COMPARISON OF THE EFFECTS OF LEXICAL AND ONTOLOGICAL INFORMATION ON TEXT CATEGORIZATION

by

CESAR KOIRALA

(Under the Direction of Khaled Rasheed)

ABSTRACT

This thesis compares the effectiveness of using lexical and ontological information for text categorization. Lexical information has been induced using stemmed features. Ontological information, on the other hand, has been induced in the form of WordNet hypernyms. Text representations based on stemming and WordNet hypernyms were evaluated using four different machine learning algorithms on two datasets. The research reports average F1 measures as the results. The results show that, for the larger dataset, stemming-based text representation gives better performance than hypernym-based text representation even though the later uses a novel hypernym formation approach. However, for the smaller data set with relatively lower feature overlap, hypernym-based text representations produce results that are comparable to the stemming-based text representation. The results also indicate that combining stemming-based representation and hypernym-based representation produces an improvement in the performance for the smaller dataset.

INDEX WORDS: Text categorization, Stemming, WordNet hypernyms, Machine Learning.

COMPARISON OF THE EFFECTS OF LEXICAL AND ONTOLOGICAL INFORMATION ON TEXT CATEGORIZATION

by

CESAR KOIRALA

B.E., Pokhara University, Nepal, 2003

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2008

COMPARISON OF THE EFFECTS OF LEXICAL AND ONTOLOGICAL INFORMATION ON TEXT CATEGORIZATION

by

CESAR KOIRALA

Major Professor: Khaled Rasheed

Committee: Walter D. Potter
Nash Unsworth

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
August 2008

DEDICATION

I dedicate this to my parents and brothers for loving me unconditionally.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Khaled Rasheed, for his constant support and guidance. This thesis would not have been the same without his expert ideas and encouragements. I would also like to thank Dr. Walter D. Potter and Dr. Nash Unsworth for their participation on my committee. I am very thankful to Dr. Michael A. Covington whose lectures on Prolog and Natural Language Processing gave me a solid foundation to conduct this research. My sincere thanks to Xia Qu for being my project partner in several courses that led to this thesis. Thanks to Dr. Rasheed, Eric, Shiwali, Sameer and Prachi for editing the thesis. Lastly, I would like to thank all my friends at UGA, especially the Head Bhangers, for unforgettable memories.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

1.1. BACKGROUND

Text categorization is the process of automatically assigning natural language texts to one or more predefined categories. With the rapid growth in the number of online documents, text categorization has become an important tool for tasks like document organization, e-mail routing, news filtering, spam filtering etc.

Text categorization can either be done using a rule-based approach or by constructing a classifier using supervised learning. Rule-based approach involves manual generation of a set of rules for specifying the category of the text and is highly accurate. However, as it needs domain experts to compose rules, it is costly in terms of labor and time. Moreover, rules are domain dependent and hence rarely transferable to another data set. Supervised learning, on the other hand, involves automatic creation of classification rules using labeled texts. In supervised learning, a classifier is first trained with some pre-classified documents (labeled texts). Then, the trained classifier is used to classify unseen documents. As rule-based approach is time consuming and domain dependent, researchers have focused more on machine learning algorithms for supervised learning of classification models.

In order to use machine learning algorithms for automatic text categorization, the texts need to be represented as vectors of features. One of the most widely used approaches for generating feature vectors from texts is the bag-of-words model. In the simplest form of the bag-

of-words model, features are the words that appear in a document. Such models do not consider any linguistic information. As the *semantic relationship between words* is not taken into account, it can result in the following two cases:

Case A: Two texts which are of the same subject but are written using different words, conveying the same meaning, may not be categorized into the same class.

Case B: Two texts using different forms of the same word may not be identified as belonging to the same class.

For dealing with Case B, we can use *stemmed words* instead of normal words. Stemming ensures that different forms of a word are changed into the same *stem*. Although the studies on the effects of stemming on categorization accuracy are not conclusive, it is commonly used for the reduction in the dimensionality of the feature space. Case A can be handled by using *hypernyms* from WordNet [9]. A hypernym is a word or a phrase that has a broad meaning. It encompasses many specific words which have similar meaning. So, even if two texts are different at the level of words, there is a fair chance that they are similar at the level of hypernyms. Using a rule-based learner, RIPPER [6], Scott and Matwin [5] were able to show a significant improvement in the classification accuracy when the bag-of-words representation of text was replaced by hypernym density representation.

Stemming and WordNet hypernyms are two different ways of inducing linguistic information into the process of text categorization. Stemming is based on the *morphological* analysis of the text and helps in the induction of *lexical* information. Hypernym analysis, on the other hand, is a way of providing *ontological* information. So there can be a debate about which kind of linguistic information better serves the purpose of improving the classification accuracy. The aim of this research is to compare the effect of lexical (stemming) and ontological

(hypernym) information on classification accuracy. For that we have compared the performance of a bag-of-words model that uses stemmed words as tokens with one that uses hypernyms.

1.2. MOTIVATION FOR THE STUDY

Scott and Matwin [4] clearly state that the hypernym-based improvement is possible only in smaller datasets. They found that for larger datasets, like the Reuters 21578 collection [16], hypernym density representation of text cannot compete with normal bag-of-words representation. The reader may then wonder *why we even bother comparing such a method to another method*. Considering the facts that Scott and Matwin [4] used binary features rather than real valued density measurements and a low height of generalization for hypernyms, we are left with reasons to believe that hypernyms might improve the classification accuracy if those limitations are eliminated. Besides, an improvement in text classification using WordNet synsets and the K-Nearest-Neighbors method has recently been shown in [3]. So, giving the hypernym-based approach (using the WordNet ontology) a chance to compete with the stemming-based approach seemed fair. To take care of the previously mentioned limitations, we have used real valued density measurements for the features. We have also suggested a novel way of obtaining the hypernyms which is not based on height of generalization as in [4] and [5]. Also, although there has been a detailed survey on the effectiveness of different machine learning algorithms on the bag-of-words model (e.g. [2]), no comparison of the algorithms for the hypernym-based model could be found in the literature. Here, we present the comparison of stemming-based bag-of-words model with hypernym-based bag-of-words model using four different machine learning algorithms. They are naïve Bayes classifiers, Bayesian networks, decision trees and support vector machines.

1.3. OUTLINE OF THE THESIS

The rest of the thesis is organized as follows. Chapter 2 presents a description of stemming and WordNet ontology. It provides a brief introduction to Porter's stemming algorithm and discusses a novel way of converting normal words to hypernyms. The different machine learning algorithms used in the research are explained in chapter 3. In chapter 4, the preprocessing steps carried out on the Reuters 21578 dataset are discussed. The actual experiments and results are presented in chapter 5. Chapter 6 shows the experiments and results for 20-Newsgroups dataset. Finally, the thesis is concluded in chapter 7 with a discussion of the results.

CHAPTER 2

LEXICAL AND ONTOLOGICAL INFORMATION

2.1. MORPHOLOGY, LEXICAL INFORMATION AND STEMMING

Morphology is the study of the patterns of word formation. Word formation can be seen as a process in which smaller units, *morphs*, combine together to form a larger unit. For example, the word 'stemming' is formed using 'stem' and 'ing'. English morphs can be either *affixes* or they can be *roots*. An affix is a generic name given to *prefixes* and *suffixes*. A root is the unit that bears the core meaning of a word. Hence in the given example, 'ing' is the suffix attached to the core 'stem' in order to form the word 'stemming'. However, combining roots to zero or more affixes is not the only way of forming English words. There are other rules like *vowel change*. One example is forming 'ran' from 'run' using a vowel change.

For effective bag-of-words based text categorization, it is important to compute accurate statistics about the proportion of the words occurring in the text. This is because the bag-of-words model recognizes similarity in the texts based on the proportions of the words. Hence, sometimes, it becomes desirable to ignore the minor differences between different forms of the same word and change them into the same form. This means we treat 'tiger' and 'tigers' as different forms of the same word and change them into the common form 'tiger'. This process provides lexical information to the bag of words model. In order to accomplish this, we need a process which can analyze the words morphologically and return their roots. Stemming is one such process that removes suffixes from the word. It ensures that morphologically different

forms of a word are changed into the same *stem* and thus helps in inducing lexical information. It is possible for stemming algorithms to produce stems that are not the roots of the words. Sometimes they even produce stems that are incomplete and make no sense. For example, a stemming algorithm might return 'acquir' as the stem of the word 'acquiring'. However, as all the morphological variations of a word are changed into the same stem, the goal of getting accurate statistics of a word is achieved. So as long as we get consistent stems for all the morphological variations of the words present in the texts, any string is acceptable as a stem.

One of the commonly used stemming algorithms is the Porter Stemming Algorithm proposed in [15]. It removes suffixes by applying a set of rules. Different rules deal with different kinds of suffixes. Each rule has certain conditions that need to be satisfied in order for the rule to be effective. The words in a text are checked against these rules in a sequential manner and if the conditions in the rule are met, the suffixes are either removed or changed. We used the prolog version of Porter's Stemming Algorithm written by Philip Brooks [18].

2.2. WORDNET ONTOLOGY AND HYPERNYMS

WordNet is an online lexical database that organizes words and phrases into synonym sets, called synsets, and records various semantic relationships between these synsets. Each synset represents an underlying lexical concept. The synsets are organized into hierarchies based on *is-a* relationships. Any word or phrase *Y* is a *hypernym* of another word or a phrase *X* if every *X is-a Y*. Thus the hypernym relationship between synsets is actually a relationship between lexical concepts and hence works as ontological information. In figure 2.1, every word or a phrase in the chain is hypernym of another word or a phrase that occurs above it in the hierarchy. For example, 'mammal' is a hypernym of 'big cat', 'feline' and 'carnivore'. In other words, mammal

is a broader concept that can encompass all those specific concepts. By changing normal words to hypernyms, we ensure that the bag-of-words model is able to correctly compute statistics about the similar concepts occurring in the texts. This change increases the chance that two texts of the same subject matter, using different words, are categorized into the same class.

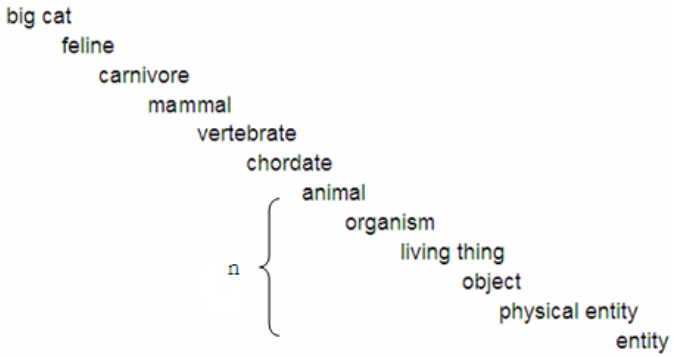

Figure 2.1: WordNet hierarchy for the word ‘tiger’

WordNet hypernym-based text representation was first suggested in [5] and further tested in [4]. Changing a normal text into a hypernym-based text requires replacing all the words in the text with their hypernyms. However, before doing that we need to decide which hypernym to choose from the chain of hypernyms available for each word. To solve this problem, Scott and Matwin used a parameter *h*, *height of generalization*, which controls the number of steps upward in the hypernym chain for each word [5]. This means at h=0, the hypernym is the word itself. In Figure 2.1, it is ‘tiger.’ At h=1, it is ‘big cat’. However, this method does not guarantee that two words that represent the same concept are changed into the same hypernym. For selecting appropriate hypernyms, we suggest a novel technique that is not based on height of generalization. We introduce a variable n which is the depth from the other end of the chain. This

means at n=0, the hypernym is the last word in the hierarchy. In Figure 2.1, it is ‘entity’. At n=3, it is ‘object’. The rationale behind doing so can be explained with the following example. At n=5, the hypernym of ‘tiger’ is ‘animal’ and so is the hypernym of ‘carnivore’. This means we were successful to show that both words represent the same concept. This method of obtaining hypernyms ensures that any two words representing the same concept are changed into the same hypernym.

Smaller values of n produce hypernyms that represent more general concepts. However, if the value of n is too small, then the concepts are over generalized. Hence, it results in similar synsets for many unrelated concepts. On the other hand, if the value is too large, the concepts might not be generalized. Hence, we might get the words themselves as the hypernyms. The appropriate level of generalization depends upon the characteristics of the text and the version of the WordNet being used [5]. In this experiment we use WordNet 3.0 and report the results for six different values of n. The values of n used for generating hypernyms were 5, 6,7,8,9 and 10.

CHAPTER 3

LEARNING ALGORITHMS

This chapter describes the classification algorithms used in the experiment. We experimented with decision trees, naïve Bayes classifiers, Bayesian networks and support vector machines.

3.1. DECISION TREES

Decision trees are very popular for classification and prediction problems because they can be learned very fast and can be easily converted into *if-then rules*, which have better human readability. They classify instances that are represented as attribute-value pairs. A decision tree classifier takes the form of a tree structure with nodes and branches. A node is a decision node if it specifies some test to be carried out on an attribute of an instance. It is a leaf node if it indicates the target classes of the instances. For classification, the attributes are tested at the decision nodes starting from the root node. Depending upon the values, the instances are sorted down the tree until all the attributes are tested. Then, the classification of an instance is given at one of the leaf nodes. Table 3.1 shows five instances that belong to different classes of a common concept 'Game'. A decision tree that can classify all these instances to their proper classes has been shown in figure 3.1. The first attribute {yes, bat, 11, yes} will be sorted down the leftmost branch of the decision tree shown in the figure and hence classified as belonging to the class 'cricket'.

Table 3.1: Instances of the target concept ‘Game’

Ball_ involved	Played_with	Players	Outdoor	Game
yes	bat	11	yes	Cricket
no	hands	2	no	Chess
yes	feet	11	yes	Soccer
yes	bat	2	no	ping pong
yes	bat	11	no	indoor cricket

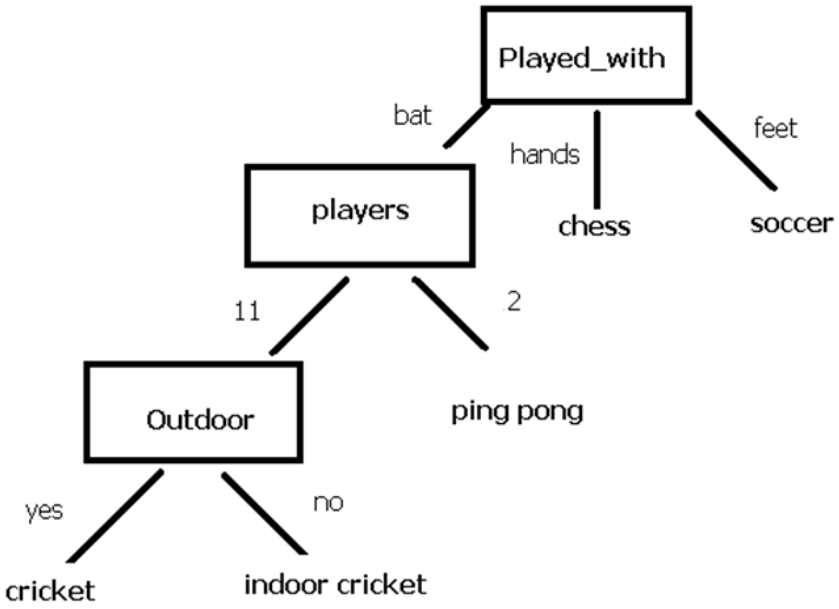

Figure 3.1: A decision tree for the concept ‘Game’

For constructing decision trees for the experiment, we relied on C4.5, a variant of ID3 learning algorithm [20]. ID3 forms a tree working in a top down fashion, selecting the best attribute as the root node. This selection is based on *information gain.* Information gain of an attribute is the expected reduction in entropy, a measure of homogeneity of the set of instances, when the instances are classified by that attribute alone. It measures how well the attribute would classify the given examples [21]. Once the attribute for the root node is determined, branches are

created for all the values associated with that attribute and then next best attribute is selected in a similar manner. This process continues for all the remaining attributes until the leaf nodes, displaying classes, are reached. The decision tree shown in figure 3.1 has been learned using ID3. C4.5 is an extension of ID3 designed such that it can handle missing attributes.

The use of decision trees for the task of text classification on the Reuters data set has been shown in several research papers including [7] and [8]. Apte, et al. achieved the high accuracy of 87.8 % using a system of 100 decision trees [8]. Decision trees produce high classification accuracy, compared to support vector machines, on the Reuters text collection [2].

3.2. BAYESIAN LEARNING

Bayesian learning in a learning method based on probabilistic approach. Using Bayes's rule, Bayesian learning algorithms can generate classification models for a given data set. This section first discusses Bayes's rule, and then it gives brief introductions to the naïve Bayes classifier and Bayesian networks.

3.3. BAYES RULE AND ITS RELEVANCE IN MACHINE LEARNING

For two events A and B, Bayes's rule can be stated as:

$$P(A|B) = P(B|A) * P(A) / P(B)$$

Here, P (A) is the prior probability of A's occurrence. It does not take into account any information about B. P (B) is the prior probability of B's occurrence. It does not take into account any information from A. P (A|B) is the conditional probability of A, given B. Similarly, P (B|A) is the conditional probability of B, given A. *How is this rule relevant to machine learning*? This question can be answered using the equation shown below. It has been adapted

from [21].

$$P(h|D) = P(D|h)\, P(h)/ P(D)$$

This equation is based on Bayes's theorem. Here, h is the hypothesis that best fits the given set of training instances D. P (h) is the prior probability that the hypothesis holds and P (D) is the probability that the training data will be observed. P (D|h) is the probability of observing D, given h and p (h|D) is the probability that the hypothesis holds, given D. Learning of such hypothesis leads to the development of classifiers based on probabilistic models. We will further discuss the relevance of Bayes's rule, in the light of two learning algorithms, in the following sections.

3.4. NAÏVE BAYES CLASSIFIER

Let us assume that the instances in a data set are described as attribute-value pairs. Let X= $\{x_1, x_2\ldots,x_n\}$ represent the set of attributes and C= $\{c_1, c_2\ldots,c_n)$ represent the classes.

Let c_i be the most likely classification of a given instance, given the attributes $x_1, x_2\ldots,x_n$.

Using Bayes's rule,

$$P(c_i| x_1, x_2\ldots,x_n) = P(x_1, x_2\ldots,x_n \mid c_i)\, P(c_i)/P(x_1, x_2\ldots,x_n)$$

As $P(x_1, x_2\ldots,x_n)$ is constant and independent of c_i, we get that the class c_i which maximizes $P(c_i \mid x_1, x_2\ldots,x_n)$ is the one that maximizes $P(x_1, x_2\ldots,x_n \mid c_i)\, P(c_i)$. This classifier is called naïve Bayes because while calculating $P(x_1, x_2\ldots,x_n \mid c_i)$ it assumes that all the attributes are independent given the class. Hence the formula changes into: $P(c_i) \prod_{k=1}^{n} P(x_k|c_i)$.

For the most likely class c_i, this posterior probability will be higher than posterior probability for any other classes. In summary, using Bayes's rule and the conditional independence assumption, the naïve Bayes algorithm gives the most likely classification of an instance, given its attributes.

Dumais et al. [2] compared the naïve Bayes classifier to decision trees, Bayesian networks and support vector machines. They report that, for text categorization, the classification accuracy of naïve Bayes classifier is not comparable to the other classifiers. Similar results have been shown in [1] and [20]. Despite that, naïve Bayes classifiers are commonly used for text categorization because of their speed and ease of implementation.

3.5. BAYESIAN NETWORKS

A naïve Bayes classifier assumes that the attributes are conditionally independent because this simplifies the computation. However, in many cases, including text categorization, this conditional independence assumption is not met. In contrast to the naïve Bayes classifier, Bayesian networks allow for stating conditional independence assumptions that apply to subsets of the attributes. This property makes them better text classifiers than naïve Bayes classifiers. Dumais et al. [2] showed an improvement in the classification accuracy of Bayes nets over naïve Bayes classifiers.

Bayesian networks can be viewed as directed graphs consisting of arcs and nodes. Arcs between the nodes infer that the attributes are dependent while the absence of an arc infers conditional independence. Any node X_i is assumed to be conditionally independent of its non descendants, given its immediate parents. Missing edges show conditional independence between the nodes. Each node has a conditional probability table associated with it, which specifies the probabilities of the values of its variable given its immediate parents.

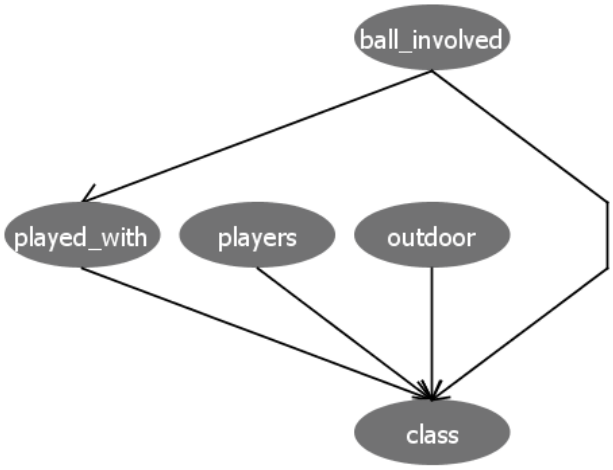

Figure 3.2: Conditional dependence/independence between the attributes of the instances in the Table 3.1.

To form Bayesian networks we used the WEKA package (described below), which contains implementations of Bayesian networks. We used the one that uses hill climbing for learning the network structure from the training data.

3.6. SUPPORT VECTOR MACHINES

The idea of support vector machines (SVM) was proposed by Vapnik [14]. It classifies a data set by constructing an N-dimensional hyperplane that separates the data into two categories.

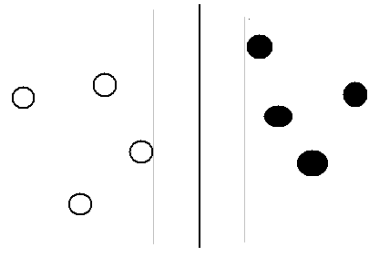

Figure 3.3: Instances in a two dimensional space separated by a line

In a simple two dimensional space, a hyperplane that separates linearly separable classes can be represented as shown in figure 3.3. In figure 3.3, black and white circles represent instances of two different classes. As shown in the figure, those instances can be properly separated by a linear separator (straight line). It is possible to find an infinite number of such lines. However, there is one linear separator that gives the greatest separation between the classes. It is called the maximum margin hyperplane and can be found using the convex hulls of the two classes. When the classes are linearly separable, the convex hulls do not overlap. The maximum margin hyperplane is the line that is farthest from both convex hulls and is orthogonal to the shortest line connecting the hulls, bisecting it. Support vectors are the instances that are closest to the maximum margin hyperplane. Figure 3.4 illustrates the maximum margin hyperplane and support vectors for the instances shown in Figure 3.3. The convex hulls have been shown as the boundaries around the two classes. The dark line that is farthest from both hulls is the maximum margin hyperplane separating the given set of instances. Support vectors are the instances that are closest to the dark line.

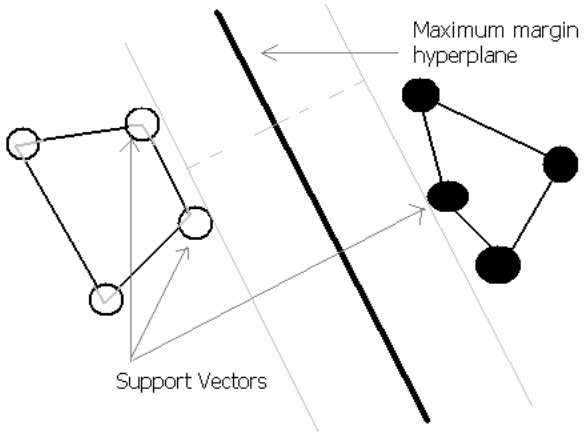

Figure 3.4: Maximum Margin Hyperplane

When there are more than two attributes, support vector machines find an N-1 dimensional hyperplane in order to optimally separate the data points represented in N dimensional space. Similarly, for finding the maximum margin hyperplane for data that are not linearly separable, they transform the input such that it becomes linearly separable. For that, support vector machines use kernel functions that transform the data to higher dimensional space where the linear separation is possible. The choice of kernel function depends upon the application.

Training a support vector machine is a quadratic optimization problem. It is possible to use any QP optimization algorithm for that purpose. We have used Platt's sequential minimal optimization algorithm [11], which is very efficient as it solves the large QP problem by breaking it down to a series of smaller QP problems [2]. Support vector machines were first used by Joachims [1] for text categorization and they have proved to be robust, eliminating the need for extensive parameter tuning. They do not need stemming of the features even when classifying highly inflectional languages [10]. Dumais et al. [2] show that support vector machines with 300 features outperform decision trees, naïve Bayes and Bayes nets in categorization accuracy. They used a simple linear version developed by Platt [11] and got better results than that of Joachims [1] on the Reuters 21578 dataset. Support vector machines are very popular algorithms for text categorization, and are termed as the best learning algorithms for this task.

CHAPTER 4

EXPERIMENTAL SETUP

This chapter describes the two document collections used in our experiments and gives the details of preprocessing techniques based on one them.

4.1. DOCUMENT COLLECTIONS

Our experiments have been carried out on the Reuters 21578 collection and the 20-Newsgroups dataset. Reuters 21578 is a collection of 21578 news articles that appeared in the Reuters newswire in 1987, and it is a standard benchmark for text categorization used by many researchers. We used 12902 articles from "ModApte split" in which 9603 documents were used as training data and the remaining 3299 as testing data. In order to compare our results with previous studies, we considered the 10 categories with the highest number of training sets as shown in Table 4.1.

Table 4.1: Data Distribution for Reuters dataset

Category	No. of training documents	No. of testing documents
Earn	2877	1087
Acq	1650	719
Money-fx	538	179
Grain	433	149
Crude	389	189
Trade	369	118
Interest	347	131
Ship	197	89
Wheat	212	71
Corn	182	56

20-Newsgroups dataset was downloaded from http://www.ai.mit.edu/~jrennie/20Newsgroups. It is a collection of newsgroup posts from mid 1990s. We used the "bydate" version of the dataset, which has duplicates removed, and the documents are sorted by date into training and testing sets. Table 4.2 shows the distribution of the documents in 20 classes.

Table 4.2: Data Distribution for 20-Newsgroups dataset

Category	No. of training documents	No. of testing documents
Alt.atheism	480	319
Comp.sys.ibm.pc.hardware	590	392
Rec.sport.baseball	597	397
Sci.med	594	396
Talk.politics.misc	465	310
Comp.graphics	584	389
Comp.os.ms-windows.misc	591	394
Comp.sys.mac.hardware	578	385
Comp.windows.x	593	395
Misc.forsale	585	390
Rec.autos	594	396
Rec.motorcycles	598	398
Rec.sport.hockey	600	399
Sci.crypt	595	396
Sci.electronics	591	393
Sci.space	593	394
Soc.religion.christian	562	398
Talk.politics.guns	546	364
Talk.politics.mideast	564	376
Talk.religion.misc	377	251

4.2. PREPROCESSING OF REUTERS 21578 COLLECTION

The Reuters-21578 dataset is originally saved in 22 files. The first 21 files contain 1000 documents and the last file contains 578 documents. All the documents are in Standard Generalized Markup Language (SGML) format. A sample of a document is shown in figure 4.1.

```
<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET"
OLDID="228" NEWID="12000">
<DATE> 1-APR-1987 11:03:23.78</DATE>
<TOPICS></TOPICS>
<PLACES><D>usa</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<UNKNOWN>

&#5;&#5;&#5;A RM
&#22;&#22;&#1;f1286&#31;reute
r f BC-BORDEN-&lt;BN>-CALLS-4-3   04-01 0067</UNKNOWN>
<TEXT>&#2;
<TITLE>BORDEN &lt;BN> CALLS 4-3/8 PCT DEBENTURES</TITLE>
<DATELINE>    NEW YORK, April 1 - </DATELINE><BODY>Borden Inc said
it called for
redemption on May one its outstanding 605,000 dlrs of 4-3/8 pct
sinking fund debentures due 1991.
    It will buy back the debentures at par plus accrued
interest of 18.23 dlrs for each 1,000 dlr debenture.
    The trustee is Chase Manhattan Bank N.A. and the paying
agency is Chemical Bank, the company said.
 Reuter
&#3;</BODY></TEXT>
</REUTERS>
```

Figure 4.1: Reuters 21578 document in SGML format.

4.3. CONVERTING SGML DOCUMENTS TO PLAIN TEXT

Besides the main text, the SGML documents contain other information like document type, title, date and place of origin, etc. embedded in the SGML tags. Not all of this information is useful for text categorization. Similarly, the tags themselves do not have any significance for text categorization, and they need to be removed from the documents so that they do not influence the process of feature selection. Hence, all the documents were processed using a java program that returned just the title and the body text of each document as shown in Figure 4.2.

YUGOSLAVIA WHEAT FREE OF WINTERKILL - USDA

There was no evidence of winterkill in Yugoslavian winter wheat during field travel along a line running northwest from Belgrade to near Maribor, the U.S. Agriculture Department's counselor in Belgrade said in a field report.The report, dated February 26, said there is evidence of delayed germination in most areas due to late seeding last fall because of dry conditions. However warm temperatures over the past three weeks have promoted some early growth and will help the crop catch up on last fall's late seeding, it said.Some Yugoslav agriculture officials are concerned about the situation because warm temperatures have brought the grain out of dormancy and taken away snow protection a little early,the report said.Cold temperatures over the next month could cause damage under these conditions,they said.The report said all wheat farmers contacted during the field trip were optimistic about the crop and the way it emerged from winter.

Figure 4.2: Reuters-21578 document in Title-Body Format.

4.4. TOKENIZATION AND STOP WORD REMOVAL

After the documents were changed into Title-Body format, they underwent *tokenization* and *stop word removal*. Words, punctuations, numbers and special characters, in the text, are all tokens. To deal with the text, we need to identify and separate all tokens, this is called tokenization. Each document was changed into a list of tokens by separating at the 'spaces' between the words. Stop words are words like 'a', 'an', 'the', 'of', 'and', etc. that occur in almost every text and also have high frequencies in the text. These words are useless in categorization because they have very low discrimination values for the categories [13]. Using a list of almost 500 words from [12], all stop words were removed from the documents. After removal of the stop words, punctuation and numbers were also removed as they too have nothing to do with the categories of the text. Figure 4.3 shows an instance of a document obtained after tokenization and stop word removal.

```
[house, farm, leader, billion, dlr, budget, cuts,
agriculture, budget, cut, additional, billion, dlrs,
chairman, key, house, agriculture, subcommittee,
implementation, program, tightening, commodity,
certificates, reconstitution, farms, possibilities,
studied, reduce, farm, spending, dan, glickman, kans,
chairman, house, agriculture, subcommittee, wheat,
soybeans, feedgrains, speaking, annual, meeting,
national, grain, feed, association, glickman, learned,
week, house, budget, committee, agriculture, committee,
reduce, fiscal, farm, budget, billion, dlrs, billion,
dlrs, level, approved, decisions, cut, farm, budget,
quickly, impact, budget, glickman, added, glickman,
committee, approve, usda, proposal, cut, target, prices,
ten, pct, administration, target, price, proposals,
dead, water, cut, budget, glickman, everthing, table,
moves, reduce, farmers, income, glickman, offered, list,
possibilities, committee, study, cut, farm, spending,
implementation, program, winter, wheat, feedgrains,
crops, introduced, glickman, result, mln, dlr, savings,
tightening, generic, pik, certificates, option,
committee, study, glickman, committing, against, action,
lawmakers, examine, recent, government, findings,
indicate, certificates, cost, cash, payments, glickman,
rules, reconstitution, farms, tightening, person,
definition, annual, payment, limitations, option, save,
mln, dlrs, increasing, acreage, set, aside,
requirements, five, pct, wheat, feedgrains, program,
sign, move, save, billion, dlrs, added, favor, change,
glickman, export, enhancement, program, eep, spending,
authority, billion, dlrs, quickly, congress, decide,
expand, program, cuts, cuts, eep, program, unlikely,
don, chopping, block, glickman]
```

Figure 4.3: Reuters-21578 document after tokenization and stop word removal

4.5. FORMATION OF TEXT REPRESENTATIONS

Each document obtained after tokenization and stop word removal was changed into two forms of text representations. In the first representation, all resulting tokens were changed into stemmed tokens using Porter's stemming algorithm. In the second representation, all tokens were replaced by hypernyms from WordNet. The hypernym-based representation had 6 different types based on the value of depth *n*. We chose the values of *n* to be 5, 6,7,8,9 and 10 representing very general to very specific hypernyms. Hence, we got seven text representations for each document.

4.6. FEATURE SELECTION

After the formation of text representations, we used TFIDF [19] for the selection of important features for categorization. For that we formed *indexing vocabularies.* For each text representation, we collected tokens from each document and stored them in a list. We then removed all redundant tokens from the list. However, we calculated the frequency of each token before removing the redundant ones. The list of tokens and their frequencies formed the indexing vocabulary. We obtained seven such vocabularies, one for each representation. The size of the indexing vocabularies for hypernym-based representation is much smaller than the normal indexing vocabulary used in the traditional bag-of-words approach. This is because many similar words are changed into a single hypernym and stored as the same concept. It also helps the reduction in size of the feature space. We calculated TFIDF for all of the tokens in the indexing vocabulary, and, then, selected the first 300 words with the largest TFIDF values as the feature set for categorization. We obtained seven such feature sets for seven text representations.

4.7. FORMATION OF NUMERICAL FEATURE VECTORS

In order to use machine learning algorithms for categorizing the documents, they need to be represented as vectors of features. For that, the tokens in the documents that were common to the tokens in the feature set were selected, and then their proportions in the document were calculated. The set of real valued numbers thus obtained formed the feature vectors for the documents. Each feature vector consisted of 301 attributes. The first 300 were real valued numbers that represented the proportion of the corresponding features in a document and the last attribute represented the category to which the document belonged. This process was carried out on all the documents seven times for seven different text representations. The results were the

numeric feature vectors in the form required by the machine learning classifiers. An example is shown in Figure 4.4.

```
0, 0.031746, 0.031746, 0, 0, 0.015873, 0.015873, 0,
0.015873, 0, 0.015873, 0, 0, 0.015873, 0, 0.015873, 0,
0, 0, 0, 0, 0, 0, 0.015873, 0, 0, 0, 0.015873, 0, 0, 0,
0.015873, 0, 0.031746, 0.015873, 0, 0, 0, 0.0793651, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.015873, 0.015873,
0.015873, 0, 0.015873, 0.015873, 0, 0, 0, 0, 0,
0.015873, 0, 0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.015873, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0.047619, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.015873, 0.015873, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0.031746, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0.015873, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.015873, 0,
0, 0, earn
```

Figure 4.4: Numerical feature vector for a document in the category ‘earn’

4.8. PREPROCESSING OF 20-NEWSGROUPS DATASET

The 20-Newsgroups dataset underwent preprocessing steps that were similar to the preprocessing steps of Reuters 21578 dataset. The documents were first changed into plain text by removing all other information except for the title and body of the text. The plain text underwent tokenization and stop word removal resulting in the raw text representation. Then the raw text was changed into hypernym-based, stemming-based and combined text representations as needed.

CHAPTER 5

EXPERIMENTS ON REUTERS 21578 DATASET

This chapter is organized as follows. First, it presents a brief description of WEKA, the package used in our experiments. It then compares the stemming-based bag-of-words model with the hypernym-based bag-of-words models, on the Reuters 21578 dataset, under four classification algorithms, all of which have been implemented in WEKA. Thereafter, it compares both models to a combined text representation formed by merging the two. Finally, it assesses the effectiveness of stemming-based and hypernym-based text representations by comparing their performances with the performance of a raw text representation. The raw text representation was formed using tokenization and stop word removal only. Neither stemming-based nor hypernym-based processing was done on it. For evaluating the performances of the learners (classification algorithms), we used precision and recall. Precision is the number of correct predictions by a learner divided by the total number of positive predictions for a category. Recall is the number of correct predictions by a learner divided by the total number of actual correct examples in the category. We have reported the F1 measure which combines precision and recall as:

$$F1\ measure = 2 * Precision * Recall / Precision + Recall$$

All the bar charts for results display standard errors of mean (SEM) along with average F1 measures. SEM is an estimated standard deviation of the error in a method calculated as:

$$SEM = s/\sqrt{n}$$

Here, s is the sample standard deviation and n is the size of the sample.

5.1. WEKA

WEKA is an acronym that stands for Waikato Environment for Knowledge Analysis. It is a collection of machine learning algorithms developed at the University of Waikato in New Zealand. WEKA is available for public use at http://www.cs.waikato.ac.nz/ml/weka/. For this research, we have used naïve Bayes, Bayesian networks, decision trees and support vector machines. The algorithms in WEKA can either be used directly or called from the user's Java code. When used directly, the users have the option of either using a command line interface or a graphical user interface (GUI). This research uses the WEKA GUI called 'Explorer'. Using Explorer, users can simply open the data files saved in 'arff' format and then choose a machine learning algorithm for performing classification/prediction on the data. Users are also provided with the facility of either supplying a separate *test set* or using *cross validation* on the current data set. The WEKA Explorer allows the users to change the parameters of the machine learning algorithms easily. For example, while using a multilayer perceptron, users can select their own values of learning rate, momentum, number of epochs etc. One of the greatest advantages of using the GUI is that it provides visualization features which allow users to view their results in various manners.

5.2. COMPARISON OF STEMMING-BASED AND HYPERNYM-BASED MODELS.

Table 5.1 summarizes the average F1 measures for all four learners for the ten most frequent Reuters 21578 categories using stemming-based and hypernym-based text representations. Stemming-based representation clearly outperformed the hypernym-based representations for all learners, for all six values of hypernym depth (n). The Bayesian network, using stemming-based representation, turned out to be the winner among the four classifiers. Support vector machines

came very close to the Bayesian networks. In terms of the percentage of correctly classified instances, support vector machines using stemming-based representation outperformed all others as shown in Table 5.2.

Table 5.1: Average F1 measures over 10 frequent Reuters 21578 categories for stemming-based and hypernym-based representations

Classification Algorithms	**Stemming based representation**	**Hypernym based representation**					
		n=5	n=6	n=7	n=8	n=9	n=10
Decision Trees	**0.6155**	0.5354	0.5067	0.5399	0.5375	0.5574	0.5526
Naïve Bayes	**0.5908**	0.5408	0.5614	0.5752	0.5765	0.5793	0.59
Bayes nets	**0.6516**	0.5938	0.6127	0.6284	0.6235	0.6429	0.6432
SVMs	**0.6305**	0.6001	0.6117	0.6263	0.6217	0.624	0.6299

Table 5.2: Percentage of correctly classified instances

Classification Algorithms	**Stemming based representation**	**Hypernym based representation**					
		n=5	n=6	n=7	n=8	n=9	n=10
Decision Trees	83.11	79.83	78.60	80.05	80.33	81.74	**81.31**
Naïve Bayes	79.94	75.50	77.63	79.18	79.25	79.25	**80.01**
Bayes nets	82.90	79.22	80.59	81.67	81.27	82.82	**83.08**
SVMs	**87.37**	84.81	85.96	86.54	86.32	86.72	87.01

Table 5.2 also supports the claims of Dumais et al. [2] that Bayesian networks showed improvements over naïve Bayes and that support vector machines are the most accurate methods for categorizing the Reuters 21578 dataset. However, performance of classification algorithms is not the main concern of this research. The main point is the comparison of the relevance of

lexical (stemming) and ontological (hypernym) information on text categorization. Based on average F1 measure (Table 5.1) and classification accuracy (Table 5.2), we can say that stemming-based feature representation is better than hypernym-based feature representation for categorizing the Reuters 21578 dataset. As shown in Figure 5.1, stemming-based representation performed better than the best performing hypernym-based representations for all four learners.

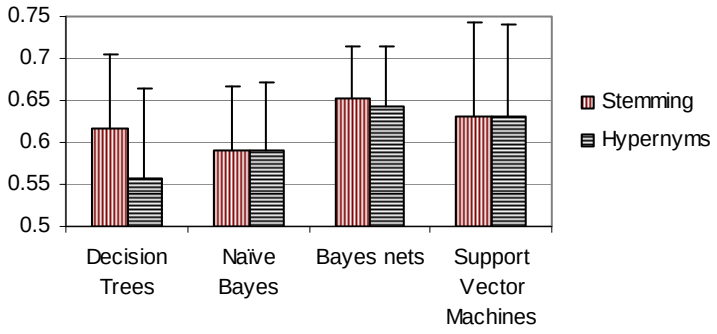

Figure 5.1: Comparison of stemming-based representation with best performing hypernym-based representation, for all four learners, in terms of average F1 measures and standard errors.

5.3. COMPARISON WITH COMBINED TEXT REPRESENTATION

More experiments were done in order to find out whether combining stemming-based and hypernym-based representations would improve the classification accuracy. For that we experimented with the hypernyms at n= 5, 7 and 10. As n=5 represents the hypernyms with general concept, 7 intermediate and 10 specific, we believed those three values to be good representatives of the hypernym space. For combination, the tokens were first stemmed and then changed into the hypernyms. Table 5.3 summarizes the average F1 measures for all four learners for the ten most frequent Reuters 21578 categories using the combined text representations.

Table 5.3: Average F1 measures over 10 frequent Reuters 21578 categories for combined text representations

Classification Algorithms	**Average F1 measures for Combined representations**		
	n=5	n=7	n=10
Decision Trees	0.511	0.5497	**0.5752**
Naïve Bayes	0.5525	0.5767	**0.5819**
Bayes nets	0.5921	0.6235	**0.6405**
Support Vector Machines	0.6085	0.6284	**0.6334**

The results did not yield improved performance over stemming-based representation. As shown in figure 5.2, for all four learners, the best results for the combined representations were not at par with the results for stemming-based representation. The combined method worked better than the hypernym-based method for decision trees but degraded the performances for naïve Bayes and Bayesian nets. Support vector machines were found to be robust to the change in the text representations. As seen in Figure 5.2, their results were consistent for hypernym-based representation, stemming-based representation and combined representation.

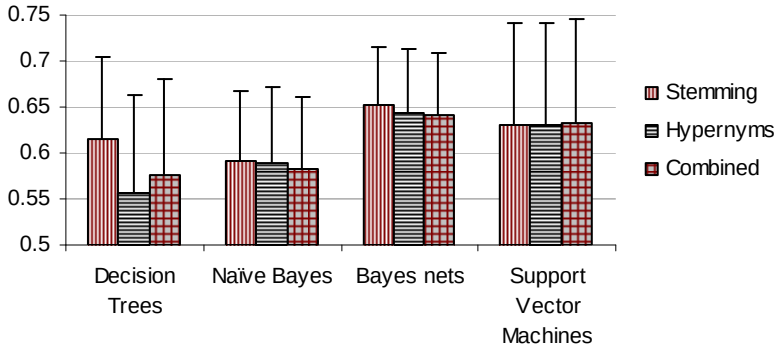

Figure 5.2: Comparison of the average F1 measures and standard errors of stemming-based representation with best performing hypernym-based and combined representations

5.4. COMPARISON WITH RAW TEXT REPRESENTATION

A set of experiments were carried out to compare the performances of stemming-based representation and hypernym-based representations with a raw text representation. The raw text representation was formed by applying tokenization and stop word removal on Reuters 21578 documents. Neither stemming-based nor hypernym-based processing was applied to the resulting documents. In order to assess the effects of stemmed tokens and hypernyms on the classification accuracy, we compared the average F1 measures of stemming-based representation and hypernym-based representations with the F1 measures of raw text representation. As stemming is based on lexical analysis and as hypernyms represent ontological information, these comparisons evaluate the effects of inducing lexical information and ontological information on text representation. Table 5.4 summarizes the average F1 measures for all four learners for the ten most frequent Reuters 21578 categories using raw text representation. Figure 5.3 compares the results shown in table 5.4 with the results for the stemming-based model and the best results for hypernym-based model.

Table 5.4: Average F1 measures over 10 frequent Reuters 21578 categories for raw text representations

Classifiers	Decision trees	Naïve Bayes	Bayes nets	Support vector machines
Average F1 measures	0.5973	0.5929	**0.6496**	0.6261

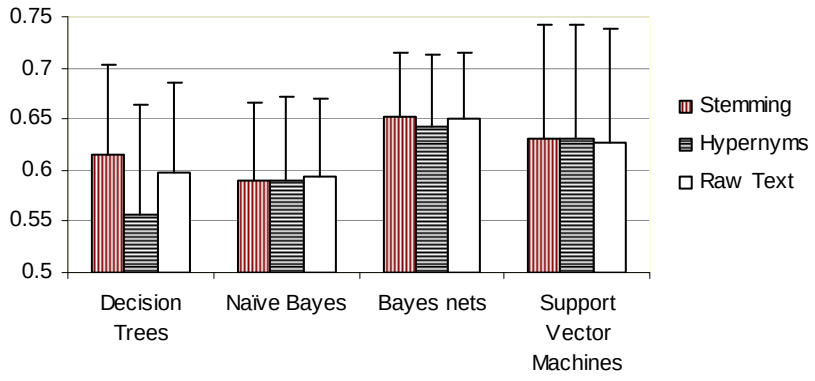

Figure 5.3: Comparison of the average F1 measures and standard errors of stemming-based representation with best performing hypernym-based and raw text representation.

As seen in the figure, decision trees, Bayesian networks and support vector machines produced better results with stemming-based representation than raw text representation. This result was significant in decision trees than the rest of the classifiers. However, for the naïve Bayes classifier, the raw text representation proved to be the best. The results were consistent with our previous experiments in which stemming-based representation performed better than hypernym-based representations and combined text representations for all classifiers. The hypernym-based approach could not yield any improvements over the raw text representation for decision trees, naïve Bayes and Bayesian networks. In fact, it degraded their performances. It produced a slight improvement over the performance of support vector machines but that improvement was not significant as support vector machines proved to be very robust to the change in the text representations.

CHAPTER 6

EXPERIMENTS ON THE 20-NEWSGROUPS DATASET

The following experiments were done to validate the conclusions derived from the experiments on the Reuters 21578 dataset. These experiments were performed on a subset of the 20-Newsgroups dataset. Five classes, out of 20, were selected as shown in Table 6.1.

Table 6.1: Data Distribution for 20-Newsgroups data subset

Category	No. of training documents	No. of testing documents
Alt.atheism	480	319
Comp.sys.ibm.pc.hardware	590	392
Rec.sport.baseball	597	397
Sci.med	594	396
Talk.politics.misc	465	310

Reuters 21578 dataset has classes like corn, grain and wheat with highly overlapping features. There is a fair chance that these common features are ontologically mapped to the same hypernyms. Suspecting that this might be the cause for the poor performance of hypernym-based representation; the five classes from the 20-Newsgroups dataset were intentionally selected to be diverse so that there would be less overlap between their features. This design can help in testing whether hypernyms produce better categorization accuracy when the classes have relatively lower feature overlapping.

6.1. COMPARISON OF VARIOUS REPRESENTATIONS

In the Reuters 21578 dataset, stemming-based representation had performed better than all six hypernym-based representations for all classifiers. While in this subset, the hypernym-based representation with n=10 outperformed stemming-based representations for Bayesian networks and decision trees. Table 6.2 summarizes the average F1 measures for all four learners for the five 20-Newsgroups categories using stemming-based, hypernym-based, and raw text representations.

Table 6.2: Average F1 measures over the subset of 20-Newsgroups dataset for stemming-based, hypernym-based, and raw text representations.

Classifiers	**Stemming based representation**	**Hypernym based representation**			**Raw data**
		n=5	n=7	n=10	
Decision Trees	0.7594	0.6394	0.7322	**0.778**	0.7494
Naïve Bayes	**0.763**	0.6904	0.7254	0.7572	0.7614
Bayesian Nets	0.7994	0.7082	0.7614	**0.8192**	0.81
Support vector machines	**0.8318**	0.748	0.7854	0.8252	0.7978

One of the reasons hypernym-based representation performed well could be the size of the dataset (number of classes involved). The size is much smaller compared to the Reuters dataset and hypernyms have been shown to perform better in smaller datasets by Scott and Matwin [5]. Also, the five classes used in the experiments have been deliberately chosen such that there is less overlap between the features of the classes. As mentioned earlier, this choice was intentional and was made in order to test whether the hypernyms could yield better categorization accuracy for a dataset with fewer overlapping features. The results have shown that the hypernym-based representations are capable of performing as well as stemming-based

representations, and even better, for such datasets. This performance of hypernyms is evident in Figure 6.1.

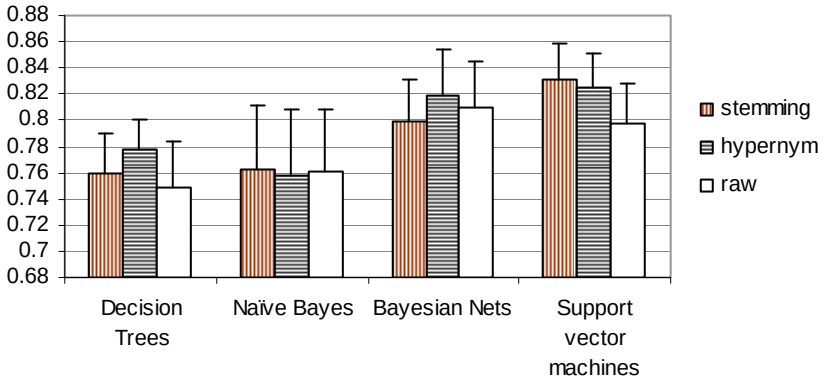

Figure 6.1: Comparison of the average F1 measures and standard errors of stemming-based representation with best performing hypernym-based representation and raw text representation

Figure 6.1 also compares the F1 measures of the best performing hypernym-based representation with the raw text representation. The best performing hypernym-based representation produced better categorization accuracy than the raw text representation for decision trees, Bayesian networks and support vector machines validating that hypernyms are indeed capable of improving the categorization accuracy if the dataset is small and there is less overlapping between the features of the classes.

Despite the good performance of hypernyms, support vector machines using stemming-based representation turned out to be the best classifier for this dataset. As Bayesian networks using stemming-based representation was the best classifier of the Reuters dataset, this leads to the conclusion that stemming-based representation with an appropriate classifier is capable of outperforming all hypernym-based representations. For decision trees, naïve Bayes classifiers

and support vector machines, stemmed features resulted in better categorization than the raw text.

6.2. EFFECTS OF COMBINED TEXT REPRESENTATIONS

For combination, for all values of n, the tokens were first stemmed and then changed into hypernyms. Table 6.3 summarizes the average F1 measures for all four learners for the 20-Newsgroups subset using the combined text representations.

Table 6.3: Average F1 measures over five 20-Newsgroups categories for combined text representations

Classifiers	**Combined representations**		
	n=5	n=7	n=10
Decision Trees	0.7252	0.7514	**0.7556**
Naïve Bayes	0.727	0.7384	**0.7674**
Bayesian Nets	0.7354	0.769	**0.802**
Support vector machines	0.7818	0.8098	**0.8346**

For the 20-Newsgroups data subset, the combined representations displayed very interesting results. Using naïve Bayes classifiers and support vector machines, for which stemming-based representation had worked better than hypernym-based representations, the combined representations produced results that were better than the individual methods. For the other two classifiers, the results for combined representations were not as good as the best results for hypernym-based representation alone. This signifies that hypernyms add to the categorization accuracy produced by stemmed features but the opposite is not true. This comparison has been shown in Figure 6.2.

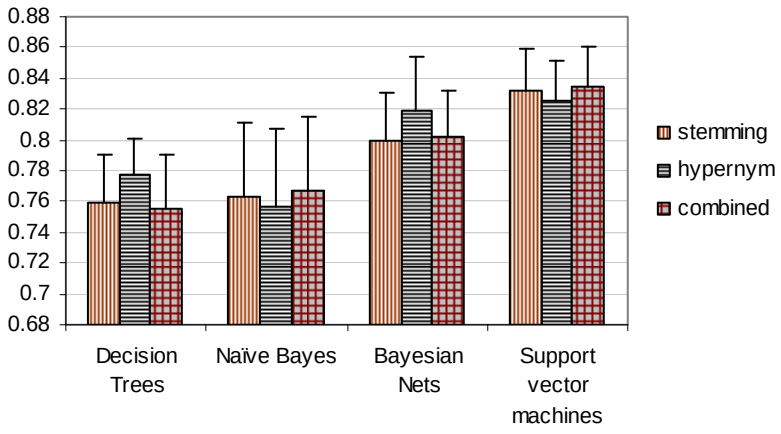

Figure 6.2: Comparison of the average F1 measures and standard errors of stemming-based representation with best performing hypernym-based and combined representations

6.3. EXPERIMENTS WITH ALL 20 CLASSES

While categorizing 20 classes of the 20-Newsgroups dataset, Stemming-based representation performed better than hypernym-based representation. Figure 6.3 compares the average F1 measures of Stemming-based representation with F1 measures of hypernym-based representation (n=10), over the 20 classes of 20-Newsgroup dataset, for all four learners.

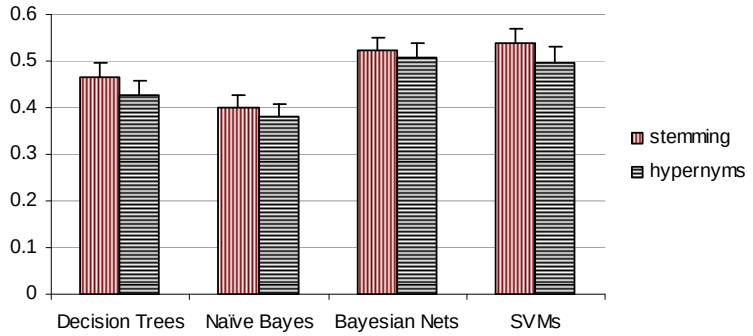

Figure 6.3: Comparison of the average F1 measures and standard errors of stemming-based representation with best performing hypernym-based representation over twenty 20-Newsgroup categories

As seen in the figure, all the classifiers yielded better accuracy with stemming-based representation. This result is anomalous to the results for the subset of 20-Newsgroups dataset in which hypernym-based representation (n=10) performed better than stemming-based representation for two classifiers. However, it is consistent with the results for Reuters 21578 dataset in which stemming-based representation clearly outperformed all hypernym-based representations for all classifiers.

CHAPTER 7

DISCUSSIONS AND CONCLUSIONS

The performance of stemming-based representation was found to be better than hypernym-based representations and combined representations for all four classifiers on the Reuters 21578 dataset. The stemming-based representation produced improved classification accuracy over the raw text representation for decision trees, Bayesian networks and support vector machines. This improvement was more significant in decision trees than other classifiers. The performance of the stemming-based approach signifies that induction of lexical information can be useful for categorizing Reuters 21578 dataset. Support vector machines were found to be robust to the change in text representation. They performed similarly for stemming-based, hypernym-based, combined and raw text representation. Such consistent performance of support vector machines reinstates the claim that they are robust and do not need stemming for producing good results [10]. Bayesian networks using stemmed features proved to be the best classifier and naïve Bayes classifiers performed the best with raw text representation.

The experiments also indicated that hypernyms do not add to the classification accuracy for the Reuters 21578 dataset. Such bad performance of hypernyms is consistent with the results shown by Scott and Matwin [4]. However, they had suspected that the use of real valued density measurements and more general hypernyms might improve the results. We used real valued density measurements and did not achieve significant improvements in the classification accuracy. Similarly, using more general hypernyms did not help either. In fact, as the features

became more general, they failed to distinguish between overlapping classes and produced worse results. Figure 7.1 shows the average F1 measures of all four classifiers at different values of n.

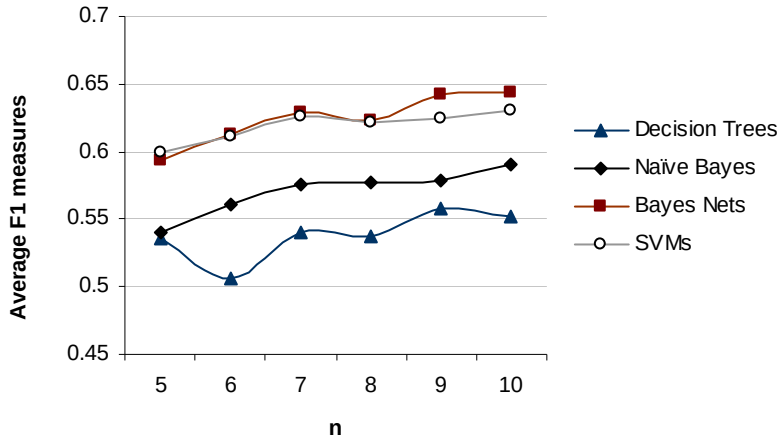

Figure 7.1: Average F1 measures over 10 frequent Reuters categories at different values of n.

As seen in Figure 7.1, the F1 measures are roughly proportional to the values of n. As the values of n increase, F1 measures also increase. Higher values of n produce hypernyms that represent more specific concepts. Thus as a concept becomes less general, categorization accuracy increases. This means that ontologically mapping the words to general concepts does not seem helpful for the task of categorizing the Reuters 21578 dataset. Hence, based on the experiments, for the Reuters 21578 dataset, incorporating lexical information seems to be more effective than incorporating ontological information.

On the other hand, for the 20-Newsgroups subset, though support vector machines using stemmed features proved to be the best individual classifier, hypernym-based text representations performed as well as stemming-based text representations. This indicated that when the dataset is

small (fewer classes) and when classes overlap less, hypernym-based text representations are capable of producing results that are comparable to the results of stemming-based representation. Like stemmed features, hypernyms were useful in improving the categorization accuracy over the raw data, for the 20-Newsgroups subset. The improvement was achieved at n=10. This signifies that ontologically mapping the words to relatively specific concepts yields better results than mapping the words to more general concepts. The improvements in the results from general to specific concepts are shown in Figure 7.2. The figure shows that for all classifiers, F1 measures increase when the values of n increase. This means that the categorization accuracy improves when the words are mapped to more specific concepts. This result is consistent with the results obtained for Reuters 21578 dataset.

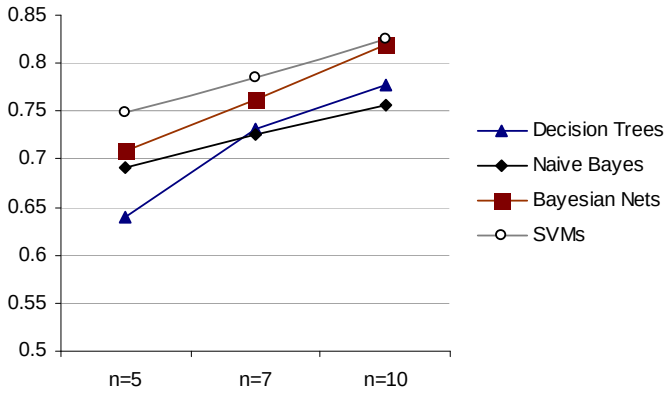

Figure 7.2: Average F1 measures over five 20-newsgroup categories at different values of n.

The experiments using combined text representations for the 20-newsgroup subset indicate that hypernyms improve the categorization accuracy produced by stemmed features but the opposite is not always true. In fact, support vector machines using the combined text

representation produced better results than the best individual classifier - SVM using stemmed features alone.

In summary, for the larger dataset (Reuters 21578) with significant feature overlap of classes, inducing lexical information was found to be more effective than inducing ontological information, irrespective of the type of the classifier used. This finding was strengthened by the superior performance of the stemming-based representation compared to the hypernym-based representation while categorizing all 20 classes of the 20-Newsgroups dataset. Using smaller dataset (20-Newsgroups subset) with fewer overlapping features, both hypernyms and stemmed features were found to be capable of improving the categorization accuracy over the raw text representation. However, this improvement was dependent upon the classifier being used. Hypernyms were able to improve the performance of decision trees, Bayesian networks and support vector machines. Stemming, on the other hand, was able to improve the performance of decision trees, naïve Bayes classifiers and support vector machines. Hypernyms performed better at higher values of n, for both datasets, indicating that ontologically mapping the words to very general concepts may not be helpful for the task of text categorization.

REFERENCES

[1] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," *in Proc. 10th European Conf. Machine Learning*, 1998, pp. 137–142.

[2] S. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive learning algorithms and representations for text categorization," in *Proc. 7th International Conf. Information and Knowledge Management*, ACM Press, 1998, pp. 148–155.

[3] P. Rosso, E. Ferretti, D. Jiminez and V. Vidal, "Text categorization and information retrieval using WordNet senses," in *Proc. 2nd Global WordNet Conf*., Brno, Czech Republic, 2004, pp. 299-304.

[4] S. Scott and S. Matwin, "Feature engineering for text classification," in *Proc. 16th International Conf. Machine Learning,* 1999, pp. 379-388.

[5] S. Scott and S. Matwin, "Text classification using WordNet hypernyms," in *Proc. COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, Montreal, 1998, pp. 45-51.

[6] W.W. Cohen, "Fast Effective Rule Induction," in *Proc. 12th International Conf. Machine Learning,* 1995, pp. 115-123.

[7] C. Apte, F. Damerau and S.M. Weiss, "Text Mining with decision rules and decision trees," in *Proc. Conf. Automated Learning and Discovery*, Carnegie Mellon Univ., June.1998.

[8] D.D. Lewis and M. Ringuette, "A comparison of two learning algorithms for text categorization," in *Proc. 3rd Annual Symposium. Document Analysis and Information Retrieval*, Las Vegas, US, 1994, pp. 81-93.

[9] G.A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K.J. Miller, "Introduction to WordNet: An On-Line Lexical Database," *Int'l J. Lexicography,* vol. 3, no. 4, 1990, pp. 235-244.

[10] E. Leopold and J. Kindermann, "Text categorization with support vector machines. How to represent texts in input space?," *Machine Learning*, vol. 46, no. 1-3, Jan. 2002, pp. 423- 444.

[11] J. Platt, "Fast training of SVMs using Sequential Minimal Optimization," in *Advances in Kernel Methods-Support Vector Learning.* Cambridge: MIT press, 1999, pp. 185-208.

[12] L.L Yin and D. Savio, "Learned Text Categorization by Backpropagation Neural Network," Masters Thesis, Dept. Comp. Science, the Hong Kong University of Science and Technology, Hong Kong, 1996.

[13] G. Salton and M.J McGill, *Introduction to Modern Information Retrieval*. New York: McGraw-Hill, 1983.

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

[15] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, July 1980, pp. 130–137.

[16] The Reuters-21578 collection is available at: http://www.daviddlewis.com/resources/testcollections/reuters21578

[17] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, San Francisco: Morgan Kaufmann, 2005.

[18] The prolog version of Porter's stemming algorithm is available at: http://tartarus.org/~martin/PorterStemmer/prolog.txt

[19] G. Salton, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Boston: Addison-Wesley, 1989.

[20] Y.Yang and X.Liu, "A re-examination of text categorization methods," *in Proc. 22 Annual International ACM SIGIR Conf. Research and Development in Information Retrieval*, 1999, pp.42-49.

[21] T. Mitchell, *Machine learning*. New York: McGraw-Hill, 1997.