Journal of Machine Learning Research 1 (2002) 1-48 Submitted 5/02; Published 10/02

Distributional Word Clusters vs. Words for Text Categorization

Ron Bekkerman RONB@CS.TECHNION.AC.IL
Department of Computer Science,
Technion, Haifa 32000 Israel

Ran El-Yaniv RANI@CS.TECHNION.AC.IL
Department of Computer Science,
Technion, Haifa 32000 Israel

Naftali Tishby TISHBY@CS.HUJI.AC.IL
School of CS and Engineering
and Center for Neural Computation,
The Hebrew University, Jerusalem 91904 Israel

Yoad Winter WINTER@CS.TECHNION.AC.IL
Department of Computer Science,
Technion, Haifa 32000 Israel

Editor: Isabelle Guyon

Abstract

We study an approach to text categorization that combines distributional clustering of words and a Support Vector Machine (SVM) classifier. The word-cluster representation is computed using the recently introduced *Information Bottleneck* method, which generates a compact and efficient representation of documents. When combined with the classification power of the SVM, this method yields high performance in text categorization. We compare this technique with SVM-based categorization using the simple minded bag-of-words (BOW) representation. The comparison is performed over three known datasets. On one of these datasets (the 20 Newsgroups) the method that is based on word clusters significantly outperforms the word-based representation in terms of categorization accuracy or representation efficiency. On the two other sets (Reuters-21578 and WebKB) the word-based representation slightly outperforms the word-cluster representation. We investigate the potential reasons for this behavior.

1. Introduction

Text categorization is a fundamental task in Information Retrieval, and much knowledge in this domain has been accumulated in the past 25 years. The "standard" approach to text categorization has so far been using a document representation in a word-based 'input space', i.e. as a vector in some high (or trimmed) dimensional Euclidean space where each dimension corresponds to a word. This method relies on classification algorithms that are trained in a supervised learning manner. Since the early days of text categorization (see, e.g., Salton and McGill, 1983), the theory and practice of classifier design has significantly advanced, and several strong learning algorithms have emerged (see, e.g., Duda et al., 2000; Vapnik, 1998; Schapire and Singer, 2000). In contrast, despite numerous attempts to introduce more sophisticated techniques for document representation, like ones that are based on higher order word statistics (Caropreso et al., 2001) or NLP (Jacobs, 1992; Basili et al., 2000), the simple minded independent word-based representation, known as *bag-of-words (BOW)*, remained very popular. Indeed, to-date the best multi-class, multi-labeled categorization results for the well-known Reuters-21578 dataset are based on the BOW representation (Dumais et al., 1998; Joachims, 1998b; Weiss et al., 1999).

Nevertheless, attempts at devising more sophisticated text representation methods are not ceasing. In this paper we give further evidence to the usefulness of a statistical feature generation technique that is based on applying the recently introduced *Information Bottleneck (IB)* clustering framework (Tishby et al., 1999; Baker and McCallum, 1998; Slonim and Tishby, 2000, 2001). In this approach, IB clustering is used for generating document representation in a word *cluster* space (instead of word space), where each cluster is a distribution over classes of documents. We show that the combination of distributional representation with a Support Vector Machine (SVM) classifier (Vapnik, 1998; Cristianini and Shawe-Taylor, 2000) allows to achieve high performance in categorization of the well known 20 Newsgroups (20NG) dataset. This categorization of 20NG outperforms the strong algorithmic word-based setup of Dumais et al. (1998), which achieved one of the best reported categorization results for the 10 largest categories of the Reuters dataset.

These findings are perhaps not too surprising, since the use of distributional word clusters (instead of words) for representing documents has several advantages. First, word clustering implicitly reduces dimensionality because it groups various features (terms or words). In contrast, popular filter-based greedy approaches for feature selection such as Mutual Information, Information Gain and TFIDF (see, e.g., Yang and Pedersen, 1997) only consider each feature individually. Second, the clustering that is achieved by the IB method provides a good solution to the statistical sparseness problem that is prominent in the straightforward word-based (and even more so in n-gram-based) document representations. Thus, the clustering of words allows for extremely compact representations with minor information compromises that allow for the use of strong but computationally intensive classifiers.

Despite advantages we found for distributional clustering in categorizing the 20NG dataset, it does not show improvement of accuracy over BOW-based categorization when used over the Reuters dataset (ModApte split) and over a subset of the WebKB dataset. We analyze this phenomenon and argue that the categories of documents in Reuters and WebKB are less "complex" than the categories of 20NG in the sense that they can almost be "optimally" categorized using a small number of keywords. This is not the case for 20NG.

The rest of this paper is organized as follows. Section 2 describes the problem of text categorization, some approaches to its solution, including feature selection and generation. Section 3 discusses related results. Section 4 briefly presents the algorithmic components we use, which involve a novel combination of existing techniques for feature selection, clustering and categorization. Section 5 briefly describes the datasets we use and their textual preprocessing in our experiments. Section 6 presents our experimental setup and Section 7 gives a detailed description of the results. Section 8 discusses these results. Section 9 details the computational efforts that were required for these experiments. Finally, in Section 10 we conclude and outline some open questions.

2. Preliminaries: Text Categorization and Feature Selection/Generation

In this section we state the text categorization problem that will be addressed, discuss its components, and introduce terminology that will facilitate the discussions in the rest of the paper.

The goal in the machine learning approach to *text categorization* is to devise a learning algorithm that can generate a classifier capable of categorizing (or classifying) text documents according to a number of predefined categories (or classes). This task has been mostly considered within a supervised learning scheme (see, e.g., Duda and Hart, 1973; Sebastiani, 2002) but it can also be considered within an unsupervised and semi-supervised learning setups (see, e.g., Slonim and Tishby, 2001; Slonim et al., 2002; El-Yaniv and Souroujon, 2001). This paper focuses on the more common supervised learning approach to text categorization.

In its simplest form, the text categorization problem can be formulated as follows. We are given a training set $\mathcal{D}_{train} = \{(d_1, \ell_1), \ldots, (d_n, \ell_n)\}$ of labeled text documents where each document d_i belongs to a document set $\mathcal{D}$ and the label $\ell_i = \ell_i(d_i)$ of d_i is within a predefined set of categories $\mathcal{C} = \{c_1, \ldots, c_m\}$. We assume that the initial representation of a document is a sequence of *words* (or *terms*).[1] The goal in text categorization is to devise a learning algorithm that given the training set $\mathcal{D}_{train}$ as input will generate a classifier (or a hypothesis) $h : \mathcal{D} \rightarrow \mathcal{C}$ that will be able to accurately classify unseen documents from $\mathcal{D}$.

While this basic text categorization problem falls within the generic field of *supervised classifier learning*, the growing interest in text categorization applications motivated dedicated research efforts on text categorization. There are some distinctive properties of text categorization that justify its dedicated study within the much larger field of supervised learning:

- *High Dimensionality.* A faithful representation of a document that is based on a sequence of words implies high dimensionality since the number of distinct words in $\mathcal{D}$ can be very large even if $\mathcal{D}$ is of moderate size.

- *Statistical sparseness.* Although the number of possible features (words) can be very large, each single document usually includes only a small fraction of them.

- *Domain knowledge.* Since documents are given in natural language, it appears that linguistic studies should help in discovering their inner structure, and therefore in understanding their meaning (see, e.g., Manning and Schütze, 1999). In addition, other statistical properties of texts (e.g. their adherence to Zipf's law) can be employed, (see, e.g., Joachims, 2001).

- *Multi-labeling.* The basic text categorization problem, as presented above, has various common extensions. In the *multi-labeled* version of text categorization, a document can belong to several classes simultaneously. That is, both $h(d)$ and $\ell(d)$ can be sets

1. Lower-level or more sophisticated initial representations can be considered. For example, a document can be given as a sequence of characters rather than words (see, e.g., Lodhi et al., 2000; Raskutti et al., 2001). More involved initial representations are optional when documents are initially represented using some markup language (e.g. XML), in which case a document is represented hierarchically and this representation may include additional formatting cues and semantic tags.

of categories rather than single categories. In case where each document has only a single label we say that the categorization is *uni-labeled*.

Although documents are initially represented as sequences of words, it is common in text categorization to consider a *bag-of-words (BOW)* representation, where a document is represented as a *multi-set* of words and word order is ignored.[2] This is also the initial representation we adopt in this paper.

Text categorization problems are most often multi-class rather than binary; that is, the cardinality m of the set of categories is larger than 2. While there are many classifier learning algorithms that can naturally handle multi-class problems (e.g. naive Bayes, Rennie, 2001), there are others that naturally handle only 2-class (or binary) problems (e.g. SVM, Cristianini and Shawe-Taylor, 2000, and in general, classifiers based on linear separation in feature or kernel space). A popular method for decomposing a multi-class classification problem into binary problems is the *one-against-all* (also called *one-per-class* or *max win*) strategy. In this method, the m-class problem is decomposed into m binary problems where in the ith binary problem a binary classifier is trained to distinguish between the ith class and the union of the other classes. This one-against-all decomposition is a special case of the general *Error-Correcting Output Coding (ECOC)* scheme for multi-class decomposition, which includes, as special cases, other decomposition schemes such as *all pairs*. A detailed description of ECOC is beyond the scope of this paper. The reader is referred to Dietterich and Bakiri (1995); Allwein et al. (2000) for details. For compatibility with other relevant studies, in this work we consider the simple minded one-against-all decomposition method.

2.1 Dimensionality Reduction, Feature Selection and Generation

The design of learning algorithms for text categorization has been usually following the classical approach in pattern recognition, where data instances (i.e. documents) first undergo a transformation of dimensionality reduction. Then a classifier learning algorithm is applied to the low-dimensionality representations (see, e.g., Duda and Hart, 1973). This transformation is of course also performed prior to applying the learned classifier to unseen instances. The incentives in using dimensionality reduction techniques are to improve classification quality (via noise reduction) and to reduce the computational complexity of the learning algorithm and of the application of the classifier to unseen documents.

There are special dimensionality reduction techniques for documents in natural language. Three of the most common techniques are *stemming*, *stop words* filtering and filtering of words with low frequency. Stemming is pruning of word suffices that play only grammatical role. Stop words are words that have only connecting function in sentences, such as prepositions, articles, etc. These words are usually very frequent, but their contribution to text categorization is often negligible. One important incentive for stemming is to reduce the statistical sparseness, which is achieved when occurrences of different words (e.g. 'walk' and 'walking') are mapped into one word ('walk') and counted together. Nevertheless, despite the variance reduction that can be achieved, stemming can clearly increase bias and the overall effect of stemming depends on the dataset at hand. An example for the biased effect

2. Dumais et al. (1998) use a BOW representation of documents as *set* of words, where each word that appears in the documents is counted only once. See Section 3.

of stemming is the mapping of an unambiguous word such as “mining” into the ambiguous word “mine”.

Other dimensionality reduction techniques typically fall into two basic schemes:

- *Feature (subset) selection* (or *feature reduction*): These techniques attempt to select the subset of features (e.g. words in text categorization) that are most useful for the categorization task. After the selection of a suitable subset, the reduced representation of a document is computed by projecting the documents over the selected words.

- *Feature generation* (or *feature extraction*): New features, which are not necessarily words, are sought for representation. Usually, the new features are synthesized from the original set of features.

Feature subset selection and feature generation are computationally challenging. A good set of (either selected or generated) features should be optimized to the text categorization goal. Intuitively, the ideal optimization criterion is an algorithmic procedure that depends on the classifier learning scheme being used (Kohavi and John, 1998). The use of the classifier as the optimization criterion is called the “wrapper approach” (for feature selection). By contrast, in the “filter approach”, the subset of features is chosen based on the data itself, regardless of the classifier. Such filters are constructed by defining a cost (or benefit) function for features or for subsets of features. At the outset, the wrapper approach is clearly computationally more demanding than the filter approach and as far as we know, wrapper-based feature selection (or generation) are rarely used in practice. In both cases (either wrapper or filter) the selection (or generation) process suffers from combinatorial explosion and under any reasonable optimality criterion, the computation of an optimal representation appears to be intractable.[3]

Whether it is selection or generation one can consider either *unsupervised selection* (resp. *generation*) or a *supervised selection* (resp. *generation*). In an unsupervised selection a single subset of features is selected for representing documents of all categories and document labels are not used in the computation. In a supervised selection, document labels are utilized in the computation. In the case of supervised selection one can distinguish between two selection methods. One method generates a “local” feature set for each category or for subsets of categories (e.g. depending on the multi-class decomposition technique used). Another kind of methods generates the same global set of features for all categories. In this paper we focus on supervised feature generation methods that generate a global set of features for all the categories.

So far, practical applications of dimensionality reduction techniques follow for the most part a heuristic approach both in setting the optimization criterion and in selecting (or generating) the set of features. In practice, the most commonly used methods for feature selection are simple filter-based indices that compute the contribution of each feature independently and then greedily collect a set of the most highly ranked features (see, e.g., Yang and Pedersen, 1997). Common approaches for feature generation attempt to combine the initially given features using some formal grammar or clustering (see, e.g., Markovitch and Rosenstein, 2002; Sebastiani, 2002).

3. Wang et al. (1999) present an axiomatization of an optimality criterion for feature selection and show that using the resulting criterion, the problem of identifying the optimal subset of features is NP-hard.

2.2 Evaluating the Performance of Text Categorization

For the most part, evaluation of text categorization performance has been done experimentally.[4] The two main issues under consideration are computational *efficiency* and categorization *effectiveness*.

The computational efficiency of the feature selection (generation) algorithm, of the classifier learning algorithm and of the constructed classifier can be of major importance when considering applications that need to categorize a large number of documents into many categories. For measuring the performance of applications that require online categorization we need to distinguish between *training time complexity* and *categorization time complexity*. With the current processing (and memory) speed and the available memory sizes, an algorithm with sub-quadratic (time and space) complexities (in the number of documents) can be considered efficient even for very large applications. Algorithms of quadratic complexity can introduce a bottleneck in some applications.

We measure the empirical effectiveness of multi-labeled text categorization in terms of the classical information retrieval parameters of "precision" and "recall" (Baeza-Yates and Ribeiro-Neto, 1999). Consider a multi-labeled categorization problem with m classes, $\mathcal{C} = \{c_1, \ldots, c_m\}$. Let h be a classifier that was trained for this problem. For a document d, let $h(d) \subseteq \mathcal{C}$ be the set of categories designated by h for d. Let $\ell(d) \subseteq \mathcal{C}$ be true categories of d.

Let $\mathcal{D}_{test} \subset \mathcal{D}$ be a *test set* of "unseen" documents that were not used in the construction of h. For each category c_i, define the following quantities:

$$
\begin{aligned}
TP_i &= \sum_{d \in \mathcal{D}_{test}} I\left[c_i \in \ell(d) \wedge c_i \in h(d)\right], \\
TN_i &= \sum_{d \in \mathcal{D}_{test}} I\left[c_i \in \ell(d) \wedge c_i \notin h(d)\right], \\
FP_i &= \sum_{d \in \mathcal{D}_{test}} I\left[c_i \notin \ell(d) \wedge c_i \in h(d)\right], \\
FN_i &= \sum_{d \in \mathcal{D}_{test}} I\left[c_i \notin \ell(d) \wedge c_i \notin h(d)\right]
\end{aligned}
$$

where $I[\cdot]$ is the indicator function. For example, FP_i (the "false positives" with respect to c_i) is the number of documents categorized by h into c_i whose true set of labels does not include c_i, etc. For each category c_i we now define the precision $P_i = P_i(h)$ of h and the recall $R_i = R_i(h)$ with respect to c_i as

$$
\begin{aligned}
P_i &= \frac{TP_i}{TP_i + TN_i} \\
R_i &= \frac{TP_i}{TP_i + FP_i}.
\end{aligned}
$$

4. While there are numerous attempts to study statistical properties and generative models for text, we are not familiar with explicit theoretical studies on the generalization abilities of learning algorithms for text classifiers, with the exception of a recent paper by Joachims (2001), which attempts to characterize conditions for the effectiveness of categorization using a Support Vector Machine.

The overall *micro-averaged precision* $P = P(h)$ and *recall* $R = R(h)$ of h is a weighted average of the individual precisions and recalls (weighted with respect to the sizes of the test set categories).[5]

$$\begin{aligned} P &= \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m}(TP_i + TN_i)} \\ R &= \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m}(TP_i + FP_i)}. \end{aligned}$$

There is a natural tradeoff between precision and recall. For example, in the extreme case where for each document $d \in \mathcal{D}_{test}$, $h(d) = C$ we get maximal recall but minimal precision, etc. If we are interested in a single quantity that measures the performance of the classifier there are a number of sensible options. The following two are often used:

- *F-measure:* The harmonic mean of precision and recall; that is $F = F(h) = \frac{2}{1/P(h)+1/R(h)}$.
- *Break-Even Point (BEP):* A flexible classifier provides the means to control the trade-off between precision and recall. For such classifiers, the value of P (and R) satisfying $P = R$ is called the break-even point. Usually it is not so easy to achieve the exact break-even point and one attempts to identify "sufficiently close" precision and recall values and use the *interpolated break-even point*, which is the (arithmetic) mean of P and R.

Note that the arithmetic mean $(P + R)/2$ is not a reliable measure when P and R are far apart. For instance, in the extreme case of a trivial classifier that achieves $P = 0$ and $R = 1$ we get an interpolated break-even point of 0.5. The harmonic mean in this case approaches 0.

The above performance measures concern multi-labeled categorization. In a uni-labeled categorization the accepted performance measure is *accuracy*, defined to be the percentage of correctly labeled documents of $\mathcal{D}_{test}$. Specifically, assuming that both $h(d)$ and $\ell(d)$ are singletons (i.e. uni-labeling), the accuracy $Acc(h)$ of h is:

$$Acc(h) = \frac{1}{|\mathcal{D}_{test}|} \sum_{d \in \mathcal{D}_{test}} I[h(d) = \ell(d)]. \tag{1}$$

Is it not hard to see that in this case the accuracy equals the precision and recall (and the break-even point).[6]

3. Related results

In this section we briefly overview results which are most relevant for the present work. Thus, we limit the discussion to related feature selection and generation techniques and

5. Some authors advocate the use of *macro-averaged* precision and recall, which are *non-weighted* averages of the individual categories precisions and recalls.
6. Assuming a uni-labeled setting the numerator in (1) is $\sum_{d \in \mathcal{D}_{test}} I[h(d) = \ell(d)] = \sum_{i=1}^{k} TP_i$, and the denominator is $|\mathcal{D}_{test}| = \sum_{d \in \mathcal{D}_{test}} (I[h(d) = \ell(d)] + I[h(d) \neq \ell(d)]) = \sum_{i=1}^{k}(TP_i + TN_i) = \sum_{i=1}^{k}(TP_i + FP_i)$.

best known categorization results over the corpora we consider (Reuters-21578, the 20 Newsgroups and WebKB). For more comprehensive surveys on text categorization the reader is referred to Sebastiani (2002); Singer and Lewis (2000) and references therein. We start with a discussion of feature selection and generation techniques.

As noted in Section 2.1 the selection of an "optimal" subset of features suffers from combinatorial explosion. Consequently, many authors use simple filter-based greedy approaches where each feature is given a score and a subset of the top-scored features is taken. However, more sophisticated methods that attempt to consider the utility of a set of features (including the interaction between features) have also been studied (see, e.g., Koller and Sahami, 1996). Many of the greedy methods are supervised and rank the features according to their contribution to separation between categories, and then extract the most "useful" features.

Yang and Pedersen (1997) empirically compare five filter-based (supervised and unsupervised) greedy methods for feature selection.[7] They conclude that among the five methods tested the χ^2 and Mutual Information (MI, see Equation (2) below) feature indices are the most effective.

Dumais et al. (1998) report on experiments with multi-labeled categorization of the Reuters dataset. Over a BOW binary representation (where each word receives a count of 1 if it occurs once or more in a document and 0 otherwise) they applied the Mutual Information index for feature selection. Specifically, let $X_c \in \{0,1\}$ be a binary random variable denoting the event that a random document belongs (or not) to category c. Similarly, let $X_w \in \{0,1\}$ be a random variable denoting the event that the word w occurred in a random document. The Mutual Information between X_c and X_w is

$$I(X_c, X_w) = \sum_{X_c, X_w \in \{0,1\}} P(X_c, X_w) \log \frac{P(X_c, X_w)}{P(X_c)P(X_w)}. \tag{2}$$

Note that when estimating $I(X_c, X_w)$ from a training set sample, we estimate $P(X_c, X_w)$, $P(X_c)$ and $P(X_w)$ using their empirical estimates. For each category c, all the words are sorted according to decreasing value of their Mutual Information with respect to c and the first k words are kept. Thus, for each category there is a specialized representation using the most discriminative words for the category.[8] For each category c after the selection of the top $k(c)$ words, each document is represented using only the $k(c)$ most discriminating words for c.

Dumais *et al.* show that together with a support vector machine (SVM) classifier this method yields a 92.0% break-even point (BEP) on the 10 largest categories in the Reuters dataset. As far as we know this is the best multi-labeled categorization result of the (10

7. The feature indices they considered were the unsupervised *document frequency*, and the supervised *information gain*, *Mutual Information*, χ^2*-test* and *term strength*. Note that there is a confusion between the terms "Information Gain" and "Mutual Information". Specifically, the index that is called in this paper *Information Gain* is in fact the standard Mutual Information as defined for example in Cover and Thomas (1991). Throughout this paper, we refer to this index (as defined in Eq. (2)) as *Mutual Information*.
8. Note that this specialized representation for each category assumes a decomposition of the multi-category categorization problem into binary problems such that each category has a special classifier (see Section 2).

largest categories of) Reuters dataset. Therefore, in this paper we adopt the SVM classifier with MI feature selection as a baseline for handling BOW-based categorization.

Rather than reducing the number of features (e.g. by filtering out low-scored features), *feature generation* techniques attempt to construct new features from existing ones. A general framework for constructing new features is to combine existing features using logical operators on existing features. There are two common applications of this approach. The first one combines features using only disjunctions. In this approach one groups features into subsets and consider each such subset as a new feature. Any occurrence of a member of a subset is then considered as occurrence of the feature. *Word clustering* belongs to this family of methods. A second approach groups features using only conjunctions, for example, by grouping consequent or close (in proximity) words into phrases. The use of *n-grams* is a common method in this family. Disjunction-based methods for feature generation are quite radically different than conjunction-based methods and they achieve different goals. One crucial difference between these methods is that disjunction methods can decrease and somewhat overcome statistical sparseness while conjunction methods can only increase it. Thus, disjunction methods can decrease variance. On the other hand, conjunction methods can sometimes decrease bias. There are quite a few "soft" variants of these two extreme techniques. For instance Latent Semantic Indexing (LSI) (Deerwester et al., 1990) can be viewed as a soft, weighted disjunction of words.

Another type of specialized feature generation for text categorization concerns the structure of the categorized texts. For instance, features such as document titles or section headings are often more informative than other features. A number of experimental studies deal with feature extraction using such structural information. In the case of web pages (where documents are encoded using a markup language such as HTML) this structural information can be easily identified and used (see, e.g., Fürnkranz, 1999; Ghani et al., 2001).

Caropreso et al. (2001) experiment with n-grams for text categorization of the Reuters dataset. They define an n-gram as an alphabetically ordered sequence of n stems of consecutive words in a sentence (after stop words were removed). As features the authors use both unigrams and bigrams. They extract the top-scored features using various feature selection indices including Mutual Information. Their results indicate that in general bigrams can better predict categories than unigrams. However, despite the fact that bigrams are the majority of the top-scored features, the addition of bigrams does not yield significant improvement of the categorization results.[9] Specifically, in 20 of the 48 reported experiments a certain increase in accuracy is observed, while in 28 others the accuracy decrease, sometimes quite sharply.

Baker and McCallum (1998) apply the distributional clustering scheme of Pereira et al. (1993) (see Section 4) for clustering words that are represented as distributions over categories of the documents. Given a set of categories $\mathcal{C} = \{c_i\}_{i=1}^m$, a distribution of a word w over the categories is $\{P(c_i|w)\}_{i=1}^m$. Then the words (represented as distributions) are clustered using an agglomerative clustering algorithm. Using a naive Bayes classifier the authors examine uni-labeled categorization accuracy over the 20NG dataset and reported an 85.7% accuracy. They also compare this representation to other feature selection and

9. These authors use the Rocchio classifier (Rocchio, 1971).

generation techniques such as Latent Semantic Indexing (see, e.g., Deerwester et al., 1990), the above Mutual Information index and the Markov "blankets" feature selection technique of Koller and Sahami (1996). The authors conclude that categorization that is based on word clusters is only slightly less accurate than the other methods, this is while keeping the word-cluster representation significantly more compact.

Tishby et al. (1999) experiment with a similar word clustering approach, using the *Information Bottleneck (IB)* method (see Section 4.1). Slonim and Tishby (2000) explore the properties of this word cluster representation and motivate it within the more general IB method. In Slonim and Tishby (2001), the authors show that categorization with a representation that is based on IB-clustering of words can improve the categorization accuracy that is achieved by a BOW representation whenever the training set is small (about 10 documents per category). Specifically, using a Naive Bayes classifier on a dataset consisting of 10 categories of 20NG, they observe 18.4% improvement in accuracy over a categorization that is based on BOW.

Many algorithms for classifier learning have been tested in the text categorization domain. Here we mainly focus on SVM categorization results as well as some other results relevant for the corpora we use. Some recent work provide strong evidence that SVM is among the best classifiers for text categorization.[10] Yang and Liu (1999) test five classifiers[11] and observe that SVM is among the classifiers that show the best performance on the Reuters dataset, with both large and small training sets. Over the ModApte split of Reuters their result is 86.0% of micro-averaged F-measure.

Joachims (1998b) uses an SVM classifier for a multi-labeled categorization of Reuters without feature selection, and achieved a break-even point of 86.4%. In Joachims (1997), he also investigates uni-labeled categorization of the 20NG dataset, and applies the Rocchio classifier (Rocchio, 1971) over TFIDF weighted (see, e.g., Manning and Schütze, 1999) BOW representation that is reduced using the Mutual Information index. He obtains 90.3% accuracy, which is, to our knowledge, the best published accuracy to-date of a uni-labeled categorization of the 20NG dataset.

As mentioned earlier, Dumais et al. (1998) study the categorization performance of some classification techniques, including SVM.[12] Their conclusion is that the SVM is superior to the other methods tested on the Reuters dataset (ModApte split). In particular, the SVM (together with Mutual Information index for feature selection) achieve a 92.0% BEP on the 10 largest categories of Reuters. To-date this is the best known result for this set.

Schapire and Singer (1998) apply a boosting algorithm based on the *AdaBoost* classifier (with one-level decision trees – also known as *decision stamps* – as the base classifiers) for the text categorization The resulting algorithm, called Boostexter, achieves 86.0% BEP on all the categories of Reuters (ModApte split).

Weiss et al. (1999) also employ boosting (using decision trees as the base classifiers and a powerful adaptive resampling method) and on Reuters (ModApte split) they obtain 87.8% of break-even point on the largest 95 categories (each having at least 2 training examples).

10. Joachims (2001) provides a theoretical account of the suitability of SVM for text categorization.
11. Specifically, they test SVM, kNN, three-layered Neural Network (with a hidden layer consisting of empirically optimized number of nodes), LLSF (see Yang and Chute, 1992) and Naive Bayes.
12. The other techniques are a variant of the Rocchio classifier, Decision Trees, Naive Bayes and Bayesian Networks.

To our knowledge this is the best result that has been achieved on (almost) the entire Reuters dataset.

Table 1 summarizes the results that were discussed in this section.

Authors	***Dataset***	***Feature Selection or Generation***	***Classifier***	***Main Result***	***Comments***
Caropreso et al. (2001)	Reuters	MI for unigrams & bigrams	Rocchio	Bigrams do not help	
Dumais et al. (1998)	Reuters	MI	SVM, Rocchio decision trees, Naive Bayes	SVM performs best: 92.0% BEP on 10 largest categories	Our baseline for Reuters (Best on 10 categories)
Joachims (1998b)	Reuters	none	SVM	86.4% BEP	
Schapire and Singer (1998)	Reuters	none	AdaBoost	86% BEP	
Weiss et al. (1999)	Reuters	none	Boosting with multiple decision trees	87.8% BEP	Best on 95 categories of Reuters
Yang and Liu (1999)	Reuters	none	SVM, kNN, LLSF, NB	86% F-measure (SVM)	95 categories
Joachims (1997)	20NG	MI	Rocchio with TFIDF	90.3% accuracy (uni-labeled)	Our baseline for 20NG
Baker and McCallum (1998)	20NG	Distrib. clustering	Naive Bayes	85.7% accuracy (uni-labeled)	
Slonim and Tishby (2000)	10 cate- gories of 20NG	Informa- tion Bottleneck	Naive Bayes	Up to 18.4 improvement on small training sets	
Joachims (1999)	WebKB	none	SVM	94.2% - "course" 79.0% - "faculty" 53.3% - "project" 89.9% - "student"	Our baseline for WebKB
Nigam et al. (1998)	WebKB	MI	Naive Bayes	82% accuracy	

Table 1: Summary of related results.

4. Methods and algorithms

The text categorization scheme we study is based on two components: on a representation scheme of documents as "distributional clusters" of words, and on a Support Vector Machine (SVM) classifier learning algorithm. In this section we describe both components. Since SVMs are rather familiar and thoroughly covered in the literature, our main focus in this section is on the Information Bottleneck method and distributional clustering.

4.1 Information Bottleneck and distributional clustering

Data clustering is a challenging task in information processing and pattern recognition. The challenge is both conceptual and computational. Intuitively, when we attempt to cluster a dataset, our goal is to partition it into subsets such that points in the same subset are more "similar" to each other than to points in other subsets. Common clustering algorithms depend on choosing a similarity measure between data points and a "correct" clustering result can be dependent on an appropriate choice of a similarity measure. However, the choice of a "correct" measure is an ill-defined task without a particular application at hand. For instance, consider a hypothetical dataset containing articles by each of two authors, so that half of the articles authored by each author discusses one topic, and the other half discusses another topic. There are two possible dichotomies of the data which could yield two different bi-partitions: according to the topic or according to the writing style. When asked to cluster this set into two sub-clusters, one cannot successfully achieve the task without knowing the goal. Therefore, without a suitable target at hand and a principled method for choosing a similarity measure suitable for the target, it can be meaningless to interpret clustering results.

The *Information Bottleneck (IB)* method of Tishby, Pereira, and Bialek (1999) is a rather new framework that can sometimes provide an elegant solution to this problematic "metric selection" aspect of data clustering. Consider a dataset given by i.i.d. observations of a random variable X. Informally, the IB method aims to construct a relevant encoding of the random variable X by partitioning X into domains that preserve as much as possible the Mutual Information between X and another "relevance" variable, Y. The relation between X and Y is made known via i.i.d. observations from the joint distribution $P(X,Y)$. Denote the desired partition (clustering) of X by $\tilde{X}$. We determine $\tilde{X}$ by solving the following variational problem: *Maximize the Mutual Information $I(\tilde{X},Y)$ with respect to the partition $P(\tilde{X}|X)$, under a minimizing constraint on $I(\tilde{X},X)$*. In particular, the Information Bottleneck method considers the following optimization problem: Maximize

$$I(\tilde{X},Y) - \beta I(\tilde{X},X),$$

over the conditional $P(\tilde{X}|X)$ where the parameter β determines the allowed amount of reduction in information that $\tilde{X}$ bears on X. Namely, we attempt to find the optimal tradeoff between the minimal partition of X and the maximum preserved information on Y. In Tishby et al. (1999), it is shown that a solution for this optimization problem is characterized by

$$P(\tilde{X}|X) = \frac{P(\tilde{X})}{Z(\beta,X)} \exp\left[-\beta \sum_Y P(Y|X) \ln\left(\frac{P(Y|X)}{P(Y|\tilde{X})}\right)\right], \tag{3}$$

where $Z(\beta,X)$ is a normalization factor, and $P(Y|\tilde{X})$ in the exponential is defined implicitly, through Bayes' rule, in terms of the partition (assignment) rules $P(\tilde{X}|X)$, $P(Y|\tilde{X}) = \frac{1}{P(\tilde{X})}\sum_X P(Y|X)P(\tilde{X}|X)P(X)$ (see Tishby et al., 1999, for details). The parameter β is a Lagrange multiplier introduced for the constrained information, but using a thermodynamical analogy β can also be viewed as an inverse temperature, and can be utilized as an *annealing* parameter to choose a desired cluster resolution.

Before we continue and present the IB clustering algorithm in the next section, we note on the contextual background of the IB method and its connection to "distributional clustering". Pereira, Tishby, and Lee (1993) introduced "distributional clustering" for distributions of verb-object pairs. Their algorithm clustered nouns represented as distributions over co-located verbs (or verbs represented as distributions over co-located nouns). This clustering routine aimed at minimizing the average distributional similarity (in terms of the Kullback-Leibler divergence, Cover and Thomas, 1991) between the conditional $P(verb|noun)$ and the noun centroid distributions (i.e. these centroids are also distributions over verbs). It turned out that this routine is a special case of the more general IB framework. IB clustering has since derived a variety of effective clustering and categorization routines (see, e.g., Slonim and Tishby, 2001; Bekkerman et al., 2001; Slonim et al., 2001) and has interesting extensions (Friedman et al., 2001).

4.2 Distributional clustering via deterministic annealing

A solution to the IB optimization satisfies the following self-consistent equations.

$$P(\tilde{X}|X) = \frac{P(\tilde{X})}{Z(\beta, X)} \exp\left[-\beta \sum_{Y} P(Y|X) \ln\left(\frac{P(Y|X)}{P(Y|\tilde{X})}\right)\right]; \quad (4)$$

$$P(\tilde{X}) = \sum_{X} P(X) P(\tilde{X}|X); \quad (5)$$

$$P(Y|\tilde{X}) = \sum_{Y} P(Y|X) P(X|\tilde{X}). \quad (6)$$

In Tishby et al. (1999), it is shown that a solution can be obtained by starting with an arbitrary solution and then iterating the equations. For any value of β this procedure is guaranteed to converge.[13] Lower values of the β parameter (high "temperatures") correspond to poor distributional resolution (i.e. fewer clusters) and higher values of β (low "temperatures") correspond to higher resolution (i.e. more clusters).

We use a hierarchical top-down clustering procedure for recovering the distributional IB clusters. A pseudo-code of the algorithm is given in Algorithm 1.[14] Starting with one cluster (very small β) that contains all the data we incrementally achieve the desired number of clusters by performing a process that consists of *annealing stages*. At each annealing stage we increment β and attempt to split existing clusters. This is done by creating (for each centroid) a new "ghost" centroid at some random small distance from the original centroid. We then attempt to cluster the points (distributions) using all (original and ghost) centroids by iterating the above IB self-consisting equations, similar to the *Expectation-Maximization (EM)* algorithm (Dempster et al., 1977). During these iterations the centroids are adjusted to their (locally) optimal positions and (depending on the annealing increment of β) some "ghost" centroids can merge back with their centroid sources.

13. This procedure is analogous to the Blahut-Arimoto algorithm in Information Theory (Cover and Thomas, 1991).
14. A similar annealing procedure procedure, known as *deterministic annealing*, was introduced in the context of clustering by Rose et. al. (Rose, 1998).

Require:
$P(X, Y)$ - Observed joint distribution of two random variables X and Y
k - desired number of centroids
β_{min}, β_{max} - minimal / maximal values of β
$\nu > 1$ - annealing rate
$\delta_{conv} > 0$ - convergence threshold, $\delta_{merge} > 0$ - merging threshold
Ensure:
Cluster centroids, given by $\{P(Y|\tilde{x}_i\}_{i=1}^k$
Cluster assignment probabilities, given by $P(\tilde{X}|X)$

Initiate $\beta \leftarrow \beta_{min}$ - current β parameter
Initiate $k_{curr} \leftarrow 1$ - current number of centroids
repeat
{ *1. "EM"-like iteration:* }
Compute $P(\tilde{X}|X)$, $P(\tilde{X})$ and $P(Y|\tilde{X})$ using Equations (4), (5) and (6) respectively
repeat
Let $P_{old}(\tilde{X}|X) \leftarrow P(\tilde{X}|X)$
Compute new values for $P(\tilde{X}|X)$, $P(\tilde{X})$ and $P(Y|\tilde{X})$ using (4), (5) and (6)
until for each x: $\|P(\tilde{X}|X = x) - P_{old}(\tilde{X}|X = x)\| < \delta_{conv}$
{ *2. Merging:* }
for all $i, j \in [1, k_{curr}]$ s.t. $i \neq j$ and $\|P(Y|X = \tilde{x}_i) - P(Y|X = \tilde{x}_j)\| < \delta_{merge}$ **do**
Merge $\tilde{x}_i$ and $\tilde{x}_j$ and decrement k_{curr}
end for
{ *3. Centroid ghosting:* }
Let $k_{curr} \leftarrow 2k_{curr}$, $\beta \leftarrow \nu\beta$
until $k_{curr} \geqslant k$ or $\beta \geqslant \beta_{max}$
If $k_{curr} > k$ then merge $k_{curr} - k$ closest centroids (each to its closest centroid neighbor)

Algorithm 1: Information Bottleneck Distributional Clustering

In this scheme (as well as in the similar deterministic annealing algorithm of Rose, 1998), one has to use an appropriate annealing rate in order to identify *phase transitions* which correspond to cluster splits.

An alternative agglomerative (bottom-up) hard-clustering algorithm was developed by Slonim and Tishby (2000). This algorithm generates hard clustering of the data and thus approximates the above IB clustering procedure. Note that the time complexity of this algorithm is $O(n^2)$ where n is the number of data points (distributions) to be clustered (see also an approximate faster agglomerative procedure in Baker and McCallum, 1998).

The application of the IB clustering algorithm in our context is straightforward. The variable X represents words that appear in training documents. The variable Y represents class labels and thus, the joint distribution $P(X, Y)$ is characterized by pairs (w, c) where w is a word and c is the class label of the document where w appears. Starting with the observed conditionals $\{P(Y = c|X = w)\}_c$ (giving for each word w its class distribution) we cluster these distributions using Algorithm 1. For a pre-specified number of clusters k the output of Algorithm 1 is: (i) k centroids, given by the distributions $\{P(\tilde{X} = \tilde{w}|X = w)\}_{\tilde{w}}$ for each word w where $\tilde{w}$ are the word centroids (i.e. there are k such word centroids which represent k word clusters); (ii) Cluster assignment probabilities given by $P(\tilde{X}|X)$. Thus, each word w may (partially) belong to all k clusters and the association weight of w to the cluster represented by the centroid $\tilde{w}$ is $P(\tilde{w}|w)$.

Word	***Clustering to 300 clusters***	***Clustering to 50 clusters***
at	$\tilde{w}_{97}$ (1.0)	$\tilde{w}_{44}$ (0.996655) $\tilde{w}_{21}$ (0.00334415)
ate	$\tilde{w}_{205}$ (1.0)	$\tilde{w}_{42}$ (1.0)
atheism	$\tilde{w}_{56}$ (1.0)	$\tilde{w}_{3}$ (1.0)
atheist	$\tilde{w}_{76}$ (1.0)	$\tilde{w}_{3}$ (1.0)
atheistic	$\tilde{w}_{56}$ (1.0)	$\tilde{w}_{3}$ (1.0)
atheists	$\tilde{w}_{76}$ (1.0)	$\tilde{w}_{3}$ (1.0)
atmosphere	$\tilde{w}_{200}$ (1.0)	$\tilde{w}_{33}$ (1.0)
atmospheric	$\tilde{w}_{200}$ (1.0)	$\tilde{w}_{33}$ (1.0)
atom	$\tilde{w}_{92}$ (1.0)	$\tilde{w}_{13}$ (1.0)
atomic	$\tilde{w}_{92}$ (1.0)	$\tilde{w}_{35}$ (1.0)
atoms	$\tilde{w}_{92}$ (1.0)	$\tilde{w}_{13}$ (1.0)
atone	$\tilde{w}_{221}$ (1.0)	$\tilde{w}_{14}$ (0.998825) $\tilde{w}_{13}$ (0.00117386)
atonement	$\tilde{w}_{221}$ (1.0)	$\tilde{w}_{12}$ (1.0)
atrocities	$\tilde{w}_{4}$ (0.99977) $\tilde{w}_{1}$ (0.000222839)	$\tilde{w}_{5}$ (1.0)
attached	$\tilde{w}_{251}$ (1.0)	$\tilde{w}_{30}$ (1.0)
attack	$\tilde{w}_{71}$ (1.0)	$\tilde{w}_{28}$ (1.0)
attacked	$\tilde{w}_{4}$ (0.99977) $\tilde{w}_{1}$ (0.000222839)	$\tilde{w}_{10}$ (1.0)
attacker	$\tilde{w}_{103}$ (1.0)	$\tilde{w}_{28}$ (1.0)
attackers	$\tilde{w}_{4}$ (0.99977) $\tilde{w}_{1}$ (0.000222839)	$\tilde{w}_{5}$ (1.0)
attacking	$\tilde{w}_{4}$ (0.99977) $\tilde{w}_{1}$ (0.000222839)	$\tilde{w}_{10}$ (1.0)
attacks	$\tilde{w}_{71}$ (1.0)	$\tilde{w}_{28}$ (1.0)
attend	$\tilde{w}_{224}$ (1.0)	$\tilde{w}_{15}$ (1.0)
attorney	$\tilde{w}_{91}$ (1.0)	$\tilde{w}_{28}$ (1.0)
attribute	$\tilde{w}_{263}$ (1.0)	$\tilde{w}_{22}$ (1.0)
attributes	$\tilde{w}_{263}$ (1.0)	$\tilde{w}_{22}$ (1.0)

Table 2: An example of the 20NG words clustered by the soft clustering scheme. $\tilde{w}_i$ are centroids to which the words refer, the centroid weights are shown in the brackets. Many of the words are related to only one cluster.

The time complexity of Algorithm 1 is $O(c_1c_2mn)$, where c_1 is an upper limit on the number of annealing stages, c_2 is an upper limit on the number of convergence stages, m is the number of categories and n is the number of data points to cluster.

In Table 2 we see an example of the output of Algorithm 1 applied to the 20NG corpus (see Section 5.2) with both $k = 300$ and $k = 50$ cluster centroids. For instance, we see that $P(\tilde{w}_4|\text{attacking}) = 0.99977$ and $P(\tilde{w}_1|\text{attacking}) = 0.000222839$. Thus, the word "attacking" mainly belongs to cluster $\tilde{w}_4$. As can be seen, all the words in the table belong to a single cluster or mainly to a single cluster. With values of k in this range this behavior is typical to most of the words in this corpus (and in fact, to also to the other two corpora we consider). Only a small fraction of less than 10% of words significantly belong to more than one cluster, for any number of clusters $50 \leqslant k \leqslant 500$. It is also interesting to see that

IB clustering often results in word stemming. For instance, "atom" and "atoms" belong to the same cluster. Moreover, contextually synonymous words are often assigned to the same cluster, as can be seen in Table 10 (in Appendix 10), which lists prominent members of one cluster that mainly captures "computer words" such as "computer", "hardware", "ibm", "multimedia", "pc", "processor", "software", "8086" etc., which compose the bulk of this cluster.

4.3 Support Vector Machines (SVMs)

The *support vector machine (SVM)* (Vapnik, 1995; Cristianini and Shawe-Taylor, 2000) is a strong inductive learning scheme that enjoys a considerable theoretical and empirical support. As noted in Section 3 there is much empirical support for using SVMs for text categorization (Joachims, 2001; Dumais et al., 1998, etc.).

Informally, for linearly separable two-class data, the (linear) SVM computes the *maximum margin* hyperplane that separates the classes. For non-linearly separable data there are two possible extensions. The first (see Cortes and Vapnik, 1995; Burges, 1998) computes a "soft" maximum margin separating hyperplane that allows for training errors. The accommodation of errors is controlled using a fixed cost parameter. The second solution is obtained by implicitly embedding the data into a high (or infinite) dimensional space where the data is likely to be separable. Then, a maximum margin hyperplane is sought in this high-dimensional space. A combination of both approaches (soft margin and embedding) is often used.

The SVM computation of the (soft) maximum margin is posed as a quadratic optimization problem that can be solved in time complexity of $O(kn^2)$ where n is the training set size and k is the dimension of each point (number of features). Thus, when applying SVM for text categorization of large datasets, an efficient representation of the text can be of major importance.

SVMs are well covered by numerous papers, books and tutorials and therefore we suppress further descriptions here. Following Joachims (2001) and Dumais et al. (1998) we use a linear SVM in all our experiments. The implementation we use is the SVM*light* package by Joachims.[15]

4.4 Putting it all together

As discussed in Section 2, for handling m-class categorization problems ($m > 2$) we chose (for both the uni-labeled and multi-labeled settings) a straightforward decomposition into m binary problems. Although this decomposition is not the best for all datasets (see, e.g., Allwein et al., 2000) it allows for a direct comparison with the related results (which in all cases used this decompositions as well).

Thus, for a categorization problem into m classes we construct m binary classifiers such that each classifier is trained to distinguish one category from the rest. In *multi-labeled* categorization experiments we construct for each category a "hard" (threshold) binary SVM and each test document is considered by all binary classifiers. The subset of categories attributed for this document is determined by the subset of classifiers that "accepted" it.

15. The SVM*light* software can be achieved at: http://svmlight.joachims.org/.

On the other hand, in *uni-labeled* experiments we construct for each category a *confidence-rated* SVM that output for a (test) document a real confidence-rate based on the distance of the point to the decision hyperplane. The (single) category of a test document is determined by the classifier that outputs the largest confidence rate (as noted earlier, this approach is sometimes called "max-win").

A major goal of our work is to compare two categorization schemes based on the two representations: The simple BOW representation together with Mutual Information feature selection (called here ***BOW+MI***) and a representation that is based on word clusters computed via IB distributional clustering (called here ***IB***).

Considering first a uni-labeled categorization, given a training set of documents in m categories, for each category c, a binary confidence-rated linear SVM classifier is trained using the following procedure: The k most discriminating words are selected according to the Mutual Information between the word w and the category c (see Equation (2)). Then each training document of category c is projected over the corresponding k "best" words and for each category c a dedicated classifier h_c is trained to separate c from the other categories. For categorizing a new (test) document d, for each category c we project d over the k most discriminating words of category c. Denoting a projected document d by d_c, we compute $h_c(d_c)$ for all categories c. The category attributed for d is $\arg\max_c h_c(d_c)$. For multi-labeled categorization the same procedure is applied except that now we train, for each category c, hard (non-confidence-rated) classifiers h_c and the subset of categories attributed for a test document d is $\{c : h_c(d_c) = 1\}$.

The structure of the IB categorization scheme is similar (in both the uni-labeled and multi-labeled settings) but now the representation of a document consists of vectors of *word cluster* counts corresponding to a cluster mapping (from words to cluster centroids) that is computed for *all* categories simultaneously using the Information Bottleneck distributional clustering procedure (Algorithm 1).

5. Datasets

5.1 Reuters-21578

The Reuters-21578 corpus contains 21578 articles taken from the Reuters newswire.[16] Each article is typically designated into one or more semantic categories such as "earn", "trade", "corn" etc., where the total number of categories is 114. We used the ModApte split, which consists of a training set of 7063 articles and a test set of 2742 articles.[17]

In both the training and test sets we preprocessed each article so that any additional information except for the title and the body was removed. In addition, we lowered the case of letters. Following Dumais et al. (1998) we generated distinct features for words that appear in article titles. In the IB-based setup (see Section 4.4) we applied a filter on low-frequent words: we removed words that appear in less or equal to W_{low_freq} words,

16. The Reuters-21578 collection can be downloaded at: http://www.research.att.com/∼lewis.
17. Note that in these figures we count documents with at least one label. The original split contains 9603 training documents and 3299 test documents where the additional articles have no labels. While in practice it may be possible to utilize additional unlabeled documents for improving performance using semi-supervised learning algorithms (see, e.g., El-Yaniv and Souroujon, 2001), in this work we simply discarded these documents.

where W_{low_freq} is determined using cross-validation (see Section 6.2). In the BOW+MI setup this filtering of low-frequency words is essentially not relevant since these words are already filtered out by the mutual information feature selection index.

5.2 20 Newsgroups

The 20 Newsgroups (20NG) corpus contains 19997 articles taken from the Usenet newsgroups collection.[18] Each article is designated into one or more semantic categories and the total number of categories is 20, all of them are of about the same size. Most of the articles have only one semantic label and about 4.5% of the articles have two or more labels. Following Schapire and Singer (2000) we used the "Xrefs" field of the article headers to detect multi-labeled documents and to remove duplications. We preprocessed each article so that any additional information except for the subject and the body was removed. In addition, we filtered out lines that seemed to be part of binary files sent as attachments or pseudo-graphical text delimiters. A line is considered to be a "binary" (or a delimiter) if it is longer than 50 symbols and contains no blanks. Overall we removed 23057 such lines (where most of these occurrences appeared in a dozen of articles). Also, we lowered the case of letters. As in the Reuters dataset, in the IB-based setup we applied a filter on low-frequent words, using the parameter W_{low_freq} that is determined via cross-validation.

5.3 WebKB: World Wide Knowledge Base

The World Wide Knowledge Base dataset (WebKB)[19] is a collection of 8282 web pages obtained from four academic domains. The WebKB was collected by Craven et al. (1998). The web pages in the WebKB set are labeled using two different polychotomies. The first is according to topic and the second is according to web domain. In our experiments we only considered the first polychotomy, which consists of 7 categories: *course*, *department*, *faculty*, *project*, *staff*, *student* and *other*. Following Nigam et al. (1998) we discarded the categories *other*[20], *department* and *staff*. The remaining part of the corpus contains 4199 documents in four categories. Table 3 specifies the 4 remaining categories and their sizes.

Category	***Number of articles***	***Proportion*** (%)
course	930	22.1
faculty	1124	26.8
project	504	12.0
student	1641	39.1

Table 3: Some essential details of WebKB categories.

Since the web pages are in HTML format, they contain much non-textual information: HTML tags, links etc. We did not filter this information because some of it is useful, for instance anchor-texts of URLs, which in some documents are the only useful textual information. We filter out non-literals and lowered the case of letters. As in the other datasets, in

18. The 20 newsgroups collection can be downloaded at: http://kdd.ics.uci.edu/.
19. The WebKB collection can be downloaded at: http://www.cs.cmu.edu/~WebKB.
20. Note however that *other* is the largest category in WebKB and consists about 45% of this set.

the IB-based setup we applied a filter on low-frequent words, using the parameter W_{low_freq} that is determined via cross-validation.

6. Experimental setup

6.1 Performance measures and performance estimation

Following Dumais et al. (1998) (and for comparison with this work), in our multi-labeled experiments (Reuters and 20NG) we report on *micro-averaged break-even point (BEP)* results. In our uni-labeled experiments (20NG and WebKB) we report on *accuracy* (see Section 2.2). Note that we experiment with both uni-labeled and multi-labeled categorization of 20NG. Although this set is in general multi-labeled, the fraction of multi-labeled articles is rather small (about 4.5%) and therefore a uni-labeled categorization of this set is also meaningful. To this end, we follow Joachims (1997) and consider our (uni-labeled) categorization of a test document to be correct if the label we assign to the document belongs to its true set of labels.

In order to better estimate the performance of our algorithms on test documents we use standard cross-validation estimation in our experiments with 20NG and WebKB. However, when experimenting with Reuters, for compatibility with the experiments of Dumais *et al.* we use its standard ModApte split (i.e. without cross-validation).

Specifically, in both 20NG and WebKB we use 4-fold cross-validation where we randomly and uniformly split each category into 4 folds and we took three folds for training and one fold for testing. Note that this 3/4:1/4 split is proportional to the training to test set size ratios of the ModApte split of Reuters. In the cross-validated experiments we always report on the estimated average (over the 4 folds) performance (either BEP or accuracy), estimated standard deviation and standard error of the mean..

6.2 Hyperparameter optimization

A major issue when working with SVMs (and in fact with almost all inductive learning algorithms) is parameter tuning. As noted earlier (Section 4.3) in our implementation we used linear SVM*light*. The only relevant parameters (for the linear kernel we use) are C (trade-off between training error and margin) and J (cost-factor, by which training errors on positive examples outweigh errors on negative examples). We optimize these parameters using one of the three folds of the training set as a *validation set*.[21] For each of these parameters we fix a small set of feasible values[22] and in general, we attempt to test performance (over the validation set) using all possible combinations of parameter values over the feasible sets.

Note that tuning the parameters C and J is different in the multi-labeled and uni-labeled settings. In the multi-labeled setting we tune the parameters of each individual (binary) classifier independently of the other classifiers. In the uni-labeled setting, parameter tuning is more complex. Since we use the max-win decomposition, the categorization of a document

21. We note that also Dumais et al. (1998) use 1/3 random subset of the training set for validated parameter tuning.
22. Specifically, for the C parameter the feasible set is $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and for J it is $\{0.5, 1, 2, \ldots, 10\}$.

is dependent on all the binary classifiers involved. For instance, if all the classifiers except for one are perfect, this last bad classifier can generate confidence rates that are maximal for all the documents, which results in extremely poor performance. Therefore, a global tuning of all the binary classifiers is necessary.

Despite this, in the case of the 20NG, where we have 20 binary classifiers, a global exhaustive search is too time-consuming and, ideally, a clever search in this high dimensional parameter space should be considered. Instead, we simply utilized the information we have on the 20NG categories to reduce the size of the parameter space. Specifically, among the 20 categories of 20NG there are some highly correlated ones and we split the list of the categories into 9 groups as in Table 4.[23] For each group the parameters are tuned together and independently of other groups. This way we achieve an approximately global parameter tuning also on the 20NG set. Note that the (much) smaller size of WebKB (both number of categories and number of documents) allow for global parameter tuning over the feasible parameter value sets without any need for approximation.

Group	***Content***
1	(a) talk.religion.misc; (b) soc.religion.christian (c) alt.atheism
2	(a) rec.sport.hockey; (b) rec.sport.baseball
3	(a) talk.politics.mideast
4	(a) sci.med; (b) talk.politics.guns; (c) talk.politics.misc
5	(a) rec.autos; (b) rec.motorcycles; (c) sci.space
6	(a) comp.os.ms-windows.misc; (b) comp.graphics; (c) comp.windows.x
7	(a) sci.electronics; (b) comp.sys.mac.hardware; (c) comp.sys.ibm.pc.hardware
8	(a) sci.crypt
9	(a) misc.forsale

Table 4: Split of the 20NG's categories into thematic groups.

In IB categorization also the parameter W_{low_freq} (see Section 5), which determines a filter on low-frequent words, has significant impact on categorization quality. Therefore, in IB categorization we search for both the SVM parameters and W_{low_freq}. To reduce the time complexity we employ the following simple search heuristic. We first fix random values of C and J and then, using the validation set, we optimize W_{low_freq}.[24] After determining W_{low_freq} we tune both C and J as described above.[25]

6.3 Fair vs. unfair parameter tuning

In our experiments with the BOW+MI and IB categorizers we sometimes perform *unfair* parameter tuning in which we tune the SVM parameters over the *test* set (rather than the

23. It is important to note that an almost identical split can be computed in a completely unsupervised manner using the Multivariate Information Bottleneck of (Friedman et al., 2001).
24. The set of feasible W_{low_freq} values we use is $\{0, 2, 4, 6, 8\}$.
25. The "optimal" determined value of W_{low_freq} for Reuters is 4, for WebKB (across all folds) it is 8 and for 20NG it is 0. The number of distinct words after removing low-frequent words is: 9,953 for Reuters ($W_{low\_freq} = 4$), about 110,000 for 20NG ($W_{low\_freq} = 0$) and about 7,000 for WebKB ($W_{low\_freq} = 8$), depending on the fold.

validation set). If a categorizer A achieves better performance than a categorizer B while B's parameters were tuned unfairly (and A's parameters were tuned fairly) then we can get stronger evidence that A performs better than B. In our experiments we sometimes use this technique to accentuate differences between the two categorizers.

7. Categorization Results

7.1 Multi-labeled categorization

Table 5 summarizes the multi-labeled categorization results obtained by the two categorization schemes (BOW+MI and IB) over Reuters (10 largest categories) and 20NG datasets. Note that the 92.0% BEP result for BOW+MI over Reuters was established by Dumais et al. (1998).[26] To the best of our knowledge, the 88.6% BEP we obtain on 20NG is the first reported result of a multi-labeled categorization of this dataset. Previous attempts at multi-labeled categorization of this set were performed by Schapire and Singer (2000), but no overall result on the entire set was reported.

Categorizer	***Reuters (BEP)***	***20NG (BEP)***
BOW+MI $k = 300$	92.0 obtained by Dumais et al. (1998)	77.7 ± 0.5 (0.31) unfair 76.5 ± 0.4 (0.25) fair
BOW+MI $k = 15000$	92.0	86.3 ± 0.5 (0.27) unfair 85.6 ± 0.6 (0.35) fair
IB $k = 300$	91.2 fair 92.6 unfair	88.6 ± 0.3 (0.21)

Table 5: Multi-labeled categorization BEP results for 20NG and Reuters. k is the number of selected words or word-clusters. All 20NG results are averages of 4-fold cross-validation. Standard deviations are given after the "±" symbol and standard errors of the means are given in brackets. "Unfair" indicates unfair parameter tuning over the test sets (see Section 6.3).

On 20NG the advantage of the IB categorizer over BOW+MI is striking when $k = 300$ words (and $k = 300$ word clusters) are used. Note that the 77.7% BEP of BOW+MI is obtained using *unfair* parameter tuning (see Section 6.3). However, this difference does not sustain when we use $k = 15,000$ words. Using this rather large number of words the BOW+MI performance significantly increases to 86.3% (again, using unfair parameter tuning), which taking into account the statistical deviations is similar to the IB BEP performance. The BOW+MI results that are achieved with fair parameter tuning show an increase in the gap between the performance of the two methods. Nevertheless, the IB categorizer achieves this BEP performance using only 300 features (word clusters), almost two order of magnitude smaller than 15,000. Thus, with respect to 20NG, the IB categorizer outperforms the BOW+MI categorizer both in BEP performance and in representation

26. This result was achieved using binary BOW representation, see Section 3. We replicated Dumais *et al.*'s experiment and in fact obtained a slightly higher BEP result of 92.3%.

efficiency. We also tried other values of the k parameter, where $300 < k \ll 15,000$ and $k > 15,000$. We found that the learning curve, as a function of k, is monotone increasing until it reaches a plateau around $k = 15,000$.

We repeat the same experiment over the Reuters dataset but there we obtain different results. Now the IB categorizer lose its BEP advantage and achieves a 91.2% BEP[27], a slightly inferior (but quite similar) performance to the BOW+MI categorizer (as reported by Dumais et al., 1998). Note that the BOW+MI categorizer does not benefit from increasing the number of features up to $k = 15,000$.

Categorizer	***WebKB (Accuracy)***	***20NG (Accuracy)***
BOW+MI $k = 300$	92.6 ± 0.3 (0.20)	85.5 ± 0.7 (0.45) unfair 84.7 ± 0.7 (0.41) fair
BOW+MI $k = 15000$	92.4 ± 0.5 (0.32)	90.9 ± 0.2 (0.12) unfair 90.2 ± 0.3 (0.17) fair
IB $k = 300$	91.0 ± 0.5 (0.32) unfair 89.5 ± 0.7 (0.41) fair	91.3 ± 0.4 (0.24)

Table 6: Uni-labeled categorization accuracy for 20NG and WebKB. k is the number of selected words or word-clusters. All accuracies are averages of 4-fold cross-validation. Standard deviations are given after the "±" symbol and standard errors of the means are given in brackets. "Unfair" indicates unfair parameter tuning over the test sets (see Section 6.3).

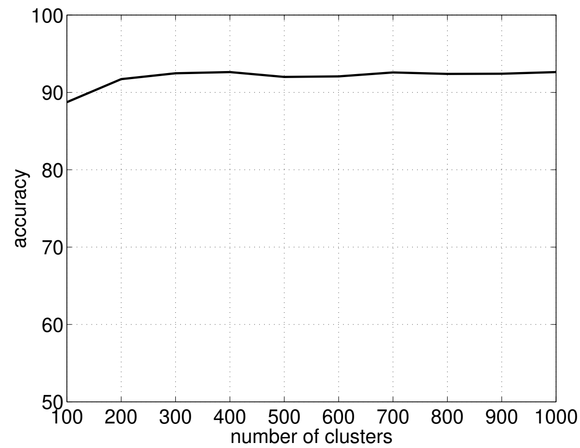

Figure 1: Accuracy vs. number of word-clusters; uni-labeled categorization of 20NG by IB categorizer. We have not performed similar tests on Reuters and WebKB.

27. Using unfair parameter tuning the IB categorizer achieves 92.6% BEP.

7.2 Uni-labeled categorization

We also perform uni-labeled categorization experiments using the BOW+MI and IB categorizers over 20NG and WebKB. The final accuracy results are shown in Table 6. These results appear to be qualitatively similar to the multi-labeled results presented above with WebKB replacing Reuters. Here again, over the 20NG set, the IB categorizer is showing a clear accuracy advantage over BOW+MI with $k = 300$ and this advantage is diminished if we take $k = 15,000$. On the other hand, we observe a comparable (and similar) accuracy of both categorizers over WebKB, and as it is with Reuters, here again the BOW+MI categorizer does not benefit by increasing the feature set size.

The use of $k = 300$ clusters in the IB categorizer is not necessarily optimal. This choice is made for compatibility with the original BOW+MI results of Dumais et al. (1998). The graph in Figure 1 plots the accuracy test results of the IB categorizer over 20NG as a function of the number of clusters. Evidently, after a relatively small number of word clusters the accuracy reaches a plateau.[28]

8. Corpora Complexity vs. Representation Efficiency

The categorization results reported above show that the performance of the BOW+MI categorizer and the IB categorizer is sensitive to the dataset being categorized. What makes the performance of these two categorizers different over different datasets? Why does the more sophisticated IB categorizer outperform the BOW-MI categorizer (with either higher accuracy or better representation efficiency) over 20NG but not over Reuters and WebKB? In this section we discuss this question and attempt to identify differences between these corpora that can account for this behavior.

One possible approach to quantify the complexity of a corpus with respect to a categorization system is to observe and analyze learning curves plotting the performance of the categorizer as a function of the number of words selected for representing each category. Before presenting such learning curves for the three corpora, we focus on the following extreme case where we categorize each of the corpora using only the *three* top words per category (where top-scores are measured using the Mutual Information of words with respect to categories). Tables 7, 8 and 9 specify for each corpus a list of the top three words for each category, together with the performance achieved by the BOW+MI (binary) classifier of the category. For comparison, we also provide the corresponding performance of BOW+MI using the 15,000 top words (i.e. potentially all the significant words in the corpus). For instance, observing Table 7, computed for Reuters, we see that based only on the words “vs”, “cts” and “loss” it is possible to achieve 93.5% BEP when categorizing the category *earn*. We note that the word “vs” appears in 87% of the articles of the category *earn* (i.e., in 914 articles among total 1044 of this category). This word appears in only 15 non-*earn* articles in the test set and therefore “vs” can, by itself, categorize *earn* with very high precision.[29] This phenomenon was already noticed by Joachims (1997), who noted that a classifier built on only one word (“wheat”) can lead to extremely high accuracy when distinguishing be-

28. No cross-validation was applied due to computational complexity of this experiment.
29. In the training set the word “vs” appears in 1900 of the 2709 earn articles (70.1%) and only in 14 of the 4354 non-*earn* articles (0.3%).

tween the Reuters category *wheat* and the other categories (within a uni-labeled setting).[30] The difference between the 20NG and the two other corpora is striking when considering the relative improvement in categorization quality when using 15,000 words. While one can dramatically improve categorization of 20NG by over 150% with many more words, we observe a relative improvement of only less than 20% and 30% in the case of Reuters and WebKB, respectively.

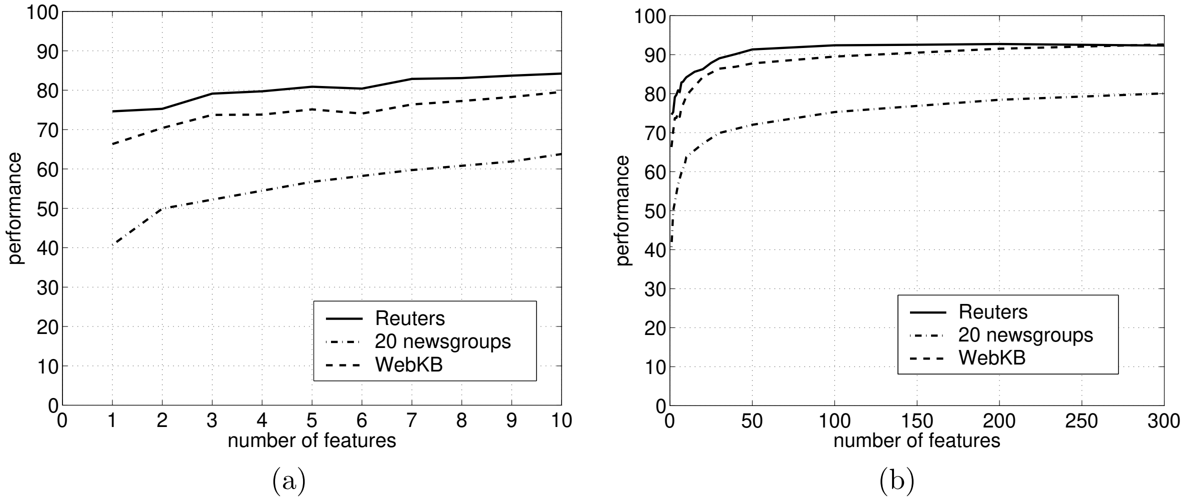

Figure 2: Learning curves (BEP or accuracy vs. number of words) for the datasets: Reuters-21578, 20NG and WebKB over the MI-sorted top 10 words (a) and the top 300 words (b) using the BOW+MI categorizer.

In Figure 2 we present, for each dataset, a learning curve plotting the obtained performance of the BOW+MI categorizer as a function of the number k of selected words.[31] As can be seen, the two curves of both Reuters and WebKB are very similar and almost reach a plateau with $k = 50$ words (that were chosen using the greedy, non-optimal Mutual Information index). This indicates that other words do not contribute much to categorization. On the other hand, the learning curve of 20NG rises slower and still exhibits a rising slope with $k = 300$ words.

The above findings indicate on systematic difference between the categorization of the 20NG dataset on the one hand, and of the Reuters and WebKB datasets, on the other hand. We identify another interesting difference between the corpora (Reuters and WebKB on one the hand and 20NG on the other). This difference is related to the hyper-parameter W_{low_freq} (see Section 5). The bottom line is that in the case of 20NG IB categorization improves when W_{low_freq} *decreases* while in the case of Reuters and WebKB it improves when W_{low_freq} *increases*. In other words, more words and even the most infrequent words

30. When using only one word per category, we observed a 74.6% BEP when categorizing Reuters (10 largest categories), 66.3% accuracy when categorizing WebKB and 40.7% BEP when categorizing 20NG.

31. In the case of Reuters and 20NG the performance is measured in terms of BEP and in the case of WebKB in terms of accuracy.

Category	*1st word*	*2nd word*	*3rd word*	*BEP on 3 words*	*BEP on 15000 words*	*Relative Improvement*
earn	vs+	cts+	loss+	93.5%	98.6%	5.4%
acq	shares+	vs−	Inc+	76.3%	95.2%	24.7%
money-fx	dollar+	vs−	exchange+	53.8%	80.5%	49.6%
grain	wheat+	tonnes+	grain+	77.8%	88.9%	14.2%
crude	oil+	bpd+	OPEC+	73.2%	86.2%	17.4%
trade	trade+	vs−	cts−	67.1%	76.5%	14.0%
interest	rates+	rate+	vs−	57.0%	76.2%	33.6%
ship	ships+	vs−	strike+	64.1%	75.4%	17.6%
wheat	wheat+	tonnes+	WHEAT+	87.8%	82.6%	-5.9%
corn	corn+	tonnes+	vs−	70.3%	83.7%	19.0%
Average				79.9%	92.0%	15.1%

Table 7: Three best words (in terms of mutual information) and their categorization BEP rate of the 10 largest categories of Reuters. The micro-average over these categories is 79.9%. '+' near a word means that the appearance of the word predicts the corresponding category, '−' means that the absence of the word predicts the category. Words in upper-case are words that appeared in article titles (see Section 5.1).

Category	*1st word*	*2nd word*	*3rd word*	*Accuracy on 3 words*	*Accuracy on 15000 words*	*Relative Improvement*
course	courses	course	homework	79.0%	95.7%	21.1%
faculty	professor	cite	pp	70.5%	89.8%	27.3%
project	projects	umd	berkeley	53.2%	80.8%	51.8%
student	com	uci	homes	78.3%	95.9%	22.4%
Average				73.3%	92.4%	26.0%

Table 8: Three best words (in terms of mutual information) and their categorization accuracy rate of the 4 representative categories of WebKB. The micro-average over these categories is 73.3%. All these words contribute by their appearance, rather than absence.

can be useful and improve the (IB) categorization of 20NG. On the other hand, such rare words do add noise in the (IB) categorization of Reuters and WebKB. Figure 3 depicts the performance of the IB classifier on the three corpora as a function of W_{low_freq}. Note again that this phenomenon is observed with respect to the IB representation and the previous discussion concerns the BOW+MI representation.

9. Computational efforts

Here we note on the computational efforts that required for running the above categorization experiments. We performed all our experiments using a 600MHz 2G RAM dual processor

Category	*1st word*	*2nd word*	*3rd word*	*Accuracy on 3 words*	*Accuracy on 15000 words*	*Relative Improvement*
alt.atheism	atheism	atheists	morality	48.7%	84.8%	74.1%
comp.graphics	image	jpeg	graphics	40.5%	83.1%	105.1%
comp.os.ms-windows.misc	windows	m	o	60.9%	84.7%	39.0%
comp.sys.ibm.pc.hardware	scsi	drive	ide	13.8%	76.6%	455.0%
comp.sys.mac.hardware	mac	apple	centris	61.0%	86.7%	42.1%
comp.windows.x	window	server	motif	46.6%	86.7%	86.0%
misc.forsale	00	sale	shipping	63.4%	87.3%	37.6%
rec.autos	car	cars	engine	62.0%	89.6%	44.5%
rec.motorcycles	bike	dod	ride	77.3%	94.0%	21.6%
rec.sport.baseball	baseball	game	year	38.2%	95.0%	148.6%
rec.sport.hockey	hockey	game	team	67.7%	97.2%	43.5%
sci.crypt	key	encryption	clipper	76.7%	95.4%	24.3%
sci.electronics	circuit	wire	wiring	15.2%	85.3%	461.1%
sci.med	cancer	medical	msg	26.0%	92.4%	255.3%
sci.space	space	nasa	orbit	62.5%	94.5%	51.2%
soc.religion.christian	god	church	sin	50.2%	91.7%	82.6%
talk.politics.guns	gun	guns	firearms	41.5%	87.5%	110.8%
talk.politics.mideast	israel	armenian	turkish	54.8%	94.1%	71.7%
talk.politics.misc	cramer	president	ortilink	23.0%	67.7%	194.3%
talk.religion.misc	jesus	god	jehovah	6.6%	53.8%	715.1%
Average				46.83%	86.40%	153.23%

Table 9: Three best words (in terms of mutual information) and their categorization accuracy rate (uni-labeled setting) of the 20 categories of 20NG. The micro-average over these categories is 46.8%. All these words contribute by their appearance, rather than absence.

Pentium III PC operated by Windows 2000. The computational bottlenecks were mainly experienced over 20NG whose size is substantially larger than the sizes of Reuters and WebKB.

Let us first consider the multi-labeled experiments with 20NG. When running the BOW+MI categorizer, the computational bottleneck was the SVM training, for which a single run (one of the 4 cross-validation folds, including both training and testing) could take a few hours, depending on the parameter values. In general, the smaller the parameters C and J are, the faster the SVM training is.[32]

As for the IB categorizer, the SVM training process was faster when the input vectors consisted of word clusters. However, the clustering itself could take up to one hour on the entire 20NG set, and required substantial amount of memory (up to 1G RAM). The overall training and testing time over the entire 20NG in the multi-labeled setting was about 16 hours (4 hours for each of the 4 folds).

32. SVM*light* and its parameters are described in Joachims (1998a).

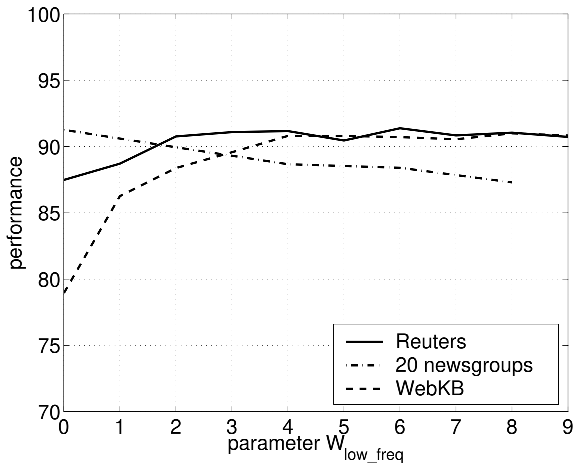

Figure 3: Performance of IB categorizer vs. the parameter W_{low_freq}, that specifies the threshold on low-frequent words filter: words that appear in less than W_{low_freq} articles are removed; uni-labeled categorization of WebKB and 20NG (accuracy), multi-labeled categorization of Reuters (BEP). Note that $W_{low\_freq} = 0$ corresponds to the case where this filter is disabled.

The computational bottleneck when running uni-labeled experiments was the SVM parameter tuning. It required a repetition for each combination of the parameters and individual classifiers (see Section 6.2). Overall the experiments with the IB categorizer took about 45 hours of CPU time, while the BOW-MI categorizer required about 96 hours (i.e. 4 days).

The experiments with the relatively small WebKB corpus were accordingly less time-consuming. In particular, the experiments with the SVM+MI categorizer required 7 hours of CPU time and those with the IB categorizer, about 8 hours. Thus, when comparing these times with the experiments on 20NG we see that the IB categorizer is less time-consuming than the BOW+MI categorizer (based on 15000 words) but the clustering algorithm requires larger memory. On Reuters the experiments ran even faster, because there was no need to apply cross-validation estimation.

10. Concluding remarks

In this study we have provided further evidence for the effectiveness of a sophisticated technique for document representation using distributional clustering of words. Previous studies of distributional clustering of words remained somewhat inconclusive because the overall absolute categorization performance were not state-of-the-art, probably due to the weak classifiers they employed (to the best of our knowledge, in all pervious studies of distributional clustering as a representation method for supervised text categorization, the classifier used was Naive Bayes).

We show that when Information Bottleneck distributional clustering is combined with an SVM classifier, it yields high performance (uni-labeled and multi-labeled) categorization of the 20NG dataset. This result indicates that sophisticated document representations can significantly outperform the standard BOW representation and achieve state-of-the-art performance. In particular, on the 20NG dataset, with respect to either multi-labeled or uni-labeled categorization, we obtain either accuracy (BEP) or representation efficiency advantages over BOW when the categorization is based on SVM.

Nevertheless, we found no accuracy (BEP) or representation efficiency advantage to this feature generation technique when categorizing the Reuters or WebKB corpora. Our study of the three corpora shows fundamental differences between these corpora. Specifically, we observe that Reuters and WebKB can be categorized with close to "optimal" performance using a small set of words, where the addition of many thousands more words provides no significant improvement. On the other hand, the categorization of 20NG can significantly benefit from the use of a large vocabulary. This indicates that the "complexity" of the 20NG corpus is in some sense larger than that of Reuters and WebKB. In addition, we see that the IB representation can significantly benefit from including even the most infrequent words when it is applied with the 20NG corpus. On the other hand, such infrequent words degrade the performance of the IB categorizer when applied to the Reuters and WebKB corpora.

Based on our experience with the above corpora we note that when testing fancy feature generation techniques for text categorization, one should avoid making definitive conclusions based only on "low-complexity" corpora such as Reuters and WebKB. It seems that sophisticated representation methods cannot outperform BOW on such corpora.

Let us conclude with some questions and directions for future research. Given a pool of two or more representation techniques and given a corpus, an interesting question is whether it is possible to combine them in a way that will be competitive with (or even sometimes outperform) the best technique in the pool. A straightforward approach would be to perform cross-validated model selection. However, this approach will be at best as good as the best technique in the pool. Another possibility is to try combining the representation techniques by devising a specialized categorizer for each representation and then use ensemble techniques. Other sophisticated approaches such as "co-training" (see, e.g., Blum and Mitchell, 1998) can also be considered.

Our application of the IB distributional clustering of words employed document class labels but generated a global clustering for all categories. Another possibility to consider is to generate specialized clustering for each (binary) classifier. Another interesting possibility to try is to combine clustering of all n-grams, with $1 \leqslant n \leqslant N$ for some small N.

The BOW+MI categorization employed Mutual Information feature selection where the number k of features (words) was identical for all categories. It would be interesting to consider a specialized k for each category. Although it might be hard to identify good set of vocabularies, this approach may lead to somewhat better categorization and is likely to generate more efficient representations.

Acknowledgements

This research was supported by the Technion and by the Israeli Ministry of Science. We sincerely thank Thorsten Joachims and Rob Schapire for their generous help in conducting this research. We would also like to thank Mark Craven, Susan Dumais, Andrew McCallum, Ron Meir, Yiftach Ravid, Yoram Singer, Noam Slonim and Tong Zhang for their prompt responses and for fruitful discussions. R. El-Yaniv is a Marcella S. Gelltman Academic Lecturer. A preliminary version of this work appeared at SIGIR 2001.

Appendix 1. Example of one 20NG word cluster

Word	***Clusters and their weights***	***Word***	***Clusters and their weights***
1m	$\tilde{w}_{148}$ (0.98813) $\tilde{w}_{78}$ (0.0118695)	funet	$\tilde{w}_{148}$ (1)
286	$\tilde{w}_{148}$ (1)	hardware	$\tilde{w}_{148}$ (1)
386	$\tilde{w}_{148}$ (1)	heine	$\tilde{w}_{148}$ (0.994673) $\tilde{w}_{170}$ (0.00532703)
42bis	$\tilde{w}_{148}$ (1)	ibm	$\tilde{w}_{148}$ (1)
44m	$\tilde{w}_{148}$ (0.999984)	install	$\tilde{w}_{148}$ (1)
4k	$\tilde{w}_{148}$ (1)	interface	$\tilde{w}_{148}$ (1)
4m	$\tilde{w}_{148}$ (0.558864) $\tilde{w}_{133}$ (0.441136)	machine	$\tilde{w}_{148}$ (0.992741) $\tilde{w}_{149}$ (0.00725856)
61801	$\tilde{w}_{148}$ (0.994673) $\tilde{w}_{170}$ (0.00532703)	machines	$\tilde{w}_{148}$ (1)
640x480	$\tilde{w}_{148}$ (1)	mag	$\tilde{w}_{148}$ (1)
64k	$\tilde{w}_{148}$ (1)	matrix	$\tilde{w}_{148}$ (1)
768	$\tilde{w}_{148}$ (1)	megabytes	$\tilde{w}_{148}$ (1)
8086	$\tilde{w}_{148}$ (0.977137) $\tilde{w}_{264}$ (0.0228628)	memory	$\tilde{w}_{148}$ (1)
8500	$\tilde{w}_{148}$ (1)	micro	$\tilde{w}_{148}$ (1)
9090	$\tilde{w}_{148}$ (1)	mode	$\tilde{w}_{148}$ (1)
9600	$\tilde{w}_{148}$ (1)	modes	$\tilde{w}_{148}$ (0.998101) $\tilde{w}_{140}$ (0.00189931)
accelerated	$\tilde{w}_{148}$ (0.999996)	mts	$\tilde{w}_{148}$ (1)
accessed	$\tilde{w}_{148}$ (1)	multimedia	$\tilde{w}_{148}$ (1)
architecture	$\tilde{w}_{148}$ (1)	networking	$\tilde{w}_{148}$ (0.841069) $\tilde{w}_{78}$ (0.158931)
baud	$\tilde{w}_{148}$ (1)	optimization	$\tilde{w}_{148}$ (1)
bbs	$\tilde{w}_{148}$ (1)	optimized	$\tilde{w}_{148}$ (1)
buffered	$\tilde{w}_{148}$ (1)	ox	$\tilde{w}_{148}$ (1)
buggy	$\tilde{w}_{148}$ (1)	pc	$\tilde{w}_{148}$ (1)
bundled	$\tilde{w}_{148}$ (1)	pcs	$\tilde{w}_{148}$ (1)
card	$\tilde{w}_{148}$ (1)	polytechnic	$\tilde{w}_{148}$ (1)
cards	$\tilde{w}_{148}$ (1)	printing	$\tilde{w}_{148}$ (1)
cd	$\tilde{w}_{148}$ (0.999082) $\tilde{w}_{78}$ (0.000917885)	proceeded	$\tilde{w}_{148}$ (1)
clone	$\tilde{w}_{148}$ (1)	processor	$\tilde{w}_{148}$ (1)
compatibility	$\tilde{w}_{148}$ (1)	processors	$\tilde{w}_{148}$ (1)
compatible	$\tilde{w}_{148}$ (1)	resolution	$\tilde{w}_{148}$ (1)
computer	$\tilde{w}_{148}$ (1)	roms	$\tilde{w}_{148}$ (1)
computers	$\tilde{w}_{148}$ (1)	scanner	$\tilde{w}_{148}$ (1)
configured	$\tilde{w}_{148}$ (1)	scanners	$\tilde{w}_{148}$ (1)
connect	$\tilde{w}_{148}$ (1)	scanning	$\tilde{w}_{148}$ (1)
dat	$\tilde{w}_{148}$ (1)	shadows	$\tilde{w}_{148}$ (1)
dial	$\tilde{w}_{148}$ (1)	simtel	$\tilde{w}_{148}$ (1)
disabling	$\tilde{w}_{148}$ (1)	simulator	$\tilde{w}_{148}$ (1)
disk	$\tilde{w}_{148}$ (1)	slower	$\tilde{w}_{148}$ (1)
diskette	$\tilde{w}_{148}$ (1)	slows	$\tilde{w}_{148}$ (1)
docs	$\tilde{w}_{148}$ (0.999975)	software	$\tilde{w}_{148}$ (1)
fastest	$\tilde{w}_{148}$ (1)	svga	$\tilde{w}_{148}$ (0.923418) $\tilde{w}_{136}$ (0.0765818)
faxes	$\tilde{w}_{148}$ (0.999999)	transferring	$\tilde{w}_{148}$ (1)
fd	$\tilde{w}_{148}$ (0.999607) $\tilde{w}_{294}$ (0.000392778)	vga	$\tilde{w}_{148}$ (1)
formatting	$\tilde{w}_{148}$ (1)	video	$\tilde{w}_{148}$ (1)
freeware	$\tilde{w}_{148}$ (0.603074) $\tilde{w}_{265}$ (0.396926)	wanderers	$\tilde{w}_{148}$ (1)

Table 10: $\tilde{w}_i$ are centroids to which the words belong; their weights are shown in the brackets.

References

E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proceedings of ICML'00, 17th International Conference on Machine Learning*, pages 9–16. Morgan Kaufmann Publishers, San Francisco, CA, 2000.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley and ACM Press, 1999.

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*, pages 96–103, Melbourne, AU, 1998. ACM Press, New York, US.

R. Basili, A. Moschitti, and M. T. Pazienza. Language-sensitive text classification. In *Proceedings of RIAO'00, 6th International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 331–343, Paris, France, 2000.

R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. On feature distributional clustering for text categorization. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *Proceedings of SIGIR'01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 146–153, New Orleans, US, 2001. ACM Press, New York, US.

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT'98: Proceedings of 11th Annual Conference on Computational Learning Theory*, pages 92–100. Morgan Kaufmann Publishers, San Francisco, US, 1998.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

M. F. Caropreso, S. Matwin, and F. Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In Amita G. Chin, editor, *Text Databases and Document Management: Theory and Practice*, pages 78–102. Idea Group Publishing, Hershey, US, 2001.

C. Cortes and V. Vapnik. Support vector networks. *Machine Learning 20*, pages 273–297, 1995.

T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, Inc., New York, 1991.

M. Craven, D. DiPasquo, D. Freitag, A. K. McCallum, T. M. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of AAAI'98, 15th Conference of the American Association for Artificial Intelligence*, pages 509–516, Madison, US, 1998. AAAI Press, Menlo Park, US.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines.* Cambridge University Press, 2000.

S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, B(39):1–38, 1977.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

R. O. Duda and P. E. Hart. *Pattern classification and scene analysis.* John Wiley & Sons Inc., New York, 1973.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd ed).* John Wiley & Sons, Inc., New York, 2000.

S. T. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of CIKM'98, 7th ACM International Conference on Information and Knowledge Management*, pages 148–155, Bethesda, US, 1998. ACM Press, New York, US.

R. El-Yaniv and O. Souroujon. Iterative double clustering for unsupervised and semi-supervised learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

N. Friedman, O. Mosenzon, N. Slonim, and N. Tishby. Multivariate information bottleneck. In *Proceedings of UAI'01, 17th Conference on Uncertainty in Artificial Intelligence*, 2001.

J. Fürnkranz. Exploiting structural information for text classification on the WWW. In David J. Hand, Joost N. Kok, and Michael R. Berthold, editors, *Proceedings of IDA'99, 3rd Symposium on Intelligent Data Analysis*, pages 487–497, Amsterdam, NL, 1999. Springer Verlag, Heidelberg, DE.

R. Ghani, S. Slattery, and Y. Yang. Hypertext categorization using hyperlink patterns and meta data. In Carla Brodley and Andrea Danyluk, editors, *Proceedings of ICML'01, 18th International Conference on Machine Learning*, pages 178–185, Williams College, US, 2001. Morgan Kaufmann Publishers, San Francisco, US.

P. S. Jacobs. Joining statistics with nlp for text categorization. In *Proceedings of the Third Conference on Applied Natural Language Processing*, pages 178–185, 1992.

T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In D. H. Fisher, editor, *Proceedings of ICML'97, 14th International Conference on Machine Learning*, pages 143–151, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.

T. Joachims. *Making large-scale support vector machine learning practical*, chapter 11, pages 169–184. MIT Press, Cambridge, MA, 1998a. in B. Scholkopf, C. Burges, A. Smola. Advances in Kernel Methods: Support Vector Machines.

T. Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML'98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998b. Springer Verlag, Heidelberg, DE. Published in the "Lecture Notes in Computer Science" series, number 1398.

T. Joachims. Estimating the generalization performance of an SVM efficiently. Technical Report LS-8 #25, Universität Dortmund, Germany, 1999.

T. Joachims. A statistical learning model of text classification with support vector machines. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *Proceedings of SIGIR'01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 128–136, New Orleans, US, 2001. ACM Press, New York, US.

R. Kohavi and G. John. *Feature Extraction, Construction and Selection : A Data Mining Perspective*, chapter The Wrapper Approach. Kluwer Academic Publishers, 1998.

D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of ICML'96, 13th International Conference on Machine Learning*, pages 284–292, Bari, IT, 1996.

H. Lodhi, J. Shawe-Taylor, N. Cristianini, and C.J.C.H. Watkins. Text classification using string kernels. In *Advances in Neural Information Processing Systems (NIPS)*, pages 563–569, 2000.

C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* The MIT Press, Cambridge, Massachusetts, 1999.

S. Markovitch and D. Rosenstein. Feature generation using general constructor functions. *Machine Learning*, 49:59–98, 2002.

K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Learning to classify text from labeled and unlabeled documents. In *Proceedings of AAAI'98, 15th Conference of the American Association for Artificial Intelligence*, pages 792–799, Madison, US, 1998. AAAI Press, Menlo Park, US.

F. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, 1993.

B. Raskutti, H. Ferrá, and A. Kowalczyk. Second order features for maximising text classification performance. In L. De Raedt and P. A. Flach, editors, *Proceedings of ECML'01, 12th European Conference on Machine Learning*, pages 419–430, Freiburg, DE, 2001. Springer Verlag, Heidelberg, DE.

J. Rennie. Improving multi-class text classification with naive bayes. Master's thesis, Massachusetts Institute of Technology, 2001.

J. Rocchio. *Relevance Feedback in Information Retrieval*, chapter 14, pages 313–323. Prentice Hall, Inc., 1971. in The SMART Retrieval System: Experiments in Automatic Document Processing.

K. Rose. Deterministic annealing for clustering, compression, classification, regression and related optimization problems. *Proceedings of the IEEE*, 86(11):2210–2238, 1998.

G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw Hill, 1983.

R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Computational Learing Theory*, pages 80–91, 1998.

R. E. Schapire and Y. Singer. BoosTexter: a boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

Y. Singer and D. Lewis. Machine learning for information retrieval: Advanced techniques, 2000. A tutorial presented at SIGIR'00, Athens, Greece. Can be achieved at: http://www.cs.huji.ac.il/~singer/papers/ml4ir.ps.gz.

N. Slonim, N. Friedman, and N. Tishby. Unsupervised document classification using sequential information maximization. In *Proceedings of SIGIR'02, 25th ACM International Conference on Research and Development in Information Retrieval*, Tampere, Finland, 2002. ACM Press, New York, US.

N. Slonim, R. Somerville, N. Tishby, and O. Lahav. Objective classification of galaxy spectra using the information bottleneck method. *Monthly Notes of the Royal Astronomical Society*, 323:270–284, 2001.

N. Slonim and N. Tishby. Agglomerative information bottleneck. In *Advances in Neural Information Processing Systems*, pages 617–623, 2000.

N. Slonim and N. Tishby. The power of word clusters for text classification. In *Proceedings of ECIR-01, 23rd European Colloquium on Information Retrieval Research*, Darmstadt, DE, 2001.

N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method, 1999. Invited paper to The 37th annual Allerton Conference on Communication, Control, and Computing.

V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, 1995.

V. N. Vapnik. *Statistical Learning Theory.* John Wiley & Sons Inc., New York, 1998.

H. Wang, D. A. Bell, and F. Murtagh. Axiomatic approach to feature subset selection based on relevance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(3): 271–277, 1999.

S. M. Weiss, C. Apté, F. J. Damerau, D. E. Johnson, F. J. Oles, T. Goetz, and T. Hampp. Maximizing text-mining performance. *IEEE Intelligent Systems*, 14(4):63–69, 1999.

Y. Yang and C. Chute. A linear least squares fit mapping method for information retrieval from natural language texts. In *Proceedings of 14th International Conference on Computational Linguistics (COLING)*, pages 447–453, 1992.

Y. Yang and X. Liu. A re-examination of text categorization methods. In M. A. Hearst, F. Gey, and R. Tong, editors, *Proceedings of SIGIR'99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pages 42–49, Berkeley, US, 1999. ACM Press, New York, US.

Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In D. H. Fisher, editor, *Proceedings of ICML'97, 14th International Conference on Machine Learning*, pages 412–420, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.