The Power of Word Clusters for Text Classification

Noam Slonim and Naftali Tishby

School of Computer Science and Engineering and
The Interdisciplinary Center for Neural Computation
The Hebrew University, Jerusalem 91904, Israel
email: {noamm,tishby}@cs.huji.ac.il

January 12, 2001

Abstract

The recently introduced *Information Bottleneck method* [21] provides an information theoretic framework, for extracting features of one variable, that are relevant for the values of another variable. Several previous works already suggested applying this method for document clustering, gene expression data analysis, spectral analysis and more. In this work we present a novel implementation of this method for supervised text classification. Specifically, we apply the information bottleneck method to find word-clusters that preserve the information about document categories and use these clusters as features for classification. Previous work [1] used a similar clustering procedure to show that word-clusters can significantly reduce the feature space dimensionality, with only a minor change in classification accuracy. In this work we reproduce these results and go further to show that when the training sample is small word clusters can yield significant improvement in classification accuracy (up to 18%) over the performance using the words directly.

1 Introduction

Automatic classification of documents is an increasingly important tool for handling the exponential growth in available on-line texts. Many algorithms have been suggested for this task in the past few years (e.g. [8] [9] [11] [15]). The most common approaches start by evaluating the co-occurrence matrix of words versus documents, given document training data. [1] It is well known, however, that such count matrices tend to be highly sparse and noisy, especially when the training data is relatively small. As a result, documents are usually represented in a high-dimensional sparse feature space, which is far from optimal for classification algorithms. A standard procedure to reduce feature dimensionality is *feature selection*. In this approach one selects a subset of words, using some pre-defined criterion, and uses only the selected words as features for the classification (see e.g. [22]). Latent Semantic Indexing [3] and probabilistic LSI [6] are other similar methods for dimensionality reduction in information retrieval tasks.

An alternative approach is to reduce feature dimensionality by grouping "similar" words into a much smaller number of word-clusters, and use these clusters as features. The crucial stage in such procedures is how to determine the "similarity" of words. Using the recently introduced *Information Bottleneck (IB)* method [21], we show that one can give this question a formal optimal solution that leans purely on information theoretical considerations.

The *IB* method is based on the following simple idea. Given the empirical joint distribution of two variables, one variable is compressed so that the mutual information about the other variable is preserved as much as possible. The method can be considered as finding a *minimal sufficient partition* or *efficient relevant coding* of one variable with respect to the other one.

Several applications already implemented this method for a variety of tasks, including gene expression data analysis [20], classification of galaxies by their spectral properties [19] and *unsupervised* document clustering [17] [18]. In the latter, the two variables correspond to the set of documents and the set of words. In the first stage *word-clusters* that capture the information about the set of documents are extracted as features, and in the second stage these features are used for clustering the documents in an unsupervised manner. The empirical results clearly showed that this representation of the documents, significantly improved the accuracy of *unsupervised* document classification.

[1] Hereafter, we use the term "word" for a maximal string of non-blank characters.

A natural question arising from that work is whether using the word clusters can provide an improvement in classification accuracy in a *supervised* scenario, when the document labels (i.e. topics) are known and used. Therefore, in the supervised case an additional variable is known, the documents categories. Since the text classification task is to predict this variable for new, unlabeled, documents, the word clusters should clearly preserve the information about this relevant category variable. Thus, the procedure is rather simple. First, find word-clusters that preserve the information about the categories as much as possible. Then, use these clusters to represent the documents in a new, low-dimensional, feature space. In this space we now use a supervised classification algorithm to predict the categories of new documents.

Previous work by Baker and McCallum [1] used a similar approach for classification of documents based on word-clusters. Specifically, it was shown there that word-clustering can be used to significantly reduce the feature dimensionality with only a small change in classification performance. We start by reproducing these results, using the *IB* framework which is better theoretically founded procedure. The motivation is to understand why our strong unsupervised results don't help in the supervised case. Are there situations where this feature extraction procedure yields significant *improvement* in supervised classification performance?

A possible answer arises from the fact that word-clusters statistics is much more robust than the original sparse word statistics. Therefore, when word statistics is relatively hard to estimate, the advantage of using (robust) word-clusters might appear. Specifically, this situation is emphasized when the amount of *labeled* training documents is small. Since labeling documents by their category is an expensive process, in many practical situations the amount of labeled data is indeed far from being satisfactory. Our experiments suggest that in such situations using word-clusters leads to a significant improvement in classification accuracy, up to 18%, over using just the words as features.

2 The Information Bottleneck Method

Most clustering algorithms start either from pairwise 'distances' between points (pairwise clustering) or with a distortion measure between a data point and a class centroid (vector quantization). Given the distance matrix or the distortion measure, the clustering task can be adapted in various ways into an optimization problem consisting of finding a small number of classes with low intraclass distortion or with high intraclass connectivity. The main problem with this approach is in the choice of the distance or distortion measures. Often this is an arbitrary choice, sensitive to the specific representation, which may reflect inaccurately the structure of the various components in the high dimensional data.

In our context, a natural measure of similarity of two words is the similarity between their joint distributions with the topic variable. Specifically, let W be the set of words and let C be the set of topics (i.e. document categories), then for every word and every category we can define

$$\hat{p}(c,w) = \frac{n(c,w)}{\sum_{c\in C}\sum_{w\in W} n(c,w)} \ , \tag{1}$$

where $n(c,w)$ is the number of occurrences of the word w in the category c. To calculate $n(c,w)$, we simply sum the occurrences of the word w in all *training* documents that belong to category c. Thus, denoting this document set by D_c, we get,

$$n(c,w) = \sum_{d\in D_c} n(d,w), \tag{2}$$

where $n(d,w)$ is the number of occurrences of the word w in the document d.

Roughly speaking, we would like words with similar distributions over the categories to belong to the same cluster. As already mentioned in [1], the intuition is rather simple. If two different words have similar distributions over the classes, they will play a similar role in the classification process, and thus might as well be clustered together.

This formulation of finding a cluster hierarchy of the members of one set (e.g. words), based on the similarity of their conditional distributions w.r.t the members of another set (e.g. categories), was first introduced in [14] and was called "distributional clustering".

The issue of selecting the 'right' distance measure between distributions remains, however, unresolved in that earlier work. Recently, Tishby, Pereira, and Bialek [21] proposed a principled approach to this problem, which avoids the arbitrary choice of a distortion or a distance measures. In this new approach, given the empirical joint distribution of two random variables $p(x,y)$, one looks for a compact representation of X, which preserves as much information as possible about the relevant variable Y. This simple intuitive idea has a natural information theoretic formulation: *find clusters of the members of the set X, denoted here by $\tilde{X}$, such that the mutual information $I(\tilde{X};Y)$ is maximized, under a constraint on the information extracted from X, $I(\tilde{X};X)$.*

The mutual information, $I(X;Y)$, between the random variables X and Y is given by (e.g. [2])

$$I(X;Y) = \sum_{x \in X, y \in Y} p(x)p(y|x) \log \frac{p(y|x)}{p(y)} \ , \tag{3}$$

and is the only consistent statistical measure of the information that variable X contains about variable Y. The compactness of the representation is determined by $I(\tilde{X};X)$, while the quality of the clusters, $\tilde{X}$, is measured by the fraction of the information they capture about Y, namely, $I(\tilde{X};Y)/I(X;Y)$. Perhaps surprisingly, this general problem has an exact optimal formal solution without any assumption about the origin of the joint distribution $p(x,y)$ [21]. This solution is given in terms of the three distributions that characterize every cluster $\tilde{x} \in \tilde{X}$: the prior probability for this cluster, $p(\tilde{x})$, its membership probabilities $p(\tilde{x}|x)$, and its distribution over the relevance variable, $p(y|\tilde{x})$. In general, the membership probabilities, $p(\tilde{x}|x)$, are "soft", i.e. every $x \in X$ can be assigned to every $\tilde{x} \in \tilde{X}$ in some (normalized) probability. The information bottleneck principle determines the distortion measure between the points x and $\tilde{x}$ to be the $D_{KL}[p(y|x) \| p(y|\tilde{x})] = \sum_y p(y|x) \log \frac{p(y|x)}{p(y|\tilde{x})}$, the Kulback-Libeler divergence [2] between the conditional distributions $p(y|x)$ and $p(y|\tilde{x})$. Specifically, the formal optimal solution is given by the following equations which must be solved together,

$$\begin{cases} p(\tilde{x}|x) = \frac{p(\tilde{x})}{Z(\beta,x)} \exp\left(-\beta D_{KL}[p(y|x) \| p(y|\tilde{x})]\right) \\ p(y|\tilde{x}) = \frac{1}{p(\tilde{x})} \sum_x p(\tilde{x}|x)p(x)p(y|x) \\ p(\tilde{x}) = \sum_x p(\tilde{x}|x)p(x) \ , \end{cases} \tag{4}$$

where $Z(\beta,x)$ is a normalization factor, and the single positive (Lagrange) parameter β determines the "softness" of the classification. Intuitively, in this procedure the information contained in X about Y is 'squeezed' through a compact 'bottleneck' of clusters $\tilde{X}$, that is forced to represent the 'relevant' part in X w.r.t. to Y. In our context, this means that in principle we may find word clusters, denoted by $\tilde{W}$, that try to maximize the amount of information preserved about the categories, C.

3 The Agglomerative Information Bottleneck Algorithm

As has been shown in [16] [18], there is a simple implementation of the information bottleneck method, restricted to the case of "hard" clusters. In this case every word $w \in W$ belongs to precisely one cluster $\tilde{w} \in \tilde{W}$. This restriction, which corresponds to the limit $\beta \to \infty$ in Eqs. (4), yields a natural distance measure between distributions which can be easily implemented in an agglomerative hierarchical clustering procedure.

Let $\tilde{w} \in \tilde{W}$ denote a specific (hard) cluster, then following [18] we define,

$$\begin{cases} p(\tilde{w}|w) = \begin{cases} 1 & \text{if } w \in \tilde{w} \\ 0 & \text{otherwise} \end{cases} \\ p(c|\tilde{w}) = \frac{1}{p(\tilde{w})} \sum_{w \in \tilde{w}} p(c,w) \\ p(\tilde{w}) = \sum_{w \in \tilde{w}} p(w). \end{cases} \tag{5}$$

Using these distributions one can easily evaluate the mutual information between the set of word clusters $\tilde{W}$ and C using Eq.(3). [2]. The general framework applied here is an agglomerative greedy hierarchical clustering algorithm. The algorithm starts with a trivial partitioning into $|W|$ singleton clusters, where each cluster contains exactly one element of W. At each step we *merge* two components of the current partition into a single new component in a way that *locally* minimizes the loss of mutual information about the categories, given in $I(\tilde{W};C)$. Every merger, $(\tilde{w}_i \ \tilde{w}_j) \Rightarrow \tilde{w}_*$, is formally defined by the

[2] In principle, the optimal clustering solution is "soft", which means that each word might be assigned to more than one cluster, with some normalized probability. Taking into account natural language properties (e.g. words disambiguation), it seems natural to apply this approach in our context, and indeed several algorithms exist which directly solve the optimal equations given in Eqs. (4) [14] [16]. However, preliminary tests showed that using these algorithms usually tend to produce similar results, thus for the sake of simplicity, in this work we focus on the simple "hard" clustering algorithm

Input: Joint probability distribution $p(c, w)$

Output: A partition of W into m clusters, $\forall m \in \{1...|W|\}$

Initialization:

- Construct $\tilde{W} \equiv W$
- $\forall i, j = 1...|W|,\ i < j$, calculate
  $d_{i,j} = (p(\tilde{w}_i) + p(\tilde{w}_j)) D_{JS}[p(c|\tilde{w}_i), p(c|\tilde{w}_j)]$

Loop:

- For $m = |W| - 1 \ldots 1$
  - Find the indices $\{i, j\}$ for which $d_{i,j}$ is minimized
  - Merge $\{\tilde{w}_i, \tilde{w}_j\} \Rightarrow \tilde{w}_*$
  - Update $\tilde{W} = \{\tilde{W} - \{\tilde{w}_i, \tilde{w}_j\}\} \bigcup \{\tilde{w}_*\}$
  - Update $d_{i,j}$ costs w.r.t. $\tilde{w}_*$
- End For

Figure 1: Pseudo-code of the agglomerative information bottleneck algorithm.

following equations

$$\begin{cases} p(\tilde{w}_*|w) = \begin{cases} 1 & \text{if } w \in \tilde{w}_i \text{ or } w \in \tilde{w}_j \\ 0 & \text{otherwise} \end{cases} \\ \\ p(c|\tilde{w}_*) = \frac{p(\tilde{w}_i)}{p(\tilde{w}_*)} p(c|\tilde{w}_i) + \frac{p(\tilde{w}_j)}{p(\tilde{w}_*)} p(c|\tilde{w}_j) \\ \\ p(\tilde{w}_*) = p(\tilde{w}_i) + p(\tilde{w}_j). \end{cases} \tag{6}$$

The decrease in the mutual information $I(\tilde{W}; C)$ due to this merger is defined by $\delta I(\tilde{w}_i, \tilde{w}_j) \equiv I(\tilde{W}_{before}; C) - I(\tilde{W}_{after}; C)$, where $I(\tilde{W}_{before}; C)$ and $I(\tilde{W}_{after}; C)$ are the information values before and after the merger, respectively. After a little algebra [17] one can see that

$$\delta I(\tilde{w}_i, \tilde{w}_j) = (p(\tilde{w}_i) + p(\tilde{w}_j)) \cdot D_{JS}[p(c|\tilde{w}_i), p(c|\tilde{w}_j)] \tag{7}$$

where the functional D_{JS} is the *Jensen-Shannon* (JS) divergence (see [13] [4]) defined as

$$D_{JS}[p_i, p_j] = \pi_i D_{KL}[p_i \| \hat{p}\,] + \pi_j D_{KL}[p_j \| \hat{p}\,], \tag{8}$$

where in our case

$$\begin{cases} \{p_i, p_j\} \equiv \{p(c|\tilde{w}_i), p(c|\tilde{w}_j)\} \\ \\ \{\pi_i, \pi_j\} \equiv \{\frac{p(\tilde{w}_i)}{p(\tilde{w}_*)}, \frac{p(\tilde{w}_j)}{p(\tilde{w}_*)}\} \\ \\ \hat{p} = \pi_i p(c|\tilde{w}_i) + \pi_j p(c|\tilde{w}_j)\ . \end{cases} \tag{9}$$

The JS-divergence is non-negative and equals zero if and only if both arguments are identical. It is upper bounded (by 1) and symmetric though it is not a metric. Note that the "merger cost", $\delta I(\tilde{w}_i, \tilde{w}_j)$, can now be interpreted as the multiplication of the 'weight' of the merged elements, $p(\tilde{w}_i) + p(\tilde{w}_j)$, by their 'distance', $D_{JS}[p(c|\tilde{w}_i), p(c|\tilde{w}_j)]$.

By introducing the information optimization criterion the resulting similarity measure directly *emerges* from the analysis. The algorithm is now very simple. At each step we perform "the best possible merger", i.e. merge the clusters $\{\tilde{w}_i, \tilde{w}_j\}$ which minimize $\delta I(\tilde{w}_i, \tilde{w}_j)$. In figure 1 we provide the pseudo code of this agglomerative procedure.

4 The Naive Bayes Classifier

We now turn to describe the classification algorithm used in this work. The Naive Bayes classifier induces a well known classification framework, with rich empirical support in the context of document classification (e.g. [1] [7] [10] [22]). Moreover, it is a relatively simple procedure, which allows for a detailed analysis of the effect of using word-clusters instead of words as features. Therefore, in this work we prefer to concentrate on using the naive Bayes classifier, and leave the combination of more sophisticated classification algorithms with the *IB* method for future work.

Let $\mathcal{D} = \{d_1, d_2, ..., d_{|\mathcal{D}|}\}$ denote the set of training documents, where each document is labeled with one of the categories in $\mathcal{C} = \{c_1, c_2, ..., c_{|\mathcal{C}|}\}$. Given some new document, our goal is to estimate the conditional probability of each category. Using Bayes rule, we know that in general,

$$p(c|d) = \frac{p(d|c)p(c)}{p(d)}. \tag{10}$$

Since we are only interested in the relative order of the categories probabilities (given d), and by definition, $p(d)$ is independent of C, we can focus on:

$$p(c|d) \propto p(d|c)p(c). \tag{11}$$

Without approximation, if we denote the ordered sequence of words that compose the document d by $d \equiv \{w_1, w_2, ..., w_{|d|}\}$, we can write

$$p(d|c) = \Pi_{i=1}^{|d|} p(w_i|w_1, w_2, ..., w_{i-1}, c). \tag{12}$$

However, using the naive Bayes assumption, we assume that the probability of each word in a document is independent of its context. More formally stated, we use the following approximation ("bag of words" model),

$$p(w_i|w_1, w_2, ..., w_{i-1}, c) = p(w_i|c), \tag{13}$$

such that,

$$p(d|c) = \Pi_{i=1}^{|d|} p(w_i|c). \tag{14}$$

Thus, to estimate $p(c|d)$, all we need is to estimate $p(w|c)$ and $p(c)$, for all words $w \in W$ and for all categories $c \in C$. As done in previous works we use the following estimators:

$$\hat{p}(c) = \frac{n(d,c)}{\sum_{c \in C} n(d,c)}, \tag{15}$$

where $n(d,c)$ is the number of training documents in the category c. The conditional probabilities of the words in c is estimated by

$$\hat{p}(w|c) = \frac{n(c,w)}{\sum_{w \in W} n(c,w)}, \tag{16}$$

where $n(c,w)$ is defined in Eq. (2). [3] Using these estimators and the above equations, we can now estimate,

$$\hat{p}(c|d) \propto \Pi_{i=1}^{|d|} \hat{p}(w|c)\hat{p}(c) = \Pi_{i=1}^{|d|} \hat{p}(w,c) \quad , \tag{17}$$

and classify d into the most probable class.

4.1 Using the *IB* word-clusters for classification

The above procedure is simply translated into using word-clusters instead of words to represent the documents. All we need is to replace Eq. (16) with the following estimation,

$$\hat{p}(\tilde{w}|c) = \frac{n(c,\tilde{w})}{\sum_{\tilde{w} \in \tilde{W}} n(c,\tilde{w})}, \tag{18}$$

where $n(c,\tilde{w})$ is naturally defined by,

$$n(c,\tilde{w}) = \sum_{d \in D_c} \sum_{w \in \tilde{w}} n(d,w). \tag{19}$$

[3] To avoid zero probabilities, we used a standard pre-processing of adding 0.5 to each $n(c,w)$.

Having done that, we can now see more formally the intuition behind using word-clusters as features. Under the words representation, we finally used Eq. (17) to determine to which class d should be assigned to. Under the word-clusters representation, we use

$$\hat{p}_{\tilde{W}}(c|d) \propto \Pi_{i=1}^{|d|} \hat{p}(\tilde{w}|c)\hat{p}(c) = \Pi_{i=1}^{|d|} \hat{p}(\tilde{w}, c) \quad , \tag{20}$$

for the same purpose. Therefore, if for example, we have two words with identical distributions over the categories, i.e. $\hat{p}(w_i, c) = \hat{p}(w_j, c) \; \forall c \in \mathcal{C}$, than these words are obviously clustered with no loss of information (Eq. (7)). Denoting the new cluster by $\tilde{w}_*$, from the merging process defined in Eqs. (6), we get that $\hat{p}(\tilde{w}_*, c) = \hat{p}(w_i, c) = \hat{p}(w_j, c) \; \forall c \in \mathcal{C}$. Thus, using $\tilde{w}_*$ instead of w_i and w_j will have no effect on the classification process (but will reduce the feature space dimensionality). In a less extreme case, where the word distributions are just approximately similar, i.e. $\hat{p}(w_i, c) \approx \hat{p}(w_j, c) \; \forall c \in \mathcal{C}$, the algorithm will still tend to cluster them (since this merger will cause a relatively small loss of information). However, in this case we will get that $\hat{p}(\tilde{w}_*, c)$ is a weighted average of $\hat{p}(w_i, c)$ and $\hat{p}(w_j, c)$. The estimation of this average is of course more robust. Thus in some cases, it might even gain an improvement in performance. Specifically, we may predict that when the estimation of $\hat{p}(w, c)$ is relatively poor, using the *IB* word-clusters might result in improving classification accuracy. Our experiments, described in the next sections, strongly support this prediction.

We also notice that the *IB* clustering procedure described in section 3, has a "built-in" property, that increase the probability of clustering first words with a relatively poor estimation of $\hat{p}(w, c)$. Specifically, this effect is due to the prior factor in Eq. (7). Thus, merging words with small priors, will usually cause a smaller decrease in the information about $\mathcal{C}$, than merging words with relatively high priors and better estimations of $\hat{p}(w, c)$.

5 The Experimental Design

In this section we describe our experiments and the datasets used in these experiments. All of the datasets used in this work are based on a standard IR corpus, the $20Newsgroups$ corpus.

5.1 The datasets

The $20Newsgroups$ corpus collected by Lang [9] contains about $20,000$ articles evenly distributed among 20 UseNet discussion groups. This natural language corpus is usually employed for evaluating text classification techniques (e.g. [1] [15] [17]). Many of these groups have similar topics (e.g. five groups discuss different issues concerning computers). In addition, as pointed out by Schapire and Singer [15] about 4.5% of the documents in this corpus are present in more than one group (since people tend to post articles to multiple newsgroups). Therefore, the classification task is typically hard, and suffers from inherent noise, while trying to estimate the relevant probabilities.

For our tests we used 6 different subsets chosen from this corpus, described in table 1. Additionally, we checked the classification over the whole corpus. Our pre-processing included ignoring all file headers, lowering the upper case characters, replacing digits with a special character and all non alpha-numeric characters with another special character. We did not use a stop-list or any stemming procedure, but ignored all words with only one occurrence.

5.2 Experimental procedure

Our main interest is in comparing performance of using words vs. using word-clusters as features. Additionally, we are concerned with how this comparison is effected by the size of the training set (i.e. the sample size). These goals induce four scenarios for comparison:

- *Test1*: using a large sample training set and the words as features.
- *Test2*: using a large sample training set and **word-clusters** as features.
- *Test3*: using a **small** sample training set and the words as features.
- *Test4*: using a **small** sample training set and **word-clusters** as features.

For each dataset we randomly split the documents into two equal sets, and used one as the training-set for *Test1* and *Test2*. Thus, we had approximately 500 training documents for each category. Then, we took a random sample of 5% of the training documents, and used them as small sample training set w.r.t to the same test documents, for the scenarios described in *Test3* and *Test4*. In these cases, thus, we had approximately 25 training documents per category. We repeated this process 10 times

Dataset	Newsgroups included	#documents	Vocabulary size
$20NG$	ALL	$19,997$	$111,805$
$10TEST$	alt.atheism, comp.sys.mac.hardware, misc.forsale, rec.autos, rec.sport.hockey sci.crypt, sci.electronics, sci.med, sci.space, talk.politics.gun.	$10,000$	$65,465$
$COMP$	comp.graphics, comp.windows.x, comp.sys.mac.hardware comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware.	$5,000$	$35,575$
$SCIENCE$	sci.crypt, sci.electronics, sci.med, sci.space.	$4,000$	$36,609$
$POLITICS$	talk.politics.mideast, talk.politics.guns talk.politics.misc.	$3,000$	$35,151$
$RELIGION$	alt.atheism, talk.religion.misc soc.religion.christian.	$2,997$	$29,774$
$SPORT$	rec.sport.baseball rec.sport.hockey.	$2,000$	$18,960$

Table 1: Data sets details.

(for every dataset), where in each iteration we used an initial different random partition of the documents into training and test sets.

Classification accuracy in each test, was assessed as a function of the number of features used, denoted here by N. For *Test1* and *Test3*, we sorted all words by the contribution to the mutual information about the category variable. More formally, we sorted all words by $I(w) \equiv \hat{p}(w) \sum_{c \in C} \hat{p}(c|w) \log \frac{\hat{p}(c|w)}{\hat{p}(c)}$, and used the subset of the top N "informative" words for the classification, where $25 \leq N \leq 2000$. Note, however, that for this sorting we obviously used **only the training documents** (while estimating $p(w, c)$). To avoid too high complexity, in *Test2* and *Test4* we only clustered the same top 2000 most informative words. We produced similar curves for these tests too, where N now determines the number of *IB* word-clusters used as features, and ranges from 25 to 750. In addition, we checked the performance while using all words, i.e. the full vocabulary.

6 Experimental Results

In figure 2 we present the results averaged over the 10 iterations for each of the datasets. The results for the large sample case, where we used approximately 500 documents per category for training (*Test1* and *Test2*), reproduce the results presented in [1]. Specifically, we see that in all datasets, we can use a rather small number of word-clusters to achieve a similar performance to that of using all the words. For example, in the $SPORT$ data set, we reduce the feature space dimensionality by 3 orders of magnitude (from approximately $20,000$ words to 25 word-clusters), while keeping the same classification accuracy, of 0.95. However, in all data sets we see that when the sample size is large, using word-clusters does not significantly improve classification performance.

Nevertheless, the results for the small sample case are rather different. In *Test3* and *Test4*, we have only about 25 training documents per category. In all our data sets we clearly see that in this case using word-clusters instead of words, yields a significant improvement in classification accuracy, between 5% to 18%, when averaging over the 10 iterations (see table 2). For example, in the $20NG$ data set, using only 50 word-clusters already yields significant improvement over the optimal performance while using words. Additionally, we see that in most data sets, the performance using word-clusters is much more robust across different iterations than the performance using the words (see error bars in figure 2). It is also interesting to note that in the small sample case, using the full vocabulary always decreases the performance relatively to using just a small subset of the most informative words.

Dataset	**Max Acc. Words**	**Max Acc. Clusters**	**improvement**
$20NG$	0.382	0.442	15.6%
$10TEST$	0.52	0.616	18.4%
$COMP$	0.473	0.508	7.4%
$SCIENCE$	0.65	0.725	11.5%
$POLITICS$	0.62	0.67	8.1%
$RELIGION$	0.525	0.553	5.3%
$SPORT$	0.685	0.745	8.8%

Table 2: Comparison of maximum accuracy (averaged over 10 iterations) in the small sample size, for using words (*Test3*) vs. using word-clusters (*Test4*).

6.1 Information curves

The *IB* framework also provides a "built-in" measure for evaluating clusters quality. Naturally, this is given by the amount of information preserved between the word-clusters and the categories, i.e. $I(\tilde{W};C)$. It is well known that this information can only decrease during the merging process (e.g. [2], page 32). However, by using the (greedy) agglomerative *IB* algorithm, we try to preserve this information as high as possible for every number of clusters. Thus, we can view the amount of information preserved as a function of the number of clusters, $|\tilde{W}|$. Additionally, we can examine the fraction of preserved information, i.e. $\frac{I(\tilde{W};C)}{I(W;C)}$ to learn how well the words are compressed into clusters. In figure 3 we present these two curves for the $10TEST$ dataset (for all 10 iterations) in both the large and the small sample cases. For the large sample case (*Test2*), we see that the variation between different iterations is relatively small. Also, we see that in less than 20 clusters (a reduction of 2 orders of magnitude), about 75% of the original information is preserved (left figure). For the small sample case (*Test4*), as could be expected, we see a much larger variation in the information curves over different iterations. However, we still see that in less than 20 clusters we preserve around 60% of the original information. Looking at absolute values of information (right figure), we see that for the small sample scenario, there is seemingly much more information to begin with than in the large sample case. This phenomena, which was evident in all our data sets, is due to the poor statistics in the small sample case, where the co-occurrence matrix of the words vs. the categories is much more sparse. In other words, most of this "information" is effectively an artifact due to the non-accurate estimates of the joint probability $p(c,w)$.

6.2 Word-clusters examples

To gain some additional intuition about the clustering process, in this sub-section we present some examples of word-clusters extracted by the agglomerative *IB* algorithm for the $SCIENCE$ data set. For the large sample case we randomly chose half of the documents and for the small sample case we took 5% of this half (i.e. about 25 documents per category). In both cases we found the 2000 most informative words (w.r.t. the categories), and clustered them while trying to maximize the information preserved about the categories. In both cases we see that 25 word-clusters preserve more than 90% of the original information. In table 3 we give some examples of the extracted clusters. Each of these clusters is represented by 10 words (randomly chosen). Roughly speaking, we can see that indeed words with semantic similarity are clustered together, not only

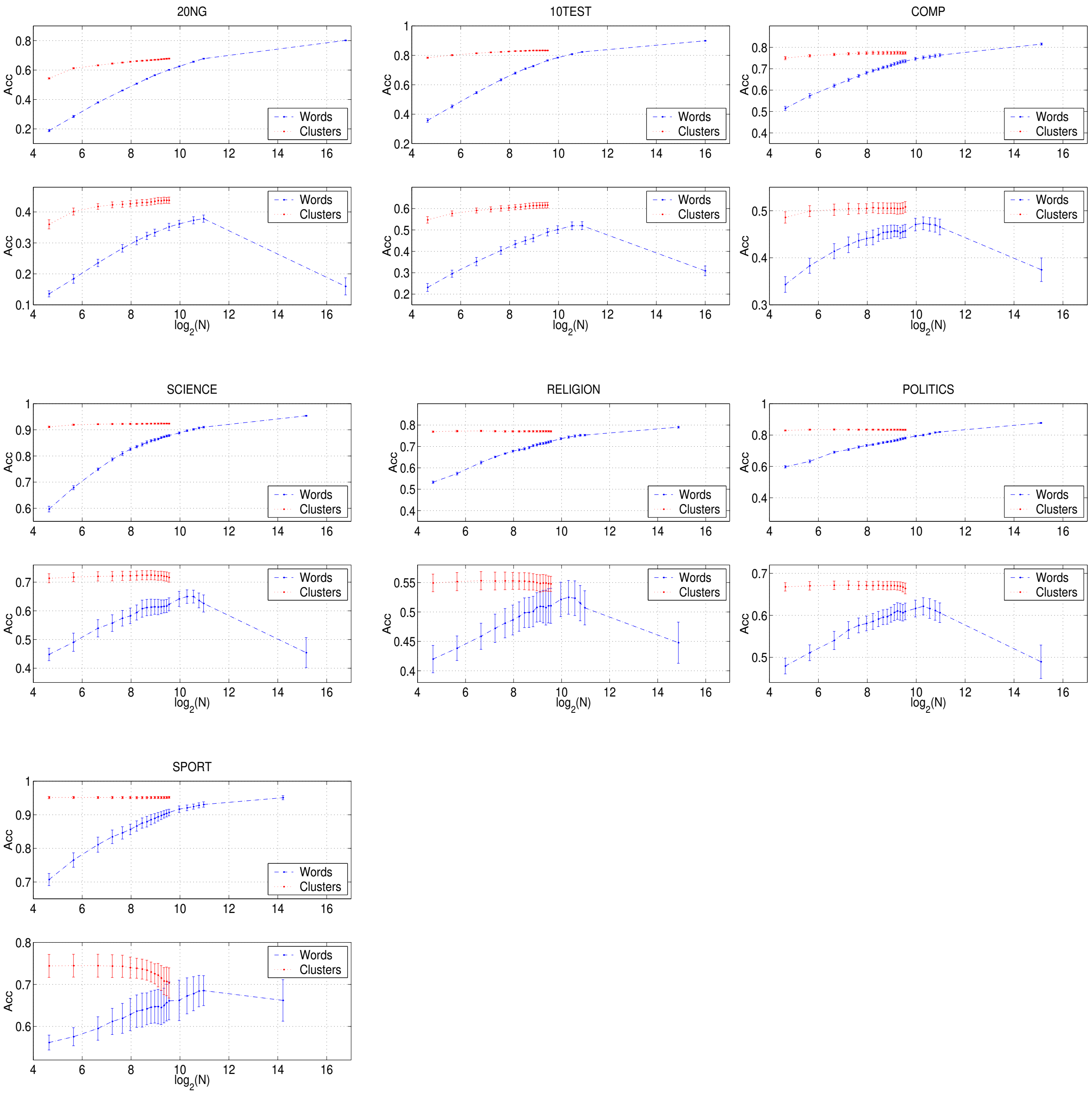

Figure 2: Averaged results for all data sets. The horizontal axis corresponds to the number of words/word-clusters used in the classification (in a semi-logarithmic scale). For each data set, the upper figure correspond to the large sample case (*Test1* and *Test2*), and the lower figure to the small sample case (*Test3* and *Test4*). The error bars represent standard deviation across the 10 different iterations.

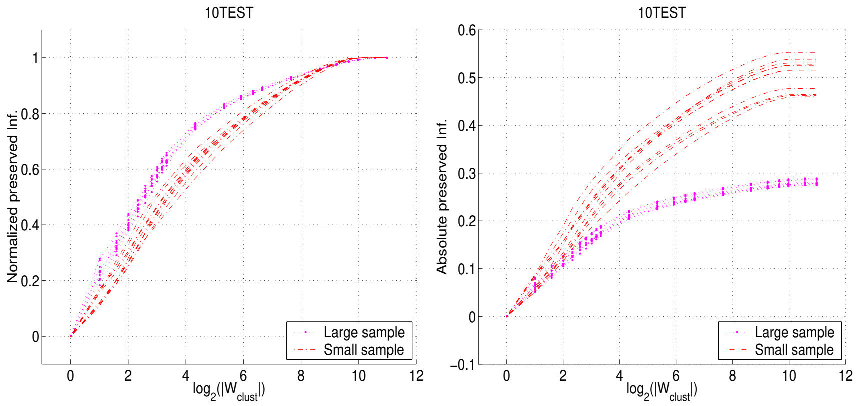

Figure 3: Left: Normalized information curves for all 10 iterations in large and small sample sizes over the *10TEST* data. Horizontal axis is the number of word clusters (in a semi-logarithmic scale) and vertical axis correspond to $\frac{I(\tilde{W};C)}{I(W;C)}$. Right: Absolute information curves for the same runs. Horizontal axis is the same as in left figure, but the vertical axis now correspond to the absolute amount of information preserved by thw word-clusters, i.e. $I(\tilde{W};C)$.

in the large sample case, but also when the sample size is very small. "Neutral" words, that have no clear correlation with one of the categories, also tend to be clustered together.

Cluster	**Large sample**	**Small sample**
$\tilde{w}_1$	space flight mission nasa shuttle moon satellite command ames solar ...	space command spacecraft vlbi jsA lunar mission groundAbased smith bursts ...
$\tilde{w}_2$	encryption encryptionA pgp qA feds fbi secretA privacy security trust ...	key bear encryption clipper DDAbit algorithm escrow security nsa AADA ...
$\tilde{w}_3$	patients cancer msg candida hiv yeast infection hicnet kidney chronic ...	jlA medicine homeopathy osteopathic eye mediocre modern diseaseA disease epilepsy ...
$\tilde{w}_4$	circuit amp plastic Dk panelA voltage circuitA rf outlet conductorA ...	trust chip device disk scheme telephone necessary secure doA else ...
$\tilde{w}_5$	a it is with do are or good one if ...	i in one any make just are what can article ...

Table 3: Clusters examples for the $SCIENCE$ data set, for the large and small sample size cases. Clusters were chosen while $|\tilde{W}| = 25$, and from each cluster 10 members (randomly chosen) are presented. Note in our pre-processing non alpha-numeric characters were replaced by 'A', digits were replaced by 'D', and upper case letters were lowered.

7 Discussion and Further Work

We present a novel implementation of the *IB* method for supervised text classification. By first extracting word-clusters that maximize the information about the categories we obtain a new, robust, low dimensional representation of the documents. Our experimental results show that when the available training data is relatively small, this method yields a significant improvement in classification accuracy (more than 18% in some cases), compared to using the original words as features. When the training data is large, we significantly reduce feature dimensionality, with only a minor change in classification performance.

Several important issues are left for future work:

• A formal analysis of the experimental results is required. Specifically, we would like to prove that "smart" dimensionality reduction techniques can improve the generalization ability when the training sample size is small.

• In this work we used a rather simple classification algorithm, with low computational complexity. More sophisticated approaches usually have higher complexity which in some cases turn them infeasible in high feature space dimension. Using the *IB* dimensionality reduction as a pre-processing can turn such more complex approaches to be more practical.

• When labeled data is expensive, active learning and query techniques can be applied for choosing specific documents for which we ask for the label (see [5]). Such query methods can significantly reduce the amount of labeled data needed to train text classifiers (see e.g. [12]). However, the statistics of labeled documents when using these methods is obviously much smaller. Combining dimensionality reduction techniques, like the one described in this work, with active learning methods, such as "Query by Committee (QBC)", seems very natural and we are currently working in this direction.

Acknowledgments

Useful discussions with Yoram Singer and Yaacov Crammer are greatly appreciated. This research was supported by grants from the Israeli Ministry of Science, and by the US-Israel Bi-national Science Foundation (BSF). N. S. would like to thank the Eshkol fellowship for its support.

References

[1] L. D. Baker and A. K. McCallum. Distributional Clustering of Words for Text Classification In *ACM SIGIR 98*, 1998.

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

[3] S. T. Dumais Using LSI for Information Filtering: TREC3 experiments. Tec. Report 500-225, National Ins. of Standards and Technology, 1995.

[4] R. El-Yaniv, S. Fine, and N. Tishby. Agnostic classification of Markovian sequences. In *Advances in Neural Information Processing (NIPS-97)*, pages 465–471, 1997.

[5] Y. Freund, H.S. Seung, E. Shamir and N. Tishby. Information, Prediction and Query By Committee. In Advances in Neural Information Processing Systems 5 (NIPS-92), 1992.

[6] T. Hofmann. Probabilistic Latent Semantic Indexing. In *ACM SIGIR 99*, pages 50–57, 1999.

[7] T. Joachims A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. In Int. Conf. on Machine Learning (ICML), 1997.

[8] T. Joachims Transductive Inference for Text Classification using Support Vector Machines. In Int. Conf. on Machine Learning (ICML), 1999.

[9] K. Lang. Learning to filter netnews. In *Proc. of the 12th Int. Conf. on Machine Learning*, pages 331–339, 1995.

[10] D. D. Lewis and M. Ringuette. A comparison of two Learning Algorithms for Text Categorization. In *Third Ann. Symp. on Document Analysis and IR*, pages 81–93, 1994.

[11] D. D. Lewis, R. E. Schapire, J. P. Callan, and R. Papka. Training algorithms for linear text classifiers. In *ACM SIGIR 96*, pages 298–306, 1996.

[12] D.D. Lewis and J. Catlett. Heterogeneous Uncertainty Sampling for Supervised Learning. In Proc. of the 11th Int. Conf. of Machine Learning, 148-156.

[13] J. Lin. Divergence Measures Based on the Shannon Entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.

[14] F. C. Pereira, N. Tishby, and L. Lee. Distributional clustering of English words. In *30th Annual Meeting of the Association for Computational Linguistics, Columbus, Ohio*, pages 183–190, 1993.

[15] R. E. Schapire and Y. E. Singer. BoosTexter: A boosting-based system for text categorization. To appear in Machine Learning.

[16] N. Slonim and N. Tishby. Agglomerative Information Bottleneck. In Proc. of Neural Information Processing Systems (NIPS-99), pages 617–623, 1999.

[17] N. Slonim and N. Tishby. Document Clustering Using word Clusters via the Information Bottleneck Method. In *ACM SIGIR 2000*, pages 208–215, 2000.

[18] N. Slonim and N. Tishby. The Hard Clustering Limit of the Information Bottleneck Method. In preparation.

[19] N. Slonim, R. Somerville, N. Tishby and O. Lahav. Objective Spectral Classification of Galaxies using the Information Bottleneck Method. In "Monthly Notices of the Royal Astronomical Society", to appear.

[20] N. Tishby and N. Slonim Data clustering by Markovian relaxation and the Information Bottleneck Method. In Proc. of Neural Information Processing Systems (NIPS-00), 2000, to appear.

[21] N. Tishby, F.C. Pereira and W. Bialek. The Information Bottleneck Method In Proc. of the 37-th Allerton Conference on Communication and Computation, 1999.

[22] Y. Yang and J. Pederson. Feature Selection in Statistical Learning of Text Categorization. In *ICML 97*, pages 412–420, 1997.