Manipulating Large Corpora for Text Classification

Fumiyo Fukumoto and **Yoshimi Suzuki**

Department of Computer Science and Media Engineering,
Yamanashi University
4-3-11 Takeda, Kofu 400-8511 Japan
fukumoto@skye.esb.yamanashi.ac.jp   ysuzuki@alps1.esi.yamanashi.ac.jp

Abstract

In this paper, we address the problem of
dealing with a large collection of data
and propose a method for text classifi-
cation which manipulates data using two
well-known machine learning techniques,
Naive Bayes(NB) and Support Vector Ma-
chines(SVMs). NB is based on the as-
sumption of word independence in a text,
which makes the computation of it far
more efficient. SVMs, on the other hand,
have the potential to handle large feature
spaces, which makes it possible to pro-
duce better performance. The training
data for SVMs are extracted using NB
classifiers according to the category hier-
archies, which makes it possible to reduce
the amount of computation necessary for
classification without sacrificing accuracy.

1 Introduction

As the volume of online documents has drastically
increased, text classification has become more im-
portant, and a growing number of statistical and ma-
chine learning techniques have been applied to the
task(Lewis, 1992), (Yang and Wilbur, 1995), (Baker
and McCallum, 1998), (Lam and Ho, 1998), (Mc-
Callum, 1999), (Dumais and Chen, 2000). Most of
them use the Reuters-21578 articles[1] in the evalu-

---

[1]The Reuters-21578, distribution 1.0, is comprised of
21,578 documents, representing what remains of the original
Reuters-22173 corpus after the elimination of 595 duplicates
by Steve Lynch and David Lewis in 1996.

ations of their methods, since the corpus has be-
come a benchmark, and their results are thus eas-
ily compared with other results. It is generally
agreed that these methods using statistical and ma-
chine learning techniques are effective for classifi-
cation task, since most of them showed significant
improvement (the performance over 0.85 F1 score)
for Reuters-21578(Joachims, 1998), (Dumais et al.,
1998), (Yang and Liu, 1999).

More recently, some researchers have applied
their techniques to larger corpora such as web
pages in Internet applications(Mladenic and Grobel-
nik, 1998), (McCallum, 1999), (Dumais and Chen,
2000). The increasing number of documents and
categories, however, often hampers the develop-
ment of practical classification systems, mainly due
to statistical, computational, and representational
problems(Dietterich, 2000). There are at least two
strategies for solving these problems. One is to
use category hierarchies. The idea behind this is
that when humans organize extensive data sets into
fine-grained categories, category hierarchies are of-
ten employed to make the large collection of cate-
gories more manageable. McCallum et. al. pre-
sented a method called 'shrinkage' to improve pa-
rameter estimates by taking advantage of a hierar-
chy(McCallum, 1999). They tested their method us-
ing three different real-world datasets: 20,000 ar-
ticles from UseNet, 6,440 web pages from the in-
dustry sector, and 14,831 pages from Yahoo, and
showed improved performance. Dumais et. al.
used SVMs and classified hierarchical web content
consisting of 50,078 web pages for training, and
10,024 for testing, with promising results(Dumais

and Chen, 2000).

The other is to use *ensemble* methods which are learning algorithms that construct a set of classifiers and then classify new data by taking a (weighted) vote of their predictions(Dietterich, 2000). One of the methods for constructing ensembles manipulates the training examples to generate multiple hypotheses. The most straightforward way is called *Bagging*. It presents the learning algorithm with a training set that consists of a sample of m examples drawn randomly with replacement from the original training set. The second method is to construct the training sets by leaving out disjoint subsets of the training data. The third is illustrated by the AD-ABOOST algorithm(Freund and Schapire, 1996). Dietterich has compared these methods(Dietterich, 2000). He reported that in low-noise data, AD-ABOOST performs well, while in high-noise cases, it yields overfitting because ADABOOST puts a large amount of weight on the mislabeled examples. Bagging works well on both the noisy and the noise-free data because it focuses on the statistical problem which arises when the amount of training data available is too small, and noise increases this statistical problem. However, it is not clear whether 'works well' means that it exponentially reduces the amount of computation necessary for classification, while sacrificing only a small amount of accuracy, or whether it is statistically significantly better than other methods.

In this paper, we address the problem of dealing with a large collection of data and report on an empirical study for text classification which manipulates data using two well-known machine learning techniques, Naive Bayes(NB) and Support Vector Machines(SVMs). NB probabilistic classifiers are based on the assumption of word independence in a text which makes the computation of the NB classifiers far more efficient. SVMs, on the other hand, have the potential to handle large feature spaces, since SVMs use overfitting protection which does not necessarily depend on the number of features, and thus makes it possible to produce better performance. The basic idea of our approach is quite simple: We solve simple classification problems using NB and more complex and difficult problems using SVMs. As in previous research, we use category hierarchies. We use all the training data for NB.

The training data for SVMs, on the other hand, is extracted using NB classifiers. The training data is learned by NB using cross-validation according to the hierarchical structure of categories, and only the documents which could not classify correctly by NB classifiers in each category level are extracted as the training data of SVMs.

The rest of the paper is organized as follows. The next section provides the basic framework of NB and SVMs. We then describe our classification method. Finally, we report some experiments using 279,303 documents in the Reuters 1996 corpus with a discussion of evaluation.

2 Classifiers

2.1 NB

Naive Bayes(NB) probabilistic classifiers are commonly studied in machine learning(Mitchell, 1996). The basic idea in NB approaches is to use the joint probabilities of words and categories to estimate the probabilities of categories given a document. The NB assumption is that all the words in a text are conditionally independent given the value of a classification variable. There are several versions of the NB classifiers. Recent studies on a Naive Bayes classifier which is proposed by McCallum et. al. reported high performance over some other commonly used versions of NB on several data collections(McCallum et al., 1998). We use the model of NB by McCallum et. al. which is shown in formula (1).

$$P(c_j \mid d_i, \hat{\theta}) = \frac{P(c_j \mid \hat{\theta})\Pi_{k=1}^{|d_i|}P(w_{d_{ik}} \mid c_j, \hat{\theta})}{\sum_{r=1}^{|C|}P(c_r \mid \hat{\theta})\Pi_{k=1}^{|d_i|}P(w_{d_{ik}} \mid c_r, \hat{\theta})}$$

where

$$P(w_t \mid c_j, \hat{\theta}) = \frac{1 + \sum_{i=1}^{|D|}N(w_t, d_i)P(c_j \mid d_i)}{\mid V \mid + \sum_{s=1}^{|V|}\sum_{i=1}^{|D|}N(w_s, d_i)P(c_j \mid d_i)}$$

$$P(c_j \mid \hat{\theta}) = \sum_{i=1}^{|D|}P(c_j \mid d_i)/ \mid D \mid \qquad (1)$$

$|V|$ refers to the number of vocabularies, $|D|$ denotes the number of labeled training documents, and $|C|$ shows the number of categories. $|d_i|$ denotes document length. $w_{d_{ik}}$ is the word in position k of

document d_i, where the subscript of w, d_{ik} indicates an index into the vocabulary. $N(w_t, d_i)$ denotes the number of times word w_t occurs in document d_i, and $P(c_j \mid d_i)$ is defined by $P(c_j \mid d_i) \in \{0,1\}$.

2.2 SVMs

SVMs are introduced by Vapnik(Vapnik, 1995) for solving two-class pattern recognition problems. It is defined over a vector space where the problem is to find a decision surface(classifier) that 'best' separates a set of positive examples from a set of negative examples by introducing the maximum 'margin' between two sets. The margin is defined by the distance from the hyperplane to the nearest of the positive and negative examples. The decision surface produced by SVMs for linearly separable space is a hyperplane which can be written as $\mathbf{w} \cdot \mathbf{x} + b = 0$ ($\mathbf{x}$, $\mathbf{w} \in \mathbf{R}^n$, $b \in \mathbf{R}$), where $\mathbf{x}$ is an arbitrary data point, and $\mathbf{w} = (w_1, \cdots, w_n)$ and b are learned from a training set of linearly separable data. Figure 1 shows an example of a simple two-dimensional problem that is linearly separable[2].
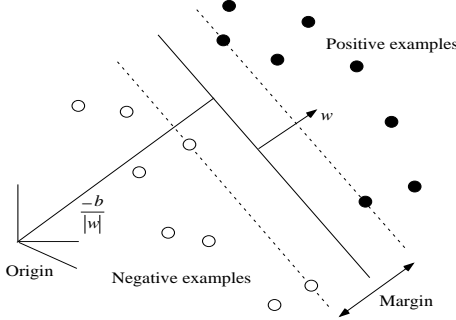

Figure 1: The decision surface of a linear SVMs

In the linearly separable case maximizing the margin can be expressed as an optimization problem:

$$Minimize: \quad -\sum_{i=1}^{n} \alpha_i + \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_i \quad (2)$$

$$s.t: \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \quad \forall i: \ \alpha_i \geq 0$$

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \quad (3)$$

where $\mathbf{x}_i = (x_{i1}, \cdots, x_{in})$ is the i-th training example and y_i is a label corresponding the i-th training example. In formula (3), each element of $\mathbf{w}$, w_k ($1 \leq$

---

[2]We focused on linear hypotheses in this work, while SVMs can handle nonlinear hypotheses using $Kernel$ functions.

$k \leq n$) corresponds to each word in the training examples, and the larger value of $w_k = \sum_i^l \alpha_i y_i x_{ik}$ is, the more the word w_k features positive examples.

We note that SVMs are basically introduced for solving binary classification, while text classification is a multi-class, multi-label classification problem. Several methods using SVMs which were intended for multi-class, multi-label data have been proposed(Weston and Watkins, 1998). We use One-$against$-the-$Rest$ version of the SVMs model in the work. A time complexity of SVMs is known as $O(l^2) \sim O(l^3)$, where l is the number of training data. We consider a time complexity of One-$against$-the-$Rest$ method. Let l be the number of training data with k categories. The average size of the training data per category is $\frac{l}{k}$. Let also $k' \cdot T(l')$ be the time needed to train all categories, where $T(l')$ represents the time for learning one binary classifier using l' training data, and k' is the number of binary classifier. The time for learning one binary classifier, $T(l')$ is represented as $T(l') = C \cdot l'^3$, where C is a constant. One-$against$-the-$Rest$ method is thus done in time Ckl^3.

3 System Design

3.1 Hierarchical classification

A well-known technique for classifying a large, heterogeneous collection such as web content is to use category hierarchies. Following the approaches of Koller and Sahami(Koller and Sahami, 1997), and Dumais's(Dumais and Chen, 2000), we employ a hierarchy by learning separate classifiers at each internal node of the tree, and then labeling a document using these classifiers to greedily select subbranches until a leaf is reached.

3.2 Manipulating training data

Our hypothesis regarding NB is that it can work well for documents which are assigned to only one category within the same category level in the hierarchical structure. We base this on some recent papers claiming that NB methods perform surprisingly well for an 'accuracy' measure which is equivalent to the standard precision under the one-category-per-document assumption on classifiers and also equivalent to the standard recall, assuming that each document has one and only one correct category per cat-

egory level(Lewis and Ringuette, 1994), (Koller and Sahami, 1997). SVMs, on the other hand, have the potential to handle more complex problems without sacrificing accuracy, even though the computation of the SVM classifiers is far less efficient than NB. We thus use NB for simple classification problems and SVMs for more complex data, i.e., the data which cannot be classified correctly by NB classifiers. We use ten-fold cross validation: All of the training data were randomly shuffled and divided into ten equal folds. Nine folds were used to train the NB classifiers while the remaining fold(held-out test data) was used to evaluate the accuracy of the classification. For each category level, we apply the following procedures. Let N_a be the total number of nine folds training documents, and N_b be the number of the remaining fold in each class level. Figure 2 illustrates the flow of our system.
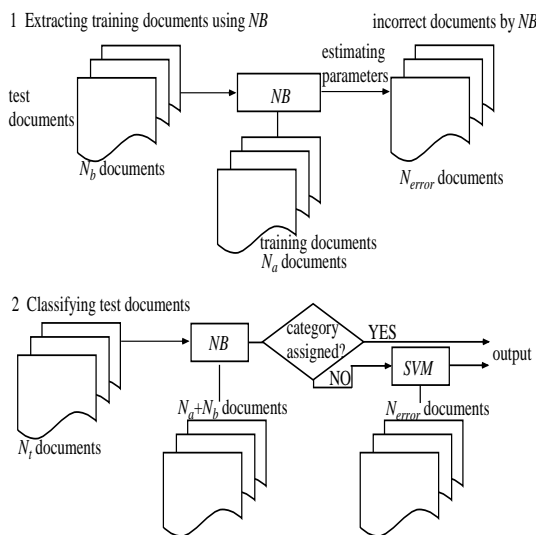

Figure 2: Flow of our system

1. Extracting training data using NB

1-1 NB is applied to the N_a documents, and classifiers for each category are induced. They are evaluated using the held-out test data, the N_b documents.

1-2 This process is repeated ten times so that each fold serves as the source of the test data once. The threshold, the probability value which produces the most accurate classifier through ten runs, is selected.

1-3 The held-out test data which could not be classified correctly by NB classifiers with the optimal parameters are extracted (N_{error} in Figure 2). They are used to train SVMs.

The procedure is applied to each category level.

2. Classifying test data

2-1 We use all the training data, N_a+N_b, to train NB classifiers and the data which is produced by procedure 1-3 to train SVMs.

2-2 NB classifiers are applied to the test data. The test data is judged to be the category c whose probability is larger than the threshold which is obtained by 1-2.

2-3 If the test data is not assigned to any one of the categories, the test data is classified by SVMs classifiers. The test data is judged to be the category c whose distance $\frac{1}{\|\vec{w}\|}$ is larger than zero.

We employ the hierarchy by learning separate classifiers at each internal node of the tree and then assign categories to a document by using these classifiers to greedily select sub-branches until a leaf is reached.

4 Evaluation

4.1 Data and Evaluation Methodology

We evaluated the method using the 1996 Reuters corpus recently made available. The corpus from 20th Aug. to 31st Dec. consists of 279,303 documents. These documents are organized into 126 categories with a four level hierarchy. We selected 102 categories which have at least one document in the training set and the test set. The number of categories in each level is 25 top level, 33 second level, 43 third level, and 1 fourth level, respectively. Table 1 shows the number of documents in each top level category.

After eliminating unlabelled documents, we obtained 271,171 documents. We divided these documents into two sets: a training set from 20th Aug. to 31th Oct. which consists of 150,939 documents, and test set from 1th Nov. to 31st Dec. which consists of 120,242 documents. We obtained a vocabulary of 183,400 unique words after eliminating words which occur only once, stemming by a part-of-speech tagger(Schmid, 1995), and stop word removal. Figure 3 illustrates the category distribution

| Category name	Training	Test
Corporate/Industrial	69,975	56,100
Economics	22,214	18,694
Government/social	45,618	36,923
Crime	6,248	4,865
Defence	1,646	1,408
International relations	7,523	6,278
Disasters	1,644	1,383
Arts	771	602
Environment	1,170	876
Fashion	71	14
Health	1,232	961
Labour issues	3,314	2,827
Obituaries	123	124
Human interest	479	418
Domestic politics	11,528	9,022
Biographies	1,115	1,041
Religion	618	418
Science and technology	359	410
Sports	5,807	4,998
Travel and tourism	149	142
War	7,064	5,228
Elections	3,070	1,944
Weather	784	474
Welfare	359	260
Markets	34,901	28,484
Total	227,782	183,894

Table 1: Top level categories
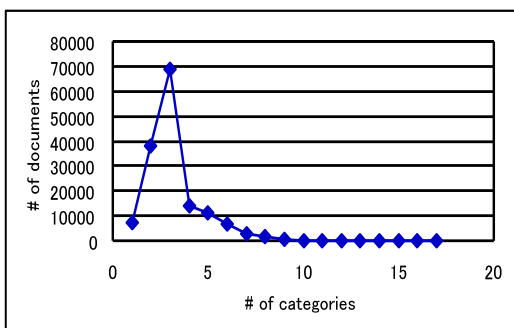
Figure 3: Category distribution in Reuters 1996

We use ten-fold cross validation for learning NB parameters. For evaluating the effectiveness of category assignments, we use the standard recall, precision, and $F1$ measures. Recall denotes the ratio of correct assignments by the system divided by the total number of correct assignments. Precision is the ratio of correct assignments by the system divided by the total number of the system's assignments.

The $F1$ measure which combines recall (r) and precision (p) with an equal weight is $F1(r, p) = \frac{2rp}{r+p}$.

4.2 Results and Discussion

The result is shown in Table 2.

	category	Performance		
		miR	miP	miF1
NB	all	0.684	0.419	0.519
	parts	0.565	0.523	0.543
SVMs	all	0.318	0.258	**0.285**
	parts	0.795	0.554	0.653
Manipulating data	all	0.703	0.704	0.704
	parts	0.720	0.692	0.700

Table 2: Categorization accuracy

'NB', 'SVMs', and 'Manipulating data' denotes the result using Naive Bayes, SVMs classifiers, and our method, respectively. 'miR', 'miP', and 'miF1' refers to the micro-averaged recall, precision, and F1, respectively. 'all' in Table 2 shows the results of all 102 categories. The micro-averaged F1 score of our method in 'all' (0.704) is higher than the NB (0.519) and SVMs scores (0.285). We note that the F1 score of SVMs (0.285) is significantly lower than other models. This is because we could not obtain a classifier to judge the category 'corporate/industrial' in the top level within 10 days using a standard 2.4 GHz Pentium IV PC with 1,500 MB of RAM. We thus eliminated the category and its child categories from the 102 categories. The number of the remaining categories in each level is 24 top, 14 second, 29 third, and 1 fourth level. 'Parts' in Table 2 denotes the results. There is no significant difference between 'all' and 'parts' in our method, as the F1 score of 'all' was 0.704 and 'parts' was 0.700. The F1 of our method in 'parts' is also higher than the NB and SVMs scores.

Table 3 denotes the amount of training data used to train NB and SVMs in our method and test data judged by each classifier. We can see that our method makes the computation of the SVMs more efficient, since the data trained by SVMs is only 23,243 from 150,939 documents.

Table 4 illustrates the results of three methods according to each category level. 'Training' in 'Manipulating data' denotes the number of documents used to train SVMs. The overall F1 value of NB, SVMs, and our method for the 25 top-level cate-

Manipulating data	# of selected documents		miR	miP	miF1
	training	test			
NB	150,939	76,650	0.798	0.674	0.730
SVMs	23,243	43,592	0.789	0.588	0.674
Total performance			0.703	0.704	0.704

Table 3: # of selected documents and categorization accuracy

	Top level(25 categories)			
	training	miR	miP	miF1
NB	147,576	0.877	0.573	0.693
SVMs	147,576	0.358	0.325	0.341
Manipulating data	22,528	0.836	0.679	0.715
	Second level(33 categories)			
	training	miR	miP	miF1
NB	129,130	0.559	0.529	0.543
SVMs	129,130	0.327	0.302	0.314
Manipulating data	17,667	0.833	0.478	0.608
	Third level(43 categories)			
	training	miR	miP	miF1
NB	92,320	0.609	0.383	0.471
SVMs	92,320	0.318	0.258	0.258
Manipulating data	12,482	0.820	0.481	0.606
	Fourth level(1 category)			
	training	miR	miP	miF1
NB	150,939	0.397	0.184	0.251
SVMs	150,939	0.318	0.258	0.258
Manipulating data	150,939	0.297	0.241	0.265

Table 4: Categorization accuracy by category level

gories is 0.693, 0.341, and 0.715, respectively. Classifying large corpora with similar categories is a difficult task, so we did not expect to have exceptionally high accuracy like Reuters-21578 (0.85 F1 score). Performance on the original training set using SVMs is 0.285 and using NB is 0.519, so this is a difficult learning task and generalization to the test set is quite reasonable.

There is no significant difference between the overall F1 value of the second(0.608) and third level categories(0.606) in our method, while the accuracy of the other methods drops when classifiers select sub-branches, in third level categories. As Dumais et. al. mentioned, the results of our experiment show that performance varies widely across categories. The highest F1 score is 0.864 ('Commodity markets' category), and the lowest is 0.284 ('Eco-

nomic performance' category).

The overall F1 values obtained by three methods for the fourth level category ('Annual result') are low. This is because there is only one category in the level, and we thus used all of the training data, 150,939 documents, to learn models.

The contribution of the hierarchical structure is best explained by looking at the results with and without category hierarchies, as illustrated in Table 5. It is interesting to note that the results of both NB and our method clearly demonstrate that incorporating category hierarchies into the classification method improves performance, whereas hierarchies degraded the performance of SVMs. This shows that the separation of one top level category(C) from the set of the other 24 top level categories is more difficult than separating C from the set of all the other 101 categories in SVMs.

Table 6 illustrates sample words which have the highest weighted value calculated using formula (3). Recall that in SVMs each value of word w_k $(1 \leq k \leq n)$ is calculated using formula (3), and the larger value of w_k is, the more the word w_k features positive examples. Table 6 denotes the results of two binary classifiers. One is a classifier that separates documents assigned the 'Economics' category(positive examples) from documents assigned a set of the other 24 top level categories, i.e. 'hierarchy'. The other is a classifier that separates documents with the 'Economics' category from documents with a set of the other 101 categories, i.e., 'non-hierarchy'. Table 6 shows that in 'Non-hierarchy', words such as 'economic', 'economy' and 'company' which feature the category 'Economics' have a high weighted value, while in 'hierarchy', words such as 'year' and 'month' which do not feature the category have a high weighted value. Further research using various subsets of the top level categories is necessary to fully understand the influence of the hierarchical structure created by

	Non-hierarchy			Hierarchy		
	miR	miP	miF1	miR	miP	miF1
NB	0.667	0.407	0.506	0.684	0.419	0.519
SVMs	0.655	0.524	**0.582**	0.318	0.258	**0.258**
Manipulating data	0.772	0.485	0.596	0.703	0.704	0.704

Table 5: Non-hierarchical v.s. Hierarchical categorization accuracy

humans.

Economics	
Hierarchy	access, Ford, Japan, Internet, economy, year, sale, service, month, market
Non-hierarchy	economic, economy, industry, ltd., company, Hollywood, business, service, Internet, access

Table 6: Sample words

Finally, we compare our results with a well-known technique, *ensemble* strategies. In the experiment using ensemble, we divided a training set into ten folds for each category level. Once the individual classifiers are trained by SVMs they are used to classify test data. Each classifier votes and the test data is assigned to the category that receives more than 6 votes[3]. The result is illustrated in Table 7.

In Table 7, 'Non-hierarchy' and 'Hierarchy' denotes the result of the 102 categories treated as a flat non-hierarchical problem, and the result using hierarchical structure, respectively. We can find that the result of *ensemble* with hierarchy(0.704 F1) outperforms the result with non-hierarchy(0.532 F1). A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are *accurate* and *diverse* (Hansen and Salamon, 1990). An accurate classifier is one that has an error rate better than random guessing on new test data. Two classifiers are diverse if they make different errors on new data points. Given our result, it may be safely said, at least regarding the Reuters 1996 corpus, that hierarchical structure is more effective for constructing ensembles, i.e., an ensemble of classifiers which are constructed by the training data with fewer than 30 categories in each level is more *accurate* and *diverse*. Table 7 shows that our method and *ensemble* perform equally (0.704 F1

---

[3] 6 votes was the best results among 10 different voting schemes in the experiment.

score) when we use hierarchical structure. However, the computation of the former is far more efficient than the latter. Furthermore, we see that our method (0.596 F1 score) slightly outperforms *ensemble* (0.532 F1 score) when the 102 categories are treated as a flat non-hierarchical problem.

5 Conclusions

We have reported an approach to text classification which manipulates large corpora using NB and SVMs. Our main conclusions are:

- Our method outperforms the baselines, since the micro-averaged $F1$ score of our method was 0.704 and the baselines were 0.519 for NB and 0.285 for SVMs.

- As shown in previous researches, hierarchical structure is effective for classification, since the result of our method using hierarchical structure led to as much as a 10.8% reduction in error rates, and up to 1.3% with NB.

- There is no significant difference between the F1 scores of our method and the *ensemble* method with hierarchical structure. However, the computation of our method is more efficient than the *ensemble* method in the experiment.

Future work includes (i) extracting features which discriminate between categories within the same top-level category, (ii) investigating other machine learning techniques to obtain further advantages in efficiency in the manipulating data approach, and (iii) evaluating the manipulating data approach using automatically generating hierarchies(Sanderson and Croft, 1999).

	Non-hierarchy			Hierarchy		
	miR	miP	miF1	miR	miP	miF1
Ensemble	0.625	0.464	**0.532**	0.704	0.705	**0.704**
Manipulating data	0.772	0.485	**0.596**	0.704	0.703	**0.704**

Table 7: Performance of *Ensemble* v.s. Manipulating data

References

L. D. Baker and A. K. McCallum. 1998. Distributional Clustering of Words for Text Classification. In *Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 96–103.

T. G. Dietterich. 2000. Ensemble Methods in Machine Learning. In *Proc. of the 1st International Workshop on Multiple Classifier Systems*.

S. Dumais and H. Chen. 2000. Hierarchical Classification of Web Content. In *Proc. of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 256–263.

S. Dumais, J. Platt, D. Heckerman, and M. Sahami. 1998. Inductive Learning Algorithm and Representations for Text Categorization. In *Proc. of ACM-CIKM98*, pages 148–155.

Y. Freund and R. E. Schapire. 1996. Experiments with a New Boosting Algorithm. In *Proc. of the 13th International Conference on Machine Learning*, pages 148–156.

L. Hansen and P. Salamon. 1990. Neural Network Ensembles. *IEEETrans. Pattern Analysis and Machine Intell.*, 12:993–1001.

T. Joachims. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proc. of the Conference on Machine Learning*, pages 96–103.

D. Koller and M. Sahami. 1997. Hierarchically Classifying Documents using Very Few Words. In *Proc. of the 14th International Conference on Machine Learning*, pages 170–178.

W. Lam and C. Y. Ho. 1998. Using a Generalized Instance Set for Automatic Text Categorization. In *Proc. of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 81–89.

D. D. Lewis and M. Ringuette. 1994. Comparison of Two Learning Algorithms for Text Categorization. In *Proc. of the 3rd Annual Symposium on Document analysis and Information Retrieval*.

D. D. Lewis. 1992. An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task. In *Proc. of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 37–50.

A. K. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng. 1998. Improving Text Classification by Shrinkage in a Hierarchy of Classes. In *Proc. of the 15th International Conference on Machine Learning*, pages 359–367.

A. K. McCallum. 1999. Multi-Label Text Classification with a Mixture Model Trained by EM. In *Revised version of paper appearing in AAAI'99 Workshop on Text Learning*.

T. Mitchell. 1996. *Machine Learning*. McGraw Hill.

D. Mladenic and M. Grobelnik. 1998. Feature Selection for Classification based on Text Hierarchy. In *Proc. of the Workshop on Learning from Text and the Web*.

M. Sanderson and B. Croft. 1999. Deriving Concept Hierarchies from Text. In *Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 206–213.

H. Schmid. 1995. Improvements in Part-of-Speech Tagging with an Application to German. In *Proc. of the EACL SIGDAT Workshop*.

V. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer.

J. Weston and C. Watkins. 1998. Multi-Class Support Vector Machines. In *Technical Report CSD-TR-98-04*.

Y. Yang and X. Liu. 1999. A Re-Examination of Text Categorization Methods. In *Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49.

Y. Yang and W. J. Wilbur. 1995. Using Corpus Statistics to Remove Redundant Words in Text Categorization. *Journal of American Society Information Science*.