L. Low. Feature selection, perceptron learning, and a usability case study for text categorization. In *20th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97)*, pages 67—73, 1997.
- [17] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.
- [18] Y. Yang and J. Pederson. A comparative study on feature selection in text categorization. In *Proc. of the Fourteenth International Conference on Machine Learning*, 1997.
- [19] Principal Direction Divisive Partitioning, D. L. Boley, *Data Mining and Knowledge Discovery* 2(4):325-344 , 1998.
- [20] Unsupervised Clustering: A Fast Scalable Method for Large Datasets, by D. Boley and V. Borst, U of Mn CSE Report TR-99-029. *An informal introduction to the methods*.
- [21] Partitioning-Based Clustering for Web Document Categorization, D. Boley, M. Gini, R. Gross, S. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, J. Moore, *Decision Support Systems*, 1999.
- [22] R. Agrawal, R. Srikant: "Fast Algorithms for Mining Association Rules", Proc. of the 20th Int'l Conference on Very Large Databases, Santiago, Chile, Sept. 1994.