A Comparison of Word- and Sense-based Text Categorization Using Several Classification Algorithms

Ath. Kehagias, V. Petridis, V.G. Kaburlasos, and P. Fragkou

6 April 2001

Abstract

Most of the text categorization algorithms in the literature represent documents as collections of *words*. An alternative which has not been sufficiently explored is the use of *word meanings*, also known as *senses*. In this paper, using several algorithms, we compare the categorization accuracy of classifiers based on words to that of classifiers based on senses. The document collection on which this comparison takes place is a subset of the annotated Brown Corpus semantic concordance. A series of experiments indicates that the use of senses does not result in any significant categorization improvement.

Index Terms — Text categorization, word senses, Brown Corpus, naive Bayes, k- nearest neighbour, FLNMAP with voting.

1 Introduction

The text categorization problem appears in a number of application domains including information retrieval (IR) [23, 24, 25], data mining [22], and Web searching [14, 16]. A number of text categorization algorithms have appeared in the literature [7, 8, 25, 37] – for an overview see [23, 26, 38, 39]. While *unsupervised* text categorization has also attracted attention [37], most of the above algorithms address the problem of *supervised* text categorization where a typical task has the following characteristics.

1. A set of *document categories* is given along with a *training set* of documents which are *labeled,* i.e. the category to which each document belongs is known.
2. The documents are *preprocessed* to extract some *document features*.
3. A *categorization algorithm* is selected and its *parameters* are tuned using the training data set.
4. The trained algorithm is applied for the categorization of a *test set* of *unlabeled documents*, in order to determine their corresponding categories.

It follows from the above description that text categorization is a typical *pattern classification* task such as the ones described in [11, 26]. From now on we will use the terms "text categorization" and "text classification" interchangeably. As with most pattern classification tasks, the initial step of *data preprocessing* is crucial for the quality of the final results (see [11, 15, 26]). In the case of text categorization, an essential aspect of preprocessing is the selection of appropriate *document features* [3, 6, 9, 17, 21, 38]. This is usually referred to as *document representation*.

While a large number of document representations have been proposed, most of them use the same starting point, namely the *words* appearing in a document. In fact, a common choice is to represent a document as a "bag of words" [26, 30], i.e. a document is represented by the set of words appearing in

it. Another commonly used and slightly richer representation takes account of the *frequency* with which words appear in a specific document. Such representations ignore important aspects of a document, for instance the *order* in which words appear in the document, the *syntax* etc. [2]. Richer representations have also been proposed (see for instance [30] where hierarchies of trees are employed) but the emphasis on words is usually retained.

In this paper we present experiments to compare document representations which utilize (a) *words* and (b) *word meanings*. It is a characteristic of natural languages that the same word may assume different meanings in different contexts. For example the word "base" may mean either a *military* camp, or the place that a *baseball* runner must touch before scoring; the word "crane" may mean either a *bird* or a *machine* that lifts and moves heavy objects; and so on. The following conjecture appears reasonable: word meanings provide more information about the content of a document (and the category to which it belongs) than words themselves.

The goal of this work is to test the above conjecture by carrying out text classification experiments on a document collection using: (a) word-based representations, and (b) sense-based representations.

In this endeavor we make use of existing lexicographic work. More specifically, an ongoing project at Princeton University has produced *Wordnet* [28] a *lexical database* which provides an extensive list of English words and their various meanings. Rather than the term "meaning", the term used in the Wordnet context is "*sense*", which we will use from now on. A companion to the Wordnet database is the *annotated Brown Corpus*. This is a collection of several hundred documents, where every document is labeled as belonging to one of fifteen categories and the words in each document are annotated with the senses they assume in the specific context.

In this paper we present experiments on the classification of Brown Corpus documents. We compare the classification efficiency of words to that of senses using several classification algorithms. The results thus obtained are useful in two ways: (a) in comparing the merit of words and senses as classification features and (b) in testing several classification algorithms on the Brown Corpus. We only know of one previous work which uses the Brown Corpus as a benchmark for classification algorithms [20]. In addition, preliminary versions of our work have appeared in [19, 34].

The remaining sections of this paper are organized as follows: in Section 2 we briefly discuss some difficulties associated with the use of senses as document features – this also leads to a discussion of the Wordnet database and the Brown Corpus; in Section 3 we present the document representations and in Section 4 the classification algorithms we have used; in Section 5 we present our experimental results; in Section 6 we discuss these results and provide some concluding remarks.

2 The Document Collection

We start with a brief discussion of the use of senses in text categorization. This leads to a description of the Wordnet lexical database and the Brown Corpus semantic concordance. Finally, we introduce the document collection which we have used in our text categorization experiments; this collection is a subset of the Brown Corpus.

2.1 Words and Senses

Any implementation of sense-based text classification must resolve the following difficulty: while words are immediately *observable* within a document, meanings are *hidden*. For instance, when the word "base" appears in a document it is not immediately obvious whether it assumes the military or the baseball meaning. Therefore a procedure is needed for recovering the *senses* from the *words* used in a specific context.

We repeat that the goal of this paper is to compare the merit (for document classification purposes) of words to that of senses as "document features". We have sidestepped the aforementioned difficulties by using existing lexicographic work, namely the *Wordnet lexical database* and the *Brown Corpus semantic concordance*, a collection of sense-tagged documents which will be discussed presently. We recognize that any practical sense-based classification algorithm must also perform *word disambiguation* [1, 12], i.e. determining the sense each word assumes in a particular context. This is a difficult problem in its own right and it is *not* addressed at the present paper. However, from the point of view of this paper, this is a side issue since we are here interested in a "direct" comparison of words and senses.

2.2 The Wordnet Lexical Database

Wordnet is an on-line lexical database which was developed at the Cognitive Science Laboratory at Princeton University under the direction of G.A. Miller [28]. Wordnet is similar to an electronic thesaurus and is organized around the distinction between words and senses. It contains a large number of nouns, verbs, adjectives and adverbs of the English language, reaching a total of nearly 130,000 words. These words are organized into synonym sets (briefly *synsets*); each synset represents (is equivalent to) the underlying lexical concept expressed by all the synonymic words; a word may belong to more than one synset. Each synset is associated to a sense, i.e. a word meaning; Wordnet contains a total of nearly 100,000 senses. An important feature of Wordnet is that synsets are linked by lexical relations, but this will not concern us here. For our purposes it is enough to acknowledge the distinction between words and senses and to note that Wordnet provides carefully worked out word- and sense-vocabularies for the English language, as well as the membership of each word into a number of senses.

2.3 The Brown Corpus Semantic Concordance

The Brown Corpus is a collection of 500 documents which are classified into fifteen categories (fourteen of these are listed in Table 1; an additonal category "Religion" has not been used, as will be explained presently); for an extended description of the Brown Corpus see [13]. The Brown Corpus semantic concordance is distributed along with Wordnet. A *semantic concordance* is the combination of a collection of documents and a thesaurus; the documents are combined in manner such that every substantive word in each document is linked to its appropriate sense in the thesaurus. Thus a semantic concordance can be viewed either as a document collection in which words have been tagged syntactically and semantically, or as a thesaurus in which example sentences can be found for many definitions. The Brown Corpus semantic concordance makes use of 352 out of the 500 Brown Corpus documents. Linguists involved in the Wordnet project manually performed *semantic tagging*, i.e. annotation of the 352 texts with WordNet senses. In 166 of these documents only verbs are annotated with tags which indicate the Wordnet "id. number" of the respective verb and the sense with which it is used in this particular instance. In the remaining 186 documents nouns, adjective and adverbs, as well as verbs are similarly annotated.

2.4 The Text Categorization Data

The documents we have used in our text categorization experiments are a subset of the Brown Corpus. In particular, we dropped the aforementioned 166 partially tagged documents and also the 4 documents that belong to the religion category. This left 182 documents falling into 14 categories. Furthermore, we have aggregated several categories into new ones. The goal of this preprocessing was to obtain a reasonably large number of sense-tagged documents in each of the categories to be used. The final

data set which we have used in our experiments consists of 182 documents divided into 7 categories and is further described in Table 1.

Table 1 to appear here

3 Document Representations

In this work we have used four document representations. Two of these are word-based and the remaining two are sense-based. We present in some detail the two word-based representations (the sense-based representations are exactly analogous).We start with the following elements.

1. A collection of training documents $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M$.
2. A collection of K class identifiers $c_1, c_2, \ldots, c_K$.
3. A set of labels $q_1, q_2, \ldots, q_M$ such that q_m is the class to which document $\mathbf{x}_m$ belongs (for $m = 1, 2, ..., M$, $q_m \in \{c_1, c_2, ..., c_K\}$).

Each of the documents is a vector of words, i.e. for $m = 1, 2, ..., M$ we have

$$\mathbf{x}_m = [x_{m1} \quad x_{m2} \quad \ldots \quad x_{mj} \quad \ldots \quad x_{mJ_m}]$$

where $m = 1, 2, ..., M$ and $j = 1, 2, ..., J_m$.

Note that J_m is the total number of words which appear in the m-th training document, whereas x_{mj} is the j-th word which appears in the m-th training document. Moreover note that x_{mj} takes values in the vocabulary $\mathbf{W}$ which is defined as vector

$$\mathbf{W} = [w_1 \quad w_2 \quad \ldots \quad w_n \quad \ldots \quad w_{N_w}],$$

where N_w is the total number of words which appear in all training documents.

We remark that, while the contents of the m-th document are stored in $\mathbf{x}_m$, which is a vector of variable length J_m, the *document representation* is a vector of fixed length N_w. We use two such representations.

1. The first representation is the *Word Boolean (WB)* document vector. The WB vector $\mathbf{b}_m$ for the m-th document is has the form

   $$\mathbf{b}_m = [b_{m1} \quad b_{m2} \quad \ldots \quad b_{mn} \quad \ldots \quad b_{mN_w}]$$

   where (for $n = 1, 2, ..., N_w$) we have $b_{mn} = 1$ if w_n appears in $\mathbf{x}_m$ and 0 otherwise.

2. The second representation is the *Word Frequency (WF)* document vector

   $$\mathbf{f}_m = [f_{m1} \quad f_{m2} \quad \ldots \quad f_{mn} \quad \ldots \quad f_{mN_w}]$$

   where (for $n = 1, 2, ..., N_w$) we have

   $$f_{mn} = \text{number of times the } n\text{-th word } (w_n) \text{ appears in the } m\text{-th document.}$$

The WB and WF vectors are the basic document representations in terms of words; two additional document representations are also computed in exactly analogous manner, making use of senses rather than words, i.e. the *Sense Boolean (SB)* document vector and the *Sense Frequency (SF)* document vector. These make use of a sense vocabulary of the form

$$\mathbf{S} = [s_1 \quad s_2 \quad \ldots \quad s_n \quad \ldots \quad s_{N_s}],$$

where N_s is the total number of senses which appear in all training documents. All the remaining details are exactly analogous to the case of word-based representations.

4 Classification Algorithms

4.1 Maximum A Posteriori (MAP) Classification

Maximum A Posteriori classification, or MAP classification for short (also known as *Naive Bayes* classification [10, 26, 29]) is effected by maximization over $c_1, ..., c_K$ of the *posterior probability*

$$\Pr(c_k|\mathbf{x}_m) \ = \Pr(c_k|x_{m1}, x_{m2}, ..., x_{mJ_m}). \tag{1}$$

I.e. the document is classified to the $\widehat{k}$-th category, where $\widehat{k} = \arg \max_{k=1,2,...,K} \Pr(c_k|x_{m1}, x_{m2}, ..., x_{mJ_m})$.

The name "Naive Bayes" comes from the following assumption: it is assumed that the probability of the j-th word (in the m-th document) only depends on the category, but not on the remaining words of the document. This allows for an easy computation of $\Pr(c_k|x_{m1}, x_{m2}, ..., x_{mJ_m})$ in terms of the conditional probabilities $\Pr(w_n|c_k)$ of the words given the category. Let us first indicate how to obtain estimates of $\Pr(w_n|c_k)$ and then we will present two alternative computations of (1).

By use of Bayes' theorem we have

$$\Pr(w_n|c_k) = \frac{\Pr(w_n, c_k)}{\Pr(c_k)} = \frac{\Pr(w_n, c_k)}{\sum_{n=1}^{N_w} \Pr(w_n, c_k)}. \tag{2}$$

The following estimate is used (for $n = 1, 2, ..., N_w$ and $k = 1, 2, ..., K$) to compute (2)

$$\widehat{\Pr}(w_n|c_k) = \frac{(\alpha + \sum_{m=1}^{M} f_{mn}\mathbf{1}_{mk})}{\sum_{n=1}^{N_w}(\alpha + \sum_{m=1}^{M} f_{mn}\mathbf{1}_{mk})}. \tag{3}$$

Here $\mathbf{1}_{mk} = 1$ if $q_m = c_k$ and 0 otherwise; α is a *tuning parameter* of the algorithm: large values of α result in a more uniform probability distribution. Let us also mention that $\Pr(c_k)$ can be estimated by

$$\widehat{\Pr}(c_k) = \frac{\sum_{m=1}^{M} \mathbf{1}_{mk}}{M}. \tag{4}$$

We present in the following two versions of the MAP algorithm for computing $\Pr(c_k|x_{m1}, x_{m2}, ..., x_{mJ_m})$ in terms of the $P(w_n|c_k)$. We present the word-based variants; the sense-based ones are exactly analogous.

4.1.1 Batch Version

Given a new (unlabeled) document $\mathbf{d}$, the MAP classifier calculates (in terms of the $P(w_n|c_k)$ probabilities) the probabilities $P(c_k|\mathbf{d})$ for $k = 1, 2, ..., K$ and categorizes $\mathbf{d}$ to the class c_k which maximizes $P(c_k|\mathbf{d})$. Computation of $\Pr(c_k|x_{m1}, x_{m2}, ..., x_{mJ_m})$ is done as follows.

$$\Pr(c_k|\mathbf{d}) = \Pr(c_k|x_{m1}, x_{m2}, ..., x_{mJ_m}) = \frac{\Pr(x_{m1}, x_{m2}, ..., x_{mJ_m}|c_k) \cdot \Pr(c_k)}{\Pr(x_{m1}, x_{m2}, ..., x_{mJ_m})} =$$

$$= \frac{\Pr(x_{m1}, x_{m2}, ..., x_{mJ_m}|c_k) \cdot \Pr(c_k)}{\sum_{i=1}^{K} \Pr(x_{m1}, x_{m2}, ..., x_{mJ_m}|c_i) \cdot \Pr(c_i)} =$$

$$= \frac{\Pr(x_{m1}|c_k) \cdot \Pr(x_{m2}|c_k) \cdot ... \cdot \Pr(x_{mJ_m}|c_k) \cdot \Pr(c_k)}{\sum_{i=1}^{K} \Pr(x_{m1}|c_i) \cdot \Pr(x_{m2}|c_i) \cdot ... \cdot \Pr(x_{mJ_m}|c_i) \cdot \Pr(c_i)} \tag{5}$$

Now, since for $j = 1, 2, ..., J_m$ we have $x_{mj} \in \mathbf{W}$, it follows that eq(5) can be computed in terms of the $\widehat{\Pr}(w_n|c_k)$ estimates (for $k = 1, 2, ..., K$ and $n = 1, 2, ..., N_w$) given by eq(3) and the $\widehat{\Pr}(c_k)$ estimates (for $k = 1, 2, ..., K$) given by eq(4). Hence the batch version of the MAP classification algorithm has one tuning parameters: α.

4.1.2 Recursive Version of the MAP algorithm

We also present a *recursive* formula for the computation of $\Pr(c_k|\mathbf{d}) = \Pr(c_k|x_{m1}, x_{m2}, ..., x_{mJ_m})$. This is related to a time series classification algorithm we have presented in [35, 36].

Defining for $m = 1, 2, ..., M$, $k = 1, 2, ..., K$ and $j = 1, 2, ..., J_m$

$$p_0^{m,k} \doteq \Pr(c_k), \quad p_j^{m,k} \doteq \Pr(c_k|x_{m1}, x_{m2}, ..., x_{mj}). \tag{6}$$

In other words $p_j^{m,k}$ is the probability that the m-th document belongs to the k-th category *having seen up to the j-th word*. We now give a recursive relation which computes $p_j^{m,1}, p_j^{m,2}, ..., p_j^{m,K}$ from $p_{j-1}^{m,1}, p_{j-1}^{m,2}, ..., p_{j-1}^{m,K}$. In this manner one can compute the category to which m-th document is most likely to belong, having read up to the j-th word of this document. We start with Bayes' rule expressed as follows:

$$\Pr(c_k|x_{m1}, x_{m2}, ..., x_{mj}) = \frac{\Pr(c_k, x_{mj}|x_{m1}, x_{m2}, ..., x_{m,j-1})}{\Pr(x_{mj}|x_{m1}, x_{m2}, ..., x_{m,j-1})} =$$

$$= \frac{\Pr(c_k, x_{mj}|x_{m1}, x_{m2}, ..., x_{m,j-1})}{\sum_{i=1}^{K} \Pr(c_i, x_{mj}|x_{m1}, x_{m2}, ..., x_{m,j-1})} =$$

$$= \frac{\Pr(x_{mj}|x_{m1}, x_{m2}, ..., x_{m,j-1}, c_k) \cdot \Pr(c_k|x_{m1}, x_{m2}, ..., x_{m,j-1})}{\sum_{i=1}^{K} \Pr(x_{mj}|x_{m1}, x_{m2}, ..., x_{m,j-1}, c_i) \cdot \Pr(c_i|x_{m1}, x_{m2}, ..., x_{m,j-1})}.$$

Using the definition of $p_j^{m,k}$ and the Naive Bayes assumption it follows

$$p_j^{m,k} = \Pr(c_k|x_{m1}, x_{m2}, ..., x_{mj}) = \frac{p_{j-1}^{m,k} \Pr(x_{mj}|c_k)}{\sum_{i=1}^{K} p_{j-1}^{m,i} \Pr(x_{mj}|c_i)}. \tag{7}$$

The above computation can be performed in terms of the estimates $\widehat{\Pr}(c_k)$ (given by eq.(4)) and $\widehat{\Pr}(w_n|c_k)$ (given by eq.(3)). In practice the computation of (7) is modified in the following manner. A *threshold parameter* h is specified by the user and in every iteration it is checked whether $p_j^{m,k}$, $k = 1, 2, ..., K$ is less than h; if it is then $p_j^{m,k}$ is set equal to h as explained in [35, 36]. Hence the recursive version of the MAP classification algorithm has two tuning parameters: α and h.

4.2 Maximum Likelihood (ML) Classification

In this algorithm, classification is effected by maximizing the Likelihood function $\Pr(\mathbf{x}_m|c_k) = \Pr(x_{m1}, x_{m2}, ..., x_{mJ_m}|c_k)$ over $c_1, ..., c_K$. This is somewhat simpler then maximizing the MAP function. We present the word-based variant of the algorithm; the sense-based variant is exactly analogous. Continuing with the notation of the previous section,

$$\Pr(\mathbf{x}_m|c_k) = \Pr(x_{m1}, x_{m2}, ..., x_{mJ_m}|c_k) = \prod_{j=1}^{J_m} \Pr(x_{mj}|c_k). \tag{8}$$

ML classification can be applied in exactly analogous manner to MAP classification, where a factor $\Pr(x_{mj}|c_k)$ in eq.(8) for the j-th word in the m-th document is replaced by the estimate $\widehat{\Pr}(w_n|c_k)$ (given by eq.(3)) for the probability of the corresponding word of the vocabulary $\mathbf{W} = [w_1 \quad w_2 \quad ... \quad w_n \quad ... \quad w_{N_w}]$. As previously, α is a tuning parameter of the algorithm.

4.3 Two K-Nearest Neighbor (KNN) Variants

We now present a classification algorithm inspired by the well known K-Nearest Neighbor (KNN) algorithm [26, 27, 29, 40]. As usual we present the word-based version of the algorithm; the sense-based version is exactly analogous. We present two variants of this algorithm; both can operate on either Boolean or relative frequency vectors.

4.3.1 K-Nearest Neighbor (KNN) Variant no.1

The algorithm can operate on either Boolean or Frequency vectors; in case Frequency vectors $\mathbf{f}_m$ are used then the first step is to compute from the $\mathbf{f}_m$'s the *relative frequency vectors* $\mathbf{r}_m$. These are defined as follows (for $m = 1, 2, ..., M$ and $n = 1, 2, ..., N_w$)

$$r_{mn} = \frac{f_{mn}}{\sum_{n=1}^{N_w} f_{mn}}.$$

We present the algorithm in terms of the Boolean vectors $\mathbf{b}_m$; the case of the $\mathbf{r}_m$ vectors is exactly analogous. The training set consists of document vectors $\mathbf{b}_1$, $\mathbf{b}_2$, ... , $\mathbf{b}_M$. Each of these vectors is stored in memory, along with the corresponding labels q_1, q_2, ... , q_M.

Now consider an incoming, unlabeled document with WB vector $\mathbf{b}$. For $m = 1, 2, ..., M$ compute the quantities

$$D_m = \frac{||\mathbf{b}_m - \mathbf{b}||^2}{||\mathbf{b}_m|| \cdot ||\mathbf{b}||} \tag{9}$$

(where $||\cdot||$ is Euclidean norm) and for $k = 1, 2, ..., K$ compute the quantities

$$C_k = \sum_{m=1}^{M} \left(\frac{1}{D_m}\right)^P \mathbf{1}_{mk} \tag{10}$$

where P is a tuning parameter and $\mathbf{1}_{mk}$ has been defined in section 4.1. Finally, assign the incoming document to the $\widehat{k}$-th category, where

$$\widehat{k} = \arg \max_{k=1,2,...,K} C_k. \tag{11}$$

The rationale of the algorithm is rather obvious: For $m = 1, 2, ..., M$, the m-th training document "votes" for the unlabeled document to be assigned to category q_m (i.e. to the the category of the m-th training document); however the votes are weighted in a manner inversely proportional to the distance of the voters from the unlabeled document (notice also the scaling in eq.(9)). The weighted votes are tallied in the C_k variables and the unlabeled document is finally assigned to the category with maximum C_k.

4.3.2 K-Nearest Neighbor (KNN) Variant no.2

This variant is identical to the one of Section 4.3.1, except that the D_m's are defined as follows

$$D_m = \frac{||\mathbf{b}_m - \mathbf{b}||}{||\mathbf{b}_m|| + ||\mathbf{b}||} \tag{12}$$

for the case of WB vectors and analogously for the cases of WF, SB, and SF vectors. Eqs.(10, 11) remain the same.

4.4 σ-FLNMAP with Voting

The σ-FLNMAP (FLN stands for *Fuzzy Lattice Neurocomputing*) has been introduced in [32] as a neural algorithm for classification by supervised clustering. In other words, the first step of classification is to cluster the training data into *homogeneous clusters*, i.e. the goal is that every datum in a cluster belongs to the same category. In the second step, the testing data are classified to the category of the cluster in which they are *maximally included* (as explained in the sequel).

The σ-FLNMAP is based on the synergetic combination of two σ-FLN neural modules for clustering [18]. As explained in [31], an FLN model applies to any domain which can be expressed as a (mathematical) lattice [4]. For instance, in [33] an FLN model for clustering is applied in a lattice of graphs obtained from a Thesaurus of synonyms in order to compute clusters of semantically related words. In this paper the σ-FLNMAP is applied to the N-dimensional (lattice) unit hypercube.

The input to the σ-FLNMAP can be either Boolean vectors $\mathbf{b}_1$,$\mathbf{b}_2$,...,$\mathbf{b}_M$ or normalized versions $\mathbf{g}_1, \mathbf{g}_2$,...,$\mathbf{g}_M$ of the frequency vectors $\mathbf{f}_1, \mathbf{f}_2$,...,$\mathbf{f}_M$. In the latter case, for $m = 1, ..., M$ we have $\mathbf{g}_m = [\ g_{m1}\ \ g_{m2}\ \ \cdots\ \ g_{mj}\ \ \cdots\ \ g_{mN_w}\ ]$, where $g_{mj} = \frac{f_{mj}}{\max\limits_{m=1,\cdots,M} f_{mj}}$. Furthermore, the vectors may be either word- or sense-derived.

Both the clustering and classification phases make use of the concept of *fuzzy inclusion*. The clusters computed by σ-FLNMAP are *hyperboxes* contained in the unit N_w-dimensional hypercube; a hyperbox is specified by the coordinates of its diagonally located bottom and top corners. In other words a hyperbox is specified by a pair of (coordinate) vectors: $A = (\mathbf{g}, \mathbf{h})$. Hence, the data points correspond to "trivial" hyperboxes, where the bottom and top corner coincide: $(\mathbf{g}, \mathbf{g})$. A *positive valuation* function $v(\cdot)$ corresponds to every box A a real number $v(A)$, which can be related to the size of A. The degree of *fuzzy inclusion* of box A in box B is specified in terms of the *inclusion measure* function $\sigma(A \sqsubset B) = \frac{v(B)}{v(A \vee B)}$ where $A \vee B$ is the smallest box which contains both A and B [1]. Given a hyperbox $A = (\mathbf{g}, \mathbf{h}) = ([g_1, ..., g_{N_w}], [h_1, ..., h_{N_w}])$, we compute $v(A) = \sum_{i=1}^{N_w} [\theta(g_i) + h_i]$ where $\theta : \mathbb{R} \to \mathbb{R}$ is a function such that $x_1 \sqsubset x_2 \Rightarrow \theta(x_1) \geq \theta(x_2)$; note that in this work the function $\theta(x) = 1 - x$ has been used. As detailed in [18], $v(.)$ defines a positive valuation function and consequently it defines an inclusion measure in the lattice of hyperboxes contained in the unit hypercube.

Having specified an inclusion measure, it is straightforward to implement both the clustering and classification algorithm. The terms "set (a box)" or "reset (a box)" in the following algorithm mean that a box B_J is "available" or "unavailable", respectively, for *accommodating* an input datum/document $\mathbf{g}_m$, where the *accommodation* of $\mathbf{g}_m$ in B_J is defined by equation $B_J = B_J \vee \mathbf{g}_m$ as detailed underneath.

1 The *clustering* phase is performed as follows[2].

$L := 1$
$B_L = \mathbf{g}_1$ (the first input document for training is memorized)
For $m = 1, ..., M$
  "set" all memorized-boxes B_i, $i = 1, ..., L$
  For $i = 1, ..., L$
    calculate $\sigma(\mathbf{g}_m \sqsubset B_i)$
  Endfor
  While (there exist "set" memorized-boxes B_i, $i = 1, ..., L$)

[1] All of the above concepts originate from "fuzzy lattice theory" where they have been expressed in more general form in [31].

[2] In this algorithm, ρ is the user-defined vigilance parameter with $\rho \in [0, 1]$.

B_J :=the box with $max\{\sigma(\mathbf{g}_m \sqcap B_i), i = 1, ..., L\}$ among the "set" memorized-boxes
If $(\sigma(B_J \sqcap \mathbf{g}_m) \geq \rho)$ **then**
$B_J = B_J \vee \mathbf{g}_m$ (accommodate $\mathbf{g}_m$ in B_J)
Exit the while loop
Else
"reset" B_J
Endif
Endwhile
If (all memorized-boxes B_i, $i = 1, ..., L$ have been "reset") **then**
$L := L + 1$
$B_L := \mathbf{g}_m$ (memorize input document $\mathbf{g}_m$)
Endif
Endfor

2 In the *classification* phase, a collection of boxes (produced during clustering) is available: $B_1, ..., B_R$. An incoming datum (document) $\mathbf{g}$, is classified to the class of the box $B_{\widehat{r}}$ which maximizes inclusion:

$$\widehat{r} \doteq \arg \max_{r=1,2,...,R} \sigma(\mathbf{g} \sqcap B_r)$$

It is known [32] that the boxes learned during "training" by the σ-FLNMAP depend on the order of data presentation. The "σ-FLNMAP with voting" emerges as a scheme which trains an *ensemble* of σ-FLNMAP modules on different permutations of the training data set and then it classifies unlabeled documents according to the "majority vote" of the ensemble. For certain values of the vigilance parameter ρ and the the number n_V of voters, the classification accuracy of the ensemble is better as well as more stable than the classification accuracy of an individual σ-FLNMAP module in the ensemble. This idea is related to *bagging* and *boosting* [5]. The parameters of the "σ-FLNMAP with Voting" algorithm are 1) the vigilance parameter ρ, and 2) the number n_V of voters in the ensemble.

5 Experimental Results

In this section we present the results of our classification experiments. First we explain the details of the following aspects of our experiments: the split of the documents into training and testing data, the document representations used, the classification algorithms used, the choice of algorithm parameters. Then we present the actual classification results.

5.1 Training and Testing Data

As has been explained in Section 2.4, our initial data set consists of 182 documents from the annotated Brown Corpus collection. Some of these documents (training data set) have been used for training the algorithms of Section 4 and the remaining documents (testing data set) have been used for evaluating the performance of the algorithms. We have used a two-thirds / one-third split of the 182 documents. I.e. we have a random split the 182 documents into a training set of 123 documents and a testing set of 59 documents. Ten random splits have been employed, resulting into 10 different train/test data sets. We refer to these data sets as set_0, set_1, ... , set_9. While the splits have been random, we have taken care that in every case each of the 7 categories is represented by a fixed number of documents in the train and test sets. The distribution of documents from each category in the train and test sets is listed in Table 2.

Table 2 to appear here

For each of the 10 data sets we have repeated a suite of classification experiments, as will be described below.

5.2 Document Representation

The first step in each suite of classification experiments is to construct a word and a sense vocabulary. This is a necessary step in constructing the document representation which will be described subsequently. The word vocabulary was created using only the knowledge available in the training data set. This means that documents in the test data set may incude words which will not be part of the vocabulary; it also implies that the vocabulary size may vary from one data set to the next. The same remarks hold for the sense vocabulary as well. In Table 3 we list the word and sense vocabulary sizes for each of the ten data sets.

Table 3 to appear here

Having constructed the word vocabulary, we proceed to obtain document representations for the 182 documents. This process is repeated for the 10 data sets. For each of the data sets we compute the corresponding WB, WF, SB and SF vectors.

5.3 Classification Algorithms

Starting from 10 different data sets (training and testing) we have produced for each such set 4 different document representations: Word Boolean (WB), Word Frequency (WF), Sense Boolean (SB), and Sense Frequency (SF). We now proceed to list the various classification algorithms described in Section 4, the tuning parameters of each algorithm and the document representations to which each of these algorithms apply. This information is summarized in Table 4.

Table 4 to appear here

It can be observed in Table 4 that each of the above algorithms has one or more parameters which influence classification performance. These parameters are collected in a parameter vector symbolized by π. For each of the above algorithms (and for each of the data sets) we perform the classification experiment several times (using various parameter vector values π_1, π_2, ... , π_L) and record the classification accuracy corrsponding for each parameter value.

5.4 Classification Results

Let us now explain the format in which the classification results are presented. The reader will recall that the basic classification experiment is repeated a large number of times:

1. using each of the 4 document representations;
2. for each document representation, using the 6 different algorithms;
3. for each document representation and each algorithm, using the different values of the parameter vector

4. for each document representation, algorithm and parameter value, using the ten different data sets.

We iterate that our goal has *not* been to evaluate classification algorithms, but to compare the classification merit of words and senses under various conditions. In the term "conditions" are included the various classification algorithms as well as the parameter values used by these algorithms.

These considerations bear upon the manner in which we present our results, which are summarized in Tables 5 – 10. Each table corresponds to a particular classification algorithm and each column in a table corresponds to a particular document representation. Furthermore, in each column are presented several classification accuracy scores: *min*imum, *fix*ed parameter, *max*imum, *ave*rage, and, occasionally, *val*idated. Let us now explain the meaning of each of these scores.

Consider for the time being the algorithm and document representation to be fixed. Now, for a fixed value of the parameter vector, call it π_l, a classification experiment is repeated ten times, once for each data set. Hence for the l-th parameter value and the i-th data set we obtain a classification accuracy c_{li} defined as follows

$$c_{li} = \frac{\text{no. of correctly classified documents in the } i\text{-th test set}}{\text{total no. of documents in the } i\text{-th test set}}$$

We now obtain the following classification scores by averaging *over data sets*.

1. The dataset-averaged performance obtained by the *best* parameter value:

$$c_{\max} = \frac{1}{10} \cdot \sum_{i=1}^{10} \max_{l=1,2,\ldots,L} c_{li}.$$

2. The dataset-averaged performance obtained by the *worst* parameter value:

$$c_{\min} = \frac{1}{10} \cdot \sum_{i=1}^{10} \min_{l=1,2,\ldots,L} c_{li}.$$

3. The dataset-averaged performance *averaged* over parameter values:

$$c_{\text{ave}} = \frac{1}{10} \cdot \sum_{i=1}^{10} \frac{\sum_{l=1}^{L} c_{li}}{L}.$$

4. The dataset-averaged performance obtained by using a *predefined* parameter value π_{l_1}, which we know by experience to yield good performance:

$$c_{\text{fix}} = \frac{1}{10} \cdot \sum_{i=1}^{10} c_{l_1 i}$$

5. The dataset-averaged performance obtained by using a *validated* parameter value π_{l_2} (using a subset of the training set as validation data set):

$$c_{\text{val}} = \frac{1}{10} \cdot \sum_{i=1}^{10} c_{l_2 i}$$

These scores give a relatively broad assesment of the merit of words and senses as document features. The classification results appear in Tables 5–10. The classification results of the maximum a posteriori (MAP) algorithm are shown in Tables 5 and 6 using, respectively, the batch version and the recursive version of the MAP algorithm. Table 7 shows the results of the maximum likelihood (ML) algorithm. The results of the K-Nearest Neighbor (KNN), in particular its variants no.1 and no.2 are shown, respectively, in Tables 8 and 9. Finally Table 10 displays the classification accuracy results using the "σ-FLNMAP with Voting" algorithm.

Table 5 to appear here

Table 6 to appear here

Table 7 to appear here

Table 8 to appear here

Table 9 to appear here

Table 10 to appear here

Let us comment on the tables. Each row of the tables corresponds to a different aspect of the word- and sense-based classifier performance. The picture that emerges from all such aspects is more or less the same: while in most (but not all) cases senses yield higher classification accuracy than words, the difference is rather marginal (in the range of 0.50% to 2.00%); furthermore in some cases words outperform senses. By looking at c_{ave} results, it can be seen that senses always yielded better classification results; but only in 1 out of 9 cases was the c_{ave} difference between words and senses higher than 2.00%. Similarly, looking at c_{fix} (which is perhaps the most representative aspect of the classifier's performance) it can be seen that in 2 out of 9 cases words performed better than senses, in 5 out of 9 cases senses performed better but the difference was less than 2.00% and only in one case was the difference between c_{fix} of senses and c_{fix} of words higher than 2.00% (namely 2.50%). Finally, the overall best result of word-based classification is 79.49% and the corresponding result for senses is 80.16%, a mere 0.67% difference.

In addition to the above tables, we also present some of our results in Figures 1 through 8. In each of these figures we list the average classification accuracy of an algorithm as a function of parameter value, i.e. we plot $c_l = \frac{1}{10} \cdot \sum_{i=1}^{10} c_{li}$ versus. l. In Figures 1, 3, 4, 5 (corresponding to batch MAP, ML, KNN variant 1, and KNN variant 2), the parameter vector π_l is one-dimensional, hence the plots are two-dimensional. Figure 2 corresponds to the recursive version of MAP classification, which uses two parameters: α and h. In this case we have also given a two dimensional plot, because we have found that that h does not influence classification accuracy very much; hence we keep a fixed value $h = 10^{-10}$ and essentially plot c_l versus. α values. Figures 6–9 refer to "σ-FLNMAP with Voting"

experiments. Recall that the "σ-FLNMAP with voting" algorithm is characterized by two parameters: ρ and n_V. Figure 6 is a plot of c_l versus. n_V values, for a fixed value $\rho = 0.94$; Figure 7 is a plot of c_l versus. ρ values, for a fixed n_V value $n_V = 13$; Figure 8 is a three-dimensional plot of c_l as a function of two variables: ρ and n_V. Fig.9 shows the effect of employing several σ-FLNMAP "voters". In particular, an improvement in classification performance results, moreover the substantial benefit of using several voters is that classification accuracy is more stable than that of an individual σ-FLNMAP voter's performance which might fluctuate considerably as illustrated in Fig.9. Moreover Fig.9 shows that for selected values of the vigilance parameter ρ, the classification performance of the "σ-FLNMAP with Voting" might be quite higher than the classification performance of its constituent- individual-σ-FLNMAP modules.

Figure 1 to appear here

Figure 2 to appear here

Figure 3 to appear here

Figure 4 to appear here

Figure 5 to appear here

Figure 6 to appear here

Figure 7 to appear here

Figure 8 to appear here

Figure 9 to appear here

Figures 1 - 9 confirm the marginal improvement in classification accuracy obtained by the use of senses. It can be seen that for every algorithm, the curve corresponding to the sense-based version and to the word-based version are very similar, with the sense-based curve consistently rising slightly above the word-based one.

6 Discussion and Conclusion

We have compared the relative merit of word- and sense-features for purposes of text classification using the Brown Corpus semantic concordance as benchmark. This comparison has been effected by experimenting with all combinations of: 1) four document representations, 2) six different classification algorithms, 3) various values of the parameters of each algorithm, and 4) ten different data sets. The experimental results have been presented in the form of both tables and diagrams. Our experiments have demonstrated that although in some cases the words result in a slightly better classification than senses, in general there exists a marginal advantage of the senses over the words with respect to classification accuracy.

The classification accuracies on the testing data/documents of six algorithms presented in this work is in the range 65-75 % or better. The lower classification accuracy of 1) the two versions of the maximum a posteriori (MAP) algorithm, and 2) the maximum likelihood (ML) algorithm can be attributed to the "Naive Bayes" assumption of the statistical independence of the words. The two variants of the k-nearest neighbor (KNN) algorithm gave better results than the previous algorithms. Nevertheless their performance is limited by the weighted summation they perform which may "smooth out" useful discriminatory features. The overall best results were obtained by the "σ-FLNMAP with Voting" classifier. The good generalization on the testing data/documents of the latter classifier is attributed to the calculation of the largest "uniform boxes" in the training data sets as explained in the text. Note that the computation of the largest uniform boxes in the training data using the technique of *maximal expansions* is known to improve classification accuracy [18]. Moreover the employment of several "voting σ-FLNMAP modules" results in a stability and high classification accuracy; this can be attributed to "data noise cancellation" due to the different permutations of the training data used to train different σ-FLNMAP modules.

We have performed the words/senses comparison assuming complete knowledge of the senses. Nevertheless, in a practical classification task the senses would have to be obtained by a disambiguation step which, in all probability, would introduce a significant error. It seems likely that the 1-2% classification accuracy advantage obtained in the experiments reported here would be more than offset by faulty disambiguation. While the evidence presented here cannot be considered conclusive it certainly seems that sense-based classifiers do not present an attractive alternative to word-based ones.

References

[1] E. Agirre, and D. Martinez, "Exploring automatic word sense disambiguation with decision lists and the Web". In *Procedings of the COLING 2000 Workshop on Semantic Annotation and Intelligent Content*, 2000.

[2] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model", Universite de Montreal, Montreal, Quebec, Canada, H3C 3J7, Technical Report #1178, 2000.

[3] A.L. Berger, S.A. Della Pietra, and V.J. Della Pietra, "A maximum entropy approach to natural language processing". *Comp. Linguistics*, vol.22, pp.39-71, 1996.

[4] G. Birkhoff, *Lattice Theory*, American Mathematical Society, Colloquium Publications, vol. 25, 1967.

[5] L. Breiman, "Bagging predictors". Technical Report 421, Dept. of Statistics, Univ. of California at Berkeley, 1994.

[6] R. Caruana, and D. Freitag, "Greedy attribute selection", in *Proceedings of the 11th Int. Conference on Machine Learning*, pp.26-28, 1994.

[7] W.W. Cohen, "Text categorization and relational learning", in *Proceedings of the 12th International Conference in Machine Learning*, pp.124-132, 1995.

[8] W.W. Cohen, and H. Hirsh, "Joins that generalize: Text classification using WHIRL", in *Proc. of the Fourth Int. Conference on Knowledge Discovery and Data Mining*, 1998.

[9] S.A. Della Pietra, V.J. Della Pietra, and J. Laferty, "Inducing features of random fields". *IEEE Trans. on Pattern Anal. and Mach. Intel*, vol. 19, 1997.

[10] P. Domingos, and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss", *Mach. Learning*, vol. 29, pp.103-130, 1997.

[11] R.O Duda, P.E. Hart, and D.G. Stork, *Pattern classification*, Wiley, 2001.

[12] G. Escudero, L. Marquez, and G. Rigau, "A comparison between supervised learning algorithms for word sense disambiguation", in *Proceedings of the 4th Conference on Computational Natural Language Learning, CoNLL'2000*, pp. 31-36, 2000.

[13] W. N. Francis, and H. Kucera, *Frequency Analysis of English Usage: Lexicon and Grammar*, Houghton Mi- in Company, Boston, 1982.

[14] A. Fujii, and T. Ishikawa, "Utilizing the World Wide Web as an encyclopedia: extracting term descriptions from semi-structured texts", in *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, pp.488-495, 2000.

[15] Q.Q. Huynh, L.N. Cooper, N. Intrator, and H. Shouval, "Classification of underwater mammals using feature extraction based on time frequency analysis and BCM theory", *IEEE Trans. on Signal Proc.*, vol.46, pp.1202-1207, 1998.

[16] T. Joachims, D. Freitag, and T. Mitchell, "Webwatcher: a tour guide for the World Wide Web", in *Proceedings of the Int. Joint Conf. on Artificial Intelligence*, 1997.

[17] T. Joachims, "Text categorization with support vector machines: learning with many relevant features", in *Proceedings of European Conference on Machine Learning*, 1998.

[18] V.G. Kaburlasos, and V. Petridis, "Fuzzy Lattice Neurocomputing (FLN) models", *Neural Networks*, vol.13, pp.1145-1170, 2000.

[19] V.G. Kaburlasos, P. Fragkou, V. Petridis, and Ath. Kehagias, "A comparison of words and senses as features for text classification", *Proc. of the 24th Annual Intl. Conf. on Research and Development in Information Retrieval (SIGIR 2001) sponsored by the Association for Computing Machinery (ACM)*, New Orleans LA, 7-12 September 2001.

[20] J. Karlgren, and D. D. Cutting, "Recognizing text genres with simple metrics using discriminant analysis", Proceedings of COLING 94, Kyoto, 1994.

[21] D. Koller, and M. Sahami, "Hierarchically classifying documents using very few words", in *Proceedings of the 14th Int. Conference on Machine Learning*, pp.170-178, 1997.

[22] R. Kosala, and H. Blockeel, "Web mining research: a survey". *ACM SIGKDD Explorations*, vol. 2, pp.1-15, 2000

[23] D.D. Lewis, *Representation and Learning in Information Retrieval*. Ph.D. Thesis, Dept. of Computer Science, Univ. of Massachussets, 1992.

[24] D.D. Lewis, R.E. Schapire, J.P. Callan, and R. Papka, "Training algorithms for linear text classifiers", in *Research and Development in Information Retrieval*, pp.298-306, 1996.

[25] D.D. Lewis, and K.S. Jones, "Natural language processing for information retrieval", *Comm. of the ACM*, vol.39, pp.92-101, 1996.

[26] C.D. Manning, and H. Schuetze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.

[27] B. Masand, G.Linoff, and D. Waktz, "Classifying news stories using memory based reasoning", in *SIGIR 92*, pp.59-65, 1992.

[28] G.A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K.J. Miller, "Introduction to WordNet: an on-line lexical database", in *International Journal of Lexicography* vol.3, 1990, pp. 235 - 244.

[29] T.M. Mitchell, *Machine Learning*, The McGraw-Hill Companies, Inc. 1997.

[30] D. Mladenic, *Machine learning of non-homogeneous distributed text data*, Ph.D. dissertation, Dept. of Computer and Information Science, Univ. of Ljubljana, 1998.

[31] V. Petridis, and V.G. Kaburlasos, "Learning in the framework of fuzzy lattices", *IEEE Transactions on Fuzzy Systems*, vol. 7, pp. 422-440, 1999. Errata in *IEEE Transactions on Fuzzy Systems*, vol. 8, p. 236, 2000.

[32] Petridis V, and Kaburlasos VG, "An intelligent mechatronics solution for sutomated tool guidance in the epidural surgical procedure", In *Proc. 7th Conf. on Mechatronics and Machine Vision in Practice (M2VIP'00)*, Hervey Bay, Queensland, Australia, pp. 201-206, 2000.

[33] V. Petridis, and V.G. Kaburlasos, "Clustering and classification in structured data domains using Fuzzy Lattice Neurocomputing (FLN)", *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 2, 2001.

[34] V. Petridis, V.G. Kaburlasos, P. Fragkou, and Ath. Kehagias, "Text Classification Using the σ-FLNMAP Neural Network", *Proc. of the 2001 Intl. Joint Conf. on Neural Networks (IJCNN'2001)*, Washington D.C., 14-19 July 2001.

[35] V. Petridis, and A. Kehagias, "Modular neural networks for Bayesian classification of time series and the partition algorithm", *IEEE Trans. on Neural Networks*, vol.7, pp.73-86, 1996.

[36] V. Petridis, and A. Kehagias, *Predictive Modular Neural Networks: Time Series Applications*, Kluwer, 1998.

[37] M.Sahami, M. Hearst, and E.Saund. "Applying the multiple cause mixture model to text categorization". In *Proceedings of the 13th International Conference in Machine Learning*, pp.435-443, 1996.

[38] Y. Yang, and X. Liu, "A re-examination of text categorization methods", in *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval*, pp.42-49, 1999.

[39] Y. Yiming, "An evaluation of statistical approaches to text categorization", *Information Retrieval*, 1998.

[40] Y. Yiming, "Noise reduction in a statistical approach to text categorization". In *SIGIR 95*, pp.256-263, 1995.

Table 1. The document collection categories and the number of documents in each category for both the original categories and the new ones.

	Original Categories	no. Documents in Original Category	New Category	no. Documents in New Category
1	Press: Reportage (A)	7	Press	12
2	Press: Editorial (B)	2		
3	Press: Reviews (C)	3		
4	Skills and Hobbies (E)	14	Skills and Hobbies	14
5	Popular Lore (F)	19	Popular Lore	19
6	Belles Lettres/Biography/Memoirs (G)	18	Belles Lettres etc.	18
7	Miscellaneous (H)	12	Miscellaneous	12
8	Learned (J)	43	Learned	43
9	General Fiction (K)	29	Fiction	64
10	Mystery and Detective Fiction (L)	11		
11	Science Fiction (M)	2		
12	Adventure and Western Fiction (N)	10		
13	Romance and Love Story (P)	6		
14	Humor (R)	6		

Table 2. The distribution of documents from each category in the training and testing data sets.

Category	CAT1	CAT2	CAT3	CAT4	CAT5	CAT6	CAT7
No. of Documents in Train Set	8	10	13	12	8	29	43
No. of Documents in Test Set	4	4	6	6	4	14	21

Table 3. The sizes of word and sense vocabularies for each of the data sets.

Data Set	Size N_W of Word Vocabulary	Size N_S of Sense Vocabulary
Set_0	15860	20134
Set_1	15698	19976
Set_2	15684	20094
Set_3	15872	20246
Set_4	15579	19943
Set_5	15833	20138
Set_6	15789	20147
Set_7	15683	19940
Set_8	15796	20128
Set_9	15705	19994

Table 4. List of the algorithms used, their parameters and the document representations to which each algorithm is applied (WB=Word Boolean, WF=Word Frequency, SB=Sense Boolean, SF=Sense Frequency).

Algorithm	**Parameters**	**Document Representations**
Maximum A Posteriori (MAP) – Batch version	á	WF, SF
Maximum A Posteriori (MAP) – Recursive Version	á, h	WF, SF
Maximum Likelihood (ML) – Naïve Bayes	á	WF, SF
KNN – Variant no.1	P	WF, WB, SF, SB
KNN – Variant no.2	P	WF, WB, SF, SB
ó-FLNMAP with Voting	ñ, n_V	WF, WB, SF, SB

Table 5. Classification accuracy for the MAP (Batch version) classification algorithm.

	words	senses
c_{min}	59.50%	64.40%
c_{fix}	72.20%	71.50%
c_{max}	72.20%	71.90%
c_{ave}	67.10%	68.80%
c_{val}	71.70%	70.30%

Table 6. Classification accuracy for the recursive MAP (Recursive version) classification algorithm.

	words	senses
c_{min}	60.30%	60.70%
c_{fix}	65.90%	67.30%
c_{max}	70.20%	70.00%
c_{ave}	65.40%	65.80%
c_{val}	66.40%	67.10%

Table 7. Classification accuracy for the ML classification algorithm.

	words	senses
c_{min}	59.66%	64.57%
c_{fix}	67.18%	69.02%
c_{max}	72.20%	72.20%
c_{ave}	67.19%	68.94%
c_{val}	70.66%	70.67%

Table 8. Classification accuracy for the KNN variant no.1 classification algorithm.

	words / Rel. Freq	senses / Rel. Freq	words / Boolean	senses / Boolean
c_{min}	60.00%	62.40%	68.80%	70.70%
c_{fix}	63.70%	65.30%	72.50%	72.40%
c_{max}	65.30%	66.80%	74.10%	73.20%
c_{ave}	62.70%	64.60%	71.20%	72.00%

Table 9. Classification accuracy for the KNN variant no.2 classification algorithm.

	words / Rel. Freq	Senses / Rel. Freq	words / Boolean	senses / Boolean
c_{min}	61.00%	63.90%	69.20%	71.20%
c_{fix}	61.90%	64.70%	70.20%	71.50%
c_{max}	63.90%	66.60%	71.20%	72.40%
c_{ave}	62.40%	65.50%	70.00%	71.60%

Table 10. Classification accuracy for the "ó-FLNMAP with Voting" classification algorithm.

	words / Rel. Freq	Senses / Rel. Freq	words / Boolean	senses / Boolean
c_{min}	61.35%	61.69%	49.15%	57.62%
c_{fix}	74.03%	74.06%	73.38%	74.06%
c_{max}	79.49%	78.81%	78.64%	80.16%
c_{ave}	70.65%	70.97%	70.40%	71.51%

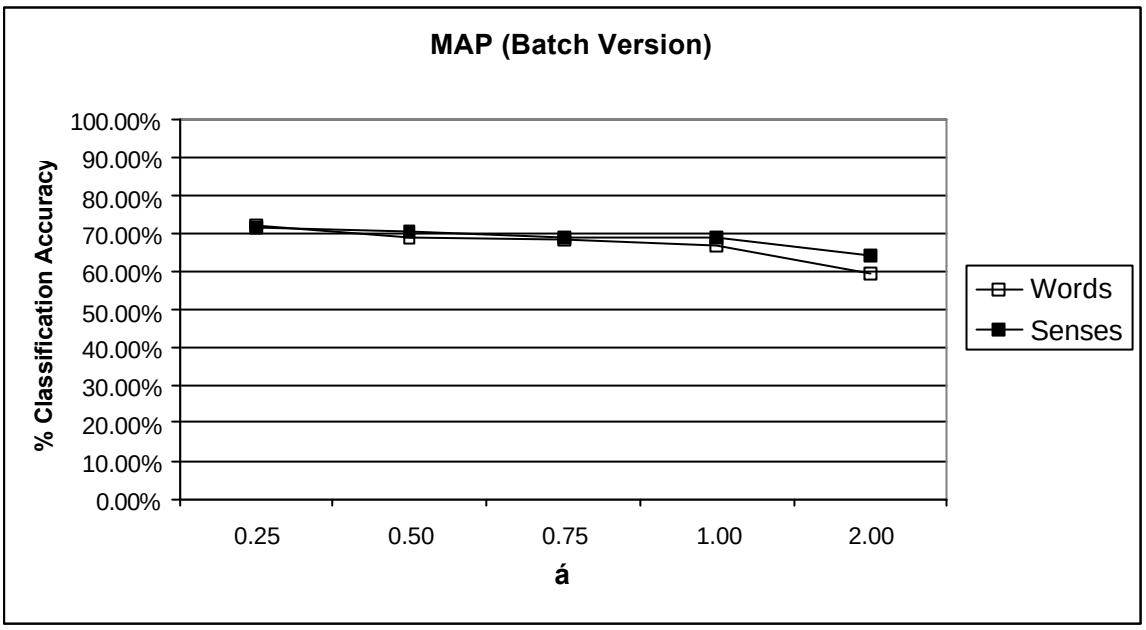

Figure 1 Performance of the Maximum A Posteriori (MAP) classification algorithm (batch version) on the fully tagged Brown Corpus documents. The percentage of the correctly classified testing documents is plotted for both word- and sense- representations versus the algorithm's tuning parameter á. The sense-representation marginally outperforms the word-representation.

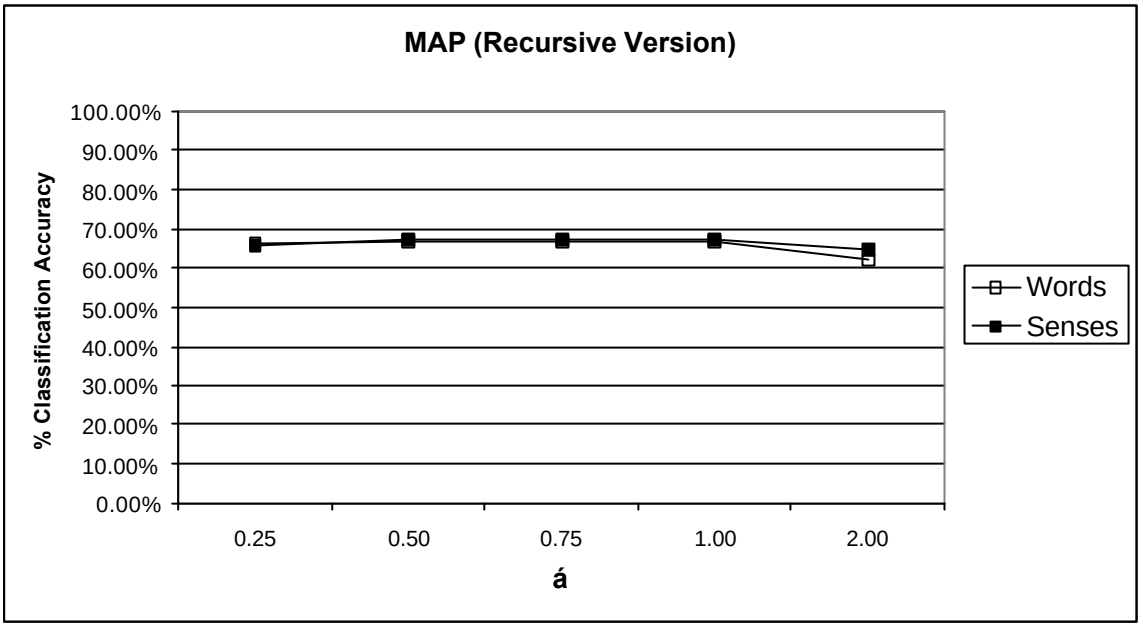

Figure 2 Performance of the Maximum A Posteriori (MAP) classification algorithm (recursive version) on the fully tagged Brown Corpus documents. The percentage of the correctly classified testing documents is plotted for both word- and sense-representations versus the algorithm's tuning parameter á, and a constant value for the algorithm's tuning parameter $h=10^{-10}$. The sense-representation has produced marginally better results than the word-representation. The picture does not change for alternative values of the tuning parameter h.

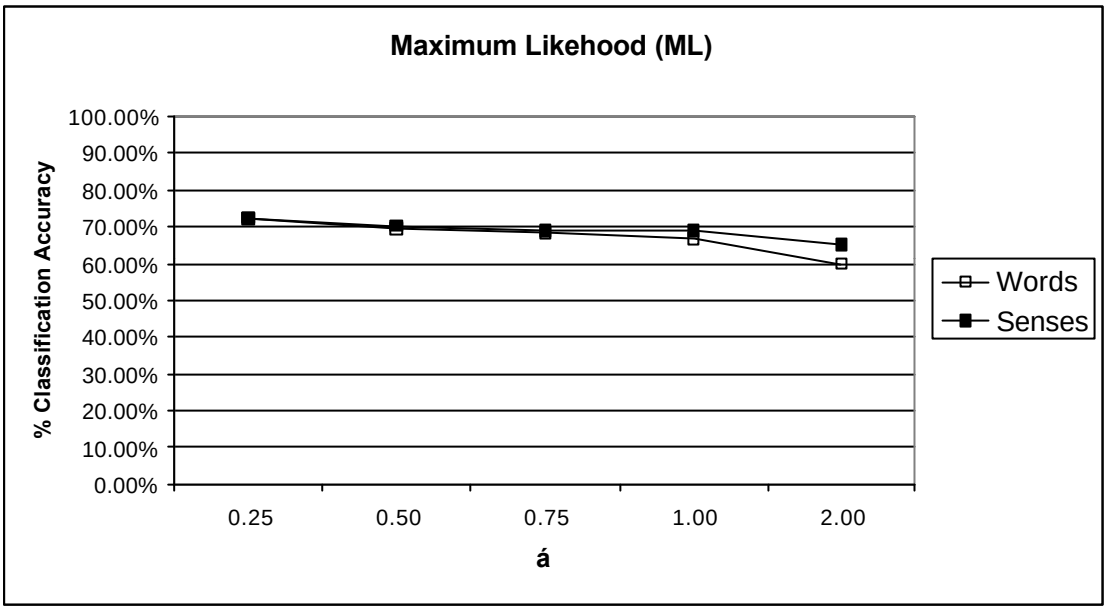

Figure 3 Performance of the Maximum Likelihood (ML) algorithm for classification on the fully tagged Brown Corpus documents. The percentage of the correctly classified testing documents is plotted for both word- and sense- representations versus the algorithm's tuning parameter á. The sense-representation yields marginally better results than the word-representation.

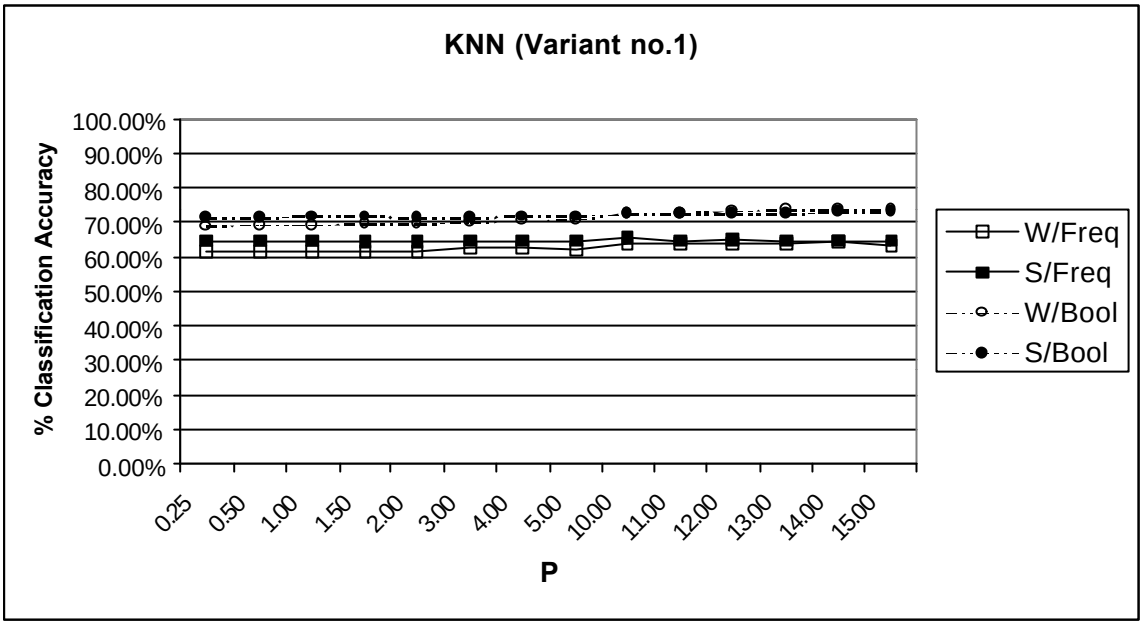

Figure 4 Performance of the K – Nearest Neighbor (KNN) classification algorithm (variant no.1) on the fully tagged Brown Corpus documents. The percentage of the correctly classified testing documents is plotted for all of word-, sense-, frequency-, and boolean- representations versus the algorithm's tuning parameter P. The boolean-representations have clearly outperformed the frequency-representations, whereas a sense-representation has yielded marginally better results than the corresponding word-representation.

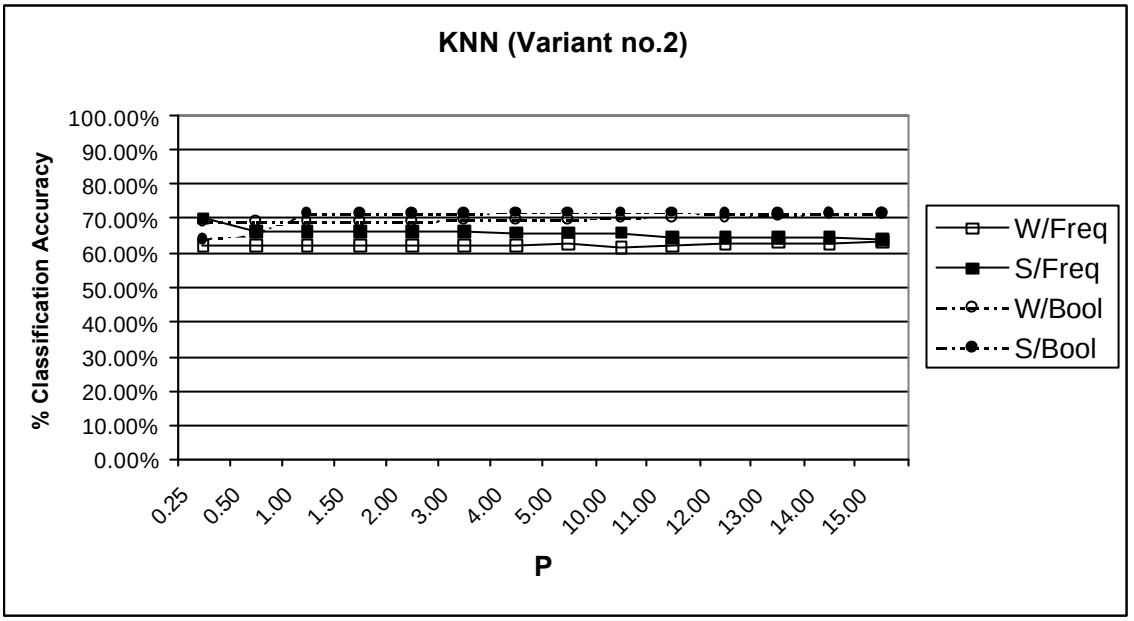

Figure 5 Performance of the K – Nearest Neighbor (KNN) classification algorithm (variant no.2) on the fully tagged Brown Corpus documents. The percentage of the correctly classified testing documents is plotted for all of word-, sense-, frequency-, and boolean-representations versus the algorithm's tuning parameter P. The boolean-representations have clearly outperformed the frequency-representations, whereas a sense-representation has yielded marginally better results than the corresponding word-representation.

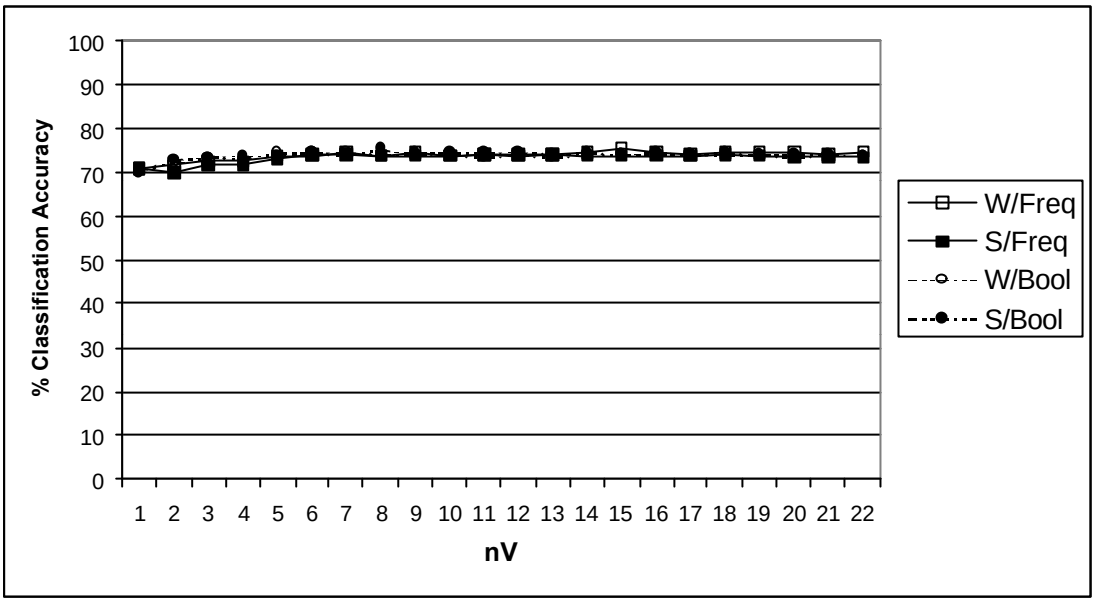

Figure 6 Performance of the "ó-FLNMAP with Voting" scheme for classification on the fully tagged Brown Corpus documents. The percentage of the correctly classified testing documents is plotted versus the number n_V of voters for a fixed value of the vigilance parameter ñ=0.94. From a practical point of view all four of word-, sense-, frequency-, and boolean-representations have yielded quite similar results.

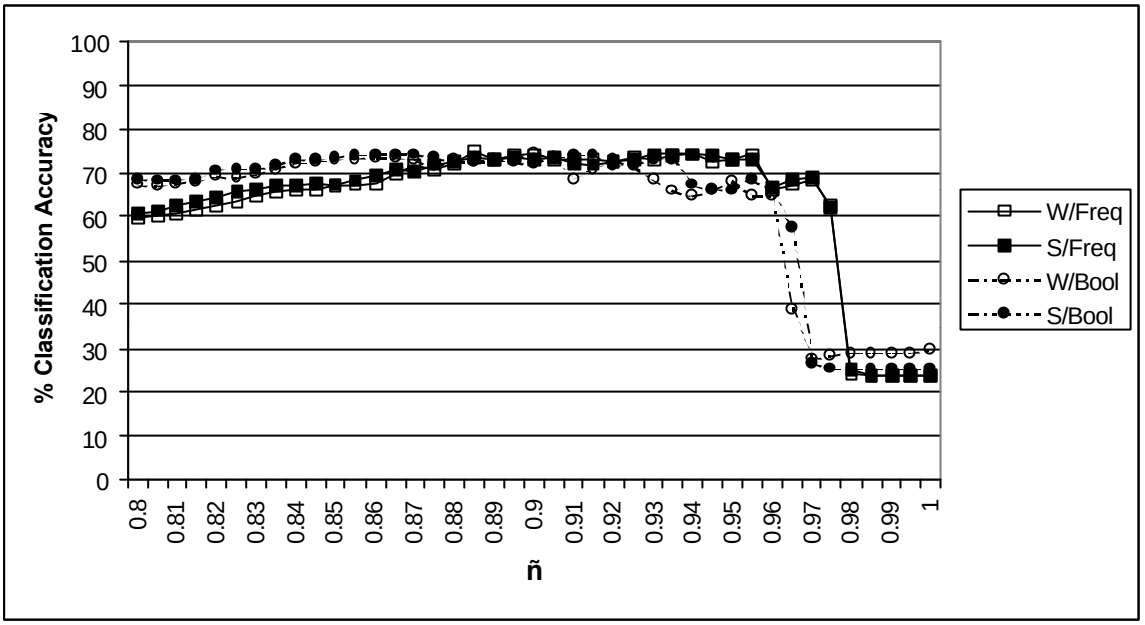

Figure 7 Performance of the "ó-FLNMAP with Voting" scheme for classification on the fully tagged Brown Corpus documents. The percentage of the correctly classified testing documents is plotted versus the vigilance parameter ñ for a fixed value of the number of voters n_V= 13. A sense-representation has produced marginally better results that the corresponding word-representation, whereas the frequency- representations have outperformed the boolean- ones.

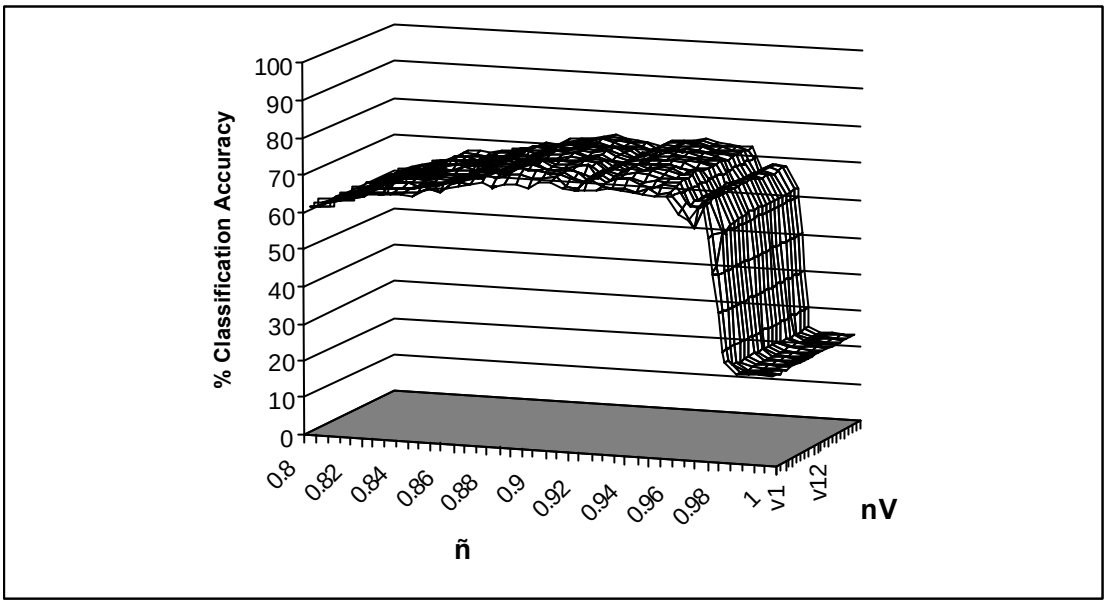

Figure 8 Performance of the "ó-FLNMAP with Voting" scheme for classification on the fully tagged Brown Corpus documents. The percentage of the correct classified testing documents is plotted versus both the vigilance parameter ñ and the number n_V of voters for the sense/frequency representation. The classification accuracy remains quite stable at its maximum for ñ≈ 0.94 and n_V≈ 13.

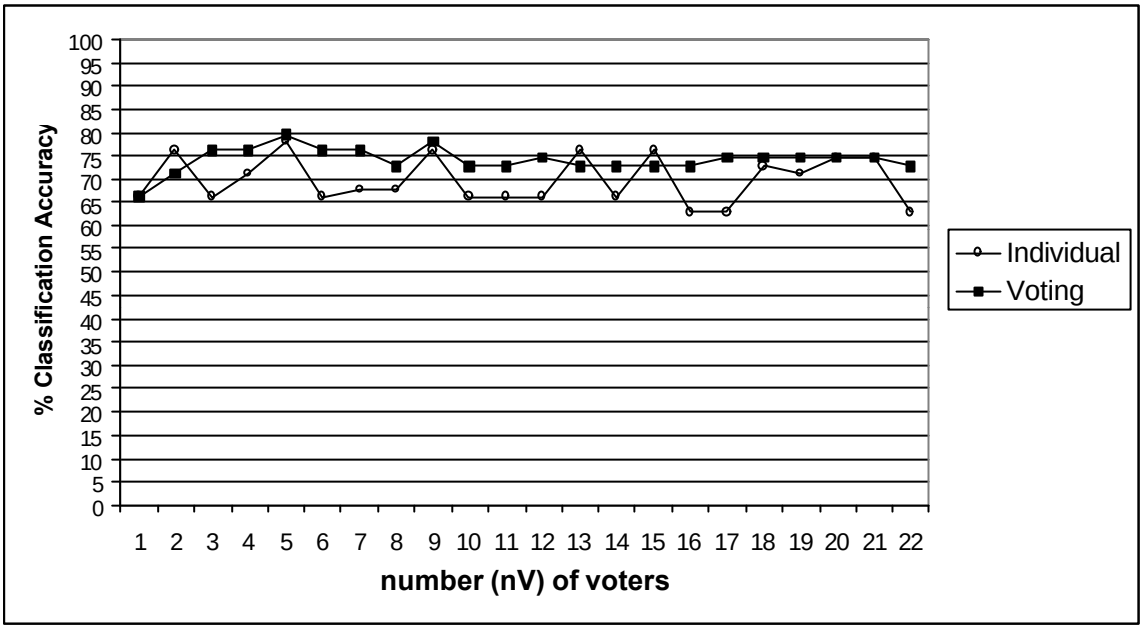

Figure 9 Percentage of classification accuracy of both individual ó-FLNMAP modules and the "ó-FLNMAP with Voting" scheme for classification on the fully tagged Brown Corpus documents versus the number n_V of voters, using word/frequency representations. The ó-FLNMAP scheme with several voters implies a stable improvement over an individual ó-FLNMAP module whose performance fluctuates considerably.