IJCSC International Journal of Computer Science & Communication Vol. 1, No. 1, January-June 2010, pp. 51-56

# Improving performance of Text Categorization: Using Multi Feature Coselection Clustering Technique and Lsquare Machine Learning

Srinivas. M[1], K.P.Supreethi[2] & S. Anitha Kumari[3]

[1]Lecturer in CSE Dept, JNTUCE, Anantapur

[2]Assistant professor, CSE Deptt. JNTUHCE, Anantapur,

[3]Assistant professor, CSE Dept, vardhaman CE, HYD

[1]sreenu2521@gmail.com, supreethi.pujari@gmail.com[2] & anitha.mahi7@gmail.com[3]

ABSTRACT

Text categorization is continuing to be one of the most researched NLP problems due to the ever-increasing amounts of electronic documents and digital libraries. In this paper, we present a novel text categorization method that combines the Multitype Features Coselection for Clustering and a learning logic technique, called Lsquare, for constructing text classifiers. The high dimensionality of text in a document has not been fruitful for the task of categorization, for which reason, feature clustering has been proven to be an ideal alternative to feature selection for reducing the dimensionality. We, therefore, use Multitype Features Coselection for Clustering (MFCC) to generate an efficient representation of documents and apply Lsquare for training text classifiers. The method was extensively tested and evaluated. The proposed method achieves higher or comparable classification accuracy and F1 results compared with SVM. MFCC improves clustering performance.

## 1. INTRODUCTION

Text Categorization (TC) is the task of assigning a given text document to one or more predefined categories. This problem has received a special and increased attention from researchers in the past few decades due to many reasons like the gigantic amount of digital and online documents that are easily accessible and the increased demand to organize and retrieve these documents efficiently. Efficient text categorization systems are beneficial for many applications. In this paper, we explore the application of a learning technique, called Lsquare [5], cluster with the Multitype Features Coselection for Clustering [4] into the text categorization problem.

Lsquare is a logic-based supervised machine learning algorithm [5]. Lsquare extracts certain logic relationships from the training data and deduces those logic formulas that can correctly classify a given (testing) data set. The main contributions of this paper are as follows:

1. The paper proposes a new TC technique based on an existing state-of-the-art feature clustering technique and a logic based learning algorithm (Lsquare).
2. The devised TC technique outperforms SVM.
3. The devised TC technique accepts only binary features (based on learning-logic), also, this aspect gives it an advantage in computational time complexity.

The rest of this paper is organized as follows: Next, we describe prior related work. Section 3 describes the learning in Lsquare. Section 4 discusses text categorization by Lsquare. Then, in Section 5, the experiments and evaluation results are explained and discussed. Finally, Section 6 discusses the conclusions and future work.

## 2. RELATED WORK

In this section, we briefly discuss the related research in text categorization and, for more details, we will refer to a certain publication, [16]. SVMs have been prominently used for text categorization (Joachims [10] & Dumais et al. [18]) using the bag-of-words model. Decision tree learners attempt to select from training data some informative words using an information gain criterion, then predict the category of a document based on the occurrence of word combinations. While traditional machine learners employ attribute value representations, the use of logic programming representations led to the establishment of ILP [6]. The effectiveness of ILP methods for TC lies in formulating classifiers based on word order. Several techniques have been proposed for feature selection and dimensionality reduction In Distributional clustering of words, words are represented as

distributions over categories of the documents where they appear [13]. Using a naive Bayes classifier, Baker and McCallum [1] applied the distributional clustering scheme of Pereira et al. [13] for text categorization. Multitype Features Coselection for Clustering method used for clustering the text documents. Both Baker and McCallum [1] and Slonim and Tishby [17] used agglomerative clustering algorithms and using Naive Bayes showed that the feature size is greatly reduced by using distributional clustering without significant loss of categorization accuracy. Other methods such as Latent Semantic indexing (e.g., [9]) have been applied and have been shown to be inferior to feature clustering [1].

## 3. THE PROPOSED METHOD

We propose a new TC method based on a successful feature clustering technique and a logic-based learning algorithm Lsquare. Our approach depends on representing the text document as a projection on clusters formed from the input data set, then applying the Lsquare learner to build text classifiers. Next, we present the MFCC.

### 3.1. Multitype Features Coselection for Clustering

#### *3.1.1. Feature Coselection*

First, it should be made clear that the selection of each type feature is limited in each type of feature, and the clustering is an iterative one. After one iteration of clustering, each data object will be assigned to a cluster. In [19], Liu et al. assumed each cluster corresponded to a real class. Using such information, they did supervised feature selection, such as Information Gain (IG) and $\chi^2$ statistic (CHI) [20] during k-means clustering. MFCC tries to fully exploit heterogeneous features of a Web page like title, URL, anchor text, hyperlink, etc., and to find more discriminative features for unsupervised learning. We first use different types of features to do clustering independently. Then, we get different sets of pseudoclass, which are all used to conduct iterative feature selection (IF) for each feature space.

Then, we get different sets of pseudoclass, which are all used to conduct iterative feature selection (IF) for each feature space. After normal selection, some data fusion methods are used to combine the selection results in each space, i.e., feature coselection. In each iteration of clustering, the coselections in several spaces are conducted one by one after clustering in all feature spaces. That is to say, all intermediate clustering results in different feature spaces have been achieved before any coselection. Thus, the sequence of coselection will not affect the final performance. The general idea of coselection for k-means clustering is described in Fig. 1.

Suppose that we categorize data objects with M heterogeneous features into L clusters. Let $fv_n$ be one dimension of the feature vector, $icr_i$ be the intermediate clustering results in the $i^{th}$ feature space, and SF be a fusion function. The pseudo algorithm is listed as follows:

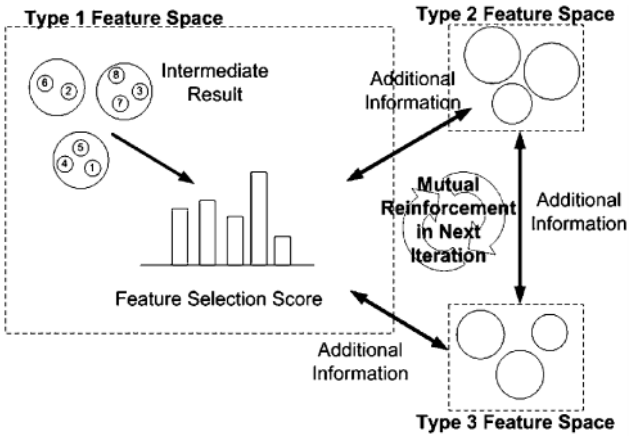


**Fig 1: The Basic Idea of Multitype Feature Coselection**

Suppose that we categorize data objects with M heterogeneous features into L clusters. Let $fv_n$ be one dimension of the feature vector, $icr_i$ be the intermediate clustering results in the $i^{th}$ feature space, and SF be a fusion function. The pseudo algorithm is listed as follows:

*Loop for N iterations of k-means clustering*
*{*
*Loop for M feature spaces*
*{*
*Do clustering in feature space m*
*}*
*Loop for M feature spaces*
*{*

For feature space m, do feature selection using results in all feature spaces. For $fv_n$, one dimension of the feature vector in space m, a feature selection score fss ($fv_n$, $icr_{i)}$ is obtained by using intermediate clustering results icri in feature space i. Then a combined score fss ($fv_n$) is achieved by fusing the scores based on different result sets.

$$fss(fv_n) = SF(fss(fv_n, icr_i)) \quad (1)$$

}
}

In (1), fss ($fv_{n,}$ $icr_i$) can be the value calculated by the selection function or rank among all features. The feature selection criteria we used are introduced in Section c. For SF, voting, average, and max are three available choices. Depending on the choices of fss and SF, we obtain five fusion models including voting, average value, max value, average rank, and max rank. The equations are listed as follows:

$$\text{Voting } (Val\ (fv_n)) = \sum_{i=1}^{M} vote(fv_n, icr_i)$$

$$vote(fv_n, icr_i) = \begin{Bmatrix} o - val(fv_n, icr_i) < st \\ 1 - val(fv_n, icr_i) >= st \end{Bmatrix}$$

$$\text{Average } (Val\ (fv_n)) = \left( \sum_{i=1}^{M} val(fv_n, icr_i) \right) / M$$

$$\text{Max } (Val\ (fv_n)) = \arg\max_{I}^{M}(val(fv_n, icr_i))$$

$$\text{Average Rank (Rank } (fv_n))) = \left( \sum_{i=1}^{M} Rank(fv_n, icr_i) \right) / M$$

$$\text{Max Rank (Rank } (fv_n)) = \arg\max_{I}^{M}(Rank(fv_n, icr_i)) \quad (2)$$

In the above equation, val($fv_n$, $icr_i$) is the value calculated by the selection function, Rank($fv_n$, $icr_i$) is the rank of $fv_n$ in the whole feature list ordered by val ($fv_n$, $icr_i$), and st is the threshold of feature selection. After feature coselection, objects will be reassigned, features will be reselected, and the pseudoclass-based selection score will be recombined in the next iteration. Finally, the iterative clustering and feature coselection are well integrated. In each of the iterations, the whole feature space should be reconsidered. The reason is that our method can help find more effective features through a mutual reinforcement process. Properly selected features will help clustering and vice versa. That is to say, some discriminative features will not be found until late in the clustering phase. This can be verified by some empirical results.

### 3.2. Lsquare

The basic idea of Lsquare is as follows: Lsquare is a two-class classification technique that is based on learning logic. It views the training data as logic formulas and the resulting classifiers are logic formulas as well. The logic formulas are represented as vectors of {0, ±1} entries. Each vector represents one document, while each {0, ±1} entry in the vector represents a term/feature in that document.

The Lsquare system accepts as input two sets A and B of {0, ±1} vectors, all having the same length. Lsquare outputs a set of 20 DNF logic formulas and 20 CNF logic formulas. Each logic formula is capable of classifying any (new) vector to whether it belongs to class A or B. These logic formulas are called a separating set and this separating set will be our text classifier.

## 4. TEXT CATEGORIZATION

Our text categorization approach depends on representing the text document as a projection on word clusters, then applying the Lsquare to build the text classifiers. Since we compare our approach with SVM on the same experimental settings, we include, in this section, some details about SVM and combining SVM with Multitype Features Coselection for Clustering for TC.

### 4.1. Lsquare with Multitype Features Coselection

Lsquare is a binary classifier that operates on a vector space model of documents. The training data are modeled as a collection of vectors, one for each document. Then, applying Lsquare to the training vectors will generate a classifier L. In the experiments, we construct a classifier for each category. For a given category C, the training data consists of two sets of vectors, one set for positive examples and an other for negative examples of that category. And, the sequence of steps of the training phase is outlined in Algorithm 1. We always select an equal number of positive and negative training documents. For example, when the training size is 40 documents, we randomly select 20 positive and 20 negative documents to train the classifiers.

Algorithm 1: Text Classification with Lsquare

The training phase

Input: training dataset D of text documents.

Output: a classifier Li for each category $C_i$.

For each category $C_i$, do the following:

1. Extract from $D$ the documents representing positive examples on $C_i$ call this set $PT_i$ (positive examples are documents labeled with category $C_i$);
2. Randomly select $|PT_i|$ documents from $D$ representing negative examples on $C_i$, call this set $NT_i$.
3. Convert set of documents $PT_i$ to set of vectors $PV_i$ using word clusters as features.
4. Convert set of documents $NT_i$ to set of vectors $NV_i$.
5. Create the training data set $T_i$ for category $C_i$:

$$T_i \leftarrow \{PVi, NVi\}$$

6. Apply Lsquare to $T_i$ to generate classifier $Li$.

One of the issues facing the learning in TC is the imbalanced data problem, that is, the availability of a much larger negative set compared to a positive set size. This occurs when considering all noncategory documents as the negative set, especially when there are a large number of categories. A number of techniques have been proposed to address this imbalanced data problem

(Hearst et al. [8] and Ruiz and Srinivasan [15]). In our method evaluation, we simply select equal amounts of positive and negative training documents, and we repeat each experiment 10 times with different randomly selected positive and negative documents.

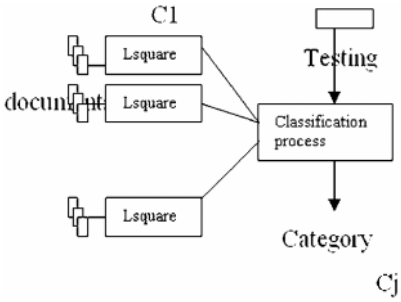


**Fig 2: Text Categorization with Lsquare.**

During the testing phase, all classifiers generated for all categories constitute the testing process and will be used to determine the category label of a given testing document, see Fig. 2. In our experiments and for each generated classifier, we use all the remaining documents which were not used for training as a testing set to evaluate the accuracy of that classifier.

### 4.3. Classifying New Documents with Lsquare

Now, for a given testing document d, we apply d to each classifier Li of each category Ci. Then, if d actually belongs to category Ci and Li classifies it so, we count this as correct; otherwise, it is counted as an error in category $C_i$. In our experiments and for each classifier Li of category $C_i$, we record the a, b, c, and d values as follows:

. a = # of $C_i$ documents that Li classifies into $C_i$.

. b = # of non-Ci documents that Li classifies into $C_i$.

. c = # of non-$C_i$ documents that Li classifies as non- $C_i$.

. d = # of $C_i$ documents that Li classifies as non- $C_i$.

## 5. EXPERIMENTS AND RESULTS

The TC approach proposed in this paper has been fully implemented and evaluated with extensive experimentation; this section presents the details of implementation, data sets, and test results.

### 5.1. Evaluation Methodology

A number of the metrics used in TC are evaluated and measured for categorization effectiveness. We use the well-known precision and recall metrics. We also can define precision and recall in terms of a, b, c, and d values defined above (in Section 4.3) as:

. . . . . $a/(a + b)$ Recall = $a/(a + c)$

Some combination of precision and recall can be more effective in measuring classifier performance. Such measures include F-measure and precision-recall Break Even Point (BEP). F-measure is known by calculating the harmonic mean of precision (P) and recall (R) and F1 is computed as:

$$F1 = 2PR/(P + R)$$

We also use the accuracy in this paper as a measure and the accuracy is computed as the ratio of correctly classified testing documents to the total number of testing documents. Of course, all these performance metrics are computed for each category separately (per category), that is, we apply all the testing documents to each classifier Li (classifier Li is for category Ci) to computer P, R, F1, and accuracy for that category Ci. Then, we calculate the average of all categories.

**Table1**
**Categorization Performance in Terms of Accuracy, BEP, and F1 for Lsquare and SVM Using WebKB on Different Training Sizes**

| Training Size # of docs | Accuracy | | Break-Even Point | | F1 | |
|---|---|---|---|---|---|---|
| | Lsquare | SVM | Lsquare | SVM | Lsquare | SVM |
| 10 | 70.72 | 58.67 | 69.40 | 67.38 | 66.02 | 64.95 |
| 20 | 93.61 | 78.38 | 87.21 | 75.12 | 86.99 | 74.33 |
| 30 | 93.40 | 91.25 | 87.92 | 84.24 | 87.67 | 84.00 |
| 40 | 94.34 | 91.27 | 88.68 | 85.23 | 88.35 | 85.20 |
| 50 | 95.19 | 91.27 | 90.40 | 85.23 | 90.19 | 85.20 |
| 60 | 95.55 | 91.44 | 91.36 | 85.63 | 91.21 | 85.62 |
| 80 | 97.27 | 93.65 | 93.95 | 88.30 | 93.93 | 88.30 |
| 100 | 96.80 | 94.59 | 93.20 | 88.69 | 93.14 | 88.69 |
| 120 | 97.32 | 94.33 | 94.10 | 89.34 | 94.06 | 89.27 |
| 150 | 97.82 | 96.17 | 94.69 | 91.93 | 94.67 | 91.93 |
| 200 | 97.88 | 96.05 | 94.79 | 91.42 | 94.78 | 91.41 |
| Macroaverage | 93.63 | 87.99 | 89.61 | 84.16 | 89.18 | 83.80 |
| Microaverage | 96.55 | 93.18 | 92.92 | 88.34 | 92.81 | 88.25 |

### 5.2. Data Sets

The experimental evaluation was performed on three well-known data sets [12] in text categorization research WebKB, 20 Newsgroup (20NG), and Reuters-21578. Here we can explain and shows the results on only 20NG Dataset. The 20 Newsgroups (20NG) corpus contains almost 20,000 articles taken from the Usenet newsgroups [13]. These articles are evenly distributed on 20 categories; actually, each category is a discussion group in this newsgroup. Furthermore, each article is assigned into one or more categories. Only less than 5 percent of the articles in this set belong to more than one category. In our experiments, following other TC projects, we ran a test on the 10 most populated categories (top 10), which are the 10 categories having highest number of documents.

While, for more complex data sets such as 20NG, even low frequency words have an effect on the categorization results [4].We used different training sizes

ranging from 10 examples to 200 examples to train text classifiers. For training size n, to generate a classifier for category C, n/2 positive examples (documents from the category C) and an equal number (~n/2) of negative examples from categories other than C are used. Then, we test the classifier on the remaining documents not used for training from all categories. Reuters-21578 is already split into training and testing sets, so we select the training documents from the documents marked for training in the ModApt split. For 20NG and WebKB, there is no such split; the training documents are first selected randomly and the remaining documents are used as testing documents. Here we can shows results only on 20NG.

**Table 2**
**Performance Results of Lsquare and SVM Using 20NG on Different Training Sizes**

| Training Size (# of documents) | Accuracy | | Break-Even Point | | F1 | |
|---|---|---|---|---|---|---|
| | Lsquare | SVM | Lsquare | SVM | Lsquare | SVM |
| 10 | 86.83 | 78.73 | 66.09 | 56.87 | 57.33 | 52.04 |
| 20 | 96.14 | 92.22 | 81.51 | 67.28 | 79.10 | 66.12 |
| 30 | 96.98 | 92.11 | 85.00 | 72.05 | 83.15 | 71.55 |
| 40 | 97.92 | 94.79 | 87.68 | 76.86 | 86.44 | 76.19 |
| 50 | 97.33 | 96.26 | 86.12 | 79.70 | 84.43 | 79.43 |
| 60 | 98.11 | 97.37 | 87.55 | 80.07 | 86.32 | 80.02 |
| 80 | 98.18 | 96.88 | 87.99 | 83.96 | 86.80 | 83.80 |
| 100 | 98.22 | 98.43 | 87.97 | 85.78 | 86.82 | 85.76 |
| 120 | 98.27 | 98.33 | 88.47 | 86.67 | 87.26 | 86.61 |
| 150 | 98.44 | 98.60 | 89.23 | 88.37 | 88.19 | 88.28 |
| 200 | 98.35 | 98.77 | 88.83 | 90.26 | 87.68 | 90.26 |
| **Macroaverage** | **96.80** | **94.77** | **85.13** | **78.90** | **83.05** | **78.19** |
| **Microaverage** | **98.00** | **97.42** | **87.80** | **84.82** | **86.45** | **84.62** |

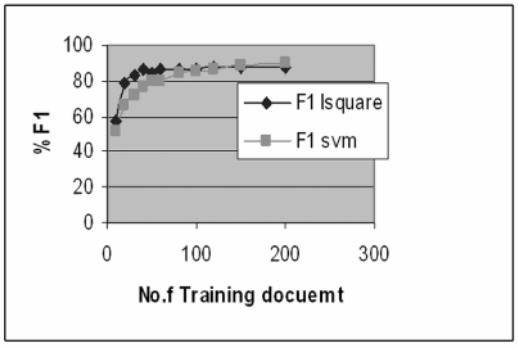


**Fig 3: F1 Results of Lsquare & SVM on the 20NG.**

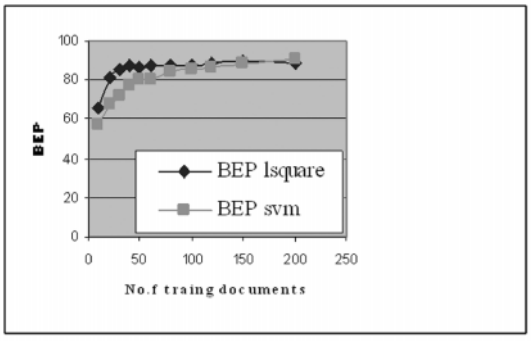


**Fig 4: BEP results of Lsquare and SVM on the 20NG.**

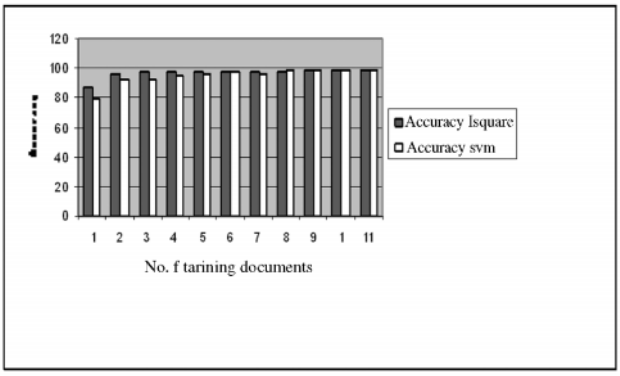


**Fig 5: Accuracy Results of Lsquare and SVM on the 20NG**

The 20NG data set to evaluate Lsquare and SVM on the same numbers of training examples. Table 3 shows the Accuracy, Break-Even Point (BEP), and F1 results of both Lsquare and SVM. The graphical illustration of F1 results is in Fig. 3. These results show that Lsquare can perform better than SVM on most training sizes. The macroaveraged and microaveraged accuracy, BEP, and F1 of Lsquare are significantly better than those of SVM. We also notice that Lsquare outperforms SVM on smaller training sizes and both have comparable performances on larger training sizes. This suggests that Lsquare is more effective when there is a small number of training examples. Thus, Table 2 suggests that our method can be a good choice when we have limited amount of labeled training data. In another set of tests on 20NG, to evaluate our method and to find the effect of the changing number of categories on the performance, we performed experiments using only six arbitrarily chosen categories of 20NG. These categories do not correspond to each other and are not representative of remaining categories.

## 6. CONCLUSION

As most of the recent text categorization research focuses on addressing specific issues in TC (e.g., feature selection, clustering, and dimensionality reduction), very few new approaches are being devised. This paper proposes a new TC approach benefiting from the recent advances in feature clustering and dimensionality reduction coupled with a fairly effective logic-based learning technique. Our method employs an effective representation of documents using Multitype Features Coselection for Clustering of words. The method was extensively tested with numerous experiments using well-known benchmark data sets and compared with exact experimental settings against SVM. The proposed method outperformed the SVM-based method on all training testing settings using WebKB data set and on most experiments conducted with the 20NG data set. On the Reuters-21578 data set, the method showed equally good and very close performance results to SVM.

## REFERENCES


[1] L.D. Baker and A.K. McCallum, "Distributional Clustering of Words for Text Classification," *Proc. Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, 1998.

[2] M. Craven, D. DiPasquo, D. Freitag, A.K. McCallum, T.M. Mitchell, K. Nigam, and S. Slattery, "Learning to Extract Symbolic Knowledge from the World Wide Web," *Proc. Nat'l Conf. Artificial Intelligence (AAAI '98)*, 1998.

[3] I. Dhillon, S. Mallela, and R. Kumar, "A Divisive Information-Theoretic Feature Clustering Algorithm for Text Classification," *J. Machine Learning Research*, **3**, 2003.

[4] Shen Huang, Zheng Chen, Yong Yu, and Wei-Ying Ma "Multitype Features Coselection for Web Document Clustering" *IEEE Transactions on Knowledge and Data Engineering*, **18**, No. 4, April 2006.

[5] G. Felici and K. Truemper, "A Minsat Approach for Learning in Logic Domains," *Informs J. Computing*, **14**, no. 1, Winter.

[6] [11] P.A. Flach, "On the Logic of Hypothesis Generation," *Applied Logic Series*, **18**, Chapter 6, pp. 89-106, 2000.

[7] G. Forman, "An Extensive Empirical Study of Feature Selection Metrics for Text Classification," *J. Machine Learning Research*, **3**, 2003.

[8] M Hearst et al., Xerox TREC4 Site Report, TREC 4, 1996.

[9] T. Hofmann, "Probabilistic Latent Semantic Indexing," *Proc. Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, Aug. 1999.

[10] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," *Proc. 10th European Conf*, Machine Learning (ECML '98), 1998.

[11] T. Joachims, "A Statistical Learning Model of Text Classification with Support Vector Machines," *Proc. Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, 2001.

[12] K. Nigam, A.K. McCallum, S. Thrun, and T.M. Mitchell, "Learning to Classify Text from Labeled and Unlabeled Documents," *Proc. Nat'l Conf. Artificial Intelligence (AAAI '98)*, 1998.

[13] F. Pereira, N. Tishby, and L. Lee, "Distributional Clustering of English Words," Proc. 31st Ann. Meeting of the ACL, pp. 183.

[14] Reuters-21578: http://www.daviddlewis.com/ resources/testcol lections/reuters21578/, 2004.

[15] M.E. Ruiz and P. Srinivisan, "Hierarchical Neural Networks for Text Categorization," *Proc. Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, 1999.

[16] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, **34**, no. 1, pp. 1-47, 2002.

[17] N. Slonim and N. Tishby, "The Power of Word Clusters for Text Classification," *Proc. 23rd European Colloquium on Information Retrieval Research (ECIR-01)*, 2001.

[18] S.T. Dumais, J. Platt, D. Heckerman, and. Sahami, "Inductive Learning Algorithms and Representations for Text Categorization," *Proc. Seventh Int'l Conf. Information and Knowledge Management*, 1998.

[19] T. Liu, S. Liu, Z. Chen, and W.-Y. Ma, "An Evaluation on Feature Selection for Text Clustering," *Proc. Int'l Conf. Machine Learning (ICML '03)*, pp. 488-495, 2003.

[20] Y. Yang and J.O. Pedersen, "A Comparative Study on Feature Selection in Text Categorization," *Proc. Int'l Conf. Machine Learning (ICML '97)*, pp. 412-420, 1997.