Enhanced Word Clustering for Hierarchical Text Classification

Inderjit S. Dhillon
Dept. of Computer Sciences
Univ. of Texas, Austin
inderjit@cs.utexas.edu

Subramanyam Mallela
Dept. of Computer Sciences
Univ. of Texas, Austin
manyam@cs.utexas.edu

Rahul Kumar
Dept. of Computer Sciences
Univ. of Texas, Austin
rahul@cs.utexas.edu

ABSTRACT

In this paper we propose a new information-theoretic divisive algorithm for word clustering applied to text classification. In previous work, such "distributional clustering" of features has been found to achieve improvements over feature selection in terms of classification accuracy, especially at lower number of features [2, 28]. However the existing clustering techniques 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, thus converging to a local minimum. 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 achieves higher classification accuracy especially at lower number of features. 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 20 Newsgroups data set and a 3-level hierarchy of HTML documents collected from Dmoz Open Directory.

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 [24, 10, 29].

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[26]. 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) 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 [25, 2, 28]. Each word cluster can then be treated as a single feature and thus dimensionality can be drastically reduced. As shown by [2, 28], such feature clustering is more effective than feature selection[30], especially at lower number of features. Also, feature clustering appears to preserve classification accuracy as compared to 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 given in both [2] and [28] are agglomerative in nature yielding sub-optimal word clusters at a high computational cost.

In this paper, we first derive a global criterion that captures the optimality of word clustering in an information-theoretic framework. This leads to an objective function for clustering that is based on the generalized Jensen-Shannon divergence[20] among an arbitrary number of 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[19] instead of squared Euclidean distances. We prove that our divisive algorithm *monotonically* decreases the objective function value, thus converging to a local minimum. 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 [2, 28]. 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 in-

Paper to appear in KDD 2002 Edmonton, Alberta, CA

formation 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 results about the algorithm, such as convergence and simultaneous maximization of "between-cluster JS-divergence". In Section 6 we present experimental results that show the superiority 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 $p(X)$, $p(C|w_t)$, etc. to make the random variable explicit.

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 [26]. A simple but effective algorithm is the Naive Bayes method [24]. For text classification, different variants of Naive Bayes have been used, but McCallum and Nigam [21] showed that the variant based on the multinomial model leads to better results. 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 in [18, 5]. For more details, see Section 4.

To counter high-dimensionality various methods of feature selection have been proposed in [30, 18, 5]. Distributional clustering of words was first proposed by Pereira, Tishby & Lee in [25] where they used "soft" distributional clustering 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 & McCallum used such hard clustering in [2], while more recently, Slonim & Tishby have used the so-called Information Bottleneck method for clustering words in [28]. Both [2] & [28] 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 much loss in classification accuracy using Naive Bayes. Similar results have been reported for SVMs[3].

Two other dimensionality/feature reduction schemes are used in latent semantic indexing (LSI) [7] and its probabilistic version [16]. Typically these methods have been applied in the *unsupervised* setting and as shown in [2], 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 derive a global criterion that explicitly captures the optimality of word clusters in an information theoretic framework. This leads to an objective function in terms of the generalized Jensen-Shannon divergence between an arbitrary number of 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 [28] which 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 Baker and McCallum's algorithm [2], greedily optimizes this merging criterion. Thus, their resulting algorithm can yield sub-optimal clusters and is computationally expensive (the algorithm in [28] is $O(m^3 l)$ in complexity where m is the total number of words and l is the number of classes). In contrast our divisive algorithm is $O(mkl)$ where k is the number of word clusters required (typically $k \ll m$). Note that our hard clustering leads to a model size of $O(k)$, whereas "soft" clustering in methods such as probabilistic LSI [16] 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. 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 & Thomas [6].

Let X be a discrete random variable that takes on values from the set $\mathcal{X}$ with probability distribution $p(x)$. The (Shannon) entropy of X [27] is defined as

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

The relative entropy or Kullback-Leibler(KL) divergence [19] between two 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 [6, 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 [20]. The JS-divergence can be generalized to measure the distance between any finite number of probability distributions as:

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

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 distribution $p(x,y)$ and the product distribution $p(x)p(y)$:

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

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: $I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$ [6].

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. We now give details on these classifiers.

4.1 Naive Bayes Classifier

Let $\mathcal{C} = \{c_1, c_2, \ldots, c_l\}$ be the set of l classes, and let $\mathcal{W} = \{w_1, \ldots, w_m\}$ 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,

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

Assuming a generative multinomial model [21] and further assuming class-conditional independence of words yields the well-known Naive Bayes classifier [24], which computes the most probable class for d as

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

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:

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

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

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

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

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

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 $\mathcal{W}$. Thus, assuming equal class priors, we see that Naive Bayes may be interpreted as finding the class 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 in [9, 12].

4.2 Support Vector Machines

Support Vector Machines(SVMs)[29] are inductive learning schemes for solving the two class pattern recognition problem. Recently SVMs have been shown to give good results for text categorization [17]. 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 the two classes. In the case of linearly separable data, the decision surface is a hyperplane that maximizes the "margin" between the two classes. This hyperplane can be written as $\vec{w}.\vec{x} - b = 0$, where $\vec{x}$ is a data point and the vector $\vec{w}$ and 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 $\vec{x_i}$. Finding the hyperplane can be translated into the following optimization problem

$$\begin{aligned} & Minimize : ||\vec{w}|| \\ \text{subject to} \quad & \vec{w}.\vec{x_i} - b \geq +1 \quad for \quad y_i = +1, \\ & \vec{w}.\vec{x_i} - b \leq -1 \quad for \quad y_i = -1. \end{aligned}$$

This minimization problem can be solved using quadratic programming techniques[29]. 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[29]. Even though SVM classifiers are described as binary classifiers they can be easily combined to handle the multi class case. 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 because they are fast to learn and classify new instances compared to non-linear SVMs. Further, linear SVMs have been shown to do well on text classification[17].

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 of *all* features are used[23]. However, hierarchical classification along with feature selection has been shown to achieve better classification results than a flat classifier[18]. 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. In order to judge the quality of the word clusters we now introduce 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 clustering usually lowers mutual information. Thus we would like to find a clustering that minimizes the decrease in mutual information, $I(C;W) - I(C;W^C)$. The following theorem states that this 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

$$I(C;W) = \sum_i \sum_t \pi_t p(c_i|w_t) \log \frac{p(c_i|w_t)}{p(c_i)}$$

$$\text{and} \quad I(C;W^C) = \sum_i \sum_j \pi(\mathcal{W}_j) p(c_i|\mathcal{W}_j) \log \frac{p(c_i|\mathcal{W}_j)}{p(c_i)}.$$

We are interested in hard clustering, so $\pi(\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \pi_t$ and $p(c_i|\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} (\pi_t / \pi(\mathcal{W}_j)) p(c_i|w_t)$, 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 $p(C|\mathcal{W}_j)$ is the (weighted) mean distribution of the constituent distributions $p(C|w_t)$. Thus,

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

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

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

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

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

where (11) is obtained by substituting the value of $p(C|\mathcal{W}_j)$ 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 $\mathcal{W}_j$ 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 $\pi(\mathcal{W}_j) = p(\mathcal{W}_j)$).

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 1. *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.,*

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

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

Algorithm Divisive_KL_Clustering($\mathcal{P}$,Π,l,k,$\mathcal{W}$)

Input: $\mathcal{P}$ is the set of distributions, $\{p(C|w_t) : 1 \leq t \leq m\}$,
Π is the set of all word priors, $\{\pi_t = p(w_t) : 1 \leq t \leq m\}$
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 $k \geq l$ split each cluster into approximately k/l clusters, otherwise merge the l clusters to get k word clusters.

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

$$\pi(\mathcal{W}_j) = \sum_{w_t \in \mathcal{W}_j} \pi_t$$

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

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

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

resolving ties arbitrarily. Thus compute the new word clusters $\mathcal{W}_j$, $1 \leq j \leq k$, 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: Divisive Algorithm for word clustering based on KL-divergences

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. □

5.1 The Algorithm

By Theorem 1 and Lemma 1, 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

$$Q(\{\mathcal{W}_j\}_{j=1}^k) = I(C;W) - I(C;W^C)$$

$$= \sum_{j=1}^{k} \sum_{w_t \in \mathcal{W}_j} \pi_t KL(p(C|w_t), p(C|\mathcal{W}_j)). \quad (13)$$

Note that it is natural that the 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 (see Section 3). 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 the algorithm in detail. Note that this divisive algorithm bears some resemblance to the k-means or Lloyd-Max algorithm, which usually uses squared Euclidean distances [11, 10, 15, 4].

Note that our initialization strategy is crucial to our algorithm, see step 1 in Figure 1 (also see [8, Section 5.1]) since it guarantees absolute continuity of each $p(C|w_t)$ with at least one cluster distribution $p(C|\mathcal{W}_j)$, i.e., guarantees that at least one KL-divergence 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 difficulty (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 [14, 1]).

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. Generally, we have found that the algorithm converges in 10-15 iterations independent of the size of the data set. Thus the total complexity is $O(mkl)$, which grows linearly with m (note that $k \ll m$). In contrast, the agglomerative algorithm of [28] costs $O(m^3 l)$ operations.

The algorithm in Figure 1 has certain pleasing properties. As we will prove in Theorem 3, our algorithm decreases the objective function value at every step and thus is guaranteed to converge to a local minimum in a finite number of steps (note that finding the global minimum is NP-complete[13]). Also, by Theorem 1 and (13) we see that our algorithm minimizes the "within-cluster" Jensen-Shannon divergence. It turns out that (see Theorem 4) that 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 2. *Given probability distributions $p_1, \ldots, p_n$, the distribution that is closest (on average) in KL-divergence is the mean probability distribution m, i.e., given any probability distribution q,*

$$\sum_i \pi_i KL(p_i, q) \geq \sum_i \pi_i KL(p_i, m), \quad (14)$$

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

$$\sum_i \pi_i \sum_x p_i(x) \left(\log p_i(x) - \log q(x)\right).$$

Similarly the RHS of (14) equals

$$\sum_i \pi_i \sum_x p_i(x) \left(\log p_i(x) - \log m(x)\right).$$

Subtracting the RHS from LHS leads to

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

The result follows since the KL-divergence is always non-negative [6, Theorem 2.6.3]. □

THEOREM 2. *The Algorithm in Figure 1 monotonically decreases the value of the objective function given in (13).*

Proof. Let $\mathcal{W}_1^{(i)}, \ldots, \mathcal{W}_k^{(i)}$ be the word clusters at iteration i, and let $p(C|\mathcal{W}_1^{(i)}), \ldots, p(C|\mathcal{W}_k^{(i)})$ be the corresponding cluster distributions. Then

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

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

THEOREM 3. *The Algorithm in Figure 1 always converges to a local minimum in a finite number of iterations.*

Proof. The result follows since the algorithm monotonically decreases the objective function value, which is bounded from below (by zero). □

We now show that the total Jensen-Shannon divergence (which is constant for a given set of probability distributions) can be written as the sum of two terms, one of which is the objective function (13) that our algorithm minimizes.

THEOREM 4. *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.,*

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

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.,

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

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

Proof. By Lemma 1, the total JS-divergence may be written as

$$\begin{aligned} JS_\pi(\{p_i : 1 \leq i \leq n\}) &= \sum_{i=1}^n \pi_i KL(p_i, m) \\ &= \sum_{i=1}^n \sum_x \pi_i p_i(x) \log \frac{p_i(x)}{m(x)} \quad (16) \end{aligned}$$

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

$$\sum_{j=1}^k \sum_{p_t \in \mathcal{P}_j} \sum_x \pi_t p_t(x) \left(\log \frac{p_t(x)}{m_j(x)} + \log \frac{m_j(x)}{m(x)} \right)$$

$$= \sum_{j=1}^k \pi(\mathcal{P}_j) \sum_{p_t \in \mathcal{P}_j} \frac{\pi_t}{\pi(\mathcal{P}_j)} KL(p_t, m_j) + \sum_{j=1}^k \pi(\mathcal{P}_j) KL(m_j, m)$$

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

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

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 relatively poor, the corresponding word cluster parameters $p(\mathcal{W}_s|c_i)$ provide more robust estimates resulting in higher classification scores.

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

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

where $p(\mathcal{W}_s|d) = n(\mathcal{W}_s|d)/|d|$. SVMs can be similarly used with word clusters as features.

6. EXPERIMENTAL RESULTS

This section provides empirical evidence that our divisive clustering algorithm of Figure 1 outperforms agglomerative clustering and various feature selection methods. We compare our results with feature selection by Information Gain and Mutual Information[30], and feature clustering using the agglomerative algorithm in [2]. We call the latter Agglomerative Clustering in this section for the purpose of comparison with our algorithm which we call Divisive Clustering. We show that Divisive Clustering achieves higher classification accuracy than the best performing feature selection method when the training data is *sparse* and show improvements over similar results reported in [28].

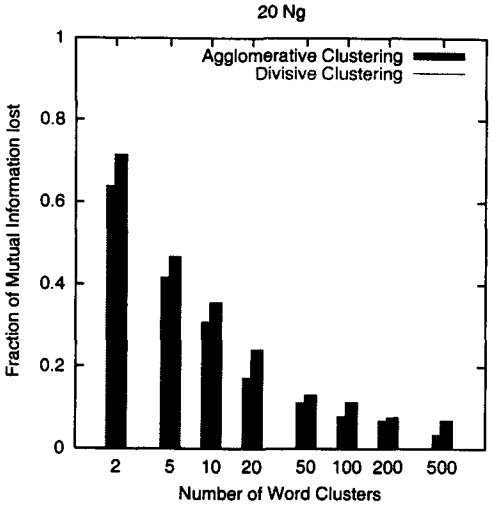

Figure 2: Fraction of Mutual Information lost while clustering words with Divisive Clustering is significantly lower compared to Agglomerative Clustering at all number of features (on 20 Newsgroups data).

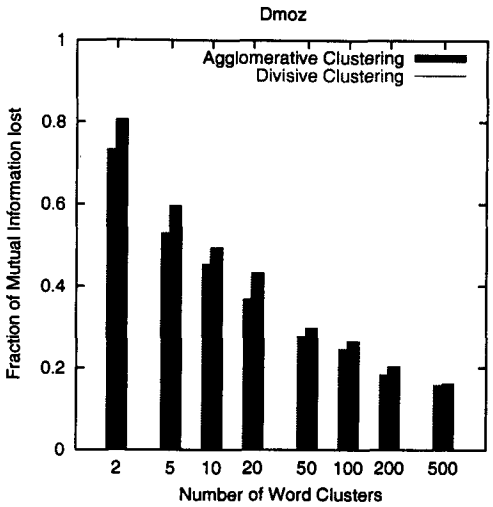

Figure 3: Fraction of Mutual Information lost while clustering words with Divisive Clustering is significantly lower compared to Agglomerative Clustering at all number of features (on Dmoz data).

6.1 Data Sets and Implementation Details

The *20 Newsgroups (20Ng)* data set, collected by Ken Lang, 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[2, 28, 21]. During indexing we skipped headers, pruned words occurring in less than 3 documents and used a stop list but did not use stemming. 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* category and crawled some of the heavily populated internal nodes beneath it resulting in a 3-deep hierarchy with 49 leaf level nodes and about 5,000 total documents. For our experimental results we ignored documents at internal nodes. The list of categories and urls we used is available at www.cs.utexas.edu/users/manyam/dmoz.txt. While indexing we skipped text between html tags, pruned words occurring in less than five documents, used a stop list but did not use stemming. The resulting vocabulary had 14,538 words.

Bow[22] 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 Agglomerative and Divisive Clustering within Bow, and used Bow's SVM implementation in our experiments.

6.2 Results

We first give evidence of the improved quality of word clusters obtained by our algorithm as compared to the agglomerative approach. 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). Figures 2 and 3 plot the fraction of mutual information lost against the number of clusters for both the divisive and agglomerative algorithms on the 20Ng and Dmoz data sets. Notice that less mutual information is lost with Divisive Clustering compared to Agglomerative Clustering at *all* number of clusters, though the difference is more pronounced at lower number of clusters.

Next we provide anecdotal evidence that our word clusters are better at preserving class information as compared to the agglomerative approach. Figure 4 shows three word clusters, Cluster 9 and Cluster 10 from Divisive Clustering and Cluster 12 from Agglomerative Clustering. These clusters were obtained while forming 20 word clusters with a 1/3-2/3 test-train split. While the clusters obtained by our algorithm could successfully distinguish between *rec.sport.hockey* and *rec.sport.baseball*, Agglomerative Clustering combined words from both classes in a single cluster. This resulted in lower classification accuracy for both classes with Agglomerative Clustering compared to Divisive Clustering. While Divisive Clustering achieved *93.33% and 94.07%* on *rec.sport.hockey* and *rec.sport.baseball* respectively, Agglomerative Clustering could only achieve *76.97% and 52.42%*.

6.2.1 Classification Results on 20 Newsgroups data

Figure 5 shows classification accuracies on the 20 Newsgroups data set for the algorithms considered. The horizontal axis indicates the number of features/clusters used in the classification model while the vertical axis indicates the percentage of test documents that were classified correctly. The results are averages of 5-10 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 techniques, SVMs and Naive Bayes (NB) for the purpose of comparison. Observe that Divisive Clustering (SVM as well as NB) achieves significantly better results at lower number of features than feature selection using Information Gain and Mutual Information. With only 50 clusters, Divisive Clustering (NB) achieves 78.05% accu-

Cluster 10 Divisive Clustering (Hockey)	Cluster 9 Divisive Clustering (Baseball)	Cluster12 Agglomerative Clustering (Hockey and Baseball)	
team	hit	team	detroit
game	runs	hockey	pitching
play	baseball	games	hitter
hockey	base	players	rangers
season	ball	baseball	nyi
boston	greg	league	morris
chicago	morris	player	blues
pit	ted	nhl	shots
van	pitcher	pit	vancouver
nhl	hitting	buffalo	ens

Figure 4: Top few words sorted by Mutual Information in Clusters obtained by Divisive and Agglomerative approaches on 20 Newsgroups data.

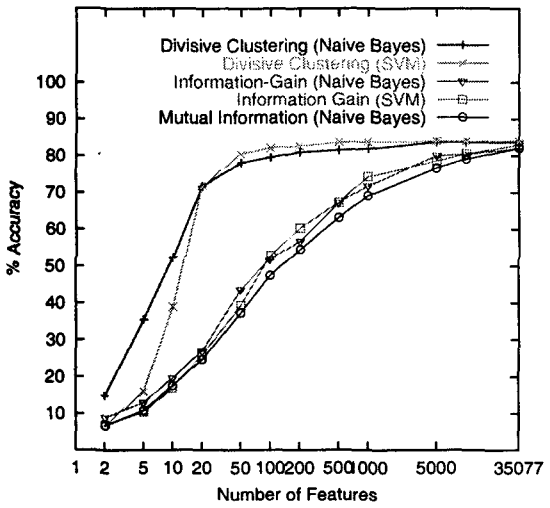

Figure 5: Classification Accuracy on 20 Newsgroups data with 1/3-2/3 test-train split.

racy — 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 closely observed these word clusters we found that many of them contained words representing a single class in the data set, for example see Figure 4. We attribute this observation to our effective initialization strategy.

In Figure 6, 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 5-10 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 Figure 5 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.

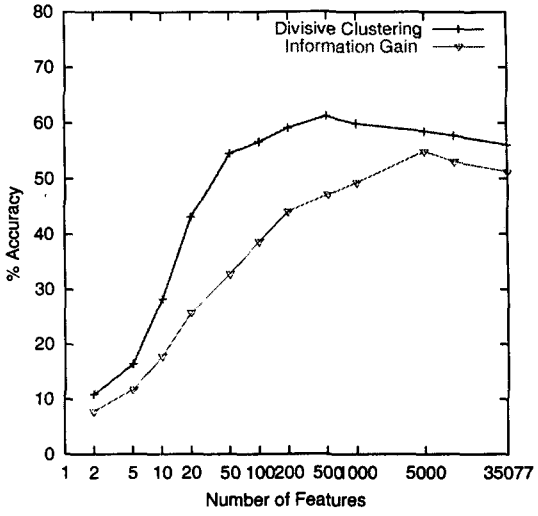

Figure 6: Classification Accuracy on 20 Newsgroups data with 2% Training data (using Naive Bayes).

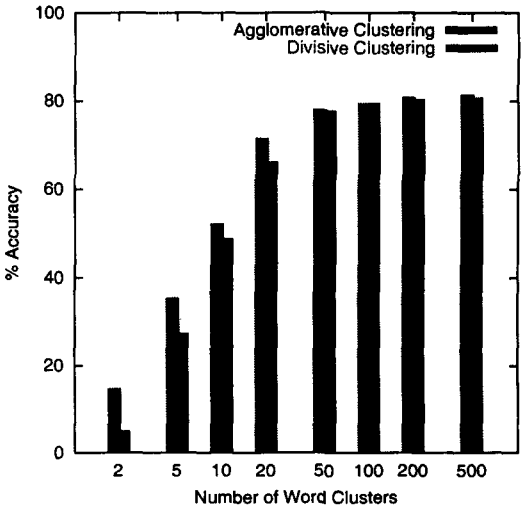

Figure 7: Divisive Clustering leads to higher accuracy than Agglomerative Clustering on 20 Ng data (1/3-2/3 test-train split with Naive Bayes).

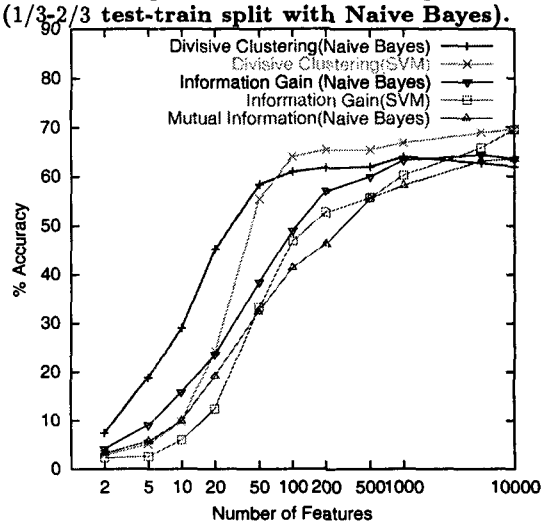

Figure 8: Classification Accuracy on Dmoz data with 1/3-2/3 test-train split.

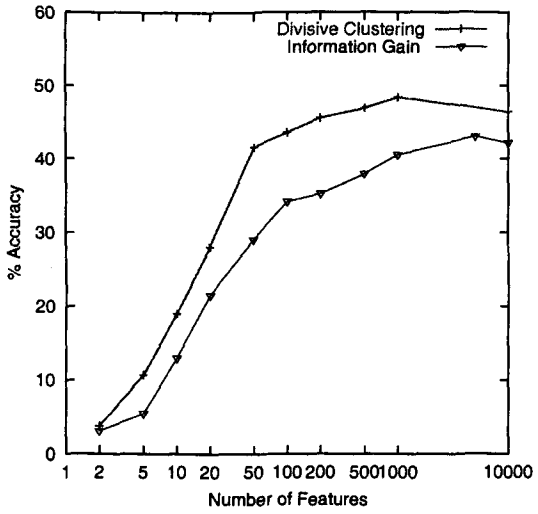

Figure 9: Classification Accuracy on Dmoz data with 2% Training data (using Naive Bayes).

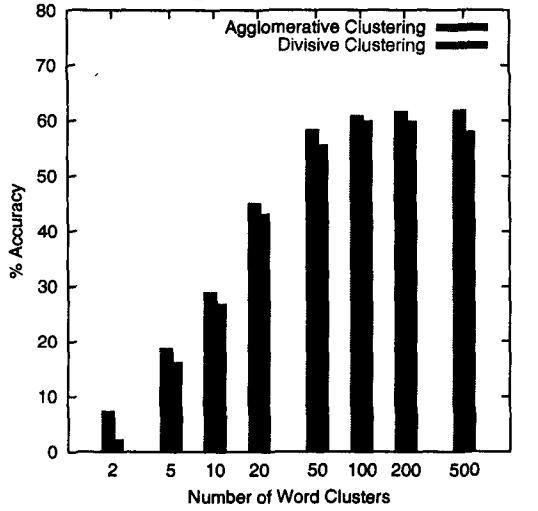

Figure 10: Divisive Clustering achieves higher accuracy than Agglomerative Clustering on Dmoz data (1/3-2/3 test-train split with Naive Bayes).

Figure 7 compares the classification accuracies of Divisive Clustering and Agglomerative Clustering on the 20 Newsgroups data using Naive Bayes. Note that Divisive Clustering achieves better classification results than Agglomerative Clustering at all number of features, though again the improvements are more significant at lower number of features.

6.2.2 *Classification Results on Dmoz data set*

Figure 8 shows 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 sometimes improves classification accuracy since HTML data appears 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). To overcome this, Baker and McCallum[2] tried a combination of feature-clustering and feature-selection methods. More rigorous approaches to this problem are a topic of future work. Further note that SVMs fare

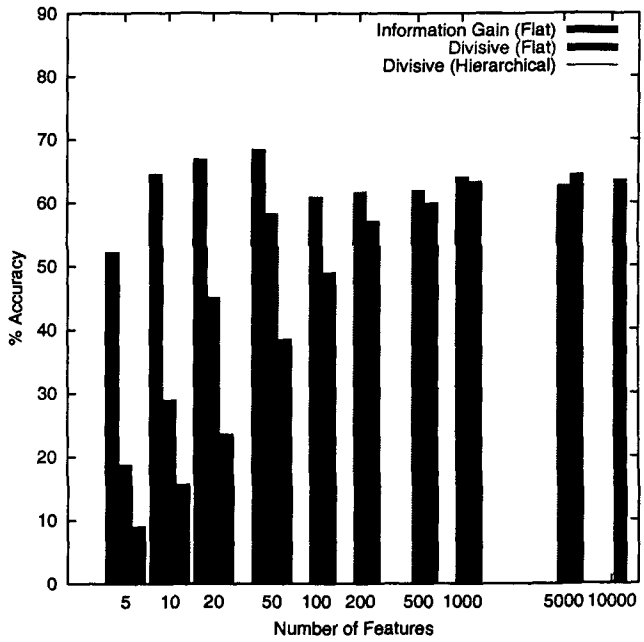

Figure 11: Classification results on Dmoz hierarchy using Naive Bayes. Observe that the Hierarchical Classifier achieves significant improvements over the Flat classifiers with very few number of features.

worse than NB at low dimensionality but better at higher dimensionality, which is consistent with known SVM behavior [29]. In future work we will 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 feature clustering is an attractive option when training data is limited. Figure 10 compares Divisive Clustering with Agglomerative Clustering on Dmoz data where we observe similar improvements as with 20 Newsgroups data.

6.2.3 *Hierarchical Classification on Dmoz Hierarchy*

Figure 11 shows classification accuracies obtained by 3 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 11 compares the Hierarchical Classifier with two flat classifiers, one that employs Information Gain for feature selection while the other uses Divisive Clustering. Note that Divisive Clustering performs remarkably well for Hierarchical Classification even at very low number of features. With just 10 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 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 using 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 that captures 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 obtains superior word clusters than the agglomerative strategies proposed previously[2, 28]. 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 significant improvements in classification accuracies especially at lower number of features. When the training data is sparse, feature clustering achieves higher classification accuracy than the maximum accuracy achieved by feature selection methods such as information gain and mutual information. 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. 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.

Acknowledgements. We are grateful to Andrew McCallum and Byron Dom for helpful discussions. For this research, ISD was supported by a NSF CAREER Grant (No. ACI-0093404) while Mallela was supported by a UT Austin MCD Fellowship.

8. REFERENCES

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

[2] L. D. Baker and A. McCallum. Distributional clustering of words for text classification. In *ACM SIGIR*, pages 96–103, 1998.

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

[4] P. Berkhin and J. D. Becher. Learning simple relations: Theory and applications. In *Second SIAM Data Mining Conference*, pages 420–436, 2002.

[5] 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*, 1997.

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

[7] 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.

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

[9] 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.

[10] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, 2nd edition, 2000.

[11] E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21(3):768, 1965.

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

[13] 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.

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

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

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

[17] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98*, pages 137–142, 1998.

[18] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *ICML*, 1997.

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

[20] J. Lin. Divergence measures based on the Shannon entropy. *IEEE Trans. Inform. Theory*, 37(1), 1991.

[21] 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.

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

[23] T. Mitchell. Conditions for the equivalence of hierarchical and non-hierarchical bayesian classifiers. Technical report, CALD, CMU, 1998.

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

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

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

[27] C. E. Shannon. A mathematical theory of communication. *Bell System Technical J.*, 27, 1948.

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

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

[30] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, 1997.