Journal of Machine Learning Research 3 (2003) 1265-1287 Submitted 5/02; Published 3/03

A Divisive Information-Theoretic Feature Clustering Algorithm for Text Classification

Inderjit S. Dhillon INDERJIT@CS.UTEXAS.EDU
Subramanyam Mallela MANYAM@CS.UTEXAS.EDU
Rahul Kumar RAHUL@CS.UTEXAS.EDU
Department of Computer Sciences
University of Texas, Austin

Editors: Isabelle Guyon and André Elisseeff

Abstract

High dimensionality of text can be a deterrent in applying complex learners such as Support Vector Machines to the task of text classification. Feature clustering is a powerful alternative to feature selection for reducing the dimensionality of text data. In this paper we propose a new information-theoretic divisive algorithm for feature/word clustering and apply it to text classification. Existing techniques for such "distributional clustering" of words are agglomerative in nature and result in (i) sub-optimal word clusters and (ii) high computational cost. In order to explicitly capture the optimality of word clusters in an information theoretic framework, we first derive a global criterion for feature clustering. We then present a fast, divisive algorithm that monotonically decreases this objective function value. We show that our algorithm minimizes the "within-cluster Jensen-Shannon divergence" while simultaneously maximizing the "between-cluster Jensen-Shannon divergence". In comparison to the previously proposed agglomerative strategies our divisive algorithm is much faster and achieves comparable or higher classification accuracies. We further show that feature clustering is an effective technique for building smaller class models in hierarchical classification. We present detailed experimental results using Naive Bayes and Support Vector Machines on the 20Newsgroups data set and a 3-level hierarchy of HTML documents collected from the Open Directory project (www.dmoz.org).

Keywords: Information theory, Feature Clustering, Classification, Entropy, Kullback-Leibler Divergence, Mutual Information, Jensen-Shannon Divergence.

1. Introduction

Given a set of document vectors $\{d_1, d_2, \ldots, d_n\}$ and their associated class labels $c(d_i) \in \{c_1, c_2, \ldots, c_l\}$, text classification is the problem of estimating the true class label of a new document d. There exist a wide variety of algorithms for text classification, ranging from the simple but effective Naive Bayes algorithm to the more computationally demanding Support Vector Machines (Mitchell, 1997, Vapnik, 1995, Yang and Liu, 1999).

A common, and often overwhelming, characteristic of text data is its extremely high dimensionality. Typically the document vectors are formed using a vector-space or bag-of-words model (Salton and McGill, 1983). Even a moderately sized document collection can lead to a dimensionality in thousands. For example, one of our test data sets contains 5,000 web pages from www.dmoz.org and has a dimensionality (vocabulary size after pruning) of 14,538. This high dimensionality can be a severe obstacle for classification algorithms based on Support Vector Machines, Linear Discriminant Analysis, *k*-nearest neighbor etc. The problem is compounded when the documents are arranged in a hierarchy of classes and a full-feature classifier is applied at each node of the hierarchy.

A way to reduce dimensionality is by the distributional clustering of words/features (Pereira et al., 1993, Baker and McCallum, 1998, Slonim and Tishby, 2001). Each word cluster can then be treated as a single feature and thus dimensionality can be drastically reduced. As shown by Baker and McCallum (1998), Slonim and Tishby (2001), such feature clustering is more effective than feature selection(Yang and Pedersen, 1997), especially at lower number of features. Also, even when dimensionality is reduced by as much as two orders of magnitude the resulting classification accuracy is similar to that of a full-feature classifier. Indeed in some cases of small training sets and noisy features, word clustering can actually increase classification accuracy. However the algorithms developed by both Baker and McCallum (1998) and Slonim and Tishby (2001) are agglomerative in nature making a greedy move at every step and thus yield sub-optimal word clusters at a high computational cost.

In this paper, we use an information-theoretic framework that is similar to Information Bottleneck (see Chapter 2, Problem 22 of Cover and Thomas, 1991, Tishby et al., 1999) to derive a global criterion that captures the optimality of word clustering (see Theorem 1). Our global criterion is based on the generalized Jensen-Shannon divergence (Lin, 1991) among multiple probability distributions. In order to find the best word clustering, i.e., the clustering that minimizes this objective function, we present a new divisive algorithm for clustering words. This algorithm is reminiscent of the *k*-means algorithm but uses Kullback Leibler divergences (Kullback and Leibler, 1951) instead of squared Euclidean distances. We prove that our divisive algorithm *monotonically* decreases the objective function value. We also show that our algorithm minimizes "within-cluster divergence" and simultaneously maximizes "between-cluster divergence". Thus we find word clusters that are markedly better than the agglomerative algorithms of Baker and McCallum (1998) and Slonim and Tishby (2001). The increased quality of our word clusters translates to higher classification accuracies, especially at small feature sizes and small training sets. We provide empirical evidence of all the above claims using Naive Bayes and Support Vector Machines on the (a) 20 Newsgroups data set, and (b) an HTML data set comprising 5,000 web pages arranged in a 3-level hierarchy from the Open Directory Project (www.dmoz.org).

We now give a brief outline of the paper. In Section 2, we discuss related work and contrast it with our work. In Section 3 we briefly review some useful concepts from information theory such as Kullback-Leibler(KL) divergence and Jensen-Shannon(JS) divergence, while in Section 4 we review text classifiers based on Naive Bayes and Support Vector Machines. Section 5 poses the question of finding optimal word clusters in terms of preserving mutual information between two random variables. Section 5.1 gives the algorithm that directly minimizes the resulting objective function which is based on KL-divergences, and presents some pleasing aspects of the algorithm, such as convergence and simultaneous maximization of "between-cluster JS-divergence". In Section 6 we present experimental results that highlight the benefits of our word clustering, and the resulting increase in classification accuracy. Finally, we present our conclusions in Section 7.

A word about notation: upper-case letters such as X, Y, C, W will denote random variables, while script upper-case letters such as $\mathcal{X}$, $\mathcal{Y}$, $\mathcal{C}$, $\mathcal{W}$ denote sets. Individual set elements will often be denoted by lower-case letters such as x, w or x_i, w_t. Probability distributions will be denoted by p, q, p_1, p_2, etc. when the random variable is obvious or by $p(X)$, $p(C|w_t)$ to make the random variable explicit. We use logarithms to the base 2.

2. Related Work

Text classification has been extensively studied, especially since the emergence of the internet. Most algorithms are based on the bag-of-words model for text (Salton and McGill, 1983). A simple but effective algorithm is the Naive Bayes method (Mitchell, 1997). For text classification, different variants of Naive Bayes have been used, but McCallum and Nigam (1998) showed that the variant based on the multinomial model leads to better results. Support Vector Machines have also been successfully used for text classification (Joachims, 1998, Dumais et al., 1998). For hierarchical text data, such as the topic hierarchies of Yahoo! (www.yahoo.com) and the Open Directory Project (www.dmoz.org), hierarchical classification has been studied by Koller and Sahami (1997), Chakrabarti et al. (1997), Dumais and Chen (2000). For some more details, see Section 4.

To counter high-dimensionality various methods of feature selection have been proposed by Yang and Pedersen (1997), Koller and Sahami (1997) and Chakrabarti et al. (1997). Distributional clustering of words has proven to be more effective than feature selection in text classification and was first proposed by Pereira, Tishby, and Lee (1993) where “soft” distributional clustering was used to cluster nouns according to their conditional verb distributions. Note that since our main goal is to reduce the number of features *and* the model size, we are only interested in “hard clustering” where each word can be represented by its unique word cluster. For text classification, Baker and McCallum (1998) used such hard clustering, while more recently, Slonim and Tishby (2001) have used the Information Bottleneck method for clustering words. Both Baker and McCallum (1998) and Slonim and Tishby (2001) use similar agglomerative clustering strategies that make a greedy move at every agglomeration, and show that feature size can be aggressively reduced by such clustering without any noticeable loss in classification accuracy using Naive Bayes. Similar results have been reported for Support Vector Machines (Bekkerman et al., 2001). To select the number of word clusters to be used for the classification task, Verbeek (2000) has applied the Minimum Description Length (MDL) principle (Rissanen, 1989) to the agglomerative algorithm of Slonim and Tishby (2001).

Two other dimensionality/feature reduction schemes are used in latent semantic indexing (LSI) (Deerwester et al., 1990) and its probabilistic version (Hofmann, 1999). Typically these methods have been applied in the *unsupervised* setting and as shown by Baker and McCallum (1998), LSI results in lower classification accuracies than feature clustering.

We now list the main contributions of this paper and contrast them with earlier work. As our first contribution, we use an information-theoretic framework to derive a global objective function that explicitly captures the optimality of word clusters in terms of the generalized Jensen-Shannon divergence between multiple probability distributions. As our second contribution, we present a divisive algorithm that uses Kullback-Leibler divergence as the distance measure, and explicitly minimizes the global objective function. This is in contrast to Slonim and Tishby (2001) who considered the merging of *just two* word clusters at every step and derived a local criterion based on the Jensen-Shannon divergence of *two* probability distributions. Their agglomerative algorithm, which is similar to the algorithm of Baker and McCallum (1998), greedily optimizes this merging criterion (see Section 5.3 for more details). Thus, their resulting algorithm does not directly optimize a global criterion and is computationally expensive — the algorithm of Slonim and Tishby (2001) is $O(m^3 l)$ in complexity where m is the total number of words and l is the number of classes. In contrast the complexity of our divisive algorithm is $O(mkl\tau)$ where k is the number of word clusters (typically $k \ll m$), and τ is the number of iterations (typically $\tau = 15$ on average). Note that our hard clustering leads to a model size of $O(k)$, whereas "soft" clustering in methods such as probabilistic LSI (Hofmann, 1999) leads to a model size of $O(mk)$. Finally, we show that our enhanced word clustering leads to higher classification accuracy, especially when the training set is small and in hierarchical classification of HTML data.

3. Some Concepts from Information Theory

In this section, we quickly review some concepts from information theory which will be used heavily in this paper. For more details on some of this material see the authoritative treatment in the book by Cover and Thomas (1991).

Let X be a discrete random variable that takes on values from the set $\mathcal{X}$ with probability distribution $p(x)$. The entropy of X (Shannon, 1948) is defined as

$$H(p) = -\sum_{x \in \mathcal{X}} p(x) \log p(x) \ .$$

The relative entropy or Kullback-Leibler(KL) divergence (Kullback and Leibler, 1951) between two probability distributions $p_1(x)$ and $p_2(x)$ is defined as

$$KL(p_1, p_2) = \sum_{x \in \mathcal{X}} p_1(x) \log \frac{p_1(x)}{p_2(x)} \ .$$

KL-divergence is a measure of the "distance" between two probability distributions; however it is not a true metric since it is not symmetric and does not obey the triangle inequality (Cover and Thomas, 1991, p.18). KL-divergence is always non-negative but can be unbounded; in particular when $p_1(x) \neq 0$ and $p_2(x) = 0$, $KL(p_1, p_2) = \infty$. In contrast, the Jensen-Shannon(JS) divergence between p_1 and p_2 defined by

$$\begin{aligned} JS_\pi(p_1, p_2) &= \pi_1 KL(p_1, \pi_1 p_1 + \pi_2 p_2) + \pi_2 KL(p_2, \pi_1 p_1 + \pi_2 p_2) \\ &= H(\pi_1 p_1 + \pi_2 p_2) - \pi_1 H(p_1) - \pi_2 H(p_2) \ , \end{aligned}$$

where $\pi_1 + \pi_2 = 1$, $\pi_i \geq 0$, is clearly a measure that is symmetric in $\{\pi_1, p_1\}$ and $\{\pi_2, p_2\}$, and is bounded (Lin, 1991). The Jensen-Shannon divergence can be generalized to measure the distance between any finite number of probability distributions as:

$$JS_\pi(\{p_i : 1 \leq i \leq n\}) = H\left(\sum_{i=1}^{n} \pi_i p_i\right) - \sum_{i=1}^{n} \pi_i H(p_i) \ , \tag{1}$$

which is symmetric in the $\{\pi_i, p_i\}$'s ($\sum_i \pi_i = 1, \pi_i \geq 0$).

Let Y be another random variable with probability distribution $p(y)$. The mutual information between X and Y, $I(X;Y)$, is defined as the KL-divergence between the joint probability distribution $p(x,y)$ and the product distribution $p(x)p(y)$:

$$I(X;Y) \quad = \quad \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \ . \tag{2}$$

Intuitively, mutual information is a measure of the amount of information that one random variable contains about the other. The higher its value the less is the uncertainty of one random variable due to knowledge about the other. Formally, it can be shown that $I(X;Y)$ is the reduction in entropy of one variable knowing the other: $I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$ (Cover and Thomas, 1991).

4. Text Classification

Two contrasting classifiers that perform well on text classification are (i) the simple Naive Bayes method and (ii) the more complex Support Vector Machines.

4.1 Naive Bayes Classifier

Let $C = \{c_1, c_2, \dots, c_l\}$ be the set of l classes, and let $W = \{w_1, \dots, w_m\}$ be the set of words/features contained in these classes. Given a new document d, the probability that d belongs to class c_i is given by Bayes rule,

$$p(c_i|d) = \frac{p(d|c_i)p(c_i)}{p(d)} \quad .$$

Assuming a generative multinomial model (McCallum and Nigam, 1998) and further assuming class-conditional independence of words yields the well-known Naive Bayes classifier (Mitchell, 1997), which computes the most probable class for d as

$$c^*(d) = \text{argmax}_{c_i} p(c_i|d) = \text{argmax}_{c_i} p(c_i) \prod_{t=1}^{m} p(w_t|c_i)^{n(w_t,d)} \tag{3}$$

where $n(w_t,d)$ is the number of occurrences of word w_t in document d, and the quantities $p(w_t|c_i)$ are usually estimated using Laplace's rule of succession:

$$p(w_t|c_i) = \frac{1 + \sum_{d_j \in c_i} n(w_t,d_j)}{m + \sum_{t=1}^{m} \sum_{d_j \in c_i} n(w_t,d_j)} \quad . \tag{4}$$

The class priors $p(c_i)$ are estimated by the maximum likelihood estimate $p(c_i) = \frac{|c_i|}{\sum_j |c_j|}$. We now manipulate the Naive Bayes rule in order to interpret it in an information theoretic framework. Rewrite formula (3) by taking logarithms and dividing by the length of the document $|d|$ to get

$$c^*(d) = \text{argmax}_{c_i} \left(\frac{\log p(c_i)}{|d|} + \sum_{t=1}^{m} p(w_t|d) \log p(w_t|c_i) \right) \quad , \tag{5}$$

where the document d may be viewed as a probability distribution over words: $p(w_t|d) = n(w_t,d)/|d|$. Adding the entropy of $p(W|d)$, i.e., $-\sum_{t=1}^{m} p(w_t|d) \log p(w_t|d)$ to (5), and negating, we get

$$\begin{aligned} c^*(d) &= \text{argmin}_{c_i} \left(\sum_{t=1}^{m} p(w_t|d) \log \frac{p(w_t|d)}{p(w_t|c_i)} - \frac{\log p(c_i)}{|d|} \right) \\ &= \text{argmin}_{c_i} \left(KL(p(W|d), p(W|c_i)) - \frac{\log p(c_i)}{|d|} \right) \quad , \end{aligned} \tag{6}$$

where $KL(p,q)$ denotes the KL-divergence between p and q as defined in Section 3. Note that here we have used W to denote the random variable that takes values from the set of words W. Thus, assuming equal class priors, we see that Naive Bayes may be interpreted as finding the class distribution which has minimum KL-divergence from the given document. As we shall see again later, KL-divergence seems to appear "naturally" in our setting.

By (5), we can clearly see that Naive Bayes is a linear classifier. Despite its crude assumption about the class-conditional independence of words, Naive Bayes has been found to yield surprisingly good classification performance, especially on text data. Plausible reasons for the success of Naive Bayes have been explored by Domingos and Pazzani (1997), Friedman (1997).

4.2 Support Vector Machines

The Support Vector Machine(SVM) (Boser et al., 1992, Vapnik, 1995) is an inductive learning scheme for solving the two-class pattern recognition problem. Recently SVMs have been shown to give good results for text categorization (Joachims, 1998, Dumais et al., 1998). The method is defined over a vector space where the classification problem is to find the decision surface that "best" separates the data points of one class from the other. In case of linearly separable data, the decision surface is a hyperplane that maximizes the "margin" between the two classes and can be written as

$$\langle \mathbf{w}, \mathbf{x} \rangle - b \quad = \quad 0$$

where $\mathbf{x}$ is a data point and the vector $\mathbf{w}$ and the constant b are learned from the training set. Let $y_i \in \{+1, -1\}$($+1$ for positive class and -1 for negative class) be the classification label for input vector $\mathbf{x_i}$. Finding the hyperplane can be translated into the following optimization problem

$$\text{Minimize} : \|\mathbf{w}\|^2$$

subject to the following constraints

$$\begin{aligned} \langle \mathbf{w}, \mathbf{x_i} \rangle \; - \; b \geq +1 \quad \text{for} \quad y_i = +1, \\ \langle \mathbf{w}, \mathbf{x_i} \rangle \; - \; b \leq -1 \quad \text{for} \quad y_i = -1 \; . \end{aligned}$$

This minimization problem can be solved using quadratic programming techniques (Vapnik, 1995). The algorithms for solving the linearly separable case can be extended to the case of data that is not linearly separable by either introducing soft margin hyperplanes or by using a non-linear mapping of the original data vectors to a higher dimensional space where the data points are linearly separable (Vapnik, 1995). Even though SVM classifiers are described for binary classification problems they can be easily combined to handle multiple classes. A simple, effective combination is to train N *one-versus-rest* classifiers for the N class case and then classify the test point to the class corresponding to the largest positive distance to the separating hyperplane. In all our experiments we used linear SVMs as they are faster to learn and to classify new instances compared to non-linear SVMs. Further linear SVMs have been shown to do well on text classification (Joachims, 1998).

4.3 Hierarchical Classification

Hierarchical classification utilizes a hierarchical topic structure such as Yahoo! to decompose the classification task into a set of simpler problems, one at each node in the hierarchy. We can simply extend any classifier to perform hierarchical classification by constructing a (distinct) classifier at each internal node of the tree using all the documents in its child nodes as the training data. Thus the tree is assumed to be "is-a" hierarchy, i.e., the training instances are inherited by the parents. Then classification is just a greedy descent down the tree until the leaf node is reached. This way of classification has been shown to be equivalent to the standard non-hierarchical classification over a flat set of leaf classes if maximum likelihood estimates for *all* features are used (Mitchell, 1998). However, hierarchical classification along with feature selection has been shown to achieve better classification results than a flat classifier (Koller and Sahami, 1997). This is because each classifier can now utilize a different subset of features that are most relevant to the classification sub-task at hand. Furthermore each node classifier requires only a small number of features since it needs to distinguish between a fewer number of classes. Our proposed feature clustering strategy allows us to aggressively reduce the number of features associated with each node classifier in the hierarchy. Detailed experiments on the Dmoz Science hierarchy are presented in Section 6.

5. Distributional Word Clustering

Let C be a discrete random variable that takes on values from the set of classes $\mathcal{C} = \{c_1, \ldots, c_l\}$, and let W be the random variable that ranges over the set of words $\mathcal{W} = \{w_1, \ldots, w_m\}$. The joint distribution $p(C,W)$ can be estimated from the training set. Now suppose we cluster the words into k clusters $\mathcal{W}_1, \ldots, \mathcal{W}_k$. Since we are interested in reducing the number of features *and* the model size, we only look at "hard" clustering where each word belongs to exactly one word cluster, i.e,

$$\mathcal{W} = \cup_{i=1}^{k} \mathcal{W}_i, \quad \text{and} \quad \mathcal{W}_i \cap \mathcal{W}_j = \phi,\ i \neq j\ .$$

Let the random variable W^C range over the word clusters. To judge the quality of word clusters we use an information-theoretic measure. The information about C captured by W can be measured by the mutual information $I(C;W)$. Ideally, in forming word clusters we would like to *exactly* preserve the mutual information; however a non-trivial clustering always lowers mutual information (see Theorem 1 below). Thus we would like to find a clustering that minimizes the decrease in mutual information, $I(C;W) - I(C;W^C)$, for a given number of word clusters. Note that this framework is similar to the one in Information Bottleneck when hard clustering is desired (Tishby et al., 1999). The following theorem appears to be new and states that the change in mutual information can be expressed in terms of the generalized Jensen-Shannon divergence of each word cluster.

Theorem 1 *The change in mutual information due to word clustering is given by*

$$I(C;W) - I(C;W^C) = \sum_{j=1}^{k} \pi(\mathcal{W}_j) JS_{\pi'}(\{p(C|w_t) : w_t \in \mathcal{W}_j\})$$

where $\pi(\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \pi_t$, $\pi_t = p(w_t)$, $\pi'_t = \pi_t / \pi(\mathcal{W}_j)$ *for* $w_t \in \mathcal{W}_j$, *and JS denotes the generalized Jensen-Shannon divergence as defined in (1).*

Proof. By the definition of mutual information (see (2)), and using $p(c_i, w_t) = \pi_t p(c_i|w_t)$ we get

$$\begin{aligned} I(C;W) &= \sum_i \sum_t \pi_t p(c_i|w_t) \log \frac{p(c_i|w_t)}{p(c_i)} \\ \text{and} \quad I(C;W^C) &= \sum_i \sum_j \pi(\mathcal{W}_j) p(c_i|\mathcal{W}_j) \log \frac{p(c_i|\mathcal{W}_j)}{p(c_i)} \ . \end{aligned}$$

We are interested in hard clustering, so

$$\pi(\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \pi_t, \quad \text{and} \quad p(c_i|\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \frac{\pi_t}{\pi(\mathcal{W}_j)} p(c_i|w_t)\ ,$$

thus implying that for all clusters $\mathcal{W}_j$,

$$\pi(\mathcal{W}_j) p(c_i|\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \pi_t p(c_i|w_t)\ , \tag{7}$$

$$p(C|\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \frac{\pi_t}{\pi(\mathcal{W}_j)} p(C|w_t)\ . \tag{8}$$

Note that the distribution $p(C|W_j)$ is the (weighted) mean distribution of the constituent distributions $p(C|w_t)$. Thus,

$$I(C;W) - I(C;W^C) \;=\; \sum_i \sum_t \pi_t p(c_i|w_t) \log p(c_i|w_t) - \sum_i \sum_j \pi(W_j) p(c_i|W_j) \log p(c_i|W_j) \tag{9}$$

since the extra $\log(p(c_i))$ terms cancel due to (7). The first term in (9), after rearranging the sum, may be written as

$$\begin{aligned} \sum_j \sum_{w_t \in W_j} \pi_t \left(\sum_i p(c_i|w_t) \log p(c_i|w_t) \right) &= -\sum_j \sum_{w_t \in W_j} \pi_t H(p(C|w_t)) \\ &= -\sum_j \pi(W_j) \sum_{w_t \in W_j} \frac{\pi_t}{\pi(W_j)} H(p(C|w_t)) \ . \end{aligned} \tag{10}$$

Similarly, the second term in (9) may be written as

$$\begin{aligned} \sum_j \pi(W_j) \left(\sum_i p(c_i|W_j) \log p(c_i|W_j) \right) &= -\sum_j \pi(W_j) H(p(C|W_j)) \\ &= -\sum_j \pi(W_j) H \left(\sum_{w_t \in W_j} \frac{\pi_t}{\pi(W_j)} p(C|w_t) \right) \end{aligned} \tag{11}$$

where (11) is obtained by substituting the value of $p(C|W_j)$ from (8). Substituting (10) and (11) in (9) and using the definition of Jensen-Shannon divergence from (1) gives us the desired result. ■

Theorem 1 gives a global measure of the goodness of word clusters, which may be informally interpreted as follows:

1. The quality of word cluster W_j is measured by the Jensen-Shannon divergence between the individual word distributions $p(C|w_t)$ (weighted by the word priors, $\pi_t = p(w_t)$). The smaller the Jensen-Shannon divergence the more "compact" is the word cluster, i.e., smaller is the increase in entropy due to clustering (see (1)).

2. The overall goodness of the word clustering is measured by the sum of the qualities of individual word clusters (weighted by the cluster priors $\pi(W_j) = p(W_j)$).

Given the global criterion of Theorem 1, we would now like to find an algorithm that searches for the optimal word clustering that minimizes this criterion. We now rewrite this criterion in a way that will suggest a "natural" algorithm.

Lemma 2 *The generalized Jensen-Shannon divergence of a finite set of probability distributions can be expressed as the (weighted) sum of Kullback-Leibler divergences to the (weighted) mean, i.e.,*

$$JS_\pi(\{p_i : 1 \le i \le n\}) = \sum_{i=1}^{n} \pi_i KL(p_i, m) \tag{12}$$

where $\pi_i \ge 0, \sum_i \pi_i = 1$ and m is the (weighted) mean probability distribution, $m = \sum_i \pi_i p_i$.

Proof. Use the definition of entropy to expand the expression for JS-divergence given in (1). The result follows by appropriately grouping terms and using the definition of KL-divergence. ■

```
Algorithm Divisive_Information_Theoretic_Clustering($\mathcal{P}$,$\Pi$,$l$,$k$,$\mathcal{W}$)

Input: $\mathcal{P}$ is the set of distributions, $\{p(C|w_t) : 1 \le t \le m\}$,
    $\Pi$ is the set of all word priors, $\{\pi_t = p(w_t) : 1 \le t \le m\}$,
    $l$ is the number of document classes,
    $k$ is the number of desired clusters.

Output: $\mathcal{W}$ is the set of word clusters $\{\mathcal{W}_1, \mathcal{W}_2, \ldots, \mathcal{W}_k\}$.

  1. Initialization: for every word $w_t$, assign $w_t$ to $\mathcal{W}_j$ such that $p(c_j|w_t) = \max_i p(c_i|w_t)$. This
     gives $l$ initial word clusters; if $k \ge l$ split each cluster arbitrarily into at least $\lfloor k/l \rfloor$ clusters,
     otherwise merge the $l$ clusters to get $k$ word clusters.

  2. For each cluster $\mathcal{W}_j$, compute

        $\pi(\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \pi_t$,  and  $p(C|\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \frac{\pi_t}{\pi(\mathcal{W}_j)} p(C|w_t)$ .

  3. Re-compute all clusters: For each word $w_t$, find its new cluster index as

        $j^*(w_t) = \mathrm{argmin}_i \; KL(p(C|w_t), p(C|\mathcal{W}_i))$ ,

     resolving ties arbitrarily. Thus compute the new word clusters $\mathcal{W}_j$, $1 \le j \le k$, as

        $\mathcal{W}_j = \{w_t : j^*(w_t) = j\}$ .

  4. Stop if the change in objective function value given by (13) is "small" (say $10^{-3}$);
     Else go to step 2.
```

Figure 1: Information-Theoretic Divisive Algorithm for word clustering

5.1 The Algorithm

By Theorem 1 and Lemma 2, the decrease in mutual information due to word clustering may be written as

$$\sum_{j=1}^{k} \pi(\mathcal{W}_j) \sum_{w_t \in \mathcal{W}_j} \frac{\pi_t}{\pi(\mathcal{W}_j)} KL(p(C|w_t), p(C|\mathcal{W}_j)) \ .$$

As a result the quality of word clustering can be measured by the objective function

$$Q(\{\mathcal{W}_j\}_{j=1}^{k}) \ = \ I(C;W) - I(C;W^C) \ = \ \sum_{j=1}^{k} \sum_{w_t \in \mathcal{W}_j} \pi_t KL(p(C|w_t), p(C|\mathcal{W}_j)) \ . \qquad (13)$$

Note that it is natural that KL-divergence emerges as the distance measure in the above objective function since mutual information is just the KL-divergence between the joint distribution and the product distribution. Writing the objective function in the above manner suggests an iterative algorithm that repeatedly (i) re-partitions the distributions $p(C|w_t)$ by their closeness in KL-divergence to the cluster distributions $p(C|\mathcal{W}_j)$, and (ii) subsequently, given the new word clusters, re-computes these cluster distributions using (8). Figure 1 describes this Divisive Information-Theoretic Clustering algorithm in detail — note that our algorithm is easily extended to give a top-down hierarchy of clusters. Our divisive algorithm bears some resemblance to the *k*-means or Lloyd-Max algorithm, which usually uses squared Euclidean distances (also see Gray and Neuhoff, 1998, Berkhin and Becher, 2002, Vaithyanathan and Dom, 1999, Modha and Spangler, 2002, to appear). Also, just as the Euclidean *k*-means algorithm can be regarded as the "hard clustering" limit of the EM algorithm on a mixture of appropriate multivariate Gaussians, our divisive algorithm can also be regarded as a divisive version of the hard clustering limit of the "soft" Information Bottleneck algorithm of Tishby et al. (1999), which is an extension of the Blahut-Arimoto algorithm (Cover and Thomas, 1991). Note, however, that the previously proposed hard clustering limit of Information Bottleneck is the agglomerative algorithm of Slonim and Tishby (2001).

Our initialization strategy is important, see step 1 in Figure 1 (a similar strategy was used by Dhillon and Modha, 2001, Section 5.1, to obtain word clusters), since it guarantees that the support set of every $p(C|w_t)$ is contained in the support set of at least one cluster distribution $p(C|\mathcal{W}_j)$, i.e., guarantees that at least one KL-divergence for w_t is finite. This is because our initialization strategy ensures that every word w_t is part of some cluster $\mathcal{W}_j$. Thus by the formula for $p(C|\mathcal{W}_j)$ in step 2, it cannot happen that $p(c_i|w_t) \neq 0$, and $p(c_i|\mathcal{W}_j) = 0$. Note that we can still get some infinite KL-divergence values but these do not lead to any implementation difficulties (indeed in an implementation we can handle such "infinity problems" without an extra "if" condition thanks to the handling of "infinity" in the IEEE floating point standard defined by Goldberg 1991, ANS 1985).

We now discuss the computational complexity of our algorithm. Step 3 of each iteration requires the KL-divergence to be computed for every pair, $p(C|w_t)$ and $p(C|\mathcal{W}_j)$. This is the most computationally demanding task and costs a total of $O(mkl)$ operations. Thus the total complexity is $O(mkl\tau)$, which grows linearly with m (note that $k \ll m$) and the number of iterations, τ. Generally, we have found that the number of iterations required is 10-15. In contrast, the agglomerative algorithm of Slonim and Tishby (2001) costs $O(m^3 l)$ operations.

The algorithm in Figure 1 has certain pleasing properties. As we will prove in Theorem 5, our algorithm decreases the objective function value at every step and thus is guaranteed to terminate at a local minimum in a finite number of iterations (note that finding the global minimum is NP-complete, see Garey et al., 1982). Also, by Theorem 1 and (13) we see that our algorithm minimizes the "within-cluster" Jensen-Shannon divergence. It turns out that (see Theorem 6) our algorithm *simultaneously maximizes* the "between-cluster" Jensen-Shannon divergence. Thus the different word clusters produced by our algorithm are "maximally" far apart.

We now give formal statements of our results with proofs.

Lemma 3 *Given probability distributions $p_1, \dots, p_n$, the distribution that is closest (on average) in KL-divergence is the mean probability distribution m, i.e., given any probability distribution q,*

$$\sum_i \pi_i KL(p_i, q) \geq \sum_i \pi_i KL(p_i, m) \ , \tag{14}$$

where $\pi_i \geq 0$, $\sum_i \pi_i = 1$ and $m = \sum_i \pi_i p_i$.

Proof. Use the definition of KL-divergence to expand the left-hand side(LHS) of (14) to get

$$\sum_i \pi_i KL(p_i, q) \; = \; \sum_i \pi_i \sum_x p_i(x) \left(\log p_i(x) - \log q(x)\right) \quad .$$

Similarly the RHS of (14) equals

$$\pi_i KL(p_i, m) \; = \; \sum_i \pi_i \sum_x p_i(x) \left(\log p_i(x) - \log m(x)\right) \quad .$$

Subtracting the RHS from LHS leads to

$$\sum_i \pi_i \sum_x p_i(x) \left(\log m(x) - \log q(x)\right) = \sum_x m(x) \log \frac{m(x)}{q(x)} = KL(m, q) \quad .$$

The result follows since the KL-divergence is always non-negative (Cover and Thomas, 1991, Theorem 2.6.3). ■

Theorem 4 *The algorithm in Figure 1 monotonically decreases the value of the objective function given in (13).*

Proof. Let $W_1^{(i)}, \ldots, W_k^{(i)}$ be the word clusters at iteration i, and let $p(C|W_1^{(i)}), \ldots, p(C|W_k^{(i)})$ be the corresponding cluster distributions. Then

$$\begin{aligned}
Q(\{W_j^{(i)}\}_{j=1}^k) \; &= \; \sum_{j=1}^k \sum_{w_t \in W_j^{(i)}} \pi_t KL(p(C|w_t), p(C|W_j^{(i)})) \\
&\geq \; \sum_{j=1}^k \sum_{w_t \in W_j^{(i)}} \pi_t KL(p(C|w_t), p(C|W_{j^*(w_t)}^{(i)})) \\
&\geq \; \sum_{j=1}^k \sum_{w_t \in W_j^{(i+1)}} \pi_t KL(p(C|w_t), p(C|W_j^{(i+1)})) \\
&= \; Q(\{W_j^{(i+1)}\}_{j=1}^k)
\end{aligned}$$

where the first inequality is due to step 3 of the algorithm, and the second inequality follows from the parameter estimation in step 2 and from Lemma 3. Note that if equality holds, i.e., if the objective function value is equal at consecutive iterations, then step 4 terminates the algorithm. ■

Theorem 5 *The algorithm in Figure 1 terminates in a finite number of steps at a cluster assignment that is locally optimal, i.e., the loss in mutual information cannot be decreased by either (a) re-assignment of a word distribution $p(C|w_t)$ to a different class distribution $p(C|W_i)$, or by (b) defining a new class distribution for any of the existing clusters.*

Proof. The result follows since the algorithm monotonically decreases the objective function value, and since the number of distinct clusterings is finite (see Bradley and Mangasarian, 2000, for a similar argument). ■

We now show that the total Jensen-Shannon(JS) divergence can be written as the sum of two terms.

Theorem 6 *Let $p_1,\ldots,p_n$ be a set of probability distributions and let $\pi_1,\ldots,\pi_n$ be corresponding scalars such that $\pi_i \geq 0$, $\sum_i \pi_i = 1$. Suppose $p_1,\ldots,p_n$ are clustered into k clusters $\mathcal{P}_1,\ldots,\mathcal{P}_k$, and let m_j be the (weighted) mean distribution of $\mathcal{P}_j$, i.e.,*

$$m_j = \sum_{p_t \in \mathcal{P}_j} \frac{\pi_t}{\pi(\mathcal{P}_j)} p_t, \quad \text{where} \quad \pi(\mathcal{P}_j) = \sum_{p_t \in \mathcal{P}_j} \pi_t \ . \tag{15}$$

Then the total JS-divergence between $p_1,\ldots,p_n$ can be expressed as the sum of "within-cluster JS-divergence" and "between-cluster JS-divergence", i.e.,

$$JS_\pi(\{p_i : 1 \leq i \leq n\}) \ = \ \sum_{j=1}^{k} \pi(\mathcal{P}_j) JS_{\pi'}(\{p_t : p_t \in \mathcal{P}_j\}) + JS_{\pi''}(\{m_i : 1 \leq i \leq k\}) \ ,$$

where $\pi'_t = \pi_t/\pi(\mathcal{P}_j)$ and we use π'' as the subscript in the last term to denote $\pi''_j = \pi(\mathcal{P}_j)$.

Proof. By Lemma 2, the total JS-divergence may be written as

$$JS_\pi(\{p_i : 1 \leq i \leq n\}) \ = \ \sum_{i=1}^{n} \pi_i KL(p_i, m) \ = \ \sum_{i=1}^{n} \sum_{x} \pi_i p_i(x) \log \frac{p_i(x)}{m(x)} \tag{16}$$

where $m = \sum_i \pi_i p_i$. With m_j as in (15), and rewriting (16) in order of the clusters $\mathcal{P}_j$ we get

$$\begin{aligned} & \sum_{j=1}^{k} \sum_{p_t \in \mathcal{P}_j} \sum_{x} \pi_t p_t(x) \left(\log \frac{p_t(x)}{m_j(x)} + \log \frac{m_j(x)}{m(x)} \right) \\ &= \sum_{j=1}^{k} \pi(\mathcal{P}_j) \sum_{p_t \in \mathcal{P}_j} \frac{\pi_t}{\pi(\mathcal{P}_j)} KL(p_t, m_j) + \sum_{j=1}^{k} \pi(\mathcal{P}_j) KL(m_j, m) \\ &= \sum_{j=1}^{k} \pi(\mathcal{P}_j) JS_{\pi'}(\{p_t : p_t \in \mathcal{P}_j\}) + JS_{\pi''}(\{m_i : 1 \leq i \leq k\}) \ , \end{aligned}$$

where $\pi''_j = \pi(\mathcal{P}_j)$, which proves the result. ■

Our divisive algorithm explicitly minimizes the objective function in (13), which by Lemma 2 can be interpreted as the average within-cluster JS-divergence. Thus, since the total JS-divergence between the word distributions is constant, our algorithm also implicitly maximizes the between-cluster JS-divergence.

This concludes our formal treatment. We now see how to use word clusters in our text classifiers.

5.2 Classification using Word Clusters

The Naive Bayes method can be simply translated into using word clusters instead of words. This is done by estimating the new parameters $p(\mathcal{W}_s|c_i)$ for word clusters similar to the word parameters $p(w_t|c_i)$ in (4) as

$$p(\mathcal{W}_s|c_i) = \frac{\sum_{d_j \in c_i} n(\mathcal{W}_s, d_j)}{\sum_{s=1}^{k} \sum_{d_j \in c_i} n(\mathcal{W}_s, d_j)}$$

where $n(\mathcal{W}_s, d_j) = \sum_{w_t \in \mathcal{W}_s} n(w_t, d_j)$. Note that when estimates of $p(w_t|c_i)$ for individual words are

```
1. Sort the entire vocabulary by Mutual Information with the class variable and select top $M$ words (usually $M = 2000$).
2. Initialize $M$ singleton clusters with the top $M$ words.
3. Compute the inter-cluster distances between every pair of clusters.
4. Loop until $k$ clusters are obtained:
    • Merge the two clusters which are most similar (see (17)).
    • Update the inter-cluster distances.
```

Figure 2: Agglomerative Information Bottleneck Algorithm (Slonim and Tishby, 2001)

.

```
1. Sort the entire vocabulary by Mutual Information with the class variable.
2. Initialize $k$ singleton clusters with the top $k$ words.
3. Compute the inter-cluster distances between every pair of clusters.
4. Loop until all words have been put into one of the $k$ clusters:
    • Merge the two clusters which are most similar (see (17)) resulting in $k-1$ clusters.
    • Add a new singleton cluster consisting of the next word from the sorted list of words.
    • Update the inter-cluster distances.
```

Figure 3: Agglomerative Distributional Clustering Algorithm (Baker and McCallum, 1998)

.

relatively poor, the corresponding word cluster parameters $p(W_s|c_i)$ provide more robust estimates resulting in higher classification scores.

The Naive Bayes rule (5) for classifying a test document d can be rewritten as

$$c^*(d) = \text{argmax}_{c_i}\left(\frac{\log p(c_i)}{|d|} + \sum_{s=1}^{k} p(W_s|d)\log p(W_s|c_i)\right) \quad ,$$

where $p(W_s|d) = n(W_s|d)/|d|$. Support Vector Machines can be similarly used with word clusters as features.

5.3 Previous Word Clustering Approaches

Previously two agglomerative algorithms have been proposed for distributional clustering of words applied to text classification. In this section we give details of their approaches.

Figures 2 and 3 give brief outlines of the algorithms proposed by Slonim and Tishby (2001) and Baker and McCallum (1998) respectively. For simplicity we will refer to the algorithm in

Figure 2 as "Agglomerative Information Bottleneck" (AIB) and the algorithm in Figure 3 as "Agglomerative Distributional Clustering" (ADC). AIB is strictly agglomerative in nature resulting in high computational cost. Thus, AIB first selects M features (M is generally much smaller than the total vocabulary size) and then runs an agglomerative algorithm until k clusters are obtained ($k \ll M$). In order to reduce computational complexity so that it is feasible to run on the full feature set, ADC uses an alternate strategy. ADC uses the entire vocabulary but maintains only k word clusters at any instant. A merge of two of these clusters results in $k-1$ clusters after which a singleton cluster is created to get back k clusters (see Figure 3 for details). Incidentally both algorithms use the following **identical** merging criterion for merging two word clusters $\mathcal{W}_i$ and $\mathcal{W}_j$:

$$\begin{aligned}\delta I(\mathcal{W}_i, \mathcal{W}_j) &= p(\mathcal{W}_i)KL(p(C|\mathcal{W}_i), p(C|\hat{\mathcal{W}})) + p(\mathcal{W}_j)KL(p(C|\mathcal{W}_j), p(C|\hat{\mathcal{W}})) \\ &= (p(\mathcal{W}_i) + p(\mathcal{W}_j))JS_\pi(p(C|\mathcal{W}_i), p(C|\mathcal{W}_j)), \qquad (17)\end{aligned}$$

where $\hat{\mathcal{W}}$ refers to the merged cluster and $p(C|\hat{\mathcal{W}}) = \pi_i p(C|\mathcal{W}_i) + \pi_j p(C|\mathcal{W}_j)$, $\pi_i = p(\mathcal{W}_i)/(p(\mathcal{W}_i) + p(\mathcal{W}_j))$, and $\pi_j = p(\mathcal{W}_j)/(p(\mathcal{W}_i) + p(\mathcal{W}_j))$.

Computationally both the agglomerative approaches are expensive. The complexity of AIB is $O(M^3 l)$ while that of ADC is $O(mk^2 l)$ where m is the number of words and l is the number of classes in the data set (typically $k, l \ll m$). Moreover both these agglomerative approaches are greedy in nature and do a local optimization. In contrast our divisive clustering algorithm is computationally superior, $O(mkl\tau)$, and optimizes not just across two clusters but over **all** clusters simultaneously.

6. Experimental Results

This section provides empirical evidence that our divisive clustering algorithm of Figure 1 outperforms various feature selection methods and previous agglomerative clustering approaches. We compare our results with feature selection by Information Gain and Mutual Information (Yang and Pedersen, 1997), and feature clustering using the agglomerative algorithms of Baker and McCallum (1998) and Slonim and Tishby (2001). As noted in Section 5.3 we will use AIB to denote "Agglomerative Information Bottleneck" and ADC to denote "Agglomerative Distributional Clustering". It is computationally infeasible to run AIB on the entire vocabulary, so as suggested by Slonim and Tishby (2001), we use the top 2000 words based on the mutual information with the class variable. We denote our algorithm by "Divisive Clustering" and show that it achieves higher classification accuracies than the best performing feature selection method, especially when training data is *sparse* and show improvements over similar results reported by using AIB (Slonim and Tishby, 2001).

6.1 Data Sets

The *20 Newsgroups (20Ng)* data set collected by Lang (1995) contains about 20,000 articles evenly divided among 20 UseNet Discussion groups. Each newsgroup represents one class in the classification task. This data set has been used for testing several text classification methods (Baker and McCallum, 1998, Slonim and Tishby, 2001, McCallum and Nigam, 1998). During indexing we skipped headers but retained the subject line, pruned words occurring in less than 3 documents and used a stop list but did not use stemming. After converting all letters to lowercase the resulting vocabulary had 35,077 words.

We collected the *Dmoz* data from the Open Directory Project (www.dmoz.org). The Dmoz hierarchy contains about 3 million documents and 300,0000 classes. We chose the top *Science*

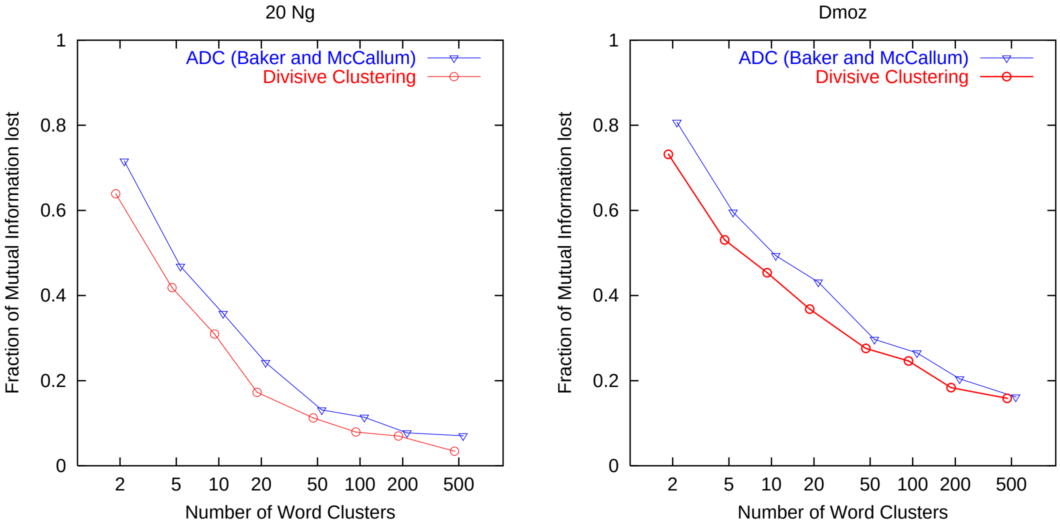

Figure 4: Fraction of Mutual Information lost while clustering words with Divisive Clustering is significantly lower compared to ADC at all feature sizes (on 20Ng and Dmoz data).

category and crawled some of the heavily populated internal nodes beneath it, resulting in a 3-deep hierarchy with 49 leaf-level nodes, 21 internal nodes and about 5,000 total documents. For our experimental results we ignored documents at internal nodes. While indexing, we skipped the text between html tags, pruned words occurring in less than five documents, used a stop list but did not use stemming. After converting all letters to lowercase the resulting vocabulary had 14,538 words.

6.2 Implementation Details

Bow (McCallum, 1996) is a library of C code useful for writing text analysis, language modeling and information retrieval programs. We extended Bow to index BdB (www.sleepycat.com) flat file databases where we stored the text documents for efficient retrieval and storage. We implemented the agglomerative and divisive clustering algorithms within Bow and used Bow's SVM implementation in our experiments. To perform hierarchical classification, we wrote a Perl wrapper to invoke Bow subroutines. For crawling www.dmoz.org we used libwww libraries from the W3C consortium.

6.3 Results

We first give evidence of the improved quality of word clusters obtained by our algorithm as compared to the agglomerative approaches. We define the fraction of mutual information lost due to clustering words as:

$$\frac{I(C;W) - I(C;W^C)}{I(C;W)} .$$

Intuitively, lower the loss in mutual information the better is the clustering. The term $I(C;W) - I(C;W^C)$ in the numerator of the above equation is precisely the global objective function that Divisive Clustering attempts to minimize (see Theorem 1). Figure 4 plots the fraction of mutual information lost against the number of clusters for Divisive Clustering and ADC algorithms on

20Ng and Dmoz data sets. Notice that less mutual information is lost with Divisive Clustering compared to ADC at *all* number of clusters, though the difference is more pronounced at lower number of clusters. Note that it is not meaningful to compare against the mutual information lost in AIB since the latter method works on a pruned set of words (2000) due to its high computational cost.

Next we provide some anecdotal evidence that our word clusters are better at preserving class information as compared to the agglomerative approaches. Figure 5 shows five word clusters, Clusters 9 and 10 from Divisive Clustering, Clusters 8 and 7 from AIB and Cluster 12 from ADC. These clusters were obtained while forming 20 word clusters with a $1/3$-$2/3$ test-train split (note that word clustering is done only on the training data). While the clusters obtained by our algorithm and AIB could successfully distinguish between *rec.sport.hockey* and *rec.sport.baseball*, ADC combined words from both classes in a single word cluster. This resulted in lower classification accuracy for both classes with ADC compared to Divisive Clustering. While Divisive Clustering achieved *93.33% and 94.07%* accuracy on *rec.sport.hockey* and *rec.sport.baseball* respectively, ADC could only achieve *76.97% and 52.42%*. AIB achieved *89.7% and 87.27%* respectively — these lower accuracies appear to be due to the initial pruning of the word set to 2000.

Divisive Clustering		ADC (Baker & McCallum)		AIB (Slonim & Tishby)	
Cluster 10	Cluster 9	Cluster12		Cluster 8	Cluster 7
(Hockey)	(Baseball)	(Hockey and Baseball)		(Hockey)	(Baseball)
team	hit	team	detroit	goals	game
game	runs	hockey	pitching	buffalo	minnesota
play	baseball	games	hitter	hockey	bases
hockey	base	players	rangers	puck	morris
season	ball	baseball	nyi	pit	league
boston	greg	league	morris	vancouver	roger
chicago	morris	player	blues	mcgill	baseball
pit	ted	nhl	shots	patrick	hits
van	pitcher	pit	vancouver	ice	baltimore
nhl	hitting	buffalo	ens	coach	pitch

Figure 5: Top few words sorted by Mutual Information in Clusters obtained by Divisive Clustering, ADC and AIB on 20 Newsgroups data.

6.3.1 CLASSIFICATION RESULTS ON 20 NEWSGROUPS DATA

Figure 6.3 shows the classification accuracy results on the 20 Newsgroups data set for Divisive Clustering and the feature selection algorithms considered. The vertical axis indicates the percentage of test documents that are classified correctly while the horizontal axis indicates the number of features/clusters used in the classification model. For the feature selection methods, the features are ranked and only the top ranked features are used in the corresponding experiment. The results are averages of 10 trials of randomized $1/3$-$2/3$ test-train splits of the total data. Note that we cluster only the words belonging to the documents in the training set. We used two classification

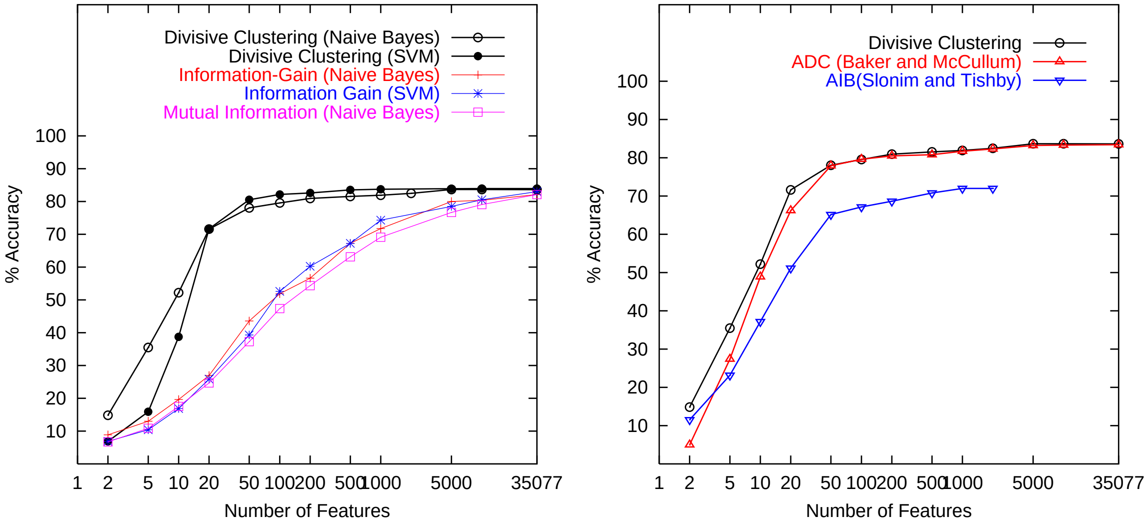

Figure 6: 20 Newsgroups data with 1/3-2/3 test-train split. (left) Classification Accuracy (right) Divisive Clustering vs. Agglomerative approaches (with Naive Bayes).

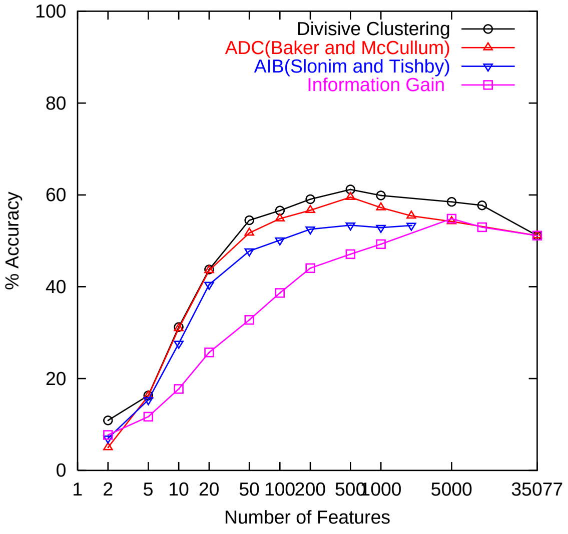

Figure 7: Classification Accuracy on 20 Newsgroups with 2% Training data (using Naive Bayes).

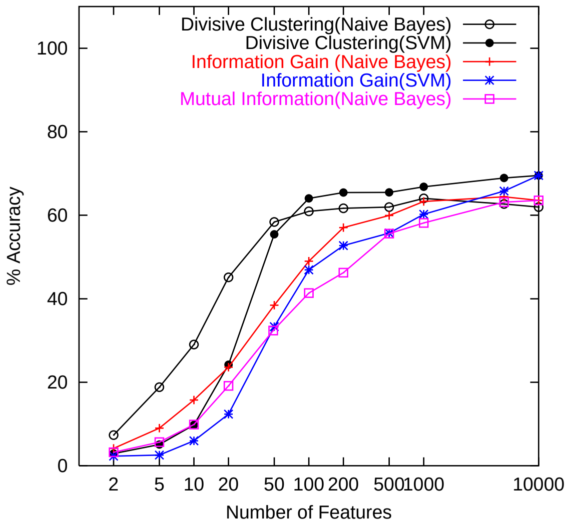

Figure 8: Classification Accuracy on Dmoz data with 1/3-2/3 test-train split.

techniques, SVMs and Naive Bayes (NB) for the purpose of comparison. Observe that Divisive Clustering (SVM and NB) achieves significantly better results at lower number of features than the Feature Selection methods Information Gain and Mutual Information. With only 50 clusters Divisive Clustering (NB) achieves 78.05% accuracy just 4.1% short of the accuracy achieved by a full feature NB classifier. We also observed that the largest gain occurs when the number of clusters equals the number of classes (for 20Ng data this occurs at 20 clusters). When we manually viewed these word clusters we found that many of them contained words representing a single class in the data set, for example see Figure 5. We attribute this observation to our effective initialization strategy.

Figure 6.3 compares the classification accuracies of Divisive Clustering and Agglomerative approaches on the 20 Newsgroups data using Naive Bayes and 1/3-2/3 test-train split. Notice that Divisive Clustering achieves either better or similar classification results than Agglomerative approaches at all feature sizes, though again the improvements are significant at lower number of features. ADC performs close to Divisive Clustering while AIB is consistently poorer. We hypothesize that the latter is due to the pruning of features to 2000 while using AIB.

A note here about the running times of ADC and Divisive Clustering. On a typical run on 20Ng data with 1/3-2/3 test-train split for obtaining 100 clusters from 35077 words, ADC took 80.16 minutes while Divisive Clustering ran in just 2.29 minutes. Thus, in terms of computational times, Divisive Clustering is much superior than the agglomerative algorithms.

In Figure 7, we plot the classification accuracy on 20Ng data using Naive Bayes when the training data is **sparse**. We took 2% of the available data, that is 20 documents per class for training and tested on the remaining 98% of the documents. The results are averages of 10 trials. We again observe that Divisive Clustering obtains better results than Information Gain at all number of features. It also achieves a significant 12% increase over the maximum possible accuracy achieved by Information Gain. This is in contrast to larger training data, where Information Gain eventually catches up as we increase the number of features. When the training data is small the word-by-class frequency matrix contains many zero entries. By clustering words we obtain more robust estimates of word class probabilities which lead to higher classification accuracies. This is the reason why all word clustering approaches (Divisive Clustering, ADC and AIB) perform better than Information Gain. While ADC is close to Divisive Clustering in performance, AIB is relatively poorer.

6.3.2 CLASSIFICATION RESULTS ON DMOZ DATA SET

Figure 8 shows the classification results for the *Dmoz* data set when we build a flat classifier over the leaf set of classes. Unlike the previous plots, feature selection here improves the classification accuracy since web pages appear to be inherently noisy. We observe results similar to those obtained on 20 Newsgroups data, but note that Information Gain(NB) here achieves a slightly higher maximum, about 1.5% higher than the maximum accuracy observed with Divisive Clustering(NB). Baker and McCallum (1998) tried a combination of feature-clustering and feature-selection methods to overcome this. More rigorous approaches to this problem are a topic of future work. Further note that SVMs fare worse than NB at low dimensionality but better at higher dimensionality. In future work we plan to use non-linear SVMs at lower dimensions to alleviate this problem.

Figure 9 plots the classification accuracy on Dmoz data using Naive Bayes when the training set is just 2%. Note again that we achieve a 13% increase in classification accuracy with Divisive Clustering over the maximum possible with Information Gain. This reiterates the observation that

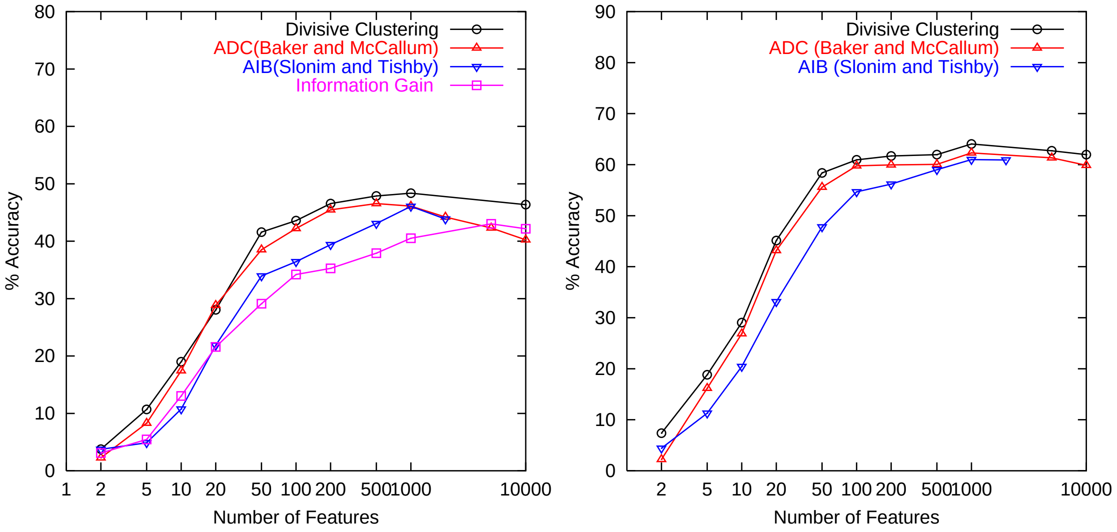

Figure 9: (left) Classification Accuracy on Dmoz data with 2% Training data (using Naive Bayes). (right) Divisive Clustering versus Agglomerative approaches on Dmoz data (1/3-2/3 test train split with Naive Bayes).

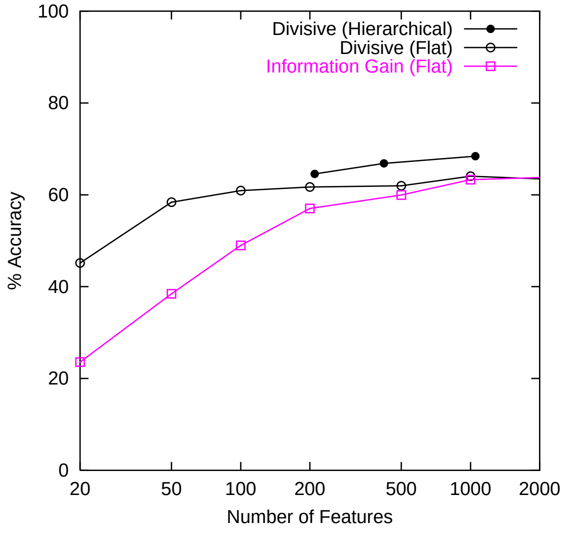

Figure 10: Classification results on Dmoz Hierarchy using Naive Bayes. Observe that the Hierarchical Classifier achieves significant improvements over the Flat classifiers with very few number of features per internal node.

feature clustering is an attractive option when training data is limited. AIB and ADC too outperform Information Gain but Divisive Clustering achieves slightly better results (see Figures 9 and 9).

6.3.3 HIERARCHICAL CLASSIFICATION ON DMOZ HIERARCHY

Figure 10 shows the classification accuracies obtained by three different classifiers on Dmoz data (Naive Bayes was the underlying classifier). By *Flat*, we mean a classifier built over the leaf set of classes in the tree. In contrast, *Hierarchical* denotes a hierarchical scheme that builds a classifier at each internal node of the topic hierarchy (see Section 4.3). Further we apply Divisive Clustering at each internal node to reduce the number of features in the classification model at that node. The number of word clusters is the same at each internal node.

Figure 10 compares the Hierarchical Classifier with two flat classifiers, one that employs Information Gain for feature selection while the other uses Divisive Clustering. A note about how to interpret the number of features for the Hierarchical Classifier. Since we are comparing a Flat Classifier with Hierarchical Classifier we need to be fair regarding the number of features used by the classifiers. If we use 10 features at each internal node of the Hierarchical Classifier we denote that as 210 features in Figure 10 since we have 21 internal nodes in our data set. Observe that Divisive Clustering performs remarkably well for Hierarchical Classification even at very low number of features. With just 10 (210 total) features, Hierarchical Classifier achieves 64.54% accuracy, which is slightly better than the maximum obtained by the two flat classifiers at any number of features. At 50 (1050 total) features, Hierarchical Classifier achieves 68.42%, a significant 6% higher than the maximum obtained by the flat classifiers. Thus Divisive Clustering appears to be a natural choice for feature reduction in case of hierarchical classification as it allows us to maintain high classification accuracies at very small number of features.

7. Conclusions and Future Work

In this paper, we have presented an information-theoretic approach to “hard” word clustering for text classification. First, we derived a global objective function to capture the decrease in mutual information due to clustering. Then we presented a divisive algorithm that directly minimizes this objective function, converging to a local minimum. Our algorithm minimizes the within-cluster Jensen-Shannon divergence, and simultaneously maximizes the between-cluster Jensen-Shannon divergence.

Finally, we provided an empirical validation of the effectiveness of our word clustering. We have shown that our divisive clustering algorithm is much faster than the agglomerative strategies proposed previously by Baker and McCallum (1998), Slonim and Tishby (2001) and obtains better word clusters. We have presented detailed experiments using the Naive Bayes and SVM classifiers on the 20 Newsgroups and Dmoz data sets. Our enhanced word clustering results in improvements in classification accuracies especially at lower number of features. When the training data is sparse, our feature clustering achieves higher classification accuracy than the maximum accuracy achieved by feature selection strategies such as information gain and mutual information. Thus our divisive clustering method is an effective technique for reducing the model complexity of a hierarchical classifier.

In future work we intend to conduct experiments at a larger scale on hierarchical web data to evaluate the effectiveness of the resulting hierarchical classifier. We also intend to explore local search strategies (such as in Dhillon et al., 2002) to increase the quality of the local optimal achieved by our divisive clustering algorithm. Furthermore, our information-theoretic clustering algorithm can be applied to other applications that involve non-negative data.

An important topic for exploration is the choice of the number of word clusters to be used for the classification task. We intend to apply the MDL principle for this purpose (Rissanen, 1989). Reducing the number of features makes it feasible to run computationally expensive classifiers such as SVMs on large collections. While soft clustering increases the model size, it is not clear how it affects classification accuracy. In future work, we would like to experimentally evaluate the tradeoff between soft and hard clustering. Other directions for exploration include feature weighting and combination of feature selection and clustering strategies.

Acknowledgments

We are grateful to Byron Dom for many helpful discussions and to Andrew McCallum for making the Bow software library (McCallum, 1996) publicly available. For this research, Inderjit Dhillon was supported by a NSF CAREER Grant (No. ACI-0093404) while Subramanyam Mallela was supported by a UT Austin MCD Fellowship.

References

IEEE Standard for Binary Floating Point Arithmetic. ANSI/IEEE, New York, Std 754-1985 edition, 1985.

L. D. Baker and A. McCallum. Distributional clustering of words for text classification. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR*, pages 96–103. ACM, August 1998.

R. Bekkerman, R. El-Yaniv, Y. Winter, and N. Tishby. On feature distributional clustering for text categorization. In *ACM SIGIR*, pages 146–153, 2001.

P. Berkhin and J. D. Becher. Learning simple relations: Theory and applications. In *Proceedings of the The Second SIAM International Conference on Data Mining*, pages 420–436, 2002.

B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *COLT*, pages 144–152, 1992.

P. S. Bradley and O. L. Mangasarian. k-plane clustering. *Journal of Global Optimization*, 16(1): 23–32, 2000.

S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan. Using taxonomy, discriminants, and signatures for navigating in text databases. In *Proceedings of the 23rd VLDB Conference, Athens, Greece*, 1997.

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

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

I. S. Dhillon, Y. Guan, and J. Kogan. Iterative clustering of high dimensional text data augmented by local search. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, 2002. To Appear.

I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, January 2001.

P. Domingos and M. J. Pazzani. On the the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

S. T. Dumais and H. Chen. Hierarchical classification of web content. In *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 256–263, Athens, GR, 2000. ACM Press, New York, US.

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.

J. H. Friedman. On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1:55–77, 1997.

M. R. Garey, D. S. Johnson, and H. S. Witsenhausen. The complexity of the generalized Lloyd-Max problem. *IEEE Trans. Inform. Theory*, 28(2):255–256, 1982.

D. Goldberg. What every computer scientist should know about floating point arithmetic. *ACM Computing Surveys*, 23(1), 1991.

R. M. Gray and D. L. Neuhoff. Quantization. *IEEE Trans. Inform. Theory*, 44(6):1–63, 1998.

T. Hofmann. Probabilistic latent semantic indexing. In *Proc. ACM SIGIR*. ACM Press, August 1999.

T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, number 1398 in Lecture Notes in Computer Science, pages 137–142. Springer Verlag, Heidelberg, DE, 1998.

D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML-97)*, 1997.

S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Stat.*, 22:79–86, 1951.

K. Lang. News Weeder: Learning to filter netnews. In *Proc. 12th Int'l Conf. Machine Learning*, San Francisco, 1995.

J. Lin. Divergence measures based on the Shannon entropy. *IEEE Trans. Inform. Theory*, 37(1): 145–151, 1991.

A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

T. M. Mitchell. Conditions for the equivalence of hierarchical and non-hierarchical bayesian classifiers. Technical report, Center for Automated Learning and Discovery, Carnegie-Mellon University, 1998.

D. S. Modha and W. S. Spangler. Feature weighting in *k*-means clustering. *Machine Learning*, 2002, to appear.

F. Pereira, N. Tishby, and L. Lee. Distributional clustering of English words. In *31st Annual Meeting of the ACL*, pages 183–190, 1993.

J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. Series in Computer Science -Vol. 15. World Scientific, Singapore, 1989.

G. Salton and M. J. McGill. *Introduction to Modern Retrieval*. McGraw-Hill Book Company, 1983.

C. E. Shannon. A mathematical theory of communication. *Bell System Technical J.*, 27:379–423, 1948.

N. Slonim and N. Tishby. The power of word clusters for text classification. In *23rd European Colloquium on Information Retrieval Research (ECIR)*, 2001.

N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

S. Vaithyanathan and B. Dom. Model selection in unsupervised learning with applications to document clustering. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML), Bled, Slovenia*. Morgan Kaufman, June 1999.

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

J. Verbeek. An information theoretic approach to finding word groups for text classification. Master's thesis, Institute for Logic, Language and Computation (ILLC-MoL-2000-03), Amsterdam, The Netherlands, September 2000.

Y. Yang and X. Liu. A re-examination of text categorization methods. In *ACM SIGIR*, pages 42–49, 1999.

Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proc. 14th International Conference on Machine Learning*, pages 412–420. Morgan Kaufmann, 1997.