ELSEVIER

Information Processing and Management 40 (2004) 421–439

INFORMATION PROCESSING & MANAGEMENT

www.elsevier.com/locate/infoproman

Co-trained support vector machines for large scale unstructured document classification using unlabeled data and syntactic information

Seong-Bae Park, Byoung-Tak Zhang [*]

School of Computer Science and Engineering, Seoul National University, 151-744 Seoul, South Korea

Received 3 October 2002; accepted 19 September 2003

Abstract

Most document classification systems consider only the distribution of content words of the documents, ignoring the syntactic information underlying the documents though it is also an important factor. In this paper, we present an approach for classifying large scale unstructured documents by incorporating both the lexical and the syntactic information of documents. For this purpose, we use the co-training algorithm, a partially supervised learning algorithm, in which two separated views for the training data are employed and the small number of labeled data are augmented by the large number of unlabeled data. Since both the lexical and the syntactic information can play roles of separated views for the unstructured documents, the co-training algorithm enhances the performance of document classification using both of them and a large number of unlabeled documents. The experimental results on Reuters-21578 corpus and TREC-7 filtering documents show the effectiveness of unlabeled documents and the use of both the lexical and the syntactic information.
© 2003 Elsevier Ltd. All rights reserved.

Keywords: Text categorization; Co-training; Support vector machines; Syntactic information; Text chunking

1. Introduction

Automatic document classification is an important research area in information retrieval and has a great potential for many applications handling text such as routing and filtering. Its aim is to

[*] Corresponding author.
E-mail addresses: sbpark@bi.snu.ac.kr (S.-B. Park), btzhang@bi.snu.ac.kr (B.-T. Zhang).

0306-4573/$ - see front matter © 2003 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2003.09.003

assign a given document to the predefined category to which it belongs. Up to now, various kinds of algorithms based on machine learning or statistics have been applied to this task and showed relatively high performance (Joachims, 1998; Kim, Hahn, & Zhang, 2000). However, most of them applied to this task have been using a simple *bag-of-words* representation of documents where each feature corresponds to a single word (Nigam, McCallum, Thrun, & Mitchell, 2000). That is, they use only the distribution of content words, assuming that the words are independent one another. But, this representation ignores *linguistic information* underlying the documents, which is another important factor in filtering texts.

Each document has its own traits in the style, and the syntactic information is one of the best measures to capture the stylistic divergence among the different kinds of documents (Biber, 1995). Although the syntactic features can give much information in classifying the documents, they are not widely used due to their lack of formal definition and complicated representation. In addition, unfortunately, the current NLP (natural language processing) techniques are not able to provide accurate results in syntax analyzing. However, some studies show that text chunks can give enough information on syntax analysis instead of full parsing (Stamatatos, Fakotakis, & Kokkinakis, 2000).

Another problem in document classification is that there are a great number of inexpensive unlabeled documents while there are a few labeled documents (Nigam et al., 2000). It is very expensive to obtain the labeled documents, since labeling of documents must be done by human experts. The co-training algorithm (Blum & Mitchell, 1998) is one of the successful algorithms that handles unlabeled data. It is in general applied to the problems where there are two distinct views of each example in the dataset. For example, web pages can be considered to have two views: one is for their contents and the other is for link information. It learns separate classifiers over each of the views, and augments a small number of labeled examples incorporating the large number of unlabeled examples. Its final prediction is made by combining its two classifiers.

In order to incorporate unlabeled examples with labeled examples, a metric to measure the confidence of labeling an unlabeled example is required in the co-training algorithm. Recently, the concept of margin is introduced with the emergence of support vector machines (Scholkopf, Burges, & Smola, 1999), where the margin is defined as the product of true label (assumed to be -1 or $+1$) and the classifier's output. If the prediction of the classifier is correct, the margin is positive, otherwise negative. When the output of the classifier is real value, it can be considered to be the confidence on prediction.

We propose a co-trained support vector machines (SVM) for document classification. It is based on the co-training algorithm, so that it effectively uses not only given small number of labeled documents but a great number of unlabeled documents. For the two views of the co-training algorithm, we use both lexical information and syntactic information. Thus, the proposed method can be applied to classify the unstructured normal documents where there is no link information. Since we adopt SVMs as base classifiers it is natural to use the margin as a confidence measure for predicting unlabeled examples.

This paper is organized as follows. Section 2 explains several approaches to incorporating unlabeled documents and linguistic information in document classification. Section 3 describes the co-training algorithm using both the lexical and the syntactic information. Section 4 reports the experimental results on Reuters-21578 corpus and TREC-7 filtering documents. Finally, Section 5 draws conclusions.

2. Related work

Incorporating unlabeled examples is not a new scheme in information retrieval. It has been a major research topic under the name of *pseudo relevance feedback*, a variance of relevance feedback. The idea of relevance feedback is to examine a portion of the retrieved documents and assign relevance values to each of them. The contents of the documents are analyzed to make the query closer towards the examined relevant documents and away from the examined non-relevant documents. The most common model for relevance feedback is proposed by Rocchio (1971). The terms in the new query are ranked by the weighted sum of the term weights in the current query, the known relevant documents, and the known non-relevant documents. That is, the new query is constructed with several top terms ranked by:

$$w_i(Q') = \alpha \cdot w_i(Q) + \beta \cdot w_i(R) + \gamma \cdot w_i(N),$$

where $w_i(Q')$ and $w_i(Q)$ are the weights of a term w_i in the new and current queries, $w_i(R)$ and $w_i(N)$ are the weights of w_i in the relevant and non-relevant documents, and α, β, and γ are the parameters to be determined. Many experiments on relevance feedback empirically show that it improves retrieval performance dramatically (Buckley & Salton, 1995; Drucker, Shahrary, & Gobbon, 2001; Leuski, 2000; Salton & Buckley, 1990). Pseudo relevance feedback typically adds new terms to the initial query by assuming that the several top documents in the initial ranked output are relevant.

In most discussions of relevance feedback, it is usually assumed that the initial query is of high quality, so that a major portion of the returned documents by the initial query are highly relevant. However, in many situations such an assumption is impractical. In document classification, we always cannot trust all the labels of unlabeled documents estimated by the current classifier, since usually there are a small number of labeled documents while there are a great number of unlabeled documents. If the trained classifier does not coincide to the intrinsic model which generates the documents, the performance will be hurt by the unlabeled documents. Thus, more sophisticated methods are required for document classification using unlabeled ones.

Many recent studies in information retrieval focus on viewing documents as models. They show that the language modeling is a very effective framework for information retrieval systems (Ponte & Croft, 1998; Zhai & Lafferty, 2001). Even the term weighting method used in traditional machine learning approach to document classification can be considered as a kind of language model—*unigram*. However, these models usually do not contain any syntactic information, though they are precision-enhancing devices.

Another way to overcome the limit of traditional term weighting method is to use syntactic and semantic information. The merits of syntactic information is investigated by Lewis and Jones (1996). Other studies also show that it is quite useful as content-identifier (Mitra, Buckley, Singhal, & Cardie, 1997; Turpin & Moffat, 1999). They state that using non-NLP generated phrases as terms in vector space retrieval is helpful at low recall level, while it is not helpful at high recall level. However, it is required to obtain more accurate syntactic information than just predefined window-size word sequences for its general use.

Because the current NLP techniques do not provide accurate information enough to be used in information retrieval, text chunking is considered to be an alternative to full parsing (Stamatatos et al., 2000). Text chunking is to divide text into syntactically related non-overlapping segments of words. Since this task is formulated as estimating an identifying function from the information (features) available in the surrounding context, various techniques have been applied to it (CoNLL, 2000).

3. Co-training algorithm for classifying unstructured documents

3.1. Co-training algorithm

The co-training algorithm is one of the successful algorithms handling unlabeled examples (Blum & Mitchell, 1998). It is in general applied to the problems where there are two distinct views of each example in the dataset. It learns separate classifiers over each of the views, and augments a small set of labeled examples by incorporating unlabeled examples. Its final prediction is made by combining their predictions to decrease classification errors. The larger is the variance of the classifiers when both classifiers are unbiased, the better is the performance of the algorithm. Since the co-training uses two classifiers with distinct views, its performance is better than any single classifier.

Fig. 1 outlines the co-training algorithm. It uses two distinct views V_1 and V_2 when learning from labeled and unlabeled data, and incrementally upgrades classifiers (h_1 and h_2) over each view. Each classifier is initialized with a few labeled examples. At every iteration, each classifier chooses unlabeled examples to add them to the set of labeled examples, L. The selected unlabeled examples are those which each classifier can determine their label with the highest confidence. After that, the classifiers are trained again using the augmented labeled set. This is repeated until all the unlabeled examples are exhausted. The final output of the algorithm is given as a combination of the two classifiers. Given an example **x** to be classified, the probability of the possible

```
Give $L$ : a set of labeled examples,
  $U$ : a set of unlabeled examples

Do Until there is no unlabeled example.
  1. Train classifier $h_1$ on view $V_1$ of $L$.
  2. Train classifier $h_2$ on view $V_2$ of $L$.
  3. Allow $h_1$ to determine labels of examples in $U$.
  4. Allow $h_2$ to determine labels of examples in $U$.
  5. Determine $U'$, a subset of $U$, whose elements are
     most confidently labeled by $h_1$ and $h_2$.
  6. $U = U \setminus U'$
  7. $L = L + U'$
```

Fig. 1. An abstract of the co-training algorithm.

class c_j is determined by multiplying two posterior probabilities. The best class c^* of $\mathbf{x}$ is set to the one with the highest probability:

$$c^* = \arg\max_{c_j \in C} \big(p(c_j|\mathbf{x}) = p_{h_1}(c_j|\mathbf{x}) p_{h_2}(c_j|\mathbf{x}) \big),$$

where C is the set of all possible classes.

Blum and Mitchell (1998) formalized the co-training settings and provided theoretical guarantee under certain assumptions. Their first assumption is that the data distribution is compatible with the target function. That is, the target functions over each view predict the same class for most examples. The second assumption is that the views are conditionally independent. If this assumption holds, the added examples at each iteration will be at least as informative as random examples. Thus, the iteration can progress though there are some mislabeled examples during the learning. However, this assumption is somewhat unrealistic in practice, since the views from the same data are inclined to be related with each other in some way. Nigam and Ghani (2000) performed thorough empirical investigation on the conditional independence of the views (Nigam & Ghani, 2000). Their experiments showed that the view independence affects the performance of the co-training algorithm, and the algorithm is still more effective than other algorithms incorporating unlabeled examples even when some dependence exist between the views.

3.2. Two views

Though an idea is proposed for applying the co-training algorithm to the problems without a known, natural division of features for two views, it is yet impractical because its too high computational complexity. Thus, most applications of the co-training algorithm are on web page classification, because there are two natural views for the web pages: a content view and a link view. However, it is not clear how to construct two independent views for the normal unstructured documents without link information.

One possible view for document classification is to treat each document as a vector whose elements are the weight to the vocabulary. Most machine learning algorithms applied to document classification adopt this representation. The main drawbacks of this representation are that (i) it assumes that each word in the documents is independent each other, and (ii) it ignores much linguistic information underlying in the documents.

Stamatatos et al. (2000) showed experimentally that the syntactic information is a reliable clue for document classification. One additional benefit in using syntactic information for document classification by the co-training algorithm is that it is somewhat independent from term weights. However, unfortunately, the current natural language processing techniques are not able to provide accurate syntactic analysis results. Thus, they showed that the *text chunks* instead of full parsing are good features enough to provide syntactic information for document classification. The chunks can be obtained in high accuracy with superficial investigation.

Therefore, we define two distinct views for unstructured documents, so that the co-training algorithm can be naturally applied to classifying them. The two views used in this paper are:

- *Lexical view*. Most machine learning algorithm applied to automatic document classification are based on $tf \cdot idf$, a commonly used term weighting scheme in information retrieval. The

tf factor is an estimation of the occurrence probability of a term if it is normalized, and the *idf* is the amount of information related to the occurrence of the term.
- *Syntactic view*. Each document is represented in a vector in which the elements are syntactic features, and the features are derived from text chunking. This information can support finding particular or specific style of the documents.

That is, h_1 is trained with lexical view, while h_2 is with syntactic view.

3.3. Syntactic features

To represent the documents in vectors whose elements are syntactic information, all documents need to be represented with chunk information. Since the unstructured documents are normally raw, all sentences in the documents must be chunked in the preprocessing step of classification. The purpose of chunking is to divide a given sentence into non-overlapping segments. Let us consider the following sentence.

(1) He reckons the current deficit will narrow to only #1.8 billion in September.

This sentence, composed of 15 words including a period, is grouped into nine segments after chunking as follows:

(2) [**NP** He] [**VP** reckons] [**NP** the current deficit] [**VP** will narrow] [**PP** to] [**NP** only #1.8 billion] [**PP** in] [**NP** September] [**O**.]

In order to chunk the sentences in the documents, the lexical information and the POS (part-of-speech) information on the contextual words are required. Brill's tagger (Brill, 1992) is used to obtain POS tags for each word in the documents. The chunk type of each word is determined by support vector machines trained with the dataset of CoNLL-2000 shared task. [1]

Although there are 12 types of phrases in CoNLL-2000 dataset, we consider, in this paper, only five types of phrases: NP, [2] VP, ADVP, PP, and O, where O implies none of NP, VP, ADVP, and PP. This is reasonable because those five types take 97.31% of the dataset (Park & Zhang, 2002) and are major phrases for constituting a sentence. Each phrase except O has two kinds of chunk types: B-XP and I-XP. For instance, B-NP represents the first word of the noun phrase, while I-NP is given to other words in the noun phrase. Thus, we consider nine chunk types.

The use of support vector machines showed the best performance in the shared task of CoNLL-2000 (Kudo & Matsumoto, 2000). The contexts used to identify the chunk type of the *i*th word w_i in a sentence are:

$$w_j, \mathrm{POS}_j \quad (j = i-2,\ i-1,\ i,\ i+1,\ i+2),$$
$$c_j \quad (j = 1-2,\ i-1),$$

[1] http://lcg-www.uia.ac.be/conll2000/chunking

[2] NP represents a noun phrase, VP a verb phrase, ADVP an adverb phrase, and PP a prepositional phrase.

Table 1
Syntactic features for document classification

Feature	Description
SF1	Detected NPs/total detected chunks
SF2	Detected VPs/total detected chunks
SF3	Detected PPs/total detected chunks
SF4	Detected ADVPs/total detected chunks
SF5	Detected Os/total detected chunks
SF6	Words included in NPs/detected NPs
SF7	Words included in VPs/detected VPs
SF8	Words included in PPs/detected PPs
SF9	Words included in ADVPs/detected ADVPs
SF10	Words included in Os/detected Os
SF11	Sentences/words

where POS_j and c_j are respectively the POS tag and the chunk type of w_i. Since SVMs are basically binary classifiers and there are nine chunk types, SVMs are extended to multi-class classifiers by *pairwise classification* (Scholkopf et al., 1999).

Table 1 shows the features used to represent documents using text chunks. Top five features represent how often the phrases are used in a document, the following five features implies how long they are, and the final feature means how long a sentence is on the average. That is, every document is represented in a 11-dimensional vector. For instance, let us consider a news article in Fig. 2. The number of NPs in this article is 13, while those of VPs, PPs, ADVPs, and Os are 7, 4, 1, and 6 respectively. The average number of words in NPs is $\frac{8+1+2+10+3+7+4+2+2+4+2+2+4}{13} = \frac{51}{13}$, and those of VPs, PPs, ADVPs, and Os are $\frac{1+3+2+1+1+3+3}{7} = \frac{14}{7}$, $\frac{1+1+1+1}{4} = \frac{4}{4}$, $\frac{1}{1}$, and $\frac{1+1+1+1+1+1}{6} = \frac{6}{6}$. Since the number of sentences is 3, this article can be represented as

$$\left\langle \frac{13}{31}, \frac{7}{31}, \frac{4}{31}, \frac{1}{31}, \frac{6}{31}, \frac{51}{13}, \frac{14}{7}, \frac{4}{4}, \frac{1}{1}, \frac{6}{6}, \frac{3}{76} \right\rangle.$$

3.4. Support vector machines as base classifiers

For the classifiers in the co-training algorithm, we adopt support vector machines that show significant improvement over other machine learning algorithms when applied to document classification (Joachims, 1998). SVMs can be understood in the context of Fig. 3. In this figure, the black circles represent the relevant documents and the white circles represents the irrelevant ones. When SVMs are constructed, two hyperplanes are formed which are shown as dotted lines in Fig. 3. One hyperplane goes through one or more relevant documents while the other goes through one or more irrelevant documents. The documents lying on the hyperplanes are the support vectors which define in fact the hyperplanes. If we define the margin as the distance from a hyperplanes to the decision boundary shown as a thick line, then a SVM maximizes this margin. When the true decision boundary is non-linear, we can maximize the margin by mapping the documents into some higher order space in which the documents can be separated linearly. This mapping functions are called the kernel functions.

Standard Oil Co and BP North America Inc said they plan to form a venture to manage the money market borrowing and investment activities of both companies. BP North America is a subsidiary of British Petroleum Co Plc, which also owns a 55 pct interest in Standard Oil. The venture will be called BP Standard Financial Trading and will be operated by Standard Oil under the oversight of a joint management committee.

(a) raw text

[**NP** Standard Oil Co and BP North America Inc] [**VP** said] [**NP** they] [**VP** plan to form] [**NP** a venture] [**VP** to manage] [**NP** the money market borrowing and investment activities of both companies] [**O**.] [**NP** BP North America] [**VP** is] [**NP** a subsidiary of British Petroleum Co Plc] [**O** ,] [**O** which] [**ADVP** also] [**VP** owns] [**NP** a 55 pct interest] [**PP** in] [**NP** Standard Oil] [**O**.] [**NP** The venture] [**VP** will be called] [**NP** BP Standard Financial Trading] [**O** and] [**VP** will be operated] [**PP** by] [**NP** Standard Oil] [**PP** under] [**NP** the oversight] [**PP** of] [**NP** a joint management committee] [**O**.]

(b) chunked text

Fig. 2. An example news article from Reuters-21578 corpus.

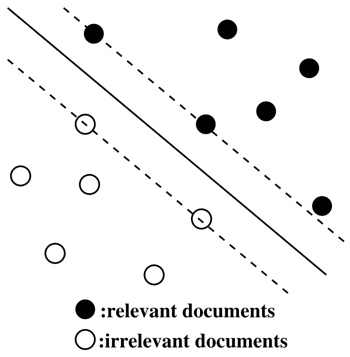


Fig. 3. The idea of support vector machines.

At each iteration of the co-training, the most confident $|U'|$ examples should be selected from U. The SVMs provide a natural way to calculate the confidence of labeling unlabeled examples by a margin. The margin m for an unlabeled example $\mathbf{x}_i$ is defined as

$$m = y_i(\mathbf{w} \cdot \mathbf{x}_i + b),$$

where $y_i \in \{-1, +1\}$ is the label predicted by the hyperplane with the trained parameters $\mathbf{w}$ and b. That implies, the margin can be considered to be a distance from $\mathbf{x}_i$ to the hyperplane, assuming that the predicted label is correct. The more distant $\mathbf{x}_i$ lies from the hyperplane, the more confident it is to predict the label of $\mathbf{x}_i$. Since SVMs are not probabilistic models, the final prediction of the co-training cannot be made by multiplying the probabilities of each classifier. Instead, the final prediction is made only by the classifier whose margin is larger than the other. Joachims (1999) proposed a *transductive SVM* to adapt SVMs to handling unlabeled data, but it focuses on how to utilize the unlabeled documents without attention to syntactic information.

4. Experiments

4.1. Datasets

4.1.1. Reuters-21578 ModApte Split

The Reuters-21578 corpus is the most commonly used benchmark corpus in text classification. It consists of over 20,000 the Reuters newswire articles in the period between 1987 and 1991, and has 135 kinds of topics while only 10 of them are used for experiments. There are three versions to divide this corpus into a training set and a test set: "ModLewis", "ModApte", and "ModHayes". Among them, "ModApte" which is most widely used is employed in this paper. In this version, there are 9603 training documents, 3299 test documents, and 27,863 unique words after stemming and stop word removal.

4.1.2. TREC-7

The dataset for the TREC-7 filtering track consists of the articles from AP News for the years between 1988 and 1990, Topics 1–50. The 1988 AP documents are used for a training set, and the 1989–1990 AP documents for a test set. We do not use any information from the topic, such as 'description'. For documents representation using lexical information, we stem them and remove words from the stop-list. No thesaurus or other datasets are used. Though there are more than 160,000 words in this dataset, we choose only 7289 most frequently appearing words. There are 79,898 articles in the training set, but only 9572 articles, just about 12% of them are labeled. Thus, we regard those 9572 articles as a labeled document set, and the remaining 88% of the articles as an unlabeled document set.

4.2. Performance measure

To evaluate the classification performance of the proposed method, we use *utility* measure, for example LF1. Let R_+ be the number of relevant and retrieved documents, N_+ the number of non-relevant but retrieved documents, R_- the number of relevant but non-retrieved documents, and N_- the number of non-relevant and non-retrieved documents. Then, *linear utility* is defined as

$$\text{Linear Utility} = aR_+ + bN_+ + cR_- + dN_-,$$

where a, b, c and d are constant coefficients. The LF1 and LF2 measures are defined:

$$\text{LF1} = 3R_+ - 2N_+,$$

$$\text{LF2} = 3R_+ - N_+.$$

The drawback of linear utility is that it is meaningless to average through all topics because a few topics will have dominant effects on the results. Thus, to show the whole performance effectively we use the *scaled utility* (Hull, 1998) defined as

$$\text{Scaled Utility} = \frac{\max\{u(S,T), U(s)\} - U(s)}{\text{Max}U(T) - U(s)},$$

where $u(S,T)$ is the utility of system S for topic T, $\text{Max}U(T)$ is the maximum possible utility score for topic T, and $U(s)$ is the utility of retrieving s non-relevant documents. Since we consider only 10 topics in Reuters-21578 corpus, we use LF1 measure. However, there are 50 topics in TREC-7, so that we use the scaled linear utility.

4.3. Experimental results

4.3.1. Effect of syntactic information

4.3.1.1. Reuters-21578. When we do not consider unlabeled examples on Reuters-21578 dataset, the effect of using additional syntactic information is given in Table 2. The classification performance using both the lexical and the syntactic information outperforms the one using any single information for most classes. We find that the larger is the number of relevant documents in the training set, the higher is the LF1 for using lexical information. This is because the SVMs tend to achieve lower error rate and higher utility value when a class has a large number of relevant documents. This trend is kept except *grain* class even when we use both information. For *grain* class, the extremely low LF1 for the syntactic information causes the performance using both information to be dominated by the lexical information.

4.3.1.2. TREC-7. Fig. 4 shows the advantage of using both the lexical and the syntactic information. These results are obtained in considering both labeled and unlabeled documents. The X-axis is the topic numbers, while the Y-axis is the difference of performance that we obtain by using both kinds of information rather than single information. Thus, positive values mean that using both information outperforms using single one. We find out by these figures that using both

Table 2
The accuracy improvement by using syntactic information on Reuters-21578 corpus

Class	Relevant documents	Lexical	Syntactic	Both
Earn	2877	2876	2504	*2895*
Acq	1650	1587	1028	*1642*
Money-fx	538	188	147	*193*
Grain	433	*26*	14	*26*
Crude	389	292	150	*312*
Trade	369	153	114	*155*
Interest	347	130	112	*141*
Wheat	212	113	88	*116*
Ship	197	98	70	*108*
Corn	181	86	68	*87*

Lexical implies when we use only $tf \cdot idf$, *Syntactic* when we use only text chunks, and *Both* when we use both of them. The performance measure is LF1.

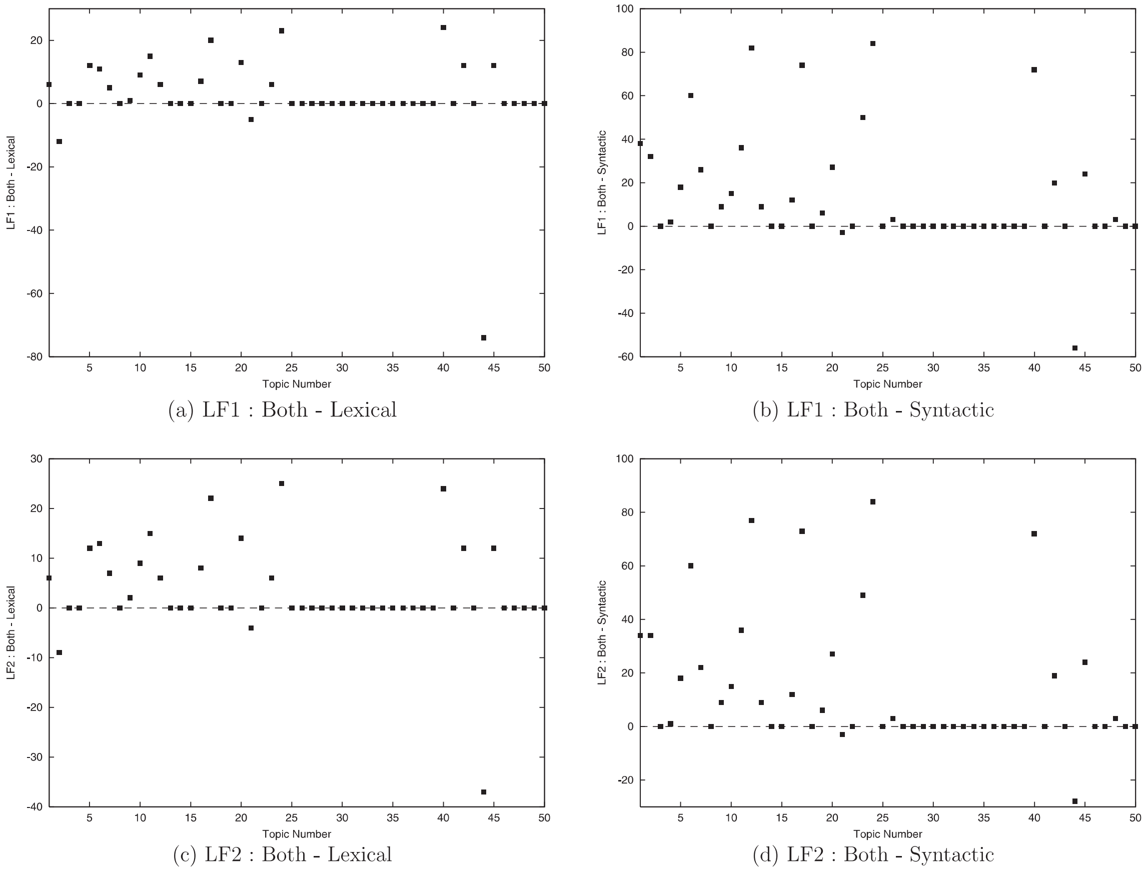


Fig. 4. The performance improvement by using both lexical and syntactic information. Two different measures are used: LF1 and LF2.

kinds of information improves the performance for both LF1 and LF2. In addition, the difference between *Both-Syntactic* is larger than *Both-Lexical* for LF1. This result coincides with Table 2 that using only lexical information is better than using only syntactic information.

The averaged performance of Fig. 4 is given in Table 3. Since the results in the table did not enter TREC pools, all the unjudged documents in the test set are ignored. The use of both information gives the best result for both measures. Using only lexical information is better than using only syntactic information in LF1, but achieves the same performance in LF2. This

Table 3
The result on TREC-7 dataset

Measure	Lexical	Syntactic	Both
LF1	0.2005	0.1680	*0.2192*
LF2	0.2010	0.2010	*0.2155*

The performance measure is the averaged scaled LF1 and LF2 when $U(s)$ is set to 0.

coincides with the work of Mitra et al. (1997) that the performance of a phrase-based retrieval system is not superior than that of a term-based retrieval system. Even though we do not use the phrases for the index of documents, a similar result is obtained that the information derived from phrases are not better than terms. We have not studied this phenomenon in greater details, but we presume three reasons for it. First, we do not consider any content words in the document for syntactic information. Thus, it overgeneralizes the documents with styles. Second, there are only 1468 reporters in the training set of TREC-7, though there are more than 79,000 articles. One reporter should write about 50 articles on the average, but more than a half of them write less than 10 articles. That is, a few writers write a large portion of the articles. This forces the syntactic character of each reporter not to reveal and the performance of using only syntactic information to be low. The other reason is that there are a great number of non-sentence forms such as tables and listings in the training set since they are news articles. They make the syntactic analysis failed.

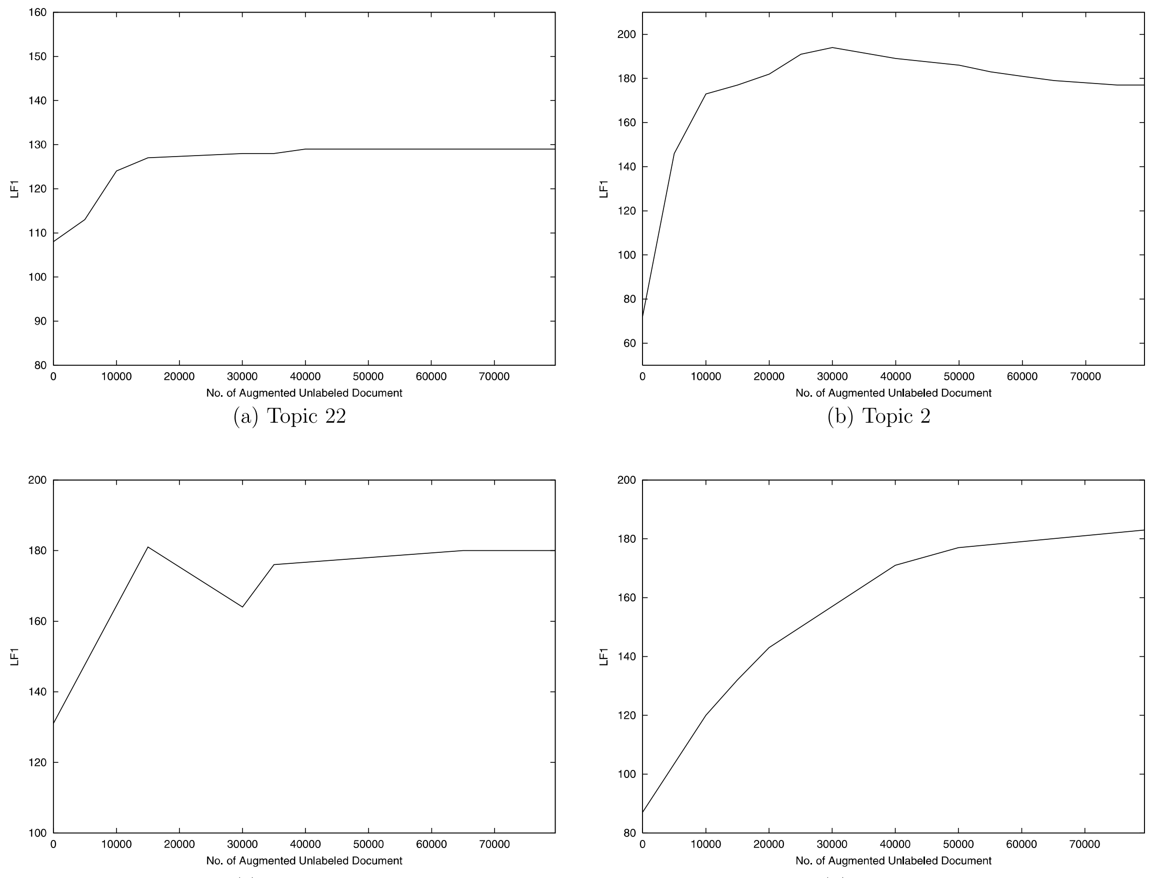
4.3.2. *Effect of unlabeled documents*

Now we will see how the unlabeled documents improves text classification under co-training algorithm.

4.3.2.1. TREC-7. Figs. 5 and 6 show that the performance of the proposed method is enhanced by unlabeled documents even though there are a small number of relevant documents. The graphs in Fig. 5 are on the topics with a great number of relevant documents, while those in Fig. 6 are on the topics with a small number of relevant documents. *Topic 22* has 358 relevant documents, but *Topic 31* has only 1 relevant ones. Most topics except *Topic 43* show the similar trends that the performance gets higher by using unlabeled documents and gets higher and higher as more unlabeled documents are used in the learning. This is a good point of support vector machines since they tend to correctly classify the relevancy with the larger number of documents.

4.3.2.2. Reuters-21578. Many previous work asserted that unlabeled documents improve the performance of document classification (Zhang & Oles, 2000). Our experiments also show that unlabeled documents are useful for better performance in document classification. Fig. 7 shows the effectiveness of unlabeled documents involved in the co-training algorithm. The X-axis represents the ratio of labeled documents to total documents, while Y-axis is the accuracy improvement by additional unlabeled documents. For *earn*, the unlabeled documents play a positive role when more than 10% of training documents are labeled. So is for *acq*, when more than 7% of training documents are labeled.

However, even when we obtain the highest improvement by unlabeled documents, it does not reach to the best performance when we know the labels of all the training examples beforehand. For example, the improvement of accuracy is 5.81% when 10% of documents are labeled in *acq*. In this case, the accuracy is just 89.93% while the accuracy with 100% labeled documents is 95.21%. This implies that some of the unlabeled documents are mislabeled during the co-training process and have a negative effect on the classifiers. The effectiveness of unlabeled documents can be maximized when the number of labeled ones is small. To fill a gap of the difference in accuracy, human intervention is needed. But, it is still an open problem when to intervene in the process.



(a) Topic 22 (b) Topic 2

(c) Topic 12 (d) Topic 11

Fig. 5. The analysis of the effects of unlabeled documents for the topics with a great number of relevant documents in TREC-7. The performance measure is LF1.

4.4. Analysis of the co-training algorithm

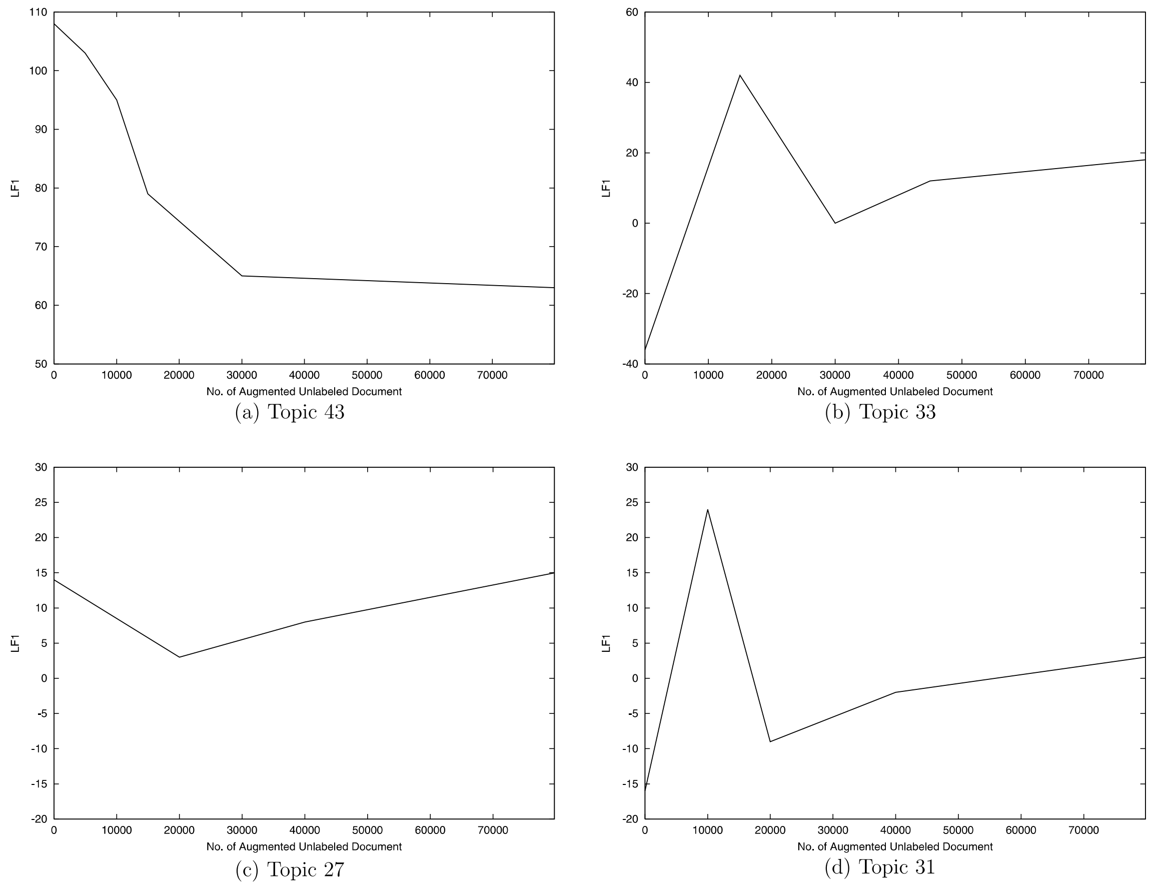
Recall that two assumptions of the co-training algorithm are:

- Each view is sufficient for classification.
- The two views should be conditionally independent each other.

In this section, we will see how these assumptions affect the performance of the co-training algorithm.

In general, the goal of most machine learning algorithms is to minimize the risk. That is, when training data is given as $\langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\rangle$ and $f(\mathbf{x})$ is a probabilistic estimator, the risk is given by a functional

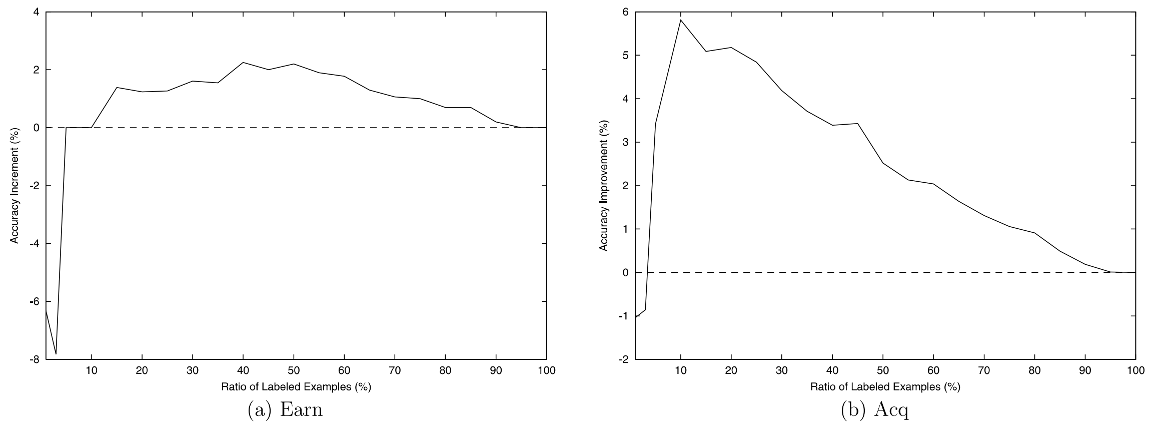
$$R(f) = \int L(y, f(\mathbf{x}))p(\mathbf{x}, y)\,\mathrm{d}\mathbf{x}\,\mathrm{d}y,$$



(a) Topic 43 (b) Topic 33

(c) Topic 27 (d) Topic 31

Fig. 6. The analysis of the effects of unlabeled documents for the topics with a small number of relevant documents in TREC-7. The performance measure is LF1.



(a) Earn (b) Acq

Fig. 7. The improvement in accuracy by using additional unlabeled documents.

where $L(y, f(\mathbf{x}))$ is a loss function and $p(\mathbf{x}, y)$ is a true distribution of data. Since $p(\mathbf{x}, y)$ is not known, the best we can do is to minimize the empirical risk

$$R_{\text{emp}}(f) = \frac{1}{m}\sum_{i=1}^{m} L(y_i, f(\mathbf{x}_i)).$$

Since both labeled and unlabeled data are used in the co-training algorithm, the empirical risk is reformulated as

$$R_{\text{emp}}(f) = R_1 + R_2 + R_3 + R_4,$$

where

$$R_1 = \sum_{\langle \mathbf{x}, y\rangle \in L} (y - h_1(\mathbf{x}))^2,$$

$$R_2 = \sum_{\langle \mathbf{x}, y\rangle \in L} (y - h_2(\mathbf{x}))^2,$$

$$R_3 = \sum_{\mathbf{x} \in U} (h_1(\mathbf{x}) - h_2(\mathbf{x}))^2,$$

$$R_4 = \left(\left(\frac{1}{|L|} \sum_{\langle \mathbf{x}, y\rangle \in L} y \right) - \left(\frac{1}{|L| + |U|} \sum_{\mathbf{x} \in L \cup U} \frac{h_1(\mathbf{x}) + h_2(\mathbf{x})}{2} \right) \right)^2.$$

R_1 and R_2 stand for the misclassification on labeled data L, and R_3 is the disagreement over unlabeled data U. If L and U are sampled from the same distribution $p(\mathbf{x}, y)$, their expectations of y should be same. This constraint is specified by R_4, because $h_1(\mathbf{x}) = h_2(\mathbf{x})$ for most $\mathbf{x}$ by the first assumption.

To see what happens if the first assumption is violated, the perceptrons with two output nodes, designed as in Fig. 8, are trained. Suppose that $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$ can be decomposed into two parts. One is a lexical part that consists of $x_1, x_2, \ldots, x_{|V|}$, and the other is a syntactic part composed of $x_{|V|+1}, \ldots, x_n$. Here, $|V|$ is a vocabulary size. The first output node is related with h_1. Thus, it is trained with $x_1, x_2, \ldots, x_{|V|}$. And, the second output node is trained with $x_{|V|+1}, \ldots, x_n$ because it is related with h_2. The final decision is made by multiplying h_1 and h_2 as done in the co-training algorithm. Table 4 shows the difference in performance when the different risk is used in training

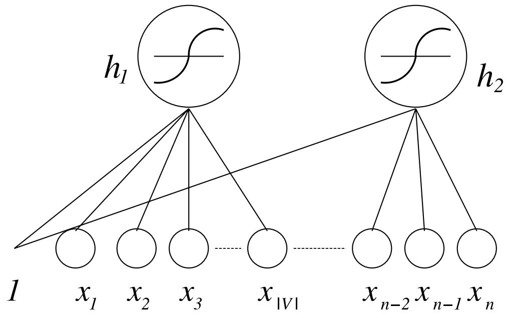


Fig. 8. A perceptron designed to see the effect of the first assumption in the co-training algorithm.

Table 4
The effect of the first assumption in the co-training algorithm

Risk	$R_1 + R_2 + R_3$	$R_1 + R_2 + R_3 + R_4$
LF1	0.2103	0.2192
LF2	0.2132	0.2155

The experiments are performed on TREC-7 dataset.

the perceptrons. If R_4 is not used in training the perceptrons on TREC-7 dataset, the perceptrons give 0.2103 of LF1 and 0.2132 of LF2. That is, lower performance is obtained by ignoring R_4, but the performance is not so much low compared with the case considering R_4. Therefore, it can be said that the first assumption is important in training the co-training algorithm even though it does not affect the performance of the co-training algorithm greatly.

Suppose that $X = \langle x_1, x_2, \ldots, x_{|V|} \rangle$ and $Y = \langle x_{|V|+1}, \ldots, x_n \rangle$. If X and Y are completely conditionally independent, their *conditional mutual information* should be zero, where the conditional mutual information of X and Y given class C is defined by

$$\begin{aligned} I(X;Y|C) &= H(X|C) - H(X|Y,C) \\ &= E_{p(x,y,c)} \log \frac{p(X,Y|C)}{p(X|C)p(Y|C)}. \end{aligned}$$

As shown in Table 5, X and Y are not completely conditionally independent. In order to see how much the performance is degraded when the conditional independence is violated, we randomly split $\mathbf{x}$ into exactly two parts. We perform the split five times and the results are shown in Table 5. The features in the random splits contain both lexical and syntactic information, and are related one another. Thus, the conditional mutual information of random splits are far larger than that of X and Y. This table shows that LF1 measure is reciprocally related with conditional mutual information. The LF1 is best when X and Y are used as two views, and the random split #2 of which conditional mutual information is largest shows the worst LF1.

Finally, Table 6 compares the *co-training* and the *self-training*. Self-training is an incremental algorithm like the co-training algorithm, but does not use two views. It builds a single SVM using only the labeled data and all features including both lexical and syntactic features. Then it labels the unlabeled data and some most confidently predicted data are added into the labeled data. This process iterates until all the unlabeled data are exhausted.

Table 5
The conditional mutual information and average LF1 for various feature splits

Features	Conditional MI (bits)	Average LF1
X and Y	12.17	567.5
Random split #1	451.12	153.9
Random split #2	532.33	121.8
Random split #3	486.92	138.3
Random split #4	459.04	151.4
Random split #5	510.81	127.0

The experiments are performed on Reuters-21578 dataset.

Table 6
The effect of the second assumption in the co-training algorithm

Class	Accuracy (%)		
	Lexical	Co-training	Self-training
Earn	95.09	96.61	95.30
Acq	93.76	95.21	93.73
Money-fx	96.15	97.12	96.15
Grain	95.48	95.51	95.51
Crude	97.00	97.67	97.00
Trade	97.79	98.42	97.79
Interest	97.18	97.67	97.18
Ship	98.15	98.58	98.15
Wheat	98.91	99.15	98.91
Corn	99.12	99.27	99.18

The experiments are performed on Reuters-21578 dataset.

The ‘lexical’ column in this table is obtained when only X is used. According to this table, self-training does not improve the accuracy compared to ‘lexical’. Though the accuracy is slightly improved over the class *earn*, *corn*, and *grain*, it does not improve for other classes. The performance is even deteriorated for the class *acq*. In the other hand, the co-training outperforms both ‘lexical’ and ‘self-training’ for all classes. Note that the only difference between co-training and self-training is that co-training uses two views rather than one. Thus, the second assumption of the co-training algorithm is closely related with its performance.

As a conclusion, the syntactic information is not completely independent from lexical information, but is reasonably independent compared with random splits of features. Since too many features are involved in document classification, it is impractical to find the completely independent features. According to the experimental results, syntactic information is relevant and independent from lexical information in this sense when it is used with traditional lexical information.

5. Conclusion

We proposed the co-trained support vector machines for document filtering. This method uses both the traditional lexical information and the syntactic information for the co-training algorithm and makes the algorithm applied to unstructured document classification, not only web page classification. In addition, with the algorithm we can incorporate ubiquitous unlabeled documents naturally to augment a small number of given labeled documents. The experiments on Reuters-21578 corpus and TREC-7 filtering documents show that using the syntactic information with the traditional lexical information improves the classification performance and the unlabeled documents are good resources to overcome the limited number of labeled documents. This is important because we can construct the classifier for large scale documents in high performance with just a small number of labeled documents. While the effectiveness of unlabeled documents is empirically proved, another problem is caused that we need a method to overcome the mislabeling during the co-training algorithm.

Acknowledgements

This research was supported by the Korean Ministry of Education under the BK21-IT Program and by the Korean Ministry of Science and Technology under NRL and BrainTech program.

References

Biber, D. (1995). *Dimensions of register variation: A cross-linguistic Comparison*. Cambridge University Press.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th annual conference of computational learning theory* (pp. 92–100).

Brill, E. (1992). A simple rule-based part-of-speech tagger. In *Proceedings of the 3rd conference on applied natural language processing* (pp. 152–155).

Buckley, C., & Salton, G. (1995). Optimizing of relevance feedback weights. In *Proceedings of the annual international ACM SIGIR conference on research and development in information retrieval* (pp. 351–357).

CoNLL (2000). *Shared task for computational natural language learning (CoNLL)*. Available: http://lcg-www.uia.ac.be/conll2000/chunking.

Drucker, H., Shahrary, B., & Gobbon, D. (2001). Relevance feedback using support vector machines. In *Proceeding of the 18th international conference on machine learning* (pp. 122–129).

Hull, D. (1998). The TREC-7 filtering track: description and analysis. In *Proceedings of the 7th text retrieval conference* (pp. 33–56).

Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the European conference on machine learning* (pp. 137–142).

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proceedings of the 16th international conference on machine learning* (pp. 200–209).

Kim, Y.-H., Hahn, S.-Y., & Zhang, B.-T. (2000). Text filtering by boosting Naive Bayes classifiers. In *Proceedings of the annual international ACM SIGIR conference on research and development in information retrieval* (pp. 168–175).

Kudo, T., & Matsumoto, Y. (2000). Use of support vector learning for chunk identification. In *Proceedings of CoNLL-2000 and LLL-2000* (pp. 142–144).

Leuski, A. (2000). Relevance and reinforcement in interactive browsing. In *Proceedings of 9th international conference on information and knowledge management* (pp. 119–126).

Lewis, D., & Jones, K. S. (1996). Natural language processing for information retrieval. *Communications of the ACM, 39*(1), 92–101.

Mitra, M., Buckley, C., Singhal, A., & Cardie, C. (1997). An analysis of statistical and syntactic phrases. In *Proceedings of RIAO-97* (pp. 200–214).

Nigam, K., & Ghani, R. (2000). Analyzing the effectiveness and applicability of co-training. In *Proceedings of the 9th international conference on information and knowledge management* (pp. 86–93).

Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. (2000). Learning to classify text from labeled and unlabeled documents. *Machine Learning, 39*, 1–32.

Park, S.-B., & Zhang, B.-T. (2002). A boosted maximum entropy model for learning text chunking. In *Proceedings of the 19th international conference on machine learning* (pp. 482–489).

Ponte, J., & Croft, W. (1998). A language modeling approach to information retrieval. In *Proceedings of the annual international ACM SIGIR conference on research and development in information retrieval* (pp. 275–281).

Rocchio, J. (1971). Relevance feedback in information retrieval. In *The SMART retrieval system: Experiments in automatic document processing* (pp. 313–323).

Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science, 41*, 288–297.

Scholkopf, B., Burges, C., & Smola, A. (1999). *Advances in kernel methods: Support vector machines*. MIT Press.

Stamatatos, E., Fakotakis, N., & Kokkinakis, G. (2000). Automatic text categorization in terms of genre and author. *Computational Linguistics, 26*(4), 471–496.

Turpin, A., & Moffat, A. (1999). Statistical phrases for vector-space information retrieval. In *Proceedings of the annual international ACM SIGIR conference on research and development in information retrieval* (pp. 309–310).

Zhai, C., & Lafferty, J. (2001). A study of smoothing methods for language models applied to Ad Hoc information retrieval. In *Proceedings of the annual international ACM SIGIR conference on research and development in information retrieval* (pp. 334–342).

Zhang, T., & Oles, F. (2000). A probability analysis on the value of unlabeled data for classification problems. In *Proceedings of the 17th international conference on machine learning* (pp. 1191–1198).