ITERATIVE SUBSPACE TEXT CATEGORIZATION

FRANCIS CHO-YIU CHIK

Master of Philosophy

The Hong Kong Polytechnic University

2013

The Hong Kong Polytechnic University

Department of Computing

Iterative Subspace Text Categorization

by

Francis Cho-yiu Chik

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Philosophy

December 2008

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

________________________________ (Signed)

Francis Cho-yiu Chik (Name of student)

Abstract

Text categorization finds many practical applications. The dominant approach involves the use of various machine learning techniques where classification rules are automatically created using information from labeled texts. The proposed method to combat the curse of dimensionality is subspace methodology. However, this has only been applied broadly in unsupervised text categorization. The performance of subspace methodology on supervised text categorization has not yet been found. The approach of iterative subspace method of pattern classification is investigated. For the topic pairs of "carcass_livestock" and "soybean_oilseed" from the Reuters-21578 collection, the results with confidence level greater than 95% under 8-fold/10-fold/12-fold cross validation shows the potential of this approach. It is expected that the performance can be further improved by using other optimization techniques.

It is still promising that there is 8.24% precision improvement of "livestock" evaluated comparing to 1-level classifier, standard Support Vector Machine (SVM), under 8-fold cross validation. There is also 11.85% improvement of "nat-gas" evaluated comparing to Soft Margin SVM classifier under 8-fold cross validation.

Acknowledgements

Many people have made invaluable contributions to the successful completion of this thesis. I would like to take this opportunity to express my sincere thanks to their kindness and guidance. I would like to express my deep gratitude to my supervisors, Dr Robert Luk and Dr Korris Chung, for their support and guidance.

I wish to thank my best friends, Lawrence Cheung and Jacqueline Lam, who have always helped me. Finally, I also wish to thank my family for their support and encouragement throughout my study.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Text categorization and its applications

Text categorization is the task in which texts are classified into one of predefined categories based on their contents. This task has various applications such as automatic email classification, news classification and webpage categorization. Those applications are becoming increasingly important in today's information-oriented society. Much knowledge in this domain has been accumulated in the past 30 years

There are mainly two types of approaches to text categorization. One is the rule-based approach where the classification rules are manually created usually by experts in the domain of the texts. Although the rule-based approach can achieve high accuracy, it is costly in terms of labor and time. Moreover, a rule-based system created for one domain can hardly be used in other domains. The second approach involves machine learning techniques where classification rules are automatically created using information from labeled texts. It enables a system for a new domain to be easily constructed. Text categorization is also called text classification, document categorization or document classification.

Generally, building an automated text categorization system consists of two key subtasks. The first task is text representation which converts the content of documents into a compact format so that they can be further processed by the text classifiers. Another task is to build the model of a text classifier to classify unlabelled documents.

The textual information is stored in many kinds of machine readable form, such as PDF, DOC, PostScript, HTML, XML and so on. Before the computer applies the text classifier to label the unknown document, the content of a document must be transformed into a compact and interpretable format so that it can be further recognized and classified by a computer or a classifier. This indexing procedure is called text representation.

The algorithms which have been applied to text categorization task have been studied extensively in recent decades and most of them are usually borrowed from the traditional pattern recognition, such as Support Vector Machines, k-Nearest Neighbor, Decision Tree, Naive Bayes, Neural Network, Linear Regression, etc. As a relatively new algorithm, Support Vector Machines [24, 54] has a better performance than other methods due to its ability to efficiently handle relatively high dimensional and large-scale data sets without decreasing classification accuracy. In essence, k-Nearest Neighbor makes prediction based on the k training texts which are closest to the test text. It is very simple and effective but not efficient in the case of high dimensional and large-scale data sets. The Decision Tree algorithm is sometimes quite effective but the consequent overfitting problem is intractable and needs to be handled manually case by case. The Naive Bayes method assumes that the terms in one document are independent even this is not the case in the real world. The Neural Network method, usually used in artificial intelligence field has shown lower classification accuracy than other machine learning methods.

1.2 Motivation

In recent decades, with the explosive growth of textual information available in the World Wide Web, the ensuing needs of organizing and accessing these documents in flexible ways also increased. Text categorization is such one solution to this problem, which classifies natural language documents into a predefined set of semantic categories.

An unresolved problem for research on text categorization is how robust the methods are used to tackle problems with a skewed category distribution. Since categories typically have an extremely non-uniform distribution in practice [89], it would be meaningful to compare the performance of different classifiers with respect to category frequencies. Most commonly, methods are compared using a single score, such as the accuracy, error occurrence rate, or averaged F1 measure [89] over all category assignments to documents. A single-valued performance measure can be either dominated by the classifier's performance on common categories or rare categories, depending on how the average performance is computed. Two conventional methods are used to evaluate the performance average across categories. Micro averaging assigns equal weight to every document, while macro averaging assigns equal weight to each category [1]. Inevitably, skewed category distribution often leads to good micro-average performance but not so desirable macro-average performance.

Text representation size for each training category also has a crucial influence on how well the text classifiers can generalize. The purpose the thesis is to improve the accuracy of text categorization by using interactive subspace clustering. Unlike subtopic clustering which utilizes unsupervised learning, subspace clustering adopts supervised learning [94]. Each instance of clustering groups the error data samples into a subcategory and the classification procedure is repeated based on the newly-formed subcategories. The process is repeated interactively.

For the problem of high dimensionality and further improvement of the category boundary, the approach of iterative subspace classification will be investigated. The mathematical assumptions behind the subspace formalism demands that the pattern classes are distributed as low-dimensional subspaces in a higher-dimensional feature space. It is encouraging that subspace approach is suitable for text categorization. However the subspace classification methods have not been popular in text categorization tasks. One possibility may be that the field of data mining has captured the attention of the researchers of unsupervised text categorization.

From the view of classification, we want to re-define a difficult classification boundary possibly due to the use of the initial choice of feature subset. We want to have a better fit by decomposing the data sets into subsets using other more effective features.

1.3 Thesis outline

The thesis is organized into six chapters. Chapter 2, Literature Review, describes related work. Before going into the main topic of Iterative Subspace Method, experiments of Subtopic Clustering are described in Chapter 3 and experiments of Boosting Method are described in Chapter 4. The foundations of text categorization are explained. In particular, through the experiments, we will see how serious the data sparseness problem and topic skewness problem are. Chapter 5, Iterative Subspace Method, presents the scheme of algorithmic components we use, which involve a novel combination of existing techniques for feature selection and categorization. Chapter 6 gives the conclusions drawn from the project.

2 Literature Review

Automatic text categorization systems have been the subject of a great deal of research and a number of different approaches have been used. Text categorization is the task of automatically classifying a text document to one predefined categories (topics). Figure 1 shows the phases of text categorization.

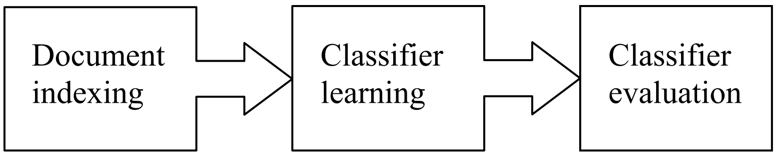


Figure 1: Phases of text categorization.

Text classification has been extensively studied. Most algorithms are based on the bag-of-words model for text [68]. Several methods from simple probabilistic Naive Bayes to the complex Support Vector Machines have been used for text categorization. An inherent problem of text data is its high dimensionality. This 'curse of dimensionality' is a well-known phenomenon in pattern recognition problems. As a consequence of the huge dimensionality of the feature space, data sets are often relatively sparse in this space.

Very little of this work has involved the use of a subspace in the text categorization process. However, this approach has been extensively used in data mining (unsupervised text categorization) [3, 62, 91, 92].

2.1 Phases of Text Categorization

2.1.1 Document Indexing

2.1.1.1 Term Selection

Term selection or Term Space Reduction (TSR) attempts to select, from the original set *T*, the set *T'* of terms. Yang and Pedersen [90] have shown that TSR may even result in a moderate increase in effectiveness, depending on the classifier, on the aggressivity of the reduction, and on the TSR technique used.

Moulinier et al. [59] have used a so-called wrapper approach, that is, one in which *T'* is identified by means of the same learning method that will be used for building the classifier [39]. Starting from an initial term set, a new term set is generated by either adding or removing a term. When a new term set is generated, a classifier based on it is built and then tested on a validation set. The term set that results in the best effectiveness is chosen. This approach has the advantage of being tuned to the learning algorithm being used; moreover, if local dimensionality reduction is performed, different numbers of terms for different categories may be chosen, depending on whether a category is or is not easily separable from the others. However, the sheer size of the space of different term sets makes its cost-prohibitive for standard text categorization applications.

A simple and effective global TSR function is the document frequency of a term, that is, only the terms that occur in the highest number of documents are retained. In a series of experiments Yang and Pedersen [90] have shown that it is possible to reduce the dimensionality by a factor of 10 with no loss in effectiveness (a reduction by a factor of 100 bringing about just a small loss).

Other more sophisticated information-theoretic functions have been used in the literature, such as DIA (Darmstadt Indexing Approach) association factor [20], chi-square [8, 22, 73, 74, 89, 90], NGL coefficient [60, 66], information gain [8, 48], mutual information [53, 66], odds ratio [66], relevancy score [85], and GSS coefficient [22].

2.1.1.2 Term Extraction

Any term extraction method consists in a method for extracting the new terms from the old one, and a method for converting the original document representations into new representations based on the newly synthesized dimensions. Two term extraction methods have been experimented with text categorization, namely term clustering and latent semantic indexing.

Term clustering tries to group words with a high degree of pairwise semantic relatedness, so that the groups may be used instead of the terms as dimensions of the vector space. Term clustering is different from term selection, since the former tends to address terms synonymous with other terms, while the latter targets non-informative terms.[1]

Lewis [50] was the first to investigate the use of term clustering in text categorization. The method he employed, called reciprocal nearest neighbor

[1] Some term selection methods, such as wrapper methods, also address the problem of redundancy.

clustering, consists in creating clusters of two terms that are one the most similar to the other according to some measure of similarity. His results were inferior to those obtained by single-word indexing, possibly due to a disappointing performance by the clustering method.

Li and Jain [53] viewed semantic relatedness between words in terms of their co-occurrence and co-absence within training documents. By using this technique in the context of a hierarchical clustering algorithm, they witnessed only a marginal effectiveness improvement. However, the small size of their experiment hardly allows any definitive conclusion to be reached.

The work of Lewis [50], Li and Jain [53] are examples of unsupervised clustering, since clustering is not affected by category labels attached to the documents. Baker and McCallum [4] provided instead an example of supervised clustering, as the distributional clustering method they employed clusters together those terms that tend to indicate the presence of the same category, or group of categories. Their experiments, carried out in the context of a Naive Bayes classifier showed only a 2% effectiveness loss with an aggressivity of 1,000, and even showed some effectiveness improvement with less aggressive levels of reduction. Later experiments by Slonim and Tishby [75] confirmed the potential of supervised clustering methods for term extraction.

Latent Semantic Indexing (LSI) [12] is a method to reduce the dimension n of the feature space. LSI provides a reduced feature space with m ($<n$) orthogonal axes. This technique compresses document vectors into vectors

of a lower-dimensional space whose dimensions are obtained as combinations of the original dimensions by looking at their patterns of co-occurrence. In text categorization, this technique is applied by deriving the mapping function from the training set and then applying it to training and test documents alike.

For text categorization works that have used LSI or similar term extraction techniques, see Schutze et al. [73], Wiener et al. [85], Hull [29], Li and Jain [53], Schutze [72], Weigend et al. [84], and Yang [87].

2.1.2 Classifier Learning

Joachims first applied Support Vector Machines to text categorization [32]. Although the model of the text used in their framework was a simple Vector Space Model, they achieved an outstanding improvement over other methods. They argue that Support Vector Machines are appropriate for text categorization because Support Vector Machines can handle high dimensional feature spaces and few relevant features, which are main properties of text categorization. Learning methodology is based on Vapnik's statistical learning theory [81].

The Naive Bayes is constructed by using the training data to estimate the probability of a class given the document feature values of a new instance. Naive Bayes classifiers account for most of the probabilistic approaches to text categorization in the literature [32, 50, 53]. Despite the fact that the assumption of conditional independence is generally not true for word appearance in documents, the Naive Bayes classifier is surprisingly effective.

2.1.3 Classifier Evaluation

Standard benchmark collections that can be used as initial corpora for text categorization are publicly available for experimental purposes. The most widely used is the Reuters-21578 collection, consisting of a set of newswire stories classified under categories related to economics. The Reuters collection accounts for most of the experimental work in text categorization so far. Unfortunately, this does not always translate into reliable comparative results, in the sense that many of these experiments have been carried out in different conditions.

Other test collections that have been frequently used are:

1. OHSUMED collection [27]
2. 20 Newsgroups collection [47]

The published experimental results allow us to attempt some considerations on the comparative performance of the text categorization methods discussed. However, we have to bear in mind that comparisons are reliable only when experiments are performed by the same author under carefully controlled conditions. They are instead more problematic when they involve different experiments performed by different authors.

Two different methods may thus be applied for comparing classifiers [89]:

1. Direct comparison

   Classifiers may be compared when they have been tested on the same collection, usually by the same researchers and with the same background conditions. This is the more reliable method.

2. Indirect comparison

   Classifiers may be compared when they have been tested on collections respectively, typically by different researchers and hence with possibly different background conditions; one or more baseline classifiers have been tested on both collections by the direct comparison method. This method is less reliable.

In the literature, inconsistent versions of Reuters-21578 collection ranged from 8,815 training documents to 14,704 training documents and 10 categories to 135 categories are used for performance evaluation (see Table 1). The common condition of Reuters-21578 is 9,603 training documents and 90 categories. Most of the results are focused on improving micro-average performance. Few focused on improving macro-average performance. Between Naive Bayes classifier (NB) and Support Vector Machines classifier, the performance of Support Vector Machines is shown to be better than Naive Bayes.

Table 1: Difference conditions of Reuters-21578 collection are used for performance evaluation.

Results reported by	Reuters-21578 collection				Micro averaging	Macro averaging
	# of documents	# of training documents	# of test documents	# of categories		
Lam et al. 1997	21,450	14,704	6,746	135		✓
Lam and Ho 1998	12,902	9,603	3,299	90	✓	
Dumais et al. 1998	12,902	9,603	3,299	10	✓	
Dumais et al. 1998	12,902	9,603	3,299	90	✓	
Joachims 1998	12,902	9,603	3,299	90	✓ (NB: 0.720) (SVM: 0.864)	
Yang 1999	21,450	14,704	6,746	135	✓	
Yang 1999	14,347	10,667	3,680	93	✓	
Yang 1999	13,272	9,610	3,662	92	✓	
Cohen and Singer 1999	21,450	14,704	6,746	135	✓	
Cohen and Singer 1999	14,347	10,667	3,680	93	✓	
Li and Yamanishi 1999	12,902	9,603	3,299	90	✓ (NB: 0.773) (SVM: 0.841)	
Yang and Liu 1999	12,902	9,603	3,299	90	✓ (NB: 0.795) (SVM: 0.859)	
Takamura and Matsumoto 2002	11,838	8,815	3,023	116	✓ (NB: 0.863) (SVM: 0.890)	
Rogati and Yang 2002	✓ (unclear)	✓ (unclear)	✓ (unclear)	✓ (unclear)	✓	✓

2.2 Curse of Dimensionality

In a small data set, data points/objects are represented by a low number of dimensions and they situate in a low dimensional space. The distance of data points are tightly packed and these data points/objects are non-equidistant from each other. However, when the number of data set increases, the number of dimensions of the data set also increases. It has been shown that in a high dimensional space, the distance between every pair of data points/objects becomes almost the same for a wide variety of data distributions and distance functions. In this case, a large data set creates a high dimensional space, in which data points/objects represented in a high dimensional space spread out and become almost equidistant from each other and distance becomes increasingly meaningless. This is known as the curse of dimensionality [3, 5, 52, 62].

To counter high-dimensionality, various feature/term selection methods have been proposed [5, 52]. Feature/term selection merely selects a ‘good’ subset of the original features/terms; whereas feature/term extraction allows extraction of arbitrary new features/terms based on original ones (see Table 2).

For text categorization at all the reduction levels of aggressiveness from using the full vocabulary as the feature space to removing 98% of the unique terms, Yang [90] reported that *information gain* and *chi-square* were most effective than *document frequency*, *mutual information* and *term strength* in aggressive term removal without losing categorization accuracy in the experiments. *Document frequency* thresholding was found comparable to the performance of *information gain* and *chi-square* with up to 98% term removal, while *term strength* was comparable with up to 50-60% term removal. *Mutual information* has an inferior performance compared to the other methods due to its bias towards rare terms and a strong sensitivity to probability estimation errors. Slonim [75] reported that word clusters (term extraction) had up to 18% improvement in classification accuracy.

Table 2: Different approaches to tackling the problem of high-dimensionality.

Terminology	Term selection	Term extraction	Clustering	Subspace
Latent Semantic Indexing [Schutze 95]	✓ (feature selection)			
Latent Semantic Indexing [Schutze 95]		✓ (Latent Semantic Indexing)		
Cluster-based [Iwayama, 95]			✓ (non-probabilistic clustering, probabilistic clustering)	
Feature selection [Yang, 97]	✓ (e.g. information gain)			
Feature selection [Li 98]	✓ (individual best features)			
Feature extraction [Li 98]		✓ (Principal Component Analysis)		
Term grouping in subspace[Li 98]				✓ (term grouping in subspace)
Subspace [Li 98]				✓ (classification algorithms)
Latent Semantic Indexing [Weigend 99]		✓ (Latent Semantic Indexing)		
Word clustering [Deerwester, 90; Baker, 98; Dhillon 02, Han 03]			✓ (term clustering)	
Feature Clustering [Dhillon ICML-2002]			✓ (term clustering)	
Two-dimensional clustering [Takamura, 02]			✓ (document clustering, term clustering)	

In automatic text categorization by unsupervised learning, subspace clustering [3, 62] is considered an extension of feature/term selection that attempts to find clusters in different subspaces of the same data set.

2.3 Subspace Methodology

Nowadays the subspace methodology has been used extensively in data mining (unsupervised text categorization) [3, 62, 91, 92]. However, this approach has not broadly been applied in the field of supervised text categorization.

Subset selection is to find the best subset among a set of features. The best subset contains the least number of dimensions which attains the highest accuracy. The remaining, unimportant dimensions are discarded.

The history of the subspace methods in data analysis was started by Hotelling [28] in the 1930s. The value of the subspace methods in data compression and optimal reproduction was observed in the 1950s by Kramer and Mathews [44]. Ten years later, Watanabe et al. [82] published the first application in pattern classification. Learning subspace methods emerged from the mid-1970s, after the pioneering work of Kohonen et al. [43]. From the beginning, these methods aimed at classification instead of optimal compression or reproduction. The guiding idea in the learning methods is to modify the bases of the subspaces in order to diminish the reuters corpusreuters corpusnumber of misclassifications. The nature of the modifications varies in different learning algorithms.

2.3.1 Classical Subspace Methods

Classical subspace classification algorithms are reviewed in this section. The style of the notations and illustrations is adopted from Oja [61]. Although there are many variants of the subspace classifier, the most fundamental one is the Class-Featuring Information Compression (CLAFIC) method [61]. The employment of the Principal Component Analysis (PCA), or the Karhunen-Loève (KLT), in classification tasks leads to the CLAFIC algorithm introduced by Watanabe et al. [82]. CLAFIC simply forms the base matrices for the classifier subspaces from the eigenvectors of the class-conditional correlation matrices. For each class j, the correlation matrix $\mathbf{R}_j = E[\mathbf{x}\mathbf{x}^T \mid \mathbf{x} \in j]$ is estimated with $\hat{\mathbf{R}}_j = n_j^{-1} \sum_{i=1}^{n_j} \mathbf{x}_{ij}\mathbf{x}_{ij}^T$. The first l_j

eigenvectors of $\hat{\mathbf{R}}_j$, $\mathbf{u}_{1j},...,\mathbf{u}_{l_j j}$, in the order of decreasing eigenvalue λ_{ij}, are then used as columns of the basis matrix $\mathbf{U}_j$,

$$\mathbf{U}_j = (\mathbf{u}_{ij} \mid (\hat{\mathbf{R}}_j - \lambda_{ij}\mathbf{I})\mathbf{u}_{ij} = \mathbf{0}, \lambda_{ij} \geq \lambda_{(i+1)j}, i = 1,...,l_j) \ , \tag{1}$$

where $\mathbf{0}$ is the zero vector. The sample mean $\hat{\boldsymbol{\mu}}$ of the pooled training set is normally subtracted from the pattern vectors before they are classified or used in initializing the CLAFIC classifier. Because the class-conditional correlations $\mathbf{R}_j$ of the input vectors $\mathbf{x}$ differ from the corresponding class-wise covariances Σ_j, the first eigendirection in each class merely reflects the direction of the class mean from the pooled mean translated to the origin. The calculation of the eigenvalues and eigenvectors of a symmetric positive definite matrix, such as $\hat{\mathbf{R}}_j$, is described, for instance, by Golub and van Loan [23]. The selection of the subspace dimensions $l_1,...,l_c$ is left open in the basic formulation of CLAFIC.

The subspaces that represent two different pattern classes may have a large common sub-subspace. This is problematic because the discrimination between these classes weakens if the subspace dimensions l_j are small. On the other hand, if the subspace dimensions are increased, the classification decisions become dominated by the less robust principal directions. This problem may be avoided if the subspaces are made mutually orthogonal. This leads to a variant of the CLAFIC known as the Method of Orthogonal Subspaces (MOSS) by Kulikowski and Watanabe [45] and Watanabe and Pakvasa [83].

Pairwise orthogonalization of two subspaces is possible whenever their dimensions satisfy the obvious condition $l_i + l_j \geq d$. In that case, two subspaces are said to be mutually orthogonal if any vector of one of the subspaces has zero projection on the other, and vice versa. This is equal to the condition that the basis vectors are orthogonal not only within, but also between, the subspaces. Thus, the projection matrices $\mathbf{P}_i$ and $\mathbf{P}_j$ of two orthogonal subspaces fulfill the condition

$$\mathbf{P}_i \mathbf{P}_j = \mathbf{P}_j \mathbf{P}_i = \mathbf{0} \ , \tag{2}$$

where $\mathbf{0}$ is the zero matrix. The orthogonalization process of MOSS is accomplished by removing the intersections of the subspaces as described, for instance, by Therrien [78]. In short, the projection operators $\mathbf{P}_j$ are replaced with mutually orthogonal operators $\mathbf{P}'_j$, which are formed by using the generating matrix $\mathbf{G}_j$,

$$\mathbf{G}_i = a_j \mathbf{P}_j + \sum_{i=1,\, i \neq j}^{c} a_i (\mathbf{I} - \mathbf{P}_i) \ . \tag{3}$$

The otherwise arbitrary positive multipliers a_j must satisfy the condition $\sum_{j=1}^{c} a_j = 1$. The eigenvalues and eigenvectors are now calculated from $\mathbf{G}_j$, and the orthogonal projection operators $\mathbf{P}'_j$ are formed from the l'_j eigenvectors $\mathbf{v}_{ij}$ which have eigenvalues equal to one,

$$\mathbf{P}'_j = \sum_{i=1}^{l'_j} \mathbf{v}_{ij} \mathbf{v}_{ij}^T \ . \tag{4}$$

Naturally, $\forall j : l'_j \le l_j$. In some cases, the procedure, however, leads to an unacceptable situation where, for some j, $l'_j = 0$, and the corresponding subspace vanishes [40].

Fukunaga and Koontz [21] reasoned that it was necessary to select such basis vectors that the projections on rival subspaces were minimized. Their original formulation of the problem and the criticism against it, presented by Foley and Sammon [14], considered only the two-class case. Instead, the Generalized Fukunaga-Koontz Method (GFK) of Kittler [42] handles an arbitrary number of classes. In the two-class case, the correlation matrices of both classes are first estimated. The KLT is then applied to their sum $\mathbf{Q} = \mathbf{R}_1 + \mathbf{R}_2$ and the eigenvalues λ_i and eigenvectors $\mathbf{u}_i$ are used in defining a transformation matrix $\mathbf{S}$, which is used to transform the original vector $\mathbf{x}$ to $\mathbf{x}'$,

$$\mathbf{S} = \left(\frac{\mathbf{u}_1}{\sqrt{\lambda_1}} \cdots \frac{\mathbf{u}_d}{\sqrt{\lambda_d}} \right) . \tag{5}$$

For the correlation matrix $\mathbf{R}'_j$ of the transformed vector $\mathbf{x}' = \mathbf{S}^T \mathbf{x}$, it holds that $\mathbf{R}'_j = \mathbf{S}^T \mathbf{R}_j \mathbf{S}$, and further $\mathbf{R}'_1 + \mathbf{R}'_2 = \mathbf{I}$. Thus, $\mathbf{R}'_1$ and $\mathbf{R}'_2$ have the same eigenvectors, and the corresponding eigenvalues are positive and sum up to unity. This leads to the following interpretation of the nature of the eigenvectors: When eigenvectors are ordered according to the descending eigenvalues, the first few eigenvectors of $\mathbf{R}'_1$ are optimal for describing the distribution of the transformed vectors $\mathbf{x}'$ which belong to the first class. On the other hand, the eigenvectors with the smallest eigenvalues describe the second class. The method was primarily developed for feature extraction and clustering, but it also lends itself directly to classification.

2.3.2 Current Performance

Four different methods including subspace method for document classification were reported by Li and Jain [53]. The subspace model [61] decomposes a given feature space into m subregions of lower dimensionality (subspace), where each region is a representative feature space for its corresponding pattern class $c_i, i = 1,...,m$. A test document is classified based on a comparison of its compressed representation in each feature space with that of different classes. Experimental results showed that the subspace classifier and the Naive Bayes classifier outperformed the other two classifiers: the nearest neighbour classifier and decision trees based on data sets of seven-class Yahoo news groups. They used the Principal Component Analysis method (LSI) to project the original feature space onto a lower dimensional subspace.

Kharechko et al. [41] reported that they needed to look for some subspace of the bag-of-words vector representation of the text documents for Text Categorization via Ellipsoid Separation. A variant of latent semantic feature extraction was used for the subspace purpose. They demonstrated that the algorithm could perform document classification up to the level of the state-of-the-art Support Vector Machines algorithm.

3 Subtopic Clustering

3.1 Introduction

An unresolved problem for research in Text Categorization (TC) is how robust the methods are used to tackle problems with a skewed category distribution. Since categories typically have an extremely non-uniform distribution in practice [89], it would be meaningful to compare the performance of different classifiers with respect to category frequencies. Most commonly, methods are compared using a single score, such as the accuracy, error occurrence rate, or averaged F1 measure [89] over all category assignments to documents. A single-valued performance measure can be either dominated by the classifier's performance on common categories or rare categories, depending on how the average performance is computed. Two conventional methods are used to evaluate the performance average across categories. Micro averaging assigns equal weight to every document, while macro averaging assigns equal weight to each category [1]. Inevitably, skewed category distribution often leads to good micro-average performance but not so desirable macro-average performance.

To improve the macro-average performance, our approach is to break the large topic classes into subtopic classes [9, 10], similar to the idea of passage-based retrieval [7], because large topics may have been generated by more than one term distribution [77]. The subtopic classes should have a significant amount of terms that occur in documents of the subtopic but not in the other subtopic. We propose to use clustering [25] to find these subtopics of a large topic class as shown in Figure 2. One important issue is to determine which topic classes are larger. This will be addressed by examining the performance with different thresholds to define large topic classes. By comparing the micro-average performance and macro-average performance before and after clustering, it is possible to identify if subtopic clustering has generated any positive result on the macro-average performance.

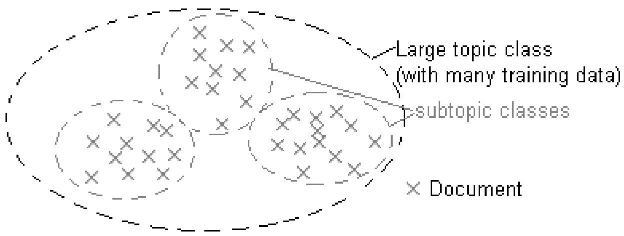


Figure 2: Visual representation of a large topic class consists of a mixture of a number of subtopic clusters.

In Section 3.2, we shall briefly describe the methodology for experimental setup and performance measure. This will be followed by results and discussion in Section 3.3. Lastly, conclusion and future work will be drawn in Section 3.4.

3.2 Methodology

3.2.1 Experimental Setup

3.2.1.1 Data Set

The Reuters-21578 document set has previously been regarded as a standard real-world benchmarking corpus for the Information Retrieval (IR) community. The ModApte split (training data set: 9,603 documents, test data set: 3,299 documents, unused: 8,676 documents) of Reuters-21578 document set is used for our experiments.

Except two large topics, including “acq” (1,488 training documents) and “earn” (2,709 training documents), the rest of the training topics have fewer than 500 documents (ranging from 1 to 460). Test documents can be assigned to more than one topic; therefore, 3,299 single-label test documents are expanded to 3,409 test documents which are used for evaluation.

The distribution of the number of training documents in a topic class is typically highly skewed. The number of terms in a topic increases logarithmically with an increase in the number of training documents. They are shown in Figure 3.

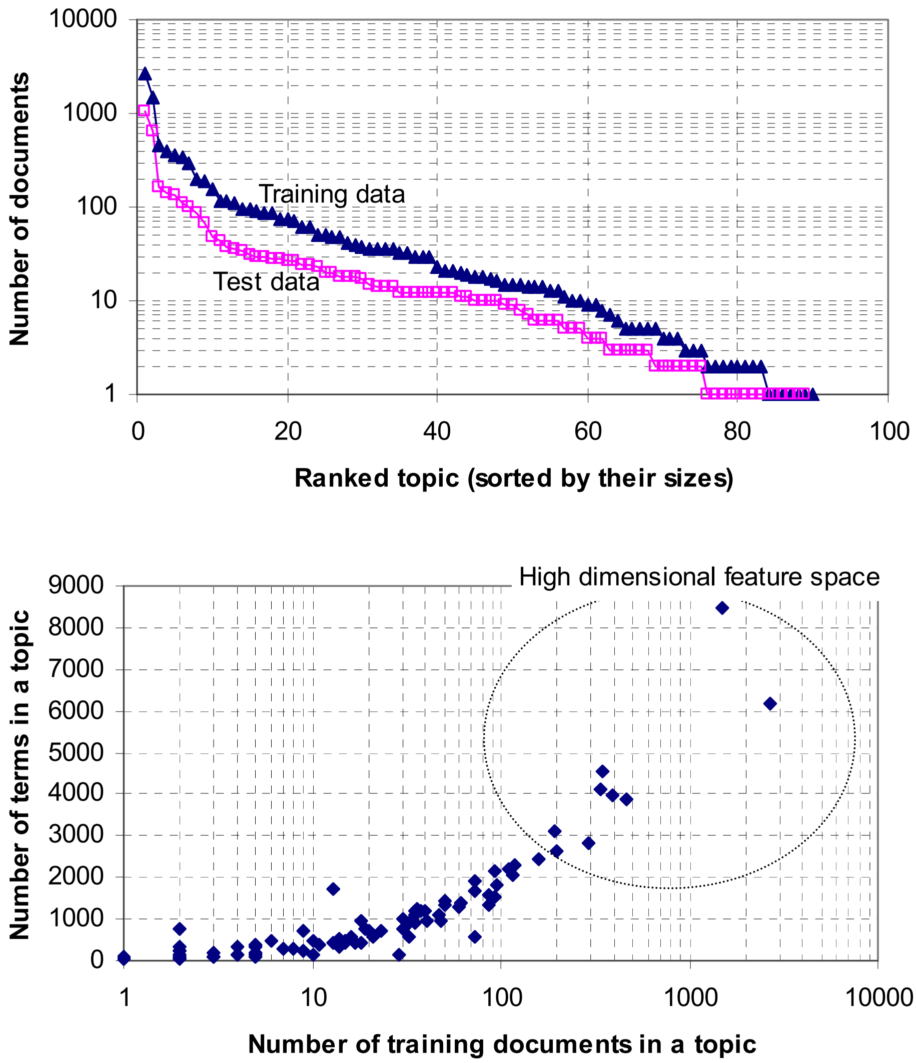


Figure 3: The number of training/test documents plotted against ranked topic sorted by their sizes (top). The number of terms in a topic plotted against the number of training documents in its topic (bottom).

3.2.1.2 Preprocessing

Preprocessing involves removing SGML tags, punctuation marks, stop words and performing word stemming to reduce the feature vector size. Bag-of-words [57] document representation (vector space model) scheme is used for feature representation. Term importance is assumed to be inversely proportional to the number of documents a particular term appears in. The term frequency (*tf*) and inverse document frequency (*idf*) are used to assign

weights to terms. The inverse document frequency for term t is defined as [67]:

$$idf(t) = \log(N / n(t)) \ . \tag{6}$$

The common non-content words are removed to reduce possible interference in classification results. It is assumed that the importance of a term increases with its use-frequency. Combining these two assumptions lead to *tfidf*:

$$tfidf(t) = tf(t) \times idf(t) \ . \tag{7}$$

Cosine normalization is used. Every document vector is divided by its Euclidean length, $((w_1)^2 + (w_2)^2 + \ldots + (w_n)^2)^{1/2}$, where w_i is the *tfidf* weight of the *i*-th term in the document. The final weight for a term hence becomes:

$$\frac{tfidf \text{ weight}}{\text{Euclidean length of the document vector}} \ . \tag{8}$$

3.2.1.3 Classifier

Instead of implementing a classifier, we use Rainbow/Libbow software package [55, 56] to perform text classification. The classifier utilizes machine learning methods such as Naive Bayes, Support Vector Machines and k-Nearest Neighbor for text classification [32, 88, 89]. As the major focus of this paper is not about the performance of classifier algorithms, only Support Vector Machines classifier for single-label classification was selected for the following experiments. Scores of performance measurements generated by the classifier will be shown in the following section.

3.2.2 Performance Measurements

3.2.2.1 Recall, Precision and F1

Classification performance is measured by both recall and precision. For evaluating the performance, three quantities are of interest for each topic.

They are: a = the number of documents correctly assigned to this topic.

b = the number of documents incorrectly assigned to this topic.

c = the number of documents incorrectly rejected from this topic.

From these quantities, we define the following performance measures:

$$\text{recall} = a/(a+c) \quad . \tag{9}$$

$$\text{precision} = a/(a+b) \quad . \tag{10}$$

In addition, we use F1 measure [79], combining recall and precision with equal weighting, to compare the overall results of the algorithms:

$$\text{F1} = (2 \times \text{recall} \times \text{precision})/(\text{recall} + \text{precision}) \quad . \tag{11}$$

Macro-average performance scores are determined by first computing the performance measures per topic and then averaging these to compute the global means. Micro-average performance scores are determined by first computing the totals of a, b and c for all topics and then these totals are used to compute the performance measures. There is an important distinction between the two types of averaging. Micro averaging gives equal weight to every document, while macro averaging gives equal weight to each topic.

For a sample test data set containing 3,409 test documents, the measurements of recall, precision and F1 plotted against the training document number of 90 topics and against ranked topic (sorted by their scores from the smallest value to the largest) are shown in Figure 4. It is observed that 61 out of 90 topics are having both recall and precision zero. The percentage of topics not classified correctly is 67.78%.

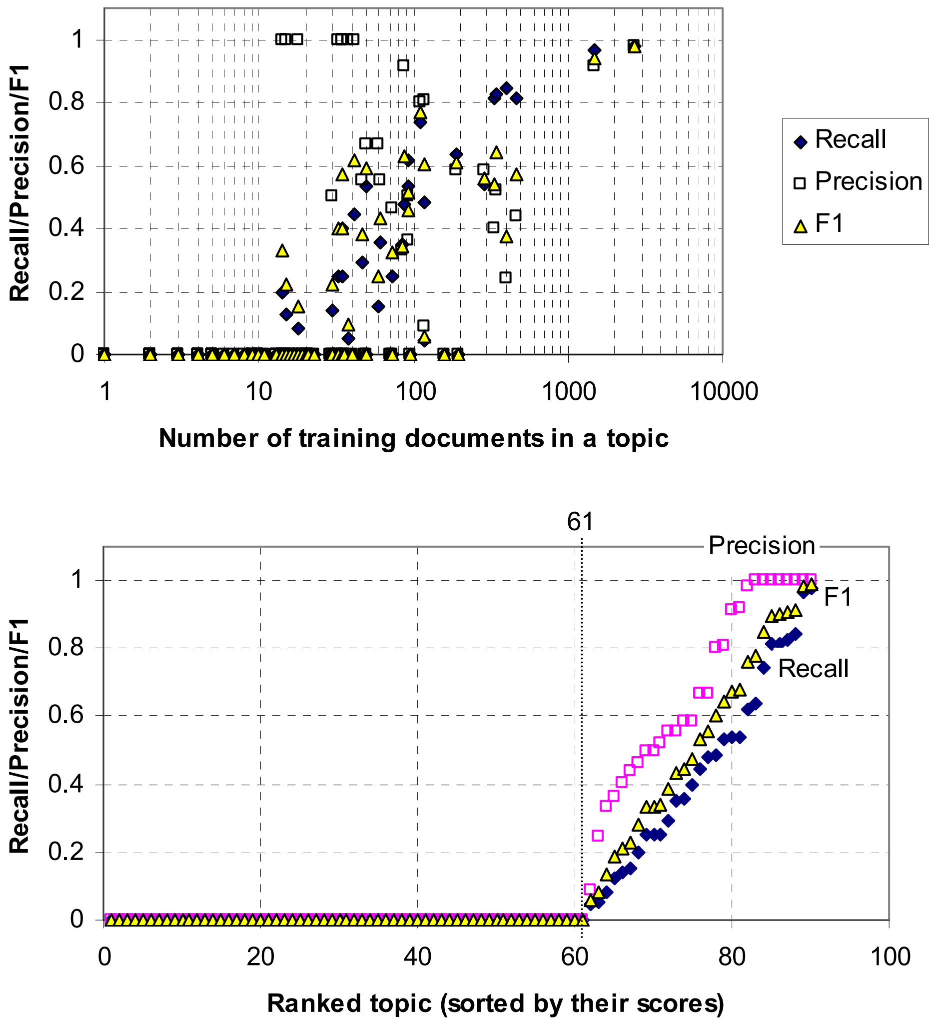


Figure 4: The distribution of recall/precision/F1 measurement plotted against the number of training documents in a topic (top). The distribution of recall/precision/F1 measurement plotted against ranked topic sorted by their scores (bottom).

Recall, precision and F1 measurement of the 90 topics in the experimental data set are unevenly distributed. The uneven distribution is due to the fact that the distribution of the number of documents in the data set is highly skewed in nature. The results of macro-average and micro-average are shown in Table 3. From the result, the macro-average recall is 14.84%, macro-average precision is 22.35% and macro-average F1 is 17.84%. The reason for this low score is due to the fact that more than half of the topics (67.78%) in the data set are zero in both recall and precision.

Table 3: The macro-average and micro-average performance calculated by a sample test data set containing 3,409 test documents.

Macro-average			Micro-average
Recall	Precision	F1	Recall/Precision/F1
14.84%	22.35%	17.84%	69.26%

All numerical values for *a*, *b,* and *c* in Equations 9-11 are listed underneath for both macro- and micro-averages.

Topic	*a*	*b*	*c*	*Recall (%)*	*Precision (%)*	*F1 (%)*
acq	622	59	21	96.73	91.34	93.96
alum	5	0	15	25.00	100.00	40.00
barley	0	0	12	0.00	0.00	0.00
bop	10	8	18	35.71	55.56	43.48
carcass	0	0	18	0.00	0.00	0.00
castor-oil	0	0	1	0.00	0.00	0.00
cocoa	8	4	7	53.33	66.67	59.26
coconut	0	0	2	0.00	0.00	0.00
coconut-oil	0	0	3	0.00	0.00	0.00
coffee	20	5	7	74.07	80.00	76.92
copper	5	4	12	29.41	55.56	38.46
copra-cake	0	0	1	0.00	0.00	0.00
corn	0	1	48	0.00	0.00	0.00
cotton	1	0	19	5.00	100.00	9.52
cotton-oil	0	0	2	0.00	0.00	0.00
cpi	4	2	22	15.38	66.67	25.00
crude	133	122	28	82.61	52.16	63.94
dfl	0	0	1	0.00	0.00	0.00
dlr	0	0	31	0.00	0.00	0.00
dmk	0	0	3	0.00	0.00	0.00
earn	1021	20	23	97.80	98.08	97.94
fuel	0	0	10	0.00	0.00	0.00
gas	2	2	12	14.29	50.00	22.22
gnp	21	37	13	61.76	36.21	45.65
gold	15	15	13	53.57	50.00	51.72
grain	113	352	21	84.33	24.30	37.73
groundnut	0	0	4	0.00	0.00	0.00
groundnut-oil	0	0	1	0.00	0.00	0.00
heat	1	0	4	20.00	100.00	33.33
hog	0	0	6	0.00	0.00	0.00
housing	0	0	3	0.00	0.00	0.00
income	0	0	5	0.00	0.00	0.00
instal-debt	0	0	1	0.00	0.00	0.00
interest	54	38	46	54.00	58.70	56.25
ipi	4	0	6	40.00	100.00	57.14

iron-steel	0	0	12	0.00	0.00	0.00
jet	0	0	1	0.00	0.00	0.00
jobs	8	0	10	44.44	100.00	61.54
l-cattle	0	0	2	0.00	0.00	0.00
lead	0	0	14	0.00	0.00	0.00
lei	0	0	2	0.00	0.00	0.00
lin-oil	0	0	1	0.00	0.00	0.00
livestock	6	7	18	25.00	46.15	32.43
lumber	0	0	6	0.00	0.00	0.00
meal-feed	0	0	18	0.00	0.00	0.00
money-fx	115	147	26	81.56	43.89	57.07
money-supply	11	1	12	47.83	91.67	62.86
naphtha	0	0	4	0.00	0.00	0.00
nat-gas	0	0	29	0.00	0.00	0.00
nickel	0	0	1	0.00	0.00	0.00
nkr	0	0	1	0.00	0.00	0.00
nzdlr	0	0	2	0.00	0.00	0.00
oat	0	0	6	0.00	0.00	0.00
oilseed	2	21	42	4.55	8.70	5.97
orange	1	0	7	12.50	100.00	22.22
palladium	0	0	1	0.00	0.00	0.00
palm-oil	0	0	10	0.00	0.00	0.00
palmkernel	0	0	1	0.00	0.00	0.00
pet-chem	0	0	12	0.00	0.00	0.00
platinum	0	0	6	0.00	0.00	0.00
potato	0	0	3	0.00	0.00	0.00
propane	0	0	3	0.00	0.00	0.00
rand	0	0	1	0.00	0.00	0.00
rape-oil	0	0	3	0.00	0.00	0.00
rapeseed	0	0	9	0.00	0.00	0.00
reserves	0	0	14	0.00	0.00	0.00
retail	0	0	2	0.00	0.00	0.00
rice	0	0	24	0.00	0.00	0.00
rubber	3	0	9	25.00	100.00	40.00
rye	0	0	1	0.00	0.00	0.00
ship	54	38	31	63.53	58.70	61.02
silver	0	0	7	0.00	0.00	0.00
sorghum	0	0	10	0.00	0.00	0.00
soy-meal	0	0	12	0.00	0.00	0.00
soy-oil	0	0	11	0.00	0.00	0.00
soybean	0	0	29	0.00	0.00	0.00
strategic-metal	0	0	11	0.00	0.00	0.00
sugar	17	4	18	48.57	80.95	60.71
sun-meal	0	0	1	0.00	0.00	0.00
sun-oil	0	0	2	0.00	0.00	0.00
sunseed	0	0	5	0.00	0.00	0.00
tea	0	0	4	0.00	0.00	0.00
tin	1	0	11	8.33	100.00	15.38
trade	91	135	21	81.25	40.27	53.85
veg-oil	13	26	24	35.14	33.33	34.21
wheat	0	0	66	0.00	0.00	0.00
wpi	0	0	9	0.00	0.00	0.00
yen	0	0	12	0.00	0.00	0.00
zinc	0	0	12	0.00	0.00	0.00

3.2.2.2 Skewness

Skewness is measured against the number of test data sets. Each test data set (consists of test documents) has the skewness, and its own scores (such as recall and precision) are calculated by the classifier.

The skewness is calculated by Kullback-Leibler (KL) distance [46]. Suppose two variables of the same type characterized by their probability distribution *f* and *f'*. The skew distance (KL distance) can be derived using as:

$$\text{skew distance} = \sum_{i=1}^{t} f_i(x) \times \log \frac{f_i(x)}{f_i'(x)} \ , \qquad (12)$$

where t is the number of topics, f is the probability distribution of test documents of the topics and f' is the equal probability distribution of test documents of the topics. For a data set containing of 90 topics, the skew distance is calculated as:

$$\text{skew distance} = \sum_{i=1}^{90} f_i(x) \times \log \frac{f_i(x)}{\frac{1}{90}} \ . \qquad (13)$$

$$f_i(x) = \frac{\text{number of test documents from topic } (i) \text{ in the test data set}}{\text{number of test documents from all topics in the test data set}} \qquad (14)$$

For skewness measurement, we use 925 test data sets where 100 test documents in each test data set are selected randomly from 3,409 test documents. Each test data set has it own skew distance. Figure 5 shows the histogram of skew distance of the 925 test data sets.

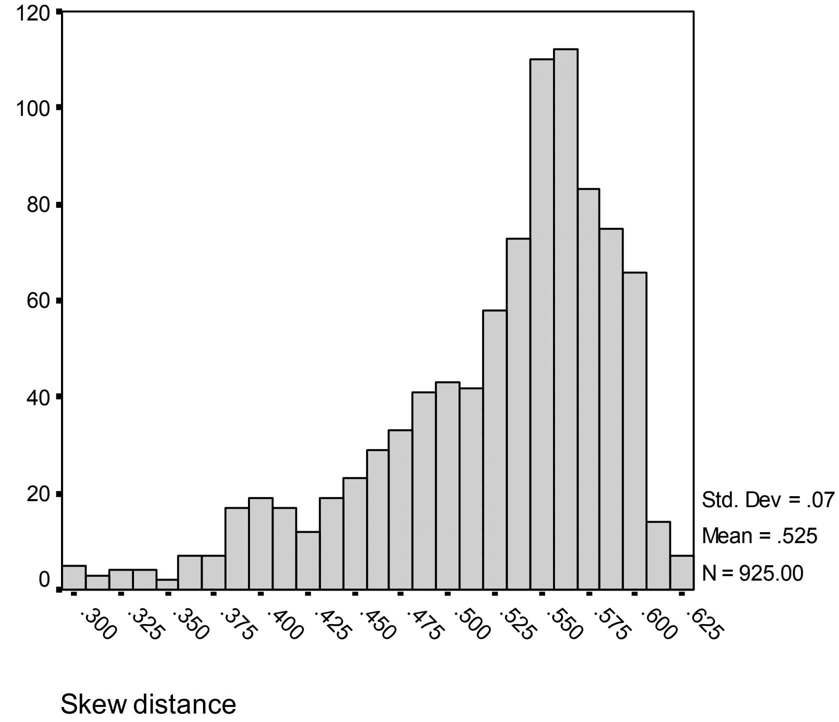
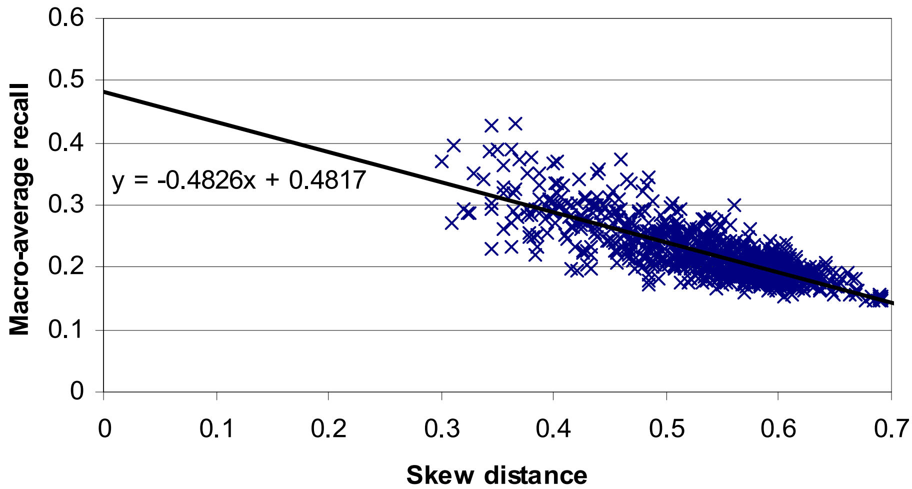
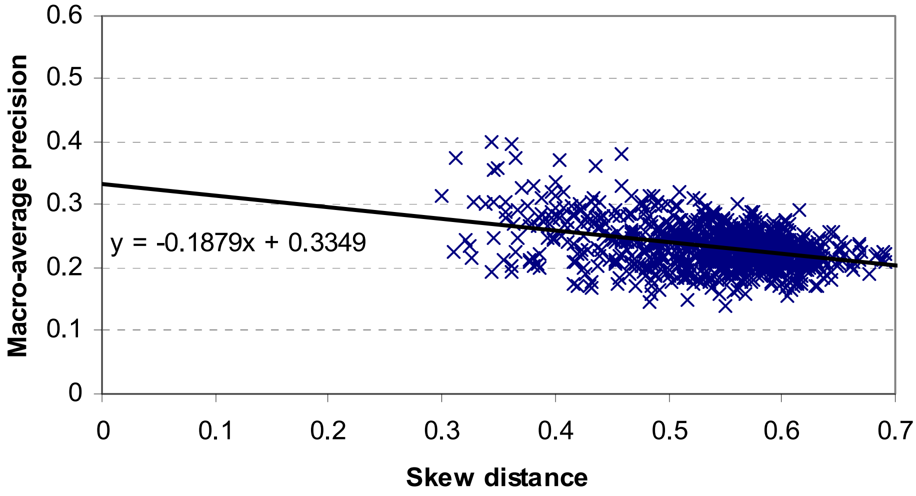


Figure 5: The histogram of skew distance of 925 test data sets. 100 test documents in each test data set are selected randomly from 3,409 test documents.

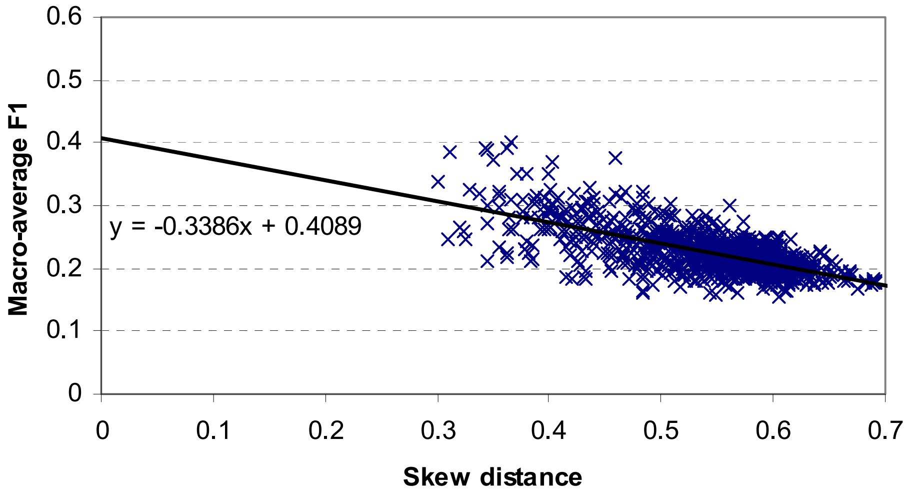
For the 925 test data sets (925 skew distances), the scores of recall, precision and F1 are plotted against the skew distance. The scatter plots are shown in Figure 6. On these plots, linear regression lines are drawn to predict the values at different skew distances. Zero skew distance is used as the reference point. The results at zero skew distance are shown in Table 4.



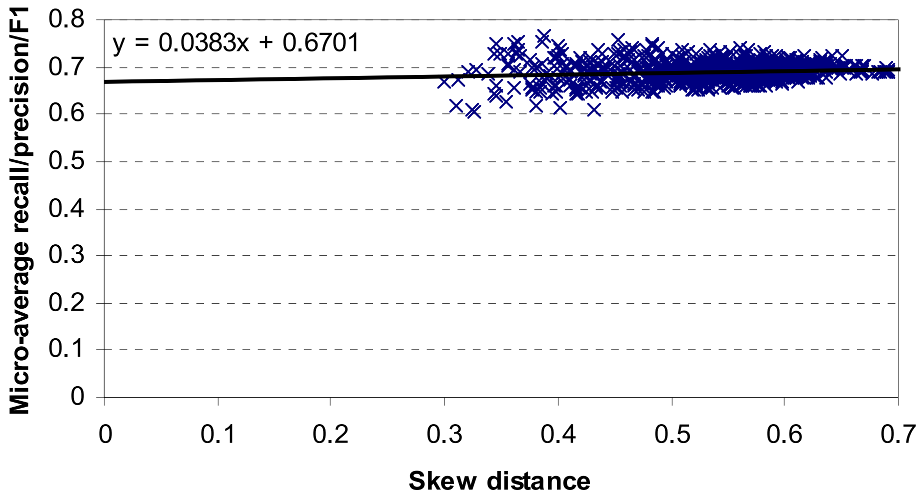
(a)


0.6
0.5
0.4
0.3
0.2
0.1
0
Macro-average precision
y = -0.1879x + 0.3349
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Skew distance

(b)


0.6
0.5
0.4
0.3
0.2
0.1
0
Macro-average F1
y = -0.3386x + 0.4089
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Skew distance

(c)



(d)

Figure 6: Macro-average recall plotted against skew distance (a). Macro-average precision plotted against skew distance (b). Macro-average F1 plotted against skew distance (c). Micro-average recall/precision/F1 plotted against skew distance (d).

Table 4: The macro-average and micro-average performance at zero skew distance.

Macro-average			Micro-average
Recall	Precision	F1	Recall/Precision/F1
48.17%	33.49%	40.89%	67.01%

3.2.3 Clustering

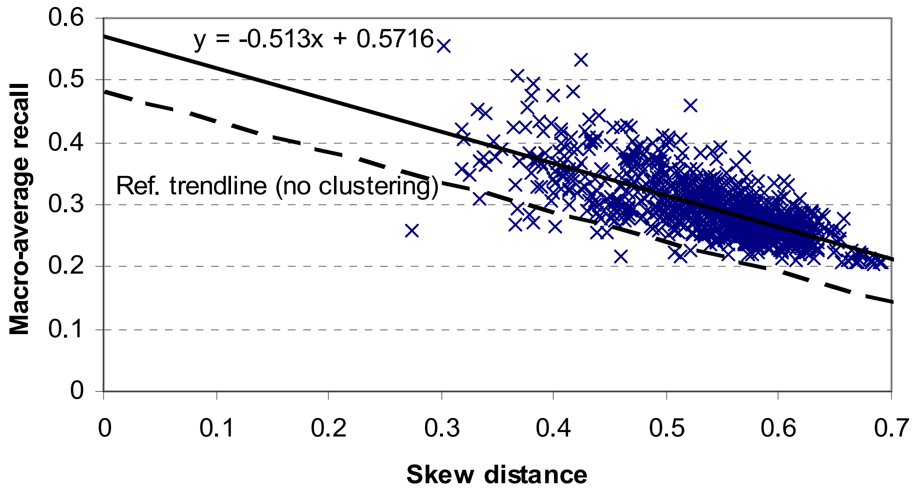
By viewing topics as clusters in a high dimensional space, we propose the use of clustering to determine subtopic clusters for large topic classes by assuming that large topic clusters are in general a mixture of a number of subtopic clusters.

The cluster analyses (hierarchical and non-hierarchical clustering) in this paper are conducted by SPSS [76]. For each topic to be clustered into subtopics, all document vectors are initially grouped together to form a document-by-word matrix with size *m* by *n* (*m* is the number of documents and *n* is the size of document vector).

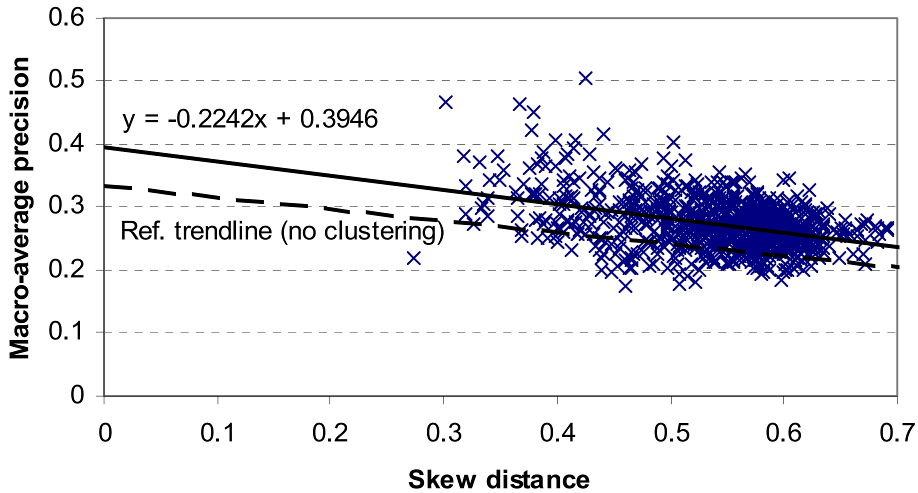
Topics with topic size which generates optimal macro-average performance (in Section 3.3.1) are selected for our experiments. For demonstration purpose, topics with topic size exceeding 100 are selected for clustering. Within the 90-topic data set, 77 topics have the number of training documents less than or equal to 100. Hence, only 13 topics meet our experimental criteria are selected for subtopic clustering. By means of complete linkage hierarchical clustering, 13 topics are clustered into 1,148 subtopics. The total number of topics and subtopics are 1,225 (77+1,148). By means of k-means non-hierarchical clustering, 13 topics have been clustered into 701 subtopics. The total number of topics and subtopics are 778 (77+701). The classifier is trained on these topics for performance evaluation. The clustered scores are compared with the previous result without subtopic clustering, by mapping clustered subtopics onto previous non-clustered topics after classification.

3.2.3.1 Hierarchical Clustering

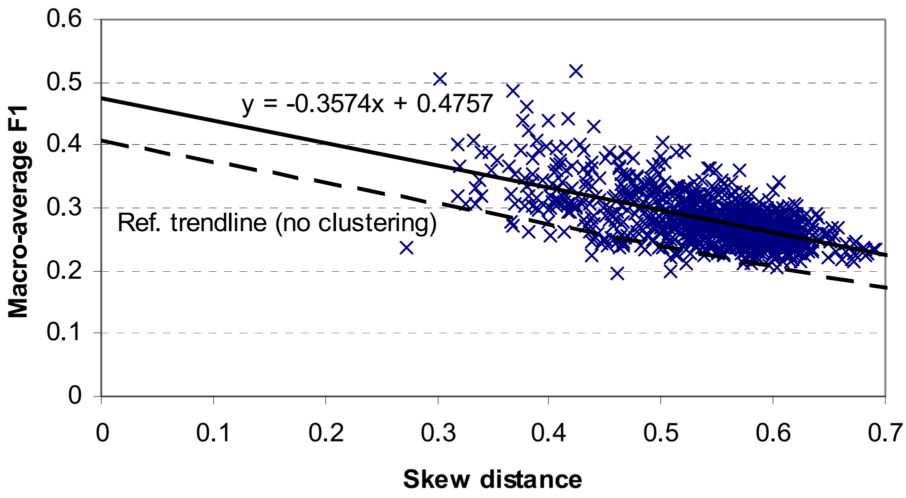
The scores of recall, precision and F1 are plotted against the skew distance. The scatter plots are shown in Figure 7. On these plots, linear regression lines are drawn to predict the values at different skew distances (zero skew distances are used as the reference point). The dotted lines are linear regressions showing the projected trends of micro-average and macro-average performance at different skew distances before subtopic clustering. Hence, the differences between the dotted and the solid lines in the graphs below demonstrate the difference in macro-average and micro-average performance before and after hierarchical clustering. Table 5 demonstrates the performance at zero skew distance.



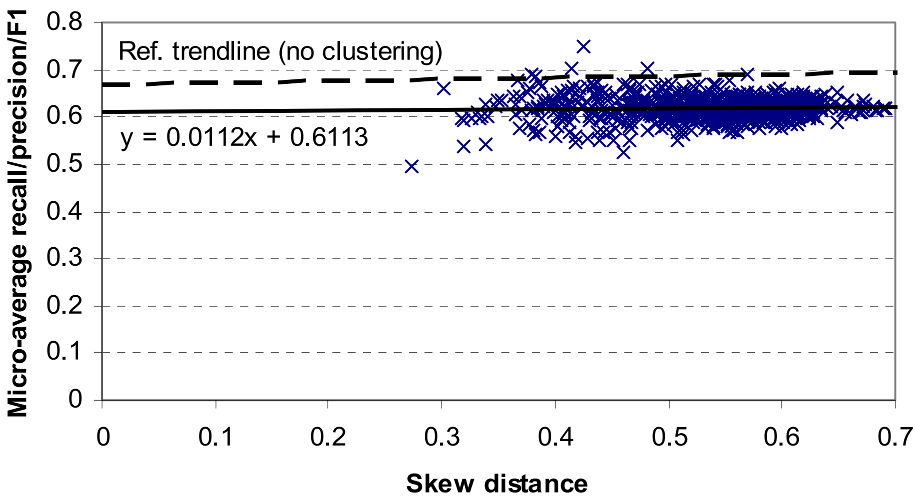
(a)



(b)



(c)



(d)

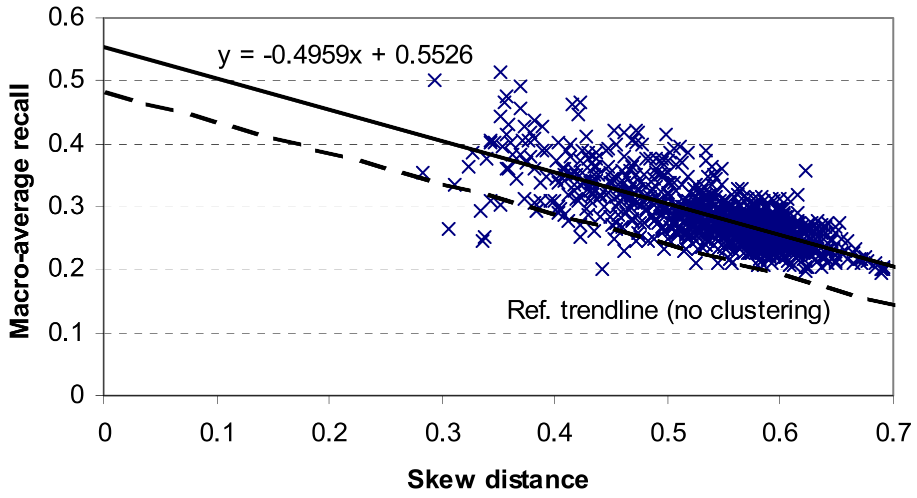
Figure 7: Macro-average recall plotted against skew distance for hierarchical clustering (a). Macro-average precision plotted against skew distance for hierarchical clustering (b). Macro-average F1 plotted against skew distance for hierarchical clustering (c). Micro-average recall/precision/F1 plotted against skew distance for hierarchical clustering (d).

Table 5: The macro-average and micro-average performance at zero skew distance from the 925 test data sets (using subtopics by complete-linkage clustering to build the classifier).

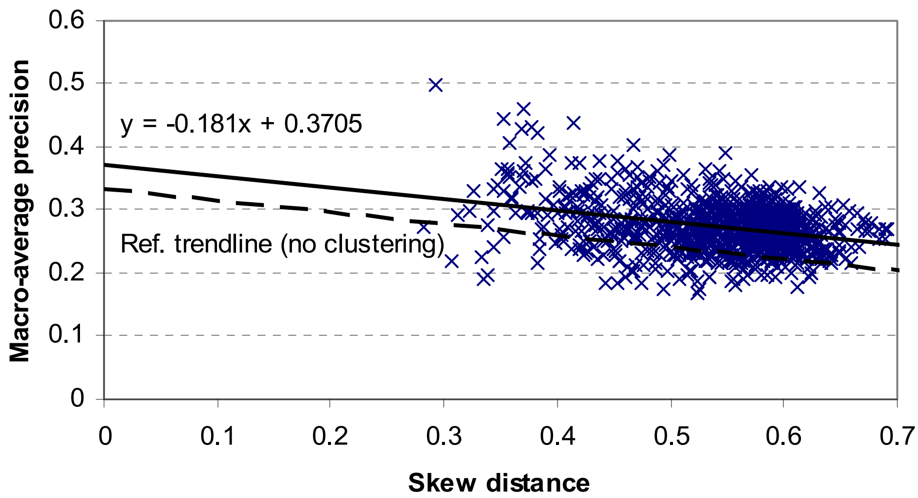
Macro-average			Micro-average
Recall	Precision	F1	Recall/Precision/F1
57.16%	39.46%	47.57%	61.13%

3.2.3.2 Non-hierarchical Clustering

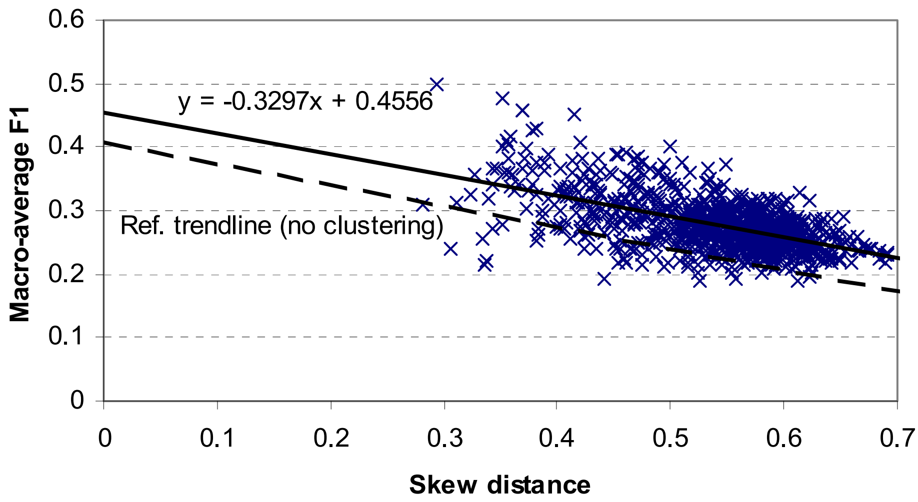
Non-hierarchical Clustering is conducted following the same procedure as Hierarchical Clustering. The scatter plots are shown in Figure 8 and Table 6 demonstrates the performance at zero skew distance.



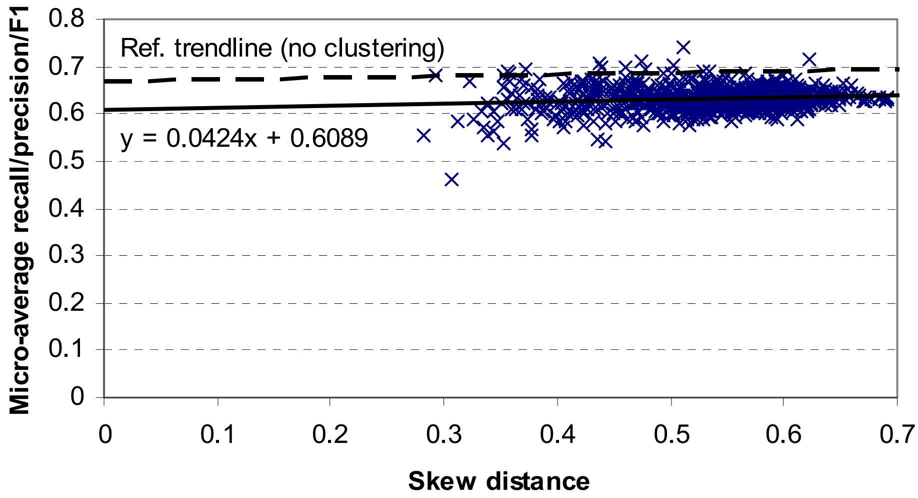
(a)



(b)



(c)



(d)

Figure 8: Macro-average recall plotted against skew distance for non-hierarchical clustering (a). Macro-average precision plotted against skew distance for non-hierarchical clustering (b). Macro-average F1 plotted against skew distance for non-hierarchical clustering (c). Micro-average recall/precision/F1 plotted against skew distance for non-hierarchical clustering (d).

Table 6: The macro-average and micro-average performance at zero skew distance from the 925 test data sets (using subtopics by k-means clustering to build the classifier).

Macro-average			Micro-average
Recall	Precision	F1	Recall/Precision/F1
55.26%	37.05%	45.56%	60.89%

3.3 Experimental Results and Discussion

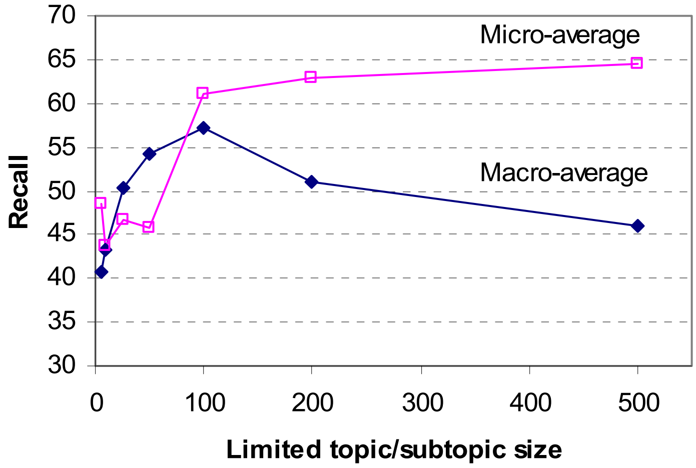
The comparison results of macro averaging and micro averaging at different cluster sizes by complete-linkage clustering are discussed in Section 3.3.1. They are calculated from the 925 test data sets at skew distance equals to 0. For macro-average performance, the optimal result is obtained when the maximum subtopic class size is set to 100.

We have also evaluated whether the complete-linkage clustering is better than k-means clustering. In Section 3.3.2, the macro-average and the micro-average result with clustering and without clustering are summarized and compared. The results are also calculated from the 925 test data sets at skew distance equals to 0.

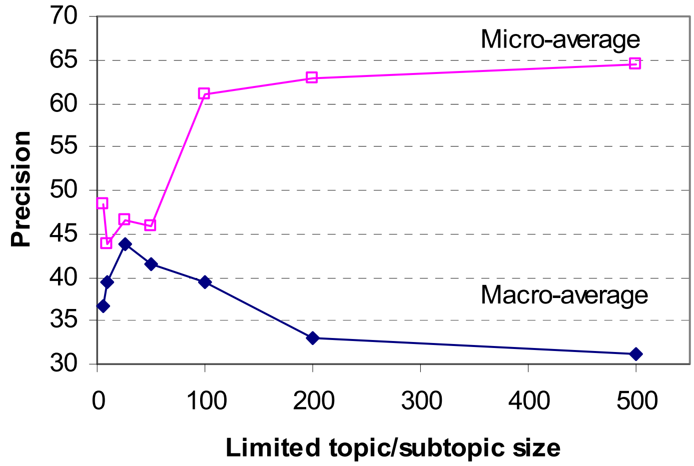
In Section 3.3.3, the percentages of topics never be classified correctly are summarized with subtopic clustered by complete-linkage clustering and k-means clustering. The scores are calculated from the sample test data set containing 3,409 test documents.

3.3.1 Comparison of Macro Averaging and Micro Averaging at Different Cluster Sizes by Complete-Linkage Clustering

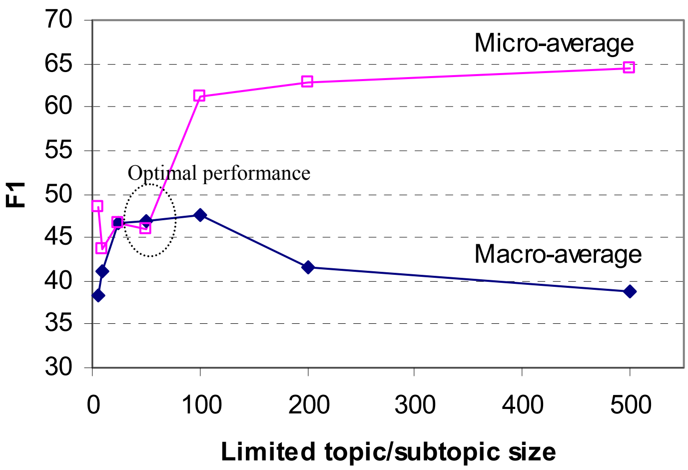
To investigate the effect of topic/subtopic size, training documents with cluster-sizes limited to 5, 10, 25, 50, 100, 200 and 500 are classified by complete-linkage clustering. Figure 9 shows the scatter plots and Table 7 shows the performance of the classifier with subtopic clustering for different maximum subtopic class sizes.


Micro-average
Macro-average
Recall
70
65
60
55
50
45
40
35
30
0
100
200
300
400
500
Limited topic/subtopic size

(a)


Micro-average
Macro-average
Precision
70
65
60
55
50
45
40
35
30
0
100
200
300
400
500
Limited topic/subtopic size

(b)



(c)

Figure 9: Macro-average recall and micro-average recall plotted against limited topic/subtopic size by using complete-linkage method (a). Macro-average precision and micro-average precision plotted against limited topic/subtopic size by using complete-linkage method (b). Macro-average F1 and micro-average F1 plotted against limited topic/subtopic size by using complete-linkage method (c).

In general, the optimal macro-average performance (F1 measurement is 47.57%) is attained when the topic size is 100. However, at a certain point when the topic size is below 100, the macro-average performance and the micro-average performance nearly coincides (i.e. theirs scores are almost the same). Under such circumstance, over-clustering is likely to occur and adversely affect the macro-average and micro-average performance.

The best micro-average performance is achieved by using the classifier without subtopic clustering, mainly due to the benefit of large topics.

Table 7: The results from the 925 test data sets (at skew distance = 0) using complete-linkage clustering with topic/subtopic size limited to 5, 10, 25, 50, 100, 200 and 500 are summarized.

Subtopic size limited to	Macro-average			Micro-average
	Recall	Precision	F1	Recall/Precision/F1
5	40.64%	36.72%	38.35%	48.43%
10	43.31%	39.48%	41.04%	43.70%
25	50.36%	43.90%	46.71%	46.59%
50	54.16%	41.42%	46.91%	45.88%
100	57.16%	39.46%	47.57%	61.13%
200	51.00%	32.98%	41.64%	62.82%
500	46.02%	31.09%	38.73%	64.44%
No clustering	48.17%	33.49%	40.89%	67.01%

3.3.2 Comparison of Macro Averaging and Micro Averaging by Complete-Linkage Clustering and K-Means Clustering

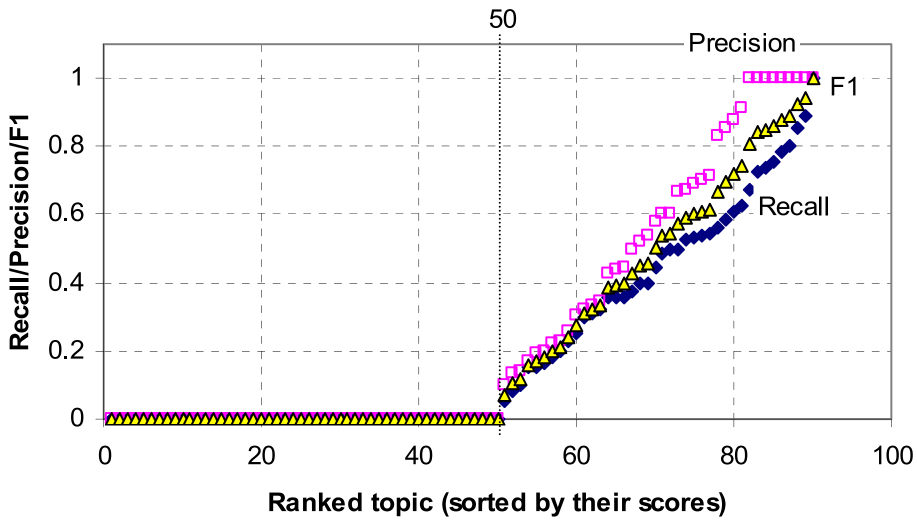
The macro-average and micro-average result calculated from the 925 test data sets at zero skew distance using complete-linkage and k-means clustering with topic/subtopic size limited to 100 are summarized in Table 8. It shows that complete-linkage clustering performs better regardless of all performance measures. While we have to accept that hierarchical clustering, such as complete-linkage, provides better performance than non-hierarchical clustering, as it is able to locate the cluster boundaries more accurately and create a higher performance in text categorization.

Table 8: The results from the 925 test data sets (at skew distance = 0) using complete-linkage clustering and k-means clustering with topic/subtopic size limited to 100 are summarized.

Clustering method	Macro-average			Micro-average
	Recall	Precision	F1	Recall/Precision/F1
No clustering	48.17%	33.49%	40.89%	67.01%
Complete-linkage	57.16%	39.46%	47.57%	61.13%
K-means	55.26%	37.05%	45.56%	60.89%

3.3.3 Comparison of Percentage of Topics with Zero Recall and Precision

The scores are calculated from a sample test data set containing 3,409 test documents. The measurements of recall, precision and F1 plotted against ranked topic (sorted by their scores from the smallest value to the largest) using complete-linkage clustering and k-means clustering are shown in Figure 10. The results are summarized in Table 9 and show that the classifier with subtopic clustering by complete-linkage method has 18.03% improvement while the result by k-means method has 16.39% improvement. Again it shows that complete-linkage clustering performs better than k-means clustering.



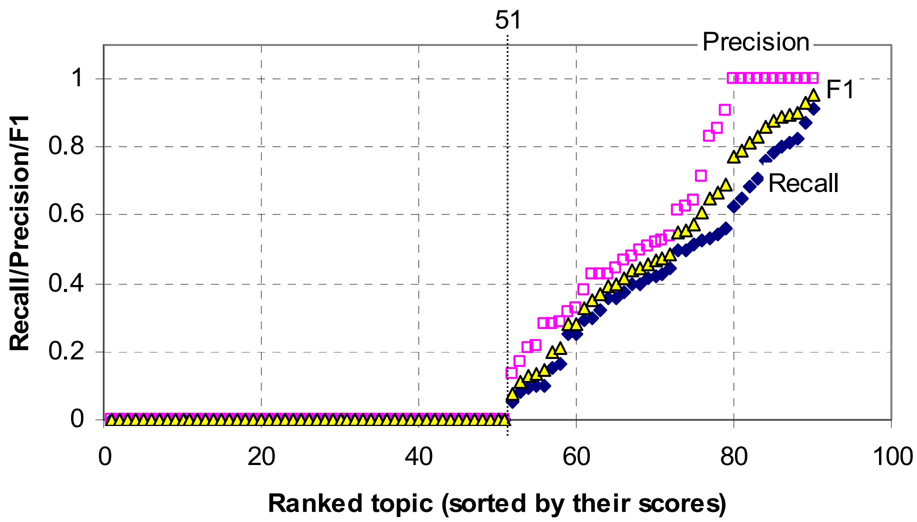


Figure 10: The distribution of recall/precision/F1 measurement plotted against ranked topic sorted by their scores using complete-linkage clustering with topic/subtopic size limited to 100 (top). The distribution of recall/precision/F1 measurement plotted against ranked topic sorted by their scores using k-means clustering with topic/subtopic size limited to 100 (bottom).

Table 9: The percentages of topics that have never been classified correctly are summarized (without subtopic, with subtopic clustered by complete-linkage clustering and with subtopic clustered by k-means clustering).

Clustering method	Topics that have never been classified correctly	Improvement
No clustering	67.78% (61 out of 90)	-
Complete-linkage	55.56% (50 out of 90)	18.03%
K-means	56.67% (51 out of 90)	16.39%

3.3.4 Comparison with Feature Reduction

Since document classification involves high-dimensional feature space, the effects of different feature reduction techniques were examined in order to improve recognition performance [53]. It is a well-known fact that the size of different text categories can vary significantly in text corpora. The Reuters-21578 collection is a common benchmark for comparing methods of text categorization [1, 13, 32, 49, 71, 88, 89]. The documents in the Reuters collection were collected from Reuters newswire in 1987. Over one third of the text classes are having less than 10 documents in the Reuters-21578 [1, 89]. The skewness problem cannot be eliminated by replacing with a larger data set corpora like the Reuters Corpus Volume 1 (RCV1) [51], i.e. the uneven distribution of document sizes of topics within a data set will always occur, and may subsequently introducing problems for text categorization.

Further experiments on feature reduction are done on the same data set to evaluate the performance. For feature reduction, only the top 500 feature weights of a topic (calculated by *tfidf*) are selected. The feature reduction vector ($\boldsymbol{x}'$) is reduced from the original vector ($\boldsymbol{x}$).

$$\boldsymbol{x}' = \{x_i'\}_{i=1}^{500} = \{x_1', x_2', \ldots, x_{500}'\}$$

where $x_1' = \max\{x_i\}_{i=1}^{500}$ and $x_i' \geq x_{i-1}' \quad \forall i = \{2,\ldots,500\}$

Different experiments of feature reduction topic are selection for the comparison. First, topics with training topic size greater than 500 (>500) are used for feature reduction; then using other training topic sizes such as 200, 100, 50, 25, 10 and 5 (as shown in Table 10)

Table 10: The numbers of topic used for feature reduction are chosen based on training topic size.

Topic with training topic size	Numbers of topic used for feature reduction
> 500	2
> 200	7
> 100	13
> 50	23
> 25	39
> 10	57
> 5	67

Each classifier result is built by 90 topics with different numbers of topic used for feature reduction. The corresponding macro-averaging and micro-averaging scores are summarized in Table 11, the scatter plots are shown in Figure 11.

Table 11: The macro-averaging and micro-averaging scores of the 7 feature reduction classifiers.

Topic with training topic size	Macro-average Recall	Macro-average Precision	Macro-average F1	Micro-average Recall / Precision / F1
> 500	18.10%	26.75%	21.59%	65.68%
> 200	21.84%	26.80%	24.07%	65.18%
> 100	23.31%	27.26%	25.13%	65.30%
> 50	23.80%	25.86%	24.79%	66.38%
> 25	23.46%	24.68%	24.05%	66.21%
> 10	22.73%	24.54%	23.60%	65.97%
> 5	22.74%	24.68%	23.67%	66.03%

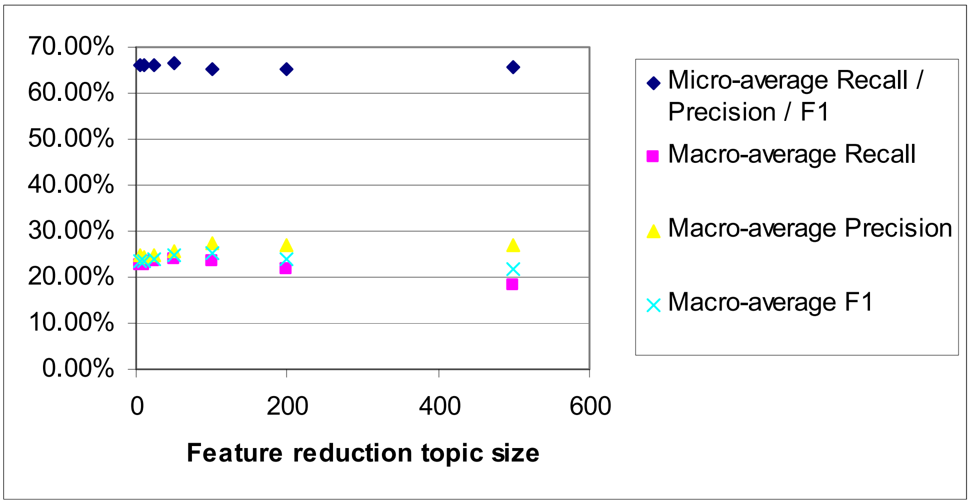


Figure 11: The scatter plot of the macro-averaging and micro-averaging scores for the 7 feature reduction classifiers.

Using the topic size greater than 500 for feature reduction as reference, the result of the topic size greater than 100 for feature reduction has 16.4% ($\frac{25.13-21.59}{21.59}$ %) improvement in macro-average performance (by F1 measurement). For difference numbers of topic used for feature reduction, the results also have improvement in macro-average F1 performance when comparing to the result of the topic size greater than 500 for feature reduction. In the circumstances, feature reduction to help to improve the classification results has the significant meaning.

3.4 Conclusions

We have shown that subtopic clustering of large topic classes can improve the macro-average performance consistently across different skewness of the test data set distribution. The optimal result shows that there is 16.34% ($\frac{47.57-40.89}{40.89}$ %) improvement in macro-average performance (by F1 measurement) when the maximum subtopic size equals to 100 by using complete-linkage clustering (hierarchical clustering). The macro-average F1 is 47.57% under the maximum subtopic size equals to 100 by using complete-linkage clustering as shown in Table 5, Table 7 and Table 8. The macro-average F1 is 40.89% without clustering as shown in Table 4, Table 7 and Table 8.)

This experiment shows that 100 is a useful threshold value that indicates the need to divide large topic classes into subtopic classes (i.e. subtopic clustering) in order to increase macro-average performance. However, there is a slight decrease in the micro-average performance and more research is needed to enhance the use of subtopic clustering for text categorization. We will further explore how the optimal size of the subtopic clusters can be determined analytically or automatically.

The comparison of hierarchical and non-hierarchical clustering shows that hierarchical clustering performs better for recall, precision and F1 performances when the maximum subtopic size is at 100. The optimal results of k-means clustering (non-hierarchical clustering) show that there is 11.42% ($\frac{45.56-40.89}{40.89}$%) improvement in macro-average F1. The macro-average F1 is 45.56% when the maximum subtopic size equals to 100 as given in Table 6 and Table 8. The macro-average F1 without clustering is 40.98% as shown in Table 4, Table 7 and Table 8. (The summarized results are shown in Table 12)

Table 12: Macro-average improvement of the results from the 925 test data sets (at skew distance = 0) using complete-linkage clustering and k-means clustering with topic/subtopic size limited to 100 are summarized.

Clustering method	Macro-average			Micro-average
	Recall	Precision	F1 (**Improvement**)	Recall/Precision/F1
No clustering	48.17%	33.49%	40.89% (-)	67.01%
Complete-linkage	57.16%	39.46%	47.57% (**16.34%**)	61.13%
K-means	55.26%	37.05%	45.56% (**11.42%**)	60.89%

For the experiment of percentage of topics with zero recall and precision (Section 3.3.3), there is 18.03% ($\frac{67.78-55.56}{67.78}$%) improvement by hierarchical clustering. It can further demonstrate the benefit of using subtopic clustering. For non-hierarchical clustering, there is also 16.39% ($\frac{67.78-56.67}{67.78}$%) improvement.

For the experiments of different numbers of topic used for feature reduction, the results improve in macro-average F1 performance when comparing to the results of topic size greater than 500 for feature reduction. The result of the topic size greater than 100 for feature reduction has 16.4% ($\frac{25.13-21.59}{21.59}$ %) improvement in macro-average F1 performance. In these circumstances, the contribution of feature reduction to improving the classification results is significant and note-worthy.

The experiments show promising results with the subtopic clustering approaches. The formation of subtopic clusters is predefined (unsupervised learning) and measured by similarity scores. In the next chapter, our proposed iterative subspace approach with Support Vector Machines is introduced for further investigation.

4 Boosting Method

4.1 Introduction

Support Vector Machines (SVMs) and boosting are two techniques for learning both having received a considerable attention in the recent years and many successful applications have been described in the literature [18, 26, 64, 65, 70]. SVMs and boosting have something in common to justify their success, namely the margin. The objective of SVMs is to maximize the separation between the classes. By using a kernel trick to map the training samples from an input space to a high dimensional feature space, SVM finds an optimal separating hyperplane in the feature space and uses a regularization parameter to balance its model complexity and training error. While SVMs explicitly maximizes the minimum margin, boosting tends to do the same thing indirectly through minimizing a cost function related to margin. Boosting is a general technique for improving performance of any given classifier [69]. It can effectively combine a number of weak classifiers into a strong classifier which can achieve an arbitrarily low error rate given sufficient training data, although each weak classifier might do a little better than random guessing.

The ensemble method, which finds a highly accurate classifier by combining many moderately accurate component classifiers, has recently been very successful in machine learning. One of the most commonly used techniques for constructing ensemble classifiers is adaptive boosting (AdaBoost). AdaBoost finds a combination of a number of weak classifiers in a stepwise additive manner. The weak classifier in each iteration step is trained on the resampled data according to the distribution based on a series of weights obtained from the training error by the learner computed up-to-date. The success of AdaBoost can be explained as enlarging the margin [70], which could enhance AdaBoost's generalization capability.

4.1.1 AdaBoost

AdaBoost is a machine learning algorithm, formulated by Yoav Freund and Robert Schapire [18]. It is a meta-algorithm, and can be used in conjunction with many other learning algorithms to improve their performance. AdaBoost is adaptive in the sense that subsequent classifiers built are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems, however, it can be less susceptible to the overfitting problem than most learning algorithms

4.1.2 LogitBoost (LogLossBoost)

LogitBoost is a boosting algorithm formulated by Jerome Friedman, Trevor Hastie, and Robert Tibshirani. The original paper [19] casts the AdaBoost algorithm into a statistical framework. Specifically, if one considers AdaBoost as a generalized additive model and then applies the cost functional of logistic regression, one can derive the LogitBoost algorithm. LogitBoost minimizes the logistic loss. LogitBoost places less emphasis on examples that are very badly classified.

4.1.3 RobustBoost (BrownBoost)

RobustBoost is a boosting algorithm that may be robust to noisy datasets. RobustBoost is an adaptive version of the boost by majority algorithm. As is true for all boosting algorithms, RobustBoost is used in conjunction with other machine learning methods. BobustBoost was introduced by Yoav Freund [15, 16].

4.1.4 Alternating Decision Tree

An alternating decision tree (ADTree) [17] is a machine learning method for classification. It generalizes decision trees and has connections to boosting.

Original boosting algorithms typically used either decision stumps or decision trees as weak hypotheses. As an example, boosting decision stumps creates a set of T weighted decision stumps (where T is the number of boosting iterations), which then vote on the final classification according to their weights. Individual decision stumps are weighted according to their ability to classify the data.

Boosting a simple learner results in an unstructured set of T hypotheses, making it difficult to infer correlations between attributes. ADTrees introduce structure to the set of hypotheses by requiring that they build off a hypothesis that was produced in an earlier iteration. The resulting set of hypotheses can be visualized in a tree based on the relationship between a hypothesis and its "parent".

Another important feature of boosted algorithms is that the data is given a different distribution at each iteration. Instances that are misclassified are given a larger weight while accurately classified instances are given reduced weight.

An ADTree consists of decision nodes and prediction nodes. Decision nodes specify a predicate condition. Prediction nodes contain a single number. ADTrees always have prediction nodes as both root and leaves. An instance is classified by an ADTree by following all paths for which all decision nodes are true and summing any prediction nodes that are traversed.

Primarily, the weak classifiers are put into a hierarchical order - the ADTree. The tree consists of two different kinds of node which alternately change on a path through the tree. Secondly, each decision node contains a weak classifier and has two prediction nodes containing the predictive values as its children. The weak classifiers in upper levels of the tree work as preconditions on those classifiers below them. And third, the root node contains the predictive value of the true-classifier. Thus, the predictive value is derived from the ratio of the number of samples between both classes and, therefore, it can be interpreted as a prior classifier. In each iteration step, the best classifier candidate is determined in conjunction with a precondition.

4.2 Methodology

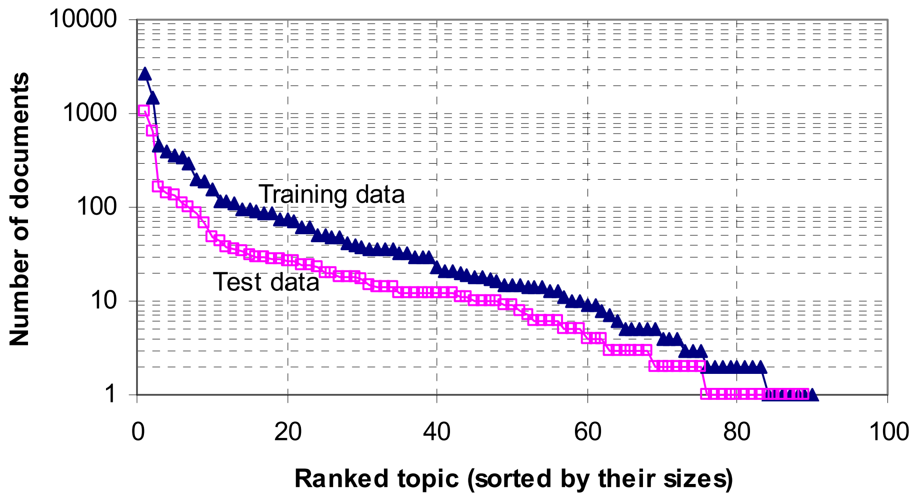
4.2.1 Experimental Setup

4.2.1.1 Data Set

The Reuters-21578 document set has previously been regarded as a standard real-world benchmarking corpus for the Information Retrieval (IR) community. The ModApte split (training data set: 9,603 documents, test data set: 3,299 documents, unused: 8,676 documents) of Reuters-21578 document set is used for our experiments.

Except two large topics, including “acq” (1,488 training documents) and “earn” (2,709 training documents), the rest of the training topics have the number of documents below 500 (ranging from 1 to 460). Test documents can be assigned to more than one topic; therefore, 3,299 single-label test documents are expanded to 3,409 test documents which are used for the evaluation exercise.

The distribution of the number of training documents in a topic class is typically highly skewed. The number of terms in a topic increases logarithmically with an increase in the number of training documents. They are shown in Figure 12.



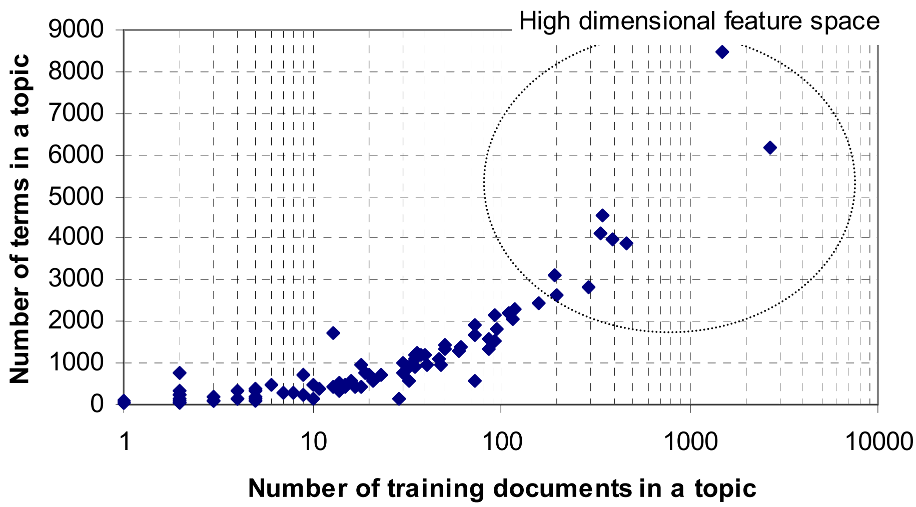


Figure 12: The number of training/test documents plotted against ranked topic sorted by their sizes (top). The number of terms in a topic plotted against the number of training documents in its topic (bottom).

4.2.1.2 Preprocessing

Preprocessing involves removing SGML tags, punctuation marks, stop words and performing word stemming to reduce the feature vector size. Bag-of-words [57] document representation (vector space model) scheme is used for feature representation. Term importance is assumed to be inversely proportional to the number of documents a particular term appears in. The term frequency (*tf*) and inverse document frequency (*idf*) are used to assign

weights to terms. The inverse document frequency for term t is defined as [67]:

$$idf(t) = \log(N / n(t)) \quad . \tag{15}$$

The common non-content words are removed to reduce possible interference in classification results. It is assumed that the importance of a term increases with its use-frequency. Combining these two assumptions lead to *tfidf*:

$$tfidf(t) = tf(t) \times idf(t) \quad . \tag{16}$$

Cosine normalization is used. Every document vector is divided by its Euclidean length, $((w_1)^2 + (w_2)^2 + \ldots + (w_n)^2)^{1/2}$, where w_i is the *tfidf* weight of the *i*-th term in the document. The final weight for a term hence becomes:

$$\frac{tfidf \text{ weight}}{\text{Euclidean length of the document vector}} \quad . \tag{17}$$

4.2.1.3 Classifier

Instead of implementing a classifier, we use JBoost [31, 93] to perform text classification. JBoost is an implementation of boosting in java. The package includes the source, the executable java, visualization scipts (mostly written in python) and a collection of examples that demonstrate the capabilities of Jboost Some of the algorithms currently implemented include AdaBoost, LogitBoost, RobustBoost and alternating decision trees.

4.2.2 Performance Measurements

Referring to Section 3.2.2.1, classification performance is measured by both recall and precision. For evaluating the performance, three quantities are of interest for each topic.

They are: a = the number of documents correctly assigned to this topic.

b = the number of documents incorrectly assigned to this topic.

c = the number of documents incorrectly rejected from this topic.

From these quantities, the performance measures (Equation 9, Equation 10 and Equation 11) are defined in Section 3.2.2.1. They are recall, precision and F1 measures:

$$\text{recall} = a/(a+c) \quad .$$

$$\text{precision} = a/(a+b) \quad .$$

$$\text{F1} = (2 \times \text{recall} \times \text{precision})/(\text{recall} + \text{precision}) \quad .$$

In this experiment, we use the subset of Reuters-21578 collection. For providing enough training data learnt by boosting method, only those topics (categories) with training document sizes which are equal to or greater than 50 are used. 25 topics can meet this requirement and 300 topic pairs for JBoost which is an implementation of boosting in java (AdaBoost, LogitBoost, RobustBoost and alternating ADTree) are generated for the experiment.

The experiment is done under 8-fold, 10-fold, and 12-fold cross validations; the training documents are sampled by systematic sampling (selected sequentially by system file ordering). The number of training documents and the number of test documents for each sample test under 8-fold, 10-fold and 12-fold cross validations are summarized in Table 13. In fact, 300 topic pairs (25 topics) are generated for performance evaluation. Therefore 24 times more of training and test documents are redundantly generated for performance evaluation. Table 14 shows the actual numbers of training and test documents are used.

Table 13: The number of training documents and the number of test documents of each sample test (25 topics) are summarized.

Sample test	8-fold cross validation		10-fold cross validation		12-fold cross validation	
	Number of training documents (25 topics)	Number of test documents (25 topics)	Number of training documents (25 topics)	Number of test documents (25 topics)	Number of training documents (25 topics)	Number of test documents (25 topics)
1	6,849	965	7,044	770	7,171	643
2	6,846	968	7,040	774	7,170	644
3	6,840	974	7,037	777	7,167	647
4	6,837	977	7,034	780	7,165	649
5	6,834	980	7,032	782	7,164	650
6	6,834	980	7,032	782	7,164	650
7	6,831	983	7,030	784	7,163	651
8	6,827	987	7,028	786	7,163	651
9	-	-	7,025	789	7,162	652
10	-	-	7,024	790	7,160	654
11	-	-	-	-	7,155	659
12	-	-	-	-	7,150	664
Total	54,698	7,814	70,326	7,814	85,954	7,814

Table 14: The number of training documents and the number of test documents of each sample test (300 topic pairs) for JBoost are summarized.

Sample test	8-fold cross validation		10-fold cross validation		12-fold cross validation	
	Number of training documents used in 300 topic pairs (25 topics)	Number of test documents used in 300 topic pairs (25 topics)	Number of training documents used in 300 topic pairs (25 topics)	Number of test documents used in 300 topic pairs (25 topics)	Number of training documents used in 300 topic pairs (25 topics)	Number of test documents used in 300 topic pairs (25 topics)
1	164,376	23,160	169,056	18,480	172,104	15,432
2	164,304	23,232	168,960	18,576	172,080	15,456
3	164,160	23,376	168,888	18,648	172,008	15,528
4	164,088	23,448	168,816	18,720	171,960	15,576
5	164,016	23,520	168,768	18,768	171,936	15,600
6	164,016	23,520	168,768	18,768	171,936	15,600
7	163,944	23,592	168,720	18,816	171,912	15,624
8	163,848	23,688	168,672	18,864	171,912	15,624
9	-	-	168,600	18,936	171,888	15,648
10	-	-	168,576	18,960	171,840	15,696
11	-	-	-	-	171,720	15,816
12	-	-	-	-	171,600	15,936
Total	1,312,752	187,536	1,687,824	187,536	2,062,896	187,536

4.3 Experimental Results and Discussion

The results from the final classifier (ADTree) and the number of rounds of boosting (AdaBoost) set to 100 are summarized in Table 15. The plot is shown in Figure 13. There is no significant difference (less than 1%) among different cross validations. Therefore further experiments will be done under 8-fold cross validation.

Table 15: The macro-average and micro-average performance of AdaBoost method evaluated under 8-fold, 10-fold and 12-fold cross validations are summarized.

	Macro-average			Micro-average
Cross validation	Recall	Precision	F1	Recall/Precision/F1
8-fold	98.30%	98.49%	98.39%	99.35%
10-fold	98.36%	98.56%	98.46%	99.38%
12-fold	98.35%	98.58%	98.47%	99.37%

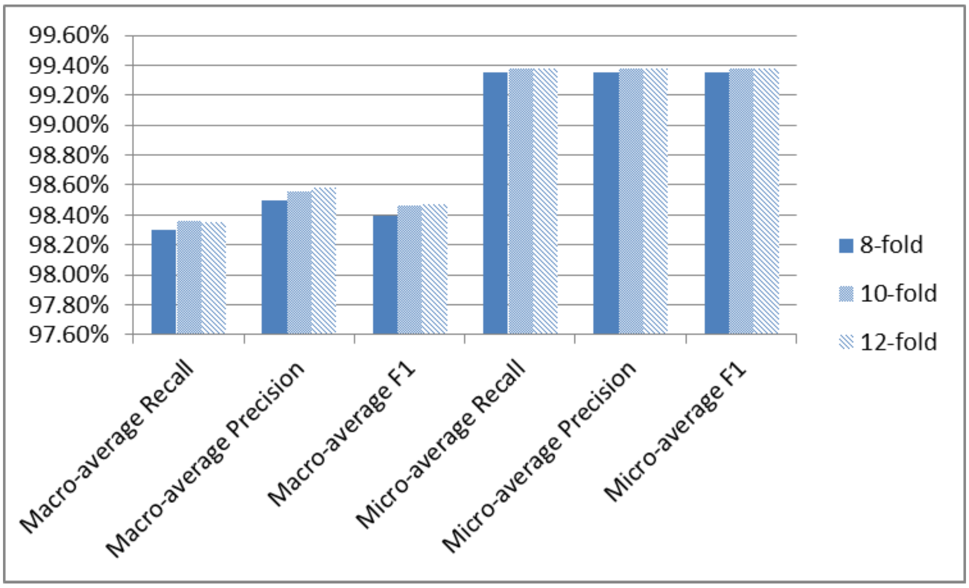


Figure 13: The macro-average and micro-average performance of AdaBoost method evaluated under 8-fold, 10-fold and 12-fold cross validations are plotted.

The results under 8-fold cross validation from the final classifier (ADTree) and the number of rounds of boosting (AdaBoost/LogitBoost/RobustBoot) set to 100 are summarized in Table 16. The plot is shown in Figure 14. The performance scores (less than 1%) between AdaBoost and LogitBoost are similar. Therefore further AdaBoost experiments will be done under 8-fold cross validation.

Table 16: The macro-average and micro-average performance of different methods (AdaBoost/LogitBoost/RobustBoost) evaluated under 8-fold fold cross validation are summarized.

	Macro-average			Micro-average
Boosting methodr	Recall	Precision	F1	Recall/Precision/F1
AdaBoost	98.30%	98.49%	98.39%	99.35%
LogitBoost	98.22%	98.45%	98.33%	99.32%
RobustBoost	90.47%	95.62%	92.97%	96.38%

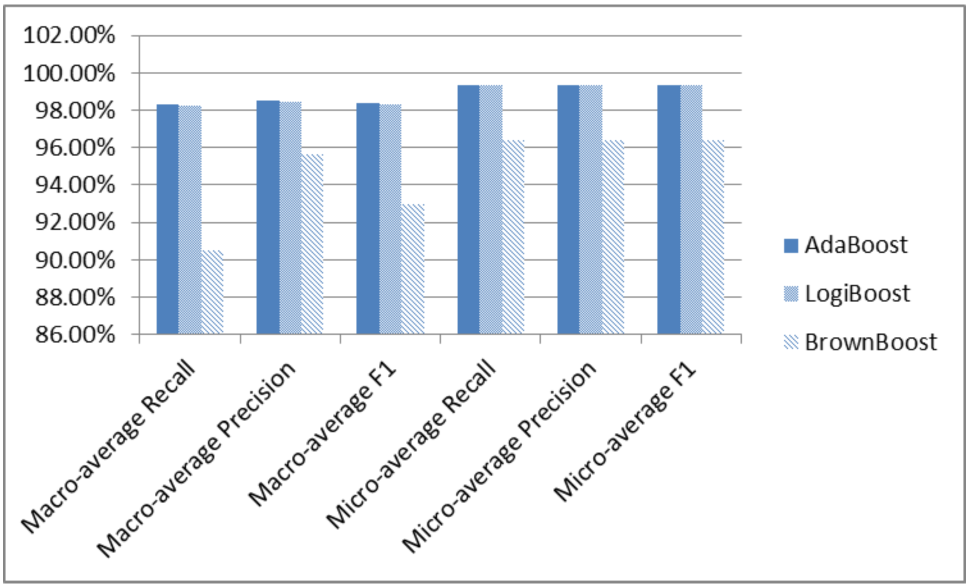
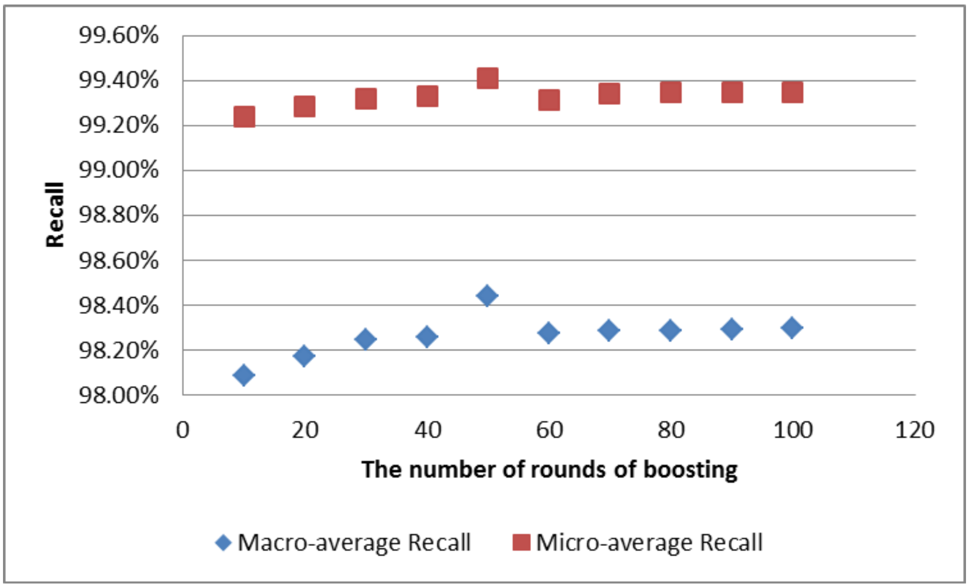


Figure 14: The macro-average and micro-average performance of different methods (AdaBoost/LogitBoost/RobustBoost) evaluated under 8-fold fold cross validation are plotted.

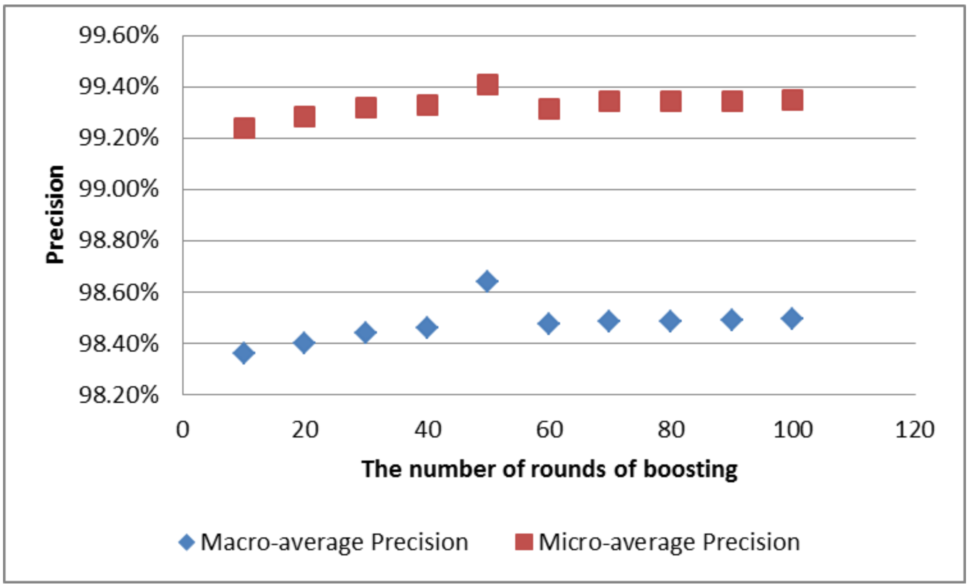
The results under 8-fold cross validation from the final classifier (ADTree) and different numbers of rounds of boosting (AdaBoost) from 10 to 100 are summarized in Table 17. The plot is shown in Table 15. The best performance scores are achieved when the number of rounds of boosting is set to 50 where the macro-average recall is 98.44%, macro-average precision is 98.64%, macro-average F1 is 98.54% and micro-average recall/precision/F1 is 99.41%.

Table 17: The macro-average and micro-average performance of AdaBoost method evaluated under different numbers of rounds of boosting (8-fold fold cross validation) are summarized.

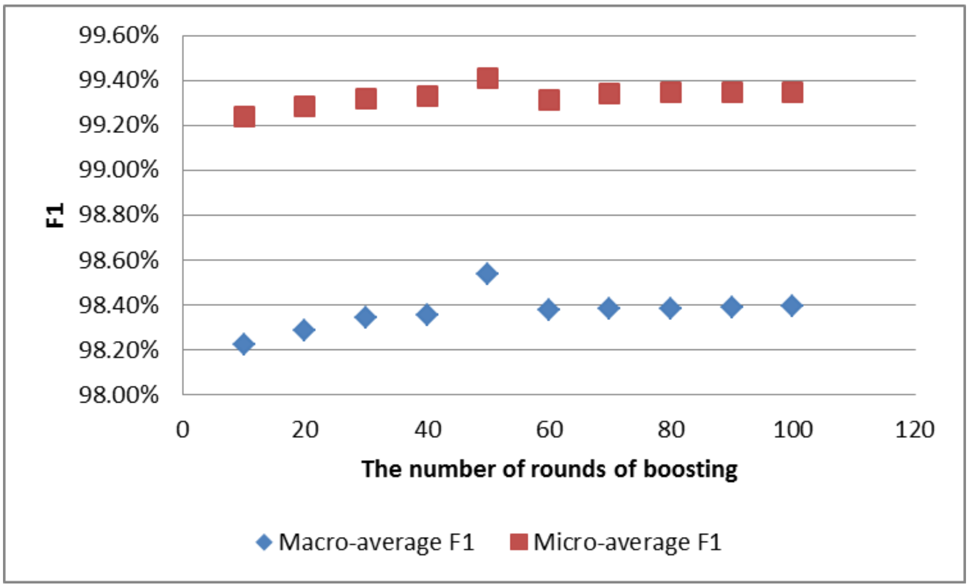
The number of rounds of boosting	Macro-average Recall	Macro-average Precision	Macro-average F1	Micro-average Recall/Precision/F1
10	98.09%	98.36%	98.22%	99.24%
20	98.17%	98.40%	98.28%	99.29%
30	98.24%	98.44%	98.34%	99.32%
40	98.26%	98.46%	98.36%	99.33%
50	98.44%	98.64%	98.54%	99.41%
60	98.28%	98.48%	98.38%	99.32%
70	98.28%	98.48%	98.38%	99.34%
80	98.29%	98.48%	98.38%	99.34%
90	98.29%	98.49%	98.39%	99.35%
100	98.30%	98.49%	98.39%	99.35%



(a)



(b)



(c)

Figure 15: The macro-average and micro-average performance of AdaBoost method evaluated under different numbers of rounds of boosting (8-fold fold cross validation) are plotted. (a) Recall (b) Precision (c) F1

5 Iterative Subspace Method

5.1 Introduction

We propose a new approach to improve the accuracy of text categorization using iterative subspace method. In a number of probabilistic approaches, texts in the same category are implicitly assumed to be generated from an identical distribution over words. However this assumption is not accurate, in the previous chapter, training texts are clustered so that the assumption is more likely to be realistic and the result shows that subtopic clustering can alleviate this problem and text categorization can be improved. In fact there is a limitation in the subtopic clustering approach. The formation of subtopic clusters are predefined (unsupervised learning) and measured by similarity scores. The idea of iterative subspace approach is that subspace generation is generated by classification performance (supervised learning). The classification task can be done by any classifier such as Naive Bayes classifier, Support Vector Machines and Artificial Neural Network.

In the case of backpropagation based artificial neural networks or perceptrons, the type of decision boundary that the network can learn is determined by the number of hidden layers the network has. If it has no hidden layers, then it can only learn linear problems. If it has one hidden layer, then it can learn problems with convex decision boundaries (and some concave decision boundaries). The network can learn more complex problems if it has two or more hidden layers.

In particular, support vector machines find a hyperplane that separates the feature space into two classes with the maximum margin. If the problem is not originally linearly separable, the kernel trick is used to turn it into a linearly separable one, by increasing the number of dimensions. Thus a general hypersurface in a small dimension space is turned into a hyperplane in a space with much larger dimensions.

Neural networks try to learn the decision boundary which minimizes the empirical error, while support vector machines try to learn the decision boundary which gives the best generalization. We conduct the experiment with Support Vector Machines for the classification tasks to validate this iterative subspace method. Support Vector Machines are used because they are effective (text) classifiers, have flexible decision boundaries by using different kernels, have geometrical properties that are relevant to our approach, and readily available for independent verification.

5.1.1 Support Vector Machines

Support Vector Machines (SVMs) [2] are binary classifiers which were originally proposed by Vapnik [81] and have achieved high accuracy in various tasks, such as object recognition [63] and digit recognition [80]. SVMs are a set of related supervised learning methods used for classification and regression. In simple words, given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. Intuitively, an SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

Suppose some given data points each belong to one of two classes, and the goal is to decide which class a new data point will be in. In the case of Support Vector Machines, a data point is viewed as a n-dimensional vector, and we want to know whether we can separate such points with a (n-1)-dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. However, we are additionally interested in finding out if we can achieve maximum separation (margin) between the two classes. By this we mean that we pick the hyperplane so that the distance from the hyperplane to the nearest data point is maximized. That is to say that the nearest distance between a point in one separated hyperplane and a point in the other separated hyperplane is maximized. Now, if such a hyperplane exists, it is clearly of interest and is known as the maximum-margin hyperplane as in general the larger the margin the lower the generalization error of the classifier and such a linear classifier is known as a maximum margin classifier. Since Support Vector Machines are linear classifiers, their separating ability is limited. To compensate for this limitation, the kernel method is usually combined with Support Vector Machines.

5.1.1.1 Separable Classes

For the case of two-class linearly separable as shown in Figure 16 which illustrates the classification task with two possible hyperplane solutions (solid-line and dotted-line). Let $x_i, i = 1,2,...,N$ be the feature vectors of the training set, X. These belong to either of two classes, ω_1, ω_2, which are assumed to be linearly separable. A hyperplane is defined as

$$g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 = 0 \quad \textbf{(18)}$$

that classifies correctly all the training vectors.

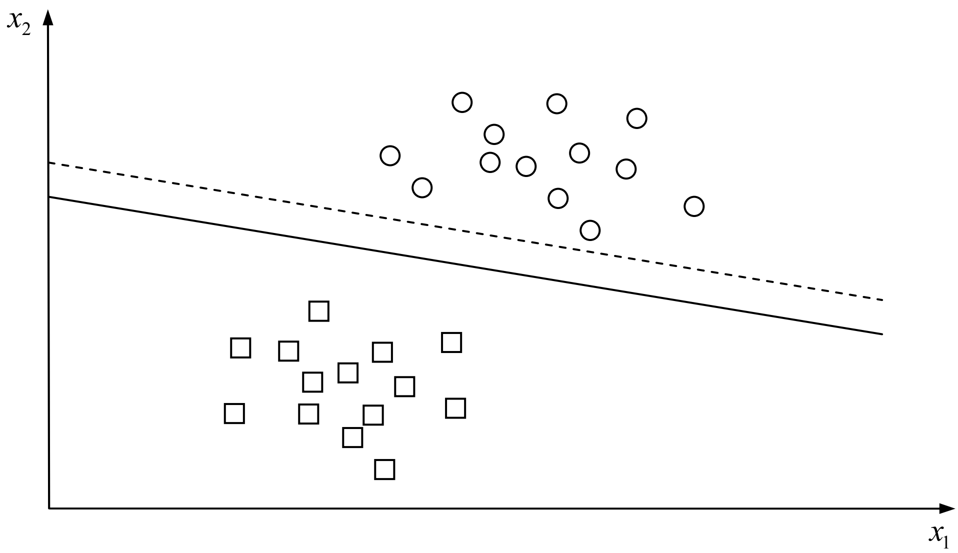


Figure 16: An example of a linearly separable two-class problem with two possible linear classifiers.

For the generalization performance of the classifier, the term *margin* that a hyperplane leaves from both classes is quantified. Every hyperplane is characterized by its direction (determined by $\boldsymbol{w}$) and its exact position in space (determined by w_0). Since we want to give no preference to either of the classes, then it is reasonable for each direction to select that hyperplane which has the same distance from the respective nearest points in ω_1 and

ω_2. The hyperplanes shown in Figure 17 with solid lines are the selected ones from the infinite set in the respective direction. The margin for "direction 1" is $2z_1$, and the margin for "direction 2" is $2z_2$.

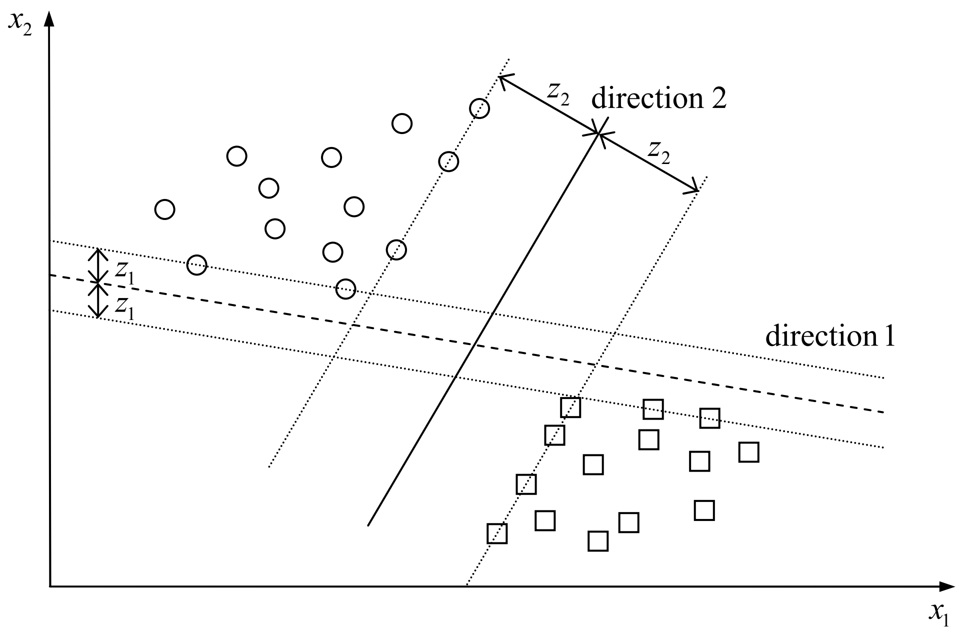


Figure 17: An example of linearly separable two-class problem with two possible linear classifiers and their corresponding support vectors.

Further Condiersing the decision hypersurface in the *l*-dimensional feature space is a hyperplane as was shown in Equation (**18**) that is

$$g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 = 0$$

where $\boldsymbol{w} = [w_1, w_2, ..., w_l]^T$ is known as the *weight vector* and w_0 as the *threshold*. If $\boldsymbol{x}_1, \boldsymbol{x}_2$ are two points on the decision hyperplane, then the following is valid

$$0 = \boldsymbol{w}^T \boldsymbol{x}_1 + w_0 = \boldsymbol{w}^T \boldsymbol{x}_2 + w_0 \Rightarrow \boldsymbol{w}^T (\boldsymbol{x}_1 - \boldsymbol{x}_2) = 0 \quad \textbf{(19)}$$

Since the difference vector $\boldsymbol{x}_1 - \boldsymbol{x}_2$ obviously lies on the decision hyperplane (for any $\boldsymbol{x}_1, \boldsymbol{x}_2$), it is apparent from Equation (**19**) that the vector $\boldsymbol{w}$ is orthogonal to the decision hyperplane.

Figure 18 shows the corresponding geometry (for $w_1 > 0, w_2 > 0, w_0 < 0$). On one side of the line it is $g(\boldsymbol{x}) > 0(+)$ and on the other side it is $g(\boldsymbol{x}) < 0(-)$.

$$d = \frac{|w_0|}{\sqrt{w_1^2 + w_2^2}} \tag{20}$$

and

$$z = \frac{|g(\boldsymbol{x})|}{\sqrt{w_1^2 + w_2^2}} \tag{21}$$

In other works, $|g(\boldsymbol{x})|$ is a measure of the Euclidean distance of the point $\boldsymbol{x}$ from the decision hyperplane. On one side of the plane $g(\boldsymbol{x})$ takes positive values and on the other negative. In the special case that $w_0 = 0$, the hyperplane passes through the origin.

Similarly, the distance of a point from a hyperlane in Figure 18 is given by

$$z = \frac{|g(\boldsymbol{x})|}{\|\boldsymbol{w}\|}$$

We can now scale $\boldsymbol{w}, w_0$ so that the value of $g(\boldsymbol{x})$, at the nearest points in w_1, w_2, is equal to 1 for w_1 and equal to -1 for w_2. This is equivalent with

1. Having a margin of $\frac{1}{\|\boldsymbol{w}\|} + \frac{1}{\|\boldsymbol{w}\|} = \frac{2}{\|\boldsymbol{w}\|}$

2. Requiring that

$$\begin{aligned} \boldsymbol{w}^T \boldsymbol{x} + w_0 &\geq 1, && \forall \boldsymbol{x} \in w_1 \\ \boldsymbol{w}^T \boldsymbol{x} + w_0 &\leq -1, && \forall \boldsymbol{x} \in w_2 \end{aligned}$$

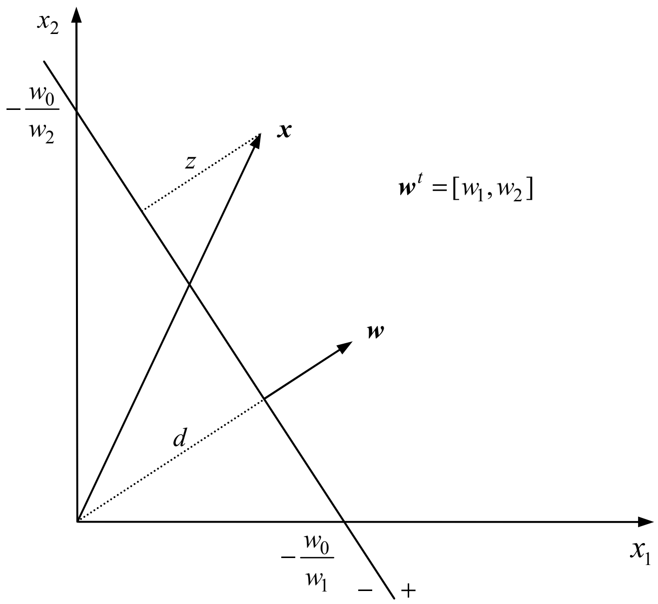


Figure 18: Geometry for the decision line.

5.1.1.2 Non-separable Classes

When the classes are not separable, the above setup is no longer valid. Figure 19 illustrates the case in which the two classes are not separable. Any attempt to draw a hyperplane will never end up with a class separation region with no data points inside it, as was the case in the linearly separable task.

Applying the kernel trick is a way to create non-linear classifiers to maximum-margin hyperplanes [6]. The resulting algorithm is formally similar, except that every dot product is replaced by a non-linear kernel function. This allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space. The transformation may be non-linear and the transformed space high dimensional; thus though the classifier is a hyperplane in the high-dimensional feature space, it may be non-linear in the original input space.

If the kernel used is a Gaussian radial basis function, the corresponding feature space is a Hilbert space of infinite dimension. Maximum margin classifiers are well regularized, so the infinite dimension does not spoil the results.

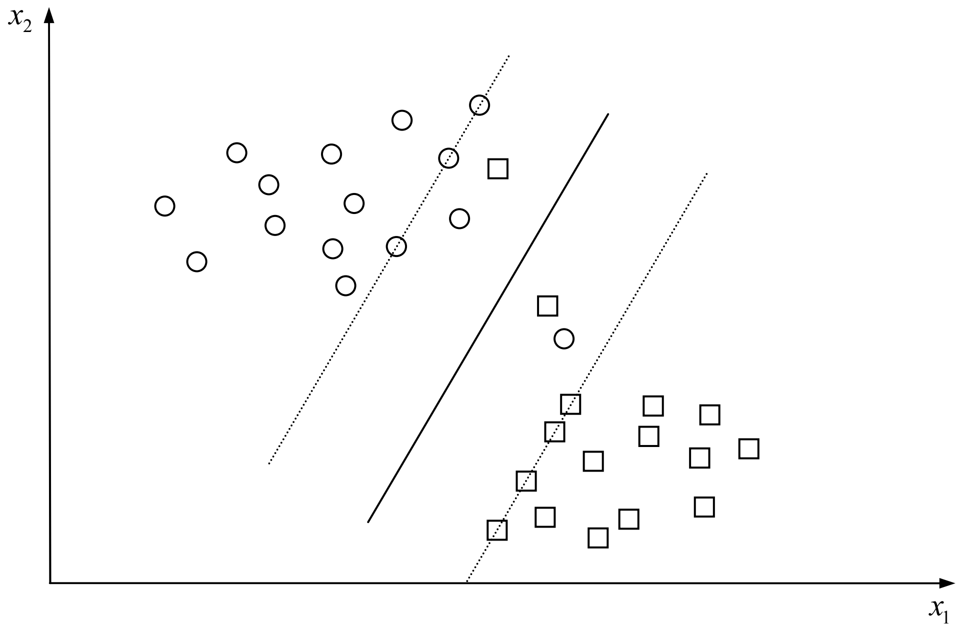


Figure 19: An example of nonseparable two-class case, points fall inside the class separation region.

5.1.2 Basic Scheme

The idea of this model is to generate subspaces from different training data set through error-driven learning. Feature selection is done on the training data set and done recursively to build classifiers. Through the iteration, suitable features can be selected from different subspaces. The process will stop when all topics are learned to build classifiers. Sub-classifiers will be generated for assisting in document classification. Better category boundary is expected to be obtained through the learning of these cascade classifiers. The proposed method of the iterative subspace generation to text categorization for document training is shown in Figure 20 and for document test is shown in Figure 21.

It is new that the proposed iterative subspace model allows suitable features to be selected from different subspaces through the iterative process to obtain the better category boundary. The main difference of our proposed iterative subspace classifier from others is trying to find a set of suitable features (subspaces) for each category through the multi-level classification (classifier). In Figure 20 and Figure 21, the classifier can be any classifier in general. In our case, Support Vector Machines are used as classifiers in the experiments. Instead of implementing a classifier, we use SVM-Light [38] to perform text classification.

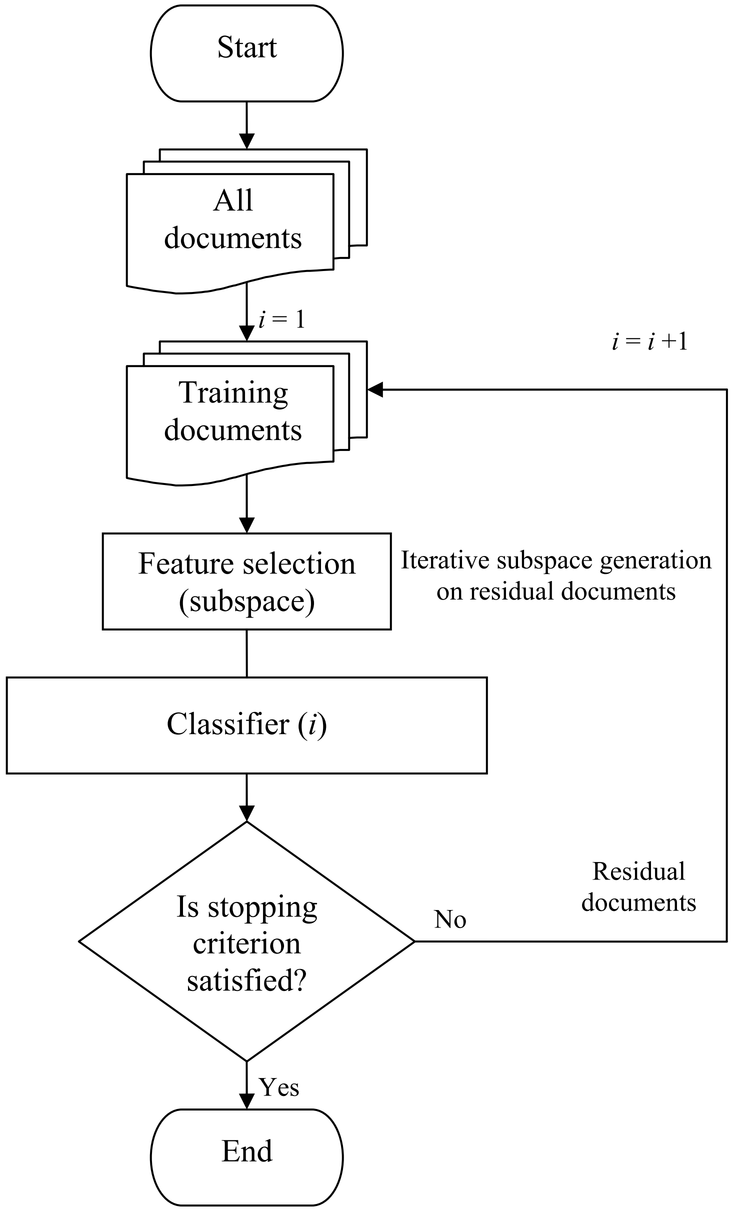


Figure 20: Flowchart of the iterative subspace generation for text categorization (document training).

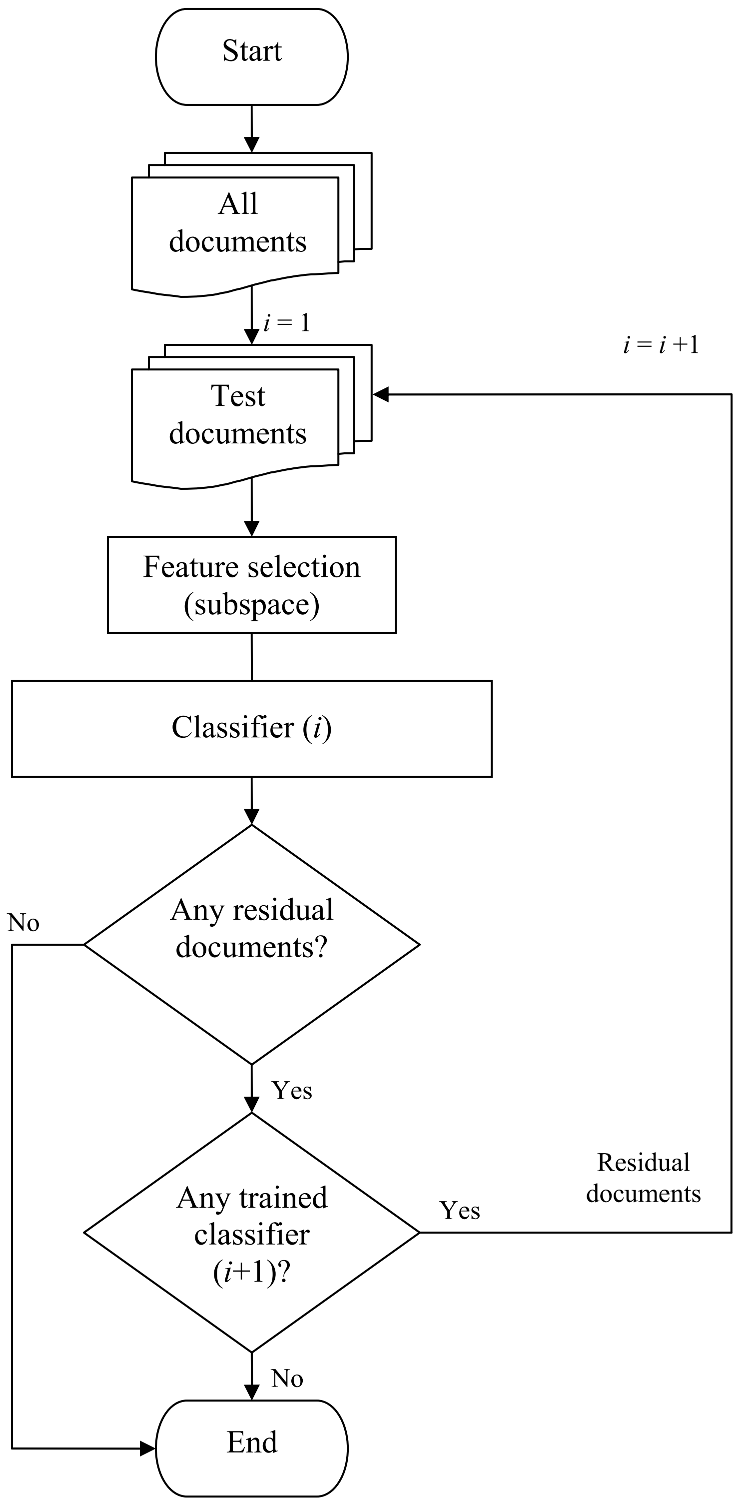


Figure 21: Flowchart of the iterative subspace generation for text categorization (document test).

5.2 Methodology

5.2.1 Experimental Setup

5.2.1.1 Data Set

The Reuters-21578 document set has previously been regarded as a standard real-world benchmarking corpus for the Information Retrieval (IR) community. The ModApte split (training data set: 9,603 documents, test data set: 3,299 documents, unused: 8,676 documents) of Reuters-21578 document set is used for our experiments.

Except two large topics, including “acq” (1,488 training documents) and “earn” (2,709 training documents), the rest of the training topics have the number of documents below 500 (ranging from 1 to 460). Test documents can be assigned to more than one topic; therefore, 3,299 single-label test documents are expanded to 3,409 test documents which are used for the evaluation exercise.

The distribution of the number of training documents in a topic class is typically highly skewed. The number of terms in a topic increases logarithmically with an increase in the number of training documents. They are shown in Figure 22.

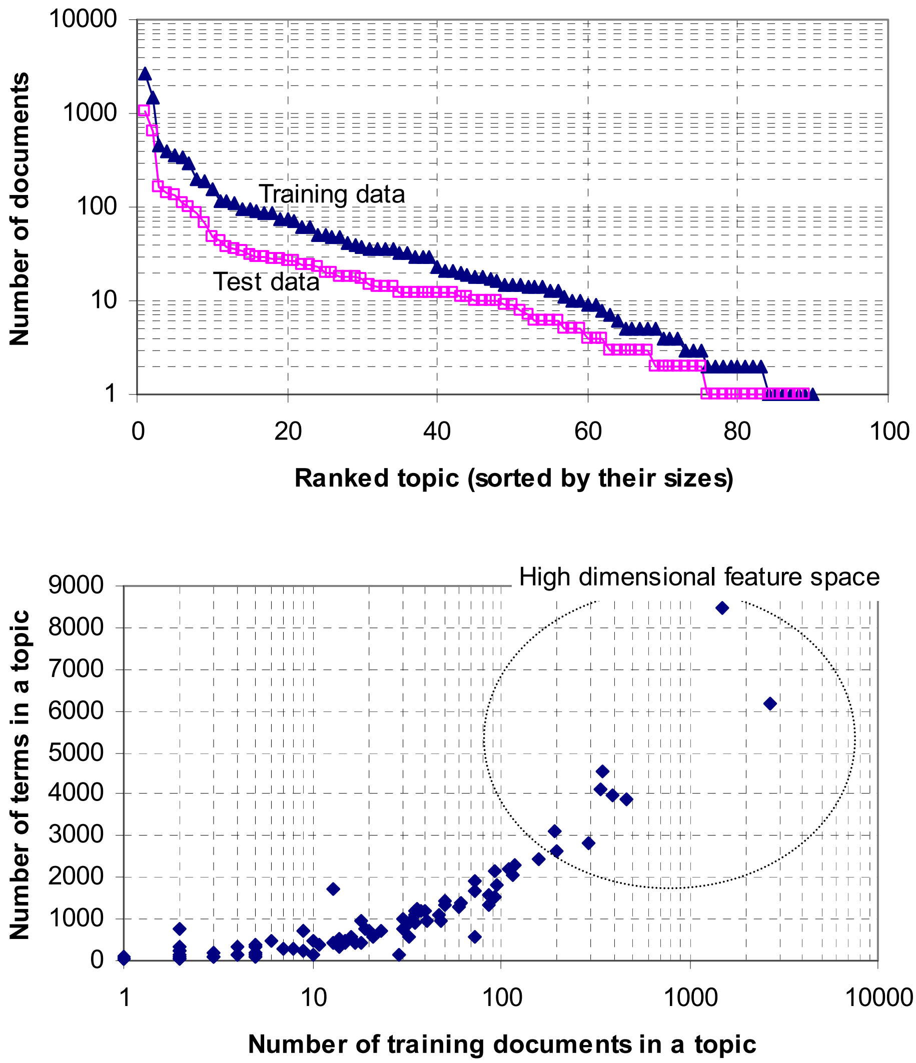


Figure 22: The number of training/test documents plotted against ranked topic sorted by their sizes (top). The number of terms in a topic plotted against the number of training documents in its topic (bottom).

5.2.1.2 Preprocessing

Preprocessing involves removing SGML tags, punctuation marks, stop words and performing word stemming to reduce the feature vector size. Bag-of-words [57] document representation (vector space model) scheme is used for feature representation. Term importance is assumed to be inversely proportional to the number of documents a particular term appears in. The term frequency (*tf*) and inverse document frequency (*idf*) are used to assign

weights to terms. The inverse document frequency for term t is defined as [67]:

$$idf(t) = \log(N / n(t)) \ . \qquad (22)$$

The common non-content words are removed to reduce possible interference in classification results. It is assumed that the importance of a term increases with its use-frequency. Combining these two assumptions lead to *tfidf*:

$$tfidf(t) = tf(t) \times idf(t) \ . \qquad (23)$$

Cosine normalization is used. Every document vector is divided by its Euclidean length, $((w_1)^2 + (w_2)^2 + \ldots + (w_n)^2)^{1/2}$, where w_i is the *tfidf* weight of the *i*-th term in the document. The final weight for a term hence becomes:

$$\frac{tfidf \text{ weight}}{\text{Euclidean length of the document vector}} \ . \qquad (24)$$

5.2.1.3 Classifier

Instead of implementing a classifier, we use SVM-Light [38] to perform text classification. SVM-Light is an implementation of Vapnik's Support Vector Machine [81] for the problem of pattern recognition, for the problem of regression, and for the problem of learning a ranking function. The optimization algorithms used in SVM-Light are described in [33, 36]. The algorithm has scalable memory requirements and can handle problems with many thousands of support vectors efficiently.

The software also provides methods for assessing the generalization performance efficiently. It includes two efficient estimation methods for both error rate and precision/recall. XiAlpha-estimates [35, 36] can be computed at essentially no computational expense, but they are conservatively biased. Almost unbiased estimates provides leave-one-out testing. SVM-Light exploits that the results of most leave-one-outs (often more than 99%) are predetermined and need not be computed [36].

New in this version is an algorithm for learning ranking functions [37]. The goal is to learn a function from preference examples, so that it orders a new set of objects as accurately as possible. Such ranking problems naturally occur in applications like search engines and recommender systems.

Futhermore, this version includes an algorithm for training large-scale transductive SVMs. The algorithm proceeds by solving a sequence of optimization problems lower-bounding the solution using a form of local search. A detailed description of the algorithm can be found in [34]. A similar transductive learner, which can be thought of as a transductive version of k-Nearest Neighbor is the Spectral Graph Transducer.

SVM-Light can also train SVMs with cost models (see [58]). The code has been used on a large range of problems, including text classification [32, 34],. Many tasks have the property of sparse instance vectors. This implementation makes use of this property which leads to a very compact and efficient representation.

5.2.2 Performance Measurements

5.2.2.1 Recall, Precision and F1

Referring to Section 3.2.2.1, classification performance is measured by both recall and precision. For evaluating the performance, three quantities are of interest for each topic.

They are: a = the number of documents correctly assigned to this topic.

b = the number of documents incorrectly assigned to this topic.

c = the number of documents incorrectly rejected from this topic.

From these quantities, the performance measures (Equation 9, Equation 10 and Equation 11) are defined in Section 3.2.2.1. They are recall, precision and F1 measures:

$$\text{recall} = a/(a+c) \ .$$

$$\text{precision} = a/(a+b) \ .$$

$$\text{F1} = (2 \times \text{recall} \times \text{precision})/(\text{recall} + \text{precision}) \ .$$

In this experiment, we use the subset of Reuters-21578 collection. For providing enough training data learnt by the proposed Iterative Subspace Method, only those topics (categories) with training document sizes which are equal to or greater than 50 are used. 25 topics can meet this requirement and 300 topic pairs for SVM classifiers (binary classifiers) are generated for the experiment.

The experiment is done under 8-fold, 10-fold, and 12-fold cross validations; the training documents are sampled by systematic sampling (selected sequentially by system file ordering). The number of training documents and the number of test documents for each sample test under 8-fold, 10-fold and 12-fold cross validations are summarized in Table 18. In fact, 300 topic pairs (25 topics) are generated for performance evaluation. Therefore 24 times more of training and test documents are redundantly generated for performance evaluation. Table 19 shows the actual numbers of training and test documents are used.

Table 18: The number of training documents and the number of test documents of each sample test (25 topics) are summarized.

Sample test	8-fold cross validation		10-fold cross validation		12-fold cross validation	
	Number of training documents (25 topics)	Number of test documents (25 topics)	Number of training documents (25 topics)	Number of test documents (25 topics)	Number of training documents (25 topics)	Number of test documents (25 topics)
1	6,849	965	7,044	770	7,171	643
2	6,846	968	7,040	774	7,170	644
3	6,840	974	7,037	777	7,167	647
4	6,837	977	7,034	780	7,165	649
5	6,834	980	7,032	782	7,164	650
6	6,834	980	7,032	782	7,164	650
7	6,831	983	7,030	784	7,163	651
8	6,827	987	7,028	786	7,163	651
9	-	-	7,025	789	7,162	652
10	-	-	7,024	790	7,160	654
11	-	-	-	-	7,155	659
12	-	-	-	-	7,150	664
Total	54,698	7,814	70,326	7,814	85,954	7,814

Table 19: The number of training documents and the number of test documents of each sample test (300 topic pairs) for SVM classifiers are summarized.

Sample test	8-fold cross validation		10-fold cross validation		12-fold cross validation	
	Number of training documents used in 300 topic pairs (25 topics)	Number of test documents used in 300 topic pairs (25 topics)	Number of training documents used in 300 topic pairs (25 topics)	Number of test documents used in 300 topic pairs (25 topics)	Number of training documents used in 300 topic pairs (25 topics)	Number of test documents used in 300 topic pairs (25 topics)
1	164,376	23,160	169,056	18,480	172,104	15,432
2	164,304	23,232	168,960	18,576	172,080	15,456
3	164,160	23,376	168,888	18,648	172,008	15,528
4	164,088	23,448	168,816	18,720	171,960	15,576
5	164,016	23,520	168,768	18,768	171,936	15,600
6	164,016	23,520	168,768	18,768	171,936	15,600
7	163,944	23,592	168,720	18,816	171,912	15,624
8	163,848	23,688	168,672	18,864	171,912	15,624
9	-	-	168,600	18,936	171,888	15,648
10	-	-	168,576	18,960	171,840	15,696
11	-	-	-	-	171,720	15,816
12	-	-	-	-	171,600	15,936
Total	1,312,752	187,536	1,687,824	187,536	2,062,896	187,536

The experiment is done under 8-fold, 10-fold, and 12-fold cross validations; the training documents are sampled by systematic sampling (selected sequentially by system file ordering). The number of training documents and the number of test documents for each sample test under 8-fold, 10-fold and 12-fold cross validations are summarized in Table 18. If fact, 300 topic pairs (25 topics) are generated for performance evaluation. Therefore 24 times more of training and test documents are redundantly generated for performance evaluation. Table 19 shows the actual numbers of training and test documents are used.

Table 20, Table 21 and Table 22 show the number of training documents of each topic (25 topics) and their performance measures (such as *a*, *b*, *c* for calculating recall, precision and F1) evaluated by standard SVM method under 8-fold, 10-fold and 12-fold cross validations.

Table 20: The number of training documents of each topic (25 topics) and their performance measures under 8-fold cross validation are summarized.

	8-fold cross validation						
Topic	Number of training documents	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	249,984	35,670	42	1,078	97.07	99.88	98.45
bop	10,416	973	515	80	92.40	65.39	76.58
carcass	8,400	514	686	80	86.53	42.83	57.30
cocoa	8,400	687	513	23	96.76	57.25	71.94
coffee	18,480	2,220	420	271	89.12	84.09	86.53
corn	26,712	3,265	551	600	84.48	85.56	85.01
cpi	10,080	985	455	50	95.17	68.40	79.60
crude	58,632	8,154	222	822	90.84	97.35	93.98
dlr	16,128	1,909	395	147	92.85	82.86	87.57
earn	455,112	64,993	23	416	99.36	99.96	99.66
gnp	15,456	1,753	455	222	88.76	79.39	83.82
gold	15,792	1,893	363	107	94.65	83.91	88.96
grain	66,192	9,244	212	1,213	88.40	97.76	92.84
interest	48,552	6,590	346	546	92.35	95.01	93.66
livestock	12,264	1,105	647	130	89.47	63.07	73.99
money-fx	77,280	10,821	219	905	92.28	98.02	95.06
money-supply	14,616	1,800	288	54	97.09	86.21	91.32
nat-gas	12,096	1,208	520	111	91.58	69.91	79.29
oilseed	19,656	2,153	655	442	82.97	76.67	79.70
ship	32,088	4,208	376	558	88.29	91.80	90.01
soybean	12,264	1,092	660	225	82.92	62.33	71.16
sugar	19,824	2,360	472	380	86.13	83.33	84.71
trade	56,616	7,858	230	909	89.63	97.16	93.24
veg-oil	14,448	1,476	588	218	87.13	71.51	78.55
wheat	33,264	4,282	470	736	85.33	90.11	87.66

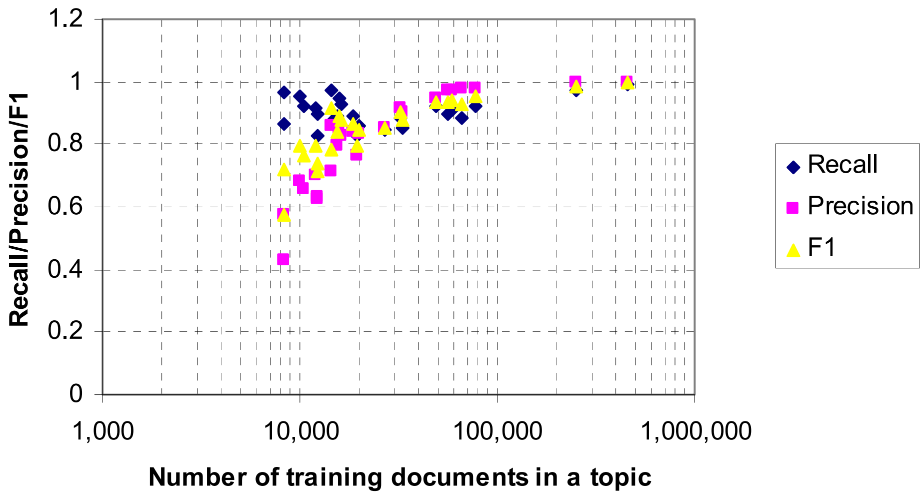
Table 21: The number of training documents of each topic (25 topics) and their performance measures under 10-fold cross validation are summarized.

	10-fold cross validation						
Topic	Number of training documents	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	321,408	35,670	42	1,063	97.11	99.88	98.47
bop	13,392	944	544	83	91.92	63.44	75.07
carcass	10,800	540	660	81	86.96	45.00	59.31
cocoa	10,800	687	513	22	96.90	57.25	71.97
coffee	23,760	2,204	436	269	89.12	83.48	86.21
corn	34,344	3,277	539	600	84.52	85.88	85.19
cpi	12,960	983	457	51	95.07	68.26	79.47
crude	75,384	8,145	231	806	91.00	97.24	94.02
dlr	20,736	1,937	367	144	93.08	84.07	88.35
earn	585,144	64,994	22	411	99.37	99.97	99.67
gnp	19,872	1,767	441	214	89.20	80.03	84.36
gold	20,304	1,889	367	105	94.73	83.73	88.89
grain	85,104	9,242	214	1,200	88.51	97.74	92.89
interest	62,424	6,590	346	541	92.41	95.01	93.69
livestock	15,768	1,115	637	137	89.06	63.64	74.23
money-fx	99,360	10,821	219	893	92.38	98.02	95.11
money-supply	18,792	1,807	281	64	96.58	86.54	91.29
nat-gas	15,552	1,226	502	109	91.84	70.95	80.05
oilseed	25,272	2,161	647	447	82.86	76.96	79.80
ship	41,256	4,216	368	544	88.57	91.97	90.24
soybean	15,768	1,080	672	221	83.01	61.64	70.75
sugar	25,488	2,362	470	377	86.24	83.40	84.80
trade	72,792	7,859	229	894	89.79	97.17	93.33
veg-oil	18,576	1,503	561	226	86.93	72.82	79.25
wheat	42,768	4,286	466	729	85.46	90.19	87.76

Table 22: The number of training documents of each topic (25 topics) and their performance measures under 12-fold cross validation are summarized.

	12-fold cross validation						
Topic	Number of training documents	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	392,832	35668	44	1046	97.15	99.88	98.50
bop	16,368	962	526	85	91.88	64.65	75.90
carcass	13,200	543	657	82	86.88	45.25	59.51
cocoa	13,200	696	504	25	96.53	58.00	72.46
coffee	29,040	2197	443	265	89.24	83.22	86.12
corn	41,976	3287	529	595	84.67	86.14	85.40
cpi	15,840	977	463	59	94.31	67.85	78.92
crude	92,136	8152	224	790	91.17	97.33	94.14
dlr	25,344	1920	384	140	93.20	83.33	87.99
earn	715,176	64996	20	403	99.38	99.97	99.68
gnp	24,288	1781	427	214	89.27	80.66	84.75
gold	24,816	1902	354	100	95.00	84.31	89.34
grain	104,016	9248	208	1199	88.52	97.80	92.93
interest	76,296	6585	351	538	92.45	94.94	93.68
livestock	19,272	1121	631	135	89.25	63.98	74.53
money-fx	121,440	10831	209	887	92.43	98.11	95.18
money-supply	22,968	1768	320	69	96.24	84.67	90.09
nat-gas	19,008	1221	507	111	91.67	70.66	79.80
oilseed	30,888	2171	637	452	82.77	77.31	79.95
ship	50,424	4227	357	539	88.69	92.21	90.42
soybean	19,272	1087	665	212	83.68	62.04	71.26
sugar	31,152	2373	459	390	85.88	83.79	84.83
trade	88,968	7866	222	889	89.85	97.26	93.40
veg-oil	22,704	1510	554	231	86.73	73.16	79.37
wheat	52,272	4280	472	711	85.75	90.07	87.86

For a sample test data set containing 7,814 test documents (25 topics) which has 187,536 (24 times of 7,814) test documents used in 300 topic pairs (25 topics) for SVM classifiers, the measurements of recall, precision and F1 plotted against the training documents number of 25 topics and against ranked topic (sorted by their F1 scores from the smallest value to the largest) under 8-fold, 10-fold and 12-fold cross validations are shown in Figure 23, Figure 24 and Figure 25.



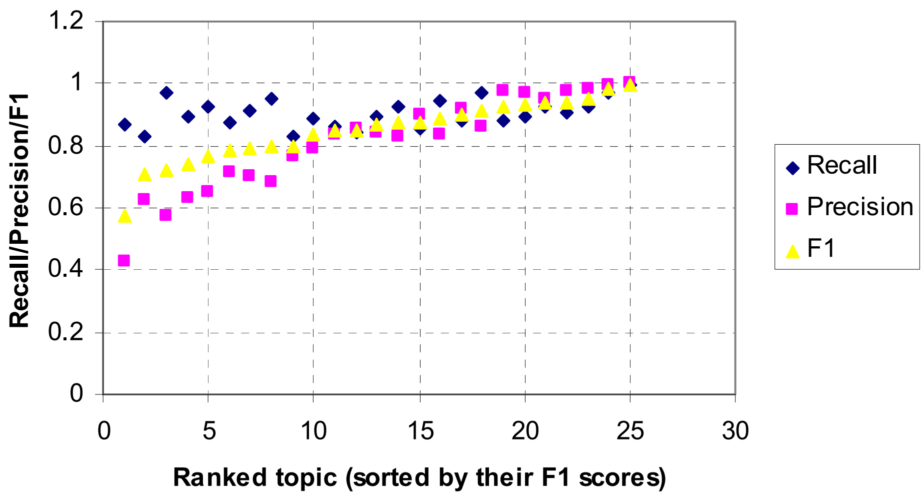


Figure 23: The distribution of recall/precision/F1 measurement under 8-fold cross validation.

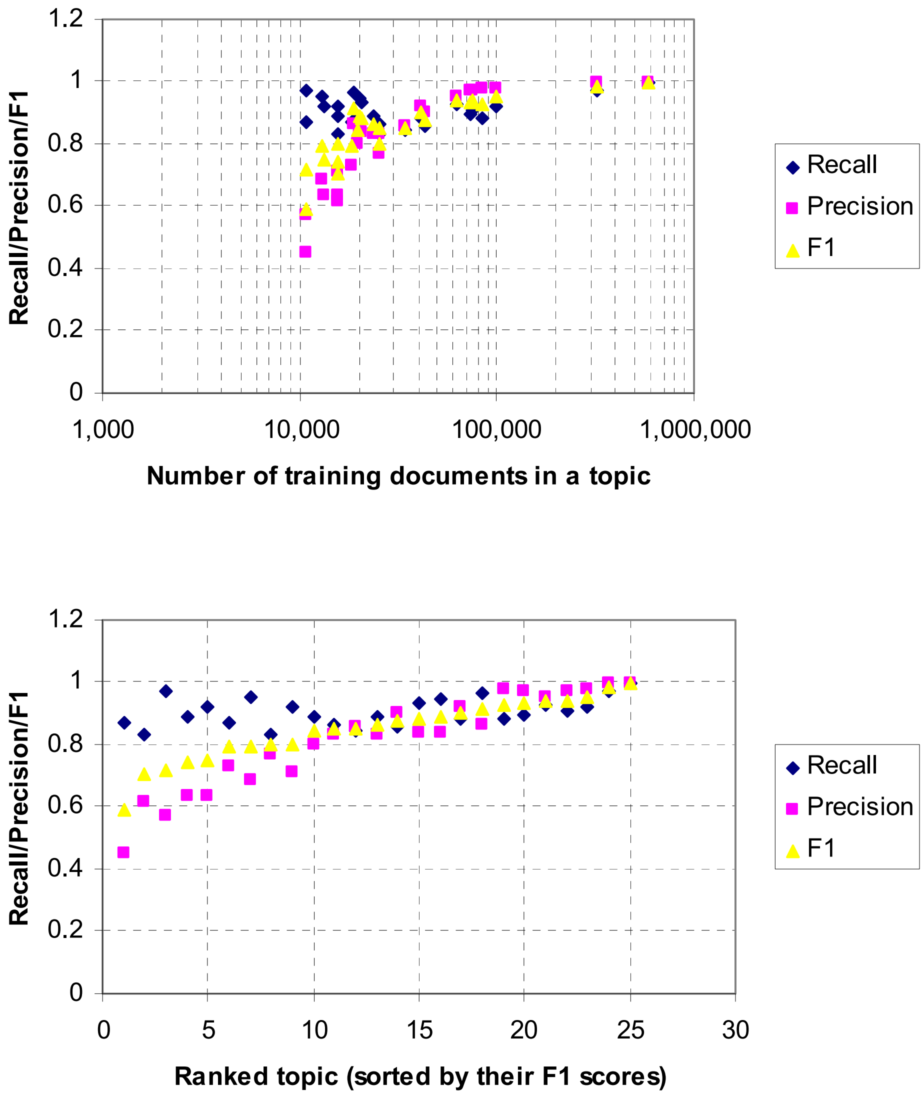


Figure 24: The distribution of recall/precision/F1 measurement under 10-fold cross validation.

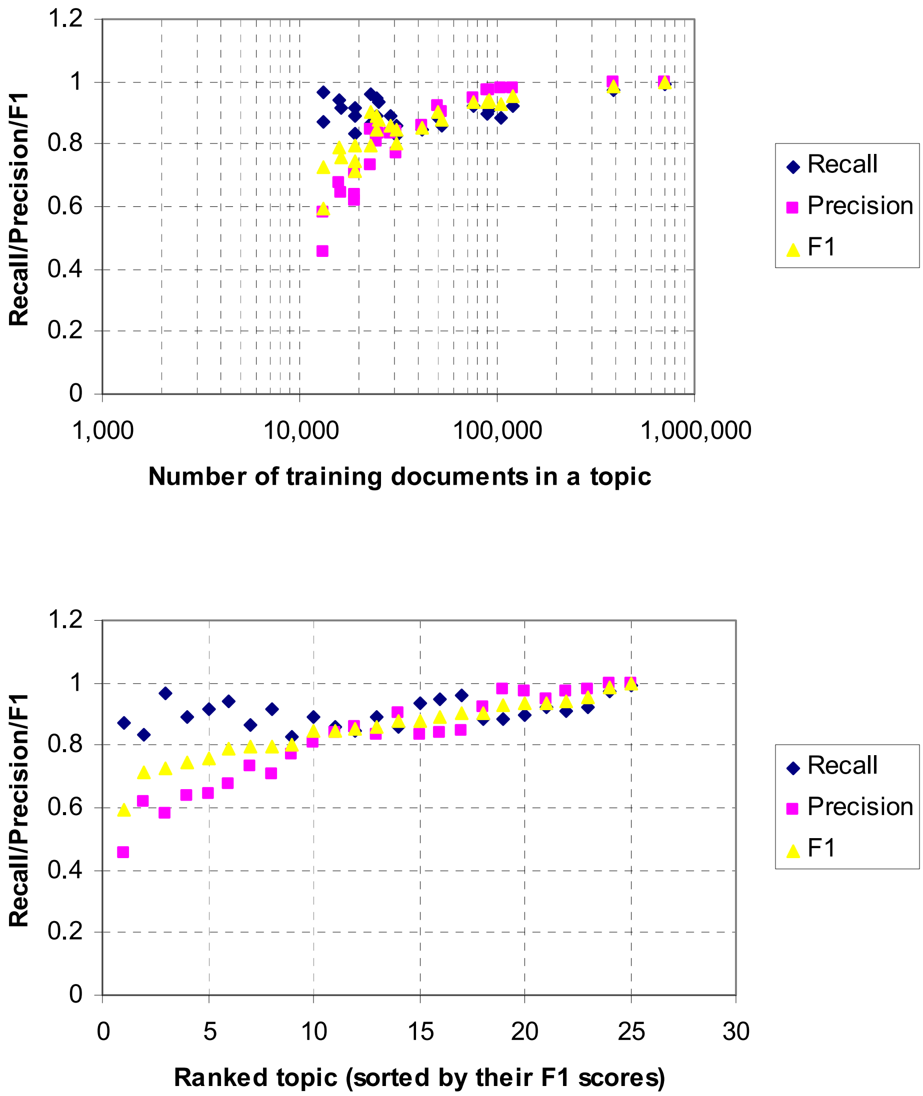


Figure 25: The distribution of recall/precision/F1 measurement under 12-fold cross validation.

Recall, precision and F1 measurement of the 300 topic pairs for SVM classifiers in the experimental data set are unevenly distributed. The uneven distribution is due to the fact that the distribution of the number of documents in the data set is highly skewed in nature.

To address multi-label classification, macro average and micro average are used to assess the overall performance across multiple labels. Macro-average performance scores are determined by first computing the performance measures per topic and then averaging these to compute the global means. Micro-average performance scores are determined by first computing the totals of a, b and c for all topics and then these totals are used to compute the performance measures. There is an important distinction between the two types of averaging. Micro averaging gives equal weight to every document, while macro averaging gives equal weight to each topic.

The results of macro-average and micro-average performance under 8-fold, 10-fold and 12-fold cross validations are shown in Table 23, Table 24 and Table 25.

Table 23: The macro-average and micro-average performance calculated under 8-fold cross validation are summarized.

Macro-average			Micro-average
Recall	Precision	F1	Recall/Precision/F1
90.46%	81.19%	84.82%	94.5%

From the result with 8-fold cross validation, the macro-average recall is 90.46%, macro-average precision is 81.19%, macro-average F1 is 84.82% and micro-average recall/precision/F1 is 94.5%.

Table 24: The macro-average and micro-average performance calculated under 10-fold cross validation are summarized.

Macro-average			Micro-average
Recall	Precision	F1	Recall/Precision/F1
90.50%	81.37%	84.97%	94.54%

From the result with 10-fold cross validation, the macro-average recall is 90.50% (0.04% higher than 8-fold cross validation), macro-average precision is 81.37% (0.22% higher than 8-fold cross validation), macro-average F1 is 84.97% (0.18% higher than 8-fold cross validation) and micro-average recall/precision/F1 is 94.54% (0.04% higher than 8-fold cross validation).

Table 25: The macro-average and micro-average performance calculated under 12-fold cross validation are summarized.

Macro-average			Micro-average
Recall	Precision	F1	Recall/Precision/F1
90.50%	81.46%	85.04%	94.58%

From the result with 12-fold cross validation, the macro-average recall is 90.50% (0.04% higher than 8-fold cross validation), macro-average precision is 81.46% (0.33% higher than 8-fold cross validation), macro-average F1 is 85.04% (0.26% higher than 8-fold cross validation) and micro-average recall/precision/F1 is 94.58% (0.08% higher than 8-fold cross validation).

The performance measures under 8-fold, 10-fold, and 12-fold are similar. Therefore some experiments such as Support Vector Machine soft margin classifier are done only 8-fold cross validation. It will be described in Section 5.3.3.

5.2.2.2 Confidence Level / Wilcoxon Matched-Pairs Signed-Ranks Test

The Wilcoxon Matched-Pairs Ranks test is a non-parametric alternative to a matched pairs t-test for the case of two related samples or repeated measurements on a single sample. The test is named for Frank Wilcoxon

(1892–1965) who, in a single paper, proposed both it and the rank-sum test for two independent samples [86].

Unlike less robust non-parametric tests such as the sign test:

- The Wilcoxon test is used to determine the magnitude of difference between matched groups.
- The Wilcoxon test is used to determine more than only the direction of difference.

Wilcoxon matched-pairs signed-ranks test is used to show the confidence level of the sample tests.

5.2.2.3 SVM-Light with different kernels

We use SVM-Light [38] to perform text classification. SVM-Light is an implementation of Vapnik's Support Vector Machine [81] for the problem of pattern recognition, for the problem of regression, and for the problem of learning a ranking function. The optimization algorithms used in SVM-Light are described in [33, 36]. The algorithm has scalable memory requirements and can handle problems with many thousands of support vectors efficiently. Three kernels (polynomial kernel, Gaussian radial basis function kernel, and sigmoid kernel) provided by the classifiers are considered as well to build the classifier while in training phase.

5.2.3 Algorithm

Figure 26 shows the algorithm for the iterative space method. The iterative space method can generate suitable features from suitable training documents. The unsuitable documents will form a residual set for the next level classification until all the training documents are used or stopping criteria (termination) is reached. The classification is done by support vector machines (SVM-Light).

The separation margin between two classes is generated by the SVM classifier. There are typically 4 types of separation margin. The details of these types are described in Section 5.2.4.

If the separation margin between two classes is well separated, the iteration can stop. It means the features are from the training documents are well learnt by the classifier. If not, the remaining documents will form the residual set for classification at the next level.

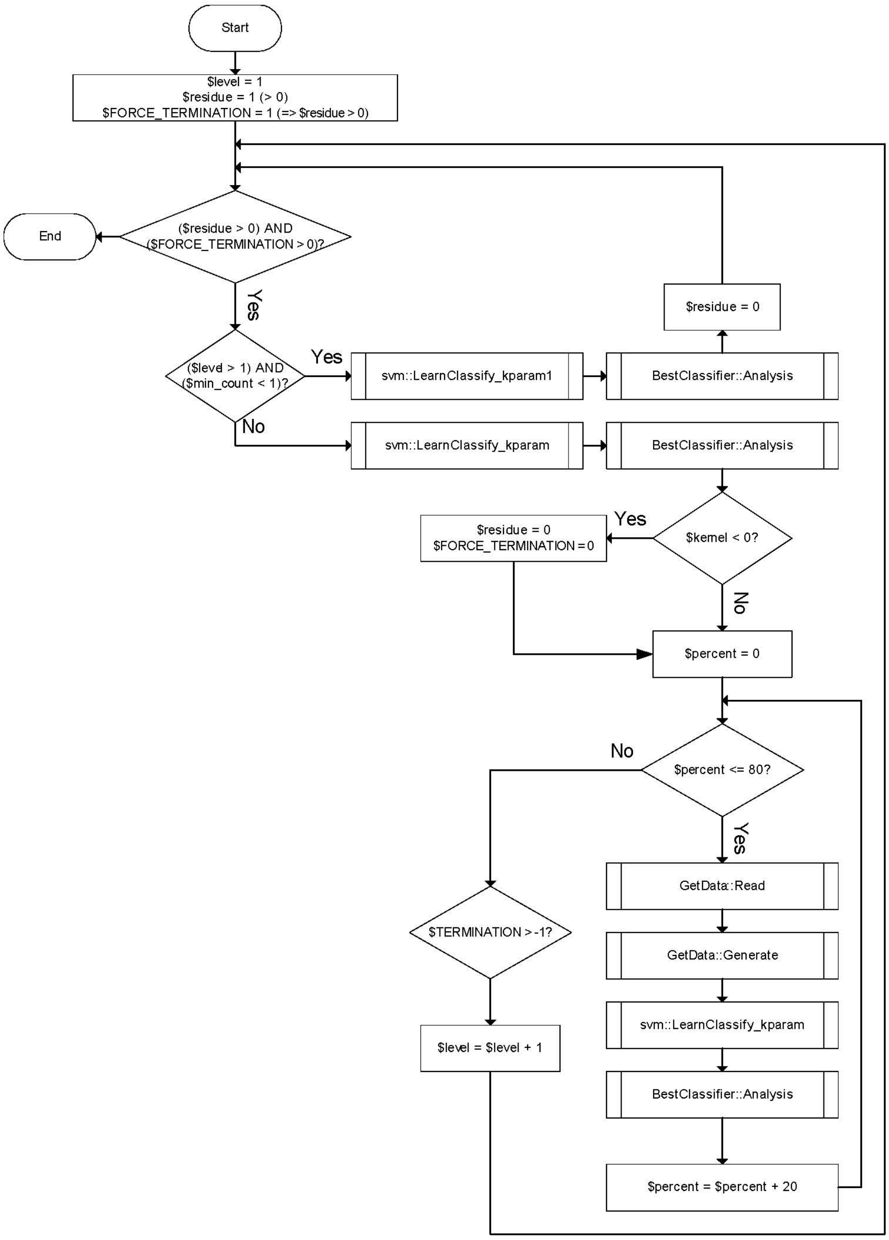


Figure 26: Implementation of the iterative space method.

5.2.4 Separation Margin

For the analysis involved in the scheme, there are 4 typically types of separation margin between two classes (Class A, Class B) which is generated by an SVM classifier. They are:

TYPE 1: Has overlap region and no clean region

TYPE 2: Has overlap region and one-clean region
(either Class A region clean or Class B region clean)

TYPE 3: Has overlap region and two-clean region
(both Class A region clean and Class B region clean)

TYPE 4: Has no overlap and two-clean region
(both Class A region clean and Class B region clean)

The 4 types of separation margin in terms of overlap and clean regions are summarized in Table 26. Overlap region has documents with Class A and Class B, clean region has documents with either Class A or Class B.

Table 26: The 4 types of separation margin in terms of overlap and clean regions.

	Overlap region	One-clean region	Two-clean region
Type 1	yes	-	-
Type 2	yes	yes	-
Type 3	yes	-	yes
Type 4	-	-	yes

The type of separation margin of two classes is calculated by using $C_{P\ Min.}$, $C_{P\ Max}$, $C_{N\ Min}$ and $C_{N\ Max}$

where

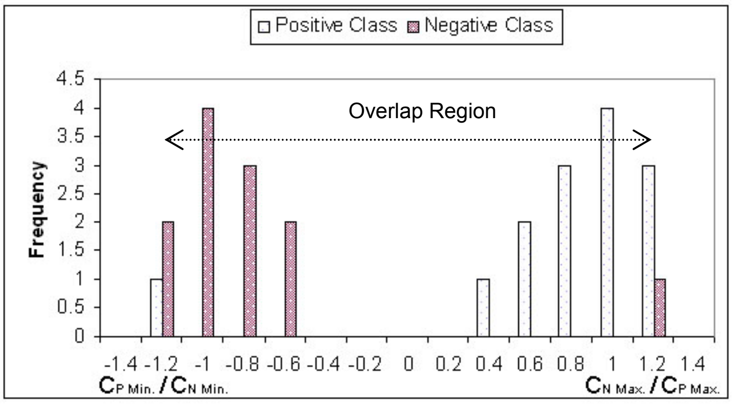
$C_{P\ Min.}$ = Positive Class (Class A) Minimum Value

$C_{P\ Max.}$ = Positive Class (Class A) Maximum Value

$C_{N\ Min.}$ = Negative Class (Class B) Minimum Value

$C_{N\ Max.}$ = Negative Class (Class B) Maximum Value

The histograms of these separation margins as well as their related properties and computations are illustrated in Figure 27, Figure 28, Figure 29 and Figure 30.



if ($C_{P\ Min.}$ <= $C_{N\ Min.}$) and ($C_{N\ Max.}$ >= $C_{P\ Max.}$) then Type 1

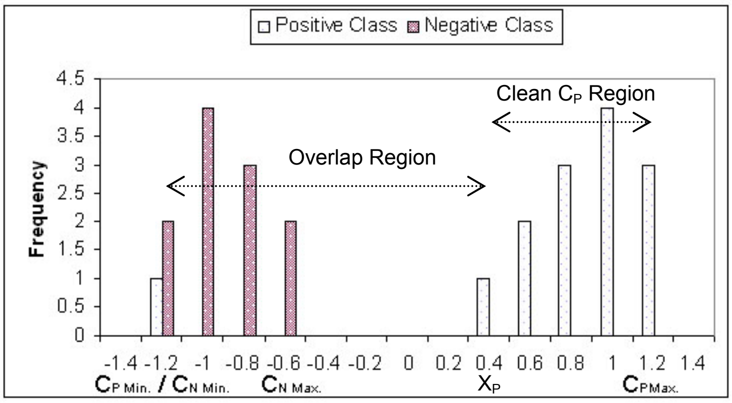
=> Clean C_P Region: width = 0

=> Clean C_N Region: width = 0

=> Overlap Region: width = $C_{N\ Max.} - C_{P\ Min.}$

end

Figure 27: Type 1 has overlap region and no clean region.



$$X_P = \min\{dist(positive) \mid dist(positive) > C_{N\ Max.}\}$$

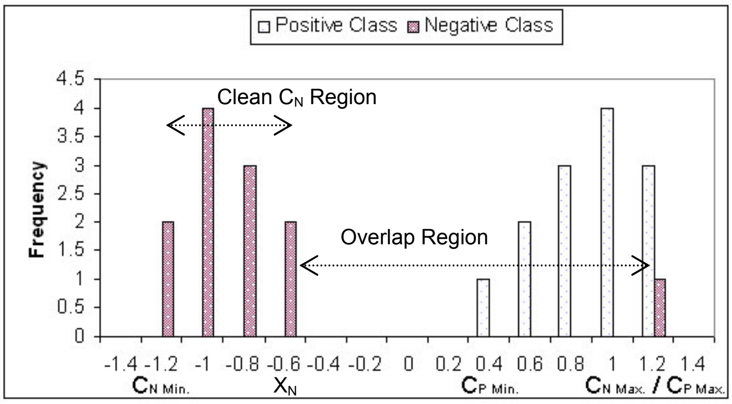
if ($C_{P\ Min.} <= C_{N\ Min.}$) and ($C_{P\ Max.} > C_{N\ Max.}$) then Type 2

=> Clean C_P Region: width = $C_{P\ Max.} - X_P$

=> Clean C_N Region: width = 0

=> Overlap Region: width = $X_P - C_{P\ Min.}$)

end



$$X_N = \min\{dist(negative) \mid dist(negative) < C_{P\ Min.}\}$$

if ($C_{N\ Max.} >= C_{P\ Max.}$) and ($C_{N\ Min.} < C_{P\ Min.}$) then Type 2

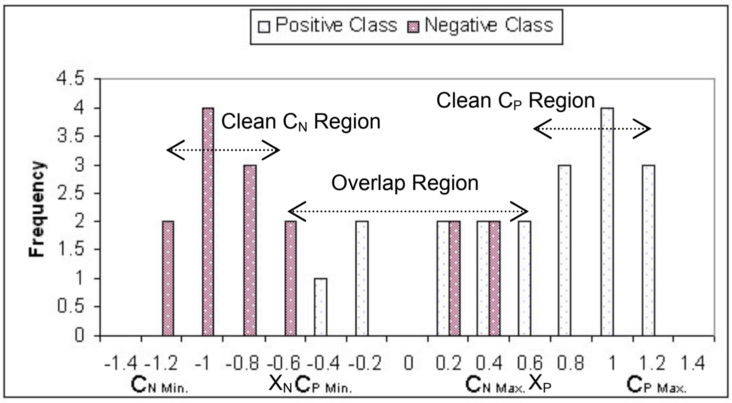
=> Clean C_P Region: width = 0

=> Clean C_N Region: width = $X_N - C_{N\ Min.}$

=> Overlap Region: width = $C_{N\ Max.} - X_N$

end

Figure 28: Type 2 has overlap region and one-clean region.



$X_P = \min\{dist(positive) \mid dist(positive) > C_{N\ Max.}\}$

$X_N = \min\{dist(negative) \mid dist(negative) < C_{P\ Min.}\}$

if ($C_{P\ Min.} < C_{N\ Max.}$) and ($C_{P\ Min.} > C_{N\ Min.}$) and ($C_{N\ Max.} < C_{P\ Max.}$) then Type3

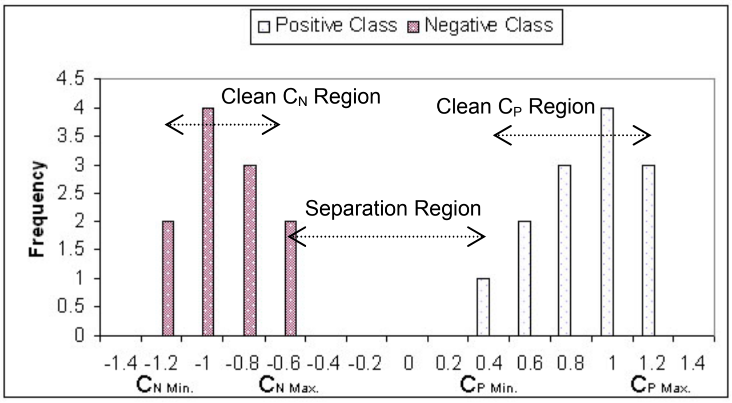
=> Clean C_P Region: width = $C_{P\ Max.}$ - X_P

=> Clean C_N Region: width = X_N - $C_{N\ Min.}$

=> Overlap Region: width = X_P - X_N

end

Figure 29: Type 3 has overlap region and two-clean region.



if ($C_{P\ Min.} >= C_{N\ Max.}$) and ($C_{P\ Min.} > C_{N\ Min.}$) and ($C_{N\ Max.} < C_{P\ Max.}$) then Type4

=> Clean C_P Region: width = $C_{P\ Max.}$ - $C_{P\ Min.}$

=> Clean C_N Region: width = $C_{N\ Max.}$ - $C_{N\ Min.}$

=> Separation Region: width = $C_{P\ Min.}$ - $C_{N\ Max.}$

end

Figure 30: Type 4 has no overlap and two-clean region.

5.3 Experimental Results and Discussion

The aim is to generate subspaces from different training data set through error-driven learning. Feature selection is done on the training data set and done recursively to build classifiers. Through the iteration, suitable features can be selected from different subspaces. The process will stop when all topics are learned to build classifiers. Sub-classifiers will be generated for assisting in document classification. Better category boundary is expected to be obtained through the learning of these cascade classifiers. For the iterative subspace generation, the basic scheme of this method is in Section 5.1.2 and the methodology is in 5.2.

It is new that the proposed iterative subspace model allows suitable features to be selected from different subspaces through the iterative process to obtain the better category boundary. The main difference of our proposed iterative subspace classifier from others is trying to find a set of suitable features (subspaces) for each category through the multi-level classification (classifier). In our case, Support Vector Machines (SVM-Light [38]) are used as classifiers in the experiments. The separation margin (SM) can be adjusted to generate subspaces from different training data set through error-driven learning.

5.3.1 Separation Margin (SM) set to 1.6, 1.8 and 2.0

In this experiment, we use the subset of Reuters-21578 collection. For providing enough training data learnt by the proposed Iterative Subspace Method, only those topics (categories) with training document sizes which are equal to or greater than 50 are used. 25 topics can meet this requirement and 300 topic pairs for SVM classifiers (binary classifiers) are generated for the experiment.

The experiment is done under 8-fold, 10-fold, and 12-fold cross validations; the training documents are sampled by systematic sampling (selected sequentially by system file ordering). The learning process will cease when any one of stopping criteria is reached. The stopping criteria are: (1) not enough data in the residual set, the size in the experiments is roughly set to be equal to one tenth of the training data; (2) the classifier for the next level can correctly classify the data with a separation margin greater than the predefined value from the data in the residual set. The predefined values used in the experiment are 1.6, 1.8 and 2.0.

Table 27 shows the numbers of improved topic (class) pairs with SM = 1.6, 1.8 and 2.0. The confidence level (CL) is calculated by the Wilcoxon Matched-Pairs Signed-Ranks Test [30] to see whether the results from standard method and iterative subspace method are significantly difference under 8 samples (8-fold cross validation), 10 samples (10-fold cross validation) and 12 samples (12-fold cross validation).

Table 27: The numbers of improved topic (class) pairs with SM = 1.6, 1.8 and 2.0 are summarized.

CL (x%)	8-fold, SM = 1.6	1.8	2.0	10-fold, SM = 1.6	1.8	2.0	12-fold, SM = 1.6	1.8	2.0
x≥90	2	1	6	2	2	6	1	1	6
90>x≥80	-	1	3	-	-	1	-	-	3
80>x≥70	-	-	1	-	-	1	-	-	2
70>x≥60	-	-	-	-	-	1	-	-	-
60>x≥50	-	2	2	-	-	1	-	1	-
50>x≥40	-	-	-	-	-	-	-	-	-
40>x≥30	-	-	-	-	1	1	-	-	1
30>x≥20	-	-	-	-	-	-	-	-	-
20>x≥10	-	2	2	-	-	1	1	1	2
10>x≥0	-	1	3	-	-	2	-	-	3

For confidence levels greater than or equal to 80 (≥80), the numbers of improved topic (class) pairs with separation margins set to 2.0 (SM = 2.0) are more than both SM = 1.8 and SM = 1.6 under different fold cross validations. It is ideal that separation margins set to 2.0 at all levels of a classifier, hence documents which fall into the margin can have higher chance to be retrained at the next level. Our proposed approach can get the benefit of separation margin set to 2.0 and the following experiments will be reported on separation margin set to 2.0.

5.3.2 Number of classifier with SM set to 2.0

Table 28, Table 29 and Table 30 show the number of training documents of each topic (25 topics) and their performance measures (such as *a*, *b*, *c* for calculating recall, precision and F1) evaluated by iterative subspace method under 8-fold, 10-fold and 12-fold cross validations.

Table 28: The number of training documents of each topic (25 topics) and their performance measures under 8-fold cross validation are summarized.

	8-fold cross validation						
Topic	Number of training documents	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	249,984	34,108	1,604	1,705	95.24	95.51	95.37
bop	10,416	952	536	179	84.17	63.98	72.70
carcass	8,400	508	692	155	76.62	42.33	54.54
cocoa	8,400	621	579	35	94.66	51.75	66.92
coffee	18,480	2,187	453	754	74.36	82.84	78.37
corn	26,712	2,879	937	805	78.15	75.45	76.77
cpi	10,080	910	530	107	89.48	63.19	74.07
crude	58,632	7,900	476	1,881	80.77	94.32	87.02
dlr	16,128	1,839	465	291	86.34	79.82	82.95
earn	455,112	64,993	23	431	99.34	99.96	99.65
gnp	15,456	1,762	446	415	80.94	79.80	80.36
gold	15,792	1,874	382	102	94.84	83.07	88.56
grain	66,192	8,313	1,143	1,530	84.46	87.91	86.15
interest	48,552	6,493	443	757	89.56	93.61	91.54
livestock	12,264	1,196	556	378	75.98	68.26	71.92
money-fx	77,280	10,381	659	1,590	86.72	94.03	90.23
money-supply	14,616	1,725	363	111	93.95	82.61	87.92
nat-gas	12,096	1,190	538	102	92.11	68.87	78.81
oilseed	19,656	2,052	756	868	70.27	73.08	71.65
ship	32,088	3,648	936	1,049	77.67	79.58	78.61
soybean	12,264	1,095	657	260	80.81	62.50	70.49
sugar	19,824	2,072	760	713	74.40	73.16	73.78
trade	56,616	6,756	1,332	1,159	85.36	83.53	84.43
veg-oil	14,448	1,370	694	568	70.69	66.38	68.47
wheat	33,264	3,829	923	938	80.32	80.58	80.45

Table 29: The number of training documents of each topic (25 topics) and their performance measures under 10-fold cross validation are summarized.

	10-fold cross validation						
Topic	Number of training documents	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	321,408	33,870	1,842	1,609	95.46	94.84	95.15
bop	13,392	926	562	135	87.28	62.23	72.66
carcass	10,800	564	636	191	74.70	47.00	57.70
cocoa	10,800	641	559	84	88.41	53.42	66.60
coffee	23,760	2,091	549	664	75.90	79.20	77.52
corn	34,344	2,886	930	834	77.58	75.63	76.59
cpi	12,960	934	506	106	89.81	64.86	75.32
crude	75,384	7,710	666	1,739	81.60	92.05	86.51
dlr	20,736	1,842	462	325	85.00	79.95	82.40
earn	585,144	64,993	23	435	99.34	99.96	99.65
gnp	19,872	1,823	385	521	77.77	82.56	80.10
gold	20,304	1,855	401	101	94.84	82.23	88.08
grain	85,104	8,314	1,142	1,634	83.57	87.92	85.69
interest	62,424	6,414	522	691	90.27	92.47	91.36
livestock	15,768	1,208	544	425	73.97	68.95	71.37
money-fx	99,360	10,493	547	1,736	85.80	95.05	90.19
money-supply	18,792	1,698	390	72	95.93	81.32	88.02
nat-gas	15,552	1,199	529	101	92.23	69.39	79.19
oilseed	25,272	2,085	723	845	71.16	74.25	72.67
ship	41,256	3,743	841	1,068	77.80	81.65	79.68
soybean	15,768	1,054	698	204	83.78	60.16	70.03
sugar	25,488	2,143	689	730	74.59	75.67	75.13
trade	72,792	6,690	1,398	1,334	83.37	82.72	83.04
veg-oil	18,576	1,396	668	583	70.54	67.64	69.06
wheat	42,768	3,848	904	949	80.22	80.98	80.59

Table 30: The number of training documents of each topic (25 topics) and their performance measures under 12-fold cross validation are summarized.

	12-fold cross validation						
Topic	Number of training documents	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	392,832	34,325	1,387	1,707	95.26	96.12	95.69
bop	16,368	942	546	139	87.14	63.31	73.34
carcass	13,200	555	645	208	72.74	46.25	56.55
cocoa	13,200	629	571	54	92.09	52.42	66.81
coffee	29,040	2,217	423	925	70.56	83.98	76.69
corn	41,976	2,934	882	828	77.99	76.89	77.43
cpi	15,840	906	534	84	91.52	62.92	74.57
crude	92,136	7,635	741	1,858	80.43	91.15	85.46
dlr	25,344	1,849	455	288	86.52	80.25	83.27
earn	715,176	64,994	22	413	99.37	99.97	99.67
gnp	24,288	1,802	406	472	79.24	81.61	80.41
gold	24,816	1,880	376	97	95.09	83.33	88.83
grain	104,016	8,248	1,208	1,476	84.82	87.23	86.01
interest	76,296	6,392	544	685	90.32	92.16	91.23
livestock	19,272	1,207	545	402	75.02	68.89	71.82
money-fx	121,440	10,321	719	1,666	86.10	93.49	89.64
money-supply	22,968	1,685	403	120	93.35	80.70	86.57
nat-gas	19,008	1,211	517	105	92.02	70.08	79.57
oilseed	30,888	1,993	815	792	71.56	70.98	71.27
ship	50,424	3,669	915	937	79.66	80.04	79.85
soybean	19,272	1,049	703	223	82.47	59.87	69.38
sugar	31,152	2,183	649	894	70.95	77.08	73.89
trade	88,968	6,627	1,461	1,191	84.77	81.94	83.33
veg-oil	22,704	1,401	663	587	70.47	67.88	69.15
wheat	52,272	3,789	963	942	80.09	79.73	79.91

Referring to Table 27 (Section 5.3.1), there are 25 topic pairs involved. These topic pairs are further investigated. For the topic pairs, the min levels and max levels of SVM classifiers used to train them under 8 samples (8-fold cross validation), 10 samples (10-fold cross validation) and 12 samples (12-fold cross validation) are summarized in Table 31.

Table 31: The min level and max level of classifiers used to train topic pairs under 8 samples (8-fold cross validation), 10 samples (10-fold cross validation) and 12 samples (12-fold cross validation).

	Topic pair	8-fold, level = (min)	(max)	10-fold, level = (min)	(max)	12-fold, level = (min)	(max)
1	bop_coffee	1	3	1	2	1	4
2	bop_trade	20	29	17	29	13	30
3	bop_veg-oil	-	-	-	-	1	5
4	carcass_livestock	19	22	21	23	21	22
5	carcass_ship	-	-	-	-	1	4
6	cocoa_soybean	1	2	1	3	-	-
7	cocoa_wheat	-	-	1	18	-	-
8	cpi_dlr	1	3	1	3	1	5
9	cpi_nat-gas	-	-	1	2	-	-
10	dlr_gnp	-	-	-	-	-	-
11	dlr_money-fx	12	37	18	39	7	39
12	gnp_crude	1	13	1	3	1	12
13	gnp_grain	1	15	1	36	1	28
14	livestock_ship	1	3	1	4	1	4
15	livestock_trade	1	17	1	22	1	29
16	money-supply_trade	1	4	1	3	-	-
17	nat-gas_crude	30	33	29	34	29	34
18	nat-gas_sugar	-	-	-	-	1	7
19	oilseed_grain	45	47	45	49	46	49
20	soybean_corn	30	32	31	34	30	33
21	soybean_grain	28	33	30	34	31	35
22	soybean_oilseed	28	32	28	33	28	32
23	soybean_trade	1	4	1	4	1	4
24	soybean_wheat	30	33	31	33	33	35
25	sugar_acq	-	-	-	-	1	2

From the results in Table 31, some topic pairs need more levels of SVM classifiers than others to build the multi-level classifiers. For examples:

1. bop_trade
2. carcass_livestock
3. dlr_money-fx
4. gnp_grain
5. nat-gas_crude
6. oilseed_grain
7. soybean_corn
8. soybean_grain
9. soybean_oilseed
10. soybean_wheat

Confidence levels of 10 topic pairs with more levels of SVM classifiers than others to build the multi-level classifiers under 8 samples (8-fold cross validation), 10 samples (10-fold cross validation) and 12 samples (12-fold cross validation) are summarized in Table 32. It is found that almost the topic pairs that are well trained by our proposal scheme (iterative subspace method) can have the improvements with high confidence level. To further investigate the classification result, these topic pairs (excluding gnp_grain) are used for the comparison between 1-level classifier and multi-level classifier (iterative subspace method) in Section 5.3.6.

Table 32: Confidence levels of 10 topic pairs with more levels of SVM classifiers than others to build the multi-level classifiers under 8 samples (8-fold cross validation), 10 samples (10-fold cross validation) and 12 samples (12-fold cross validation).

	Confidence Level		
Topic pair	8-fold	10-fold	12-fold
bop_trade	75	93.75	75
carcass_livestock	98.438	99.8047	99.8047
dlr_money-fx	99.2188	98.438	87.11
gnp_grain	87.5	0	31.25
nat-gas_crude	93.75	75	93.75
oilseed_grain	93.75	98.438	89.45
soybean_corn	96.875	98.047	98.438
soybean_grain	87.5	87.5	93.75
soybean_oilseed	99.2188	99.6094	99.8047
soybean_wheat	89.06	61.72	92.578

5.3.3 Support Vector Machine (SVM) Soft Margin Classifier Experiments

Support vector machine soft margin classifiers introduced by Cortes and Vapnik [11] are important learning algorithms for classification problems. For the experiments, SVM-Light classifiers with different soft margins (trade-off between training error and margin) are used to perform the evaluation. 12 experiments with different c (float number) parameters (with SVM-Light classifier) are selected and they are 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 10 and 100. The performance scores are generated under 8-fold cross validation and shown in Appendix.

From these 12 experiments, the best result is obtained when c parameter is set to 10. The result is shown in Table 33.

Table 33: The macro-average and micro-average performance of Soft Margin SVM (with c=10) evaluated under 8-fold cross validation are summarized.

	Macro-average			Micro-average
Classifier	Recall	Precision	F1	Recall/Precision/F1
Standard SVM	90.46%	81.19%	84.82%	94.5%
Soft Margin SVM	87.21%	91.69%	89.22%	95.88%

There is also 11.85% improvement of "nat-gas" evaluated by multi-level classifier (proposed iterative subspace method) comparing to Soft Margin SVM classifier under 8-fold cross validation.

Table 34: 2 topics out of 25 topics have better recall than the performances of Soft Margin SVM classifier under 8-fold cross validation.

	Soft Margin SVM		Multi-level classifier		
Topic	Recall	Precision	Recall	Precision	Improvement
gold	90.43%	-	94.84%	-	4.88%
nat-gas	82.35%	-	92.11%	-	11.85%

5.3.4 Support Vector Machine (SVM) Soft Margin Classifier with Iterative Subspace Method

In this experiment, we use the subset of Reuters-21578 collection. For providing enough training data learnt by the proposed Iterative Subspace Method, only those topics (categories) with training document sizes which are equal to or greater than 50 are used. 25 topics can meet this requirement and 300 topic pairs for SVM classifiers (binary classifiers) are generated for the experiment. It is the same as in Section 5.3.1

The experiment is done under 8-fold cross validations; the training documents are sampled by systematic sampling (selected sequentially by system file ordering). The learning process will cease when any one of stopping criteria is reached. The stopping criteria are: (1) not enough data in the residual set, the size in the experiments is roughly set to be equal to one tenth of the training data; (2) the classifier for the next level can correctly classify the data with a separation margin greater than the predefined value from the data in the residual set. The predefined value used in the experiment is 2.0.

From the finding in Section 5.3.3, the best result is obtained when c parameter is set to 10. The predefined c value of. SVM-Light classifier with soft margin used in the experiment is set to 10 to perform the evaluation. SVM classifiers with fixed linear and polynomial kernel functions are used for the comparison. The performance scores are generated under 8-fold cross validation and shown in Table 35.

Table 35: The macro-average and micro-average performance of Iterative Subspace Method (multi-level classifier with soft margin c=10) evaluated under 8-fold cross validation are summarized.

SVM	kernel function	Macro-average Recall	Macro-average Precision	Macro-average F1	Micro-average Recall/Precision/F1
1- level	adaptive	90.70%	83.17%	86.77%	94.87%
Multi-level	adaptive	90.69%	83.17%	86.76%	94.86%
1- level	linear	91.69%	87.21%	89.40%	95.88%
Multi-level	linear	91.76%	86.88%	89.25%	95.78%
1-level	polynomial	91.69%	87.21%	89.39%	95.88%
Multi-level	polynomial	91.76%	86.89%	89.26%	95.78%

5.3.5 Comparison of Macro Averaging and Micro Averaging between 1-level classifier and multi-level classifier (Iterative Subspace Method)

The results of macro-average and micro-average performance evaluated by 1-level classifier and multi-level classifier (iterative subspace method) under 8-fold, 10-fold and 12-fold cross validations are shown in Table 36, Table 37 and Table 38.

Table 36: The macro-average and micro-average performance of Iterative Subspace Method (multi-level classifier) evaluated under 8-fold cross validation are summarized.

Classifier	Macro-average Recall	Macro-average Precision	Macro-average F1	Micro-average Recall/Precision/F1
1-level (standard SVM)	90.46%	81.19%	84.82%	94.5%
Multi-level	83.89%	77.05%	79.67%	91%

From the result of our proposed iterative subspace method (multi-level classifier) with 8-fold cross validation, macro-average F1 (79.67%) and micro-average F1 (91%) are not improved comparing to standard SVM method (1-level classifier) where macro-average F1 is 84.82% and micro-average F1 is 94.5%.

Table 37: The macro-average and micro-average performance of Iterative Subspace Method (multi-level classifier) evaluated under 10-fold cross validation are summarized.

	Macro-average			Micro-average
Classifier	Recall	Precision	F1	Recall/Precision/F1
1-level (standard SVM)	90.50%	81.37%	84.97%	94.54%
Multi-level	83.64%	77.28%	79.77%	90.87%

From the result of our proposed iterative subspace method (multi-level classifier) with 10-fold cross validation, macro-average F1 (79.77%) and micro-average F1 (90.87%) are not improved comparing to standard SVM method (1-level classifier) where macro-average F1 is 84.97% and micro-average F1 is 94.54%.

Table 38: The macro-average and micro-average performance of Iterative Subspace Method (multi-level classifier) evaluated under 12-fold cross validation are summarized.

	Macro-average			Micro-average
Classifier	Recall	Precision	F1	Recall/Precision/F1
1-level (standard SVM)	90.50%	81.46%	85.04%	94.58%
Multi-level	83.58%	77.13%	79.61%	90.89%

From the result of our proposed iterative subspace method (multi-level classifier) with 12-fold cross validation, macro-average F1 (79.61%) and micro-average F1 (90.89%) are not improved comparing to standard SVM method (1-level classifier) where macro-average F1 is 85.04% and micro-average F1 is 94.58%.

From the results, macro-averaging and micro-averaging performances of proposed iterative subspace method are not better than the performances of standard SVM method. However, some topics out of 25 topics have better precision or recall (from Table 28, Table 29 and Table 30) than the performances of standard SVM (from Table 20, Table 21 and Table 22).

Table 39, Table 40 and Table 41 show the recall or precision improvements under 8-fold, 10-fold and 12-fold cross validation.

Table 39: 4 topics out of 25 topics have better precision or recall than the performances of standard SVM under 8-fold cross validation.

	1-level classifier (standard SVM)		Multi-level classifier		
Topic	Recall	Precision	Recall	Precision	Improvement
gold	94.65%	-	94.84%	-	0.2%
livestock	-	63.07%	-	68.27%	8.24%
nat-gas	91.58%	-	92.11%	-	0.58%
soybean	-	62.33%	-	62.5%	0.27%

Table 40: 4 topics out of 25 topics have better precision or recall than the performances of standard SVM under 10-fold cross validation.

	1-level classifier (standard SVM)		Multi-level classifier		
Topic	Recall	Precision	Recall	Precision	Improvement
carcass	-	45%	-	47%	4.44%
gnp	-	80.03%	-	82.56%	3.16%
gold	94.73%	-	94.84%	-	0.12%
soybean	83.01%	-	83.78%	-	0.93%

Table 41: Topics out of 25 topics have better precision or recall than the performances of standard SVM under 12-fold cross validation.

	1-level classifier (standard SVM)		Multi-level classifier		
Topic	Recall	Precision	Recall	Precision	Improvement
coffee	-	83.22%	-	83.98%	0.91%
gold	95.01%	-	95.09%	-	0.08%
nat-gas	91.67%	-	92.02%	-	0.38%

It is still promising that there is 8.24% precision improvement of "livestock" evaluated by multi-level classifier (proposed iterative subspace method) comparing to 1-level classifier (standard SVM) under 8-fold cross validation.

In Section 5.3.4, SVM soft margin classifier shows the proposed iterative subspace method can perform effectively. The performance measures between 1-level (standard SVM) and multi-level (iterative subspace method) are significant reduced. The minimum difference of F1 measure is 0.01%

and the maximum difference of F1 measure is 0.15% (from Table 35). From Table 36, Table 37 and Table 38, the minimum difference of F1 measure is 3.5% and the maximum difference of F1 measure is 5.43%. The performance and efficiency can be affected by different widths of separation margin (soft margin). It is expected that the performance can be further improved by using other optimization techniques.

5.3.6 Comparison between 1-level classifier and multi-level classifier (Iterative Subspace Method)

For 8 samples (8-fold cross validation), the classification results of 9 topic pairs with high confidence levels and well trained classifiers (more levels of SVM classifiers) than others are shown in Figure 31 to Figure 38.

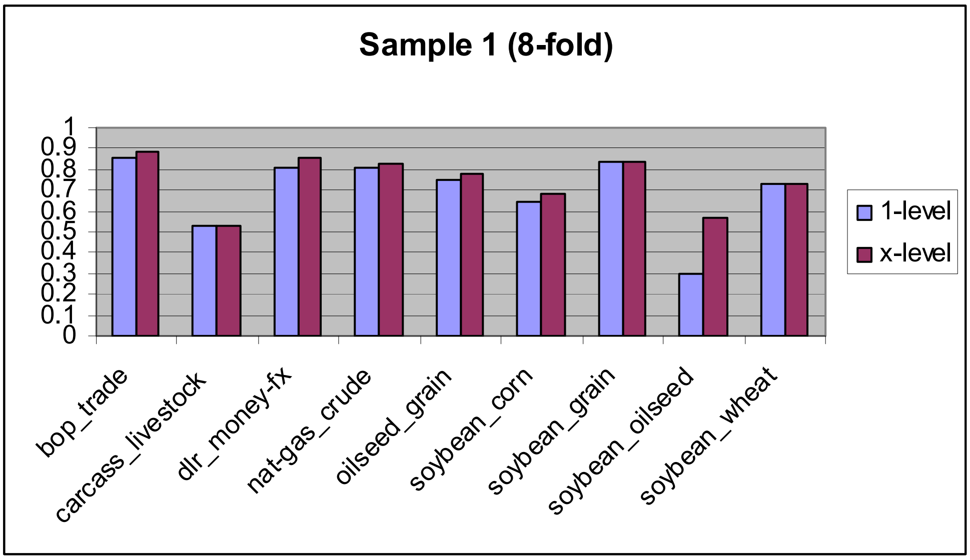


Figure 31: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 1).

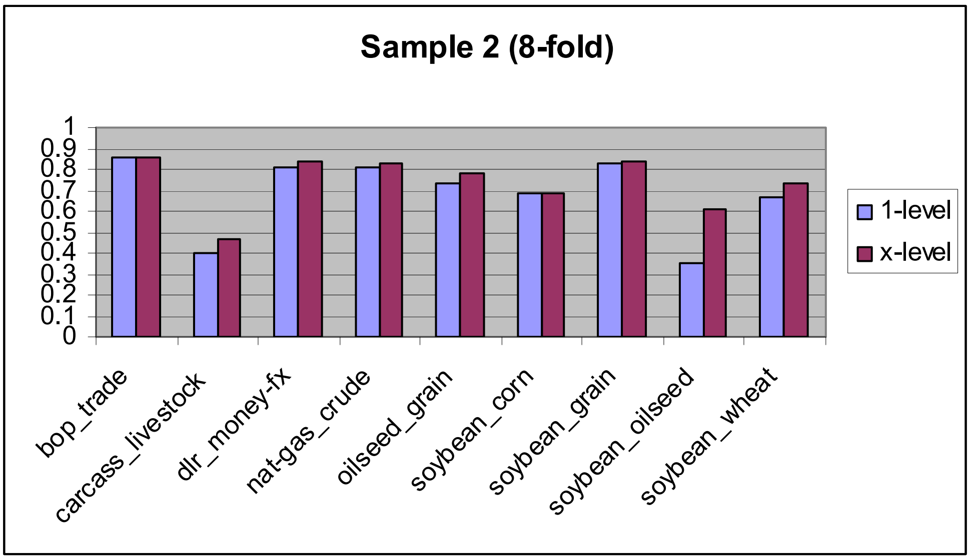


Figure 32: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 2).

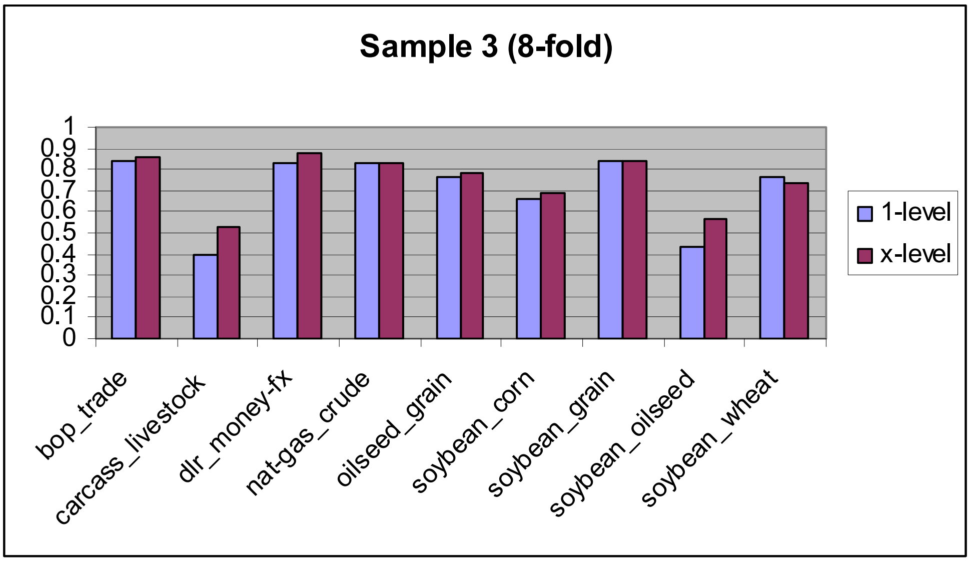


Figure 33: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 3).

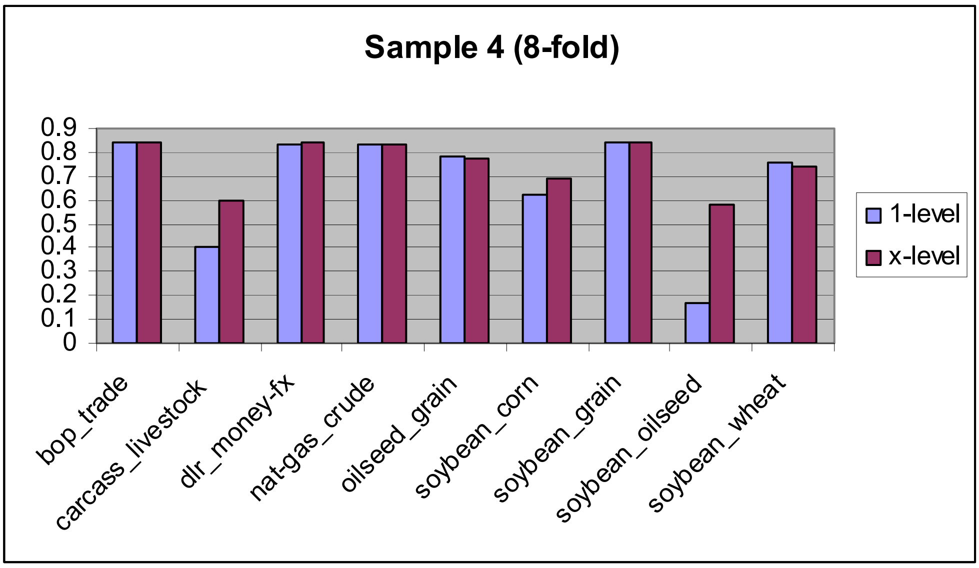


Figure 34: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 4).

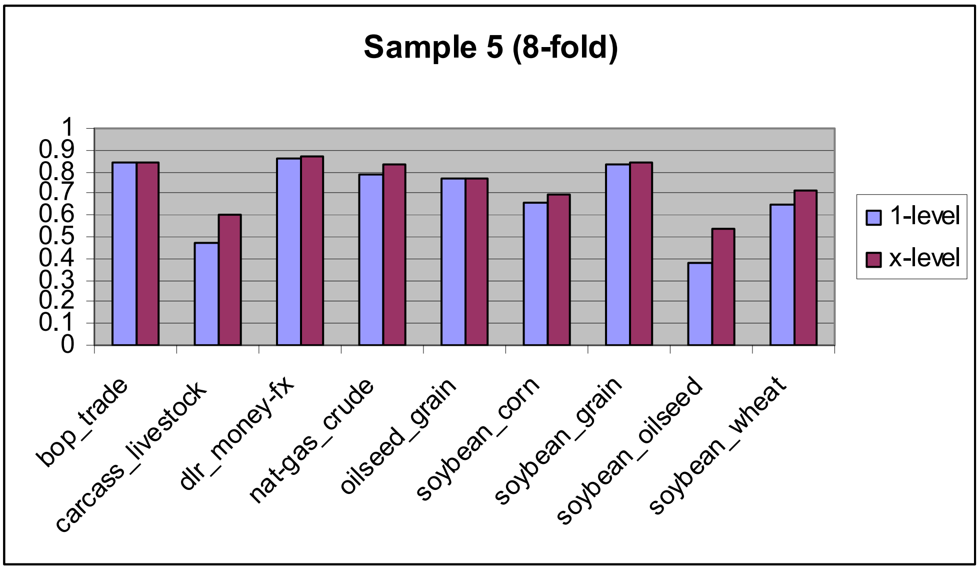


Figure 35: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 5).

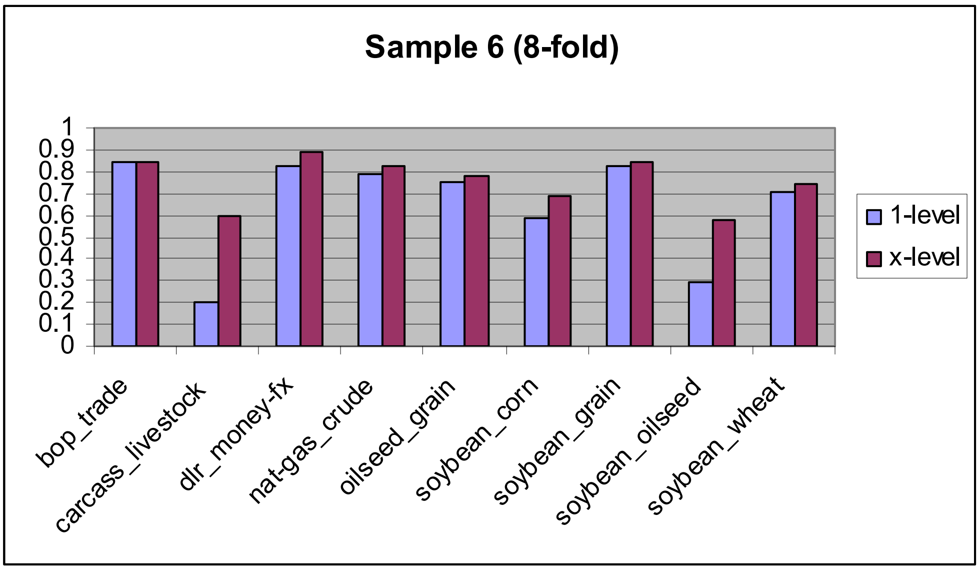


Figure 36: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 6).

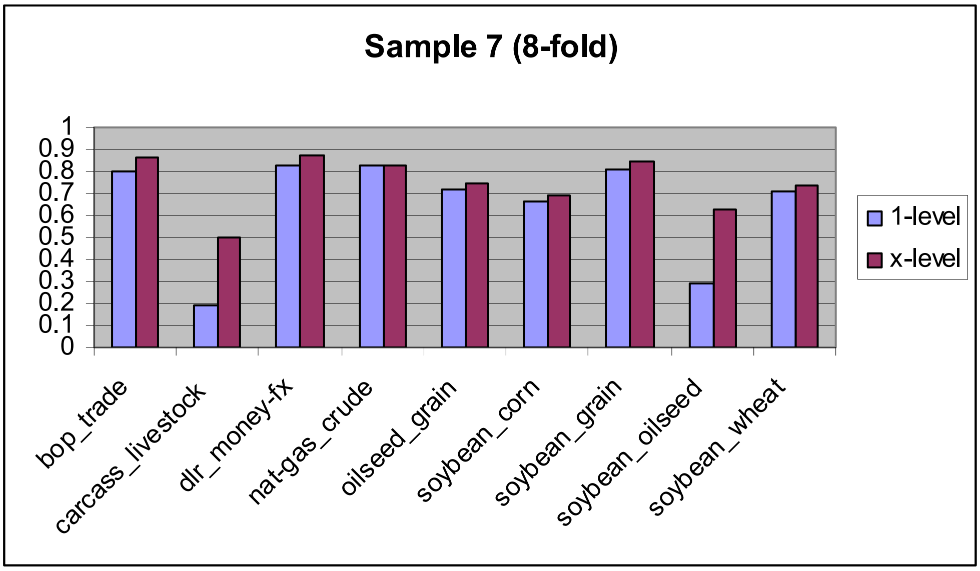


Figure 37: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 7).

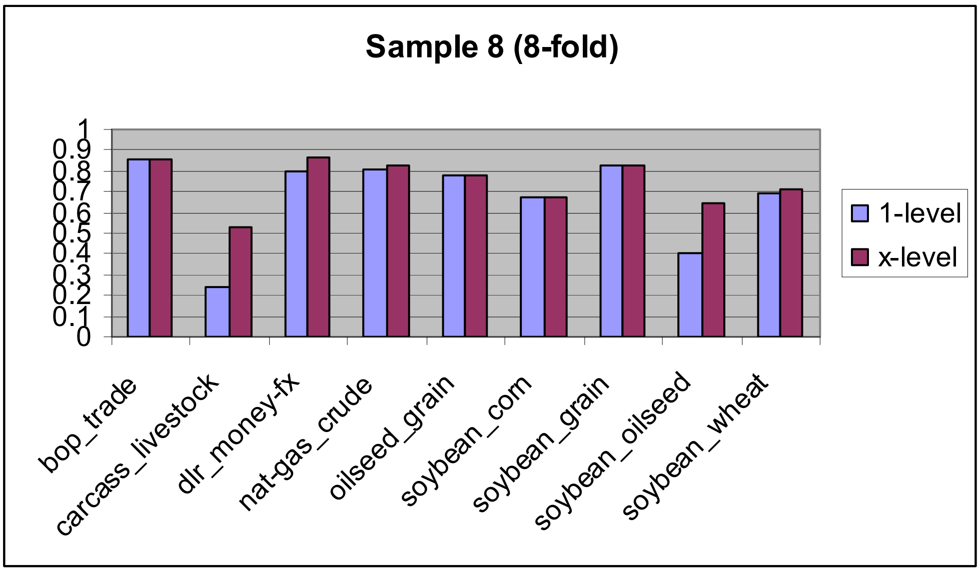


Figure 38: Classification results of 1-level classifier and multi-level classifier with 8-fold cross validation (sample 8).

For 10 samples (10-fold cross validation), the classification results of 9 topic pairs with high confidence levels and well trained classifiers (more levels of SVM classifiers) than others are shown in Figure 39 to Figure 48.

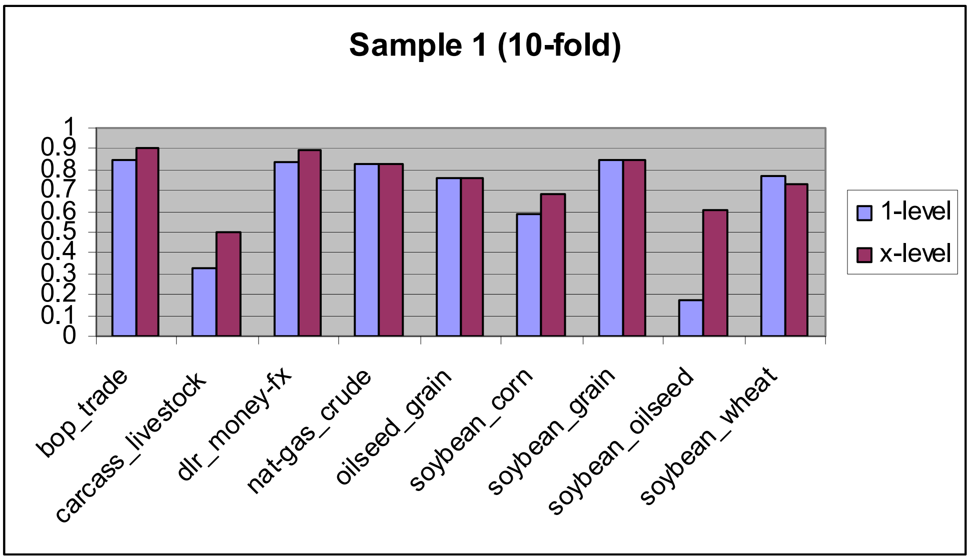


Figure 39: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 1).

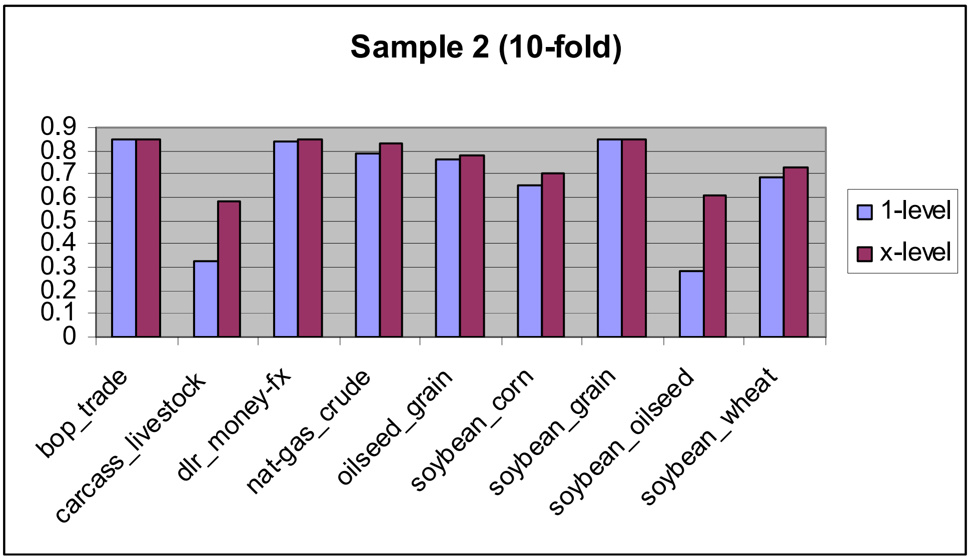


Figure 40: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 2).

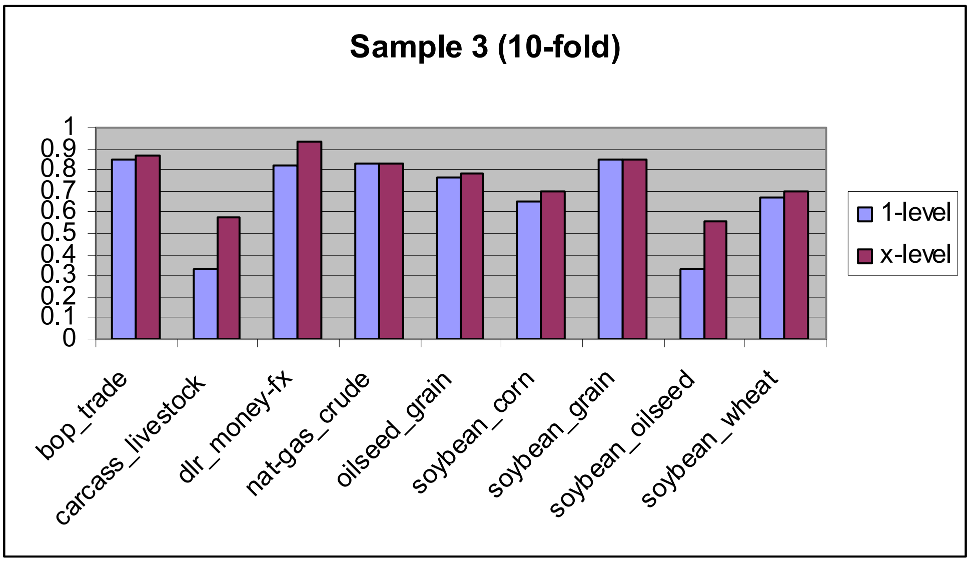


Figure 41: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 3).

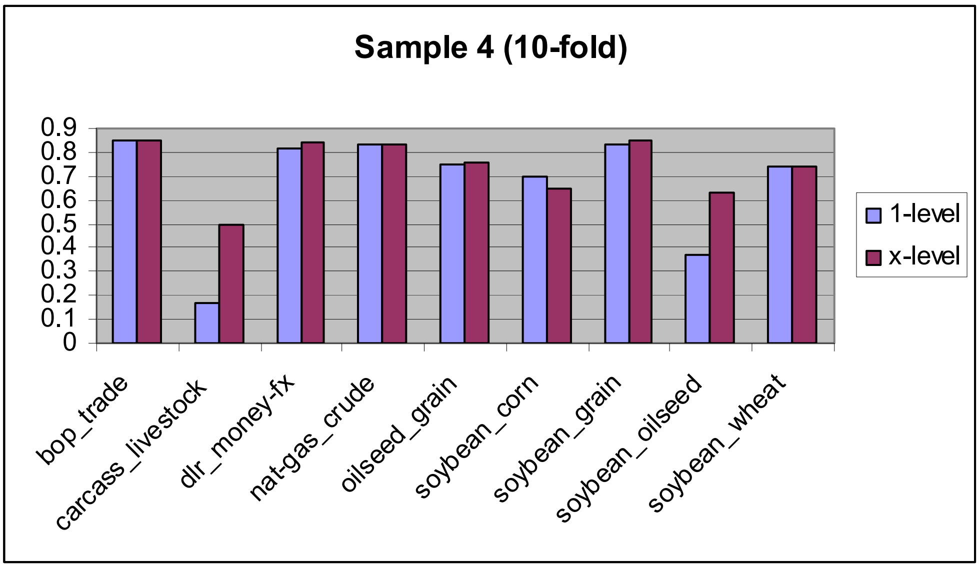


Figure 42: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 4).

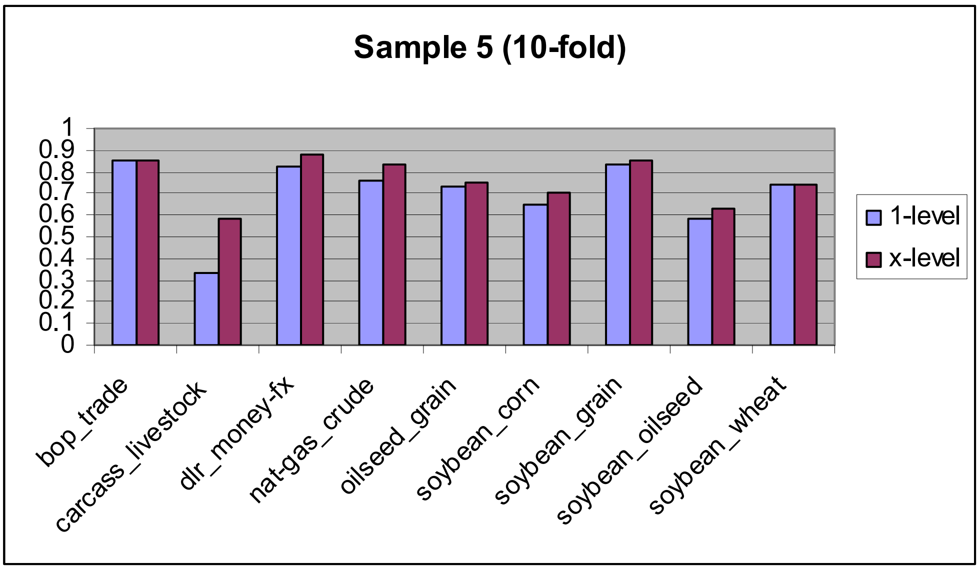


Figure 43: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 5).

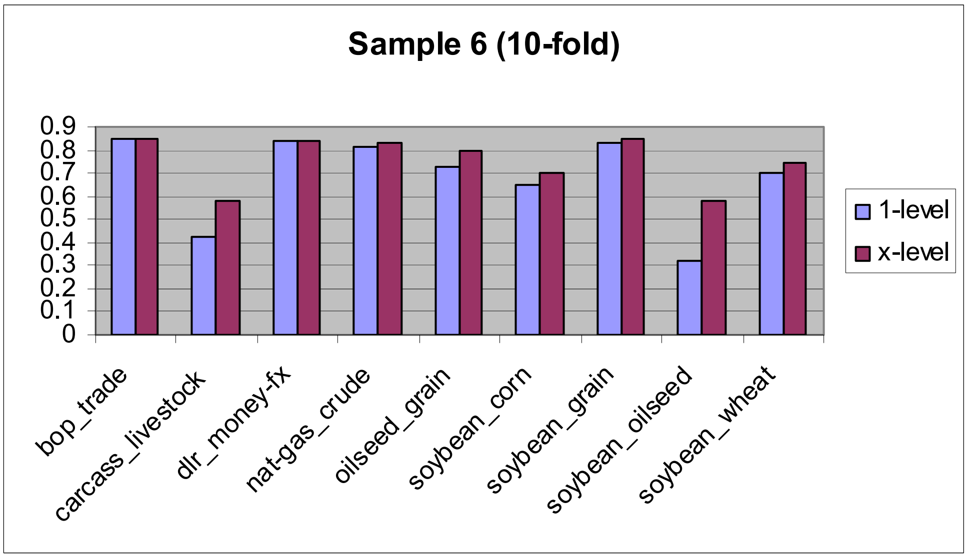


Figure 44: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 6).

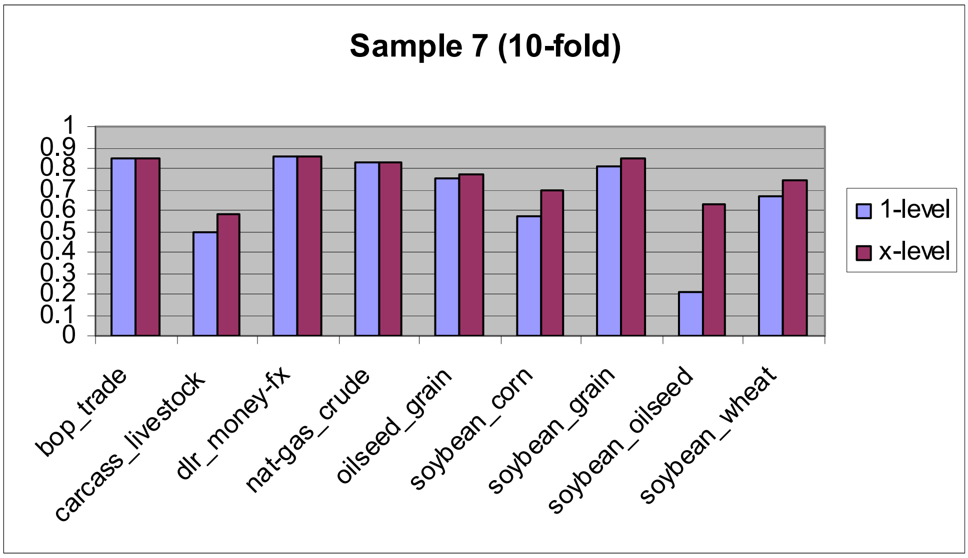


Figure 45: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 7).

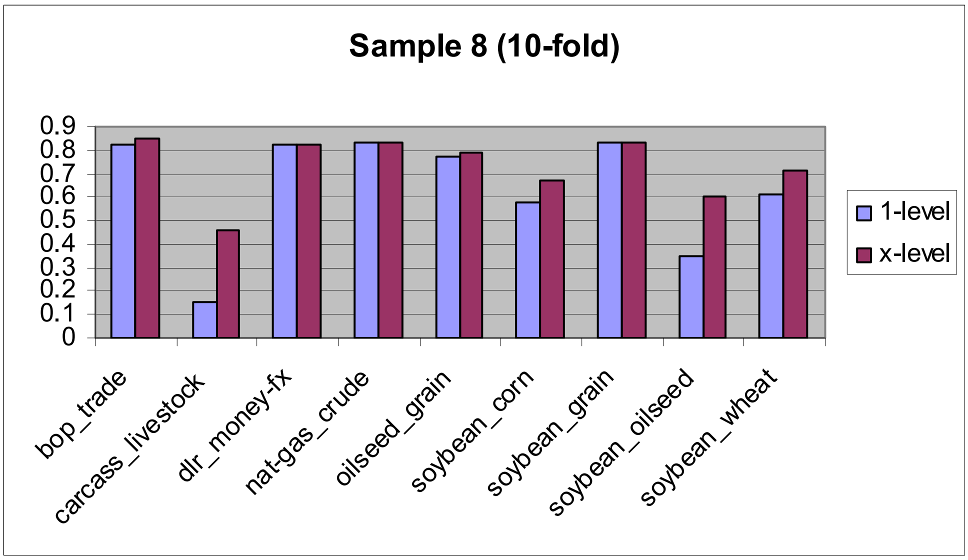


Figure 46: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 8).

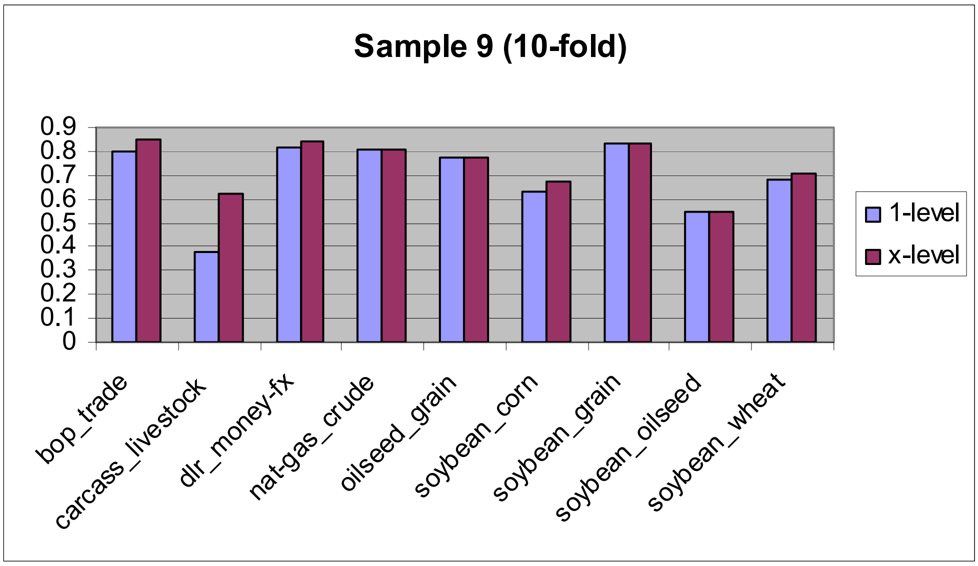


Figure 47: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 9).

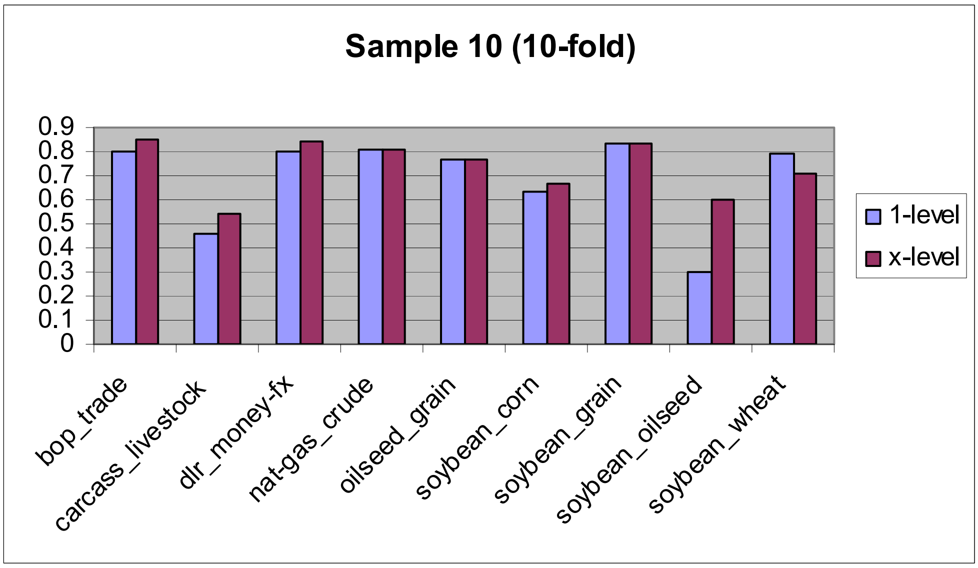


Figure 48: Classification results of 1-level classifier and multi-level classifier with 10-fold cross validation (sample 10).

For 12 samples (12-fold cross validation), the classification results of 9 topic pairs with high confidence levels and well trained classifiers (more levels of SVM classifiers) than others are shown in Figure 49 to Figure 60.

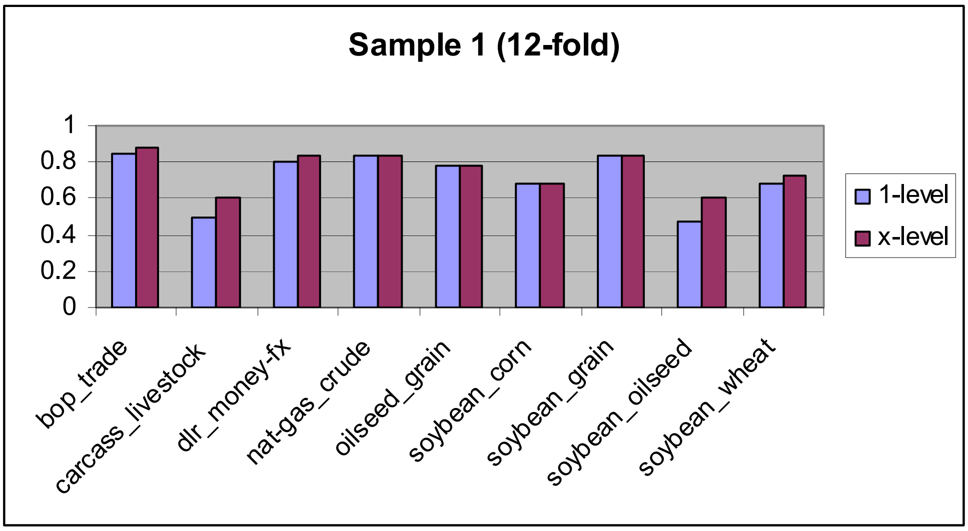


Figure 49: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 1).

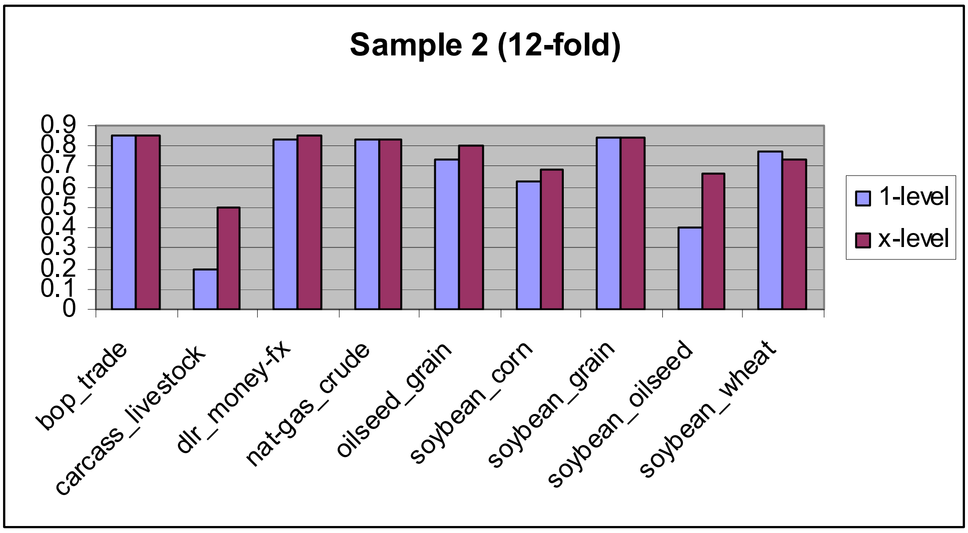


Figure 50: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 2).

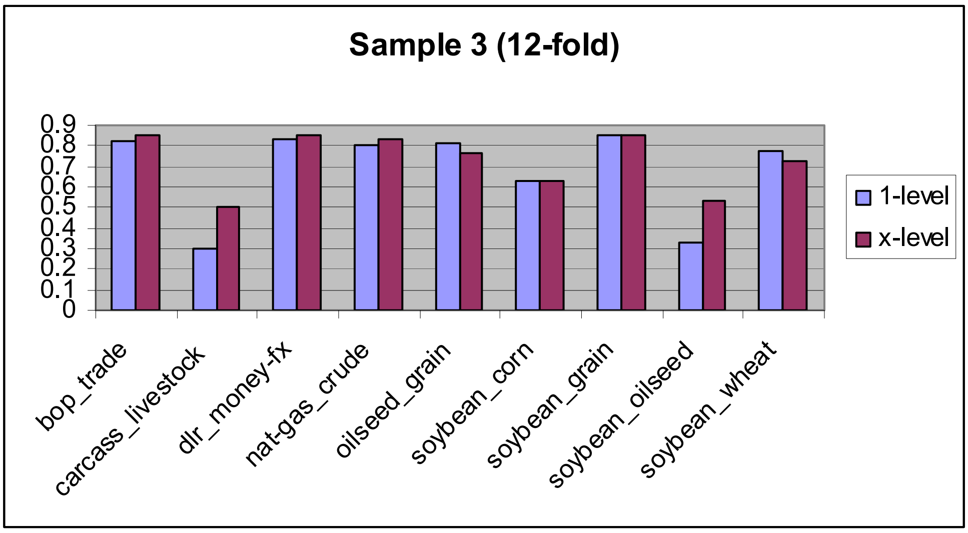


Figure 51: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 3).

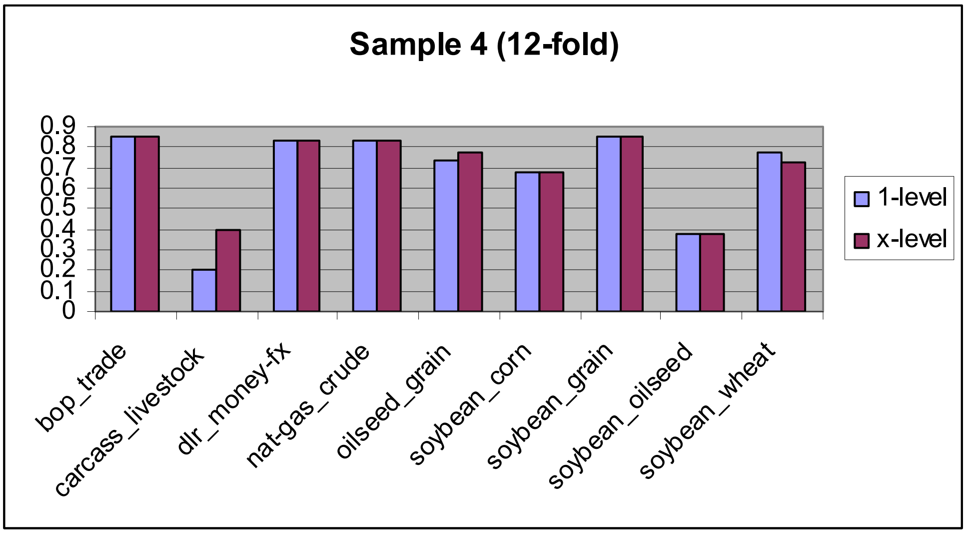


Figure 52: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 4).

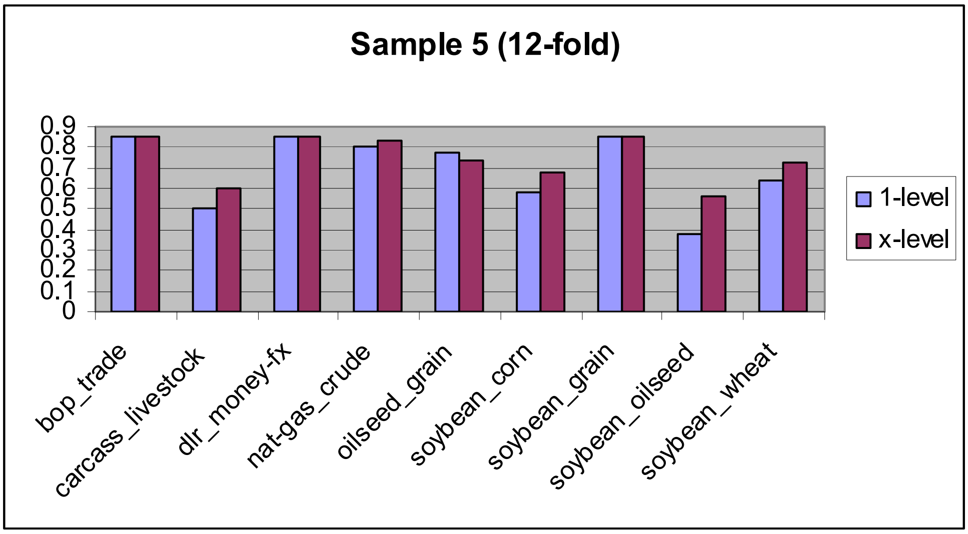


Figure 53: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 5).

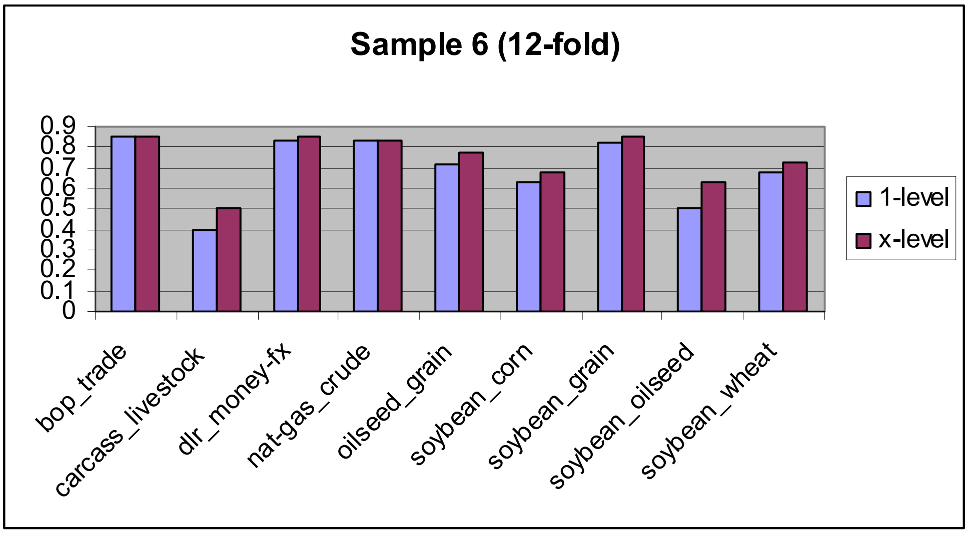


Figure 54: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 6).

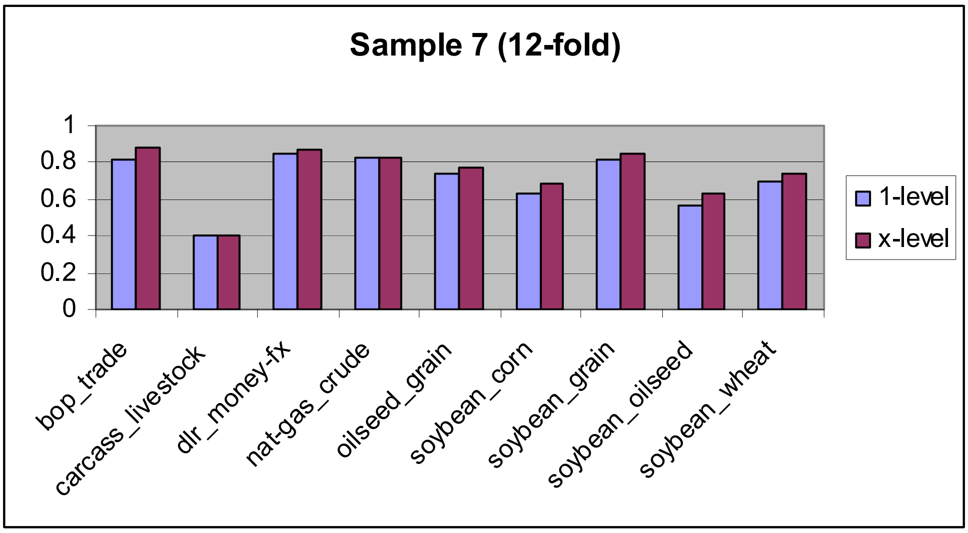


Figure 55: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 7).

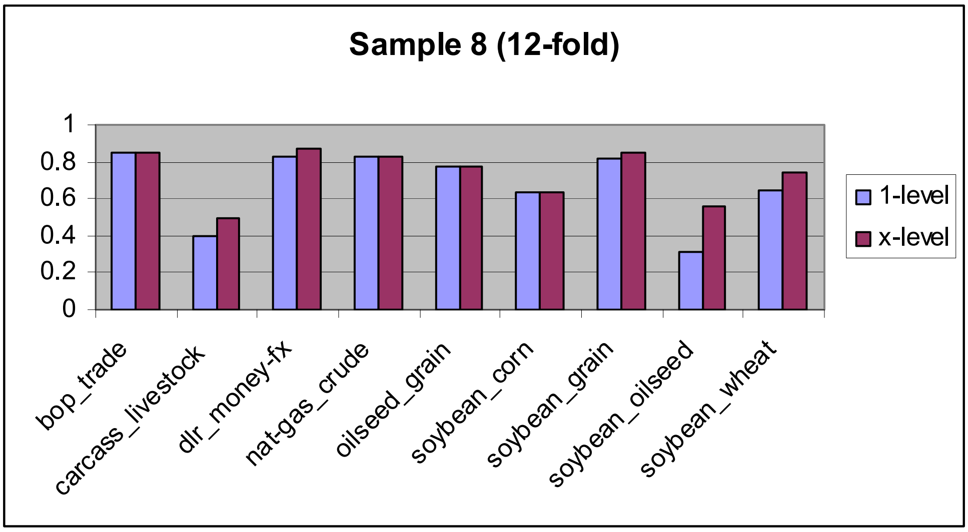


Figure 56: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 8).

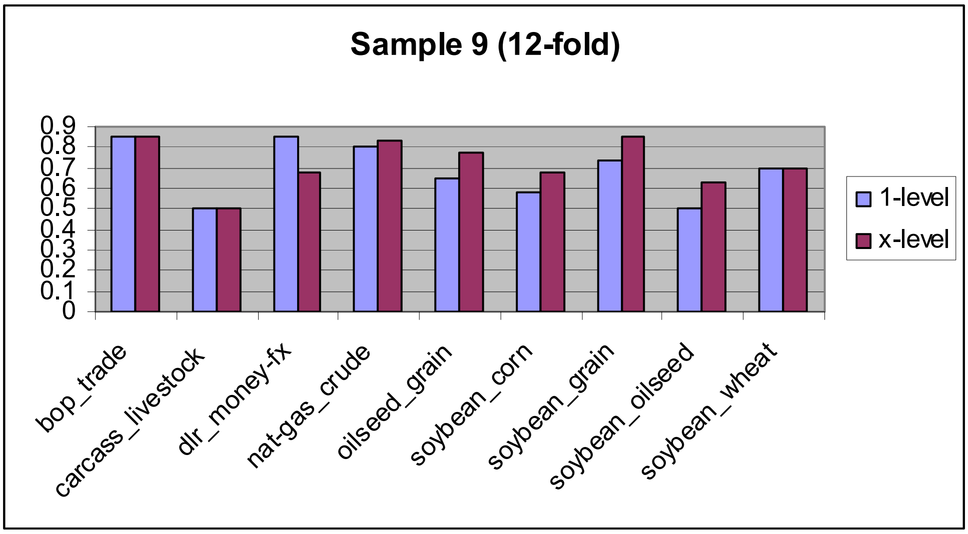


Figure 57: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 9).

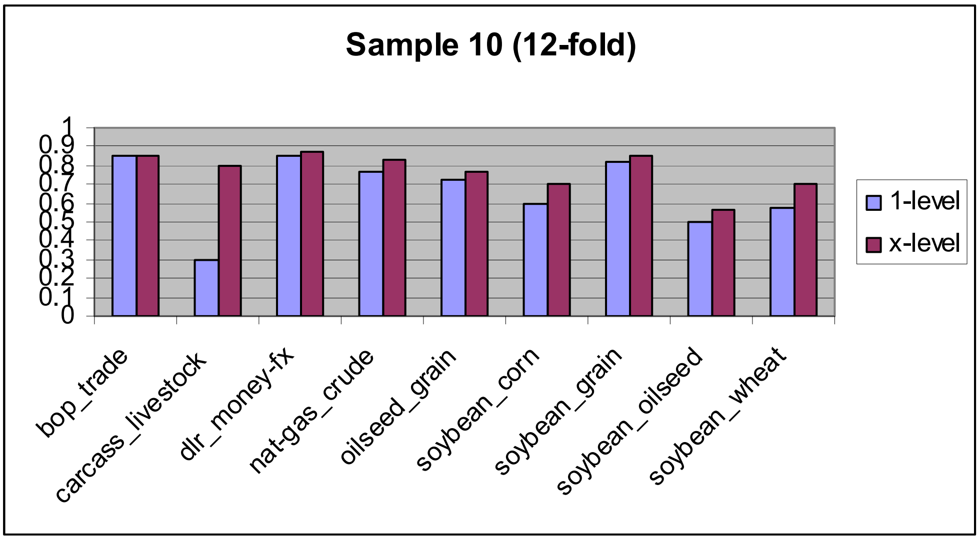


Figure 58: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 10).

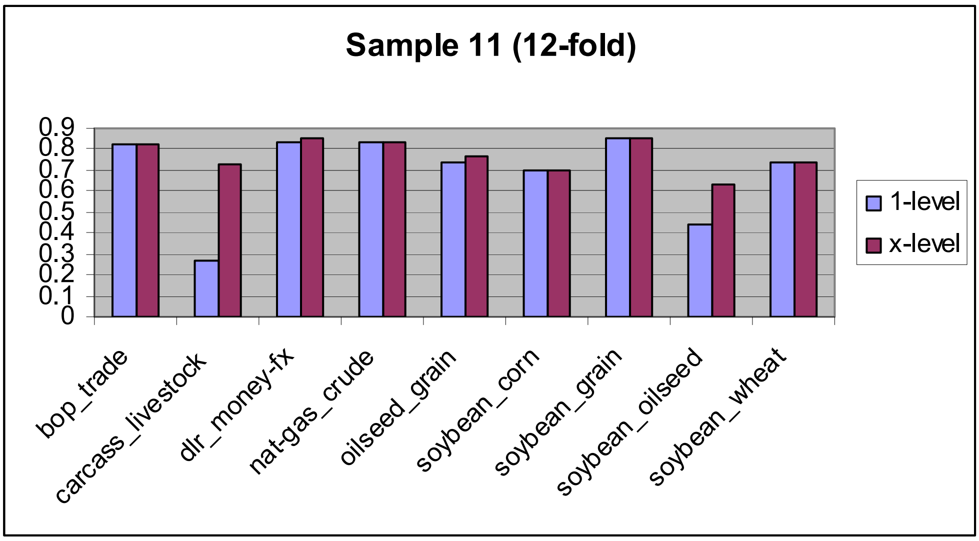


Figure 59: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 11).

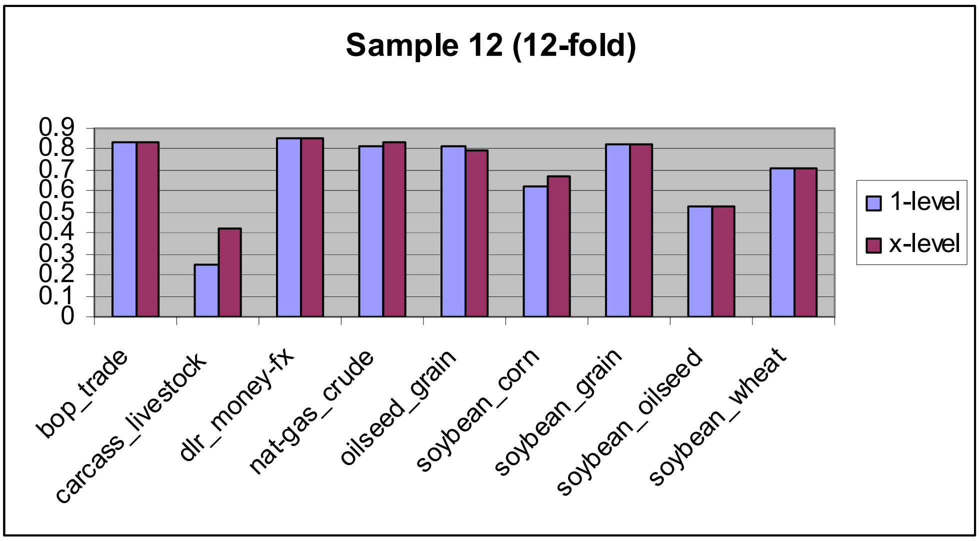


Figure 60: Classification results of 1-level classifier and multi-level classifier with 12-fold cross validation (sample 12).

5.3.7 Predication Distribution of the Last Level of the Iterative Subspace Method

In Section 5.3.2, it is found that almost the topic pairs that are well trained by our proposal scheme (iterative subspace method) can have the improvements with high confidence level. However most of the topic pairs cannot be well trained, especially at the last level. The experiment is done to observe the prediction distribution of the last level of the iterative subspace method (multi-level classifier). Six topic pairs are selected. They are:

1. sugar_trade
2. veg-oil_trade
3. carcass_veg-oil
4. dlr_trade
5. cocoa_coffee
6. cocoa_sugar

The plots are shown in Figure 61, Figure 62, Figure 63, Figure 64, Figure 65 and Figure 66 respectively.

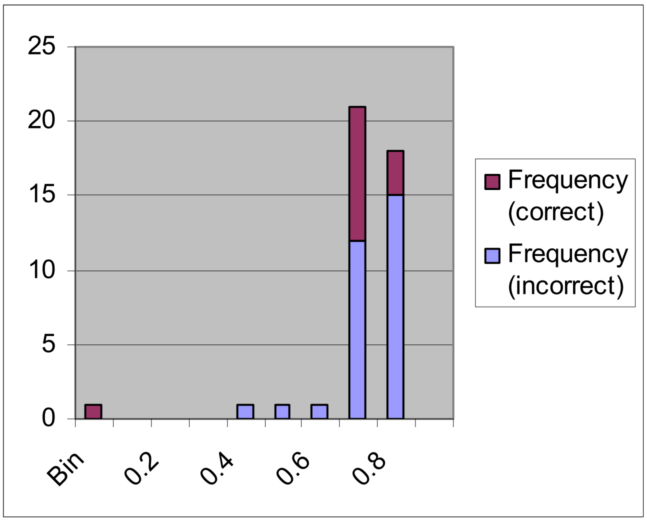


Figure 61: The prediction distribution plot of the last level of sugar_trade classifier.

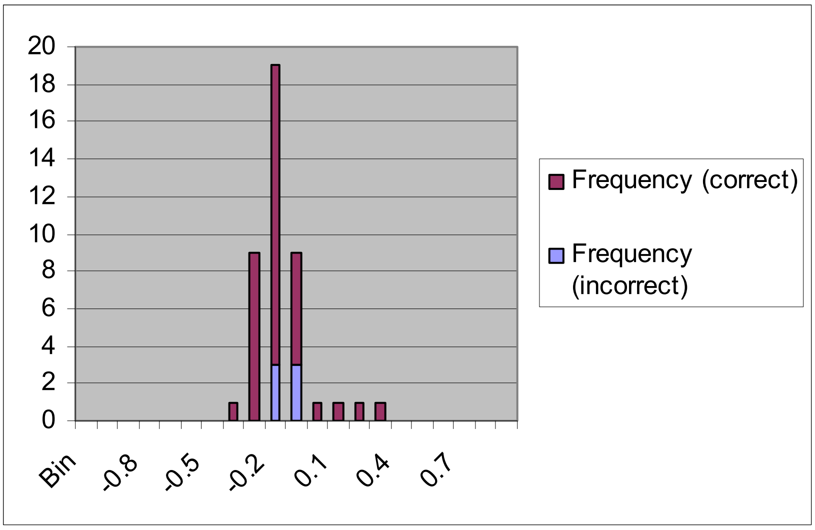


Figure 62: The prediction distribution plot of the last level of veg-oil_trade classifier.

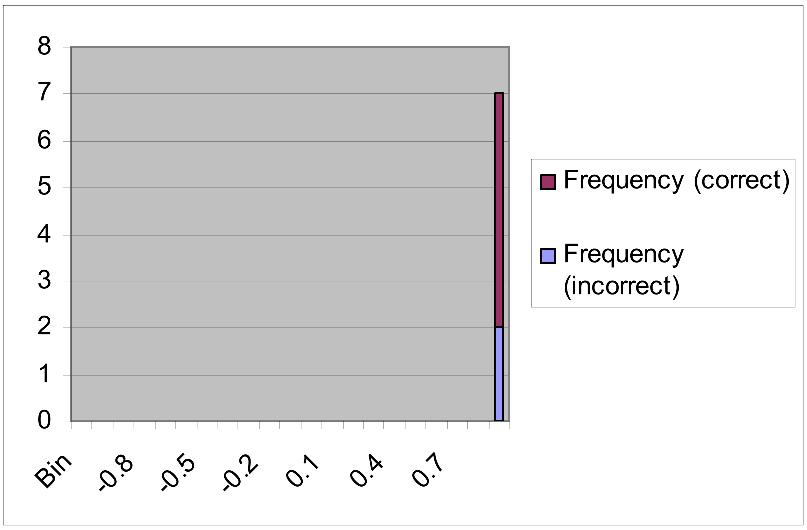


Figure 63: The prediction distribution plot of the last level of carcass_veg-oil classifier.

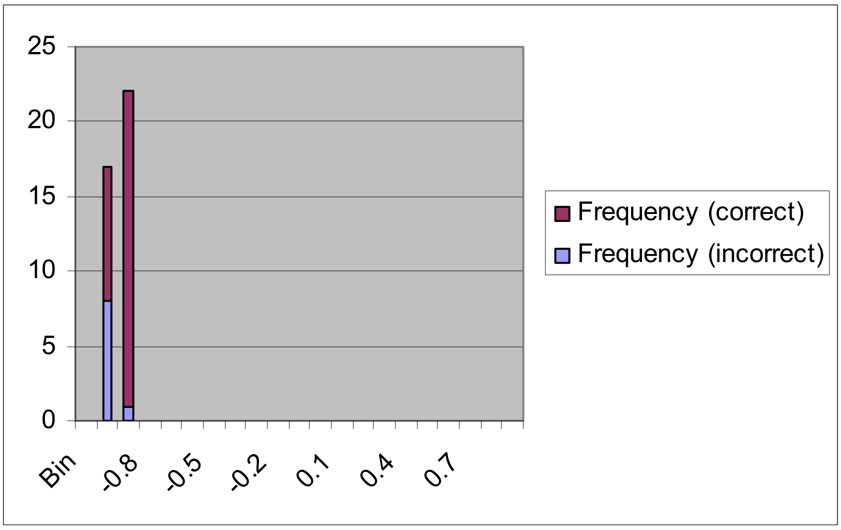


Figure 64: The prediction distribution plot of the last level of dlr_trade classifier.

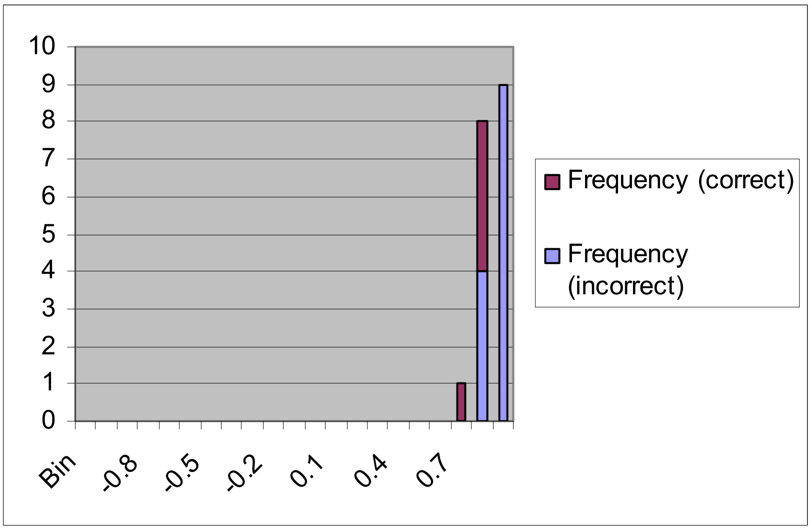


Figure 65: The prediction distribution plot of the last level of cocoa_coffee classifier.

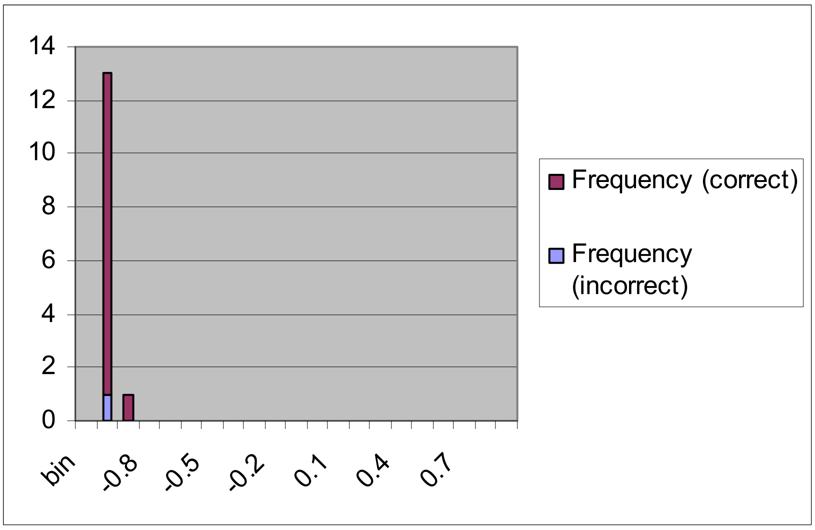


Figure 66: The prediction distribution plot of the last level of cocoa_sugar classifier.

6 Conclusion

One of the most prominent methods to combat the curse of dimensionality is subspace methodology. However, this has only been applied broadly in unsupervised text categorization. The performance of subspace methodology on supervised text categorization has not yet been found. In addition to the problem of high dimensionality, another common problem of text categorization is the uneven distribution of category size which often occurs in a large data set. This often leads to good micro-average performance but not so desirable in macro-average performance. The experiment of subtopic clustering (break large topics into sub-topics by clustering) shows significant improvement.

Due to the problem of high dimensionality and further improvement of the category boundary (subtopic clustering), the approach of iterative subspace classification is further investigated. The mathematical assumptions behind the subspace formalism demands that the pattern classes are distributed as low-dimensional subspaces in a higher-dimensional feature space. It is encouraging that subspace approach is suitable for text categorization. However the subspace classification methods have not been popular in text categorization tasks. One possibility may be that the field of data mining has captured the attention of the researchers of unsupervised text categorization.

From the view of classification, we want to re-define a difficult classification boundary possibly due to the use of the initial choice of feature subset. We want to have a better fit by decomposing the data sets into subsets using other more effective features. Subtopic clustering and proposed Iterative Subspace Method are expected to have the capability to address the issue.

The approach of iterative subspace method of pattern classification has been investigated. For the topic pairs of "carcass_livestock" and "soybean_oilseed" from the Reuters-21578 collection, the result with confidence level greater than 95% under 8-fold/10-fold/12-fold cross validation shows that this approach has good potential. Other topic pairs, such as the topic pair of "bop_trade", "dlr_money-fx", "nat-gas_crude", "oilseed_grain", "soybean_corn", "soybean_grain" and "soybean_wheat" can also achieve the improvement with high confidence level greater under some samples.

The macro-average and micro-average measures of proposed Iterative Subspace Method are not better than others. However it is still promising that there is 8.24% precision improvement of "livestock" evaluated comparing to 1-level classifier, standard Support Vector Machine (SVM), under 8-fold cross validation. There is also 11.85% improvement of "nat-gas" evaluated comparing to Soft Margin SVM classifier under 8-fold cross validation.

The performance and efficiency can be affected by different widths of separation margin. It is expected that the performance can be further improved by using other optimization techniques. The prediction distribution experiment of the last level of the iterative subspace method shows that the correct and incorrect prediction values are closely distributed. It is the main reason why they cannot be further improved.

7 References

[1] K. Aas, and L. Eikvil, *Text Categorisation: A Survey,* Technical report 941, Norwegian Computing Center, 1999.

[2] S. Abe, *Support Vector Machines for Pattern Classification*: Springer, 2005.

[3] R. Agrawal, J. Gehrke, D. Gunopulos *et al.*, “Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications,” in SIGMOD International Conference on Management of Data, 1998, pp. 94-105.

[4] L. D. Baker, and A. K. McCallum, “Distributional Clustering of Words for Text Classification,” in Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval, 1998, pp. 96-103.

[5] K. Beyer, J. Goldstein, R. Ramakrishnan *et al.*, “When is Nearest Neighbour Meaningful?,” in Proceedings of the 7th International Conference on Database Theory, Jerusalem , Israel, 10-12 January 1999, pp. 217-235.

[6] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A Training Algorithm for Optimal Margin Classifiers,” in Proceedings of the 5th Annual Workshop on Computational Learning Theory, Pittsburgh, Pennsylvania, United States, 1992, pp. 144-152.

[7] J. P. Callan, “Passage-level Evidence in Document Retrieval,” in SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Dublin, Ireland, 1994, pp. 302-310.

[8] M. F. Caroperso, S. Matwin, and F. Sebastiani, “A Learner-independent Evaluation of the Usefulness of Statistical Phrases for Automated Text Categorization,” in Text Databases and Document Management: Theory and Practice, Hershey, PA, 2001, pp. 78-102.

[9] F. Chik, R. Luk, and K. Chung, “An Evaluation of Using Clustering for Text Categorization,” in ACM-HK Postgraduate Research Day, 2005.

[10] F. C. Y. Chik, R. W. P. Luk, and K. F. L. Chung, “Text Categorization Based on Subtopic Clusters,” in Proceedings of the 10th International Conference on Natural Language Processing and Information Systems, Alicante, Spain, 2005, pp. 203-214.

[11] C. Cortes, and V. Vapnik, “Support-vector Networks,” *Machine Learning,* vol. 20, no. 3, pp. 273-297, 1995.

[12] S. C. Deerwester, S. T. Dumais, T. K. Landauer *et al.*, “Indexing by Latent Semantic Analysis,” *Journal of the Society for Information Science,* vol. 41, no. 6, pp. 391-497, 1990.

[13] S. Dumais, J. Platt, D. Heckerman *et al.*, “Inductive Learning Algorithms and Representations for Text Categorization,” in CIKM '98: Proceedings of the seventh International Conference on Information and Knowledge Management, Bethesda, Maryland, United States, 1998, pp. 148-155.

[14] D. H. Foley, and J. W. Sammon, "An Optimal Set of Discriminant Vectors," *IEEE Transactions on Computers,* vol. 24, no. 3, pp. 281-289, 1975.
[15] Y. Freund, "An Adaptive Version of the Boost by Majority Algorithm," *Machine Learning,* vol. 43, no. 3, pp. 293-318, Jun, 2001.
[16] Y. Freund, "A More Robust Boosting Algorithm," May, 2009.
[17] Y. Freund, and L. Mason, "The Alternating Decision Tree Learning Algorithm," in Proceedings of the 16th International Conference on Machine Learning, 1999, pp. 124-133.
[18] Y. Freund, and R. E. Schapire, "A Short Introduction to Boosting," *Journal of Japanese Society for Artificial Intelligence,* vol. 14, no. 5, pp. 771-780, September, 1999, 1999.
[19] J. Friedman, T. Hastie, and R. Tibshirani, "Additive Logistic Regression: A Statistical View of Boosting," *Annals of Statistics,* vol. 28, no. 2, pp. 337-374, Apr, 2000.
[20] N. Fuhr, S. Hartmann, G. Knorz *et al.*, "AIR/X - A Rule-based Multistage Indexing System for Large Subject Fields," in Proceedings of RIAO-91, 3rd International Conference on Recherche d'Information Assistee par Ordinateur, Barcelona, Spain, 1991, pp. 606-623.
[21] K. Fukunaga, and W. L. G. Koontz, "Application of the Karhunen-Loeve Expansion to Feature Selection and Ordering," *IEEE Transactions on Computers,* vol. 19, no. 4, pp. 311-318, 1970.
[22] L. Galavotti, F. Sebastiani, and M. Simi, "Experiments on the Use of Feature Selection and Negative Evidence in Automated Text Categorization," in Proceedings of ECDL-00, 4th European Conference on Research and Advanced Technology for Digital Libraries, Lisbon, Portugal, 2000, pp. 59-68.
[23] G. H. Golub, and C. F. Van Loan, *Matrix Computations*, 2nd ed., Baltimore, Md.: Johns Hopkins University Press, 1989.
[24] L. Hamel, *Knowledge Discovery with Support Vector Machines*, Hoboken, N.J.: John Wiley & Sons, 2009.
[25] V. Hatzivassiloglou, L. Gravano, and A. Maganti, "An Investigation of Linguistic Features and Clustering Algorithms for Topical Document Clustering," in SIGIR '00: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Athens, Greece, 2000, pp. 224-231.
[26] M. A. Hearst, "Support Vector Machines," *IEEE Intelligent Systems & Their Applications,* vol. 13, no. 4, pp. 18-21, Jul-Aug, 1998.
[27] W. Hersh, C. Buckley, T. Leone *et al.*, "OHSUMED: An Interactive Retrieval Evaluation and New Large Text Collection for Research," in Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval, Dublin, Ireland, 1994, pp. 192-201.
[28] H. Hotelling, "Analysis of a Complex of Statistical Variables Into Principal Components," *Journal of Educational Psychology,* vol. 24, 1933.

[29] D. A. Hull, “Improving Text Retrieval for the Routing Problem using Latent Semantic Indexing,” in Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval, Dublin, Ireland, 1994, pp. 282-289.
[30] Institute of Phonetic Sciences. "IFA Services Statistics, Signed Rank Test," http://www.fon.hum.uva.nl/Service/Statistics/Signed_Rank_Test.html.
[31] JBoost. "Software Package for Boosting Algorithms: A Java implementation including AdaBoost, LogitBoost, RobustBoost, and BoosTexter," http://jboost.sourceforge.net/.
[32] T. Joachims, “Text Categorization with Suport Vector Machines: Learning with Many Relevant Features,” in ECML '98: Proceedings of the 10th European Conference on Machine Learning, London, UK, 1998, pp. 137-142.
[33] T. Joachims, "Making Large-scale Support Vector Machine Learning Practical," *Advances in Kernel Methods: Support Vector Learning*, pp. 169-184: MIT Press, 1999.
[34] T. Joachims, “Transductive Inference for Text Classification using Support Vector Machines,” in Proceedings of the 16th International Conference on Machine Learning, 1999, pp. 200-209.
[35] T. Joachims, “Estimating the Generalization Performance of an SVM Efficiently,” in Proceedings of the 17th International Conference on Machine Learning, 2000, pp. 431-438.
[36] T. Joachims, *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*: Kluwer Academic Publishers, 2002.
[37] T. Joachims, “Optimizing Search Engines using Clickthrough Data,” in Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, 2002, pp. 133-142.
[38] T. Joachims. "SVM-Light Support Vector Machine (Version: 6.02)," http://svmlight.joachims.org/.
[39] G. H. John, R. Kohavi, and K. Pfleger, “Irrelevant Features and the Subset Selection Problem,” in Proceedings of ICML-94, 11th International Conference on Machine Learning New Brunswick, NJ, 1994, pp. 121-129.
[40] J. Karhunen, and E. Oja, *Some Comments on the Subspace Methods of Classification,* Report, Helsinki University of Technology, 1980.
[41] A. Kharechko, J. Shawe-Taylor, R. Herbrich *et al.*, “Text Categorization via Ellipsoid Separation,” in Learning Methods for Text Understanding and Mining, Grenoble, France, 26 - 29 January 2004.
[42] J. Kittler, *The Subspace Approach to Pattern Recognition,* Progress in Cybernetics and Systems Research 3, University of Surrey, 1978.
[43] T. Kohonen, G. Nemeth, K.-J. Bry *et al.*, *Classification of Phonemes by Learning Subspaces,* Report TKK-F-A348, Helsinki University of Technology, 1978.

[44] H. P. Kramer, and M. V. Mathews, “A Linear Coding for Transmitting a Set of Correlated Signals,” *IRE Transactions on Information Theory,* vol. 2, no. 3, pp. 41-46, 1956.
[45] C. A. Kulikowski, and S. Watanabe, “Multiclass Subspace Methods in Pattern Recognition,” in Proceedings of the National Electronics Conference, Chicago, 1970.
[46] S. Kullback, and R. A. Leibler, “On Information and Sufficiency,” *The Annals of Mathematical Statistics,* vol. 22, no. 1, pp. 79-86, 1951.
[47] K. Lang, “NewsWeeder: Learning to Filter Netnews,” in Proceedings of ICML-95, 12th International Conference on Machine Learning, Lake Tahoe, CA, 1995, pp. 331-339.
[48] L. S. Larkey, “Automatic Essay Grading using Text Categorization Techniques,” in Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval, Melbourne, Australia, 1998, pp. 90-95.
[49] D. D. Lewis. "Reuters-21578 Text Categorization Test Collection Distribution 1.0," http://www.daviddlewis.com/resources/testcollections/reuters21578/.
[50] D. D. Lewis, “An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task,” in Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval, Copenhagen, Denmark, 1992, pp. 37-50.
[51] D. D. Lewis, Y. Yang, T. G. Rose *et al.*, “RCV1: A New Benchmark Collection for Text Categorization Research,” *Journal of Machine Learning Research,* vol. 5, pp. 361-397, 2004.
[52] T. Li, and S. Ma, “IFD: Iterative Feature and Data Clustering,” in Proceedings of the 2004 SIAM International conference on Data Mining, Sheffield, United Kingdom, 2004, pp. 218-225.
[53] Y. H. Li, and A. K. Jain, “Classification of Text Documents,” *The Computer Journal,* vol. 41, no. 8, pp. 537-546, 1998.
[54] Y. Liang, *Support Vector Machines and Their Application in Chemistry and Biotechnology*, Boca Raton: CRC Press, 2011.
[55] A. K. McCallum. "Bow: A Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering," http://www.cs.cmu.edu/~mccallum/bow/.
[56] A. K. McCallum. "Rainbow," http://www.cs.cmu.edu/~mccallum/bow/rainbow/.
[57] T. M. Mitchell, *Machine Learning*, New York: McGraw-Hill, 1997.
[58] K. Morik, P. Brockhausen, and T. Joachims, “Combining Statistical Learning with a Knowledge-Based Approach - A Case Study in Intensive Care Monitoring,” in Proceedings of the 16th International Conference on Machine Learning, 1999, pp. 268-277.
[59] I. Moulinier, G. Raskinis, and J.-G. Ganascia, “Text Categorization: A Symbolic Approach,” in Proceedings of SDAIR-96, 5th Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, NV, 1996.

[60] H. T. Ng, W. B. Goh, and K. L. Low, "Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization," in Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information Retrieval, Philadelphia, PA, 1997, pp. 67-73.
[61] E. Oja, *Subspace Methods of Pattern Recognition*: Research Studies Press, 1983.
[62] L. Parsons, E. Haque, and H. Liu, "Subspace Clustering for High Dimensional Data: A Review," *SIGKDD Explorations Newsletter,* vol. 6, no. 1, pp. 90-105, 2004
[63] M. Pontil, and A. Verri, "Support Vector Machines for 3D Object Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, no. 6, pp. 637-646, 1998.
[64] C. Rudin, C. Cortes, M. Mohri *et al.*, "Margin-based Ranking Meets Boosting in the Middle," in Proceedings of the 18th Annual Conference on Learning Theory, Bertinoro, Italy, 2005, pp. 63-78.
[65] C. Rudin, R. E. Schapire, and I. Daubechies, "Analysis of Boosting Algorithms Using the Smooth Margin Function," *Annals of Statistics,* vol. 35, no. 6, pp. 2723-2768, Dec, 2007.
[66] M. E. Ruiz, and P. Srinivasan, "Hierarchical Neural Networks for Text Categorization," in Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval, Berkeley, CA, 1999, pp. 281-282.
[67] G. Salton, and C. Buckley, *Term Weighting Approaches in Automatic Text Retrieval*, Cornell University, Ithaca, NY, USA, 1987.
[68] G. Salton, and M. J. McGill, *Introduction to Modern Retrieval*: McGraw-Hill Book Company, 1983.
[69] R. E. Schapire, "A Brief Introduction to Boosting," in Proceedings of the 16th International Joint Conference on Artificial Intelligence, 1999.
[70] R. E. Schapire, Y. Freund, P. Bartlett *et al.*, "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," *Annals of Statistics,* vol. 26, no. 5, pp. 1651-1686, Oct, 1998.
[71] R. E. Schapire, and Y. Singer, "BoosTexter: A Boosting-based System for Text Categorization," *Machine Learning,* vol. 39, no. 2/3, pp. 135-168, 2000.
[72] H. Schutze, "Automatic Word Sense Discrimination," *Computational Linguistics,* vol. 24, no. 1, pp. 97-124, 1998.
[73] H. Schutze, D. A. Hull, and J. O. Pedersen, "A Comparison of Classifiers and Document Representations for the Routing Problem," in Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval, Seattle, WA, 1995, pp. 229-237.
[74] F. Sebastiani, A. Sperduti, and N. Valdambrini, "An Improved Boosting Algorithm and its Application to Automated Text Categorization," in Proceedings of CIKM-00, 9th ACM International Conference on Information and Knowledge Management, McLean, VA, 2000, pp. 78-85.

[75] N. Slonim, and N. Tishby, “The Power of Word Clusters for Text Classification,” in 23rd European Colloquium on Information Retrieval Research (ECIR), 2001.
[76] SPSS Inc. "Statistical Package for the Social Sciences," http://www.spss.com/.
[77] H. Takamura, and Y. Matsumoto, “Two-dimensional Clustering for Text Categorization,” in COLING-02: Proceedings of the 6th Conference on Natural Language Learning, Morristown, NJ, USA, 2002, pp. 1-7.
[78] C. W. Therrien, “Eigenvalue Properties of Projection Operators and Their Application to the Subspace Method of Feature Extraction,” *IEEE Transactions on Computers,* vol. 24, no. 9, pp. 944-948, 1975.
[79] C. J. Van Rijsbergen, *Information Retrieval*, 2nd ed., London: Butterworths, 1979.
[80] V. Vapnik, *Statistical Learning Theory*, New York: John Wiley, 1998.
[81] V. N. Vapnik, *The Nature of Statistical Learning Theory*, New York: Springer, 1995.
[82] S. Watanabe, P. F. Labert, C. A. Kulikowski *et al.*, "Evaluation and Selection of Variables in Pattern Recognition," *Computer and Information Sciences II*, J. T. Tou, ed., pp. 91-122, New York: Academic Press, 1967.
[83] S. Watanabe, and N. Pakvasa, “Subspace Method in Pattern Recognition,” in Proceedings of the 1st International Joint Conference on Pattern Recognition, 1973, pp. 25–32.
[84] A. S. Weigend, E. D. Wiener, and J. O. Pedersen, “Exploiting Hierarchy in Text Categorization,” *Information Retrieval,* vol. 1, no. 3, 1999.
[85] E. D. Wiener, J. O. Pedersen, and A. S. Weigend, “A Neural Network Approach to Topic Spotting,” in Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, NV, 1995, pp. 317-332.
[86] F. Wilcoxon, “Individual Comparisons by Ranking Methods,” *Biometrics Bulletin,* vol. 1, no. 6, pp. 80-83, 1945.
[87] Y. Yang, “Noise Reduction in a Statistical Approach to Text Categorization,” in Proceedings of SIGIR-95, 18th ACM International Conference on Resarch and Development in Information Retrieval, Seattle, WA, 1995, pp. 256-263.
[88] Y. Yang, “An Evaluation of Statistical Approaches to Text Categorization,” *Information Retrieval,* vol. 1, no. 1-2, pp. 69-90, 1999.
[89] Y. Yang, and X. Liu, “A Re-examination of Text Categorization Methods,” in SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, California, United States, 1999, pp. 42-49.
[90] Y. Yang, and J. O. Pedersen, “A Comparative Study on Feature Selection in Text Categorization,” in Proceedings of the 14th International Conference on Machine Learning, 1997, pp. 412-420.

[91] M. L. Yiu, and N. Mamoulis, “Iterative Projected Clustering by Subspace Mining,” *IEEE Transactions on Knowledge and Data Engineering (TKDE),* vol. 17, no. 2, pp. 176-189, February 2005.

[92] M. L. Yiu, and N. Mamoulis, “Frequent-pattern Based Iterative Projected Clustering,” in Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM), Melbourne, Florida, USA, November 2003, pp. 689-692.

[93] C. Zhang, and Y. Ma, *Ensemble Machine Learning: Methods and Applications*: New York: Springer, 2012.

[94] X. Zhu, and A. B. Goldberg, *Introduction to Semi-supervised Learning*, [San Rafael, Calif.]: Morgan & Claypool, 2009.

8 Appendix

Experimental result evaluated by SVM Soft Margin Classifier with c=0.1 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35703	3199	9	99.97	91.78	95.70
bop	186	160	1302	12.50	53.76	20.28
carcass	49	16	1151	4.08	75.38	7.75
cocoa	34	1	1166	2.83	97.14	5.51
coffee	1320	894	1320	50.00	59.62	54.39
corn	2386	1240	1430	62.53	65.80	64.12
cpi	120	100	1320	8.33	54.55	14.46
crude	7546	2056	830	90.09	78.59	83.95
dlr	1056	789	1248	45.83	57.24	50.90
earn	64973	2372	43	99.93	96.48	98.18
gnp	828	613	1380	37.50	57.46	45.38
gold	942	698	1314	41.76	57.44	48.36
grain	9020	2354	436	95.39	79.30	86.61
interest	5808	1775	1128	83.74	76.59	80.01
livestock	387	287	1365	22.09	57.42	31.90
money-fx	10790	2445	250	97.74	81.53	88.90
money-supply	711	508	1377	34.05	58.33	43.00
nat-gas	306	229	1422	17.71	57.20	27.04
oilseed	1532	1006	1276	54.56	60.36	57.31
ship	3063	1399	1521	66.82	68.65	67.72
soybean	435	331	1317	24.83	56.79	34.55
sugar	1651	1111	1181	58.30	59.78	59.03
trade	7114	1963	974	87.96	78.37	82.89
veg-oil	619	441	1445	29.99	58.40	39.63
wheat	3383	1587	1369	71.19	68.07	69.59

	Recall (%)	Precision (%)	F1 (%)
Macro-average	51.99	68.24	59.02
Micro-average	85.30	85.30	85.30

Experimental result evaluated by SVM Soft Margin Classifier with c=0.2 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35693	2328	19	99.95	93.88	96.82
bop	236	141	1252	15.86	62.60	25.31
carcass	49	16	1151	4.08	75.38	7.75
cocoa	76	1	1124	6.33	98.70	11.90
coffee	1460	753	1180	55.30	65.97	60.17
corn	2752	1156	1064	72.12	70.42	71.26
cpi	170	84	1270	11.81	66.93	20.07
crude	8058	1754	318	96.20	82.12	88.61
dlr	1206	589	1098	52.34	67.19	58.84
earn	64992	1489	24	99.96	97.76	98.85
gnp	950	547	1258	43.03	63.46	51.28
gold	1149	482	1107	50.93	70.45	59.12
grain	9266	1878	190	97.99	83.15	89.96
interest	6331	1330	605	91.28	82.64	86.74
livestock	508	220	1244	29.00	69.78	40.97
money-fx	10852	1862	188	98.30	85.35	91.37
money-supply	1271	316	817	60.87	80.09	69.17
nat-gas	375	229	1353	21.70	62.09	32.16
oilseed	1687	860	1121	60.08	66.23	63.01
ship	3656	1265	928	79.76	74.29	76.93
soybean	507	301	1245	28.94	62.75	39.61
sugar	1702	870	1130	60.10	66.17	62.99
trade	7821	1735	267	96.70	81.84	88.65
veg-oil	770	439	1294	37.31	63.69	47.05
wheat	3971	1383	781	83.56	74.17	78.59

	Recall (%)	Precision (%)	F1 (%)
Macro-average	58.1396	74.68	65.38
Micro-average	88.254	88.25	88.25

Experimental result evaluated by SVM Soft Margin Classifier with c=0.3 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35691	1777	21	99.94	95.26	97.54
bop	464	92	1024	31.18	83.45	45.40
carcass	94	19	1106	7.83	83.19	14.32
cocoa	288	4	912	24.00	98.63	38.61
coffee	1824	556	816	69.09	76.64	72.67
corn	3052	949	764	79.98	76.28	78.09
cpi	528	61	912	36.67	89.64	52.05
crude	8131	1427	245	97.08	85.07	90.68
dlr	1607	331	697	69.75	82.92	75.77
earn	64997	957	19	99.97	98.55	99.25
gnp	1275	410	933	57.74	75.67	65.50
gold	1474	333	782	65.34	81.57	72.56
grain	9306	1653	150	98.41	84.92	91.17
interest	6476	1004	460	93.37	86.58	89.84
livestock	629	152	1123	35.90	80.54	49.66
money-fx	10876	1500	164	98.51	87.88	92.89
money-supply	1606	182	482	76.92	89.82	82.87
nat-gas	574	179	1154	33.22	76.23	46.27
oilseed	1913	665	895	68.13	74.20	71.04
ship	4009	982	575	87.46	80.32	83.74
soybean	644	205	1108	36.76	75.85	49.52
sugar	1907	651	925	67.34	74.55	70.76
trade	7892	1438	196	97.58	84.59	90.62
veg-oil	982	351	1082	47.58	73.67	57.82
wheat	4237	1182	515	89.16	78.19	83.32

	Recall (%)	Precision (%)	F1 (%)
Macro-average	66.76	82.97	73.98
Micro-average	90.90	90.90	90.90

Experimental result evaluated by SVM Soft Margin Classifier with c=0.4 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35686	1374	26	99.93	96.29	98.08
bop	674	74	814	45.30	90.11	60.29
carcass	207	23	993	17.25	90.00	28.95
cocoa	506	6	694	42.17	98.83	59.11
coffee	2018	427	622	76.44	82.54	79.37
corn	3189	800	627	83.57	79.94	81.72
cpi	784	39	656	54.44	95.26	69.29
crude	8141	1148	235	97.19	87.64	92.17
dlr	1780	221	524	77.26	88.96	82.69
earn	64998	637	18	99.97	99.03	99.50
gnp	1555	319	653	70.43	82.98	76.19
gold	1674	243	582	74.20	87.32	80.23
grain	9317	1477	139	98.53	86.32	92.02
interest	6529	775	407	94.13	89.39	91.70
livestock	822	137	930	46.92	85.71	60.64
money-fx	10899	1254	141	98.72	89.68	93.99
money-supply	1723	111	365	82.52	93.95	87.86
nat-gas	769	147	959	44.50	83.95	58.17
oilseed	2054	540	754	73.15	79.18	76.05
ship	4134	763	450	90.18	84.42	87.21
soybean	828	168	924	47.26	83.13	60.26
sugar	2127	500	705	75.11	80.97	77.93
trade	7905	1211	183	97.74	86.72	91.90
veg-oil	1189	289	875	57.61	80.45	67.14
wheat	4326	1019	426	91.04	80.94	85.69

	Recall (%)	Precision (%)	F1 (%)
Macro-average	73.42	87.35	79.78
Micro-average	92.69	92.69	92.69

Experimental result evaluated by SVM Soft Margin Classifier with c=0.5 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35679	1151	33	99.91	96.87	98.37
bop	848	70	640	56.99	92.37	70.49
carcass	317	26	883	26.42	92.42	41.09
cocoa	598	10	602	49.83	98.36	66.15
coffee	2115	348	525	80.11	85.87	82.89
corn	3263	695	553	85.51	82.44	83.95
cpi	920	37	520	63.89	96.13	76.76
crude	8156	955	220	97.37	89.52	93.28
dlr	1859	180	445	80.69	91.17	85.61
earn	64995	464	21	99.97	99.29	99.63
gnp	1704	257	504	77.17	86.89	81.75
gold	1804	169	452	79.96	91.43	85.32
grain	9324	1328	132	98.60	87.53	92.74
interest	6556	614	380	94.52	91.44	92.95
livestock	970	139	782	55.37	87.47	67.81
money-fx	10902	1075	138	98.75	91.02	94.73
money-supply	1775	75	313	85.01	95.95	90.15
nat-gas	975	121	753	56.42	88.96	69.05
oilseed	2159	483	649	76.89	81.72	79.23
ship	4200	634	384	91.62	86.88	89.19
soybean	975	155	777	55.65	86.28	67.66
sugar	2242	433	590	79.17	83.81	81.42
trade	7914	1039	174	97.85	88.40	92.88
veg-oil	1365	238	699	66.13	85.15	74.45
wheat	4353	872	399	91.60	83.31	87.26

	Recall (%)	Precision (%)	F1 (%)
Macro-average	77.82	89.63	83.31
Micro-average	93.83	93.83	93.83

Experimental result evaluated by SVM Soft Margin Classifier with c=0.6 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35677	970	35	99.90	97.35	98.61
bop	929	74	559	62.43	92.62	74.59
carcass	430	41	770	35.83	91.30	51.47
cocoa	653	13	547	54.42	98.05	69.99
coffee	2189	299	451	82.92	87.98	85.37
corn	3316	631	500	86.90	84.01	85.43
cpi	982	36	458	68.19	96.46	79.90
crude	8170	828	206	97.54	90.80	94.05
dlr	1906	158	398	82.73	92.35	87.27
earn	64996	350	20	99.97	99.46	99.72
gnp	1778	231	430	80.53	88.50	84.33
gold	1894	126	362	83.95	93.76	88.59
grain	9326	1240	130	98.63	88.26	93.16
interest	6578	519	358	94.84	92.69	93.75
livestock	1086	137	666	61.99	88.80	73.01
money-fx	10889	966	151	98.63	91.85	95.12
money-supply	1806	67	282	86.49	96.42	91.19
nat-gas	1116	110	612	64.58	91.03	75.56
oilseed	2221	439	587	79.10	83.50	81.24
ship	4240	542	344	92.50	88.67	90.54
soybean	1058	154	694	60.39	87.29	71.39
sugar	2328	378	504	82.20	86.03	84.07
trade	7903	928	185	97.71	89.49	93.42
veg-oil	1484	223	580	71.90	86.94	78.71
wheat	4352	769	400	91.58	84.98	88.16

	Recall (%)	Precision (%)	F1 (%)
Macro-average	80.63	90.74	85.39
Micro-average	94.55	94.55	94.55

Experimental result evaluated by SVM Soft Margin Classifier with c=0.7 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35671	855	41	99.89	97.66	98.76
bop	994	77	494	66.80	92.81	77.69
carcass	509	46	691	42.42	91.71	58.01
cocoa	692	17	508	57.67	97.60	72.50
coffee	2240	273	400	84.85	89.14	86.94
corn	3336	592	480	87.42	84.93	86.16
cpi	1027	43	413	71.32	95.98	81.83
crude	8174	739	202	97.59	91.71	94.56
dlr	1935	150	369	83.98	92.81	88.18
earn	64999	284	17	99.97	99.57	99.77
gnp	1820	211	388	82.43	89.61	85.87
gold	1951	106	305	86.48	94.85	90.47
grain	9319	1178	137	98.55	88.78	93.41
interest	6588	464	348	94.98	93.42	94.20
livestock	1172	135	580	66.90	89.67	76.63
money-fx	10874	881	166	98.50	92.51	95.41
money-supply	1826	63	262	87.45	96.66	91.83
nat-gas	1217	101	511	70.43	92.34	79.91
oilseed	2264	410	544	80.63	84.67	82.60
ship	4258	478	326	92.89	89.91	91.37
soybean	1128	152	624	64.38	88.13	74.41
sugar	2379	362	453	84.00	86.79	85.38
trade	7895	856	193	97.61	90.22	93.77
veg-oil	1545	206	519	74.85	88.24	81.00
wheat	4346	698	406	91.46	86.16	88.73

	Recall (%)	Precision (%)	F1 (%)
Macro-average	82.54	91.43	86.76
Micro-average	95.00	95.00	95.00

Experimental result evaluated by SVM Soft Margin Classifier with c=0.8 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35667	779	45	99.87	97.86	98.86
bop	1042	77	446	70.03	93.12	79.94
carcass	576	67	624	48.00	89.58	62.51
cocoa	729	20	471	60.75	97.33	74.81
coffee	2272	251	368	86.06	90.05	88.01
corn	3343	558	473	87.60	85.70	86.64
cpi	1062	46	378	73.75	95.85	83.36
crude	8179	679	197	97.65	92.33	94.92
dlr	1951	138	353	84.68	93.39	88.82
earn	65000	244	16	99.98	99.63	99.80
gnp	1854	201	354	83.97	90.22	86.98
gold	1983	98	273	87.90	95.29	91.45
grain	9314	1130	142	98.50	89.18	93.61
interest	6606	425	330	95.24	93.96	94.59
livestock	1224	141	528	69.86	89.67	78.54
money-fx	10867	814	173	98.43	93.03	95.66
money-supply	1836	66	252	87.93	96.53	92.03
nat-gas	1268	90	460	73.38	93.37	82.18
oilseed	2285	403	523	81.37	85.01	83.15
ship	4266	441	318	93.06	90.63	91.83
soybean	1182	153	570	67.47	88.54	76.58
sugar	2413	347	419	85.20	87.43	86.30
trade	7887	790	201	97.51	90.90	94.09
veg-oil	1586	205	478	76.84	88.55	82.28
wheat	4330	651	422	91.12	86.93	88.98

	Recall (%)	Precision (%)	F1 (%)
Macro-average	83.85	91.76	87.63
Micro-average	95.30	95.30	95.30

Experimental result evaluated by SVM Soft Margin Classifier with c=0.9 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35662	724	50	99.86	98.01	98.93
bop	1073	80	415	72.11	93.06	81.26
carcass	630	75	570	52.50	89.36	66.14
cocoa	760	25	440	63.33	96.82	76.57
coffee	2292	242	348	86.82	90.45	88.60
corn	3347	527	469	87.71	86.40	87.05
cpi	1091	50	349	75.76	95.62	84.54
crude	8177	640	199	97.62	92.74	95.12
dlr	1959	134	345	85.03	93.60	89.11
earn	64999	212	17	99.97	99.67	99.82
gnp	1874	196	334	84.87	90.53	87.61
gold	2005	96	251	88.87	95.43	92.04
grain	9301	1086	155	98.36	89.54	93.75
interest	6611	414	325	95.31	94.11	94.71
livestock	1280	141	472	73.06	90.08	80.68
money-fx	10856	775	184	98.33	93.34	95.77
money-supply	1844	64	244	88.31	96.65	92.29
nat-gas	1308	84	420	75.69	93.97	83.85
oilseed	2298	396	510	81.84	85.30	83.53
ship	4268	418	316	93.11	91.08	92.08
soybean	1205	165	547	68.78	87.96	77.19
sugar	2438	325	394	86.09	88.24	87.15
trade	7883	765	205	97.47	91.15	94.20
veg-oil	1615	203	449	78.25	88.83	83.20
wheat	4320	603	432	90.91	87.75	89.30

	Recall (%)	Precision (%)	F1 (%)
Macro-average	84.80	91.99	88.25
Micro-average	95.50	95.50	95.50

Experimental result evaluated by SVM Soft Margin Classifier with c=1 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35657	683	55	99.85	98.12	98.98
bop	1097	85	391	73.72	92.81	82.17
carcass	671	82	529	55.92	89.11	68.71
cocoa	782	29	418	65.17	96.42	77.77
coffee	2306	229	334	87.35	90.97	89.12
corn	3350	515	466	87.79	86.68	87.23
cpi	1118	52	322	77.64	95.56	85.67
crude	8172	605	204	97.56	93.11	95.28
dlr	1976	131	328	85.76	93.78	89.59
earn	64999	189	17	99.97	99.71	99.84
gnp	1892	194	316	85.69	90.70	88.12
gold	2024	92	232	89.72	95.65	92.59
grain	9290	1064	166	98.24	89.72	93.79
interest	6620	401	316	95.44	94.29	94.86
livestock	1307	146	445	74.60	89.95	81.56
money-fx	10842	730	198	98.21	93.69	95.90
money-supply	1852	65	236	88.70	96.61	92.48
nat-gas	1341	82	387	77.60	94.24	85.12
oilseed	2291	387	517	81.59	85.55	83.52
ship	4269	399	315	93.13	91.45	92.28
soybean	1224	183	528	69.86	86.99	77.49
sugar	2455	308	377	86.69	88.85	87.76
trade	7877	733	211	97.39	91.49	94.35
veg-oil	1637	203	427	79.31	88.97	83.86
wheat	4319	581	433	90.89	88.14	89.49

	Recall (%)	Precision (%)	F1 (%)
Macro-average	85.51	92.10	88.68
Micro-average	95.64	95.64	95.64

Experimental result evaluated by SVM Soft Margin Classifier with c=10 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35650	602	62	99.83	98.34	99.08
bop	1166	92	322	78.36	92.69	84.92
carcass	765	126	435	63.75	85.86	73.17
cocoa	868	42	332	72.33	95.38	82.27
coffee	2340	229	300	88.64	91.09	89.84
corn	3271	629	545	85.72	83.87	84.78
cpi	1165	77	275	80.90	93.80	86.88
crude	8169	516	207	97.53	94.06	95.76
dlr	2064	135	240	89.58	93.86	91.67
earn	65001	138	15	99.98	99.79	99.88
gnp	1920	171	288	86.96	91.82	89.32
gold	2040	91	216	90.43	95.73	93.00
grain	8905	880	551	94.17	91.01	92.56
interest	6744	326	192	97.23	95.39	96.30
livestock	1355	164	397	77.34	89.20	82.85
money-fx	10869	483	171	98.45	95.75	97.08
money-supply	1869	69	219	89.51	96.44	92.85
nat-gas	1423	100	305	82.35	93.43	87.54
oilseed	2276	462	532	81.05	83.13	82.08
ship	4268	384	316	93.11	91.75	92.42
soybean	1309	267	443	74.71	83.06	78.67
sugar	2465	316	367	87.04	88.64	87.83
trade	7882	575	206	97.45	93.20	95.28
veg-oil	1713	252	351	82.99	87.18	85.03
wheat	4317	596	435	90.85	87.87	89.33

	Recall (%)	Precision (%)	F1 (%)
Macro-average	87.21	91.69	89.40
Micro-average	95.88	95.88	95.88

Experimental result evaluated by SVM Soft Margin Classifier with c=100 under 8-fold cross validation.

Topic	*a*	*b*	*c*	Recall (%)	Precision (%)	F1 (%)
acq	35642	626	70	99.80	98.27	99.03
bop	1165	89	323	78.29	92.90	84.97
carcass	747	166	453	62.25	81.82	70.71
cocoa	862	52	338	71.83	94.31	81.55
coffee	2327	254	313	88.14	90.16	89.14
corn	3224	658	592	84.49	83.05	83.76
cpi	1165	86	275	80.90	93.13	86.58
crude	8157	533	219	97.39	93.87	95.59
dlr	2058	155	246	89.32	93.00	91.12
earn	65000	136	16	99.98	99.79	99.88
gnp	1924	174	284	87.14	91.71	89.36
gold	2042	90	214	90.51	95.78	93.07
grain	8731	912	725	92.33	90.54	91.43
interest	6736	342	200	97.12	95.17	96.13
livestock	1347	206	405	76.88	86.74	81.51
money-fx	10816	501	224	97.97	95.57	96.76
money-supply	1871	81	217	89.61	95.85	92.62
nat-gas	1436	98	292	83.10	93.61	88.04
oilseed	2250	525	558	80.13	81.08	80.60
ship	4234	419	350	92.36	91.00	91.67
soybean	1308	297	444	74.66	81.50	77.93
sugar	2386	382	446	84.25	86.20	85.21
trade	7857	578	231	97.14	93.15	95.10
veg-oil	1655	320	409	80.18	83.80	81.95
wheat	4282	634	470	90.11	87.10	88.58

	Recall (%)	Precision (%)	F1 (%)
Macro-average	86.64	90.76	88.65
Micro-average	95.57	95.57	95.57