

# Hierarchical Document Classification Using Automatically Generated Hierarchy

Tao Li

School of Computer Science  
Florida International University  
Miami, FL 33199  
*taoli@cs.fiu.edu*

Shenghuo Zhu

NEC Labs America, Inc.  
Cupertino, CA 95014  
*zsh@sv.nec-labs.com*

Mitsunori Ogihara

Department of Computer Science  
University of Rochester  
Rochester, NY 14627-0226  
*ogihara@cs.rochester.edu*

November 10, 2005

## **Abstract**

Automated text categorization has witnessed a booming interest with the exponential growth of information and the ever-increasing needs for organizations. The underlying hierarchical structure identifies the relationships of dependence between different categories and provides valuable sources of information for categorization. Although considerable research has been conducted in the field of hierarchical document categorization, little has been done on automatic genera-

tion of topic hierarchies. In this paper, we propose the method of using linear discriminant projection to generate more meaningful intermediate levels of hierarchies in large flat sets of classes. The linear discriminant projection approach first transforms all documents onto a low-dimensional space and then clusters the categories into hierarchies accordingly. The paper also investigates the effect of using generated hierarchical structure for text classification. Our experiments show that generated hierarchies improve classification performance in most cases.

## **1 Introduction**

### **1.1 Hierarchical Document Classification**

At the turn of the millennium we are faced with an explosive growth of the Internet and of digital documents at our disposal. It is an important issue to develop methods for efficiently accessing this enormous, ever-growing sea of documents. Automatic classification of on-line documents is of particular importance. This problem is generally cast as a supervised learning problem, in which models for classifying documents correctly into pre-defined categories are sought. Many studies in this area focus on *flat classification*, in which the predefined categories are treated individually and equally so that no structures exist to define relationships among them (Yang & Liu, 1999; D'Alessio et al., 2000). Limitations to the flat classification approach exists in the fact that, as the Internet grows, the number of possible categories increases and the borderlines between document classes are blurred. To resolve this issue, Koller and Sahami (1997) suggest the use of hierarchical structures. In such a hierarchical structure document types become more specific as we go down in the hierarchy.

There are two reasons that the hierarchical document classification is more useful. First, rather than issue keyword-based queries from general-purpose search engines, many users prefer to look for information by browsing hierarchical catalogs and by issuing queries that are corresponding to specific topics. Experiments have shown that an interface that organizes on the fly the keyword-based queries into hierarchies improves

usability, search success rate and user satisfaction (Chen & Dumais, 2000).

Second, hierarchical structures identify the relationships of dependence between the categories and provide a valuable information source for many problems. Generally, the use of hierarchical structures allows for efficiencies in both learning and representation. Hierarchical structures enable the use of a divide-and-conquer approach and thus result in higher efficiency and accuracy.

## **1.2 Automatic Hierarchy Generation**

Recently several researchers have studied the use of hierarchies for text classification and obtained promising results (D'Alessio et al., 2000; Sun & Lim, 2001). However, the previous studies were mostly conducted on corpora with predefined hierarchical structures and little has been done on automatic generation of topic hierarchies.

This motivates us to address the issue of automatically building hierarchies of documents. Such studies are meaningful for the following reasons: First, manual building of hierarchies is an expensive task since the process requires domain experts to evaluate the relevance of documents to the topics. Second, there may exist document domains in which there are no natural hierarchies and even domain experts have difficulties in evaluating the semantics. Third, automatic hierarchy generation based upon document statistics may generate hierarchies that provide better statistical correlations among categories. Once such a statistically more significant hierarchy has been build, the hierarchy can be incorporated into various classification methods to help achieve better performance.

## **1.3 Content of This Paper**

The first step in automatic hierarchy generation is to infer class relationships (e.g., measure the similarities between categories). The similarity should reflect the intrinsic characteristics of the categories and provide dependence relationships for efficient learning methods. Once the similarity measure has been determined, a hierarchical

clustering method can be used to build the hierarchy.

One simple way to infer class relationships is to compare the class representatives. As we all know, however, the data dimension is very high in vector space model for document analysis. It has been shown that in a high dimensional space, the distance between every pair of points is almost the same for a wide variety of data distributions and distance functions due to the *curse of dimensionality* (Beyer et al., 1999). In this paper, we utilize linear discriminant projection for generating more meaningful intermediate levels of hierarchies in large flat sets of classes. Linear discriminant projection approach first transforms all the documents onto a low-dimensional space and then clusters the categories into hierarchies according to the distances of the centroids of each category in the transformed space. Discriminant analysis approaches are well known to learn discriminative feature transformations in statistical pattern recognition literature (Fukunaga, 1990). Fisher discriminant analysis (Fisher, 1936) finds discriminative feature transform as eigenvectors of matrix  $T = S_w^{-1}S_b$  where  $S_w$  is the intra-class covariance matrix and  $S_b$  is the inter-class covariance matrix. Basically  $T$  captures both compactness of each class and separations between classes and hence eigenvectors corresponding to largest eigenvalues of  $T$  would constitute a discriminative feature transform. The transformed feature space would reflect the inherent similarity structure between the classes.

The rest of the paper is organized as follows: Section 2 introduces the linear discriminant projection approach. Section 3 discusses the use of hierarchical clustering method for hierarchy generation. Section 4 shows experiments of automatic hierarchy generation on real datasets. Section 5 presents the effect of using the generated hierarchies for text categorization. Section 6 compares the human-generated hierarchy and the automatically generated hierarchy. Section 7 reviews the related work on text categorization. Finally, Section 8 provides conclusions and discussions.

## 2 Linear Discriminant Projection Approach

The first step in hierarchy generation is to define the similarity measure between categories upon which hierarchies are built. In this section, we present the linear discriminant projection approach for inferring class relationships. Its core idea is to compare the class representatives in a low-dimensional space so that the comparison is more “meaningful”. More specifically, after finding the transformation, the similarity between two classes is defined to be the distance between their centroids in the transformed spaces.

### 2.1 Finding the Transformation

The notations used through the discussion of this paper are listed in the Table 1.

Notations	Descriptions
$A$	document-term matrix
$n$	number of data points, i.e., documents
$N$	number of the dimensions, i.e, terms
$k$	number of class
$S_i$	covariance matrix of the $i$ -th class
$S_b$	between-class scatter matrix
$S_w$	within-class scatter matrix
$S_t$	total scatter matrix
$G$	reduction transformation
$m_i$	centroid of the $i$ -th class
$m$	global centroid of the training set

Table 1: Notations

Given a document-term matrix  $A = (a_{ij}) \in R^{n \times N}$ , where each row corresponds to a document and each column corresponds to a particular term, we consider finding a linear transformation  $G \in R^{N \times \ell}$  ( $\ell < N$ ) that maps each row  $a_i$ , for  $1 \leq i \leq n$ , of  $A$  in the  $N$ -dimensional space to a row  $y_i$  in the  $\ell$ -dimensional space:

$$G : a_i \in R^{1 \times N} \rightarrow y_i \in R^{1 \times \ell}. \quad (1)$$

The resulting data matrix  $A^L = A \cdot G \in R^{n \times \ell}$  contains  $\ell$  columns, i.e. there are

$\ell$  features for each document in the reduced (transformed) space. It is also clear that the features in the reduced space are linear combinations of the features in the original high dimensional space, where the coefficients of the linear combinations depend on the transformation matrix  $G$ .

Linear discriminant projection tries to compute the optimal transformation matrix  $G$  such that the class structure is preserved. More details are given below.

Assume there are  $k$  classes in the data set. Suppose  $m_i, S_i, P_i$  are the mean vector, covariance matrix, and a prior probability of the  $i$ -th class, respectively, and  $m$  is the total mean. Then the between-class scatter matrix  $S_b$ , the within-class scatter matrix  $S_w$ , and the total scatter matrix  $S_t$  are defined as follows (Fukunaga, 1990):

$$S_b = \sum_{i=1}^k P_i (m_i - m)(m_i - m)^T, \quad (2)$$

$$S_w = \sum_{i=1}^k P_i S_i, \quad (3)$$

$$S_t = S_b + S_w. \quad (4)$$

For the covariance matrix  $S_i$  for the  $i$ th class, we can decompose it as  $S_i = X_i X_i^T$ , where  $X_i$  has the same number of columns as the number of data points in the  $i$ -th class. Define the matrices

$$\begin{aligned} H_w &= [\sqrt{P_1} X_1, \dots, \sqrt{P_k} X_k] \\ &\in R^{N \times n}, \\ H_b &= [\sqrt{P_1}(m_1 - m), \dots, \sqrt{P_k}(m_k - m)] \\ &\in R^{N \times k}. \end{aligned} \quad (5)$$

Then the scatter matrices  $S_w$  and  $S_b$  can be expressed as

$$S_w = H_w H_w^T, \quad S_b = H_b H_b^T. \quad (6)$$

In the lower-dimensional space resulting from the linear transformation  $G$ , the within-cluster and between-cluster matrices become

$$\begin{aligned} S_w^L &= (G^T H_w)(G^T H_w)^T = G^T H_w H_w^T G \\ &= G^T S_w G, \\ S_b^L &= (G^T H_b)(G^T H_b)^T = G^T H_b H_b^T G \\ &= G^T S_b G. \end{aligned} \quad (7)$$

An optimal transformation  $G$  would maximize trace ( $S_b^L$ ) and minimize trace ( $S_w^L$ ).

A common optimization for computing optimal  $G$  is

$$G = \arg \max_G \text{trace} \left( (G^T S_w G)^{-1} G^T S_b G \right). \quad (8)$$

The solution can be readily obtained by solving a eigenvalue decomposition problem on  $S_w^{-1} S_b$ , provided that the within-class scatter matrix  $S_w$  is nonsingular. Since the rank of the between-class scatter matrix is bounded above by  $k - 1$ , there are at most  $k - 1$  discriminant vectors.

## 2.2 Extension on General Case

In general, the within-class scatter matrix  $S_w$  may be singular especially for document-term matrix where the dimension is very high. A common way to deal with it is to use generalized eigenvalue decomposition (Howland & Park, 2003; Li et al., 2003a)

Let  $K = \begin{bmatrix} H_b^T \\ H_w^T \end{bmatrix}$ , which is a  $n + k$  by  $N$  matrix. By the generalized singular

value decomposition, there exist orthogonal matrices  $U \in R^{k \times k}$ ,  $V \in R^{n \times n}$ , and a nonsingular matrix  $X \in R^{N \times N}$ , such that

$$\begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix}^T KX = \begin{bmatrix} \Sigma_1 & 0 \\ \Sigma_2 & 0 \end{bmatrix}. \quad (9)$$

where

$$\Sigma_1 = \begin{bmatrix} I_b & 0 & 0 \\ 0 & D_b & 0 \\ 0 & 0 & 0_b \end{bmatrix}, \quad \Sigma_2 = \begin{bmatrix} 0_w & 0 & 0 \\ 0 & D_w & 0 \\ 0 & 0 & I_w \end{bmatrix}.$$

Here  $I_b, 0_w \in R^{r \times r}$  are identity and zero matrices respectively with

$$r = \text{rank}(K) - \text{rank}(H_w^T),$$

$$D_w = \text{diag}(\alpha_{r+1}, \dots, \alpha_{r+s}),$$

and

$$D_b = \text{diag}(\beta_{r+1}, \dots, \beta_{r+s}) \in R^{s \times s}$$

are diagonal matrix with

$$s = \text{rank}(H_b) + \text{rank}(H_w) - \text{rank}(K),$$

satisfying

$$1 > \alpha_{r+1} \geq \dots \geq \alpha_{r+s} > 0,$$

$$0 < \beta_{r+1} \leq \dots \leq \beta_{r+s} < 1,$$

and

$$\alpha_i^2 + \beta_i^2 = 1 \text{ for } i = r+1, \dots, r+s.$$

and

$$0_b \in R^{k-r-s \times t-r-s}, I_w \in R^{n-r-s \times t-r-s}$$

are zero and identity matrices respectively, where  $t$  is the rank of the matrix  $K$ .

From (9), we have

$$\begin{aligned} H_b^T X &= U \begin{bmatrix} \Sigma_1 & 0 \end{bmatrix} \\ H_w^T X &= V \begin{bmatrix} \Sigma_2 & 0 \end{bmatrix} \end{aligned}$$

and

$$\begin{aligned} X^T H_b H_b^T X &= \begin{bmatrix} \Sigma_1^T \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} = D_1, \\ X^T H_w H_w^T X &= \begin{bmatrix} \Sigma_2^T \Sigma_2 & 0 \\ 0 & 0 \end{bmatrix} = D_2, \end{aligned}$$

Hence a natural extension of the proposed linear discriminant projection in Section 2.1 is to choose the first  $q = r + s$  columns of the matrix  $X$  in (9) as the transformation matrix  $G$ .

### 2.3 Defining the Similarity

After finding the transformation  $G$ , we define the similarity between two classes to be the distance between their centroids in the transformed spaces. In other words, two categories are similar if they are “close” to each other in the transformed space. The linear discriminant projection finds the transformation that preserves the class structure by minimizing the sum of squared within-class scatter while maximizing the sum of squared between-class scatter and hence the distances in the transformed space should be able to reflect the inherent structure of the dataset.

## 2.4 Linear Discriminant Projection and Latent Semantic Indexing

To make it more clear on the linear discriminant projection approach, we compare the method with the well-known Latent Semantic Indexing and give some concrete examples. A well-known transformation method in information retrieval is, Latent Semantic Indexing (LSI) (Deerwester et al., 1990) is a well-known transformation method in information retrieval, which applies Singular Value Decomposition (SVD) on the document-term matrix and finds eigenvectors with largest eigenvalues as the directions related to the dominant combinations of terms occurring the dataset (*latent semantics*). The transformation matrix constructed from these eigenvectors then projects a document onto the latent semantic space. Although LSI has been proven to be extremely useful in various information retrieval tasks, it is not an optimal transformation for text categorization since LSI is completely unsupervised. In other words, LSI aims at optimal transformation of the original data into the lower dimensional space in terms of mean squared error but it pays no attention to the underlying class structure. Linear discriminant projection explicitly utilizes the intra-class and inter-class covariance matrices and tends to preserve the class structure.

To illustrate how useful linear discriminant projection is, we present some examples here. Consider a dataset consisting of nine sentences as shown in Figure 1. These sentences are from three different topics: user interaction, graph theory and distributed systems.

- 1(1) Human interface for user response
- 2(1) A survey of user opinion of computer system response time
- 3(1) Relation of user-perceived response time to error measurement
- 4(2) The generation of random, binary, unordered trees
- 5(2) The intersection graph of paths in trees
- 6(2) Graph Minors IV: Widths of trees and well-quasi-ordering
- 7(3) A survey of distributed shared memory system
- 8(3) RADAR: A multi-user distributed system
- 9(3) Management interface tools for distributed computer system

Figure 1: Dataset Consisting of Nine Sentences

By removing words/terms that occur only once, we obtain the document-term matrix. Suppose that the first and second sentences in each class are used for training data. Then the transformation shown in Figure 2(a) is obtained. The plot of the LSI algorithm is shown in Figure 2(b).

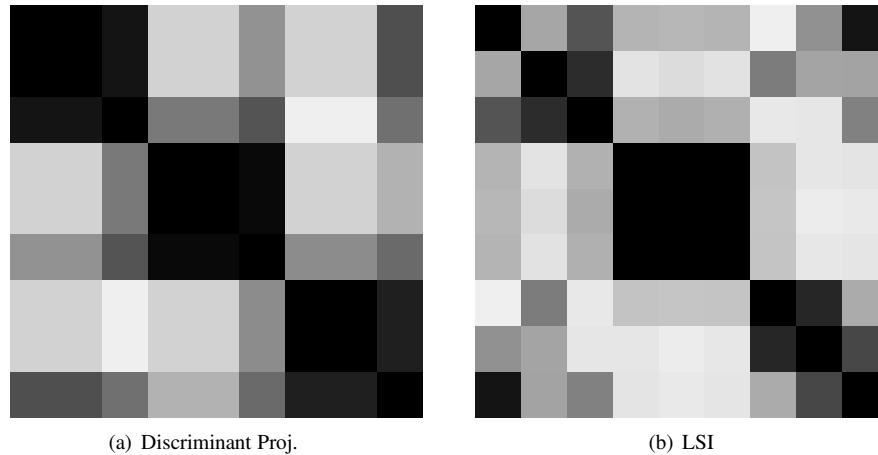
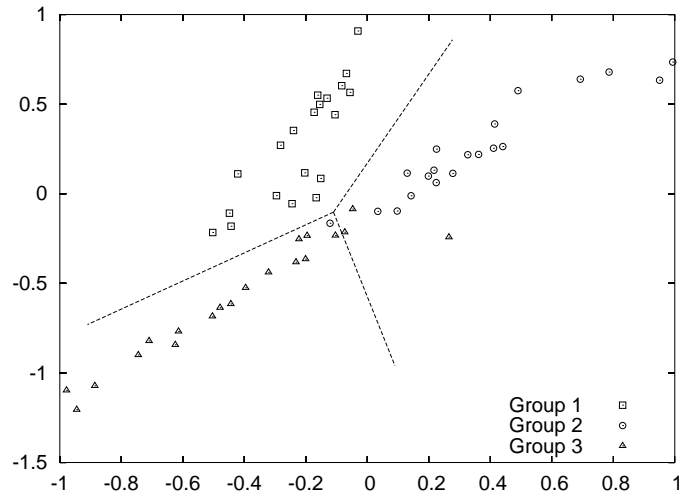


Figure 2: Document similarity. Both  $x$ -axis and  $y$ -axis represent the documents. Each block represents the similarity between the corresponding row and column documents. The darker the contrast, the more similar the documents. For perfect class structure preserving, we expect three consecutive dark squares along the main diagonal. Linear discriminant projection gives almost perfect separation results while LSI does not.

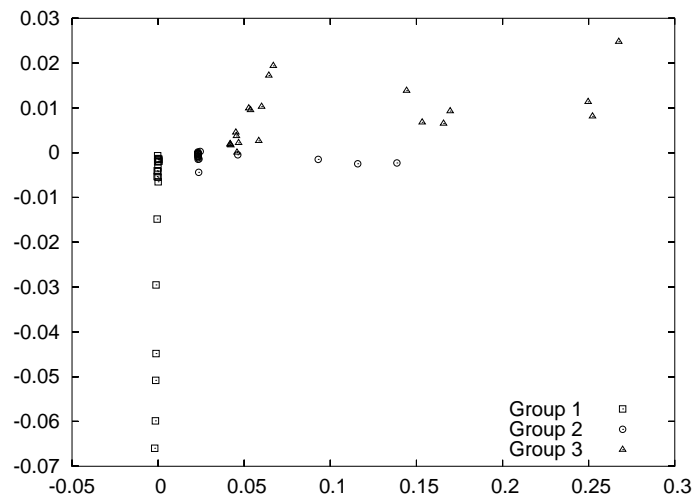
Here is another example. We perform linear discriminant projection on a set of news articles from the 20Newsgroup dataset <sup>1</sup>. 60 articles are selected as training data from three newsgroups, with 20 articles per newsgroup. Another 60 articles from those newsgroups are selected as testing data. Figures 3(a) and 3(b) show the 60 testing articles plotting onto planes of the first two dominant directions obtained via linear discriminant projection and LSI, respectively.

These two examples show that linear discriminant projection has discrimination power and is able to reflect the inherent similarity structure of the classes. Hence the distance between the centroids is a good measure for the similarity between categories.

<sup>1</sup>The description of the dataset is presented in Section 4.



(a) Discriminant proj.



(b) LSI

Figure 3: Visualizations of linear discriminant projection and LSI. Note that  $x$ -axis and  $y$ -axis represent the first two dominant directions. The three groups could be well separated using linear discriminant projection.

### 3 Hierarchy Generation

After obtaining the similarities/distances between classes, we use the Hierarchical Agglomerative Clustering (HAC) algorithm of (Jain & Dubes, 1988) to generate automatic topic hierarchies from a given set of flat classes. The result of hierarchical clustering is a dendrogram where similar classes are organized into hierarchies. There are many different HAC algorithms with different policies on combining clusters such as single-linkage, complete-linkage, Ward’s method and the UPGMA method. Different HAC algorithms may produce different dendrogram structures. Single-linkage clustering is known to be confused by nearby overlapping clustering and tend to produce long chains which form loose clusters. Complete-linkage tends to produce very tight clusters. Ward’s method is only compatible with Euclidean metric and not compatible with non-Euclidean metric and semi-metric. We choose UPGMA (Unweighted Pair-Groups Method Average method), which is known to be simple, efficient and stable (Jain & Dubes, 1988). In UPGMA, the average distance between clusters is calculated from the distance between each point in a cluster and all other points in another cluster. The two clusters with the lowest average distance are joined together to form the new cluster.

### 4 Experiments on hierarchy Generation

We use a wide range of datasets in our experiments. Most of the datasets have been widely used in the information retrieval literature. The number of classes ranged from seven to 105 and the number of documents ranged from 2,340 to 20,000. We anticipate that these data sets would provide us enough insights on automatic hierarchy generation. The datasets and their characteristics are summarized in Table 2. **20Newsgroups** dataset<sup>2</sup> contains about 20,000 articles evenly divided among 20 Usenet newsgroups.

**WebKB** dataset contains webpages gathered from university computer science depart-

<sup>2</sup><http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/news20.html>.

Datasets	# documents	# classes
20Newsgroups	20,000	20
WebKB	8,280	7
Industry Sector	9,637	105
Reuters-top 10	2,900	10
Reuters-2	8,000	42
K-dataset	2,340	20

Table 2: Data Sets Descriptions

ments. There are about 8300 documents and they are divided into seven categories: student, faculty, staff, course, project, department and other. **Industry Sector** dataset consists of company homepages classified in a hierarchy of industry sectors<sup>3</sup>. **Reuters:** The Reuters-21578 Text Categorization Test collection contains documents collected from the Reuters newswire in 1987. It is a standard text categorization benchmark and contains 135 categories. In our experiments, we used two subsets of the data collection. The first one includes the ten most frequent categories among the 135 topics, which we call Reuters-top10. The second one contains the documents that have unique topics (documents that have multiple class assignments are ignored), which we call Reuters-2. **K-dataset**<sup>4</sup> contains 2340 documents consisting news articles from Reuters news service via the Web in October 1997.

#### 4.1 Data Preprocessing

To preprocess the datasets, we remove the stop words using a standard stop list and perform the stemming operations with a Porter stemmer. All HTML tags and all header fields except subject and organization are ignored. In all our experiments, we first randomly choose 70% for hierarchy building (and later training in categorization), the remaining 30% is then used for testing. The 70% training set is further preprocessed by selecting the top 1000 words by information gain. The feature selection is done with the rainbow package<sup>5</sup>. All of our experiments are performed on a P4 2GHz machine

<sup>3</sup><http://www.cs.cmu.edu/~TextLearning/datasets.html>.

<sup>4</sup><ftp://ftp.cs.umn.edu/dept/users/boley/PDDPdata/>.

<sup>5</sup><http://www.cs.cmu.edu/~mccalum/bow>.

with 512M memory running Linux 2.4.9-31.

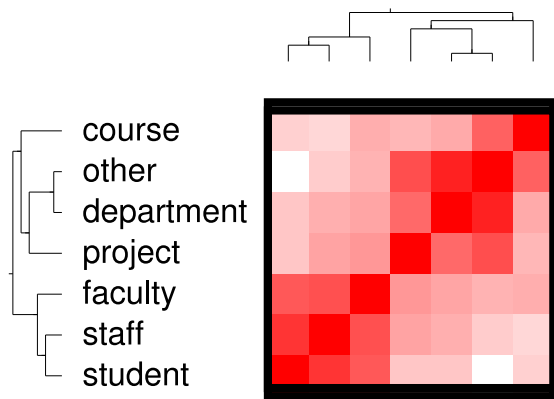
## 4.2 Experimental Results

Figure 4 shows the hierarchies of WebKB, 20Newsgroups and Reuters-top10 built via linear discriminant projection. To save space, we do not include the hierarchies on K-dataset, Reuters-2 and Industry sector. The block in the graphs represents the similarity between the corresponding row and column document categories and the darker the more similar.

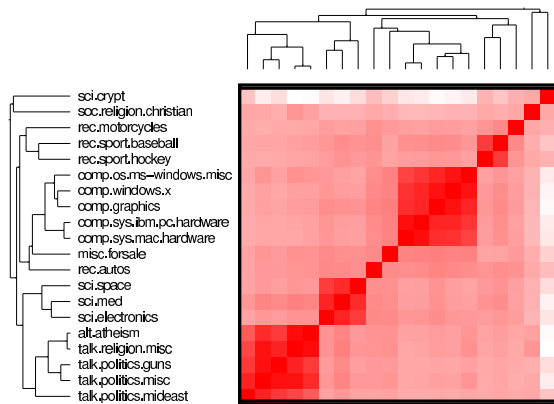
We can observe from the dendrogram the semantic similarity of classes. For example, on WebKB, *faculty*, *staff* and *student* are close to each other. Note that *faculty*, *staff* and *student* are people and they are different from other datasets. Hence the linear discriminant projection approach tends to group them together. We can observe similar phenomena on 20Newsgroup and Reuters-top10. On 20Newsgroups, *talk.politics.guns* and *talk.politics.misc* are grouped together. On Reuters-top10, for example, the close pairs of classes using linear discriminant projection are: (*ship*, *crude*), (*money-fx*, *interest*), (*grain*, *wheat*) and (*earn*, *acq*).

## 5 Exploiting the Generated Hierarchy for Classification

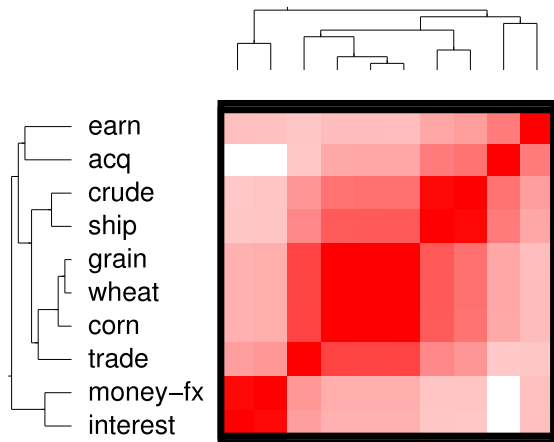
In this section, we investigate the effects of exploiting the automatically generated hierarchy for classification. In the following experiments we use classification accuracy as the evaluation measure. Different measures such as precision-recall graphs and  $F_1$  measure (Yang & Liu, 1999) have been used in the literature. Note that Precision/recall is usually used for retrieval tasks, though you can treat retrieval tasks as two-class classification ones. Therefore, precision/recall is only defined for a specific label. For example, you can say the precision/recall of class *acq* is 0.9. This holds for F-measures too. Here we focus on the general multi-class classification problem instead of multi-label problem where a document can belong to multiple classes. Since the goal of text categorization algorithm is to achieve low misclassification error on a test set (Schapire



(a) WebKB



(b) 20NG



(c) Reuters

Figure 4: Hierarchies of WebKB, 20Newsgroups and Reuters-top 10 using linear discriminant projection.

& Singer, 2000) and the datasets used in the experiments are relatively balanced, we use accuracy as the performance measure.

An obvious approach to utilization of the hierarchy is a top-down level-based approach that arranges the clusters in a two-level tree hierarchy and trains a classifier at each internal node. We analyze the generated dendrogram to determine the clusters that provide maximum inter-class separation and find the best grouping of classes at the top level. The dendrogram is scanned in a bottom-up fashion to find the distances at which successive clusters get merged. We clip the dendrogram at the point where the cluster merge distances begin increasing sharply. In our experiments, we clip the dendrogram when the current merge distance is at least two times larger than previous one. For example, on 20Newsgroups dataset with linear discriminant projection approach, we have 8 top-level groups as shown in Table 3. The experimental results reported here are obtained via two-level classification.

groups	members
1	<i>alt.atheism, talk.region.misc, talk.politics.guns, talk.politics.misc, talk.politics.mideas</i>
2	<i>sci.space, sci.med sci.electronic</i>
3	<i>comp.os.mswindows.misc, comp.sys.ibm.pc.hardware, comp.graphs, comp.sys.mac.hardware, comp.windows.x</i>
4	<i>rec.sport.baseball, rec.sport.hockey, rec.motorcycles</i>
5	<i>misc.forsale</i>
6	<i>soc.religion.christian</i>
7	<i>rec.autos</i>
8	<i>sci.crypt</i>

Table 3: Top level groups for 20Newsgroups via linear projection.

LIBSVM<sup>6</sup> is used as our classifier. LIBSVM is a library for support vector classification and regression and supports multi-class classification. In addition, we use linear kernel in all our experiments as it gives best results on our experiments.

We first build a top-level classifier (L1 classifier) to discriminate among the top-level clusters of labels. At the second level (L2) we build classifiers within each cluster of classes. Each L2 classifier can concentrate on a smaller set of classes that confuse

<sup>6</sup><http://www.csie.ntu.edu.tw/~cjlin/libsvm>

with each other. In practice, each classifier has to deal with a more easily separable problem, and can use an independently optimized feature set; this should lead to slight improvements in accuracy apart from the gain in training and testing speed. Table 4 gives the performance comparisons of flat classification with hierarchical classification. We observe the improved performance on all datasets.

Datasets	Flat	Linear Projection	Linear Projection
		Level One	Overall
20Newsgroups	0.952	0.985	0.963
WebKB	0.791	0.860	0.804
Industry Sector	0.691	0.739	0.727
Reuters-top 10	0.826	0.963	0.921
Reuters-2	0.923	0.938	0.927
K-dataset	0.915	0.961	0.921

Table 4: Accuracy Table. The flat column gives the accuracy of flat classification, the “Level One” column shows the level one accuracy while the “Overall” column represent the overall accuracy.

From Table 4, we observe that Reuters-top10 has the most significant gain in accuracy using hierarchy. Figure 5 presents the accuracy comparison for each class of Reuters-top10. Shown in Figure 5, each class’ accuracy is improved by using the generated hierarchy except the accuracy of *trade* stays unchanged. The accuracy of *corn* is improved significantly from about 7% to 60%. In flat classification, almost all the documents in *corn* class are misclassified to *grain* and *wheat* classes. Using hierarchical classification, by grouping *corn*, *grain* and *wheat* together, A second-level classifier using an independently optimized feature set can then be designed to focus on separation of the three similar classes. The performance improvement is thus obtained.

## 6 Manually-Built Hierarchy vs Generated Hierarchy

Once we have the automatic approach for hierarchy generation, it is natural to compare the generated hierarchy with the manually-built one. In this section, we illustrate their difference via three experiments on 20Newsgroups, WebKB and Reuters-top10.

Table 5, Table 6 and Table 7 present the human-built (two-level) hierarchies for

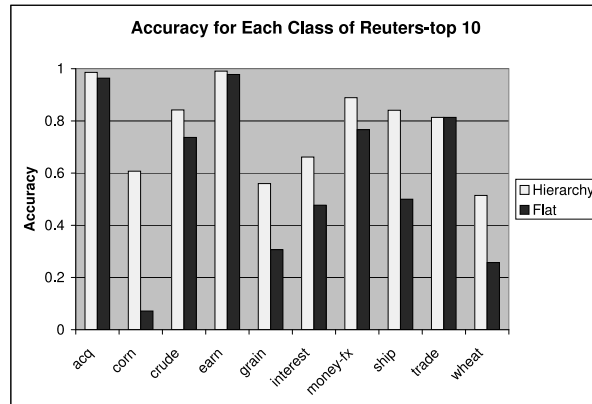


Figure 5: Accuracy comparison on each class of Reuters-top10

the three dataset respectively. The human-built hierarchies are generated by a subject (A 41-year old, male, professor). The subject is instructed to group the categories with strong confidence. To further understand the differences between two kinds of hierarchies, we also compare their hierarchical categorization performances, as listed in Table 8. For comparison purposes, the experimental results are based on two-level classifiers.

As you can observe from the comparisons, human-built hierarchy is purely based on “human semantics”, but not necessarily optimized for classification purpose. In all the three datasets, using the automatic generated hierarchy, the classification accuracies are slightly higher than those using human-built hierarchy. Hence, an important research direction is to combine the automatic and manual approaches for generating both statistically significant and intuitively meaningful hierarchies.

## 7 Related Work

### 7.1 Hierarchical Classification

Generally the problem of hierarchy classification consists of two tasks: classification approaches and hierarchy generation. The related work is summarized in Figure 6.

groups	members
1	<i>talk.region.misc, talk.politics.guns, talk.politics.misc, talk.politics.mideas</i>
2	<i>sci.electronic, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware</i>
3	<i>comp.os.mswindows.misc, sci.crypt, comp.graphs, comp.windows.x</i>
4	<i>rec.sport.baseball, rec.sport.hockey</i>
5	<i>misc.forsale</i>
6	<i>alt.atheism, soc.religion.christian</i>
7	<i>rec.autos, rec.motorcycles</i>
8	<i>sci.space, sci.med</i>

Table 5: Human-generated 8 top-level groups for 20Newsgroups.

groups	members
1	<i>course, project</i>
2	<i>Faculty, Student, Staff</i>
3	<i>department, others</i>

Table 6: Human-generated top-level groups for WebKB.

This present paper complements the existing hierarchical classification research by investigating the automatic hierarchy generation for text categorization. Our work also differs from traditional document clustering in the sense that the goal is to cluster the document categories into hierarchies, not to organize documents into groups according to the similarity measures defined on the set of documents, as defined in (Jain & Dubes, 1988). In our work, we propose the linear discriminant projection approach to measure the similarities between document categories and then use clustering techniques to generate hierarchies.

groups	members
1	<i>corn, grain, wheat</i>
2	<i>earn, interest, money-fix</i>
3	<i>acq</i>
4	<i>crude</i>
5	<i>ship</i>
6	<i>trade</i>

Table 7: Human-generated top-level groups for Reuters-top 10.

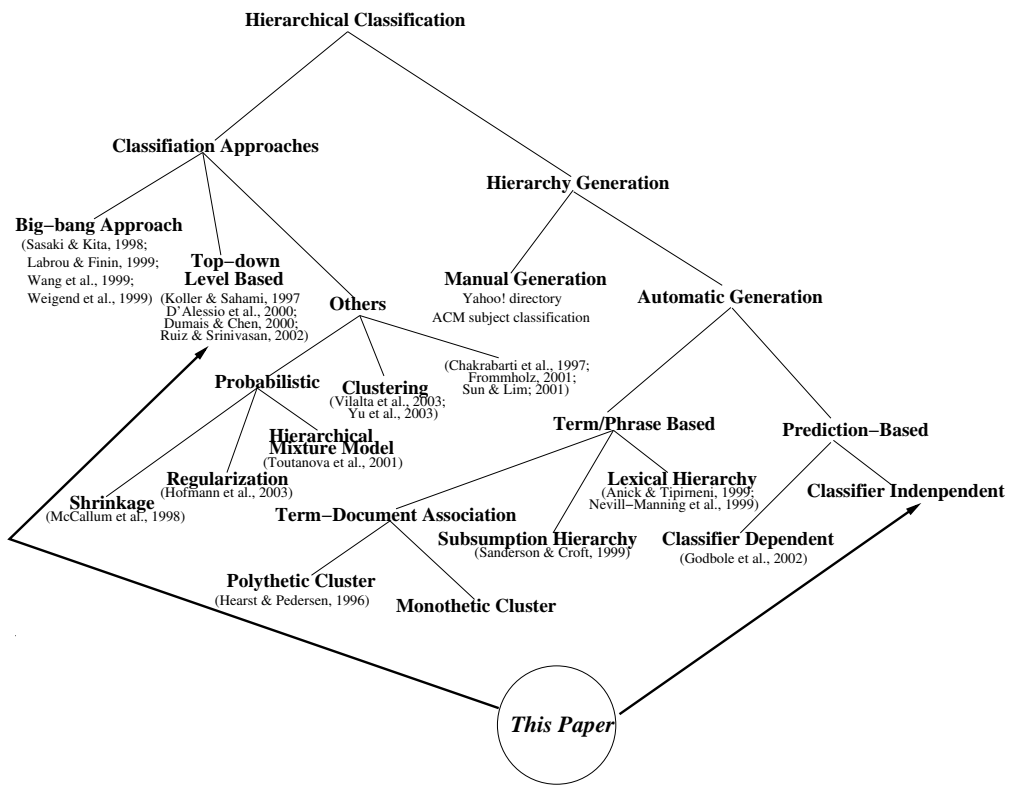


Figure 6: Summary of Related Work on Hierarchical Classification

Datasets	Human-generated	Automatic Generated
20Newsgroups	(0.956, 0.954)	(0.985, 0.963)
WebKB	(0.811, 0.795)	(0.860, 0.804)
Reuters-top 10	(0.928, 0.9475)	(0.963, 0.901)

Table 8: Performance comparisons of human-generated hierarchy with automatic generated hierarchy. The entries are in the format of (level one, flat). The accuracy of automatic generated hierarchy was taken from the linear projection approach.

#### 7.1.1 RELATED WORK ON HIERARCHICAL CLASSIFICATION APPROACHES

Recently several research papers have appeared with a focus on the use of hierarchies for text categorization. Broadly speaking two approaches are adopted by existing hierarchical classification methods: the big-bang approach and the top-down level-based approach (Sun & Lim, 2001). In the big-bang approach, only a single classifier is used in the classification process and the concept of “closeness” or hierarchy information is used to improve the effectiveness. The big-bang approach has been taken in (Weigend et al., 1999; Sasaki & Kita, 1998; Wang et al., 1999). In the top-down level-based approach, one or more classifiers are constructed at each level of the category tree and each classifier works as a flat classifier at its level. Documents are classified into sub-categories using some flat classification method(s) at the root level, then they are classified into each of the sub-categories until they reach a leaf category, or they cannot be further classified into any sub-category. The top-down level-based approach has been implemented in (D’Alessio et al., 2000; Dumais & Chen, 2000; Koller & Sahami, 1997; Ruiz & Srinivasan, 2002).

There is some other work on hierarchical classification approaches by using probabilistic methods, hierarchical mixture models, and clustering techniques etc. McCallum et al. (1998) use a Bayesian framework and a technique called Shrinkage to improve parameter estimates for the probability of words in given classes. The idea is to smooth the parameter estimate of a node by interpolation with all its parent node specified by the hierarchical organization of the class. Hofmann et al. (2003) present the idea of constructing hierarchical discriminant functions and using a regularization approach

by taking advantage of the prior knowledge encoded by the taxonomy. Toutanova et al. (2001) present a hierarchical mixture model. In (Toutanova et al., 2001), the hierarchy information is used to improve the term estimates and to obtain a differentiation of words in the hierarchy according to their level of generality and specificity. Vilalta and Rish (2003) propose a method to improve the performance of Naive Bayes by decomposing each class into clusters and Yu et al. (2003) propose the idea of using hierarchical micro-clustering to facilitate support vector classification. Chakrabarti et al. (1997) use hierarchical structures to both select features and combine class information from multiple levels of the hierarchy. Frommholz (2001) proposes a method where hierarchical structures are used to post-process the results of a flat classification algorithm. Sun and Lim (2001) propose a top-down level-based method that can classify documents to both internal and leaf categories and can define a new set of performance measures that consider the semantic and dependence relationships among categories.

#### 7.1.2 RELATED WORK ON HIERARCHY GENERATION

The organization of a collection of documents into hierarchies derived automatically from the collection itself has been studied in information retrieval (Hearst & Pedersen, 1996; Sanderson & Croft, 1999; Lawrie et al., 2001; Lawrie & Croft, 2000; Nevill-Manning et al., 1999). These studies usually is based on selecting topic terms (salient terms that can identify main themes in the document set) and/or frequently occurring phrases. These selected terms/phrases are then used to build subsumption hierarchies, based on the subsuming relationships among the terms in a bottom-up fashion (Sanderson & Croft, 1999), or Lexical hierarchies, based on the lexical dispersion of the phrases in a top-down fashion (Anick & Tipirneni, 1999; Nevill-Manning et al., 1999). These term/phrase based hierarchy are mainly used for document summarization and navigation rather than document classification.

On the other hand, most of the classification techniques described in Section 7.1.1 have been conducted using existing hierarchically structured corpora. Little has been

done on automatic generation of the hierarchy structure. Recently, Godbole et al. (2002) present a technique for exploiting the hierarchy structure to scale up multi-class classification. In their work, a naive Bayes classifier is first used to quickly compute a confusion matrix and then similarity is defined in the confusion space, in the sense that two class are similar if they confuse or mis-classify test instances among a similar set of classes. Our approach is different from this approach since ours are classifier-independent and they build hierarchies from training documents instead of “flat” classifiers, that is, any classifiers can be used for classification. In addition, our approach takes advantage of all the training documents while the confusion matrix approach requires to reserve a part of the training documents for construction of the confusion matrix.

## 7.2 Linear Discriminant Projection

Several earlier research efforts have shown that a projection-based approach could be promising. Frankl and Maehara (1988) show that a projection of a set of  $n$  points in  $\mathcal{R}^m$  to a random subspace of dimension about  $(9/\epsilon^2)\log n$  preserves (to within a  $\epsilon$  factor) all relative inter-point distances with high probability. Kleinberg (1997) projects these points to  $\Theta(m\log^2 m)$  randomly directed lines to answer approximated nearest-neighbor queries efficiently. Little work has been reported on using discriminant projection approach in the document analysis domain. Chakrabarti et al. (2002) propose a fast text classification approach via multiple linear projections. It first projects training instances to low-dimensional space and then using decision trees on the projected spaces. Li et al. (Li et al., 2003a; Li et al., 2003b) experimentally investigate the use of discriminant analysis for multi-class classification problems (e.g., text categorization). To the best of our knowledge, our work is the first investigation on using discriminant projection for hierarchy generation.

## 8 Discussions and Conclusions

In this paper, we propose the approach to automatic generation of hierarchy structures via linear discriminant projection. The linear discriminant projection approach measures the similarities between categories by the distances between category centroids in the transformed space and builds hierarchies from training documents instead of “flat” classifiers. A hierarchical clustering method is then used to generate the dendrogram based on the similarity measures. Finally we investigate the effect of using generated hierarchical structure for text categorization. Our experiments demonstrate that generated hierarchies can improve classification performance in most cases.

There are several natural avenues for future research. First, as we discussed in Section 6, an immediate task is to combine the automatic and manual approaches for generating both statistically significant and intuitively meaningful hierarchies. Second, the categories among different branches of a hierarchy usually have smaller correlations than those within the same branch. The hierarchy information thus could be used to design efficient error-correcting output code (ECOC) for multi-class classification. Designing ECOC code is not an easy task since it usually requires row vectors and column vectors of code matrices to be uncorrelated. The hierarchical information gives additional hints to design the ECOC codes. Third, currently we use two-level hierarchy for text categorization and it is important to see how it can be generalized to more than two levels where error cascading may be an issue. Finally, the question on how to incorporate the hierarchy information for efficient text categorization and feature selection needs further investigation.

### Acknowledgment

The authors would like to thank the anonymous reviewers for their invaluable comments. The project is partially supported by a 2005 IBM Faculty Award and a 2005 IBM Shared University Research Award.

## References

- Anick, P., & Tipirneni, S. (1999). The paraphrase search assistant: Terminological feedback for iterative information seeking. *In Proceedings of the 22nd Annual ACM International Conference on Research and Development in Information Retrieval (SIGIR 1999)* (pp. 153–159).
- Beyer, K. S., Goldstein, J., Ramakrishnan, R., & Shaft, U. (1999). When is nearest neighbor meaningful? *Proceedings of 7th International Conference on Database Theory(ICDT'99)* (pp. 217–235). Springer.
- Chakrabarti, S., Dom, B. E., Agrawal, R., & Raghavan, P. (1997). Using taxonomy, discriminants, and signatures for navigating in text databases. *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB'97)* (pp. 446–455). Morgan Kaufmann Publishers.
- Chakrabarti, S., Roy, S., & Soundalgekar, M. V. (2002). Fast and accurate text classification via multiple linear discriminant projections. *Proceedings of the 28th International Conference on Very Large Data Bases(VLDB'02)*. Morgan Kaufmann Publishers.
- Chen, H., & Dumais, S. T. (2000). Bringing order to the web: automatically categorizing search results. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'00)* (pp. 145–152). ACM Press.
- D'Alessio, S., Murray, K., Schiaffino, R., & Kershenbaum, A. (2000). The effect of using hierarchical classifiers in text categorization. *Proceeding of RIAO-00, 6th International Conference "Recherche d'Information Assistee par Ordinateur"* (pp. 302–313).
- Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41, 391–407.

- Dumais, S. T., & Chen, H. (2000). Hierarchical classification of Web content. *Proceedings of the 23rd ACM International Conference on Research and Development in Information Retrieval (SIGIR'00)* (pp. 256–263). ACM Press.
- Fisher, R. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, 179–188.
- Frankl, P., & Maehara, H. (1988). The johnson-lindenstrauss lemma and the sphericity of some graphs. *Journal of Combinatorial Theory*, 44, 355–362.
- Frommholz, I. (2001). Categorizing web documents in hierarchical catalogues. *Proceedings of the 23rd European Colloquium on Information Retrieval Research (ECIR-01)*.
- Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. New York: Academic Press. 2nd edition.
- Godbole, S., Sarawagi, S., & Chakrabarti, S. (2002). Scaling multi-class support vector machine using inter-class confusion. .
- Hearst, M. A., & Pedersen, J. O. (1996). Reexamining the cluster hypothesis: Scatter/gather on retrieval results. *Proceedings of the 19th Annual ACM International Conference on Research and Development in Information Retrieval (SIGIR 1996)* (pp. 76–84). ACM Press.
- Hofmann, T., Cai, L., & Ciaramita, M. (2003). Learning with taxonomies: classifying documents and words. *Proceedings of workshop on Syntax, Semantics and Statistics, NIPS 2003*.
- Howland, P., & Park, H. (2003). Generalizing discriminant analysis using the generalized singular value decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

- Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice Hall.
- Kleinberg, J. M. (1997). Two algorithms for nearest-neighbor search in high dimensions. *ACM Symposium on Theory of Computing* (pp. 599–608).
- Koller, D., & Sahami, M. (1997). Hierarchically classifying documents using very few words. *ICML*.
- Lawrie, D., Croft, W. B., & Rosenberg, A. (2001). Finding topic words for hierarchical summarization. *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval(SIGIR'01)* (pp. 349–357). New Orleans, Louisiana, United States: ACM Press.
- Lawrie, D. J., & Croft, W. B. (2000). Discovering and comparing topic hierarchies. *In Proceedings of RIAO 2000*.
- Li, T., Zhu, S., & Ogihara, M. (2003a). Efficient multi-way text categorization via generalized discriminant analysis. *ACM CIKM* (pp. 317–324).
- Li, T., Zhu, S., & Ogihara, M. (2003b). Using discriminant analysis for multi-class classification. *Proceedings of the Third IEEE International Conference on Data Mining (ICDM 2003)* (pp. 589–592).
- McCallum, A. K., Rosenfeld, R., Mitchell, T. M., & Ng, A. Y. (1998). Improving text classification by shrinkage in a hierarchy of classes. *Proceedings of the 15th International Conference on Machine Learning (ICML'98)* (pp. 359–367). Morgan Kaufmann Publishers, San Francisco, US.
- Nevill-Manning, C. G., Witten, I. H., & Paynter, G. W. (1999). Lexically-generated subject hierarchies for browsing large collections. *International Journal on Digital Libraries*, 2, 111–123.
- Ruiz, M. E., & Srinivasan, P. (2002). Hierarchical text categorization using neural networks. *Information Retrieval*, 5, 87–118.

- Sanderson, M., & Croft, W. B. (1999). Deriving concept hierarchies from text. *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR'99)* (pp. 206–213). ACM Press.
- Sasaki, M., & Kita, K. (1998). Rule-based text categorization using hierarchical categories. *Proc. of IEEE SMC*.
- Schapire, R. E., & Singer, Y. (2000). Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39, 135–168.
- Sun, A., & Lim, E.-P. (2001). Hierarchical text classification and evaluation. *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM'01)* (pp. 521–528). IEEE Computer Society.
- Toutanova, K., Chen, F., Popat, K., & Hofmann, T. (2001). Text classification in a hierarchical mixture model for small training sets. *Proceedings of 10th ACM International Conference on Information and Knowledge Management (CIKM'01)* (pp. 105–113). ACM Press.
- Vilalta, R., & Rish, I. (2003). A decomposition of classes via clustering to explain and improve naive Bayes. *Proceedings of 14th European Conference on Machine Learning (ECML 2003)* (pp. 444–455). Springer.
- Wang, K., Zhou, S., & Liew, S. C. (1999). Building hierarchical classifiers using class proximity. *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB'99)* (pp. 363–374). Morgan Kaufmann Publishers , San Francisco, US.
- Weigend, A. S., Wiener, E. D., & Pedersen, J. O. (1999). Exploiting hierarchy in text categorization. *Information Retrieval*, 1, 193–216.
- Yang, Y., & Liu, X. (1999). A re-examination of text categorization methods. *Proceed-*

*ings of the 22nd Annual International Conference on Research and Development in Information Retrieval (SIGIR'99)* (pp. 42–49). ACM Press.

Yu, H., Yang, J., & Han, J. (2003). Classifying large data sets using SVMs with hierarchical clusters. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD 2003)* (pp. 306–315). ACM Press.