FEATURE GENERATION FOR TEXTUAL INFORMATION RETRIEVAL USING WORLD KNOWLEDGE

EVGENIY GABRILOVICH

FEATURE GENERATION FOR TEXTUAL INFORMATION RETRIEVAL USING WORLD KNOWLEDGE

RESEARCH THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

EVGENIY GABRILOVICH

SUBMITTED TO THE SENATE OF THE TECHNION — ISRAEL INSTITUTE OF TECHNOLOGY
KISLEV, 5767 HAIFA DECEMBER, 2006

THIS RESEARCH THESIS WAS DONE UNDER THE SUPERVISION OF DR. SHAUL MARKOVITCH IN THE DEPARTMENT OF COMPUTER SCIENCE

ACKNOWLEDGMENTS

I am indebted to my advisor, Shaul Markovitch, for the guidance he afforded me during my studies. Shaul gave me plenty of freedom to explore the directions I was most interested in. At the same time, however, he constantly reminded me of the importance of focusing on the essential things and foregoing lower-hanging fruit that carry little benefit in the larger perspective. Shaul's comments on paper drafts were always to the point, and invariably led to much more coherent and simply better papers. From our countless discussions, I adopted a useful habit of always thinking about evaluation before launching a new study. Shaul also provided me with support at the times when my spirit was low, always insisting that there is light at the end of the tunnel.

I am thankful to Susan Dumais and Eric Horvitz for hosting me as a summer intern at Microsoft Research. I also thank Arkady Borkovsky and Hadar Shemtov for arranging my summer internship at Yahoo.

I am grateful to Alex Gontmakher, my longtime friend and office roommate. Alex offered his help on things too many to mention, from program optimization to idiosyncrasies of Linux. Alex was also solely responsible for persuading me to study Perl, which then quickly became my scripting language of choice, and for spending hours in teaching me new Perl idioms.

I am thankful to Erez Halahmi, my former manager at Comverse Technology, for teaching me how to manage large software projects. Without his couching, I would hardly be able to write tens of thousands of lines of code during my Ph.D. studies, and would likely still be engrossed in debugging.

Throughout my doctoral studies, I benefited from numerous discussions with my friends and colleagues, who shared their wisdom, offered views from the perspective of their fields of expertise, and just helped me when I needed some advice. I am grateful to all of them for their time and assistance: Gregory Begelman, Beata Beigman-Klebanov, Ron Bekkerman, Ido Dagan, Dmitry Davidov, Gideon Dror, Ofer Egozi, Saher Esmeir, Ran El-Yaniv, Anna Feldman, Ariel Felner, Lev Finkelstein, Maayan Geffet, Oren Glickman, Yaniv Hamo, Alon Itai, Shyam Kapur, Richard Kaspersky, Ronny Lempel, Dmitry Leshchiner, Irit Opher, Dan Roth, Eytan Ruppin, Dmitry Rusakov, Fabrizio Sebastiani, Vitaly Skachek, Frank Smadja, Gennady Sterlin, Vitaly Surazhsky, Stan Szpakowicz, Peter Turney, and Shuly Wintner.

I am especially thankful to my wife Lena for her unconditional love, for putting up with the countless hours I spent on my thesis work, and for being there when I needed it.

Finally, I am deeply beholden to my parents, Margarita Sterlina and Solomon Gabrilovich, for their immeasurable love and support. I am also thankful to my parents for teaching me the value of knowledge and education. No words in any natural language would be sufficient to thank my parents for all they have done for me. This thesis is dedicated to them.

THE GENEROUS FINANCIAL HELP OF THE TECHNION IS
GRATEFULLY ACKNOWLEDGED

To my parents,
Margarita Efimovna Sterlina and Solomon Isaakovich Gabrilovich

An investment in knowledge always pays the best interest.

– Benjamin Franklin

Contents

Abstract **1**

1 Introduction **5**

1.1 Proposed Solution . . . 7

1.2 Contributions of This Thesis . . . 10

1.3 Thesis Outline . . . 11

2 Background **13**

2.1 Text Categorization . . . 13

2.1.1 Document Features . . . 14

2.1.2 Feature Selection . . . 16

2.1.3 Feature Valuation . . . 19

2.1.4 Metrics . . . 20

2.1.5 Notes on Evaluation . . . 22

2.2 Problems With the Bag of Words Approach . . . 22

2.3 Feature Generation . . . 25

3 Feature Generation Methodology **29**

3.1 Overview . . . 29

3.2 Requirements on Suitable Knowledge Repositories . . . 31

3.3 Building a Feature Generator . . . 33

3.3.1 Attribute Selection . . . 34

3.3.2 Feature Generation per se . . . 34

3.4 Contextual Feature Generation . . . 35

3.4.1 Analyzing Local Contexts . . . 36

3.4.2 Feature Selection . . . 37

3.4.3 Feature Valuation . . . 38

3.4.4 Revisiting the Running Example . . . 38

3.5 Using Hierarchically-Structured Knowledge Repositories . . . 39

3.6 Using Knowledge Repositories that Define Arbitrary Relations Between Concepts . . . 42

4 Instantiation of the Feature Generation Methodology for the ODP and Wikipedia 45

4.1 Using the Open Directory for Feature Generation 45

4.1.1 Multiplying Knowledge Through Web Crawling 46

4.1.2 Noise Reduction and Attribute Selection 46

4.1.3 Implementation Details 50

4.2 Using Wikipedia for Feature Generation 51

4.2.1 Wikipedia as a Knowledge Repository 53

4.2.2 Feature Generator Design 54

4.2.3 Using the Link Structure 55

4.2.4 Implementation Details 57

5 Empirical Evaluation of Feature Generation for Text Categorization 61

5.1 Test Collections 61

5.1.1 Reuters-21578 61

5.1.2 20 Newsgroups (20NG) 62

5.1.3 Movie Reviews 62

5.1.4 Reuters Corpus Version 1 (RCV1) 63

5.1.5 OHSUMED 64

5.1.6 Short Documents 65

5.1.7 Automatic Acquisition of Data Sets 67

5.2 Experimentation Procedure 68

5.2.1 Text Categorization Infrastructure 69

5.2.2 Baseline Performance of $\mathcal{H}$OGWARTS 70

5.2.3 Using the Feature Generator 70

5.3 ODP-based Feature Generation 70

5.3.1 Qualitative Analysis of Feature Generation 70

5.3.2 The Effect of Feature Generation 76

5.3.3 The Effect of Contextual Analysis 77

5.3.4 The Effect of Knowledge Breadth 77

5.3.5 The Utility of Feature Selection 79

5.3.6 The Effect of Category Size 81

5.3.7 The Effect of Feature Generation for Classifying Short Documents 82

5.3.8 Processing Time 83

5.4 Wikipedia-based Feature Generation 85

5.4.1 Qualitative Analysis of Feature Generation 85

5.4.2 The Effect of Feature Generation 97

5.4.3 The Effect of Knowledge Breadth 99

5.4.4 Classifying Short Documents 99

5.4.5 Using Inter-article links as Concept Relations 100

6 Using Feature Generation for Computing Semantic Relatedness of Texts 103

6.1 Explicit Semantic Analysis 103

6.2 Empirical Evaluation of Explicit Semantic Analysis 105

6.2.1 Test Collections 106

6.2.2 The Effect of External Knowledge 107

7 Related work 109

7.1 Beyond the Bag of Words 109

7.2 Feature Generation for Text Categorization 109

7.2.1 Feature Generation Using Electronic Dictionaries 111

7.2.2 Comparing Knowledge Sources for Feature Generation: ODP versus WordNet 113

7.2.3 Using Unlabeled Examples 116

7.2.4 Other Related Studies 117

7.3 Semantic Similarity and Semantic Relatedness 118

8 Conclusions 123

A Text Categorization with Many Redundant Features: Using Aggressive Feature Selection to Make SVMs Competitive with C4.5 127

A.1 Introduction 127

A.2 Experimental Methodology 129

A.2.1 Datasets 130

A.2.2 Predicting the Utility of Feature Selection 131

A.2.3 Extended Feature Set Based on WordNet 132

A.2.4 Feature Selection Algorithms 133

A.2.5 Classification Algorithms and Measures 133

A.3 Empirical Evaluation 134

A.3.1 Validation of $\mathcal{H}$OGWARTS Performance 134

A.3.2 Predicting the Utility of Feature Selection with Outlier Count 136

A.3.3 Comparison of Classifiers 136

A.3.4 The Effect of Using Different Feature Sets 138

A.3.5 The Effect of Using Different FS Algorithms 138

A.3.6 Testing the Relevancy of Features 138

A.4 Discussion 139

B Parameterized Generation of Labeled Datasets for Text Categorization Based on a Hierarchical Directory 143
B.1 Introduction . . . 143
B.2 Parameterization of Dataset Generation . . . 146
B.2.1 Metrics . . . 147
B.2.2 Properties of Individual Categories . . . 150
B.3 Methodology for Automatic Dataset Generation . . . 151
B.3.1 Acquisition of the Raw Data . . . 151
B.3.2 Filtering the Raw Data to Cope with Noise . . . 153
B.4 Empirical Evaluation . . . 154
B.4.1 Data Acquisition . . . 154
B.4.2 Text Categorization Infrastructure . . . 154
B.4.3 Correlation Between Distance Metrics and Text Categorization Accuracy . . . 155
B.4.4 Correlation Between Distance Metrics and MAA . . . 156
B.4.5 Versatility of Dataset Generation . . . 157
B.5 Conclusions and Future Work . . . 159
References 163
Hebrew Abstract i

List of Figures

3.1 Standard approach to text categorization 30
3.2 Induction of text classifiers using the proposed framework for feature generation 30
3.3 Building a feature generator 35
3.4 Feature generation 35
3.5 Performing feature generation using the multi-resolution approach 36
3.6 Feature generation example 39
3.7 Building a feature generator using hierarchically-structured knowledge repositories 41
3.8 Feature generation using a hierarchical ontology 42
3.9 Feature generation with arbitrary relations 43

4.1 Feature generation with Wikipedia links as relations 57
4.2 Feature generation with Wikipedia links as relations, where only more general features are constructed 58

5.1 Varying context length (Movies) 79
5.2 Feature selection (Movies) 81
5.3 Feature selection (RCV1/Topic-16) 81
5.4 RCV1 (Industry): Average improvement versus category size . . . 82
5.5 RCV1 (Topic): Average improvement versus category size 83

6.1 Knowledge-based semantic interpreter 106

A.1 Distribution of features by IG in several datasets 132
A.2 $\mathcal{H}$OGWARTS performance on existing datasets 135
A.3 Improvement in SVM accuracy at different FS levels vs. using 100% features 135
A.4 Comparison of performance of SVM, C4.5 and KNN with 100% features 137
A.5 Classification using a bag of words 139
A.6 Classification using an extended feature set 139
A.7 SVM accuracy vs. FS level 140

A.8 C4.5 accuracy vs. FS level 140
A.9 Removing the best features by IG 141

B.1 Locating categories at requested text distance 152
B.2 SVM accuracy vs. graph distance 156
B.3 SVM accuracy vs. text distance 156
B.4 Text distance vs. graph distance 157
B.5 MAA vs. graph distance 158
B.6 MAA vs. text distance 158
B.7 Distribution of graph distances in ODP 159
B.8 Distribution of text distances in ODP (sample) 159

List of Tables

4.1 Examples of attribute selection using information gain 49

5.1 Definition of RCV1 category sets used in the experiments 65
5.2 Definition of OHSUMED category sets used in the experiments . . 66
5.3 Baseline performance of $\mathcal{H}$OGWARTS text categorization platform 71
5.4 Top ten ODP concepts generated for a sample sentence 76
5.5 Top ten ODP concepts generated for a sample sentence 77
5.6 Text categorization with and without feature generation 78
5.7 Text categorization with and without feature generation, when only a subset of the ODP is used 80
5.8 Text categorization of short documents 84
5.9 Test collection sizes 85
5.10 The effect of feature generation 98
5.11 Comparison of two Wikipedia snapshots 99
5.12 The effect of feature generation using a newer Wikipedia snapshot 100
5.13 Feature generation for short documents 101
5.14 Feature generation for short documents using inter-article links . . 102

6.1 Correlation of word relatedness scores with human judgements . . 108
6.2 Correlation of text relatedness scores with human judgements . . 108

A.1 Classifier accuracy at different FS levels 137
A.2 Statistical significance of differences in classifier accuracy 137

Abstract

Imagine an automatic news filtering system that tracks company news. Given the news item “FDA approves ciprofloxacin for victims of anthrax inhalation”, how can the system know that the drug mentioned is an antibiotic produced by Bayer? Or consider an information professional searching for data on RFID technology—how can a computer understand that the item “Wal-Mart supply chain goes real time” is relevant for the search? Algorithms we present can do just that.

When humans approach text processing tasks, such as text categorization, they interpret documents in the context of their background knowledge and experience. On the other hand, conventional information retrieval systems represent documents as bags of words, and are restricted to learning from individual word occurrences in the (necessarily limited) training set. We propose to enrich document representation through automatic use of vast repositories of human knowledge. To this end, we use Wikipedia and the Open Directory Project, the largest encyclopedia and Web directory, respectively. Wikipedia articles and ODP categories represent knowledge concepts. In the preprocessing phase, a feature generator analyzes the input documents and maps them onto relevant concepts. The latter give rise to a set of generated features that either augment or replace the standard bag of words. Feature generation is accomplished through contextual analysis of document text, thus implicitly performing word sense disambiguation. Coupled with the ability to generalize from words to concepts, this approach addresses the two main problems of natural language processing—synonymy and polysemy.

Categorizing documents with the aid of knowledge-based features leverages information that cannot be deduced from the training documents alone. Empirical results confirm that this knowledge-intensive representation brings text categorization to a qualitatively new level of performance across a diverse collection of datasets.

We also adapt our feature generation methodology for another task in natural language processing, namely, automatic assessment of semantic relatedness of words and texts. Previous state of the art results are based on Latent Semantic Analysis, which represents documents in the space of “latent concepts”

computed using Singular Value Decomposition. We propose Explicit Semantic Analysis, which uses the feature generator methodology to represent the meaning of text fragments in a high-dimensional space of features based on natural concepts identified and described by humans. Computing semantic relatedness in this space yields substantial improvements, as judged by the very high correlation of computed scores with human judgments.

List of Abbreviations

20NG	20 Newsgroups
BOW	Bag of words
DF	Document Frequency
ESA	Explicit Semantic Analysis
FG	Feature Generation
FS	Feature Selection
FV	Feature Valuation
HTML	Hyper Text Markup Language
IDF	Inverse Document Frequency
IG	Information Gain
IR	Information Retrieval
KNN	K Nearest Neighbors
LSA	Latent Semantic Analysis
LSI	Latent Semantic Indexing
MAA	Maximum Achievable Accuracy
ML	Machine Learning
ODP	Open Directory Project
RCV1	Reuters Corpus Volume 1
RCV1-v2	Reuters Corpus Volume 1 (version 2)
SGML	Standard Generalized Markup Language
SVM	Support Vector Machine
TC	Text Categorization
TF	Term Frequency
TFIDF	Term Frequency/Inverse Document Frequency
TREC	Text REtrieval Conference
URL	Uniform Resource Locator
WWW	World Wide Web
WN	WordNet
XML	eXtensible Markup Language

Chapter 1

Introduction

Recent proliferation of the World Wide Web, and common availability of inexpensive storage media to accumulate over time enormous amounts of digital data, have contributed to the importance of intelligent access to this data. It is the sheer amount of data available that emphasizes the *intelligent* aspect of access—no one is willing to or capable of browsing through but a very small subset of the data collection, carefully selected to satisfy one's precise information need.

The branch of Computer Science that deals with facilitating access to large collections of data is called Information Retrieval (IR). The field of Information Retrieval[1] spans a number of sub-areas, including information retrieval *per se*, as performed by users of Internet search engines or digital libraries; text categorization, which labels text documents with one or more predefined categories (possibly organized in a hierarchy); information filtering (or routing), which matches input documents with users' interest profiles, and question answering, which aims to extract specific (and preferably short) answers rather then provide full documents containing them.

Text categorization (TC) deals with assigning category labels to natural language documents. Categories come from a fixed set of labels (possibly organized in a hierarchy) and each document may be assigned one or more categories. Text categorization systems are useful in a wide variety of tasks, such as routing news and e-mail to appropriate corporate desks, identifying junk email, or correctly handling intelligence reports.

The majority of existing text classification systems represent text as a *bag of words*, and use a variant of the vector space model with various weighting schemes (Salton and McGill, 1983). State-of-the-art systems for text categorization use a

[1] While the term "Information Retrieval" does not by itself imply that the information being retrieved is homogeneous (and in fact multimedia IR systems dealing with collections of sound and image files are becoming more popular), in what follows we only discuss IR applications to textual data.

variety of induction techniques, such as support vector machines, k-nearest neighbor algorithm, and neural networks. The bag of words (BOW) method is very effective in easy to medium difficulty categorization tasks where the category of a document can be identified by several easily distinguishable keywords. However, its performance becomes quite limited for more demanding tasks, such as those dealing with small categories or short documents.

Early text categorization systems were predominantly manually crafted, and since the advent of machine learning techniques to the field in early 1990s, significant improvements have been obtained. However, after a decade of steady improvements, the performance of the best document categorization systems appears to have reached a plateau. No system is considerably superior to others, and improvements are becoming evolutionary (Sebastiani, 2002). In his landmark survey, Sebastiani (2002) even hypothesized that "[the effectiveness of automated text categorization] is unlikely to be improved substantially by the progress of research."

There have been various attempts to extend the basic BOW approach. Several studies augmented the bag of words with n-grams (Caropreso, Matwin, and Sebastiani, 2001; Peng and Shuurmans, 2003; Mladenic, 1998b; Raskutti, Ferra, and Kowalczyk, 2001) or statistical language models (Peng, Schuurmans, and Wang, 2004). Others used linguistically motivated features based on syntactic information, such as that available from part-of-speech tagging or shallow parsing (Sable, McKeown, and Church, 2002; Basili, Moschitti, and Pazienza, 2000). Additional studies researched the use of word clustering (Baker and McCallum, 1998; Bekkerman, 2003; Dhillon, Mallela, and Kumar, 2003), as well as dimensionality reduction techniques such as LSA (Deerwester et al., 1990; Hull, 1994; Zelikovitz and Hirsh, 2001; Cai and Hofmann, 2003). However, these attempts had mostly limited success.

We believe that the bag of words approach is inherently limited, as it can only use those pieces of information that are explicitly mentioned in the documents, and only if the same vocabulary is consistently used throughout. The BOW approach cannot generalize over words, and consequently words in the testing document that never appeared in the training set are necessarily ignored. Nor can synonymous words that appear infrequently in training documents be used to infer a more general principle that covers all the cases. Furthermore, considering the words as an unordered bag makes it difficult to correctly resolve the sense of polysemous words, as they are no longer processed in their native context. Most of these shortcomings stem from the fact that the bag of words method has no access to the wealth of world knowledge possessed by humans, and is therefore easily puzzled by facts and terms that cannot be easily deduced from the training set.

To illustrate the limitations of the BOW approach, consider document #15264 in Reuters-21578, which is one of the most frequently used datasets in text categorization research. This document discusses a joint mining venture by a consortium of companies, and belongs to the category "copper." However, this fairly long document mentions only briefly that the aim of this venture is mining copper; rather, its main focus is on the mutual share holdings of the companies involved (Teck Corporation, Cominco, and Lornex Mining), as well as other mining activities of the consortium. Consequently, the three very different text classifiers that we used (SVM, KNN and C4.5) failed to classify the document correctly. This comes as no surprise—"copper" is a fairly small category, and none of these companies, nor the location of the venture (Highland Valley in British Columbia) is ever mentioned in the training set for this category. The failure of the bag of words approach is therefore unavoidable, as it cannot reason about the important components of the story.

We analyze typical problems and limitations of the BOW method in more detail in Section 2.2.

1.1 Proposed Solution

In order to break through the existing performance barrier, a fundamentally new approach is apparently necessary. One possible solution is to depart completely from the paradigm of induction algorithms in an attempt to perform deep understanding of the document text. Yet, considering the current state of natural language processing systems, this does not seem to be a viable option (at least for the time being). Lacking full natural language understanding, we believe that in many cases common-sense knowledge and domain-specific knowledge may be used to improve the effectiveness of text categorization by generating more informative features than the mere bag of words.

Over a decade ago, Lenat and Feigenbaum (1990) formulated the *knowledge principle*, which postulated that "If a program is to perform a complex task well, it must know a great deal about the world it operates in." The recognition of the importance of world knowledge led to the launching of the CYC project (Lenat and Guha, 1990; Lenat, 1995).

We therefore propose an alternative solution that capitalizes on the power of existing induction techniques while enriching the language of representation, namely, exploring new feature spaces. Prior to text categorization, we employ a *feature generator* that uses common-sense and domain-specific knowledge to enrich the bag of words with new, more informative and discriminating features. Feature generation is performed automatically, using machine-readable repositories of knowledge. Many sources of world knowledge have become available in recent years, thanks to rapid advances in information processing, and Internet proliferation in particular. Examples of general purpose knowledge bases include the Open Directory Project (ODP), Yahoo! Web Directory, and the Wikipedia encyclopedia.

Feature generation (also known as *constructive induction*) studies methods that endow the learner with the ability to modify or enhance the representation language. Feature generation techniques search for new features that describe the target concepts better than the attributes supplied with the training instances. These techniques were found useful in a variety of machine learning tasks (Matheus, 1991; Fawcett, 1993; Markovitch and Rosenstein, 2002). A number of feature generation algorithms were proposed (Pagallo and Haussler, 1990; Matheus and Rendell, 1989; Hu and Kibler, 1996; Murphy and Pazzani, 1991), which led to significant improvements in performance over a range of classification tasks.

Feature generation methods were also attempted in the field of text categorization (Kudenko and Hirsh, 1998; Mikheev, 1999; Scott, 1998). However, their application did not yield any substantial improvement over the standard approaches. By design, these methods were mostly limited to the information present in the texts to be classified. Specifically, they made little use of linguistic or semantic information obtained from external sources.

Our aim is to empower machine learning techniques for text categorization with a substantially wider body of knowledge than that available to a human working on the same task. This abundance of knowledge will to some extent counterbalance the superior inference capabilities of humans.

In this thesis we use two repositories of world knowledge, which are the largest of their kind—the Open Directory and the Wikipedia encyclopedia. The Open Directory catalogs millions of Web sites in a rich hierarchy of 600,000 categories, and represents the collective knowledge of over 70,000 volunteer editors. Thus, in the above Reuters example, the feature generator "knows" that the companies mentioned are in the mining business, and that Highland Valley happens to host a copper mine. This information is available in Web pages that discuss the companies and their operations, and are cataloged in corresponding ODP categories such as MINING_AND_DRILLING and METALS. Similarly, Web pages about Highland Valley are cataloged under REGIONAL/NORTH_AMERICA/CANADA/BRITISH_COLUMBIA. Wikipedia is by far the largest encyclopedia in existence with over 1 million articles contributed by hundreds of thousands of volunteers. Even though Wikipedia editors are not required to be established researchers or practitioners, the open editing approach yields remarkable quality. A recent study (Giles, 2005) found Wikipedia accuracy to rival that of Encyclopaedia Britannica. We discuss the ODP and Wikipedia in more detail in Sections 4.1 and 4.2, respectively.

To tap into this kind of knowledge, we build an *auxiliary* text classifier that is capable of matching documents with the most relevant concepts of the Open

Directory and Wikipedia. We then augment the conventional bag of words with new features that correspond to these concepts. Representing documents for text categorization in this knowledge-rich space of words and constructed features leads to substantially greater categorization accuracy. It is essential to mention that this entire scheme works fully automatically. Given a knowledge repository, the feature generator examines documents and enriches their representation in a completely mechanical way.

We chose text categorization (TC) as the first exploration area of information retrieval. Lewis (1992a) suggested that text categorization is more suitable for studying feature effectiveness than text retrieval. This is because in the case of retrieval, user requests are usually short and ambiguous, limiting the possibilities to experiment with different indexing terms. On the other hand, in the case of TC, documents are long and manually classified, allowing statistical analysis of features without any additional user intervention. This allows to study text representation separately from query interpretation.

In order to intuitively explain the necessity for feature construction, let us draw a parallel with a (remotely related) field of speech processing. Speech signals are usually sampled at a fixed rate of several dozen times a second, yielding a feature vector for each signal frame of 20–50 milliseconds. Obviously, understanding the *contents* of speech using these vectors alone would be a Sisyphean task. On the other hand, analyzing these feature vectors at the macro-level and combining them into much longer sequences allows one to achieve very good results. A similar situation occurs in image processing, where values of individual pixels are combined into higher-level features. Of course, text words carry significantly more meaning than speech frames or image pixels. Nevertheless, as we show in this thesis, feature construction based on background knowledge leads to more sophisticated features that greatly contribute to the performance of automatic text processing.

It is interesting to observe that traditional machine learning data sets, such as those available from the UCI data repository (Blake and Merz, 1998), are only available as feature vectors, while their feature set is essentially fixed. On the other hand, textual data is almost always available in raw text format. Thus, in principle, possibilities for feature generation are more plentiful and flexible.

Our approach is not limited to text categorization and can be applied to other tasks in natural language processing. In order to demonstrate the generality of our approach, we also apply our feature generation methodology for assessing semantic relatedness of natural language texts.

Prior work on semantic relatedness of words and texts was based on purely statistical techniques that did not make use of background knowledge, such as the Vector Space Model (Baeza-Yates and Ribeiro-Neto, 1999) or LSA (Deerwester et al., 1990), as well as on using the WordNet electronic dictionary (Fellbaum, 1998). Here we propose a novel method, called Explicit Semantic Analysis (ESA), for fine-grained semantic interpretation of unrestricted natural language texts. ESA uses our feature generation techniques to represent meaning of input texts in a high-dimensional space of concepts derived from the ODP and Wikipedia. The feature generator maps a text fragment into a long feature vector in this space. Comparing vectors in this space using any familiar distance metric (e.g., the cosine metric (Zobel and Moffat, 1998)) allows to automatically compute the degree of semantic relatedness between input fragments of natural language text. Empirical evaluation confirms that the use of ESA improves the existing state of the art in the field by 34% for computing relatedness of individual words, and by 20% for longer texts.

1.2 Contributions of This Thesis

This thesis embodies several contributions.

1. We proposed a framework and a collection of algorithms that perform feature generation using very large-scale repositories of human knowledge. Performing feature generation using external information effectively capitalizes on human knowledge encoded in these repositories, leveraging information that cannot be deduced solely from the texts being classified.

2. We proposed a novel kind of contextual analysis performed during feature generation, which views the document text as a sequence of local contexts, and implicitly performs word sense disambiguation.

3. Instantiating our feature generation methodology for two specific knowledge repositories, the Open Directory and Wikipedia, led to major improvements in text categorization performance over a broad range of test collections, breaking the existing performance plateau. Particularly notable improvements have been observed in categorizing short documents, as well as categories with few training examples.

4. We formulated a new approach to automatic semantic interpretation of natural language texts using repositories of knowledge concepts. To this end, we used our feature generation methodology that transforms an input fragment of text into a high dimensional concept space. The Explicit Semantic Analysis we proposed based on this methodology led to major improvements in assessing semantic relatedness of texts.

5. We also describe a way to further enhance the knowledge embedded in the Open Directory by several orders of magnitude through crawling the World Wide Web.

1.3 Thesis Outline

The rest of the paper is organized as follows. In Section 2 we provide background on text categorization and feature generation. Section 3 describes our feature generation methodology that uses repositories of human knowledge to overcome limitations of the conventional bag of words approach. Section 4 instantiates this methodology with two particular knowledge resources, the Open Directory Project and the Wikipedia encyclopedia. In Section 5 we outline the implementation details of our system, and report the results of evaluating the proposed methodology empirically on a variety of test collections. Section 6 presents an application of our feature generation methodology to the task of automatically assessing the degree of semantic relatedness of natural language texts. In Section 7 we discuss our methodology in the context of prior work and related literature. Section 8 concludes the thesis and outlines directions for future research.

Chapter 2

Background

In this section we provide some background in a number of related areas. Section 2.1 reviews the existing approaches to text categorization, while Section 2.3 presents an account of feature generation.

2.1 Text Categorization

Text categorization (TC, also known as *text classification*) deals with assigning category labels to natural language documents. Categories normally come from a fixed set of labels, and may optionally be organized in a hierarchy. Under various definitions, the documents may be labeled with one or many categories. If each document is labeled with precisely one category, the problem is called *single-labeled*. If documents may be assigned either no categories or several categories at once, then we are dealing with *multi-labeled* categorization.

Research in automatic text categorization started in the 1960s, while most articles cite Maron's work on probabilistic indexing (Maron, 1961) as the first major work in the field. At the beginning, many text categorization systems were built around manually-defined sets of rules, as exemplified by the Construe system (Hayes et al., 1990) developed for Reuters. Obviously, it is very time-consuming to acquire rules by manual labor, and moreover, such rules cannot be easily reused across domains (or even across data sets from the same domain that have different category focus and hence different word usage patterns). Consequently, the *machine learning* approach prevails, where the classifier is built automatically by learning from a training set of documents.

In order to improve categorization accuracy, researchers occasionally augment the automatic induction process through some manual intervention. Most frequently this is done by defining additional features; for example, for a junk email filtering problem, Sahami et al. (1998) used a set of non-textual features such as the time of the day the message was sent, or whether it had any files attached.

In the *operational* text categorization setting (that is, in commercial systems), incorporation of fine-tuned manually defined rules is also occasionally used. To complete this description of human involvement, we shall mention the issue of *comprehensibility* of the learned model. As in other application areas of machine learning, humans who operate the computer tend to feel more confident about the classification result if they can "understand" the way it was produced. Some systems, notably, those using decision trees or explicitly manipulating decision rules (e.g., Ripper (Cohen, 1995)), construct models that can be readily interpreted by humans. In other systems, such as those built around neural networks, this issue might constitute a considerable challenge.[1]

Sebastiani (2002) and Yang (1999) present two very elaborate surveys in the area of text categorization.

2.1.1 Document Features

The absolute majority of works in the field use plain language words as features. In the dichotomy defined by Sebastiani (2002), these works only use *endogenous* knowledge (i.e., extracted from the documents proper, as opposed to externally supplied, or *exogenous*, knowledge). Whenever plain words are used as features, they may be optionally *stemmed*, by collapsing morphological variants to the same indexing term. Note, however, that results on the usefulness of stemming remain inconclusive (Sebastiani, 2002, Section 5.1). Our experimental system has a stemming component whose invocation is subject to run-time configuration. This component is based on our own, enhanced implementation of the Porter (Porter, 1980) algorithm. A number of works investigated the usefulness of phrases (either *syntactically* or *statistically* motivated), but the results were mostly discouraging, even though intuitively one would expect that phrases do carry information important for classification.

Fuernkranz, Mitchell, and Riloff (2000) used linguistic phrases based on information extraction patterns produced by the AutoSlog-TS system. The motivation behind these features is that they are supposed to capture some of the syntactic structure of natural language text. For example, given a sentence "I am a student of CS at CMU", the following features are extracted: "I am __", "__ is student", "student of __", and "student at __". The main conclusion of this study was that linguistic features can improve the precision of TC at the low recall end; they do not improve precision at the high recall end, since they have very narrow focus.

[1] Models built by support vector machines, which operate feature vectors in multidimensional hyperspaces, are usually considered to be non-trivial for human interpretation. Interestingly, a work by Dumais et al. (1998), sketches a possible way of such interpretation by examining the magnitude of values in the model vectors.

Dumais et al. (1998) used several kinds of NLP-derived phrases, namely, factoids (e.g., "Salomon_Brothers_International", "April_8"), multi-word dictionary entries (e.g., "New_York", "interest_rate"), and noun phrases (e.g., "first_quarter", "modest_growth"). Again, these features had no impact on classification accuracy with Naive Bayes, and even hurt slightly with SVMs.

A series of works by Mladenic and Grobelnik (Mladenic and Grobelnik, 1998b; Mladenic and Grobelnik, 1998a; Mladenic, 1998b) used Naive Bayes to classify Web documents collected with the aid of *Yahoo!* search engine. In addition to plain words, they considered n-grams (up to 5-grams) that were built iteratively, constructing i-grams on the ith pass, and deleting infrequent features after each iteration.

Caropreso, Matwin, and Sebastiani (2001) used a more sophisticated notion of n-grams, where each n-gram comprised an alphabetically ordered sequence of n word stems. Using an ordered sequence of stems (with stop words removed) allowed to approximate *concept indexing*; for example, expressions such as "information retrieval" and "the retrieving of information" were effectively collapsed to the same feature. N-grams and unigrams (regular words) competed against each other to be selected by the feature selection algorithm. The authors evaluated the n-grams in a so-called "learner-independent" way, by scoring the candidate features with different feature selection functions rather then directly analyzing text categorization performance. This research concluded that albeit bigrams can frequently be better predictors of class membership (as judged by their feature selection scores such as information gain), their addition does not necessarily improves classification results, and sometimes may even adversely affect the performance.

Lewis (1992a) conjectured that phrase indexing is less effective than word indexing and requires more features. He argued that although phrases have better semantic qualities (expressing more complex concepts) than plain words, they are used far less frequently, therefore, their poor statistical properties outweigh any semantic advantages they may have.

Sahami et al. (1998) designed a system for junk email filtering, which used domain-specific features such as the (Internet) domain from which the message was sent, the time of the day, the percentage of punctuation characters, or the presence of attachments. The authors defined approximately 20 non-phrasal manually crafted features that "required very little person-effort to create". These domain-specific features were combined with automatically collected terms (due to the regular bag-of-words representation), and then feature selection was performed using the mutual information criterion. This work further suggested the use of domain-specific features for the TC task in general, and proposed examples of appropriate features, such as document authors, author affiliations, and publishers. It should be noted that these are actually *extra-linguistic features*, since they do not carry genuine linguistic knowledge, or knowledge about the domain of the texts to be classified.

Ghani et al. (2000) described a data mining system that used several types of features to discover new facts about public companies. The authors did not use feature generation *per se*, but nevertheless their sources of features are interesting, and relevant to our present discussion. The database of companies comprised a collection of HTML pages from Web sites describing these companies' activities. Three kinds of features were used. *Extracted features* (e.g., performs-activity, officers) were obtained from the HTML pages using information extraction techniques, while additional extracted features (e.g., sector) were determined by Naive Bayes classification of Web pages. *Wrapped features* (e.g., competitor, subsidiary) resulted from a collection of *wrappers* developed for the Hoovers Internet database of companies[2]. The wrappers assumed a uniform format of Hoovers pages containing information about the companies, and extracted the values of predefined fields. *Abstracted features* (e.g., same-state, reciprocally-competes) were specified manually to provide the data mining algorithms with background knowledge; the values for these features were obtained from cross-referencing other features. Based on these features, the authors employed C5.0 (an improved version of C4.5 (Quinlan, 1993)) to mine previously unobserved dependencies in the data. For example, the system could detect that many companies that offer computer software and services retain Pricewaterhouse Coopers or Ernst & Young as their auditors.

2.1.2 Feature Selection

Term (or feature) selection is necessary to reduce noise, as well as to prevent overfitting. Some machine learning techniques exhibit inferior performance when presented with too many attributes[3], so it is essential to select only the best ones.

Lewis (1992a) and Sebastiani (2002) note that in order to avoid overfitting, the number of training examples should be commensurate with the number of features; a common rule of thumb is that the number of training examples per class should be at least ten times the number of features (Jain, Duin, and Mao, 2000). Sebastiani (1999) brings a simple yet instructive example for the case of possible overfitting. If a classifier for category *Cars for sale* were trained on only three examples, in two of which the car sold was yellow, then the classifier might mistakenly consider "yellowness" an essential property of this category. In the light of this example, it would be interesting to apply *knowledge-based* feature

[2] *Hoovers Online*, `http://www.hoovers.com`.

[3] One notable example is the k-nearest neighbor (KNN) algorithm, which usually does not weigh features differently according to their discriminative ability, and thus spurious features simply increase the amount of noise.

selection techniques, to ascertain which features represent *intrinsic* properties of the concept to be learned.

Two main approaches for feature selection are *filtering* and *wrapper* model (John, Kohavi, and Pfleger, 1994). The filtering approach receives a set of features, and *filters* it independently from the induction algorithm. The wrapper model *searches* for good feature subsets, and evaluates them using n-fold cross-validation on the training data. This scheme may be used in conjunction with any induction algorithm, which is used for evaluating feature subsets on the validation set. The search for feature subsets can be performed using simple greedy algorithms such as *backward elimination* or *forward selection*, or more complex ones that can both add and delete features at each step.

Since the wrapper model requires much more computation, filtering is the more common type of feature selection. This is especially true in the domain of textual information retrieval, where using the bag-of-words model results in a huge number of features. A number of feature selection techniques were described in the TC literature, while Yang and Pedersen (1997) found document frequency (DF), information gain (IG) and χ^2 (CHI) to be the most effective (reducing the feature set by 90-98% with no performance penalty, or even a small performance increase due to removal of noise). Yang and Pedersen (1997) also observed that contrary to a popular belief in information retrieval that common terms are less informative, document frequency, which prefers frequent terms (except for stop words), was found to be quite effective for text categorization.

Feature selection may be either *local*, resulting in category-specific features, or *global*, yielding collection-wide features. Document frequency is immediately suitable for global feature selection, while in order to adapt information gain and χ^2 to global operation, the sum of scores, weighted average or the maximum score (over categories) can be used.

When category-specific features are used, a problem may arise for categories with little training data. When only a few documents are available for a category, the number of candidate features is simply too small, so even if all of them are selected, document vectors in this category may be extremely sparse (and some may even be empty). To overcome this problem, one can unconditionally select some minimum number of features regardless of their actual scores, or "back off" to global features whenever an insufficient number of local features are available.

Recently, Joachims (1998) argued that *support vector machines* are very robust even in the presence of numerous features. He further claimed that the multitude of text features are indeed useful for text categorization. To substantiate this claim, Joachims used the Naive Bayes classifier with feature sets of increasing size, where features were first ordered by their discriminative capacity (as predicted by the information gain criterion), and then the most powerful features were *removed.* The classifier trained on these "low-utility" features performed markedly better than random assignment of categories to documents, thus implying that all features are relevant and should be used. Based on these findings, many later works using SVMs did not perform any feature selection at all (Leopold and Kindermann, 2002; Lewis et al., 2004). At the same time, others achieved very decent results while using some form of feature selection (notably, the result reported by Dumais et al. (1998) is considered the best one for the Reuters-21578 collection), so the evidence in this respect remains inconclusive.[4]

Observe that filtering techniques ignore mutual dependence between features, even though features are usually not completely independent. In the domain of text categorization, where features are plain words, there is naturally a considerable dependence among them. One simple approach to address this issue has been proposed by Soucy and Mineau (2001). They first select a small number of features according to the information gain criterion, and then further select only those features that both have high information gain (above some predefined threshold) *and* do not co-occur too often with the features already selected. Another feature selection technique recently proposed for text categorization addresses a situation that arises in processing huge data sets, such as *Reuters Corpus Volume 1* (Lewis et al., 2004). In such cases, there exists a trade-off between the size of the feature space and the amount of training documents that can be used for learning. To circumvent this problem, Brank et al. (2002) proposed to first train a linear SVM classifier in the full feature space using only a fraction of the training data, then use the trained model to rank the features and retain only the best ones. This way, feature selection involves examining the normal vector to the hyperplane separating the classes, and removing features that correspond to the vector components with low absolute values (since they have less impact on the classification outcome than those with high values). Finally, a new model is trained (using either the same or a different classifier), which only makes use of the features selected in the previous step, but now taking advantage of all the training data.

In addition to the "principled" feature selection schemes described above, two additional steps are frequently performed, namely, removal of stop words[5] and

[4] In our own experiments, using 10% of features instead of the entire feature set has little effect for Reuters-21578 (-0.1% ... +0.3% depending on the category set), and a small positive effect of 1.3% for the Movie Reviews data set (Pang, Lee, and Vaithyanathan, 2002). For the 20 Newsgroups collection (Lang, 1995), using all the features improves SVM results by as much as 4.7%, but this is due to the particular nature of newsgroup postings, which exhibit a very large and diversified vocabulary.

[5] In an application of text classification techniques to information extraction, where the task was to estimate the relevancy of extracted patterns to various categories, Riloff (1995) found that stemming and removal of function words (e.g., prepositions) may harm the performance of TC considerably. This happens because specific words forms may be more characteristic than

words occurring in the collection less than some predefined number of times (or less than in some predefined number of documents). For the former task, many researchers adopted the stop word lists developed by Lewis (1992b) and Salton (1971).

2.1.3 Feature Valuation

After the features have been selected, they need to be assigned values for each document vector. This step is commonly known as *feature valuation* or *term weighting*. Numerous term weighting schemes are available, while most can be described as particular cases of the *tf.idf* family introduced by Salton and Buckley (1988) in the SMART project.

Each scheme can be represented as a triple of parameters XYZ, where X stands for the term frequency factor, Y for the document frequency, and Z for the normalization method. A list of the most frequently used schemes is given below[6], and further details are available in (Hersh et al., 1994; Salton and Buckley, 1988; Singhal, 1998; Manning and Schuetze, 2000, pp.541–544).

The following definitions describe the weighting of term t_k in document d_j. We use N to denote the total number of documents in the collection, $count(t_k, d_j)$ — the number of times t_k occurs in d_j (*term frequency*), and df_k — the number of documents in the collection that contain t_k (*document frequency*).

- X (term frequency)
  - l (logarithmic) $= 1 + \log(count(t_k, d_j))$
  - L (log-average) $= \frac{1+\log(count(t_k,d_j))}{1+\log(tf_{avg}(d_j))}$,
    where $tf_{avg}(d_j)$ — average term frequency in the document
  - a (augmented) $= 0.5 + \frac{0.5*count(t_k,d_j)}{tf_{max}(d_j)}$,
    where $tf_{max}(d_j) = \max_i count(t_i, d_j)$ — maximum term frequency in the document
  - n (natural) $= count(t_k, d_j)$
  - b (binary) $= \begin{cases} 1, & if\ t_k \in d_j \\ 0, & otherwise \end{cases}$
- Y (document frequency)
  - t (inverse document frequency) $= \log(\frac{N}{df_k})$

others of particular categories. Manning and Schuetze (2000) also note that "little" words often prove useful for the task of author identification (which can be easily cast as a classification problem).

[6] All these schemes are implemented in our $\mathcal{H}$OGWARTS text categorization system.

  - n (natural) $= 1.0$

- Z (normalization)
  - c (cosine) $= \frac{1}{\sqrt{\sum_i (weight\ of\ term_i)^2}}$
  - u (pivoted unique normalization) $=$ $\frac{1}{(1-slope)+slope*\frac{number\ of\ unique\ words\ in\ the\ document}{average\ number\ of\ unique\ words\ per\ document}}$
  - n (no normalization)

Of these, the *ltc* scheme has been found the most effective (Yang, 1999; Sebastiani, 2002). Putting everything together, *ltc* stands for logarithmic weighting of occurrence counts (l), inverse document frequency (t), and cosine normalization (c):

$$tfidf(t_k, d_j) = tf(t_k, d_j) \cdot \log \frac{N}{df_k},$$

where

$$tf(t_k, d_j) = \begin{cases} 1 + \log count(t_k, d_j), & if\ count(t_k, d_j) > 0 \\ 0, & otherwise \end{cases}.$$

Finally, cosine normalization is applied to *tf.idf* weights to disregard differences in document length, by weighting all components of the feature vector as follows:

$$w_{k_j} = \frac{tfidf(t_k, d_j)}{\sqrt{\sum_{s=1}^{r} tfidf(t_s, d_j)^2}},$$

where r is the number of selected features.

2.1.4 Metrics

Following (Yang, 1999), we define text categorization performance measures using the following two-way contingency table:

	Classification = Yes	Classification = No
Correct = Yes	a	b
Correct = No	c	d

Then the following metrics can be introduced:

- *precision:* $p = a/(a + c)$ (undefined when $a + c = 0$);
- *recall:* $r = a/(a + b)$ (undefined when $a + b = 0$);

- *fallout:* $f = c/(c+d)$ (undefined when $c+d=0$);
- *accuracy:* $Acc = (a+d)/n$, where $n = a+b+c+d > 0$;
- *error:* $Err = (b+c)/n$, where $n = a+b+c+d > 0$.

Historically, *accuracy* and *error* are the standard metrics used in machine learning experiments. However, these metrics are hardly suitable for most text categorization applications. In real-life text collections there are many more negative examples than positive ones (usually, by orders of magnitude). Therefore, since *accuracy* and *error* have the total number of examples in the denominator, they are very insensitive to changes in true classification performance, and commonly produce very large (in case of *accuracy*) and very small (in case of *error*) values (*fallout* suffers from a similar problem).

The basic metrics commonly used to evaluate text categorization performance are *precision* and *recall*, taken from the mainstream information retrieval research. When it is desirable to visualize the performance of a TC system, the value of precision may be plotted for a number of values of recall; borrowing from electrical engineering terminology, the resulting graph is usually called a *receiver operating curve* (ROC).

Sometimes, it is convenient to have a single measure instead of two, in which case the F-measure (van Rijsbergen, 1979, Chapter 7) may be used:

$$F_\beta(p,r) = \frac{(\beta^2+1)pr}{\beta^2 p + r},$$

where p and r denote *precision* and *recall*, respectively.

The β parameter allows fine-tuning the relative importance of precision over recall. When both metrics are equally important, the F_1 measure is used: $F_1 = 2pr/(p+r)$.

The precision-recall *Break-Even Point (BEP)* is occasionally used as an alternative to F_1. It is obtained by either tuning the classifier so that precision is equal to recall, or sampling several $(precision, recall)$ points that bracket the expected BEP value and then interpolating (or extrapolating, in the event that all the sampled points lie on the same side).

In the presence of two or more categories, it is handy to have a single value that reflects the overall performance. In such a case, either *micro-averaging* or *macro-averaging* of category-specific performance measures is used. The former accumulates a, b, c, d values over the documents in all categories, and then computes precision and recall (as well as F_1 and BEP) values. The latter simply averages precision and recall computed for each category individually.[7] Macro-averaging

[7] In the case of macro-averaging, there is an issue of evaluating the ratio 0/0 that can result in some underpopulated categories. Various approaches set it to either 0, 1, or some very small

ignores the relative sizes of categories, which may or may not be appropriate in each case at hand. Therefore, although most researchers report micro-averaged metrics, others perform both types of averaging.

2.1.5 Notes on Evaluation

It should be remembered that there is an upper limit on the performance of text categorization systems, as even humans occasionally disagree on assignment of categories to documents. This is a manifestation of *inter-indexer inconsistency* (Sebastiani, 2002) common in information retrieval. Rose, Stevenson, and Whitehead (2002) studied this phenomenon in depth, analyzing the consistency of classification of Reuters news items by a group of Reuters editors. They found the inter-editor consistency to be quite high (about 95%), but still not 100%. Observe also that this high correlation was probably due to the fact that Reuters employs seasoned information professionals, whose judgement is further bound by strict in-house policies on labeling news stories; therefore, in less constrained circumstances the agreement among humans would probably be lower.

2.2 Problems With the Bag of Words Approach

1. Words that appear in *testing* documents but not in *training* documents are completely ignored by the BOW approach. Since the classification model is built with a subset of words that appear in the training documents, words that do not appear there are excluded by definition. Lacking the ability to analyze such words, the system may overlook important parts of the document being classified.

   Example: Document #15264 from Reuters-21578 described in the Introduction presents a perfect example of this limitation. This document describes a copper-mining venture formed by a group of companies, whose names are not mentioned even once in the training set, and are thus ignored by the classification model.

2. Words that appear infrequently in the training set, or appear just once, are mostly ignored even if they are essential for proper classification. It often happens that human annotators assign a document to a certain category based on some notion briefly mentioned in the document. If the words that describe this notion do not appear with sufficient frequency elsewhere in the training set, then the system will overlook the real reason for this

positive value. Observe that the different decisions can adversely affect the overall averaged performance score.

document's annotation. Consequently, it will either come up with some spurious association between the actual category and unrelated words or ignore this document as a training example altogether.

Example: Suppose we have a collection of pharmaceutical documents and are trying to learn the concept of antibiotics. If a particular training document describes the results of a clinical trial for a new antibiotic drug, and mentions it only by a brand name that does not appear elsewhere in the training set, the system will likely miss this important piece of evidence.

3. The problem described in the previous item can manifest itself in a more extreme way. Suppose we have a group of related words, where each word appears only a few times in the collection, and few documents contain more than one word of the group. As a result, the connection between these words remains implicit and cannot be learned without resorting to external knowledge. External knowledge, however, allows us to determine that certain words are related. Furthermore, we can use the generalization ability of hierarchical knowledge organization to establish that the words correspond to specific instances of the same general notion.

   Example: Consider a collection of clinical narrative reports on administering various antibiotic drugs. Since such reports are circulated among medical professionals, they are likely to refer to specific drugs by name, while omitting the knowledge already shared by the target audience. Hence, the reports will likely not explain that each drug is actually an antibiotic. In the absence of this vital piece of knowledge, the BOW approach can easily fail to learn the notion shared by the reports.

4. A critical limitation of the BOW approach lies in its ignorance of the connections between the words. Thus, even more difficult than the problem described in the previous item, is the one where we have several related phrases or longer contexts, while the connection between them is not stated in any single document.

   Example: Consider again a collection of clinical reports, which are inherently rich in diverse medical terminology. Often, each report describes the case of a single patient. Thus, without extensive medical knowledge it would be nearly impossible to learn that Lown-Ganong-Levine Syndrome and Wolff-Parkinson-White Syndrome are different kinds of arrhythmia, while Crigler-Najjar Syndrome and Gilbert Syndrome are two kinds of liver diseases.

5. Because contextual adjacency of words is not taken into account by the BOW approach, word sense disambiguation can only be performed at the

level of entire documents, rather than at much more linguistically plausible levels of a single sentence or paragraph.

Example: As an extreme example of this limitation, consider a document about the Jaguar company establishing a conservation trust to protect its namesake[8] animal. This fairly long document is devoted mainly to the preservation of wildlife, while briefly covering the history of the car manufacturer in its last paragraph. Taken as a single bag of words, the document will likely be classified as strongly related to jaguar the animal, while the cursory mention of Jaguar the company will likely be ignored.

Some of these limitations are due to data sparsity—after all, if we had infinite amounts of text on every imaginable topic, the bag of words would perform much better. Many studies in machine learning and natural language processing addressed the sparsity problem. Simple approaches like smoothing (Chen and Goodman, 1996) allocate some probability mass for unseen events and thus eliminate zero probabilities. Although these approaches facilitate methods that are sensitive to zero probabilities (e.g., Naive Bayes), they essentially do not introduce any new information. More elaborate techniques such as transfer learning (Bennett, Dumais, and Horvitz, 2003; Do and Ng, 2005; Raina, Ng, and Koller, 2006) and semi-supervised learning (Goldberg and Zhu, 2006; Ando and Zhang, 2005a; Ando and Zhang, 2005b), leverage cooccurrence information from similar learning tasks or from unlabeled data. Other studies that addressed the sparsity problem include using the EM algorithm with unlabeled data (Nigam, McCallum, and Mitchell, 2006; Nigam et al., 2000), latent semantic kernels (Cristianini, Shawe-Taylor, and Lodhi, 2002), transductive inference (Joachims, 1999b), and generalized vector space model (Wong, Ziarko, and Wong, 1985).

Humans avoid these limitations due to their extensive world knowledge, as well as their ability to understand the words in context rather than just view them as an unordered bag. In this thesis we argue that the limitations of the bag of words can be overcome by endowing computers with access to the wealth of human knowledge. Recall the sample Reuters document we considered above—when a Reuters editor originally handled this news item, she most likely knew quite a lot about the business of the companies mentioned, and based on her deep domain knowledge she easily assigned the document to the category "copper." It is this kind of knowledge that we would like machine learning algorithms to have access to.

[8] `http://www.jaguarusa.com/us/en/company/news_events/archive/Jaguar_Conservation_trust_longcopy.htm`

2.3 Feature Generation

Feature generation (FG), also known as *feature construction*, *constructive induction* or *bias shift*, is a process of building new features based on those present in the examples supplied to the system, possibly using the domain theory (i.e., information about goals, constraints and operators of the domain) (Fawcett, 1993). Feature construction techniques can be useful when the attributes supplied with the data are insufficient for concise concept learning.

Matheus (1991) proposed to use constructive induction to address the problems of disjunctive regions in the instance space (i.e., discontinuous concepts). He posed the following issues as the main questions of constructive induction, and suggested approaches to answer them from the instance-based, hypothesis-based, and knowledge-based points of view.

1. When should new features be constructed?
   - *Instance-based detection:* estimate irregularity of the membership function from the distribution of observed instances.
   - *Hypothesis-based detection:* an initial hypothesis fails to meet some performance criterion.
   - *Knowledge-based detection:* use domain knowledge.
2. What constructive operators should be used and which of the existing features should they be applied to?
   - *Instance-based selection:* search for patterns among training instances.
   - *Hypothesis-based selection:* for example, analyze the structure of branches in decision trees.
   - *Knowledge-based selection:* use domain theory.
3. Which (if any) features should be discarded?
   - *Instance-based evaluation:* use probabilistic or information-theoretic measures.
   - *Hypothesis-based evaluation:* for example, rank features according to how well they are used within the hypothesis.
   - *Knowledge-based evaluation:* Domain knowledge is generally used for the generation rather than evaluation of features. Nevertheless, features can be evaluated according to how well they conform to the domain knowledge.

Feature generation has been studied in a number of works over the recent years, and several notable algorithms have been proposed. Most prominent examples include the FRINGE (Pagallo and Haussler, 1990), CITRE (Matheus and Rendell, 1989) and GALA (Hu and Kibler, 1996) algorithms that manipulate boolean combinations of features, the ID2-of-3 (Murphy and Pazzani, 1991) algorithm that uses M-of-N concepts, various genetic algorithms that apply crossover and mutation operations to feature bit strings, and the FICUS algorithm (Markovitch and Rosenstein, 2002) that generalizes over previous approaches by using constructor functions and searches the space of generated features.

Callan (1993) suggested a feature generation technique suitable for search domains (e.g., the n-queens problem). He observed that "search goal descriptions are usually not monolithic, but rather consist of subexpressions, each describing a goal or constraint". Taken together they characterize the goal state, but they can also be used independently to measure progress in reaching the goal. This work proposed a set of heuristics for decomposing goal specifications into their constituent parts, in order to use them as features.

Classical approaches to feature generation belong to two main classes:

- **Data-driven**, in which new features are created by combining existing features in various ways. Feedback from the learned concept is typically used to suggest plausible feature combinations.
  Limitations: The amount of the improvement is limited, since the algorithm starts with the example features, and combines features one step at a time. If useful feature are complex combinations of example features, the system will have to generate and test prohibitively many features until a useful one is derived.

- **Analytical, using domain theory** to deduce appropriate new features. Using information about the domain helps create complex features in one step.
  Limitations: such systems can only create features that follow deductively from the domain theory. Many real-world domains require useful features that are not deducible from the domain theory, and these analytical systems are incapable of deriving them.

Fawcett (1991) proposed a hybrid theory of feature generation, so that useful features can be derived from abstractions and combinations of abstractions of the domain theory. Abstractions are created by relaxing conditions specified in a domain theory, using a hybrid of data-driven (bottom-up) and theory-driven (top-down) approaches.

It is important to distinguish between *feature generation* and *feature selection*. While the former attempts to construct new features not present in the original description of the data, the latter starts with a set of features and attempts to *decimate* it. Feature selection is frequently used in tandem with feature generation. Some approaches to feature generation employ a *generate and test* strategy, where a set of new features is created, which is then filtered according to some fitness criterion, then another FG iteration is performed using the original and selected constructed features, and so on. Using all the generated features without pruning the set heavily after each iteration may result in a combinatorial explosion of the number of features.

Chapter 3

Feature Generation Methodology

In Section 2.2 we discussed a number of problems with the BOW approach. We now proceed to developing a feature generation methodology that will address and alleviate these problems using repositories of human knowledge.

3.1 Overview

The proposed methodology allows principled and uniform integration of one or more sources of external knowledge to construct new features. These knowledge sources define a collection of concepts that are assigned to documents to qualify their text. In the preprocessing step, we build a feature generator capable of representing documents in the space of these concepts. The feature generator is then invoked prior to text categorization to assign a number of relevant concepts to each document. Subsequently, these concepts give rise to a set of constructed features that provide background knowledge about the document's contents. The constructed features can then be used either in conjunction with or in place of the original bag of words. The resulting set optionally undergoes feature selection, and the most discriminative features are retained for document representation.

We use traditional text categorization techniques to learn a text categorizer in the new feature space. Figure 3.1 depicts the standard approach to text categorization. Figure 3.2 outlines the proposed feature generation framework; observe that the "Feature generation" box replaces the "Feature selection" box framed in **bold** in Figure 3.1.

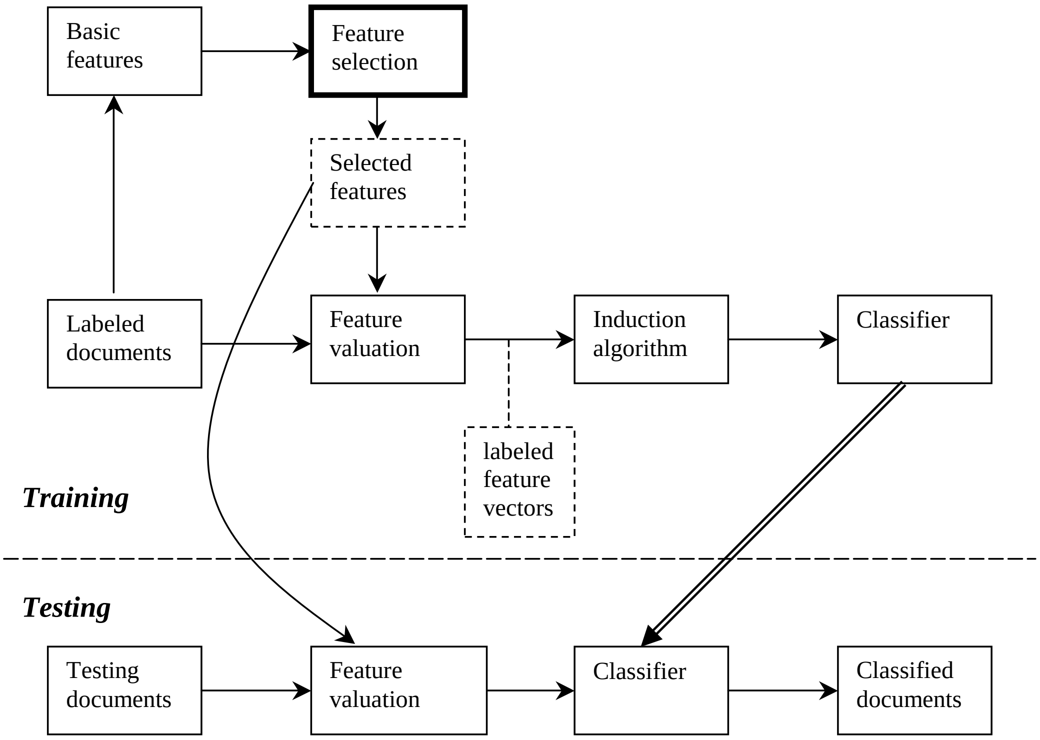

Figure 3.1: Standard approach to text categorization

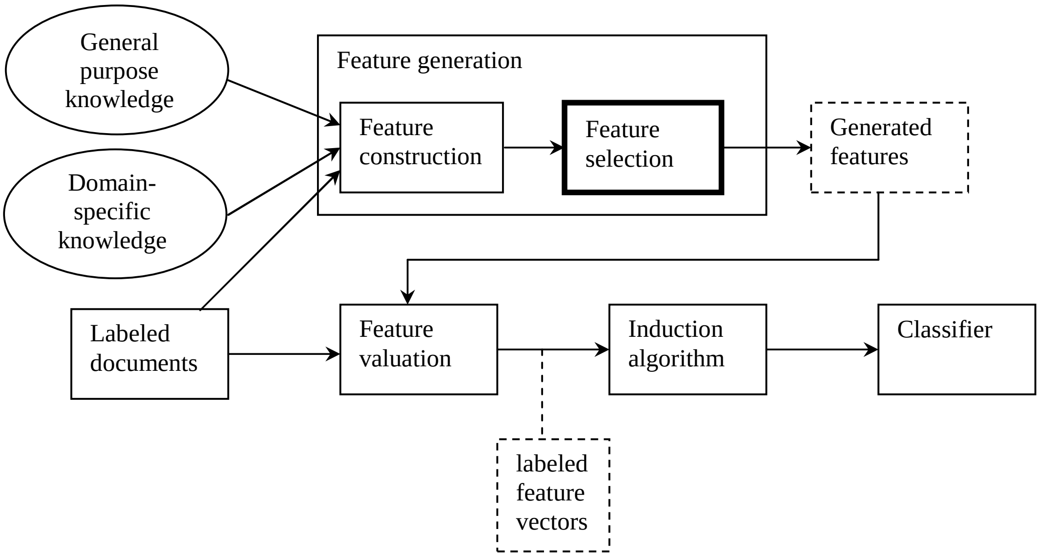

Figure 3.2: Induction of text classifiers using the proposed framework for feature generation

3.2 Requirements on Suitable Knowledge Repositories

We impose the following requirements on knowledge repositories for feature generation:

1. The repository contains a collection of *concepts*, which are defined by humans and correspond to notions used by humans in commonsense or domain-specific reasoning. Formally, let KR be a knowledge repository that contains concepts $C = \{c_0, \ldots, c_n\}$.

2. There is a collection of texts associated with each concept. The feature generator uses these texts to learn the definition and scope of the concept, in order to be able to assign it to relevant documents. We refer to these texts as *textual objects*, and denote the set of such objects associated with concept c_i as $T_i = \{t_{0,1}, \ldots, t_{i,m_i}\}$.

3. Optionally, there is a collection of relations between concepts, $R = \{r_1, \ldots, r_l\}$, where each relation is a set of pairs of concepts, $r_k = \{\langle c_i, c_j \rangle\}$. For example, one such relation could be a generalization ("is-a") relation, which organizes the concepts into a hierarchical structure. In what follows, Section 3.5 discusses the extension of our methodology to hierarchically-structured knowledge bases, and Section 3.6 discusses the use of arbitrary relations.

Let W be a set of words that appear in documents to be classified. Our goal is to build a mapping function $f : W^* \rightarrow 2^C$. We propose building the mapping function using text categorization techniques. This is a very natural thing to do, as text categorization is all about assigning documents or parts thereof to a predefined set of categories (concepts in our case). One way to do so is to use a binary learning algorithm $L(Pos, Neg)$ to build a set of n binary classifiers, $f_0, \ldots, f_n$, such that $f_i : W^* \rightarrow \{0, 1\}$. This way, individual classifiers are built using the chosen learning algorithm: $f_i = L(T_i, \bigcup_{0 \leq j \leq n, j \neq i} T_j)$. Another way to build such a mapping function is to devise a hierarchical text classifier that takes advantage of the hierarchical organization of categories. In this paper, we use a simpler approach of building a single classifier that simultaneously considers all the concepts for each input sequence of words.

We believe that the above requirements are not overly restrictive. In fact, there are quite a few sources of common-sense and domain-specific knowledge that satisfy these requirements. We list below several notable examples.

- Internet directories such as the Yahoo Web Directory[1], the Open Directory Project[2] and the LookSmart directory[3] catalog huge numbers of URLs organized in an elaborate hierarchy. The Web sites pointed at by these URLs can be crawled to gather a wealth of information about each directory node. Here each directory node defines a concept, and crawling the Web sites cataloged under the node provides a collection of textual objects for that node.

- The Medical Subject Headings (MeSH) taxonomy (MeSH, 2003), which defines over 18,000 categories and is cross-linked with the MEDLINE database of medical articles, is a notable example of a domain-specific knowledge base. The MEDLINE links allow to easily associate MeSH nodes with numerous scientific articles that are highly relevant to the node, yielding a set of textual objects for that node.

- Other domain-specific knowledge repositories are also available, notably in the terminology-rich law domain, which includes the KeySearch taxonomy by WestLaw[4] and the Web-based FindLaw hierarchy[5] (both of them cross-linked with material relevant for each node).

- The US Patent Classification[6] and the International Patent Classification[7] are exceptionally elaborate taxonomies, where each node is linked to relevant patents.

- The online Wikipedia encyclopedia[8] has a fairly shallow hierarchy but its nodes contain very high-quality articles, which are mostly noise-free (except for occasional spamming).

- In the brick-and-mortar world, library classification systems such as the Universal Decimal Classification (UDC) (Mcilwaine, 2000), the Dewey Decimal Classification (Dewey et al., 2003) or the Library of Congress Classification (Chan, 1999) provide structuring of human knowledge for classifying books. By the very virtue of their definition, each classification node can be associated with the text of books cataloged under the node. Interestingly,

[1] `http://dir.yahoo.com`

[2] `http://www.dmoz.org`

[3] `http://search.looksmart.com/p/browse`

[4] `http://west.thomson.com/westlaw/keysearch`

[5] `http://www.findlaw.com`

[6] `http://www.uspto.gov/go/classification`

[7] `http://www.wipo.int/classifications/ipc/en`

[8] `http://www.wikipedia.org`

modern book-scanning efforts such as those underway at Google and Amazon can eventually make it possible to build feature generators powered by the knowledge available in printed books.

In this work we use the ODP and Wikipedia as our knowledge repositories, due to the easy accessibility of their data on the Web. In the next section, we shall discuss the instantiation of our methodology for these two knowledge repositories. However, our methodology is general enough to facilitate other knowledge sources such as those listed above, and in our future work we intend to explore their utility as well, focusing in particular on the MeSH hierarchy for domain-specific feature generation.

A note on terminology is in order here. The most commonly used term for nodes of directories of knowledge is "category." In text categorization, however, this term normally refers to topical labels assigned to documents. To prevent possible confusion, we use the word "concept" to refer to the former notion. We represent such concepts as vectors in a high-dimensional space of "attributes." Again, we avoid using the term "features," which is reserved for denoting individual entries of document vectors in text categorization per se.

3.3 Building a Feature Generator

The first step in our methodology is preprocessing, performed once for all future text categorization tasks. In the preprocessing step we induce a text classifier that maps pieces of text onto relevant knowledge concepts, which later serve as generated features. The resulting classifier is called a *feature generator* according to its true purpose in our scheme. The feature generator represents concepts as vectors of their most characteristic words, which we call *attributes* (reserving the term *features* to denote the properties of documents in text categorization).

The feature generator operates similarly to a regular text classifier—it first learns a classification model in the space of concept attributes, and then identifies a set of concepts that are most appropriate to describe the contents of the input text fragment. Observe that the number of concepts to which the feature generator can classify document text is huge, as suitable knowledge repositories may contain tens and even hundreds of thousands of concepts. Few machine learning algorithms can efficiently handle so many different classes and about an order of magnitude more of training examples. Suitable candidates include the nearest neighbor and the Naive Bayes classifier (Duda and Hart, 1973), as well as prototype formation methods such as Rocchio (Rocchio, 1971) or centroid-based (Han and Karypis, 2000) classifiers.

3.3.1 Attribute Selection

Prior to learning a text classifier that will act as feature generator, we represent each concept as an attribute vector. To this end, we pool together all the textual objects for the concept, and represent the accumulated description with a vector of words. Using all encountered words as attributes is impractical because it yields a classification model that is too big, and because this would inevitably increase the level of noise. The former consideration is essential to allow fitting the induced model into computer memory. The latter consideration is particularly important for Web-based knowledge repositories, which are inherently plagued with noise ranging from intentional directory spamming to merely irrelevant information. To remedy the situation, we perform *attribute selection* for each concept prior to learning the feature generator.

To this end, we use standard attribute selection techniques (Sebastiani, 2002) such as information gain, and identify words that are most characteristic of a concept versus all other concepts. This approach to attribute selection is reminiscent of the approaches described by Chakrabarti et al. (1997) and by Koller and Sahami (1997). Let us denote by D_i the collection of textual objects associated with concept c_i, $D_i = \bigcup_{k=0 \ldots m_i} t_{i,k}$, and by $\overline{D_i}$ the collection of textual objects for all other concepts, $\overline{D_i} = \bigcup_{l=0 \ldots n, l \neq i} \bigcup_{k=0 \ldots m_l} t_{l,k}$. Then, we can assess the discriminative capacity of each word $w \in D_i$ with respect to $\overline{D_i}$. It is essential to note that conventional attribute selection techniques select attributes for c_i from the entire lexicon, $D_i \cup \overline{D_i}$. In our case, however, we aim at selecting words that are most characteristic for the concept, and therefore we limit the selection only to words that actually appear in the textual objects for that concept, that is, D_i.

Figure 3.3 shows the algorithm for building a feature generator. The algorithm uses a global structure $Text(c_i)$ that accumulates textual objects for concept c_i (attributes for the concept are then selected from the words occurring in this pool). We manipulate $Text(c_i)$ as an unordered bag of words. Attribute vectors for each category are stored in $Vector(c_i)$.

3.3.2 Feature Generation per se

Given a fragment of text for which we desire to generate features, we represent it as an attribute vector, and then compare it to the vectors of all knowledge concepts. The comparison can use any distance metric for comparing vectors in a high-dimensional space; in this work, we use the cosine metric. The desired number of highest-scoring concepts are then returned as generated features. Figure 3.4 outlines this process.

```
Algorithm BuildFeatureGenerator
  # Compute attribute vectors for all concepts
  BuildVectors()

  # Use an induction algorithm to train a feature generator FG
  # using the attribute vectors $Vector(c_i)$
  $FG \leftarrow InduceClassifier(\{Vector(c_i)\})$
------------------------------------------------------------
Algorithm BuildVectors()
  For each $c_i \in C = \{c_0, \ldots, c_n\}$ do
    $Text(c_i) \leftarrow \bigcup_{k=0 \ldots m_i} t_{i,k}$

    # Build the attribute vector by performing attribute selection
    # among the words of $Text(c_i)$
    $Vector(c_i) \leftarrow AttributeSelection(Text(c_i))$
    # Assign values to the selected attributes
    $Vector(c_i) \leftarrow tfidf(Vector(c_i))$
```

Figure 3.3: Building a feature generator

```
Algorithm FG($text, distanceMetric, numConcepts$)
  $TextVector \leftarrow tfidf(text)$
  For each $c_i \in C = \{c_0, \ldots, c_n\}$ do
    $Score(c_i) \leftarrow distanceMetric(TextVector, Vector(c_i))$
  Let $GeneratedConcepts$ be a set of $numConcepts$ concepts
    with the highest $Score(c_i)$
  Return $GeneratedConcepts$
```

Figure 3.4: Feature generation

3.4 Contextual Feature Generation

Feature generation precedes text categorization, that is, before the induction algorithm is invoked to build the text categorizer, the documents are fed to the feature generator.

Traditionally, feature generation uses the basic features supplied with the training instances to construct more sophisticated features. In the case of text processing, however, important information about word ordering will be lost if the traditional approach is applied to the bag of words. Therefore, we argue that

```
Algorithm ContextualFeatureGeneration($D$)
  Let $CT$ be a series of contexts for $D$
  $CT \leftarrow words(D) \cup sentences(D) \cup paragraphs(D) \cup \{D\}$
  Let $F$ be a set of features generated for $D$
  $F \leftarrow \emptyset$
  For each context $ct \in CT$ perform feature generation:
    $F \leftarrow F \cup FG(ct)$
  Represent $D$ as $BagOfWords(D) \cup F$
```

Figure 3.5: Performing feature generation for document D using the multi-resolution approach

feature generation becomes much more powerful when it operates on the raw document text. But should the generator always analyze the whole document as a single unit, as do regular text classifiers?

3.4.1 Analyzing Local Contexts

We believe that considering the document as a single unit can often be misleading: its text might be too diverse to be readily mapped to the right set of concepts, while notions mentioned only briefly may be overlooked. Instead, we propose to partition the document into a series of non-overlapping segments (called *contexts*), and then generate features at this finer level. Each context is classified into a number of concepts in the knowledge base, and pooling these concepts together to describe the entire document results in *multi-faceted* classification. This way, the resulting set of concepts represents the various aspects or sub-topics covered by the document.

Potential candidates for such contexts are simple sequences of words, or more linguistically motivated chunks such as sentences or paragraphs. The optimal resolution for document segmentation can be determined automatically using a validation set. We propose a more principled *multi-resolution* approach that simultaneously partitions the document at several levels of linguistic abstraction (windows of words, sentences, paragraphs, up to taking the entire document as one big chunk), and performs feature generation at each of these levels. We rely on the subsequent *feature selection* step (Section 3.4.2) to eliminate extraneous features, preserving only those that genuinely characterize the document. Figure 3.5 presents the feature generation algorithm.

In fact, the proposed approach tackles the two most important problems in natural language processing, namely, *synonymy* (the ability of natural languages to express many notions in more than one way), and *polysemy* (the property of natural language words to convey more than a single sense, while certain words may have as many as dozens of different, sometimes unrelated senses). When individual contexts are classified, *word sense disambiguation* is implicitly performed, thus resolving word polysemy to some degree. A context that contains one or more polysemous words is mapped to the concepts that correspond to the sense *shared* by the context words. Thus, the correct sense of each word is determined with the help of its neighbors. At the same time, enriching document representation with high-level concepts and their generalizations addresses the problem of synonymy, as the enhanced representation can easily recognize that two (or more) documents actually talk about related issues, albeit using different vocabularies.

For each context, the feature generator yields a list of concepts ordered by their score, which quantifies their appropriateness to the context. A number of top-scoring concepts are used to actually generate features. For each of these concepts we generate one feature that represents the concept itself. If the knowledge repository also defines relations between concepts, these relations can be used for generating additional features (see Sections 3.5 and 3.6).

3.4.2 Feature Selection

Using support vector machines in conjunction with bag of words, Joachims (1998) found that SVMs are very robust even in the presence of numerous features. He further observed that the multitude of features are indeed useful for text categorization. These findings were corroborated in more recent studies (Rogati and Yang, 2002; Brank et al., 2002; Bekkerman, 2003) that observed either no improvement or even small degradation of SVM performance after feature selection. Consequently, many later works using SVMs did not apply feature selection at all (Leopold and Kindermann, 2002; Lewis et al., 2004).

This situation changes drastically as we augment the bag of words with generated features. First, nearly any technique for automatic feature generation can easily generate huge numbers of features, which will likely aggravate the "curse of dimensionality." Furthermore, it is feature selection that allows the feature generator to be less than a perfect classifier. When some of the concepts assigned to the document are correct, feature selection can identify them and seamlessly eliminate the spurious ones. We further analyze the utility of feature selection in Section 5.3.5.

Note also that the categories to which the documents are categorized most likely correspond to a mix of knowledge repository concepts rather than a single one. Therefore, as the feature generator maps documents to a large set of related concepts, it is up to feature selection to retain only those that are relevant to the particular categorization task in hand.

In a related study (Gabrilovich and Markovitch, 2004) we described a class of problems where feature selection can improve SVM performance even for a bag of words. In this work, we formulated the notion of feature redundancy, and proposed a criterion for quantifying this phenomenon in order to predict the usefulness of feature selection. Further details can be found in Appendix A.

3.4.3 Feature Valuation

In regular text categorization, each word occurrence in document text is initially counted as a unit, and then feature valuation is performed, usually by subjecting these counts to TFIDF weighting (Salton and Buckley, 1988; Debole and Sebastiani, 2003). To augment the bag of words with generated features and to use a single unified feature set, we need to assign weights to generated features in a compatible manner.

Each generated feature is assigned the basic weight of 1, as in the single occurrence of a word in the bag of words. However, this weight is further multiplied by the classification score produced for each classified concept by the feature generator ($Score(c_i)$ in Figure 3.4). This score quantifies the degree of affinity between the concept and the context it was assigned to.

3.4.4 Revisiting the Running Example

Let us revisit the example from Section 1, where we considered a document that belongs to the "copper" category of Reuters-21578. Figure 3.6 illustrates the process of feature generation for this example. While building the feature generator in the preprocessing stage, our system learns the scope of mining-related ODP categories such as Business/Mining_and_Drilling, Science/Technology/Mining and Business/Industrial_Goods_and_Services/Materials/Metals. These categories contain related URLs, such as `http://www.teckcominco.com` and `http://www.miningsurplus.com`, which belong to the (now merged) Teck Cominco company. The company's prominence and frequent mention causes the words "Teck" and "Cominco" to be included in the set of attributes selected to represent the above categories.

During feature generation, the document is segmented into a sequence of contexts The feature generator analyzes these contexts and uses their words (e.g., "Teck" and "Cominco") to map the document to a number of mining-related concepts in the ODP (e.g., Business/Mining_and_Drilling). These concepts, as well as their ancestors in the hierarchy, give rise to a set of generated features that augment the bag of words. Observe that the training documents for the category "copper" underwent similar processing when a text classifier was induced. Consequently, features based on these concepts *were selected* during feature selection

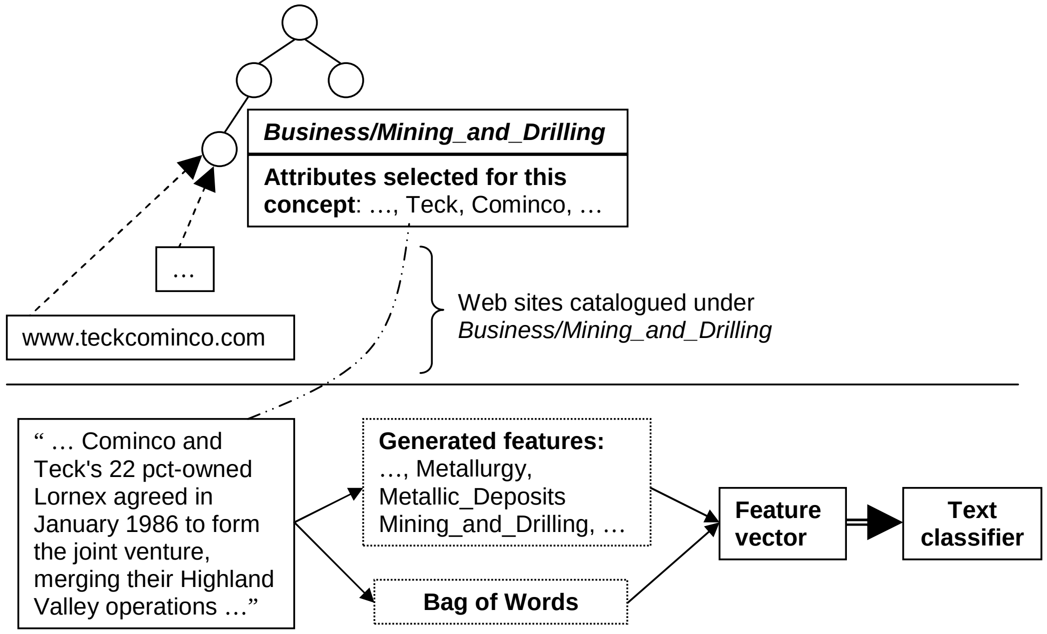
Business/Mining_and_Drilling
Attributes selected for this concept: …, Teck, Cominco, …
…
www.teckcominco.com
Web sites catalogued under Business/Mining_and_Drilling
" … Cominco and Teck's 22 pct-owned Lornex agreed in January 1986 to form the joint venture, merging their Highland Valley operations …"
Generated features: …, Metallurgy, Metallic_Deposits Mining_and_Drilling, …
Bag of Words
Feature vector
Text classifier

Figure 3.6: Feature generation example

and retained in document vectors, thanks to their high predictive capacity. It is due to these features that the document is now categorized correctly, while without feature generation it consistently caused BOW classifiers to err.

3.5 Using Hierarchically-Structured Knowledge Repositories

We now elaborate on Requirement 3 (Section 3.2) that allows knowledge repositories to optionally define relations between concepts. The simplest and most common organization of a set of concepts is using a hierarchical structure, which establishes an "is-a" relation between concepts. This way, each concept is more general that all of its children.

The Open Directory Project that we use in this study is an example of such knowledge repository. Consider, for instance, the path Top/Computers/Artificial_Intelligence/Machine_Learning/Datasets, which leads to a leaf concept in the ODP tree. In this example, the parent concept Top/Computers/Artificial_Intelligence/Machine_Learning is more general than the leaf concept Datasets, and concept

Top/Computers/Artificial_Intelligence is more general than Machine_Learning. The root concept Top is more general than any other ODP concept.

Let us now formalize this extended setting. Let c_0 be the *root concept*, which is more general than any other concept. Let $Parent(c_i)$ be a function that uniquely associates a node with its parent in the hierarchy, whereas $Parent(c_0)$ is undefined. Let $Children(c_i)$ be a function that associates a node with a set of its children, where for leaf nodes $Children(c_i) = \emptyset$. When concept c_i is more general than another concept c_j, we denote this by $c_i \sqsubseteq c_j$; this happens when $c_j \in Children^*(c_i)$, where $Children^*$ denotes the recursive application of the function (obviously, $\forall j > 0 : c_0 \sqsubseteq c_j$). Similarly, let $Parent^*(c_i)$ denote the set of ancestors of c_i, obtained through the recursive application of the $Parent$ function (of course, $\forall j > 0 : c_0 \in Parents^*(c_j)$).

Two aspects of our methodology can benefit from hierarchical organization of concepts. First, while building the feature generator, we make use of the text objects associated with each concept to learn its scope, in order to be able to assign this concept to documents in text categorization. Hierarchical organization allows us to greatly extend the amount of text associated with each concept, by taking the texts associated with all of the descendants of this concept. This is possible because the descendants represent more specific concepts, and thus it makes perfect sense to use their sample texts to enrich the text pool for the ancestor concept. Thus, for example, if a certain concept c_i is only associated with a few textual objects, we can learn its scope much more reliably by aggregating the textual objects associated with all of its descendants, $Children^*(c_i)$. Figure 3.7 provides pseudocode of the algorithm for building a feature generator using hierarchically-structured knowledge repositories.

Accumulation of textual objects from the descendants of a concept has implications for attribute selection. Let us denote by D_i^{hier} the collection of textual objects of c_i and its descendants, $D_i^{hier} = \bigcup_{k=0 \ldots m_i} t_{i,k} \cup \bigcup_{j=0 \ldots n \text{ } s.t. \text{ } c_j \in Children^*(c_i)} \bigcup_{k=0 \ldots m_j} t_{j,k}$, and by $\overline{D_i^{hier}}$ the collection of textual objects for all other concepts. Attribute selection now has to identify attributes from D_i^{hier} that are most characteristic of c_i.

However, there is an additional benefit in using a hierarchical ontology, as it allows us to perform powerful generalizations during feature construction. As explained in Section 3.3.2, the default feature construction strategy is to use the feature generator to map the document text into one or more pertinent concepts that are classified based on the document text. For the sake of this discussion, let ct denote a particular document context that undergoes feature construction, and let it be classified into concepts $c_1, \ldots, c_p$. In the presence of hierarchical organization of concepts, we can now map this context to additional concepts, namely, $\bigcup_{j=1 \ldots p} Parent^*(c_j)$. That is, the context is also mapped to concepts

```
Algorithm BuildFeatureGenerator
  # Compute attribute vectors for all concepts
  BuildVectors($c_0$)

  # Use an induction algorithm to train a feature generator FG
  # using the attribute vectors $Vector(c_i)$
  $FG \leftarrow InduceClassifier(\{Vector(c_i)\})$
------------------------------------------------------------------
Algorithm BuildVectors($c_i$)
  $Text(c_i) = \emptyset$

  # Traverse the hierarchy bottom-up, collecting the textual objects
  # of the descendants of each concept
  For each child $c_j \in Children(c_i)$ do
    BuildVectors($c_j$)
    $Text(c_i) \leftarrow Text(c_i) \cup \bigcup_{k=0 \ldots m_j} t_{j,k}$

  # Now add the textual objects for the concept itself
  $Text(c_i) \leftarrow Text(c_i) \cup \bigcup_{k=0 \ldots m_i} t_{i,k}$

  # Build the attribute vector by performing attribute selection
  # among the words of $Text(c_i)$
  $Vector(c_i) \leftarrow AttributeSelection(Text(c_i))$
  # Assign values to the selected attributes
  $Vector(c_i) \leftarrow tfidf(Vector(c_i))$
```

Figure 3.7: Building a feature generator using hierarchically-structured knowledge repositories

that are more general than the originally classified ones. Figure 3.8 presents the extended feature generation algorithm.

When knowledge concepts are organized hierarchically, the feature generation algorithm can take advantage of such organization. This way, instead of considering all existing concepts simultaneously, it can work top-down into the hierarchy, identifying several most suitable concepts at each level, as in the hierarchical text classifiers described in the literature (Koller and Sahami, 1997; Dumais and Chen, 2000; Ruiz and Srinivasan, 2002). One possible drawback of such approach, however, is that erroneous decisions made early in the process cannot be corrected later.

```
Algorithm FG($text, distanceMetric, numConcepts$)
  $TextVector \leftarrow tfidf(text)$
  For each $c_i \in C = \{c_0, \ldots, c_n\}$ do
    $Score(c_i) \leftarrow distanceMetric(TextVector, Vector(c_i))$
  Let $GeneratedConcepts$ be a set of $numConcepts$ concepts
    with highest $Score(c_i)$

  $Ancestors \leftarrow \emptyset$
  For each $c_j \in GeneratedConcepts$ do
    $Ancestors \leftarrow Ancestors \cup Parent^*(c_j)$

  Return $GeneratedConcepts \cup Ancestors$
```

Figure 3.8: Feature generation using a hierarchical ontology

3.6 Using Knowledge Repositories that Define Arbitrary Relations Between Concepts

Knowledge concepts can be subject to many other relations in addition to generalization. Examples of such relations include meronymy ("part of") and holonymy, synonymy, as well as more specific relations such as "capital of", "birthplace/birthdate of" etc. A notable example of a knowledge repository that features such relations is the Wikipedia encyclopedia, where relations are represented by hypertext links between Wikipedia articles.

As opposed to strict hierarchical organization built on the "is-a" relation, it makes little sense to use arbitrary relations to enrich the text pool associated with each concept, as explained in the previous section. However, we can still use these relations for feature construction. This way, whenever a text fragment is classified to a certain concept c_i, we consider generating features based on the concepts that stand in some relation to c_i. Since different relations might reflect different strength of connection between concepts, it might be necessary to quantify this strength in some way, in order to construct features based on concepts that are truly relevant to the input text.

Figure 3.9 presents the pseudocode of the feature generation algorithm extended for the case of arbitrary relations.

```
Algorithm FG($text, distanceMetric, numConcepts$)
  $TextVector \leftarrow tfidf(text)$
  For each $c_i \in C = \{c_0, \ldots, c_n\}$ do
    $Score(c_i) \leftarrow distanceMetric(TextVector, Vector(c_i))$
  Let $GeneratedConcepts$ be a set of $numConcepts$ concepts
    with highest $Score(c_i)$

  $Related \leftarrow \emptyset$
  For each $c_j \in GeneratedConcepts$ do
    For each $c_k$ such that $\exists r_l : \langle c_j, c_k \rangle \in r_l$ do
      If $Strength(\langle c_j, c_k \rangle) > threshold$ then
        $Related \leftarrow Related \cup \{c_k\}$

  Return $GeneratedConcepts \cup Related$
```

Figure 3.9: Feature generation with arbitrary relations

Chapter 4

Instantiation of the Feature Generation Methodology for the ODP and Wikipedia

In this Chapter we instantiate our feature generation methodology for two specific knowledge repositories—the Open Directory Project (ODP, 2006) and the Wikipedia encyclopedia (Wikipedia, 2006).

4.1 Using the Open Directory for Feature Generation

We now instantiate the general methodology presented in Section 3 to use the Open Directory project as a knowledge repository (Gabrilovich and Markovitch, 2005).

The Open Directory comprises a hierarchy of approximately 600,000 nodes that catalog over 4,000,000 Web sites, each represented by a URL, a title, and a brief summary of its contents. The directory is organized as a tree where each node has a title (defined by its location within the directory, e.g., COMPUTERS/ARTIFICIAL_INTELLIGENCE), and about one-third of all nodes have a short textual description. Every ODP node is associated with a collection of URLs to Web sites cataloged under that node, while each URL has a title and a concise summary of the corresponding Web site. The project constitutes an ongoing effort promoted by over 65,000 volunteer editors around the globe, and is arguably the largest publicly available Web directory.[1] Being the result of *pro bono* work,

[1] Although the actual size of Yahoo! has not been publicly released in the recent years, it is estimated to be about half the size of the Open Directory. This estimate is based on brute-force exhaustive crawling of the Yahoo! hierarchy. See

the Open Directory has its share of drawbacks, such as non-uniform coverage, duplicate subtrees in different branches of the hierarchy, and sometimes biased coverage due to peculiar views of the editors in charge. At the same time, however, ODP embeds a colossal amount of human knowledge in a wide variety of areas, covering even very specific scientific and technical concepts. Armed with this knowledge, the ODP-based feature generator constructs new features that denote ODP categories, and adds them to the bag of words. The augmented feature space provides text classifiers with a cornucopia of additional information.

4.1.1 Multiplying Knowledge Through Web Crawling

We can use the titles and summaries of the URLs as training examples for learning the feature generator. Although these descriptions alone constitute a sizeable amount of information, we devised a way to increase the volume of training data by several orders of magnitude. We do so by crawling the Web sites pointed at by all cataloged URLs, and obtain a small representative sample of each site. Following the scheme introduced by Yang, Slattery, and Ghani (2002), each link cataloged in the ODP is used to obtain a small representative sample of the target Web site. To this end, we crawl each cataloged site in the BFS order, starting from the URL listed in the directory. A predefined number of Web pages are downloaded, and then concatenated into a synthetic *meta-document*. This meta-document, along with the URL title and summary, constitutes the textual object for that site. Pooling together the meta-documents for all sites associated with an ODP node gives us a wealth of additional information about it.

4.1.2 Noise Reduction and Attribute Selection

Using so much knowledge requires a host of filtering mechanisms that control the quality and utility of the generated features. We now describe these mechanisms in detail. In what follows, we distinguish between *structural noise*, which is inherent to the ODP structure, and *content noise*, which is found in the texts we obtain through crawling the cataloged URLs.

Structural noise

However elaborate the Open Directory is, it necessarily contains concepts that are detrimental to feature generation. These include concepts too specific or situated too deep in the hierarchy, or having too few textual objects to build a representative attribute vector. It is important to observe, however, that whenever we

`http://sewatch.com/reports/directories.html` and `http://www.geniac.net/odp` for more details.

prune small concepts, we assign all of their textual content to their parents. Here again we benefit from the hierarchical organization of the directory, which allows us to aggregate small fragments of specific knowledge at a higher conceptual level, where its accumulated mass becomes sufficient to define a more general concept.

We identified the following potential sources of noise in the Open Directory:

1. The branch Top/World concentrates material in languages other than English. This entire branch is therefore pruned.

2. Some top-level branches contain concepts that are hardly useful for subsequent text categorization.

   (a) Top/News is a very elaborate subtree devoted to listing numerous CNN stories on various topics organized by date. The nodes of this subtree represent past dates, and do not correspond to useful knowledge concepts.

   (b) Top/Adult lists adult-oriented Web sites, and we believe that the concepts of this subtree are of little use for general purpose text categorization.

   (c) Top/Kids_and_Teens roughly duplicates the structure of the ODP but only lists resources suitable for children.

   All these branches are pruned as well.

3. Overly small categories (usually situated very deep in the hierarchy) that only contain a handful of URLs, and therefore their scope cannot be learned reliably. We therefore eliminate categories with fewer than 10 URLs or those situated below depth level 7 (the textual content of pruned categories is assigned to their parents).

4. The Top/Regional branch contains approximately one third of the entire mass of the ODP data, and is devoted to listing English language sites about various geographical regions of the world. This branch is further divided into continents, countries and smaller localities, up to the level of cities, towns and landmarks. However, the hierarchy does not stop at this level, and for most localities it provides much more elaborate classification, similar to that of the higher ODP levels. For example, under the path Top/Regional/North_America/United_States/New_York/Localities/N/New_York_City one finds further subdivisions such as Arts_and_Entertainment, Business_and_Economy, Health, Shopping and Society_and_Culture. A similar set of categories duplicating higher-level notions (Top/Arts, Top/Business etc.) can be also

found at the state level (i.e., at Top/Regional/North_America/United_States/New_York).

ODP classification principles[2] prescribe that businesses that operate in a particular locality (in this example, local to the State of New York or to New York City) should normally be catalogued under the most specific applicable categories, while businesses with global reach should be catalogued somewhere under Top/Business; the rationale for choosing other categories (e.g., Top/Society/... vs. Top/Regional/North_America/United_States/New_York/Society_and_Culture) is similar. However, we believe that when the ODP is used as a knowledge repository to support text categorization, such fine-grained distinctions (e.g., architect offices in Manhattan) are of little use. These categories only pollute the hierarchy with numerous small nodes, each of which only has a small chance of being assigned to any given context.

Therefore, we eliminate overly specific categories under Top/Regional by pruning all paths at the level of geographical names. When the feature generator operates on a context describing a particular New York business, it will map the latter to the New York City node, as well as to one or more appropriate nodes under Top/Business.

5. Web spam, which comes in the form of URLs that are hardly authoritative or representative of their host category, but are nonetheless included in the directory by a minority of unscrupulous editors. We do not explicitly address the problem of spam here, as it lies beyond the scope of our current study.

Content noise

Texts harvested from the WWW are quite different from clean passages in formal written English, and without adequate noise reduction crawled data may do more harm than good. To reduce content noise we perform attribute selection as explained in Section 3.3.1. For example, Table 4.1 shows the top 10 attributes selected for sample ODP concepts using information gain as the attribute selection criterion. As we can see, the attributes selected for all the sample concepts are very intuitive and plausible.

[2] See `http://dmoz.org/guidelines` and `http://dmoz.org/erz/index.html` for general ODP editorial guidelines, and `http://dmoz.org/Regional/faq.html` for Regional-specific issues.

[3] Many crawled Web pages under Top/Regional/Europe/Switzerland contain non-English material, therefore we observe words like “Schweiz” (German for Switzerland) and “der” (German masculine definite article), which survived stop words removal that is only performed for English.

ODP concept	**Top 10 selected attributes**
Top/Business/Financial_Services	finance, loan, mortgage, equity, insurance, lender, bank, investment, transaction, payment
Top/Computers/Artificial_Intelligence	neural, artificial, algorithm, intelligence, AAAI, Bayesian, probability, IEEE, cognitive, inference
Top/Health/Nutrition	nutrition, diet, nutrient, vitamin, dietary, cholesterol, carbohydrate, intake, protein, fat
Top/Home/Cooking	recipe, sauce, ingredient, soup, salad, casserole, stew, bake, butter, cook
Top/Recreation/Travel	travel, itinerary, trip, destination, cruise, hotel, tour, adventure, travelogue, departure
Top/Regional/Europe/Switzerland[3]	Switzerland, Swiss, Schweiz, und, Suiss, sie, CHF, der, Zurich, Geneva
Top/Science	science, research, scientific, biology, laboratory, analysis, university, theory, study, scientist
Top/Shopping/Gifts	gift, birthday, occasion, basket, card, shipping, baby, keepsake, order, wedding
Top/Society/History	war, history, military, army, civil, historian, soldier, troop, politics, century
Top/Sports/Golf	golf, golfer, tee, hole, fairway, tournament, championship, clubhouse, PGA, par

Table 4.1: Examples of attribute selection using information gain

Learning the Feature Generator

In our current implementation, the feature generator works as a centroid-based classifier (Han and Karypis, 2000), which represents each category as a centroid vector of the pool of textual objects associated with it.[4] Given a fragment of text supplied as input for feature generation, the classifier represents it as an attribute vector in the same space. It then compares this vector to those of all the concepts, and returns the desired number of best-matching ones. Attribute vectors are compared using the cosine metric (Zobel and Moffat, 1998); the value

[4] The centroid classifier offers a simple and efficient way for managing the multitude of concepts in the Open Directory; additional machine learning techniques suitable for learning the feature generator have been mentioned in Section 3.3.

of the metric is treated as the classification score. A number of top-scoring concepts are retained for each input text as generated features. The feature generator also performs *generalization* of these concepts, and constructs features from the classified concepts *per se* as well as their ancestors in the hierarchy.

4.1.3 Implementation Details

To evaluate the utility of knowledge-based feature generation, we implemented the proposed methodology using the Open Directory as a source of world knowledge. Throughout the experiments we used an ODP snapshot as of April 2004. Crawling of URLs cataloged in the Open Directory was performed over the period of April–August 2004. In what follows, we describe the implementation details and design choices of our system.

Constructing the Feature Generator

All ODP data is publicly available in machine-readable RDF format at `http://rdf.dmoz.org`. We used the file `structure.rdf.u8`, which defines the hierarchical structure of the directory, as well as provides category names and descriptions, and the file `content.rdf.u8`, which associates each category with a list of URLs, each having a title and a concise summary of the corresponding Web site. After pruning the Top/World branch, which contains non-English material, and Top/Adult branch, which lists adult-oriented Web sites, we obtained a collection of over 400,000 concepts and 2,800,000 URLs, organized in a very elaborate hierarchy with maximum depth of 13 levels and median depth of 7. Further pruning of too small and deep categories, as well as pruning of the Top/Regional subtree at the level of geographical names as explained in Section 4.1.2, reduced the number of concepts to 63,000 (the number of URLs was not reduced, since the entire URL population from pruned nodes is moved to their parents).

Titles and summaries of the URLs amounted to 436 Mb of text. In order to increase the amount of information available for training the feature generator, we further populated the ODP hierarchy by crawling all of its URLs, and taking the first 10 pages (in the BFS order) encountered at each site to create a representative meta-document of that site. As an additional noise removal step, we discarded meta-documents containing fewer than 5 distinct terms. This operation yielded 425 Gb worth of HTML files. After eliminating all the markup and truncating overly long files at 50 Kb, we ended up with 70 Gb of additional textual data. Compared to the original 436 Mb of text supplied with the hierarchy, we obtained over a 150-fold increase in the amount of data.

Applying our methodology to a knowledge repository of this scale required an enormous engineering effort. After tokenization and removal of stop words, numbers and mixed alphanumeric strings (e.g., “Win2k” or “4Sale”), we obtained 20,800,000 distinct terms. Further elimination of rare words (occurring in less than 5 documents) and applying the Porter stemming algorithm (Porter, 1980) resulted in a more manageable number of 2,900,000 distinct terms that were used to represent ODP nodes as attribute vectors. Up to 1000 most informative attributes were selected for each ODP node using the Document Frequency criterion (other commonly used feature selection techniques, such as Information Gain, χ^2 and Odds Ratio (Yang and Pedersen, 1997; Rogati and Yang, 2002; Mladenic, 1998a), yielded slightly inferior results in text categorization).

In order to speed up consequent classification of document contexts, we also built an *inverted index* that, given a word, provides a list of concepts that have it in their attribute vector (i.e., the word has been *selected* for this concept).

When assigning weights to individual entries in attribute vectors, we took into consideration the location of original word occurrences. For example, words that occurred in URL titles were assigned higher weight than those in the summaries. Words originating from the summaries or meta-documents corresponding to links prioritized[5] by the ODP editors were also assigned additional weight. We completely ignored node descriptions since these are only available for about 40% of the nodes, and even then the descriptions are rarely used to actually describe the corresponding concept; in many cases they just contain instructions to the editors or explain what kinds of sites should *not* be classified under the node.

Finally, the set of attribute vectors undergoes *tf.idf* weighting, and serves for building a centroid-based feature generator.

4.2 Using Wikipedia for Feature Generation

From time immemorial, the human race strived to organize its collective knowledge in a single literary work. From “Naturalis Historiae” by Pliny the Elder to the contemporary mammoth “Encyclopaedia Britannica”, encyclopedias have been major undertakings to systematically assemble all the knowledge available to the mankind.

Back in the early years of AI research, Buchanan and Feigenbaum (1982) formulated the *knowledge as power hypothesis*, which postulated that “The power of an intelligent program to perform its task well depends primarily on the quantity and quality of knowledge it has about that task.” Lenat et al. (1990) argued that without world knowledge computer programs are very *brittle*, and can only

[5] ODP editors can highlight especially prominent and important Web sites; sites marked as such appear at the top of category listings and are emphasized with an asterisk (in RDF data files, the corresponding links are marked up with a `<priority>` tag).

carry out tasks that have been fully foreseen by their designers.

When computer programs face tasks that require human-level intelligence, it is only natural to use an encyclopedia to endow the machine with the breadth of knowledge available to humans. There are, however, several obstacles on the way to using encyclopedic knowledge. First, such knowledge is available in textual form, and using it requires natural language understanding, a major problem in its own right. Furthermore, language understanding may not be enough, as texts written *for* humans normally assume the reader possesses a large amount of common-sense knowledge, which is omitted even from most detailed encyclopedia articles (Lenat, 1997). To address this situation, Lenat and his colleagues launched the CYC project, which aims to explicitly catalog the common sense knowledge of the humankind.

In this thesis we propose and evaluate a way to render text categorization systems with true encyclopedic knowledge, based on the largest encyclopedia in existence—Wikipedia.

Let us illustrate the importance of encyclopedic knowledge with a couple of examples. Given a very brief news title "Bernanke takes charge", a casual observer can infer little information from it. However, using the algorithm we developed for consulting Wikipedia, we find out the following relevant concepts: Ben Bernanke, Federal Reserve, Chairman of the Federal Reserve, Alan Greenspan (Bernanke's predecessor), Monetarism (an economic theory of money supply and central banking), inflation and deflation. As another example, consider the title "Apple patents a Tablet Mac". Unless the reader is well-versed in the hi-tech industry and gadgets, she will likely find it hard to predict the contents of the news item. Using Wikipedia, we identify the following related concepts: Mac OS (the Macintosh operating system) Laptop (the general name for portable computers, of which Tablet Mac is a specific example), Aqua (the GUI of Mac OS X), iPod (another prominent product by Apple), and Apple Newton (the name of Apple's early personal digital assistant).

Observe that documents manipulated by a text categorization system are given *in the same form* as the encyclopedic knowledge we intend to use—plain text. Therefore, we can use text similarity algorithms to automatically identify encyclopedia articles relevant to each document, and then leverage the knowledge gained from these articles in subsequent processing. It is this key observation that allows us to circumvent the obstacles we enumerated above, and use encyclopedia directly, without the need for deep language understanding or pre-cataloged common-sense knowledge. Also, it is essential to note that we do *not* use encyclopedia to simply increase the amount of the training data for text categorization; neither do we use it as a text corpus to collect word cooccurrence statistics. Rather, we use the knowledge distilled from the encyclopedia to enrich the representation of documents, so that a text categorizer is induced in the augmented, knowledge-rich feature space.

4.2.1 Wikipedia as a Knowledge Repository

What kind of knowledge repository should be used for feature generation? In the previous section, we assumed the external knowledge is available in the form of a generalization hierarchy, and used the Open Directory Project as an example. This method, however, had a number of drawbacks, which can be corrected by using Wikipedia.

First, requiring the knowledge repository to define an "is-a" hierarchy limits the choice of appropriate repositories. Moreover, hierarchical organization embodies only one particular relation between the nodes (generalization), while numerous other relations, such as relatedness or meronymy/holonymy, are ignored. Second, large-scale hierarchies tend to be extremely unbalanced, so that the relative size of some branches is disproportionately large or small due to peculiar views of the editors. Such phenomena are indeed common in the ODP. For example, the Top/Society branch is heavily dominated by one of its children—Religion and Spirituality; the Top/Science branch is dominated by its Biology child; a considerable fraction of the mass of Top/Recreation is concentrated in Pets. Finally, to learn the scope of every ODP concept, short URL summaries associated with the concepts were augmented by crawling the URLs themselves. This procedure allowed us to accumulate many gigabytes worth of textual data, but at a price, as texts obtained from the Web are often quite far from formal writing and plagued with noise. Crawling a typical Web site often brings auxiliary material that has little to do with the site theme, such as legal disclaimers, privacy statements, and help pages.

In this section we propose to perform feature generation using Wikipedia, which is currently the largest knowledge repository on the Web (Gabrilovich and Markovitch, 2006b). Wikipedia is available in dozens of languages, while its English version is the largest of all, containing 300+ million words in nearly one million articles contributed by over 160,000 volunteer editors. For the sake of comparison, the other well-known encyclopedia, Britannica, is about an order of magnitude smaller, with 44 million words in 65,000 articles (`http://store.britannica.com`, visited on February 10, 2006).

Compared to the ODP, Wikipedia possesses several advantageous properties. First, its articles are much cleaner than typical Web pages, and mostly qualify as standard written English. Although Wikipedia offers several orthogonal browsing interfaces, their structure is fairly shallow, and we propose to treat Wikipedia as having essentially no hierarchy. This way, mapping documents onto relevant Wikipedia concepts yields truly multi-faceted classification of the document text, and avoids the problem of unbalanced hierarchy branches. Moreover, by not requiring the knowledge repository to be hierarchically organized, our approach is suitable for new domains, for which no ontology is available. Finally, Wikipedia articles are heavily cross-linked, in a way reminiscent of linking on the Web. We believe that these links encode many interesting relations between the concepts, and constitute an important source of information in addition to the article texts. We explore using inter-article links in Section 4.2.3.

4.2.2 Feature Generator Design

Although Wikipedia has almost a million articles, not all of them are equally useful for feature generation. Some articles correspond to overly specific concepts (e.g., Metnal, the ninth level of the Mayan underworld), or are otherwise unlikely to be useful for subsequent text categorization (e.g., specific dates or a list of events in a particular year). Other articles are just too short, so we cannot reliably classify texts onto the corresponding concepts. We developed a set of simple heuristics for pruning the set of concepts, by discarding articles that have fewer than 100 non stop words or fewer than 5 incoming and outgoing links. We also discard articles that describe specific dates, as well as Wikipedia disambiguation pages.

The feature generator performs classification of texts onto Wikipedia concepts. Observe that input texts are given *in the same form* as Wikipedia articles, that is, in the form of plain text. Therefore, we can use conventional text classification algorithms (Sebastiani, 2002) to rank the concepts represented by these articles according to their relevance to the given text fragment. It is this key observation that allows us to use encyclopedia directly, without the need for deep language understanding or pre-cataloged common-sense knowledge.

However, this is a very peculiar classification problem with hundreds of thousands of classes, each having a single positive example—the article text. Conventional induction techniques can hardly be applied in these settings, so we opted to use a simple and efficient centroid classifier (Han and Karypis, 2000), which represents each concept with an attribute vector of the article text.

When using a centroid classifier, it is essential to perform attribute selection to reduce noise. However, since we only have a single article for each concept, standard attribute selection techniques cannot be applied, so we postpone noise control to the next step. Each concept is represented as an attribute vector, whose entries are assigned weights using a *tf.idf* scheme (Debole and Sebastiani, 2003). Then, we build an *inverted index* that maps each attribute into a list of concepts in which it appears. The primary purpose of inverted index is to speed up vector matching. In addition to that we use it to discard insignificant associations between attributes and concepts. This is done by removing those concepts whose weights for a given attribute are too low. This scheme allows us to circumvent the scarceness of text objects for each concept—we cast the problem of attribute selection per concept as *concept selection* per attribute.

4.2.3 Using the Link Structure

It is only natural for an electronic encyclopedia to provide cross-references in the form of hyperlinks. As a result, a typical Wikipedia article has many more links to other entries than articles in conventional printed encyclopedias.

This link structure can be used in several ways. Observe that each link is associated with an *anchor text* (clickable highlighted phrase). The anchor text is not always identical to the canonical name of the target article, and different anchor texts are used to refer to the same article in different contexts. For example, anchor texts pointing at Federal Reserve include "Fed", "U.S. Federal Reserve Board", "U.S. Federal Reserve System", "Board of Governors of the Federal Reserve", "Federal Reserve Bank", "foreign reserves" and "Free Banking Era". Thus, anchor texts provide alternative names, variant spellings, and related phrases for the target concept, which we use to enrich the article text for the target concept.

Similarly to the WWW, incoming links contribute to the significance of an article. Indeed, the highest number of incoming links—over 100,000—point at the article United States. We use the number of incoming links to express a slight preference for more significant concepts in feature generation, by multiplying the FG score of each concept by $log(log(number\ of\ incoming\ links))$.

Finally, inter-article links often reflect important relations between concepts that correspond to the linked articles. We evaluate the use of such relations for feature generation in the next section.

Inter-article Links as Concept Relations

As a rule, the presence of a link implies some relation between the concepts it connects. For example, the article on the United States links to Washington, D.C. (country capital) and North America (the continent where the country is situated). It also links to a multitude of other concepts, which are definitely related to the source concept, albeit it is more difficult to define those relations; examples include United States Declaration of Independence, President of the United States, and Elvis Presley.

Let us briefly recap the way we would like to use inter-concept relations for feature generation. Let ct be a text fragment, and let it be mapped by the feature generator to a sequence of concepts $C_{ct} = c_1, \ldots, c_p$. We would like to generate additional features for ct, based on concepts that stand in some relation to C_{ct}. When using Wikipedia, it is therefore logical to consider generating features based on concepts that are linked from the articles corresponding to the initially classified concepts, namely, C_{ct}. This way, we will generate features using the knowledge encoded in the links connecting the concepts.

However, our observations reveal that the existence of a link does not always imply the two articles are strongly related.[6] In fact, many words and phrases in a typical Wikipedia article link to other articles just because there are entries for the corresponding concepts. For example, the Education subsection in the article on the UNITED STATES has gratuitous links to concepts HIGH SCHOOL, COLLEGE, and LITERACY RATE.

Therefore, in order to use Wikipedia links for feature generation, it is essential to filter the linked concepts according to their relevance to the context. To this end, we examine the related concepts linked to those in C_{ct}, and retain those with highest scores for the original context ct. If a newly considered concept is linked to more than one concept in C_{ct}, its FG score is multiplied accordingly. Finally, the desired number of highest-scoring related concepts is retained to produce additional features. Figure 4.1 illustrates the proposed algorithm.

Concept generality filter

Recall that when using the Open Directory, we generated additional features that were *by definition* more general than the originally classified ones. Wikipedia provides numerous relations in addition to the simple "is-a", but are features constructed from them equally useful for text categorization?

Relevance of the newly constructed features is certainly important, but is not the only criterion. Suppose that we are given an input text "Google search". Which additional feature is likely to be more useful: NIGRITUDE ULTRAMARINE (a specially crafted meaningless phrase used in a search engine optimization contest) or WEBSITE? Now suppose the input is "artificial intelligence"—which feature is likely to contribute more to the representation of this input, JOHN MCCARTHY (COMPUTER SCIENTIST) or LOGIC? We believe that in both examples, the second feature would be more useful because it is not overly specific.

Consequently, we conjecture that in text categorization we should generate additional link-based features sparingly, taking only those features that are "more general" than those that triggered them. But how can we judge the generality of concepts? While this may be tricky to achieve in the general case (no pun intended), we propose the following task-oriented criterion. Given two concepts c_a and c_b, we compare the numbers of links pointing at them. Then, we say that c_a is "more general" than c_b if its number of incoming links is at least an order of magnitude larger, that is, if $log_{10}(\#inlinks(c_a)) - log_{10}(\#inlinks(c_b)) > 1$.

[6] The opposite is also true—the absence of a link may simply be due to an oversight. Adafre and de Rijke (2005) studied the problem of discovering missing links in Wikipedia.

```
Algorithm FG($ct, distanceMetric, numConcepts, numSearched,$
                    $numExamined, numRelated$)
  $TextVector \leftarrow tfidf(ct)$
  For each $c_i \in C = \{c_0, \ldots, c_n\}$ do
    $Score(c_i) \leftarrow distanceMetric(TextVector, Vector(c_i))$
  Let $Generated$ be a set of $numConcepts$ concepts with highest $Score(c_i)$
  Let $Searched$ be a set of $numSearched$ concepts with highest $Score(c_i)$
  Let $Examined$ be a set of $numExamined$ concepts with highest $Score(c_i)$

  Let $Links = \{\langle c_a, c_b \rangle\}$ be a set of links between Wikipedia concepts
  For each $c_k \in Searched$ do
    $RelWeight(c_k) \leftarrow 0$
  For each $c_j \in Examined$ do
    For each $c_k$ such that $\langle c_j, c_k \rangle \in Links$ do
      If $c_k \in Searched$ then
        $RelWeight(c_k) \leftarrow RelWeight(c_k) + Score(c_k)$

  Let $Related$ be a set of $numRelated$ concepts with highest $RelWeight(c_k)$
  Return $Generated \cup Related$
```

Figure 4.1: Feature generation with Wikipedia links as relations

Figure 4.2 illustrates the algorithm that only generates more general features. We use boldface font to highlight the difference from the previous version.

We show examples of additional features generated using inter-article links in Section 5.4.1. In Section 5.4.5 we report the results of using inter-article links for feature generation. In that section we also specifically examine the effect of constructing features from concepts that are more general than the concepts that triggered them.

4.2.4 Implementation Details

We used Wikipedia snapshot as of November 11, 2005. After parsing the Wikipedia XML dump, we obtained 1.8 Gb of text in 910,989 articles. Upon removing small and overly specific concepts that have fewer than 100 words, fewer than 5 incoming or outgoing links, category pages, disambiguation pages and the like, 171,332 articles were left that defined concepts used for feature generation. We processed the text of these articles by first tokenizing it, removing stop words and rare words (occurring in fewer than 3 articles), and stemmed the remaining

```
Algorithm FG($ct, distanceMetric, numConcepts, numSearched,$
             $numExamined, numRelated$)
  $TextVector \leftarrow tfidf(ct)$
  For each $c_i \in C = \{c_0, \ldots, c_n\}$ do
    $Score(c_i) \leftarrow distanceMetric(TextVector, Vector(c_i))$
  Let $Generated$ be a set of $numConcepts$ concepts with highest $Score(c_i)$
  Let $Searched$ be a set of $numSearched$ concepts with highest $Score(c_i)$
  Let $Examined$ be a set of $numExamined$ concepts with highest $Score(c_i)$

  Let $Links = \{\langle c_a, c_b \rangle\}$ be a set of links between Wikipedia concepts
  For each $c_k \in Searched$ do
    $RelWeight(c_k) \leftarrow 0$
  For each $c_j \in Examined$ do
    For each $c_k$ such that $\langle c_j, c_k \rangle \in Links$ do
      If $c_k \in Searched$ then
        If $\mathbf{\log_{10}(\#inlinks(c_k)) - \log_{10}(\#inlinks(c_j)) > 1}$ then
          $RelWeight(c_k) \leftarrow RelWeight(c_k) + Score(c_k)$

  Let $Related$ be a set of $numRelated$ concepts with highest $RelWeight(c_k)$
  Return $Generated \cup Related$
```

Figure 4.2: Feature generation with Wikipedia links as relations, where only more general features are constructed

words; this yielded 296,157 distinct terms, which were used to represent concepts as attribute vectors.

Preprocessing of Wikipedia XML dump

Wikipedia data is publicly available online at `http://download.wikimedia.org`. All the data is distributed in XML format, and several packaged versions are available: article texts, edit history, list of page titles, interlanguage links etc. In this project, we only use the article texts, but ignore the information on article authors and page modification history. Before building the feature generator, we perform a number of operations on the distributed XML dump:

- We simplify the original XML by removing all those fields that are not used in feature generation, such as author ids and last modification times.
- Wikipedia syntax defines a proprietary format for inter-article links, whereas the name of the article referred to is enclosed in brackets (e.g.,

"[United States]"). We map all articles to numeric ids, and for each article build a list of ids of the articles it refers to. We also count the number of incoming and outgoing links for each article.

- Wikipedia defines a *redirection* mechanism, which maps frequently used variant names of entities into canonical names. For examples, United States of America is mapped to United States. We resolve all such redirections during initial preprocessing.
- Another frequently used mechanism is *templates*, which allows articles to include frequently reused fragments of text without duplication, by including pre-defined and optionally parameterized templates on the fly. To speed up subsequent processing, we resolve all template inclusions at the beginning.
- We also collect all anchor texts that point at each article.

This preprocessing stage yields a new XML file, which is then used for building the feature generator.

Inverted Index Pruning

The algorithm for pruning the inverted index operates as follows. We first sort all the concepts for a given word according to their *tf.idf* weights in the decreasing order. We then scan the resulting sequence of concepts with a sliding window of length 100, and truncate the sequence when the difference in scores between the first and last concepts in the window drops below 5% of the highest-scoring concept for this word (which is positioned first in the sequence).

Chapter 5

Empirical Evaluation of Feature Generation for Text Categorization

In this chapter we evaluate the benefits of using external knowledge for text categorization.

5.1 Test Collections

We used the following test collections to evaluate our methodology.

5.1.1 Reuters-21578

This data set contains one year worth of English-language stories distributed over the Reuters newswire in 1986–1987, and is arguably the most often used test collection in text categorization research. Reuters-21578 is a cleaned version of the earlier release named Reuters-22173, which contained errors and duplicate documents.

The collection contains 21578 documents (hence the name) in SGML format. Of those, 12902 documents are *categorized*, i.e., assigned a category label or marked as not belonging to any category. Other documents do not have an explicit classification; that is, they can reasonably belong to some categories (judged by their content), but are not marked so. Several train/test splits of the collection has been defined, of which ModApte (Modified Apte) is the most commonly used one. The ModApte split divides the collection chronologically, and allocates the first 9603 documents for training, and the rest 3299 documents for testing.

The documents are labeled with 118 categories; there are 0–16 labels per document, with the average of 1.04. The category distribution is extremely skewed:

the largest category ("earn") has 3964 positive examples, while 16 categories have only one positive example. Several category sets were defined for this collection:

- 10 largest categories ("earn", "acq", "money-fx", "grain", "crude", "trade", "interest", "ship", "wheat", "corn").
- 90 categories with at least one document in the training set and one in the testing set (Yang, 2001).
- Galavotti, Sebastiani, and Simi (2000) used a set of 115 categories with at least one training example (three categories, "cottonseed", "f-cattle" and "sfr" have no training examples under the ModApte split).
- The full set of 118 categories with at least one positive example either in the training or in the testing set.

Following common practice, we used the ModApte split and two category sets, 10 largest categories and 90 categories with at least one training and testing example.

5.1.2 20 Newsgroups (20NG)

The 20 Newsgroups collection (Lang, 1995) is comprised of 19997 postings to 20 Usenet newsgroups. Most documents have a single label, defined as the name of the newsgroup it was sent to; about 4% of documents have been *cross-posted*, and hence have several labels. Each newsgroup contains exactly 1000 positive examples, with the exception of "soc.religion.christian" which contains 997 documents.

Some categories are quite close in scope, for example, "comp.sys.ibm.pc.-hardware" and "comp.sys.mac.hardware", or "talk.religion.misc" and "soc.religion.christian". A document posted to a single newsgroup may be reasonably considered appropriate for other groups too (the author may have simply not known of other similar groups, and thus not cross-posted the message); this naturally poses additional difficulty for classification.

It should be noted that Internet news postings are very informal, and therefore the documents frequently contain non-standard and abbreviated words, foreign words, and proper names, as well as a large amount of markup characters (used for attribution of authorship or for message separation).

5.1.3 Movie Reviews

The *Movie Reviews* collection (Pang, Lee, and Vaithyanathan, 2002) represents a slightly different classification task than standard text categorization, referred to as *sentiment classification.* The collection contains 1400 reviews of movies, half of which express positive *sentiment* (opinion) about the movie, and half negative. The reviews were collected from the "rec.arts.movies.reviews" newsgroup, archived at the Internet Movie Database (IMDB, `http://www.imdb.com`). The classification problem in this case is to determine the *semantic orientation* of the document, rather than to relate its content to one of the predefined topics. This problem is arguably more difficult than topical text categorization, since the notion of semantic orientation is quite general. We saw this collection as an opportunity to apply feature generation techniques to this new task.

Recent works on semantic orientation include (Turney and Littman, 2002; Turney, 2002; Pang, Lee, and Vaithyanathan, 2002).[1] The two former studies used unsupervised learning techniques based on latent semantic indexing, estimating semantic distance between a given document and two reference words that represent polar opinions, namely, "excellent" and "poor". The latter work used classical TC techniques.

5.1.4 Reuters Corpus Version 1 (RCV1)

RCV1 is the newest corpus released by Reuters (Lewis et al., 2004; Rose, Stevenson, and Whitehead, 2002). It is considerably larger than its predecessor, and contains over 800,000 news items, dated between August 20, 1996 and August 19, 1997. The stories are labeled with 3 category sets, *Topics*, *Industries* and *Regions*.

- *Topics* are most close in nature to the category set of the old Reuters collection (Reuters-21578). There are 103 topic codes, with 3.24 categories per document on the average. The topics are organized in a hierarchy, and the *Hierarchy Policy* required that if a category is assigned to a document, all its ancestors in the hierarchy should be assigned as well. As a result, as many as 36% of all Topic assignments are due to the four most general categories, *CCAT*, *ECAT*, *GCAT*, and *MCAT*. Consequently, the *micro-averaged* performance scores are dominated by these categories (Lewis et

[1] The field of *genre* classification, which attempts to establish the *genre* of document, is somewhat related to sentiment classification. Examples of possible genres are radio news transcripts and classified advertisements. The work by Dewdney, VanEss-Dykema, and MacMillan (2001) cast this problem as text categorization, using *presentation features* in addition to words. Their presentation features included part of speech tags and verb tenses, as well as mean and variance statistics of sentence and word length, punctuation usage, and the amount of whitespace characters. Using support vector machines for actual classification, the authors found that the performance due to the presentation features alone was at least as good as that achieved with plain words, and that the combined feature set usually resulted in an improvement of several percentage points.

al., 2004), and *macro-averaging* becomes of interest.[2] The *Minimum Code Policy* required that each document was assigned at least one Topic and one Region code.

- *Industries* are more fine-grained than Topics, and are therefore harder for classification. These categories are also organized in a hierarchy, although the *Hierarchy Policy* was only partially enforced for them.

- *Region* codes correspond to geographical places, and are further subdivided into countries, regional groupings and economic groupings. Lewis et al. (2004) argue that Region codes might be more suitable for *named entity recognition* than for text categorization.

As noted by Lewis et al. (2004), the original RCV1 distribution contains a number of errors; in particular, there are documents that do not conform to either *Minimum Code* or *Hierarchy Policy*, or labeled with erratic codes. Lewis et al. (2004) proposed a procedure to correct these errors, and defined a new version of the collection, named *RCV1-v2* (as opposed to the original distribution, referred to as *RCV1-v1*). All our experiments are based on *RCV1-v2*.

In our experiments we used Topic and Industry categories. Due to the sheer size of the collection, processing all the categories in each set would take unreasonably long, allowing us to conduct only few experiments. Following the scheme introduced by Brank et al. (2002), we used 16 Topic and 16 Industry categories, which constitute a representative sample of the full groups of 103 and 354 categories, respectively. We also randomly sampled the Topic and Industry categories into 5 sets of 10 categories each. Table 5.1 gives the full definition of the category sets we used. To further speed up experimentation, we used a subset of the corpus with 17,808 training documents (dated August 20–27, 1996) and 5341 testing documents (dated August 28–31, 1996).

5.1.5 OHSUMED

OHSUMED (Hersh et al., 1994) is a subset of the MEDLINE database, which contains 348,566 references to documents published in medical journals over the period of 1987–1991. Each reference contains the publication title, and about two-thirds (233,445) also contain an abstract. Each document is labeled with several MeSH categories (MeSH, 2003). There are over 14,000 distinct categories in the collection, with an average of 13 categories per document. OHSUMED is frequently used in information retrieval and text categorization research.

[2] This is why micro-averaged scores for Topic codes are so much higher than macro-averaged ones, see Section 5.2.2.

Set name	Categories comprising the set
Topic-16	e142, gobit, e132, c313, e121, godd, ghea, e13, c183, m143, gspo, c13, e21, gpol, m14, c15
Topic-10A	e31, c41, c151, c313, c31, m13, ecat, c14, c331, c33
Topic-10B	m132, c173, g157, gwea, grel, c152, e311, c21, e211, c16
Topic-10C	c34, c13, gtour, c311, g155, gdef, e21, genv, e131, c17
Topic-10D	c23, c411, e13, gdis, c12, c181, gpro, c15, g15, c22
Topic-10E	c172, e513, e12, ghea, c183, gdip, m143, gcrim, e11, gvio
Industry-16	i81402, i79020, i75000, i25700, i83100, i16100, i1300003, i14000, i3302021, i8150206, i0100132, i65600, i3302003, i8150103, i3640010, i9741102
Industry-10A	i47500, i5010022, i3302021, i46000, i42400, i45100, i32000, i81401, i24200, i77002
Industry-10B	i25670, i61000, i81403, i34350, i1610109, i65600, i3302020, i25700, i47510, i9741110
Industry-10C	i25800, i41100, i42800, i16000, i24800, i02000, i34430, i36101, i24300, i83100
Industry-10D	i1610107, i97400, i64800, i0100223, i48300, i81502, i34400, i82000, i42700, i81402
Industry-10E	i33020, i82003, i34100, i66500, i1300014, i34531, i16100, i22450, i22100, i42900

Table 5.1: Definition of RCV1 category sets used in the experiments

Following Joachims (1998), we used a subset of documents from 1991 that have abstracts, taking the first 10,000 documents for training and the next 10,000 for testing. To limit the number of categories for the experiments, we randomly generated 5 sets of 10 categories each. Table 5.2 gives the full definition of the category sets we used.

5.1.6 Short Documents

We conjectured that knowledge-based feature generation should be particularly beneficial for categorization of short documents. To verify this conjecture, we derived several datasets of short documents from the test collections described above. Recall that about one-third of OHSUMED documents have titles but no abstract, and can therefore be considered short documents "as-is." We used the same range of documents as defined in Section 5.1.5, but considered only those without abstracts; this yielded 4,714 training and 5,404 testing documents. For all other datasets, we created a short document from each original document by taking only the title of the latter (with the exception of Movie Reviews, where

Set name	**Categories comprising the set** (parentheses contain MeSH identifiers)
OHSUMED-10A	B-Lymphocytes (D001402); Metabolism, Inborn Errors (D008661); Creatinine (D003404); Hypersensitivity (D006967); Bone Diseases, Metabolic (D001851); Fungi (D005658); New England (D009511); Biliary Tract (D001659); Forecasting (D005544); Radiation (D011827)
OHSUMED-10B	Thymus Gland (D013950); Insurance (D007341); Historical Geographic Locations (D017516); Leukocytes (D007962); Hemodynamics (D006439); Depression (D003863); Clinical Competence (D002983); Anti-Inflammatory Agents, Non-Steroidal (D000894); Cytophotometry (D003592); Hydroxy Acids (D006880)
OHSUMED-10C	Endothelium, Vascular (D004730); Contraceptives, Oral, Hormonal (D003278); Acquired Immunodeficiency Syndrome (D000163); Gram-Positive Bacteria (D006094); Diarrhea (D003967); Embolism and Thrombosis (D016769); Health Behavior (D015438); Molecular Probes (D015335); Bone Diseases, Developmental (D001848); Referral and Consultation (D012017)
OHSUMED-10D	Antineoplastic and Immunosuppressive Agents (D000973); Receptors, Antigen, T-Cell (D011948); Government (D006076); Arthritis, Rheumatoid (D001172); Animal Structures (D000825); Bandages (D001458); Italy (D007558); Investigative Techniques (D008919); Physical Sciences (D010811); Anthropology (D000883)
OHSUMED-10E	HTLV-BLV Infections (D006800); Hemoglobinopathies (D006453); Vulvar Diseases (D014845); Polycyclic Hydrocarbons, Aromatic (D011084); Age Factors (D000367); Philosophy, Medical (D010686); Antigens, CD4 (D015704); Computing Methodologies (D003205); Islets of Langerhans (D007515); Regeneration (D012038)

Table 5.2: Definition of OHSUMED category sets used in the experiments

documents have no titles).

It should be noted, however, that substituting a title for the full document is a poor man's way to obtain a collection of classified short documents. When documents were originally labeled with categories, the human labeller saw each document *in its entirety.* In particular, a category might have been assigned to a document on the basis of facts mentioned in its body, even though the information may well be missing from the (short) title. Thus, taking all the categories of the original documents to be "genuine" categories of the title is often misleading. However, because we know of no publicly available test collections of short documents, we decided to construct datasets as explained above. Importantly, OHSUMED documents without abstracts have been classified as such by humans; working with the OHSUMED-derived dataset can thus be considered a "pure" experiment.

5.1.7 Automatic Acquisition of Data Sets

Although numerous works studied text categorization in the past, good test collections are by far less abundant. In part, this scarcity can be attributed to the huge manual effort required to collect a sufficiently large body of text, categorize it, and ultimately produce in a machine-readable format (usually XML or SGML). Most works use the Reuters-21578 collection (Reuters, 1997) as the primary benchmark. Others use 20 Newsgroups (Lang, 1995) and OHSUMED (Hersh et al., 1994), while TREC[3] filtering experiments often use the data from the TIPSTER corpus.

Although the Reuters corpus became a standard reference in the field, it has a number of significant shortcomings. According to Dumais and Chen (2000), "the Reuters collection is small and very well organized compared with many realistic applications". Scott (1998) also notes that the Reuters corpus has a very restricted vocabulary, since Reuters in-house style prescribes using uniform unambiguous terminology to facilitate quick comprehension. As a consequence, good classifiers (e.g., SVM or KNN) yield very reasonable performance even using a simple bag-of-words approach, without the need for more elaborate features.

Mainly due to these achievements in Reuters classification, Sebastiani (2002) notes that "[automated TC] has reached effectiveness levels comparable to those of trained professionals ... and, more importantly, it is unlikely to be improved substantially by the progress of research". While this argument might be appropriate for the Reuters-21578 corpus, we believe it does not apply to the general case. For example, the state of the art performance on the OHSUMED collection

[3] Text REtrieval Conferences administered by the U.S. National Institute of Science and Technology (NIST).

is only around 50–60% (Yang, 2001; Yang, 1999). We believe that the performance of TC on more representative real-life corpora still has way to go. The recently introduced new Reuters corpus (RCV1), which features very large size and three orthogonal label sets definitely constitutes a new challenge. At the same time, acquisition of additional corpora suitable for TC research remains a major challenge.

To this end, as a part of this research we developed a methodology for automatic acquisition of labeled datasets for text categorization. This methodology allows one to define a set of parameters in order to generate datasets with desired properties, based on the Open Directory. We present and evaluate this methodology in Appendix B. It is essential to note that since these datasets have been derived from the Open Directory, we cannot use them to test the effect of using the ODP for feature generation. Indeed, we did not use these datasets to evaluate our feature generation methodology. In Appendix A we used these datasets in a study of feature selection.

5.2 Experimentation Procedure

We used support vector machines[4] as our learning algorithm to build text categorizers, since prior studies found SVMs to have the best performance for text categorization (Sebastiani, 2002; Dumais et al., 1998; Yang and Liu, 1999). Following established practice, we use the precision-recall break-even point (BEP) to measure text categorization performance. For the two Reuters datasets and OHSUMED we report both micro- and macro-averaged BEP, since their categories differ in size significantly. Micro-averaged BEP operates at the document level and is primarily affected by categorization performance on larger categories. On the other hand, macro-averaged BEP averages results for individual categories, and thus small categories with few training examples have large impact on the overall performance.

For both Reuters datasets (Reuters-21578 and RCV1) and OHSUMED we used a fixed train/test split as defined in Section 5.1, and consequently used macro sign test (S-test) (Yang and Liu, 1999) to assess the statistical significance of differences in classifier performance. For 20NG and Movies we performed 4-fold cross-validation, and used paired t-test to assess the significance. We also used the Wilcoxon signed-ranks test (Demsar, 2006) to compare the baseline and the FG-based classifiers over multiple data sets.

[4] We used the SVM^{light} implementation (Joachims, 1999a).

5.2.1 Text Categorization Infrastructure

We conducted the experiments using a text categorization platform of our own design and development named $\mathcal{H}$OGWARTS[5] (Davidov, Gabrilovich, and Markovitch, 2004). We opted to build a comprehensive new infrastructure for text categorization, as surprisingly few software tools are publicly available for researchers, while those that are available allow only limited control over their operation. $\mathcal{H}$OGWARTS facilitates full-cycle text categorization including text preprocessing, feature extraction, construction, selection and valuation, followed by actual classification. The system currently provides XML parsing, part-of-speech tagging (Brill, 1995), sentence boundary detection, stemming (Porter, 1980), WordNet (Fellbaum, 1998) lookup, a variety of feature selection algorithms, and *tf.idf* feature weighting schemes. $\mathcal{H}$OGWARTS has over 250 configurable parameters that control its *modus operandi* in minute detail. $\mathcal{H}$OGWARTS interfaces with SVM, KNN and C4.5 text categorization algorithms, and computes all standard measures of categorization performance. $\mathcal{H}$OGWARTS was designed with a particular emphasis on processing efficiency, and portably implemented in the ANSI C++ programming language and C++ Standard Template Library. The system has built-in loaders for Reuters-21578 (Reuters, 1997), RCV1 (Lewis et al., 2004), 20 Newsgroups (Lang, 1995), Movie Reviews (Pang, Lee, and Vaithyanathan, 2002), and OHSUMED (Hersh et al., 1994), while additional datasets can be easily integrated in a modular way.

Each document undergoes the following processing steps. Document text is first tokenized, and title words are replicated twice to emphasize their importance. Then, stop words, numbers and mixed alphanumeric strings are removed, and the remaining words are stemmed. The bag of words is next merged with the set of features generated for the document by analyzing its contexts as explained in Section 3.4, and rare features occurring in fewer than 3 documents are removed.

Since earlier studies found that most BOW features are indeed useful for SVM text categorization (Section 3.4.2), we take the bag of words in its entirety (with the exception of rare features removed in the previous step). The generated features, however, undergo feature selection using the information gain criterion. Finally, feature valuation is performed using the "ltc" *tf.idf* function (logarithmic term frequency and inverse document frequency, followed by cosine normalization) (Salton and Buckley, 1988; Debole and Sebastiani, 2003).

[5] *Hogwarts School of Witchcraft and Wizardry* is the educational institution attended by Harry Potter (Rowling, 1997).

5.2.2 Baseline Performance of $\mathcal{H}$OGWARTS

We now demonstrate that the performance of basic text categorization in our implementation (column "Baseline" in Table 5.3) is consistent with the state of the art as reflected in other published studies (all using SVM). On Reuters-21578, Dumais et al. (1998) achieved micro-BEP of 0.920 for 10 categories and 0.870 for all categories. On 20NG, Bekkerman (2003) obtained BEP of 0.856.[6] Pang, Lee, and Vaithyanathan (2002) obtained accuracy of 0.829 on Movies. The minor variations in performance are due to differences in data preprocessing in the different systems; for example, for the Movies dataset we worked with raw HTML files rather than with the official tokenized version, in order to recover sentence and paragraph structure for contextual analysis. For RCV1 and OHSUMED, direct comparison with published results is more difficult because we limited the category sets and the date span of documents to speed up experimentation.

5.2.3 Using the Feature Generator

We used the *multi-resolution* approach to feature generation, classifying document contexts at the level of individual words, complete sentences, paragraphs, and finally the entire document.[7] For each context, features were generated from the 10 best-matching concepts produced by the feature generator, as well as for all of their ancestors (in the case of the ODP-based FG).

5.3 ODP-based Feature Generation

5.3.1 Qualitative Analysis of Feature Generation

We now study the process of feature generation on a number of actual examples.

Feature Generation per se

In this section we demonstrate ODP-based feature generation for a number of sample sentences taken from CNN and other Web sites. For each example, we discuss a number of highly relevant features found among the top ten generated ones.

[6] Using distributional clustering of words, Bekkerman et al. (2003) obtained BEP of 0.886 on this dataset in the multi-labeled setting.

[7] The 20NG dataset is an exception, owing to its high level of intrinsic noise that renders identification of sentence boundaries extremely unreliable, and causes word-level feature generation to produce too many spurious classifications. Consequently, for this dataset we restrict the multi-resolution approach to individual paragraphs and the entire document only.

Dataset	Baseline	
	micro BEP	macro BEP
Reuters-21578		
10 categories	0.925	0.874
90 categories	0.877	0.602
RCV1		
Industry-16	0.642	0.595
Industry-10A	0.421	0.335
Industry-10B	0.489	0.528
Industry-10C	0.443	0.414
Industry-10D	0.587	0.466
Industry-10E	0.648	0.605
Topic-16	0.836	0.591
Topic-10A	0.796	0.587
Topic-10B	0.716	0.618
Topic-10C	0.687	0.604
Topic-10D	0.829	0.673
Topic-10E	0.758	0.742
OHSUMED		
OHSUMED-10A	0.518	0.417
OHSUMED-10B	0.656	0.500
OHSUMED-10C	0.539	0.505
OHSUMED-10D	0.683	0.515
OHSUMED-10E	0.442	0.542
20NG	0.854	
Movies	0.813	

Table 5.3: Baseline performance of $\mathcal{H}$OGWARTS text categorization platform

- **Text:** *"Rumsfeld appeared with Gen. Richard Myers, chairman of the Joint Chiefs of Staff."*

  Sample generated features:

  – SOCIETY/ISSUES/GOVERNMENT_OPERATIONS, SOCIETY/POLITICS — both Donald Rumsfeld and Richard Myers are senior government officers, hence the connection to government operations and politics. Their names have been selected for these ODP concepts, since they appear in many Web sites cataloged under them, such as the National Security Archive at the George Washington University (`http://www.gwu.edu/~nsarchiv`) and the John F. Kennedy School of Government at Harvard University (`http://www.ksg.harvard.edu`).

  - Society/Issues/Warfare_and_Conflict/Specific_Conflicts/Iraq, Science/Technology/Military_Science, Society/Issues/Warfare_and_Conflict/Weapons—again, both persons mentioned were prominent during the Iraq campaign.
  - Society/History/By_Region/North_America/United_States/Presidents/Bush,_George_Walker — Donald Rumsfeld serves as Secretary of Defense under President George W. Bush.
  - Society/Politics/Conservatism — Rumsfeld is often seen as holding conservative views on a variety of political issues.

- **Text:** *"The new film follows Anakin's descent into evil and lust for power."*

  Sample generated features:

  - Arts/Movies/Titles/Star_Wars_Movies is the root of the ODP subtree devoted to the "Star Wars" movie series. The word "Anakin" has been selected as an attribute for this concept due to its numerous occurrences in the cataloged Web sites such as `http://www.theforce.net` and `http://www.starwars.com`.
  - Arts/Performing_Arts/Acting/Actors_and_Actresses/Christensen,_Hayden is the actor who played Anakin Skywalker; this particular piece of information cannot be inferred from the short input sentence without elaborate background knowledge.

- **Text:** *"On a night when Dirk Nowitzki (34 points), Jerry Stackhouse (29), Josh Howard (19) and Jason Terry (17) all came up big, he couldn't match their offensive contributions."*

  Sample generated features:

  - Sports/Basketball/Professional/NBA/Dallas_Mavericks— even though the sentence mentions neither the particular sport nor the name of the team, the power of context is at its best, immediately yielding the correct classification as the best-scoring generated feature. The names of the players mentioned in the context occur often in the Web sites cataloged under this concept, including such resources as `http://www.nba.com/mavericks`, `http://dallasbasketball.com`, and `http://sports.yahoo.com/nba/teams/dal`.

- **Text:** *"Herceptin is a so-called targeted therapy because of its ability to attack diseased cells and leave healthy ones alone."*

Sample generated features:

- Health/Conditions_and_Diseases/Cancer/Breast, Society/Issues/Health/Conditions_and_Diseases/Cancer/Alternative_Treatments, Health/Support_Groups/Conditions_and_Diseases/Cancer provide relevant additional information for Herceptin, a medication for breast cancer. The name of this medicine has been selected for these concepts due to its occurrences in cataloged Web sites such as `www.breastcancer.org`, `www.hopkinsmedicine.org/breastcenter` and `cancer.gov/cancerinfo/wyntk/breast`.

- Finally, we give an example of how the power of context can be used for word sense disambiguation. The following pair of sentences use the word "tie" in two different meanings—once as a necktie and once as a kind of connection. Even though these sentences contain no distinguishing proper names, the context of the polysemous words allows the feature generator to produce correct suggestions in both cases

  Text: *"Kinship with others is based either on blood **ties** or on marital **ties**."*

  Sample generated features:

  - Society/Genealogy
  - Home/Family
  - Society/Relationships
  - Science/Social_Sciences/Sociology

  Text: *"Our **tie** shop includes plain solid colour **ties**, novelty **ties**, patterned silk **ties**, and men's bow **ties**."*

  Sample generated features:

  - Shopping/Clothing/Men's/Neckties
  - Shopping/Clothing/Accessories/Men's
  - Business/Consumer_Goods_and_Services/Clothing/Accessories/Ties_and_Scarves

Evidently, many of the generated features could not have been accessed by conventional text classification methods, since heavy use of world knowledge is required to deduce them.

Actual Text Categorization Examples Under a Magnifying Glass

Thanks to feature generation, our system correctly classifies the running example document #15264. Let us consider additional testing examples from Reuters-21578 that are incorrectly categorized by the BOW classifier. Document #16143 belongs to the category "money-fx" (money/foreign exchange) and discusses the devaluation of the Kenyan shilling. Even though "money-fx" is one of the 10 largest categories, the word "shilling" does not occur in its training documents even once. However, the feature generator easily recognizes it as a kind of currency, and produces features such as Recreation/Collecting/Paper_Money and Recreation/Collecting/Coins/World_Coins. While analyzing document contexts it also uses other words such as "Central Bank of Kenya" and "devaluation" to correctly map the document to ODP concepts Society/Government/Finance, Science/Social_Sciences/Economics and Business/Financial_Services/Banking_Services. Even though the behavior of the Kenyan shilling was never mentioned in the training set, these high-level features were also constructed for many training examples, and consequently the document is now classified correctly.

Similarly, document #18748 discusses Italy's balance of payments and belongs to the category "trade" (interpreted as an economic indicator), while the word "trade" itself does not occur in this short document. However, when the feature generator considers document contexts discussing Italian deficit as reported by the Bank of Italy, it correctly maps them to concepts such as Society/Government/Finance, Society/Issues/Economic/International/Trade, Business/International_Business_and_Trade. These features, which were also generated for training documents in this category (notably, document #271 on Japanese trade surplus, document #312 on South Korea's account surplus, document #354 on tariff cuts in Taiwan and document #718 on U.S.-Canada trade pact), allow the document to be categorized correctly.

Let us also consider a few documents from the Movie Reviews dataset that confuse the BOW classifier (here we consider a training/testing split induced by one particular cross-validation fold). Recall that this dataset represents a sentiment classification task, where documents are classified according to the sentiment of the review (positive or negative) rather than its topic. Document #19488 contains a negative review of Star Wars Episode 1, but at the word level it is difficult to judge its true sentiment since positive and negative words are interspersed. For instance, the sentence "Anakin is annoying and unlikeable, instead of cute and huggable as Lucas no doubt intended" contains two words with positive connotation ("cute and huggable") that counterbalance the two words with negative ones ("annoying and unlikeable"). However, given contexts like "The two leads are hideously boring, static characters given little to do and too much time to do it," the feature generator produces features such as Arts/Movies/Reviews/Top_Lists/Bad_Films. This ODP node catalogs Web sites devoted to reviews of bad movies, and the wording of this sample context looks similar to that used in known negative reviews (as cataloged in the ODP). In fact, this particular feature is one of the most informative ones generated for this dataset, and it is also produced for contexts like "Next up we have the dialogue, which is amusingly bad at its best, painful at its worst" and "What ensues is a badly scripted and horribly directed 114 minutes of cinema hell," both found in negative reviews.

As another example, consider document #15111, which contains a positive review of the movie "Soldier." This review, which constantly switches between criticizing and praising the film, easily perplexes the BOW classifier. Interestingly, given the sentence "It is written by David Webb Peoples, who penned the screenplay to the classic Blade Runner and the critically-acclaimed 12 Monkeys," the feature generator constructs the highly informative feature Arts/Movies/Reviews/Top_Lists/Good_Films. This is made possible by the references to known good films ("Blade Runner" and "12 Monkeys") that are listed in Web sites devoted to good films (`http://www.filmsite.org` and `http://us.imdb.com/top_250_films`, for example). The same feature was also generated for a number of training documents, and thus helps the classifier to categorize the document correctly.

The Importance of Feature Selection

To understand the utility of feature selection, consider a sample sentence from our running example, Reuters document #15264: "Cominco's share of production was 43,000 short tons of copper, 340,000 ounces of silver and 800 ounces of gold." Table 5.4 gives the top ten ODP concepts generated as features for this context. Most of the assigned concepts deal with mining and drilling, and will eventually be useful features for document classification. However, the concepts Business/Investing/Commodities,_Futures/Precious_Metals, Shopping and Business/Investing/Commodities,_Futures/Precious_Metals/Gold have been triggered by the words "gold" and "silver," which are mentioned incidentally and do not describe the gist of the document. Feature selection is therefore needed to eliminate features based on these extraneous concepts.

As another example, consider the following sentence taken from the same document: "'Cominco, 29.5 percent owned by a consortium led by Teck, is optimistic that the talks will soon be concluded,' spokesman Don Townson told Reuters," along with its top ten classifications given in Table 5.5. Here, the concept Society/Issues is triggered by the word "Reuters." In turn, the concept Business/Marketing_and_Advertising/Consulting/Sales is triggered by the

#	ODP concept
1	Business/Mining_and_Drilling/Mineral_Exploration_and_Extraction
2	Business/Mining_and_Drilling
3	Business/Mining_and_Drilling/Mineral_Exploration_and_Extraction/ Base_Metals
4	Science/Technology/Mining
5	Business/Mining_and_Drilling/Consulting
6	Business/Investing/Commodities,_Futures/Precious_Metals
7	Shopping
8	Business/Mining_and_Drilling/Mining_Equipment
9	Business/Investing/Commodities,_Futures/Precious_Metals/Gold
10	Science/Technology/Mining/Investments

Table 5.4: The top ten ODP concepts generated for the sentence "Cominco's share of production was 43,000 short tons of copper, 340,000 ounces of silver and 800 ounces of gold."

name of the company spokesman, Don Townson. As it happens, a sales consulting company named "Townson & Alexander Consulting Services" is catalogued under this concept. Based on the crawled content of this site, the word "Townson" and other sales-related words in the context (e.g., "percent," "owned," "optimistic," and "consortium") taken together yield this concept in the results. Again, this sales-related concept is hardly useful for categorizing copper-related documents, and features based on it would therefore not be selected.

5.3.2 The Effect of Feature Generation

Table 5.6 shows the results of using feature generation for text categorization, with significant improvements ($p < 0.05$) shown in bold. We consistently observed larger improvements in macro-averaged BEP, which is dominated by categorization effectiveness on small categories. This goes in line with our expectations that the contribution of external knowledge should be especially prominent for categories with few training examples. As can be readily seen, categorization performance was improved for all datasets, with notably high improvements for Reuters RCV1, OHSUMED and Movies. Given the performance plateau currently reached by the best text categorizers, these results clearly demonstrate the advantage of knowledge-based feature generation.

#	ODP concept
1	Business/Mining_and_Drilling/Mineral_Exploration_and_Extraction/ Base_Metals
2	Business/Mining_and_Drilling/Mineral_Exploration_and_Extraction
3	Business/Mining_and_Drilling
4	Business/Mining_and_Drilling/Consulting
5	Society/Issues
6	Regional/North_America/Canada/British_Columbia/Localities/ Kimberley
7	Science/Technology/Mining
8	Business/Marketing_and_Advertising/Consulting/Sales
9	Regional/North_America/Canada/Quebec/Regions/Northern_Quebec
10	Science/Environment/Mining

Table 5.5: The top ten ODP concepts generated for the sentence "'Cominco, 29.5 percent owned by a consortium led by Teck, is optimistic that the talks will soon be concluded,' spokesman Don Townson told Reuters."

5.3.3 The Effect of Contextual Analysis

We now explore the various possibilities for defining document contexts for feature generation, i.e., chunks of document text that are classified onto the ODP to construct features. Figure 5.1 shows how text categorization performance on the Movies dataset changes for various contexts. The x-axis measures context length in words, and the *FG/words* curve corresponds to applying the feature generator to the context of that size. With these word-level contexts, maximum performance is achieved when using pairs of words (x=2). The *Baseline* line represents text categorization without feature generation. The *FG/doc* line shows what happens when the entire document is used as a single context. In this case, the results are somewhat better than without feature generation (*Baseline*), but are still inferior to the more fine-grained word-level contexts (*FG/words*). However, the best performance by far is achieved with the multi-resolution approach (*FG/multi*), in which we use a series of linguistically motivated chunks of text, starting with individual words, and then generating features from sentences, paragraphs, and finally the entire document.

5.3.4 The Effect of Knowledge Breadth

In the experiments reported in Section 5.3.2 we performed feature generation using the entire ODP. It is interesting to observe, however, that four out of the five

Dataset	Baseline		Feature generation		Improvement vs. baseline	
	micro BEP	macro BEP	micro BEP	macro BEP	micro BEP	macro BEP
Reuters-21578						
10 categories	0.925	0.874	0.930	0.884	+0.5%	+1.1%
90 categories	0.877	0.602	0.880	0.614	+0.3%	+2.0%
RCV1						
Industry-16	0.642	0.595	0.648	**0.613**	+0.9%	**+3.0%**
Industry-10A	0.421	0.335	**0.457**	**0.420**	**+8.6%**	**+25.4%**
Industry-10B	0.489	0.528	**0.530**	**0.560**	**+8.4%**	**+6.1%**
Industry-10C	0.443	0.414	**0.468**	**0.463**	**+5.6%**	**+11.8%**
Industry-10D	0.587	0.466	0.588	**0.496**	+0.2%	**+6.4%**
Industry-10E	0.648	0.605	0.657	**0.639**	+1.4%	**+5.6%**
Topic-16	0.836	0.591	0.840	**0.660**	+0.5%	**+11.7%**
Topic-10A	0.796	0.587	0.803	**0.692**	+0.9%	**+17.9%**
Topic-10B	0.716	0.618	0.727	**0.655**	+1.5%	**+6.0%**
Topic-10C	0.687	0.604	0.694	**0.618**	+1.0%	**+2.3%**
Topic-10D	0.829	0.673	0.836	0.687	+0.8%	+2.1%
Topic-10E	0.758	0.742	0.762	0.756	+0.5%	+1.9%
OHSUMED						
OHSUMED-10A	0.518	0.417	**0.537**	**0.479**	**+3.7%**	**+14.9%**
OHSUMED-10B	0.656	0.500	0.659	**0.548**	+0.5%	**+9.6%**
OHSUMED-10C	0.539	0.505	0.547	**0.540**	+1.5%	**+6.9%**
OHSUMED-10D	0.683	0.515	0.688	**0.549**	+0.7%	**+6.6%**
OHSUMED-10E	0.442	0.542	0.452	**0.573**	+2.3%	**+5.7%**
20NG	0.854		**0.858**		**+0.5%**	
Movies	0.813		**0.842**		**+3.6%**	

Table 5.6: Text categorization with and without feature generation

datasets we used have a fairly narrow scope.[8] Specifically, both Reuters datasets (Reuters-21578 and RCV1) contain predominantly economic news and therefore match the scope of the Top/Business branch of the ODP. Similarly, Movie Reviews contains opinions about movies, and therefore fits the scope of Top/Arts. OHSUMED contains medical documents, which can be modelled within the scope of Top/Health and Top/Science. In the light of this, it could be expected that restricting the feature generator to a particular ODP branch that corresponds

[8] The 20 Newsgroups dataset consists of 20 diverse categories, each of which corresponds to one or more ODP branches.

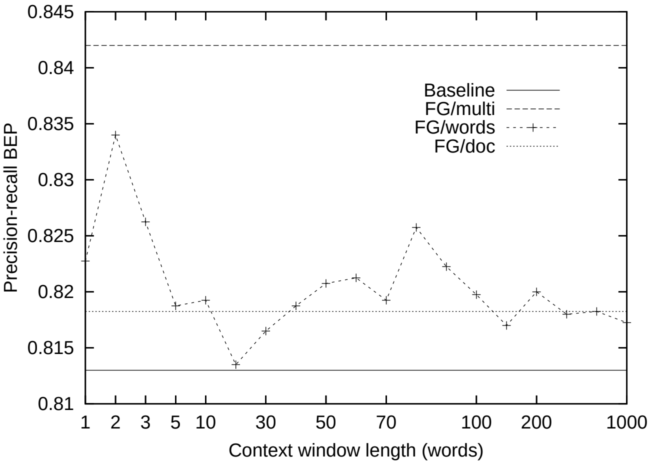

Figure 5.1: Varying context length (Movies)

to the scope of the test collection would result in much better categorization accuracy due to the elimination of noise in "unused" ODP branches.

Experimental results (Table 5.7) disprove this hypothesis. As can be seen, in the absolute majority of cases the improvement over the baseline is much larger when the entire ODP is used (cf. Table 5.6). These findings show the superiority of wide general-purpose knowledge over its domain-specific subsets.

5.3.5 The Utility of Feature Selection

Under the experimental settings defined in Section 5.2.3, feature generation constructed approximately 4–5 times as many features as are in the bag of words (after rare features that occurred in less than 3 documents were removed). We conducted two experiments to understand the effect of feature selection in conjunction with feature generation.

Since earlier studies found that feature selection from the bag of words impairs SVM performance (Section 3.4.2), in our first experiment we apply feature selection only to the generated features and use the selected ones to augment the (entire) bag of words. In Figures 5.2 and 5.3, the *BOW* line depicts the baseline performance without generated features, while the *BOW+GEN* curve shows the performance of the bag of words augmented with progressively larger fractions of generated features (sorted by information gain). For both datasets, the performance peaks when only a small fraction of the generated features are used, while retaining more generated features has a noticeable detrimental effect.

Our second experiment examined the performance of the generated features alone, without the bag of words (*GEN* curve in Figures 5.2 and 5.3). For Movies,

Dataset	Domain-specific ODP subset			Full ODP	
	Subset description	micro BEP	macro BEP	micro BEP	macro BEP
Reuters-21578	Top/Business				
10 categories		+0.4%	+0.6%	+0.5%	+1.1%
90 categories		+0.1%	+1.2%	+0.3%	+2.0%
RCV1	Top/Business				
Industry-16		+1.9%	+2.2%	+0.9%	**+3.0%**
Topic-16		+0.5%	+1.4%	+0.5%	**+11.7%**
OHSUMED	Top/Health				
OHSUMED-10A		+2.1%	+1.7%	**+3.7%**	**+14.9%**
OHSUMED-10B		+0.2%	+1.2%	+0.5%	**+9.6%**
OHSUMED-10C		+1.7%	+2.8%	+1.5%	**+6.9%**
OHSUMED-10D		+0.3%	+1.9%	+0.7%	**+6.6%**
OHSUMED-10E		+2.7%	+1.8%	+2.3%	**+5.7%**
OHSUMED	Top/Health + Top/Science				
OHSUMED-10A		**+5.4%**	**+3.6%**	**+3.7%**	**+14.9%**
OHSUMED-10B		+0.3%	**+3.4%**	+0.5%	**+9.6%**
OHSUMED-10C		+0.6%	**+3.8%**	+1.5%	**+6.9%**
OHSUMED-10D		+0.9%	**+5.8%**	+0.7%	**+6.6%**
OHSUMED-10E		+1.6%	+1.8%	+2.3%	**+5.7%**
Movies	Top/Arts	**+2.6%**		**+3.6%**	

Table 5.7: Text categorization with and without feature generation, when only a subset of the ODP is used

discarding the BOW features leads to somewhat worse performance, but the decrease is far less significant than what could be expected—using only the generated features we lose less than 3% in BEP compared with the BOW baseline. For 20NG, a similar experiment sacrifices about 10% of the BOW performance, as this dataset is known to have a very diversified vocabulary, for which many studies found feature selection to be particularly harmful. Similarly, for OHSUMED, using only the generated features sacrifices up to 15% in performance, reinforcing the value of precise medical terminology that is discarded in this experiment. However, the situation is reversed for both Reuters datasets. For Reuters-21578, the generated features alone yield a 0.3% improvement in micro- and macro-BEP for 10 categories, while for 90 categories they only lose 0.3% in micro-BEP and 3.5% in macro-BEP compared with the bag of words. For RCV1/Industry-16, disposing of the bag of words reduces BEP performance by 1–3%. Surprisingly, for

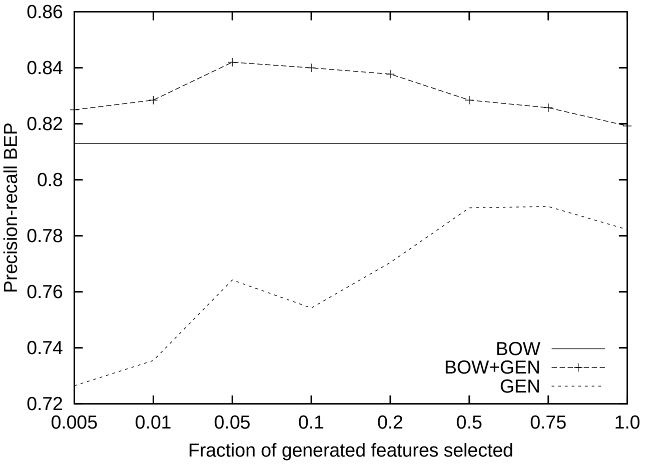

Figure 5.2: Feature selection (Movies)

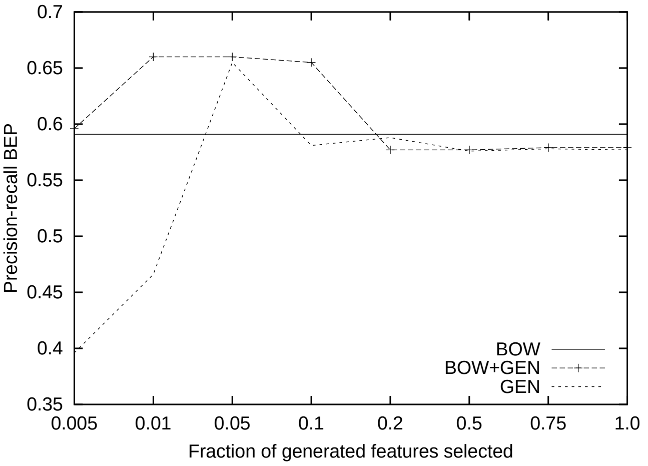

Figure 5.3: Feature selection (RCV1/Topic-16)

RCV1/Topic-16 (Figure 5.3) the generated features *per se* command a 10.8% improvement in macro-BEP, rivalling the performance of *BOW+GEN*, which gains only another 1% (Table 5.6). We interpret these findings as further reinforcement that the generated features improve the quality of the representation.

5.3.6 The Effect of Category Size

We saw in Section 5.3.2 that feature generation greatly improves text categorization for smaller categories, as can be evidenced in the greater improvements in macro-BEP. To explore this phenomenon further, we depict in Figures 5.4 and 5.5

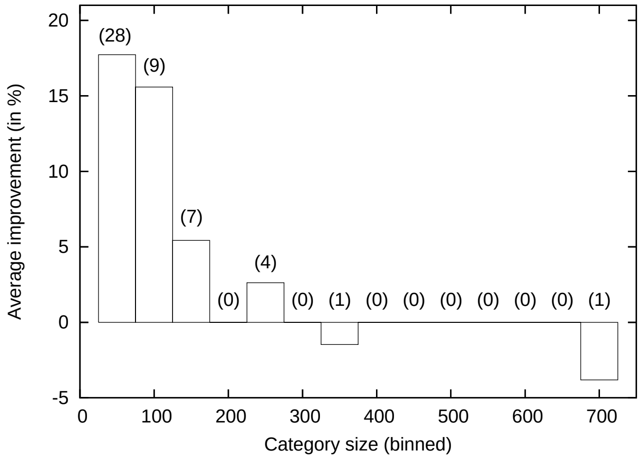

Figure 5.4: RCV1 (Industry): Average improvement versus category size

the relation between the category size and the improvement due to feature generation for RCV1 (the number of categories in each bin appears in parentheses above the bars). To this end, we pooled together the categories that comprised the individual sets (10A–10E) in the Industry and Topic groups, respectively.

As we can readily see, smaller categories tend to benefit more from knowledge-based feature generation. These graphs also explain the more substantial improvements observed for Industry categories compared to Topic categories—as can be seen from the graphs, Topic categories are larger than Industry categories, and the average size of Topic categories (among those we used in this study) is almost 6 times larger than that of Industry categories.

5.3.7 The Effect of Feature Generation for Classifying Short Documents

We conjectured that knowledge-based feature generation might be particularly useful for classifying short documents. To evaluate this hypothesis, we used the datasets defined in Section 5.1.6.

Table 5.8 presents the results of this experiment. As we can see, in the majority of cases (except for RCV1 Topic category sets), feature generation leads to greater improvement on short documents than on regular documents. Notably, the improvements are particularly high for OHSUMED, where "pure" experimentation on short documents is possible (see Section 5.1.6).

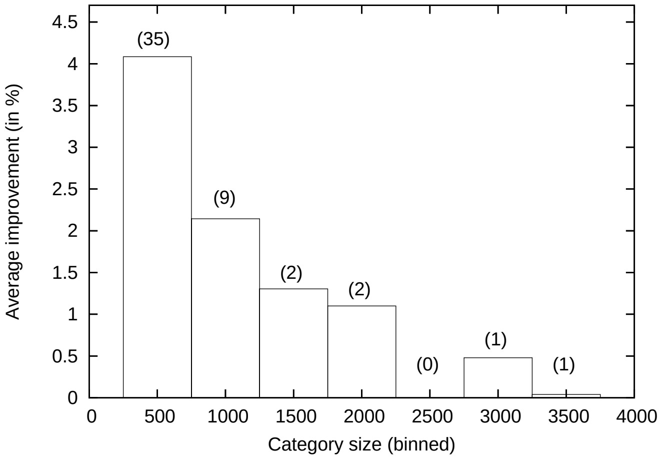

Figure 5.5: RCV1 (Topic): Average improvement versus category size

5.3.8 Processing Time

Using the ODP as a source of background knowledge requires additional computation. This extra computation includes the (one-time) preprocessing step where the feature generator is built, as well as the actual feature generation performed on documents prior to text categorization. The processing times reported below were measured on a workstation with dual Xeon 2.2 GHz CPU and 2 Gb RAM running the Microsoft Windows XP Professional operating system (Service Pack 1).

Parsing the ODP structure (file `structure.rdf.u8`) took 3 minutes. Parsing the list of ODP URLs (file `content.rdf.u8`) required 3 hours, and parsing the crawled ODP data (meta-documents collected from all cataloged URLs) required 2.6 days. Attribute selection for ODP concepts took 1.5 hours. The cumulative one-time expenditure for building the feature generator was therefore just under 3 days (not counting the actual Web crawling that was performed beforehand).

We benchmarked feature generation in two scenarios—individual words and 10-word windows. In the former case, the feature generator classified approximately 310 words per second, while in the latter case it classified approximately 45 10-word windows per second (i.e., 450 words per second).[9] These times constitute the additional overhead required by feature generation compared with regular text categorization. Table 5.9 lists the sizes of the test collections we experimented with (see Section 5.1). To speed up experimentation, we used subsets of the entire RCV1 and OHSUMED collections; these subsets are comparable in

[9] Classifying word windows is more efficient due to the sharing of data structures when processing the words in a single context.

Dataset	Short documents						Full documents	
	Baseline		Feature generation		Improvement vs. baseline		Improvement vs. baseline	
	micro BEP	macro BEP	micro BEP	macro BEP	micro BEP	macro BEP	micro BEP	macro BEP
Reuters-21578								
10 categories	0.868	0.774	0.868	0.777	+0.0%	+0.4%	+0.5%	+1.1%
90 categories	0.793	0.479	0.794	0.498	+0.1%	**+4.0%**	+0.3%	+2.0%
RCV1								
Industry-16	0.454	0.400	0.466	**0.415**	+2.6%	**+3.7%**	+0.9%	**+3.0%**
Industry-10A	0.249	0.199	**0.278**	**0.256**	**+11.6%**	**+28.6%**	**+8.6%**	**+25.4%**
Industry-10B	0.273	0.292	**0.348**	**0.331**	**+27.5%**	**+13.4%**	**+8.4%**	**+6.1%**
Industry-10C	0.209	0.199	**0.295**	**0.308**	**+41.1%**	**+54.8%**	**+5.6%**	**+11.8%**
Industry-10D	0.408	0.361	**0.430**	**0.431**	**+5.4%**	**+19.4%**	+0.2%	**+6.4%**
Industry-10E	0.450	0.410	**0.490**	**0.459**	**+8.9%**	**+12.2%**	+1.4%	**+5.6%**
Topic-16	0.763	0.529	0.763	0.534	+0.0%	+0.9%	+0.5%	**+11.7%**
Topic-10A	0.718	0.507	0.720	0.510	+0.3%	+0.6%	+0.9%	**+17.9%**
Topic-10B	0.647	0.560	0.644	0.560	-0.5%	+0.0%	+1.5%	**+6.0%**
Topic-10C	0.551	0.471	0.561	0.475	+1.8%	+0.8%	+1.0%	**+2.3%**
Topic-10D	0.729	0.535	0.730	**0.553**	+0.1%	**+3.4%**	+0.8%	+2.1%
Topic-10E	0.643	0.636	0.656	0.646	+2.0%	+1.6%	+0.5%	+1.9%
OHSUMED								
OHSUMED-10A	0.302	0.221	**0.357**	**0.253**	**+18.2%**	**+14.5%**	**+3.7%**	**+14.9%**
OHSUMED-10B	0.306	0.187	**0.348**	**0.243**	**+13.7%**	**+29.9%**	+0.5%	**+9.6%**
OHSUMED-10C	0.441	0.296	**0.494**	**0.362**	**+12.0%**	**+22.3%**	+1.5%	**+6.9%**
OHSUMED-10D	0.441	0.356	0.448	**0.419**	+1.6%	**+17.7%**	+0.7%	**+6.6%**
OHSUMED-10E	0.164	0.206	**0.211**	**0.269**	**+28.7%**	**+30.6%**	+2.3%	**+5.7%**
20NG	0.699		**0.740**		**+5.9%**		**+0.5%**	

Table 5.8: Text categorization of short documents with and without feature generation (the improvement percentage in the two rightmost columns is computed relative to the baseline shown in Table 5.6)

size with 20 Newsgroups and Reuters-21578.

In the light of the improvements in categorization accuracy due to feature generation, we believe that the extra processing time is well compensated for. In operational text categorization systems, documents rarely arrive in huge batches of hundreds of thousands at a time. For example, the RCV1 dataset contains all English-language news items published by Reuters over the period of one year. Therefore, in practical settings, once the classification model has been trained, the number of documents it needs to classify per time unit is much more reasonable,

Dataset	Number of documents	Number of words[10]
20NG	19,997	5.5 million
Movies	1,400	0.95 million
Reuters-21578	21,902	2.8 million
RCV1		
- full	804,414	196 million
- used in this study	23,149	5.5 million
OHSUMED		
- full	348,566	57 million
- used in this study	20,000	3.7 million

Table 5.9: Test collection sizes

and can be easily facilitated by our system.

5.4 Wikipedia-based Feature Generation

In this section we evaluate the feature generator based on Wikipedia.

5.4.1 Qualitative Analysis of Feature Generation

We start with demonstrating the results of feature generation on a number of actual examples.

Feature Generation per se

To illustrate our approach, we show features generated for several text fragments. Whenever applicable, we provide short explanations of the generated concepts; in most cases, the explanations are taken from Wikipedia itself (Wikipedia, 2006).

- **Text:** *"Wal-Mart supply chain goes real time"*

  Sample generated features:

  - Wal-Mart
  - Sam Walton — Wal-Mart founder
  - Sears Holdings Corporation, Target Corporation, Albertsons — prominent competitors of Wal-Mart

[10] Measured using the 'wc' utility available on UNIX systems.

  - RFID — Radio Frequency Identification, a technology that Wal-Mart uses very extensively to manage its stock
  - Hypermarket — superstore (a general concept, of which Wal-Mart is a specific example)
  - United Food and Commercial Workers — a labor union that has been trying to organize Wal-Mart's workers

- **Text:** *"Scientific methods in biology"*

  Sample generated features:

  - Biology
  - Scientific classification
  - Science
  - Chemical biology
  - Binomial nomenclature — the formal method of naming species in biology
  - Nature (journal)
  - Social sciences
  - Philosophy of biology
  - Scientist
  - History of biology

- **Text:** *"With quavering voices, parents and grandparents of those killed at the World Trade Center read the names of the victims in a solemn recitation today, marking the third anniversary of the terror attacks. The ceremony is one of many planned in the United States and around the world to honor the memory of the nearly 3,000 victims of 9/11."*

  Sample generated features:

  - September 11, 2001 attack memorials and services
  - United Airlines Flight 93 — one of the four flights hijacked on September 11, 2001
  - Aftermath of the September 11, 2001 attacks
  - World Trade Center
  - September 11, 2001 attacks
  - Oklahoma City bombing — a terrorist attack in Oklahoma City in 1995

  - World Trade Center bombing
  - Arlington National Cemetery — an American military cemetery
  - World Trade Center site
  - Jewish bereavement

- **Text:** *"A group of European-led astronomers has made a photograph of what appears to be a planet orbiting another star. If so, it would be the first confirmed picture of a world beyond our solar system."*

  Sample generated features:

  - Planet
  - Solar system
  - Astronomy
  - Planetary orbit
  - Extrasolar planet
  - Pluto
  - Jupiter
  - Neptune
  - Minor planet
  - Mars

- **Text:** *"Nearly 70 percent of Americans say they are careful about what they eat, and even more say diet is essential to good health, according to a new nationwide health poll in which obesity ranked second among the biggest health concerns."*

  Sample generated features:

  - Veganism — a philosophy of avoiding animal-derived food
  - Vegetarianism
  - Obesity
  - Atkins Nutritional Approach
  - Binge eating disorder
  - Dick Gregory — an American nutritionist
  - Nutrition
  - Super Size Me — a documentary film about an individual who ate only McDonald's fast food for one full month

  - Health insurance
  - Eating disorder

- **Text:** *"U.S. intelligence cannot say conclusively that Saddam Hussein has weapons of mass destruction, an information gap that is complicating White House efforts to build support for an attack on Saddam's Iraqi regime. The CIA has advised top administration officials to assume that Iraq has some weapons of mass destruction. But the agency has not given President Bush a "smoking gun," according to U.S. intelligence and administration officials."*

  Sample generated features:

  - Iraq disarmament crisis
  - Yellowcake forgery — falsified intelligence documents about Iraq's alleged attempt to purchase yellowcake uranium
  - Senate Report of Pre-War Intelligence on Iraq
  - Iraq and weapons of mass destruction
  - Iraq Survey Group
  - September Dossier — a paper on Iraq's weapons of mass destruction published by the UK government in 2002
  - Iraq war
  - Scott Ritter — UN weapons inspector in Iraq
  - Iraq War Rationale
  - Operation Desert Fox — US and UK joint military campaign in Iraq in 1998

- **Text:** *'The development of T-cell leukaemia following the otherwise successful treatment of three patients with X-linked severe combined immune deficiency (X-SCID) in gene-therapy trials using haematopoietic stem cells has led to a re-evaluation of this approach. Using a mouse model for gene therapy of X-SCID, we find that the corrective therapeutic gene IL2RG itself can act as a contributor to the genesis of T-cell lymphomas, with one-third of animals being affected. Gene-therapy trials for X-SCID, which have been based on the assumption that IL2RG is minimally oncogenic, may therefore pose some risk to patients."*

  Sample generated features:

  - Leukemia
  - Severe combined immunodeficiency

  - Cancer
  - Non-Hodgkin lymphoma — a particular cancer type
  - AIDS
  - ICD-10 Chapter II: Neoplasms; Chapter III: Diseases of the blood and blood-forming organs, and certain disorders involving the immune mechanism — a disease code of the ICD (International Statistical Classification of Diseases and Related Health Problems)
  - Bone marrow transplant
  - Immunosuppressive drug
  - Acute lymphoblastic leukemia
  - Multiple sclerosis

- Finally, it is particularly interesting to juxtapose the features generated for fragments that contain ambiguous words. To this end, we show features generated for two phrases that contain the word "bank" in two different senses, "Bank of America" (financial institution) and "Bank of Amazon" (river bank). As can be readily seen, our feature generation methodology is capable of performing word sense disambiguation by considering ambiguous words in the context of their neighbors.

  Text: *"**Bank** of America"*

  Sample generated features:

  - Bank
  - Bank of America
  - Bank of America Plaza (Atlanta)
  - Bank of America Plaza (Dallas)
  - MBNA — a bank holding company acquired by Bank of America
  - VISA (credit card)
  - Bank of America Tower, New York City
  - NASDAQ
  - MasterCard
  - Bank of America Corporate Center

  Text: *"**Bank** of Amazon"*

  Sample generated features:

- Amazon River
- Amazon Basin
- Amazon Rainforest
- Amazon.com
- Rainforest
- Atlantic Ocean
- Brazil
- Loreto Region - a region in Peru, located in the Amazon Rainforest
- River
- Economy of Brazil

- As another example, consider a pair of contexts that contain the word "jaguar", where the first context contains this ambiguous word in the sense of a car model, and the second one—in the sense of an animal.

  Text: *"**Jaguar** car models"*

  Sample generated features:

  - Jaguar (car)
  - Jaguar (S-Type) — a particular Jaguar car model
  - Jaguar X-type — a particular Jaguar car model
  - Jaguar E-Type — a particular Jaguar car model
  - Jaguar XJ — a particular Jaguar car model
  - Daimler Motor Company — a car manufacturing company that became a part of Jaguar in 1960
  - British Leyland Motor Corporation - another vehicle manufacturing company that merged with Jaguar
  - Luxury vehicles
  - V8 engine — an internal combustion engine used in some Jaguar car models
  - Jaguar Racing — a Formula One team used by Jaguar to promote its brand name

  Text: *"**Jaguar** (Panthera onca)"*

  Sample generated features:

  - Jaguar

  - Felidae — a family that include lions, tigers, jaguars, and other related feline species
  - Black panther
  - Leopard
  - Puma
  - Tiger
  - Panthera hybrid
  - Cave lion
  - American lion
  - Kinkajou — another carnivore mammal

Using Inter-article Links for Generating Additional Features

In Section 4.2.3, we presented an algorithm that generates additional features using inter-article links as relations between concepts. In what follows, we show a series of text fragments, where for each fragment we show (a) features generated with the regular FG algorithm, (b) features generated using Wikipedia links, and (c) more general features generated using links. As we can see from the examples, the features constructed from the links are in most cases highly relevant to the input text.

- **Text:** *"Google search"*

  Regular feature generation:

  - Search engine
  - Google Video
  - Google
  - Google (search)
  - Google Maps
  - Google Desktop
  - Google (verb)
  - Google News
  - Search engine optimization
  - Spamdexing — search engine spamming

  Features generated using links:

  - PageRank
  - AdWords
  - AdSense
  - Gmail
  - Google Platform
  - Website
  - Sergey Brin
  - Google bomb
  - MSN Search
  - Nigritude ultramarine — a meaningless phrase used in a search engine optimization contest in 2004

  More general features only:

  - Website
  - Mozilla Firefox
  - Portable Document Format
  - Algorithm
  - World Wide Web

- **Text:** *"artificial intelligence"*

  Regular feature generation:

  - Artificial intelligence
  - A.I. (film)
  - MIT Computer Science and Artificial Intelligence Laboratory
  - Artificial life
  - Strong AI
  - Swarm intelligence
  - Computer Science
  - Frame problem
  - Cognitive science
  - Carl Hewitt

  Features generated using links:

  - Robot
  - John McCarthy (computer scientist)
  - Artificial consciousness
  - Marvin Minsky
  - Planner programming language
  - Actor model — a model of concurrent computation formulated by Carl Hewitt and his colleagues
  - Logic
  - Scientific Community Metaphor
  - Natural language processing
  - Lisp programming language

  More general features only:

  - Robot
  - Massachusetts Institute of Technology
  - Psychology
  - Consciousness
  - Lisp programming language

- **Text:** *"Israel terror"*

  Regular feature generation:

  - Israel
  - Palestinian political violence
  - Terrorism
  - Labour (Israel)
  - Terrorism against Israel
  - Israel Defense Forces
  - Shabak
  - Steve Israel — an American politician who has worked extensively on military and terrorism-related issues; his interests include national security and the State of Israel
  - Israeli peace camp

  - Agudat Israel — a religious political party in Israel, which has recently become more conscious of issues related to Israel's security

  Features generated using links:

  - Oslo Accords
  - Al-Aqsa Intifada
  - Israeli-Palestinian conflict
  - 1982 Lebanon War
  - British Mandate of Palestine
  - Israel Border Police
  - Israel's unilateral disengagement plan
  - History of Israel
  - Israeli Security Forces
  - Israel-Jordan Treaty of Peace

  More general features only:

  - Jew
  - Gaza Strip
  - West Bank
  - British Mandate of Palestine
  - Six-Day War

- **Text:** *"programming tools"*

  Regular feature generation:

  - Tool
  - Programming tool
  - Computer software
  - Integrated development environment
  - Computer-aided software engineering
  - Macromedia Flash
  - Borland
  - Game programmer
  - C programming language

  - Performance analysis

  Features generated using links:

  - Compiler
  - Debugger
  - Source code
  - Software engineering
  - Microsoft
  - Revision control
  - Scripting language
  - GNU
  - Make
  - Linux

  More general features only:

  - Microsoft
  - Software engineering
  - Linux
  - Compiler
  - GNU

- **Text:** *"A group of European-led astronomers has made a photograph of what appears to be a planet orbiting another star. If so, it would be the first confirmed picture of a world beyond our solar system."*

  Regular feature generation:

  - Planet
  - Solar system
  - Astronomy
  - Planetary orbit
  - Extrasolar planet
  - Pluto
  - Jupiter
  - Neptune

  - Minor planet
  - Mars

  Features generated using links:

  - Asteroid
  - Earth
  - Oort cloud — a postulated cloud of comets
  - Comet
  - Sun
  - Saturn
  - Moon
  - Mercury (planet)
  - Asteroid belt
  - Orbital period

  More general features only:

  - Earth
  - Moon
  - Asteroid
  - Sun
  - National Aeronautics and Space Administration

- **Text:** *"Nearly 70 percent of Americans say they are careful about what they eat, and even more say diet is essential to good health, according to a new nationwide health poll in which obesity ranked second among the biggest health concerns."*

  Regular feature generation:

  - Veganism — a philosophy of avoiding animal-derived food
  - Vegetarianism
  - Obesity
  - Atkins Nutritional Approach
  - Binge eating disorder
  - Dick Gregory — an American nutritionist

- Nutrition
- Super Size Me — a documentary film about an individual who eats only McDonald's fast food for one full month
- Health insurance
- Eating disorder

Features generated using links:

- Raw food diet
- Diabetes mellitus
- Healthy eating
- Body mass index
- Omega-3 fatty acid — an important nutritional component
- Dieting
- Milk
- United States — this classification is quite interesting, as the issue discussed in the input text fragment is very characteristic of the American life style
- Hypertension
- Egg (food)

More general features only:

- United States
- Diabetes mellitus
- Cancer
- Food
- McDonald's

5.4.2 The Effect of Feature Generation

Table 5.10 shows the results of using Wikipedia-based feature generation, with significant improvements ($p < 0.05$) shown in bold. We consistently observed larger improvements in macro-averaged BEP, which is dominated by categorization effectiveness on small categories. This goes in line with our expectations that the contribution of encyclopedic knowledge should be especially prominent

Dataset	Baseline		Wikipedia		Improvement	
	micro	macro	micro	macro	micro	macro
	BEP	BEP	BEP	BEP	BEP	BEP
Reuters-21578 (10 cat.)	0.925	0.874	0.932	0.887	+0.8%	+1.5%
Reuters-21578 (90 cat.)	0.877	0.602	0.883	0.603	+0.7%	+0.2%
RCV1 Industry-16	0.642	0.595	0.645	**0.617**	+0.5%	**+3.7%**
RCV1 Industry-10A	0.421	0.335	**0.448**	**0.437**	**+6.4%**	**+30.4%**
RCV1 Industry-10B	0.489	0.528	**0.523**	**0.566**	**+7.0%**	**+7.2%**
RCV1 Industry-10C	0.443	0.414	**0.468**	**0.431**	**+5.6%**	**+4.1%**
RCV1 Industry-10D	0.587	0.466	0.595	0.459	+1.4%	-1.5%
RCV1 Industry-10E	0.648	0.605	0.641	0.612	-1.1%	+1.2%
RCV1 Topic-16	0.836	0.591	0.843	**0.661**	+0.8%	**+11.8%**
RCV1 Topic-10A	0.796	0.587	0.798	**0.682**	+0.3%	**+16.2%**
RCV1 Topic-10B	0.716	0.618	0.723	**0.656**	+1.0%	**+6.1%**
RCV1 Topic-10C	0.687	0.604	0.699	0.618	+1.7%	+2.3%
RCV1 Topic-10D	0.829	0.673	0.839	0.688	+1.2%	+2.2%
RCV1 Topic-10E	0.758	0.742	0.765	0.755	+0.9%	+1.8%
OHSUMED-10A	0.518	0.417	**0.538**	**0.492**	**+3.9%**	**+18.0%**
OHSUMED-10B	0.656	0.500	0.667	**0.534**	+1.7%	**+6.8%**
OHSUMED-10C	0.539	0.505	0.545	**0.522**	+1.1%	**+3.4%**
OHSUMED-10D	0.683	0.515	0.692	**0.546**	+1.3%	**+6.0%**
OHSUMED-10E	0.442	0.542	**0.462**	**0.575**	**+4.5%**	**+6.1%**
20NG	0.854		**0.862**		**+1.0%**	
Movies	0.813		**0.842**		**+3.6%**	

Table 5.10: The effect of feature generation

for categories with few training examples. Categorization performance was improved for virtually all datasets, with notable improvements of up to 30.4% for RCV1 and 18% for OHSUMED. Using the Wilcoxon test, we found that the Wikipedia-based classifier is significantly superior to the baseline with $p < 10^{-5}$ in both micro- and macro-averaged cases. Given the performance plateau currently reached by the best text categorizers, these results clearly demonstrate the advantage of knowledge-based feature generation.

	Wikipedia snapshot as of November 11, 2005	**Wikipedia snapshot as of March 23, 2006**
Combined article text	1.8 Gb	2.9 Gb
Number of articles	910,989	1,187,839
Concepts used	171,332	241,393
Distinct terms	296,157	389,202

Table 5.11: Comparison of two Wikipedia snapshots

5.4.3 The Effect of Knowledge Breadth

Wikipedia is being constantly expanded with new material as volunteer editors contribute new articles and extend the existing ones. Consequently, we conjectured that such addition of information should be beneficial for feature generation, as it would rely on a larger knowledge base.

To test this assumption, we acquired a new Wikipedia snapshot as of March 26, 2006. Table 5.11 presents a comparison in the amount of information between two Wikipedia snapshots we used. Table 5.12 shows the effect of feature generation using the newer snapshot. As we can see, using the larger amount of knowledge leads on average to greater improvements in text categorization performance. Although the difference between the performance of the two versions is admittedly small, it is consistent across datasets (a similar situation happens when assessing the role of external knowledge for computing semantic relatedness, see Section 6.2.2).

5.4.4 Classifying Short Documents

We conjectured that Wikipedia-based feature generation should be particularly useful for classifying short documents, similarly to using ODP (cf. Section 5.3.7).

Table 5.13 presents the results of this evaluation on the datasets defined in Section 5.1.6. In the majority of cases, feature generation yielded greater improvement on short documents than on regular documents. Notably, the improvements are particularly high for OHSUMED, where "pure" experimentation on short documents is possible (see Section 5.1.6). According to the Wilcoxon test, the Wikipedia-based classifier is significantly superior to the baseline with $p < 2 \cdot 10^{-6}$. These findings confirm our hypothesis that encyclopedic knowledge should be particularly useful when categorizing short documents, which are inadequately represented by the standard bag of words.

Dataset	Baseline		Wikipedia (26/03/06)		Improvement (26/03/06)		Improvement (05/11/05)	
	micro	macro	micro	macro	micro	macro	micro	macro
	BEP	BEP	BEP	BEP	BEP	BEP	BEP	BEP
Reuters-21578 (10 cat.)	0.925	0.874	0.935	0.891	+1.1%	+1.9%	+0.8%	+1.5%
Reuters-21578 (90 cat.)	0.877	0.602	0.883	0.600	+0.7%	-0.3%	+0.7%	+0.2%
RCV1 Industry-16	0.642	0.595	0.648	**0.616**	+0.9%	**+3.5%**	+0.5%	**+3.7%**
RCV1 Industry-10A	0.421	0.335	**0.457**	**0.450**	**+8.6%**	**+34.3%**	**+6.4%**	**+30.4%**
RCV1 Industry-10B	0.489	0.528	**0.527**	**0.559**	**+7.8%**	**+5.9%**	**+7.0%**	**+7.2%**
RCV1 Industry-10C	0.443	0.414	**0.458**	0.424	**+3.4%**	+2.4%	**+5.6%**	**+4.1%**
RCV1 Industry-10D	0.587	0.466	0.607	**0.448**	+3.4%	**-3.9%**	+1.4%	-1.5%
RCV1 Industry-10E	0.648	0.605	0.649	0.607	+0.2%	+0.3%	-1.1%	+1.2%
RCV1 Topic-16	0.836	0.591	0.842	**0.659**	+0.7%	**+11.5%**	+0.8%	**+11.8%**
RCV1 Topic-10A	0.796	0.587	0.802	**0.689**	+0.8%	**+17.4%**	+0.3%	**+16.2%**
RCV1 Topic-10B	0.716	0.618	0.725	**0.660**	+1.3%	**+6.8%**	+1.0%	**+6.1%**
RCV1 Topic-10C	0.687	0.604	0.697	**0.627**	+1.5%	**+3.8%**	+1.7%	+2.3%
RCV1 Topic-10D	0.829	0.673	0.838	0.687	+1.1%	+2.1%	+1.2%	+2.2%
RCV1 Topic-10E	0.758	0.742	0.762	0.752	+0.5%	+1.3%	+0.9%	+1.8%
OHSUMED-10A	0.518	0.417	**0.545**	**0.490**	**+5.2%**	**+17.5%**	**+3.9%**	**+18.0%**
OHSUMED-10B	0.656	0.500	0.667	**0.529**	+1.7%	**+5.8%**	+1.7%	**+6.8%**
OHSUMED-10C	0.539	0.505	0.553	**0.527**	+2.6%	**+4.4%**	+1.1%	**+3.4%**
OHSUMED-10D	0.683	0.515	0.694	**0.550**	+1.6%	**+6.8%**	+1.3%	**+6.0%**
OHSUMED-10E	0.442	0.542	**0.461**	**0.588**	**+4.3%**	**+8.5%**	**+4.5%**	**+6.1%**
20NG	0.854		**0.859**		**+0.6%**		**+1.0%**	
Movies	0.813		**0.850**		**+4.5%**		**+3.6%**	
Average					*+2.50%*	*+6.84%*	*+2.11%*	*+6.71%*

Table 5.12: The effect of feature generation using a newer Wikipedia snapshot (dated March 26, 2006)

5.4.5 Using Inter-article links as Concept Relations

Using inter-article links for generating additional features, we observed further improvements in text categorization performance on short documents. As we can see in Table 5.14, in the absolute majority of cases using links to generate more general features only is a superior strategy.

Dataset	Baseline		Wikipedia		Improvement	
	micro	macro	micro	macro	micro	macro
	BEP	BEP	BEP	BEP	BEP	BEP
Reuters-21578 (10 cat.)	0.868	0.774	0.877	0.793	+1.0%	+2.5%
Reuters-21578 (90 cat.)	0.793	0.479	0.803	**0.506**	+1.3%	**+5.6%**
RCV1 Industry-16	0.454	0.400	**0.481**	**0.437**	**+5.9%**	**+9.2%**
RCV1 Industry-10A	0.249	0.199	**0.293**	**0.256**	**+17.7%**	**+28.6%**
RCV1 Industry-10B	0.273	0.292	**0.337**	**0.363**	**+23.4%**	**+24.3%**
RCV1 Industry-10C	0.209	0.199	**0.294**	**0.327**	**+40.7%**	**+64.3%**
RCV1 Industry-10D	0.408	0.361	**0.452**	**0.379**	**+10.8%**	**+5.0%**
RCV1 Industry-10E	0.450	0.410	**0.474**	**0.434**	**+5.3%**	**+5.9%**
RCV1 Topic-16	0.763	0.529	0.769	0.542	+0.8%	+2.5%
RCV1 Topic-10A	0.718	0.507	0.725	**0.544**	+1.0%	**+7.3%**
RCV1 Topic-10B	0.647	0.560	0.643	0.564	-0.6%	+0.7%
RCV1 Topic-10C	0.551	0.471	**0.573**	**0.507**	**+4.0%**	**+7.6%**
RCV1 Topic-10D	0.729	0.535	0.735	**0.563**	+0.8%	**+5.2%**
RCV1 Topic-10E	0.643	0.636	0.670	0.653	+4.2%	+2.7%
OHSUMED-10A	0.302	0.221	**0.405**	**0.299**	**+34.1%**	**+35.3%**
OHSUMED-10B	0.306	0.187	**0.383**	**0.256**	**+25.2%**	**+36.9%**
OHSUMED-10C	0.441	0.296	**0.528**	**0.413**	**+19.7%**	**+39.5%**
OHSUMED-10D	0.441	0.356	**0.460**	**0.402**	**+4.3%**	**+12.9%**
OHSUMED-10E	0.164	0.206	**0.219**	**0.280**	**+33.5%**	**+35.9%**
20NG	0.699		**0.749**		**+7.1%**	

Table 5.13: Feature generation for short documents

DATASET	Baseline		Wikipedia		Wikipedia + links		Wikipedia + links (more general features only)	
	micro BEP	macro BEP	micro BEP	macro BEP	micro BEP	macro BEP	micro BEP	macro BEP
Reuters-21578 (10 cat.)	0.868	0.774	0.877	0.793	0.878	0.796	0.880	0.801
Reuters-21578 (90 cat.)	0.793	0.479	0.803	0.506	0.804	0.506	0.809	0.507
RCV1 Industry-16	0.454	0.400	0.481	0.437	0.486	0.445	0.488	0.444
RCV1 Topic-16	0.763	0.529	0.769	0.542	0.769	0.539	0.775	0.545
20NG	0.699		0.749		0.753		0.756	
DATASET			Improvement over baseline		Improvement over baseline		Improvement over baseline	
Reuters-21578 (10 cat.)	–	–	+1.0%	+2.5%	+1.2%	+2.8%	+1.4%	+3.5%
Reuters-21578 (90 cat.)	–	–	+1.3%	+5.6%	+1.4%	+5.6%	+2.0%	+5.8%
RCV1 Industry-16	–	–	+5.9%	+9.2%	+7.1%	+11.3%	+7.5%	+11.0%
RCV1 Topic-16	–	–	+0.8%	+2.5%	+0.8%	+1.9%	+1.6%	+3.0%
20NG	–		+7.1%		+7.7%		+8.1%	

Table 5.14: Feature generation for short documents using inter-article links

Chapter 6

Using Feature Generation for Computing Semantic Relatedness of Texts

How related are “cat” and “mouse”? And what about “preparing a manuscript” and “writing an article”? The ability to quantify semantic relatedness of texts underlies many fundamental tasks in computational linguistics, including word sense disambiguation, information retrieval, word and text clustering, and error correction (Budanitsky and Hirst, 2006). Reasoning about semantic relatedness of natural language utterances is routinely performed by humans but remains an unsurmountable obstacle for computers. Humans do not judge text relatedness merely at the level of text words. Words trigger reasoning at a much deeper level that manipulates *concepts*—the basic units of meaning that serve humans to organize and share their knowledge. Thus, humans interpret the specific wording of a document in the much larger context of their background knowledge and experience. Lacking such elaborate resources, computers need alternative ways to represent texts and reason about them.

In this Chapter, we discuss the application of our feature generation methodology to automatic assessment of semantic relatedness of words and texts (Gabrilovich and Markovitch, 2007; Gabrilovich and Markovitch, 2006a).

6.1 Explicit Semantic Analysis

In supervised text categorization, one is usually given a collection of labeled text documents, from which one can induce a text categorizer. Consequently, words that occur in the training examples can serve as valuable features—this is how the bag of words approach was born. There are, however, other tasks in natural language processing, in which labeled training examples can hardly be produced, because the decisions are essentially “one-off”. A notable example of such a task is automatic assessment of semantic relatedness of words and texts. Here, given a pair of text fragments, we need to quantify their relatedness on some scale, say, between 0 and 1. In such cases, the very words of the text fragments are likely to be of marginal usefulness, and when the two fragments are one word long, the words are probably useless at all. This happens because all the data available to us is limited to the two input fragments, which in most cases share few words, if at all.

Besides the inappropriateness of the bag of words, another obvious conjecture is that external knowledge is likely to be of substantial benefit for assessing semantic relatedness, as this is exactly the kind of knowledge that humans apply to this task. Therefore, we propose to apply our feature generation methodology to the task of computing semantic relatedness. Given a pair of text fragments whose semantic relatedness needs to be established, we use the feature generator to construct knowledge-based features for each fragment that will replace its bag of words. However, instead of using only a few top-scoring concepts, we now consider all the available concepts. The role of the feature generator is, therefore, to quantify the affinity of the input text fragment to each of the knowledge concepts. The output of feature generation in this case is a vector of weights, one per concept, which quantifies the relevance of the concept to the text fragment. In other words, text fragments are represented in the space of all knowledge concepts. Since our task is inherently not supervised, we cannot perform feature selection in the conventional sense. Instead, we rely on the weights assigned to the concepts in order to only select those concepts that are strongly relevant to the input. Concepts that are marginally relevant to the input have their weights dropped to zero, thus eliminating spurious associations.

We represent text fragments as a weighted mixture of a predetermined set of *natural* concepts, which are defined by humans themselves and can be easily explained. An important advantage of our approach is thus the use of vast amounts of highly organized human knowledge. Compared to Latent Semantic Analysis, our methodology explicitly uses the knowledge collected and organized by humans. Compared to lexical resources such as WordNet, our methodology leverages knowledge bases that are orders of magnitude larger and more comprehensive. Importantly, the Web-based knowledge repositories we use in this work undergo constant development so their breadth and depth steadily increase over time.

Viewed more generally, our methodology can be seen as building a *semantic interpreter*, which maps fragments of natural language text into a weighted sequence of concepts ordered by their relevance to the input. This way, weighted vectors of concepts that represent input texts can be viewed as their *interpretation* *vectors*. The meaning of a text fragment is thus interpreted in terms of its affinity with a host of knowledge concepts. Computing semantic relatedness of texts then amounts to comparing their vectors in the space defined by the concepts, for example, using the cosine metric (Zobel and Moffat, 1998). Our semantic analysis is *explicit* in the sense that we manipulate manifest concepts grounded in human cognition, rather than "latent concepts" used by LSA. Therefore, we call our approach Explicit Semantic Analysis (ESA).

To speed up semantic interpretation, we build an *inverted index*, which maps each word into a list of concepts in which it appears. The inverted index is also used to discard insignificant associations between words and concepts by removing those concepts whose weights for a given word are too low.

Given a text fragment, we first represent it as an attribute vector using *tf.idf* scheme. The semantic interpreter iterates over the text words, retrieves corresponding entries from the inverted index, and merges them into a weighted vector of concepts that represents the given text. Let $T = \{w_i\}$ be the input text, and let $\langle v_i \rangle$ be its attribute vector, where v_i is the weight of word w_i. Let $\langle k_j \rangle$ be an inverted index entry for word w_i, where k_j quantifies the strength of association of word w_i with knowledge concept $c_j \in \{c_0, \ldots, c_n\}$ (where n is the total number of concepts). Then, the semantic interpretation vector V for text T is a vector of length n, in which the weight of each concept c_j is defined as $\sum_{w_i \in T} v_i \cdot k_j$. Entries of this vector reflect the affinity of the corresponding concepts to text T. Figure 6.1 illustrates the processes of building and using the semantic interpreter.

6.2 Empirical Evaluation of Explicit Semantic Analysis

Humans have an innate ability to judge semantic relatedness of texts. Human judgements on a reference set of text pairs can thus be considered correct by definition, a kind of "gold standard" against which computer algorithms are evaluated. Several studies measured inter-judge correlations and found them to be consistently high (Budanitsky and Hirst, 2006; Jarmasz, 2003; Finkelstein et al., 2002a), $r = 0.88 - 0.95$. These findings are to be expected—after all, it is this consensus that allows people to understand each other. Consequently, our evaluation amounts to computing the correlation of ESA relatedness scores with human judgments.

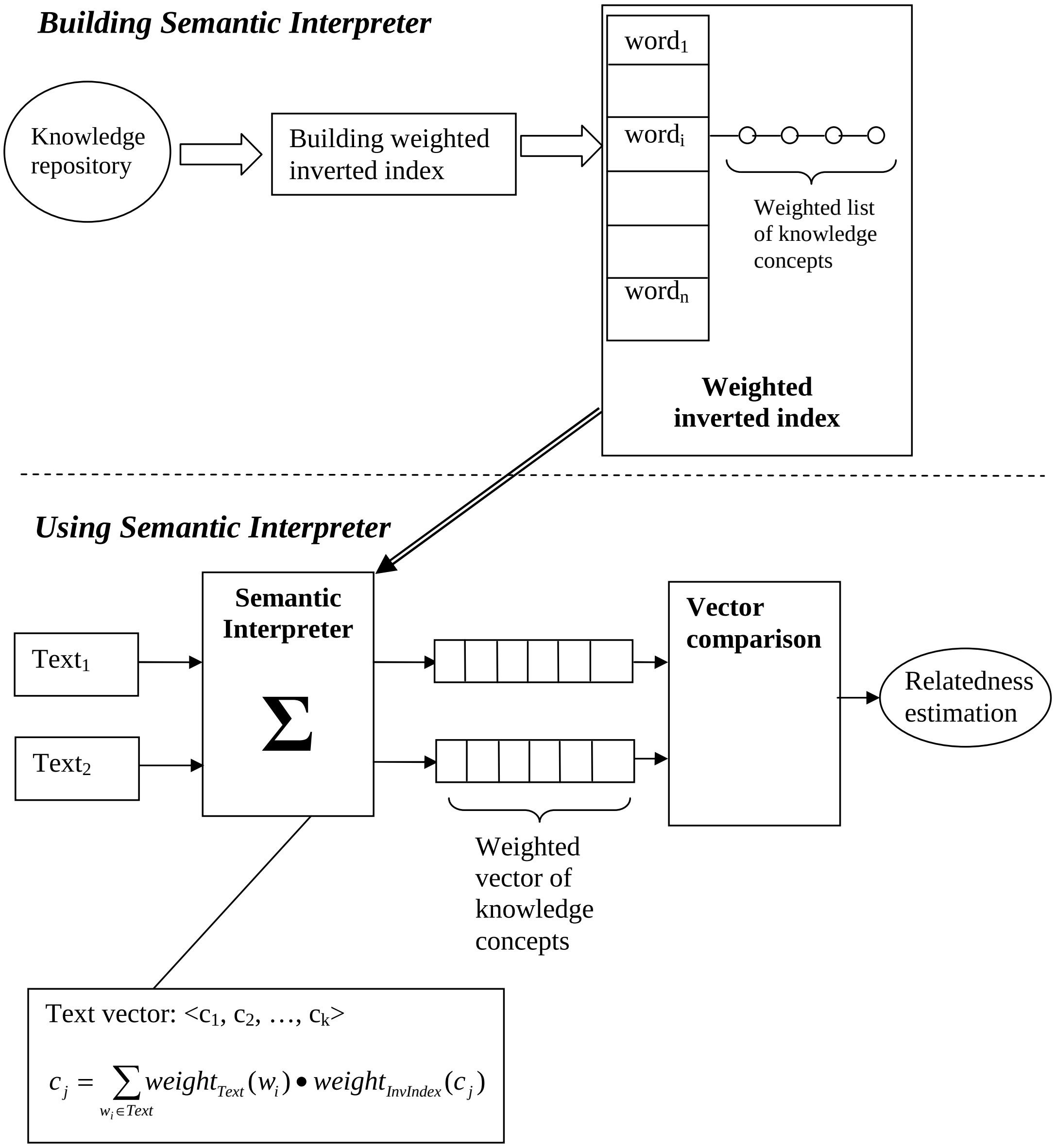
Building Semantic Interpreter
Knowledge repository
Building weighted inverted index
word1
wordi
wordn
Weighted list of knowledge concepts
Weighted inverted index
Using Semantic Interpreter
Text1
Text2
Semantic Interpreter
Σ
Weighted vector of knowledge concepts
Vector comparison
Relatedness estimation
Text vector: <c1, c2, …, ck>
$c_j = \sum_{w_i \in Text} weight_{Text}(w_i) \bullet weight_{InvIndex}(c_j)$

Figure 6.1: Knowledge-based semantic interpreter

6.2.1 Test Collections

In this work, we use two datasets that to the best of our knowledge are the largest publicly available collections of their kind.[1] To assess word relatedness, we use

[1] Recently, Zesch and Gurevych (2006) discussed automatic creation of datasets for assessing semantic similarity. However, the focus of their work was on automatic generation of a set of

the WordSimilarity-353 collection (Finkelstein et al., 2002b; Finkelstein et al., 2002a), which contains 353 word pairs. Each pair has 13–16 human judgements made by individuals with university degrees having either mother-tongue-level or otherwise very fluent command of the English language. Word pairs were assigned relatedness scores on the scale from 0 (totally unrelated words) to 10 (very much related or identical words). Judgements collected for each word pair were then averaged to produce a single relatedness score.

For document similarity, we used a collection of 50 documents from the Australian Broadcasting Corporation's news mail service (Lee, Pincombe, and Welsh, 2005; Pincombe, 2004). The documents were between 51 and 126 words long, and covered a variety of topics. The judges were 83 students from the University of Adelaide, Australia, who were paid a small fee for their work. These documents were paired in all possible ways, and each of the 1,225 pairs has 8–12 human judgements (averaged for each pair). To neutralize the effects of ordering, document pairs were presented in random order, and the order of documents within each pair was randomized as well.

Importantly, instructions for human judges in both test collections specifically directed the participants to assess the *degree of relatedness* of words and texts involved. For example, in the case of antonyms, judges were instructed to consider them as “similar” rather than “dissimilar”.

For both test collections, we use the correlation of computer-assigned scores with human scores to assess the algorithm performance.

6.2.2 The Effect of External Knowledge

Table 6.1 shows the results of applying our methodology to estimating relatedness of individual words. As we can see, both ESA techniques yield substantial improvements over previous state of the art results. Notably, ESA also achieves much better results than another recently introduce method based on Wikipedia (Strube and Ponzetto, 2006). We provide a detailed comparison of our approach with this latter work in Section 7.3. Table 6.2 shows the results for computing relatedness of entire documents.

In Section 5.4.3 we examined the effect of knowledge breadth by comparing feature generators based on two Wikipedia versions. Here we also evaluate the benefits of using a larger knowledge base for ESA. As we can see in both experiments, using a newer Wikipedia snapshot leads to better results (although the difference between the performance of two versions is admittedly small).

sufficiently diverse word pairs, thus relieving the humans of the need to construct word lists manually. Obviously, establishing the “gold standard” semantic relatedness for each word pair is still performed manually by human judges.

Algorithm	Correlation with human judgements
WordNet-based techniques (Jarmasz, 2003)	0.33–0.35
Roget's Thesaurus-based technique (Jarmasz, 2003)	0.55
LSA (Finkelstein et al., 2002a)	0.56
WikiRelate! (Strube and Ponzetto, 2006)	0.19–0.48
ESA-Wikipedia (March 26, 2006 version)	0.75
ESA-Wikipedia (November 11, 2005 version)	0.74
ESA-ODP	0.65

Table 6.1: Correlation of word relatedness scores with human judgements on the WordSimilarity-353 collection

Algorithm	Correlation with human judgements
Bag of words (Lee, Pincombe, and Welsh, 2005)	0.1–0.5
LSA (Lee, Pincombe, and Welsh, 2005)	0.60
ESA-Wikipedia (March 26, 2006 version)	0.72
ESA-Wikipedia (November 11, 2005 version)	0.71
ESA-ODP	0.69

Table 6.2: Correlation of text relatedness scores with human judgements on Lee et al.'s document collection

On both test collections, Wikipedia-based semantic interpretation is superior to the ODP-based one. We believe that two factors contribute to this phenomenon. First, axes of a multi-dimensional interpretation space should ideally be as independent as possible. The hierarchical organization of the Open Directory reflects the generalization relation between concepts and obviously violates this independence requirement. Second, to increase the amount of training data for building the ODP-based semantic interpreter, we crawled all the URLs listed in the ODP. This allowed us to increase the amount of textual data by several orders of magnitude, but also brought about a non-negligible amount of noise, which is common in Web pages. On the other hand, Wikipedia articles are virtually noise-free, and mostly qualify as Standard Written English. Thus, the textual descriptions of Wikipedia concepts are arguably more focused than those of the ODP concepts.

Chapter 7

Related work

This section puts our methodology in the context of related prior work.

7.1 Beyond the Bag of Words

To date, quite a few attempts have been made to deviate from the orthodox bag of words paradigm, usually with limited success. In particular, representations based on phrases (Lewis, 1992a; Dumais et al., 1998; Fuernkranz, Mitchell, and Riloff, 2000), named entities (Kumaran and Allan, 2004), and term clustering (Lewis and Croft, 1990; Bekkerman, 2003) have been explored. However, none of these techniques could possibly overcome the problem underlying the various examples we reviewed in this paper—lack of world knowledge.

In mainstream Information Retrieval, query expansion techniques are used to augment queries with additional terms. However, this approach does not enhance queries with high-level concepts beyond words or phrases. It occasionally uses WordNet (Fellbaum, 1998) as a source of external knowledge, but queries are more often enriched with individual words, which are chosen through relevance feedback (Mitra, Singhal, and Buckley, 1998; Xu and Croft, 2000), by consulting dictionaries and thesauri (Voorhees, 1994; Voorhees, 1998), or by analyzing the context around the query term (Finkelstein et al., 2002a). Ballesteros and Croft (1997) studied query expansion with phrases in the context of cross-lingual information retrieval.

7.2 Feature Generation for Text Categorization

Feature generation techniques were found useful in a variety of machine learning tasks (Markovitch and Rosenstein, 2002; Fawcett, 1993; Matheus, 1991). These techniques search for new features that describe the target concept better than the ones supplied with the training instances. A number of proposed feature generation algorithms (Pagallo and Haussler, 1990; Matheus and Rendell, 1989; Hu and Kibler, 1996; Murphy and Pazzani, 1991) led to significant improvements in performance over a range of classification tasks. However, even though feature generation is an established research area in machine learning, only a few works have applied it to text processing (Kudenko and Hirsh, 1998; Mikheev, 1999; Cohen, 2000; Scott, 1998). In contrast to our approach, these techniques did not use any exogenous knowledge.

Kudenko and Hirsh (1998) proposed a domain-independent feature generation algorithm that uses Boolean features to test whether certain sub-sequences appear a minimum number of times. They applied the algorithm to three toy problems in topic spotting and book passage categorization.

Mikheev (1999) used a feature collocation lattice as a feature generation engine within maximum entropy framework, and applied it to document categorization, sentence boundary detection and part-of-speech tagging. This work utilized information about individual words, bigrams and trigrams to pre-build the feature space, and then selected a set of feature cliques with the highest log-likelihood estimate.

Cohen (2000) conducted research on the following problem: given a set of labeled instances *not accompanied by a feature set*, is it possible to automatically discover features useful for classification according to the given labels? Problems of this kind occur, for example, when classifying names of musical artists by music genres, or names of computer games by categories such as quest or action. The paper proposed to collect relevant Web pages, and then define features based on words from HTML headers that co-occur with the names to be classified. The fact that a word appears in an HTML header usually signifies its importance, and hence potential usefulness for classification. The author also identified another source of features based on *positions* inside HTML documents, where position is defined as a sequence of tags in the HTML parsing tree, between the root of the tree and the name of interest. For example, if a name frequently appears inside tables, this characteristic may be defined as a feature.

Fuhr (1985) introduced the *Darmstadt Indexing Approach (DIA)*, which defines features as *properties* of terms, documents or categories, rather than mere terms or phrases. Thus, meta information such as positions of words within documents, document lengths or the cardinality of category training sets may all be considered as features. Sebastiani (2002) notes that the DIA allows uniform usage of these new features along with conventional term- or phrase-based representations.

Bekkerman et al. (2001) represented documents by word clusters rather than by individual words, within the framework of the *information bottleneck* approach (Pereira, Tishby, and Lee, 1993; Tishby, Pereira, and Bialek, 1999). The resulting clusters were then used as new features that replaced the original words.

7.2.1 Feature Generation Using Electronic Dictionaries

Several studies performed feature construction using the WordNet electronic dictionary (Fellbaum, 1998) and other domain-specific dictionaries (Scott, 1998; Urena-Lopez, Buenaga, and Gomez, 2001; Wang et al., 2003; Bloehdorn and Hotho, 2004).

Scott (1998) attempted to augment the conventional bag-of-words representation with additional features, using the symbolic classification system RIPPER (Cohen, 1995). This study evaluated features based on syntactically[1] and statistically motivated phrases, as well as on WordNet *synsets*[2]. In the latter case, the system performed generalizations using the hypernym hierarchy of WordNet, and completely replaced a bag of words with a bag of synsets. While using hypernyms allowed RIPPER to produce more general and more comprehensible rules and achieved some performance gains on small classification tasks, no performance benefits could be obtained for larger tasks, which even suffered from some degradation in classification accuracy. Consistent with other published findings (Lewis, 1992a; Dumais et al., 1998; Fuernkranz, Mitchell, and Riloff, 2000), the phrase-based representation also did not yield any significant performance benefits over the bag-of-words approach.[3]

Urena-Lopez, Buenaga, and Gomez (2001) used WordNet in conjunction with Rocchio (Rocchio, 1971) and Widrow-Hoff (Lewis et al., 1996; Widrow and Stearns, 1985, Ch. 6) linear classifiers to fine-tune the category vectors. Wang et al. (2003) used Medical Subject Headings (MeSH) (MeSH, 2003) to replace the bag of words with canonical medical terms; Bloehdorn and Hotho (2004) used a similar approach to augment Reuters-21578 documents with WordNet synsets and OHSUMED medical documents with MeSH terms.

It should be noted, however, that WordNet was not originally designed to be a powerful knowledge base, but rather a lexical database more suitable for peculiar lexicographers' needs. Specifically, WordNet has the following drawbacks when used as a knowledge base for text categorization:

- WordNet has a fairly small coverage—for the test collections we used in this paper, up to 50% of their unique words are missing from WordNet. In

[1] Identification of syntactic phrases was performed using a noun phrase extractor built on top of a part of speech tagger (Brill, 1995).

[2] A *synset* is WordNet notion for a sense shared by a group of synonymous words.

[3] Sebastiani (2002) casts the use of bag of words versus phrases as utilizing *lexical semantics* rather than *compositional semantics*. Interestingly, some bag-of-words approaches (notably, KNN) may be considered *context-sensitive* as they do not assume independence between either features (terms) or categories (Yang and Pedersen, 1997).

particular, many proper names, slang and domain-specific technical terms are not included in WordNet, which was designed as a general-purpose dictionary.

- Additional information about synsets (beyond their identity) is very limited. This is because WordNet implements a *differential* rather than *constructive* lexical semantics theory, so that glosses that accompany the synsets are mainly designed to distinguish the synsets rather than provide a definition of the sense or concept. Usage examples that occasionally constitute part of the gloss serve the same purpose. Without such auxiliary information, reliable word sense disambiguation is almost impossible.

- WordNet was designed by professional linguists who are trained to recognize minute differences in word senses. As a result, common words have far too many distinct senses to be useful in information retrieval (Mihalcea, 2003); for example, the word "make" has as many as 48 senses as a verb alone. Such fine-grained distinctions between synsets present an additional difficulty for word sense disambiguation.

Both our approach and the techniques that use WordNet manipulate a collection of concepts. However, there are a number of crucial differences. All previous studies only performed feature generation for individual words only. Our approach can handle arbitrarily long or short text fragments alike. Considering words in context allows our approach to perform word sense disambiguation. Approaches using WordNet cannot achieve disambiguation because information about synsets is limited to merely a few words, while in both the ODP and Wikipedia concepts are associated with huge amounts of text. Even for individual words, our approach provides much more sophisticated mapping of words to concepts, through the analysis of the large bodies of texts associated with concepts. This allows us to represent the meaning of words (or texts) as a weighted combination of concepts, while mapping a word in WordNet amounts to simple lookup, without any weights. Furthermore, in WordNet the senses of each word are mutually exclusive. In our approach, concepts reflect different aspects of the input, thus yielding weighted multi-faceted representation of the text.

In the next section we illustrate the limitations of WordNet on two specific examples.

7.2.2 Comparing Knowledge Sources for Feature Generation: ODP versus WordNet

To demonstrate the shortcomings of WordNet as a source for knowledge-based feature generation, we juxtapose WordNet-based and ODP-based feature generation for two sample sentences examined in Section 5.3.1 (we repeat the ODP context classifications for readers' convenience). We opted to compare WordNet with the Open Directory, since both knowledge repositories are hierarchically organized.

We used WordNet version 1.6. In what follows, WordNet synsets are denoted with curly braces, and noun and verb synsets are followed by their immediate hypernym (more general synset), if applicable.

- **Text:** *"Rumsfeld appeared with Gen. Richard Myers, chairman of the Joint Chiefs of Staff."*

  ODP classifications:

  - Society/Issues/Government_Operations
  - Society/Politics
  - Society/Issues/Warfare_and_Conflict/Specific_Conflicts/Iraq
  - Science/Technology/Military_Science
  - Society/Issues/Warfare_and_Conflict/Weapons
  - Society/History/By_Region/North_America/United_States/Presidents/Bush,_George_Walker
  - Society/Politics/Conservatism

  WordNet classifications:

  - {**Rumsfeld**} → { }; *(word not present in WordNet)*
  - {look, **appear**, seem} → {be}; {**appear**}; {**appear**, come out} → {happen, materialize}; {**appear**, seem} → {be}; {**appear**, come along}; {**appear**} → {perform, execute, do}
  - {**Gen**} → {information, info}
  - {**Richard**} → { }; *(word not present in WordNet)*
  - {**Myers**} → { }; *(word not present in WordNet)*
  - {president, **chairman**, chairwoman, chair, chairperson} → {presiding officer}; {chair, **chairman**} → {head, lead}

- {**joint**, articulation, articulatio} → {body part}; {**joint**} → {spot}; {articulation, join, **joint**, juncture, junction} → {connection, connexion, link}; {roast, **joint**} → {cut, cut of meat}; {**joint**} → {junction, conjunction}; {**joint**, marijuana cigarette, reefer, stick} → {cigarette, cigaret, coffin nail, butt, fag}
- {**joint**} → {fit, go}; {**joint**, articulate} → {supply, provide, render, furnish}; {**joint**} → {fasten, fix, secure}
- {**joint** (vs. separate)}; {joint}
- {head, **chief**, top dog} → {leader}; {foreman, **chief**, gaffer, honcho, boss} → {supervisor}
- {**staff**} → {force, personnel}; {**staff**} → {stick}; {**staff**, faculty} → {body}; {**staff**} → {symbol}; {**staff**, stave} → {musical notation}
- {**staff**} → {provide, supply, ply, cater}

- **Text:** *"Herceptin is a so-called targeted therapy because of its ability to attack diseased cells and leave healthy ones alone."*

  ODP classifications:

  - Health/Conditions_and_Diseases/Cancer/Breast
  - Society/Issues/Health/Conditions_and_Diseases/Cancer/Alternative_Treatments
  - Health/Support_Groups/Conditions_and_Diseases/Cancer

  WordNet classifications:

  - {**Herceptin**} → { }; *(word not present in WordNet)*
  - {alleged (prenominal), **so-called**, supposed} → {questionable (vs. unquestionable)}
  - {**target**, aim, place, direct, point} → {aim, take, train, take aim, direct}
  - {**therapy**} → {medical care, medical aid}
  - {**ability**} → {quality}
  - {**ability**, power} → {cognition, knowledge}

- {**attack**, onslaught, onset, onrush} → {operation}; {**attack**} → {turn, play}; {fire, **attack**, flak, blast} → {criticism, unfavorable judgment}; {approach, **attack**, plan of attack} → { conceptualization, conceptualisation, formulation, formularizing, formularising}; {**attack**, attempt} → {battery, assault, assault and battery}; {**attack**, tone-beginning} → {beginning, start, commencement}; {**attack**} → {affliction}; {**attack**, assault} → {attention, attending};
- {**attack**, assail} → {fight, struggle}; {**attack**, round, assail, lash out, snipe, assault} → {criticize, criticise, pick apart}; {**attack**, aggress} → {act, move}; {assail, assault, set on, **attack**}; {**attack**} → {begin, get, start out, start, set about, set out, commence}; {**attack**} → {affect}
- {assault (prenominal), **attack** (prenominal)} → {offensive (vs. defensive)};
- {**diseased**, morbid, pathologic, pathological} → {unhealthy (vs. healthy)};
- {**cell**} → {compartment}; {**cell**} → {entity, something}; {**cell**, electric cell} → {electrical device}; {**cell**, cadre} → {political unit}; {**cell**, cubicle} → {room}; {**cell**, jail cell, prison cell} → {room}
- {**leave**, leave of absence} → {time off}; {**leave**} → {permission}; {farewell, **leave**, leave-taking, parting} → {departure, going, going away, leaving};
- {**leave**, go forth, go away}; *(16 more verb senses omitted for brevity)*
- {**healthy** (vs. unhealthy)}; {**healthy**} → {sound (vs. unsound)}; {**healthy**, salubrious, good for you (predicate)} → {wholesome (vs. unwholesome)}; {fit (vs. unfit), **healthy**} → {able, able-bodied}; {**healthy**, intelligent, levelheaded, sound} → {reasonable (vs. unreasonable), sensible};
- {**one**, 1, I, ace, single, unity} → {digit}; {**one**} → {unit}
- {**alone** (predicate)} → {unsocial (vs. social)}; {**alone** (predicate), lone (prenominal), lonely (prenominal), solitary} → {unaccompanied (vs. accompanied)}; {**alone** (predicate), only} → {exclusive (vs. inclusive)}; {**alone** (predicate), unique, unequaled, unequalled, unparalleled} → {incomparable (vs. comparable), uncomparable}
- {entirely, exclusively, solely, **alone**, only}; {**alone**, unaccompanied}

Evidently, WordNet classifications are overly general and diverse because context words cannot be properly disambiguated. Furthermore, owing to lack of proper names, WordNet cannot possibly provide the wealth of information encoded in the Open Directory, which easily overcomes the drawbacks of WordNet. Crawling all the Web sites cataloged in the Open Directory results in exceptionally wide word coverage. Furthermore, the crawled texts provide a plethora of information about each ODP concept.

7.2.3 Using Unlabeled Examples

To the best of our knowledge, with the exception of the above studies that used WordNet, there have been no attempts to date to automatically use large-scale repositories of structured background knowledge for feature generation. An interesting approach to using non-structured background knowledge was proposed by Zelikovitz and Hirsh (2000). This work uses a collection of unlabeled examples as intermediaries in comparing testing examples with the training ones. Specifically, when an unknown test instance does not appear to resemble any labeled training instances, unlabeled examples that are similar to both may be used as "bridges." Using this approach, it is possible to handle the situation where the training and the test document have few or no words in common. The unlabeled documents are utilized to define a cosine similarity metric, which is then used by the KNN algorithm for actual text categorization. This approach, however, suffers from efficiency problems, as looking for intermediaries to compare every two documents makes it necessary to explore a combinatorial search space.

In a subsequent paper, Zelikovitz and Hirsh (2001) proposed an alternative way to use unlabeled documents as background knowledge. In this work, unlabeled texts are pooled together with the training documents to compute a Latent Semantic Analysis (Deerwester et al., 1990) model. LSA analyzes a large corpus of unlabeled text, and automatically identifies so-called "latent concepts" using Singular Value Decomposition. The resulting LSA metric then facilitates comparison of test documents to training documents. The addition of unlabeled documents significantly increases the amount of data on which word cooccurrence statistics is estimated, thus providing a solution to text categorization problems where training data is particularly scarce. However, subsequent studies found that LSA can rarely improve the strong baseline established by SVM, and often even results in performance degradation (Wu and Gunopulos, 2002; Liu et al., 2004). In contrast to LSA, which manipulates virtual concepts, our methodology relies on using concepts identified and described by humans.

In Section 6.2 we reported the results of applying our methodology to the problem of computing semantic relatedness of words and texts, for which previous state of the art results have been based on LSA. To this end, we formulated Explicit Semantic Analysis (ESA), which represents fragments of text in the space of knowledge concepts defined in the Open Directory or in Wikipedia. Compared with the existing state of the art, using ESA resulted in substantial improvements in correlation of computed relatedness scores with human judgments. These findings prove that the benefits of using distilled human knowledge are much greater than merely using cooccurrence statistics gathered from a collection of auxiliary unlabeled texts.

7.2.4 Other Related Studies

There have been numerous previous attempts to add knowledge to machine learning techniques. Transfer learning approaches (Bennett, Dumais, and Horvitz, 2003; Do and Ng, 2005; Raina, Ng, and Koller, 2006) leverage information from different but related learning tasks. Pseudo-relevance feedback (Ruthven and Lalmas, 2003) uses information from the top-ranked documents, which are assumed to be relevant to the query; for example, characteristic terms from such documents may be used for query expansion (Xu and Croft, 1996). Recent studies on semi-supervised methods (Goldberg and Zhu, 2006; Ando and Zhang, 2005a; Ando and Zhang, 2005b) infer information from unlabeled data, which is often available in much larger amounts than labeled data. However, all these approaches amount to using shallow cooccurrence-style knowledge. On the other hand, the methods we propose in this thesis use much deeper knowledge cataloged by humans, which comes in the form of concepts that correspond to the nodes of the Open Directory or to the articles of Wikipedia.

While our approach relies on existing repositories of classified knowledge, there is a large body of research on extracting facts through Web mining (Cafarella et al., 2005; Etzioni et al., 2004), so it would be interesting to consider using such extracted facts to drastically increase the amount of available knowledge, especially when measures are taken to ascertain correctness of the extracted information (Downey, Etzioni, and Soderland, 2005).

Our use of local contexts to facilitate fine-grained feature generation is reminiscent of the intra-document dynamics analysis proposed by Gabrilovich, Dumais, and Horvitz (2004) for characterization of news article types. The latter work manipulated sliding contextual windows of the same size to make their scores directly comparable. As we showed in Section 5.3.3, the multi-resolution approach, which operates at several levels of linguistic abstraction, is superior to fixed-size windows for the case of text categorization. Incidentally, the term “Local Context Analysis” is also used in an entirely different branch of Information Retrieval. Xu and Croft (2000) used this term to refer to a particular kind of query expansion, where a query is expanded *in the context* of top-ranked retrieved documents.

In our methodology, the feature generator is implemented as a text classifier that maps local document contexts onto knowledge concepts, which then serve as additional document features. This approach is similar to the use of classifiers as features. Bennett, Dumais, and Horvitz (2005) used the reliability-indicator methodology (Toyama and Horvitz, 2000) to combine several regular text classifiers (decision tree, SVM, Naive Bayes and unigram) with the aid of a meta-classifier.

Our approach that uses structured background knowledge is somewhat reminiscent of explanation-based learning (Mitchell, Keller, and Kedar-Cabelli, 1986; Dejong and Mooney, 1986), where generalizations of previously seen examples are reused in future problem solving tasks, thus mimicking humans' ability to learn from a single example.

7.3 Semantic Similarity and Semantic Relatedness

In this thesis we dealt with "semantic relatedness" rather than "semantic similarity" or "semantic distance", which are also often used in the literature. In their extensive survey of relatedness measures, Budanitsky and Hirst (2006) argued that the notion of relatedness is more general than that of similarity, as the former subsumes many different kind of specific relations, including meronymy, antonymy, functional association, and others. They further maintained that computational linguistics applications often require measures of relatedness rather than the more narrowly defined measures of similarity. For example, word sense disambiguation can use any *related* words from the context, and not merely *similar* words. Budanitsky and Hirst (2006) also argued that the notion of semantic distance might be confusing due to the different ways it has been used in the literature.

Prior work on computing semantic relatedness pursued three main directions: comparing text fragments as bags of words in vector space (Baeza-Yates and Ribeiro-Neto, 1999; Rorvig, 1999), using lexical resources, and using Latent Semantic Analysis (Deerwester et al., 1990). The former technique is the simplest, but performs sub-optimally when the texts to be compared share few words, for instance, when the texts use synonyms to convey similar messages. This technique is also trivially inappropriate for comparing individual words. The latter two techniques attempt to circumvent this problem.

Lexical databases such as WordNet (Fellbaum, 1998) or Roget's Thesaurus (Roget, 1852) encode relations between words such as synonymy, hypernymy, and meronymy. Quite a few metrics have been defined that compute relatedness using various properties of the underlying graph structure of these resources (Budanitsky and Hirst, 2006; Jarmasz, 2003; Resnik, 1999; Lin, 1998; Jiang and Conrath, 1997). The obvious drawback of this approach is that creation of lexical resources requires lexicographic expertise as well as a lot of time and effort, and consequently such resources cover only a small fragment of the language lexicon. Specifically, such resources contain few proper names, neologisms, slang, and domain-specific technical terms. Furthermore, these resources have strong lexical orientation in that they predominantly contain information about individual words, but little world knowledge in general.

On the other hand, LSA (Deerwester et al., 1990) is a purely statistical technique, which leverages word cooccurrence information from a large unlabeled corpus of text. LSA does not rely on any human-organized knowledge; rather, it "learns" its representation by applying Singular Value Decomposition to the words-by-documents cooccurrence matrix. LSA is essentially a dimensionality reduction technique that identifies a number of most prominent dimensions in the data, which are assumed to correspond to "latent concepts". Meanings of words and documents are then compared in the space defined by these concepts. Latent semantic models are notoriously difficult to interpret, since the computed concepts cannot be readily mapped into natural concepts manipulated by humans. The Explicit Semantic Analysis method we proposed circumvents this problem, as it represents meanings of text fragments using natural concepts defined by humans.

Our approach to estimating semantic relatedness of words is somewhat reminiscent of distributional (or co-occurrence) similarity (Lee, 1999; Dagan, Lee, and Pereira, 1999). Indeed, we compare the meanings of words by comparing the occurrence patterns across a large collection of natural language documents. However, the compilation of these documents is not arbitrary, rather, the documents are aligned with encyclopedia articles, while each of them is focused on a single topic. Furthermore, distributional similarity methods are inherently suitable for comparing individual words, while our method can compute similarity of arbitrarily long texts.

Prior work in the field mostly focused on semantic *similarity* of words, using R&G (Rubenstein and Goodenough, 1965) list of 65 word pairs and M&C (Miller and Charles, 1991) list of 30 word pairs. When only the similarity relation is considered, using lexical resources was often successful enough, reaching the correlation of 0.70–0.85 with human judgements (Budanitsky and Hirst, 2006; Jarmasz, 2003). In this case, lexical techniques even have a slight edge over ESA-Wikipedia, whose correlation with human scores is 0.723 on M&C and 0.816 on R&G.[4] However, when the entire language wealth is considered in an attempt to capture more general semantic relatedness, lexical techniques yield substantially inferior results (see Table 6.1). WordNet-based techniques, which only consider the generalization ("is-a") relation between words, achieve correlation of only 0.33–0.35 with human judgements (Budanitsky and Hirst, 2006; Jarmasz, 2003).

[4] WikiRelate! (Strube and Ponzetto, 2006) achieved relatively low scores of 0.31–0.54 on these domains.

Jarmasz & Szpakowicz's ELKB system (Jarmasz, 2003) based on Roget's Thesaurus (Roget, 1852) achieves a higher correlation of 0.55 due to its use of a richer set of relations.

Studying semantic similarity and relatedness of words is related to assessing the similarity of relations. An example of this task is to establish that word pairs *carpenter:wood* and *mason:stone* are relationally similar, as the words in both pairs stand in the same relation (profession:material). State of the art results on relational similarity are based on Latent Relational Analysis (Turney, 2006; Turney, 2005).

Sahami and Heilman (2006) proposed to use the Web as a source of additional knowledge for measuring similarity of short text snippets. To this end, they defined a kernel function that sends two snippets as queries to a search engine, and compares the bags of words for the two sets of returned documents. A major limitation of this technique is that it is only applicable to short texts, because sending a long text as a query to a search engine is likely to return few or even no results at all. On the other hand, our approach is applicable to text fragments of arbitrary length.

The above-mentioned WordNet-based techniques are inherently limited to individual words, and their adaptation for comparing longer texts requires an extra level of sophistication (Mihalcea, Corley, and Strapparava, 2006). In contrast, our method treats both words and texts in essentially the same way.

A recent paper by Strube and Ponzetto (2006) also used Wikipedia for computing semantic relatedness. However, their method, called WikiRelate!, is radically different from ours. Given a pair of words w_1 and w_2, WikiRelate! searches for Wikipedia articles, p_1 and p_2, that respectively contain w_1 and w_2 in their titles. Semantic relatedness is then computed based on various distance measures between p_1 and p_2. These measures either rely on the texts of the pages, or path distances within the category hierarchy of Wikipedia. Our approach, on the other hand, represents each word as a weighted vector of Wikipedia concepts. Semantic relatedness is then computed by comparing the two concept vectors.

Thus, the differences between the two approaches are:

1. WikiRelate! can only process words that actually occur in titles of Wikipedia articles. ESA only requires that the word appears within the text of Wikipedia articles.

2. WikiRelate! is limited to single words while ESA can compare texts of any length.

3. WikiRelate! represents the semantics of a word by either the text of the article associated with it, or by the node in the category hierarchy. ESA

has a much more structured semantic representation consisting of a vector of Wikipedia concepts.

Indeed, as we have shown in Section 6.2, the richer representation of ESA yields much better results.

Chapter 8

Conclusions

In this thesis we proposed a feature generation methodology for textual information retrieval. In order to render machine learning algorithms with common-sense and domain-specific knowledge of humans, we use very large-scale knowledge repositories to build a *feature generator*. These knowledge repositories, which have been manually crafted by human editors, provide a fully automatic way to tap into the collective knowledge of tens and hundreds of thousands of people. The feature generator analyzes document text and augments the conventional bag of words representation with relevant concepts from the knowledge repository. The enriched representation contains information that cannot be deduced from the document text alone.

In Section 2.2 we listed several limitations of the bag of words approach, and in the subsequent sections we showed how they are resolved by our methodology. In particular, external knowledge allows us to reason about words that appear in the testing set but not in the training set. We use hierarchically organized knowledge to make powerful generalizations, making it possible to know that certain infrequent words belong to more general classes of words. Externally supplied knowledge can also help in those cases when some information vital for classification is omitted from training texts because it is assumed to be shared by the target readership.

In this work we instantiated our feature generation methodology with two specific knowledge repositories, the Open Directory Project and the Wikipedia encyclopedia. We succeeded to make use of an encyclopedia without deep language understanding, specially crafted inference rules or relying on additional commonsense knowledge bases. This was made possible by applying standard text classification techniques to match document texts with relevant Wikipedia articles. The Wikipedia-based results are superior to the ODP-based ones on a number of datasets, and are comparable to it on others. Moreover, using Wikipedia imposes fewer restrictions on suitable knowledge repositories, and does not assume the availability of an ontology. In our future work, we intend to study possible ways for combining two or more knowledge repositories for improving text categorization performance even further.

We also described multi-resolution analysis, which examines the document text at several levels of linguistic abstraction and performs feature generation at each level. When polysemous words are considered in their native context, word sense disambiguation is implicitly performed. Considering local contexts allows the feature generator to cope with word synonymy and polysemy. Furthermore, when the document text is processed at several levels of granularity, even briefly mentioned aspects can be identified and used. These might easily have been overlooked if the document were processed as one large chunk of text.

Empirical evaluation definitively confirmed the value of knowledge-based feature generation for text categorization across a range of datasets. Recently, the performance of the best text categorization systems became similar, as if a plateau has been reached, and previous work mostly achieved small improvements. Using the ODP and Wikipedia allowed us to reap much greater benefits and to bring text categorization to a qualitatively new level of performance, with double-digit improvements observed on a number of datasets. Given the domain-specific nature of some test collections, we also compared the utility of narrow domain-specific knowledge with that of larger amounts of information covering all branches of knowledge (Section 5.3.4). Perhaps surprisingly, we found that even for narrow-scope test collections, a wide coverage knowledge base yielded substantially greater improvements than its domain-specific subsets. This observation reinforces the *breadth hypothesis*, formulated by Lenat and Feigenbaum (1990), that "to behave intelligently in unexpected situations, an agent must be capable of falling back on increasingly general knowledge."

We also applied our feature generation methodology to the problem of automatically assessing semantic relatedness of words and texts. To this end, we presented a novel technique, called Explicit Semantic Analysis, for representing semantics of natural language texts using natural concepts. In contrast to existing methods, ESA offers a uniform way for computing relatedness of both individual words and arbitrarily long text fragments. Moreover, using natural concepts makes the ESA model easy to interpret, as can be seen in the examples we provided. Compared with the previous state of the art, using ESA results in substantial improvements in correlation of computed relatedness scores with human judgements: from $r = 0.56$ to 0.75 for individual words and from $r = 0.60$ to 0.72 for texts. Consequently, we anticipate ESA to give rise to the next generation of natural language processing tools.

We believe that this research constitutes a step towards the long-standing aspiration of Artificial Intelligence to endow natural language processing with humans' knowledge about the world. However, our study only scratches the surface of what can be achieved with knowledge-rich features. In our future work, we plan to investigate new algorithms for mapping document contexts onto knowledge concepts, as well as new techniques for selecting attributes that are most characteristic of every concept. We intend to apply focused crawling to collect only relevant Web pages when cataloged URLs are crawled; we also plan to apply page segmentation techniques to eliminate noise from crawled pages (Yu et al., 2003). In this work we capitalized on inter-article links of Wikipedia in several ways, including the use of anchor text and the number of incoming links for each article, as well as creating additional features from linked concepts. In our future work we intend to investigate more elaborate techniques for leveraging the high degree of cross-linking between Wikipedia articles.

The Wiki technology underlying the Wikipedia project is often used nowadays in a variety of open-editing initiatives. These include corporate intranets that use Wiki as a primary documentation tool, as well as numerous domain-specific encyclopedias on topics ranging from mathematics to Orthodox Christianity.[1] Therefore, we believe our methodology may be used for augmenting document representation in domains for which no ontologies exist. It is also essential to note that Wikipedia is available in numerous languages, while different language versions are cross-linked at the level of concepts. We believe this information can be leveraged to use Wikipedia-based semantic interpretation for improving machine translation.

In addition to the ODP and Wikipedia, we also plan to make use of additional knowledge repositories. Among domain-specific knowledge bases, it would be particularly interesting to use the Medical Subject Headings (MeSH) hierarchy to improve classification of biomedical documents. Recently, several projects have been launched that intend to digitize large numbers of books. The largest and arguably best known among these projects are Google Print[2] and Amazon's Search Inside the Book[3]. If the content of numerous books is made available for research purposes, it would be extremely interesting to use their text in conjunction with one of the library classification schemes (e.g., UDC) to build a book-based feature generator.

Over the recent years, collaborative tagging projects (also known as folksonomies) became widespread on the Internet (Marlow et al., 2006). We believe it would be very interesting to use the data accumulated through these tagging efforts as knowledge sources for feature generation. This way, we would use tags as concepts, and the tagged textual objects (such as blog postings and Web pages) as material for learning the scope of these concepts.

[1] See `http://en.wikipedia.org/wiki/Category:Online_encyclopedias` for a longer list of examples.

[2] `http://books.google.com`

[3] `http://www.amazon.com/Search-Inside-Book-Books/b?ie=UTF8&node=10197021`

We conjecture that knowledge-based feature generation will also be useful for other information retrieval tasks beyond text categorization, and we intend to investigate this in our future work. Specifically, we intend to apply feature generation to information search and word sense disambiguation. In the search scenario, we are studying ways to augment both queries and documents with generated features. This way, documents will be indexed in the augmented space of words and concepts. Current approaches to word sense disambiguation represent contexts that contain ambiguous words using the bag of words augmented with part-of-speech information. We believe representation of such contexts can be greatly improved if we use feature generation to map these contexts into relevant knowledge concepts. Anecdotal evidence (such as the examples presented in Sections 5.3.1 and 5.4.1) implies our method has much promise for improving the state of the art in word sense disambiguation.

In its present form, our method can inherently be applied only for improving representation of textual documents. Indeed, to date we applied our feature generation methodology for improving the performance of text categorization and for computing semantic relatedness of texts. However, we believe our approach can also be applied beyond mere text, as long as the objects to be manipulated are accompanied with some textual description. As an example, consider a collection of medical records containing test results paired with narrative reports. Performing feature generation from narrative reports is likely to produce pertinent concepts that can be used for augmenting the original record. Indeed, prior studies (Hripcsak et al., 1995) showed that natural language processing techniques can be used to extract vital information from narrative reports in automated decision-support systems.

Appendix A

Text Categorization with Many Redundant Features: Using Aggressive Feature Selection to Make SVMs Competitive with C4.5

Text categorization algorithms usually represent documents as bags of words and consequently have to deal with huge numbers of features. Most previous studies found that the majority of these features are relevant for classification, and that the performance of text categorization with support vector machines peaks when no feature selection is performed. We describe a class of text categorization problems that are characterized with many *redundant* features. Even though most of these features are relevant, the underlying concepts can be concisely captured using only a few features, while keeping all of them has substantially detrimental effect on categorization accuracy. We develop a novel measure that captures feature redundancy, and use it to analyze a large collection of datasets. We show that for problems plagued with numerous redundant features the performance of C4.5 is significantly superior to that of SVM, while aggressive feature selection allows SVM to beat C4.5 by a narrow margin.

A.1 Introduction

Text categorization deals with assigning category labels to natural language documents. Categories come from a fixed set of labels, and each document may be assigned one or more categories. The absolute majority of works in the field employ the so-called "bag of words" approach and use plain language words as features (Sebastiani, 2002). Using a bag of words usually leads to an explosion in the number of features, so that even moderately-sized test collections often have thousands or even tens of thousands of features. In such high-dimensional spaces, feature selection (FS) is often necessary to reduce noise and avoid overfitting. Prior studies found support vector machines (SVM) and K-Nearest Neighbor (KNN) to be the best performing algorithms for text categorization (Dumais et al., 1998; Yang and Liu, 1999).

Joachims (1998) found that *support vector machines* are very robust even in the presence of numerous features, and further observed that the multitude of text features are indeed useful for text categorization. To substantiate this claim, Joachims used a Naive Bayes classifier with feature sets of increasing size, where features were first ordered by their discriminative capacity (using the information gain criterion), and then the most informative features were *removed*. The classifier trained on the remaining "low-utility" features performed markedly better than random labeling of documents with categories, thus implying that all features are relevant and should be used. These findings were later corroborated in more recent studies (Brank et al., 2002; Rogati and Yang, 2002) that observed either no improvement or even small degradation of SVM performance after feature selection. On the 20 Newsgroups collection (Lang, 1995), which is one of the standard text categorization datasets, feature selection significantly degrades the accuracy of SVM classification (Bekkerman, 2003) due to a very large and diversified vocabulary of newsgroup postings. Consequently, many later works using SVMs did not perform any feature selection at all (Leopold and Kindermann, 2002; Lewis et al., 2004).

In this paper we describe a class of text categorization problems that are characterized by many *redundant* features. The corresponding datasets were collected in the course of our prior work (Davidov, Gabrilovich, and Markovitch, 2004), where we proposed a methodology for parameterized generation of labeled datasets for text categorization based on the Open Directory Project (ODP). Further details on parameterized generation of labeled datasets can be found in Appendix B. In our present work we use a subset of 100 datasets whose categorization difficulty (as measured by baseline SVM accuracy) is evenly distributed from very easy to very hard. We observed that even though the datasets differ significantly in their difficulty, many of them are comprised of categories that can be told apart using a small number of words. For example, consider distinguishing the documents about Boulder, Colorado, from those about Dallas, Texas. A few proper names of local landmarks and a handful of words describing local industries and other peculiarities often suffice to distinguish texts about the two cities. Given these discriminators, other words add little differentiation power, and are therefore *redundant*. As we show in Section A.3, support vector machines—which are usually quite robust in the presence of many features—do not fare well when a few good discriminators are vastly outnumbered by features with little *additional differentiation power*.

We further demonstrate that on such datasets C4.5 significantly outperforms SVM and KNN, although the latter are usually considered substantially superior text classifiers. When no feature selection is performed, C4.5 constructs small decision trees that capture the concept much better then either SVM or KNN. Surprisingly, even when feature selection is optimized for each classifier, C4.5 formulates a powerful classification model, significantly superior to that of KNN and only marginally less capable than that of SVM. We also show the crucial importance of aggressive feature selection for this class of problems on a different document representation. In this experiment we extend the conventional bag of words with features constructed using the WordNet electronic dictionary by generalizing original words; again, SVM performance steadily increases as fewer features are selected.

To account for this phenomenon, we developed a novel measure that predicts feature redundancy in datasets. This measure analyzes the distribution of features by their information gain, and reliably predicts whether feature selection will be beneficial or harmful for a given dataset. Notably, computation of this measure does not require to actually build a classifier, nor to invoke it on a validation set to determine an optimal feature selection level.

The main contributions of this paper are threefold. First, we describe a class of text categorization problems that have many redundant features, and for which aggressive feature selection is essential to achieve decent level of SVM performance. The existence of such class of problems is in contrast to most of prior research in text categorization, which found the majority of features (except the rarest ones) to be relevant, and specifically beneficial for SVM classification. Second, we use two different feature sets to show that without an aggressive feature selection, SVM classification is substantially inferior to that of C4.5, which was previously shown to be a less capable text classifier. Finally, we develop a measure that, given a dataset, predicts whether feature selection would be beneficial for it. This measure performs outlier detection in the distribution of features by information gain, without actually classifying the documents.

A.2 Experimental Methodology

We conducted a series of experiments to explore the utility of feature selection for datasets plagued with redundant features. In what follows, we first describe the construction of the datasets used in the experiments, and then proceed to developing a measure that predicts the utility of feature selection for a given dataset.

A.2.1 Datasets

Acquiring datasets for text categorization based on Web directories has been often performed in prior studies, which used Yahoo! (Mladenic and Grobelnik, 1998b), ODP (Chakrabarti et al., 2002; Cohen et al., 2002) and the Hoover's Online company database (Yang, Slattery, and Ghani, 2002). This approach allows to eliminate the huge manual effort required to actually label the documents, by first selecting a number of categories (= directory nodes) to define the labels, and then collecting the documents from the subtrees rooted at these categories to populate the dataset.

In our prior work (Davidov, Gabrilovich, and Markovitch, 2004) we developed a methodology for automatically acquiring labeled datasets for text categorization from hierarchical directories of documents, and implemented a system that performed such acquisition based on the Open Directory Project (`http://dmoz.org`). In the present paper we use a subset of 100 datasets acquired using this methodology. Each dataset consists of a pair of ODP categories with an average of 150 documents, and corresponds to a binary classification task of telling these two categories apart (documents are single-labeled, that is, every document belongs to exactly one category). When generating datasets from Web directories, where each category contains links to actual Internet sites, we construct text documents representative of those sites. Following the scheme introduced by Yang, Slattery, and Ghani (2002), each link cataloged in the ODP is used to obtain a small representative sample of the target Web site. To this end, we crawl the target site in BFS order, starting from the URL listed in the directory. A predefined number of Web pages (5 in this work) are downloaded, and concatenated into a *synthetic document*, which is then filtered to remove HTML markup; the average document size after filtering is 11.2 Kilobytes.

The datasets vary significantly by their difficulty for text categorization, and baseline SVM accuracy obtained on them is nearly uniformly distributed between 0.6 and 0.92. To list a few examples, datasets in our collection range from easy ones containing such pairs of ODP categories as `Games/Video_Games/Shooter` and `Recreation/Autos/Makes_and_Models/Volkswagen`, to medium difficulty ones with `Arts/Music/Bands_and_Artists` vs. `Arts/Celebrities`, to hard ones such as `Regional/North_America/United_States/Virginia/Richmond/Business_and_Economy` vs. `Regional/North_America/United_States/Florida/Fort_Myers/Business_and_Economy`. The full collection of 100 datasets, along with additional statistics and all the raw data used in our experiments is available at `http://techtc.cs.technion.ac.il/techtc100` .

A.2.2 Predicting the Utility of Feature Selection

In Section A.3 we show that the majority of datasets we used in this study benefit greatly from aggressive feature selection. We conjectured that these datasets have a small number of features that together allow to learn the underlying concept concisely, while the rest of the features do more harm than good. To understand this phenomenon, we examined the distribution of features in each dataset by their information gain.

Figure A.1 shows this distribution for several sample datasets.[1] Empirically, we observed that datasets with feature distribution similar to Dataset 46 benefit from feature selection immensely (for this particular dataset, aggressive feature selection improved SVM accuracy from 0.60 to 0.93). Such datasets have several features with high information gain, while the rest of their features have markedly lower IG scores. In contrast to these, datasets similar to Dataset 1 are characterized with smooth spectrum of IG values—in such cases feature selection will often eliminate features that carry essential information; indeed, for this dataset feature selection caused SVM accuracy to drop from 0.86 to 0.74. For comparison, we show a similarly looking graph for the 20 Newsgroups (20NG) dataset, which is often used for text categorization experiments and for which feature selection was found particularly harmful (Bekkerman, 2003).

Interestingly, high IG values of best-scoring features do not necessarily imply that feature selection will substantially improve the accuracy. For instance, Dataset 31 has several features with very high information gain, but its IG graph declines gracefully over subsequent features, and does not fall as sharp as for Dataset 46. Consequently, feature selection only improves SVM accuracy from 0.92 to 0.95—a much more modest gain than for Dataset 46. On the other hand, Dataset 61 has somewhat lower initial IG values, but its IG graph declines very sharply. Feature selection was shown to be of high utility for this dataset as well, boosting the accuracy from 0.64 to 0.84.

The above results imply that the absolute values of information gain are of less importance than the *speed of decline* of IG values across features. To quantify this phenomenon, we need to assess the number of *outliers*—features whose information gain is highly above that of all other features. Under this definition the desired measure becomes easy to formulate. We first compute the information gain for all features, and then count the number of features whose information gain is higher than 3 standard deviations above the average. Although the underlying distribution cannot be assumed to be normal, this familiar statistical criterion works very reliably in practice. Formally, let $\mathcal{D}$ be a dataset and let $\mathcal{F}$

[1] Dataset ids refer to the full listing table at `http://techtc.cs.technion.ac.il/techtc100` .

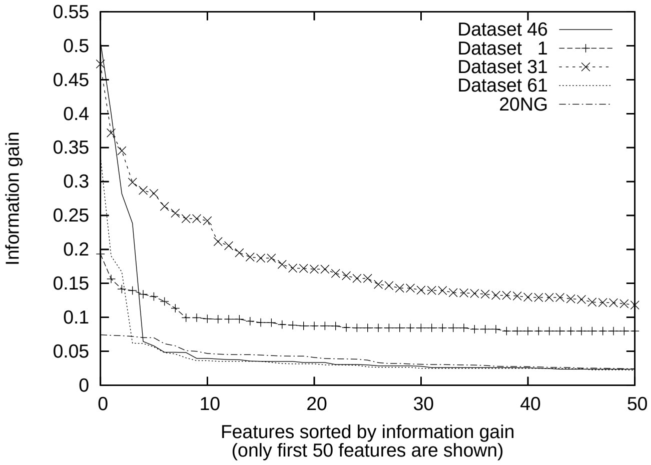

Figure A.1: Distribution of features by IG in several datasets

be a set of its features. We define the *Outlier Count (OC)* as

$$OC(\mathcal{D}, \mathcal{F}) = |\{f \in \mathcal{F} : IG(f) > \mu_{IG} + 3 \cdot \sigma_{IG}\ \}|\ ,$$

where μ_{IG} and σ_{IG} are the average and standard deviation of information gain of the features in $\mathcal{F}$. In Section A.3 we show that Outlier Count reliably predicts the utility of feature selection for a variety of datasets.

A.2.3 Extended Feature Set Based on WordNet

Several studies in text categorization performed feature construction using the WordNet electronic dictionary (Fellbaum, 1998). In this work we show that aggressive feature selection can significantly improve categorization accuracy for document representation extended with constructed features.

Scott and Matwin (1999), and later Wermter and Hung (2002), used WordNet to change document representation from a bag of words to a bag of synsets (WordNet notion of concepts), by using the hypernymy relation to generalize word senses. Since many words are not found in WordNet (e.g., neologisms, narrow technical terms, and proper names), we opted for *extending* a bag of words with WordNet-based features rather than completely changing document representation to a bag of synsets. To this end, we first perform feature generation by generalizing document words using WordNet, and then decimate the generated features through feature selection. In Section A.3.4 we demonstrate that feature selection is as important for generated features as it is for regular features (plain language words).

A.2.4 Feature Selection Algorithms

A variety of feature selection techniques have been tested for text categorization, while Information Gain, χ^2, Document Frequency (Yang and Pedersen, 1997; Rogati and Yang, 2002), Bi-Normal Separation (Forman, 2003) and Odds Ratio (Mladenic, 1998a) were reported to be the most effective. Adopting the probabilistic notation from Sebastiani (2002), we use $P(t_k, c_i)$ to denote the joint probability that a random document contains term t_k and belongs to category c_i, and N to denote the number of training documents. The above feature selection techniques are then defined as follows:

1. Information Gain (IG):
$\sum_{c\in\{c_i,\overline{c_i}\}} \sum_{t\in\{t_k,\overline{t_k}\}} P(t,c) \cdot \log \frac{P(t,c)}{P(t)P(c)}$

2. χ^2 (CHI): $N \cdot \frac{P(t_k,c_i)P(\overline{t_k},\overline{c_i})-P(t_k,\overline{c_i})P(\overline{t_k},c_i)}{P(t_k)P(\overline{t_k})P(c_i)P(\overline{c_i})}$

3. Document Frequency (DF): $N \cdot P(t_k)$

4. Bi-Normal Separation (BNS):
$|F^{-1}(P(t_k|c_i)) - F^{-1}(P(t_k|\overline{c_i}))|$, where F is the cumulative probability function of the standard Normal distribution

5. Odds Ratio (OR): $\frac{P(t_k|c_i)\cdot(1-P(t_k|\overline{c_i}))}{(1-P(t_k|c_i))\cdot P(t_k|\overline{c_i})}$

6. Random (RND)

Actual feature selection is performed by selecting the top scoring features, using either a predefined threshold on the feature score or a fixed percentage of all the features available. In addition to these “principled” selection schemes, we unconditionally remove stop words and words occurring in less than three documents.

A.2.5 Classification Algorithms and Measures

We used the datasets described in Section A.2.1 to compare the performance of *Support Vector Machines* (Vapnik, 1995), *C4.5* (Quinlan, 1993), and *K-Nearest Neighbor* (Duda and Hart, 1973). In this work we used the SVM^{light} implementation (Joachims, 1999a) with a linear[2] kernel.

We used classification accuracy as a measure of text categorization performance. Studies in text categorization usually work with multi-labeled datasets

[2] Joachims (1998) observed that most text categorization problems are linearly separable, and consequently most studies in the field used a linear SVM kernel (Bekkerman, 2003; Forman, 2003; Brank et al., 2002).

in which each category has much fewer positive examples than negative ones. In order to adequately reflect categorization performance in such cases, other measures of performance are conventionally used (Sebastiani, 2002), including precision, recall, F_1, and precision-recall break-even point (BEP). However, for single-labeled datasets all these measures can be proved to be equal to accuracy, which is the measure of choice in the machine learning community.

A.3 Empirical Evaluation

In this section we evaluate the role of feature selection for several classification algorithms operating on datasets with many redundant features. We conducted the experiments using a text categorization platform of our own design and development called $\mathcal{H}$OGWARTS. All accuracy values reported below were obtained using 4-fold cross-validation scheme.

When working with support vector machines, it is essential to perform adequate parameter tuning. In the case of a linear kernel (and under the assumption of equal cost of errors on positive and negative examples), the only relevant parameter is C, namely, the trade-off between training error and margin. To optimize this parameter, we set aside one fold of the training data as a validation set, and for each feature selection level selected the best C value from among $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10^1, 10^2, 10^3, 10^4\}$.

A.3.1 Validation of $\mathcal{H}$OGWARTS Performance

In this section we demonstrate that the results of classifying existing datasets with $\mathcal{H}$OGWARTS are consistent with those in other published studies. Figure A.2 shows the results of using SVM in conjunction with IG feature selection to classify three datasets frequently used in text categorization studies: 10 largest categories of Reuters-21578 (Reuters, 1997), 20 Newsgroups (Lang, 1995), and Movie Reviews (Pang, Lee, and Vaithyanathan, 2002).[3] Using all features, $\mathcal{H}$OGWARTS achieved BEP of 0.922 on Reuters, 0.854 on 20 Newsgroups and 0.818 on Movie Reviews. These results are very similar to the performance obtained by other researchers (all using SVM). Dumais et al. (1998) achieved BEP of 0.92 for the 10 largest Reuters categories. Bekkerman (2003) obtained BEP of 0.856 on the 20 Newsgroups dataset. Pang, Lee, and Vaithyanathan (2002) obtained accuracy of 0.829 on the Movie Reviews dataset.

As can be seen in Figure A.2, any level of feature selection harms the performance on all of these datasets. The graphs for χ^2 and BNS feature selection

[3] Since the former two of these datasets are multi-labeled, we use precision-recall break-even point (BEP) as a measure of classification performance rather than accuracy (see Section A.2.5).

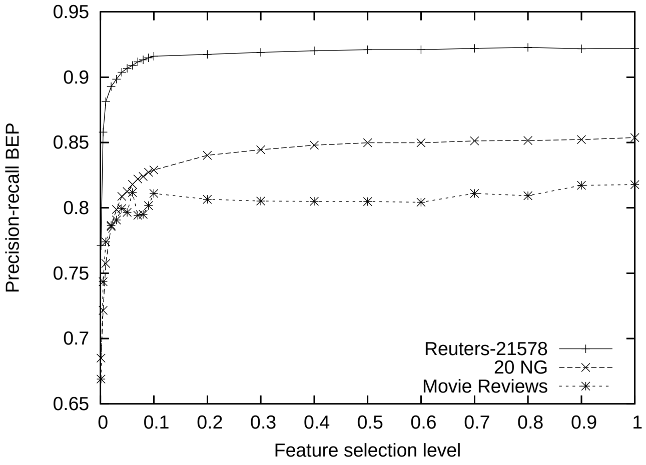

Figure A.2: $\mathcal{H}$OGWARTS performance on existing datasets (feature selection with IG)

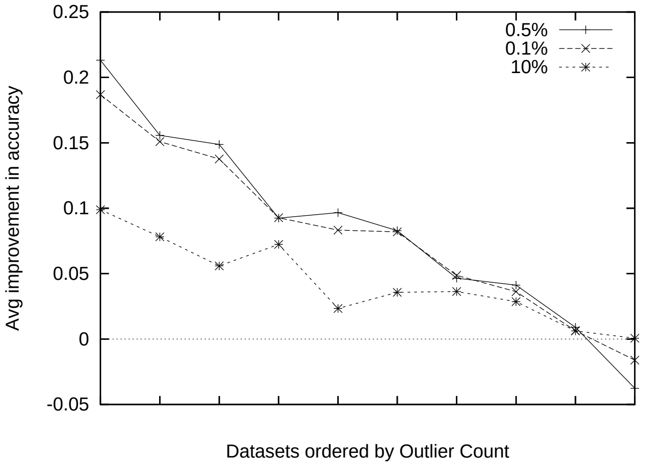

Figure A.3: Improvement in SVM accuracy at different FS levels vs. using 100% features

algorithms exhibit behavior very similar to IG, so we do not show them here owing to lack of space. Note that all the experiments reported in the rest of the paper use the 100 datasets we acquired as explained in Section A.2.1.

A.3.2 Predicting the Utility of Feature Selection with Outlier Count

We now show that the Outlier Count measure defined in Section A.2.2 reliably predicts the utility of feature selection. Figure A.3 shows the improvement in SVM accuracy at several feature selection levels versus the baseline accuracy obtained using 100% of features. As we can see, Outlier Count strongly correlates with the magnitude of improvement that can be obtained through feature selection. We observe that at lower values of Outlier Count aggressive feature selection is highly beneficial. Conversely, at higher OC values much more moderate (if any) feature selection should be performed, while aggressive selection causes degradation in accuracy. The next section examines the correlation of Outlier Count with the differences in performance between individual classifiers.

The Outlier Count for the datasets we used is nearly uniformly distributed between 6 and 62, with a single outlier value (no pun intended!) of 112 for Dataset 1 (Figure A.1), for which feature selection caused SVM accuracy to drop from 0.86 to 0.74. For other datasets frequently used for text categorization, Outlier Count for Reuters-21578 is 78, Movie Reviews—154, and 20 Newsgroups—391, which explains why feature selection does for them more harm than good.

Based on these findings, we conclude that using Outlier Count for ordering datasets reflects the degree to which a dataset can be concisely described by only a few features, while the rest of the features are predominantly redundant and have detrimental effect on classification results.

A.3.3 Comparison of Classifiers

Figure A.4 compares the performance of SVM, KNN and C4.5 on the 100 datasets ordered by Outlier Count. When no feature selection is employed, the performance of C4.5 mostly dominates that of SVM and KNN, and only declines in the rightmost part of the graph, which contains datasets where a few features are not sufficient for learning the concept.

Table A.1 shows classifier accuracy without feature selection and with the optimal feature selection level for each classifier. We used *paired t-test* to assess the significance of differences in classifier accuracy over the 100 datasets (see Table A.2). Without any feature selection, the differences between classifiers were found to be very significant at $p < 5 \cdot 10^{-3}$ or lower. For individual classifiers, the improvement in accuracy due to feature selection was extremely significant at $p < 10^{-13}$.

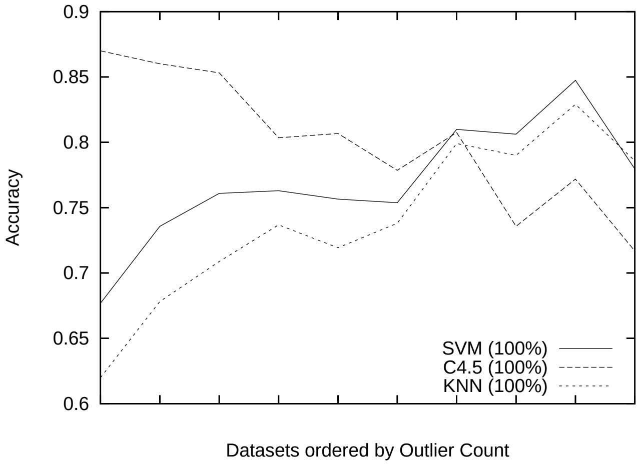

Figure A.4: Comparison of performance of SVM, C4.5 and KNN with 100% features

Table A.1: Classifier accuracy at different FS levels

Classifier	Accuracy with 100% features	Accuracy with the optimal FS level
SVM	0.769	0.853 (using 0.5% features)
C4.5	0.800	0.843 (using 0.5% features)
KNN	0.741	0.827 (using 2% features)

Table A.2: Statistical significance of differences in classifier accuracy (p values)

Classifier (FS level)	C4.5 (100%)	KNN (100%)	SVM (0.5%)	C4.5 (0.5%)	KNN (2%)
SVM (100%)	$5 \cdot 10^{-3}$	$4 \cdot 10^{-9}$	$4 \cdot 10^{-15}$	$2 \cdot 10^{-10}$	$6 \cdot 10^{-11}$
C4.5 (100%)		$2 \cdot 10^{-5}$	$6 \cdot 10^{-14}$	$2 \cdot 10^{-15}$	$3 \cdot 10^{-4}$
KNN (100%)			$2 \cdot 10^{-16}$	$6 \cdot 10^{-13}$	$6 \cdot 10^{-14}$
SVM (0.5%)				$9 \cdot 10^{-3}$	$4 \cdot 10^{-8}$
C4.5 (0.5%)					$5 \cdot 10^{-3}$

A.3.4 The Effect of Using Different Feature Sets

Figure A.5 compares the performance of classifiers at different feature selection levels (using Information Gain). As we can see, C4.5 performs better than SVM except for the most aggressive FS levels, where their accuracy becomes nearly equal. Interestingly, C4.5 stays high above KNN at most FS levels.

Figure A.6 presents a similar graph for the extended feature set based on WordNet. Here we use all features of the conventional bag of words, and only apply feature selection to the constructed features. C4.5 clearly manages the multitude of redundant features much better than both SVM and KNN. It is also noteworthy that the accuracy of SVM and KNN increases steadily as feature selection becomes more aggressive, while the improvement in their performance with 0.5% features vs. 100% features is strongly significant at $p < 10^{-18}$.

When using the optimal FS level (0.5% for both regular words and WordNet concepts), the addition of WordNet features is only responsible for a minor improvement in SVM accuracy from 0.853 to 0.854.

A.3.5 The Effect of Using Different FS Algorithms

Figures A.7 and A.8 show the effect of using different feature selection algorithms (see Section A.2.4) with SVM and C4.5. Consistently with prior studies (Forman, 2003; Rogati and Yang, 2002), we observe that IG, CHI and BNS are the best performers, while the difference between them is not statistically significant.[4] In contrast with prior studies, we observe that on the family of datasets we described, the best performance of SVM is obtained when only using a tiny fraction of features (0.5% for the three best FS techniques).

A.3.6 Testing the Relevancy of Features

In previous sections we showed that text categorization can greatly benefit from aggressive feature selection. We now address the question whether the features discarded by selection are at all relevant for classification. Following Joachims (1998), we sorted all features by their information gain, and then removed progressively larger fractions (0.1%, 0.5%, 1%, . . . , 10%, 20%, . . . , 100%) of the *most informative* features. As can be seen in Figure A.9, the performance of an SVM classifier trained on the remaining features is noticeably better than random up to very high levels of such harmful "selection". These results corroborate earlier findings by Joachims (1998), and support our hypothesis that the

[4] The graph for KNN looks substantially similar and also confirms the superiority of IG, CHI and BNS (with negligible differences), so we omit it owing to lack of space.

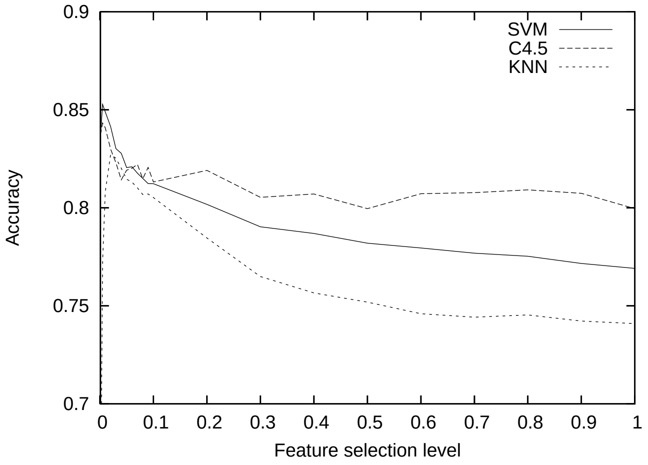

Figure A.5: Classification using a bag of words

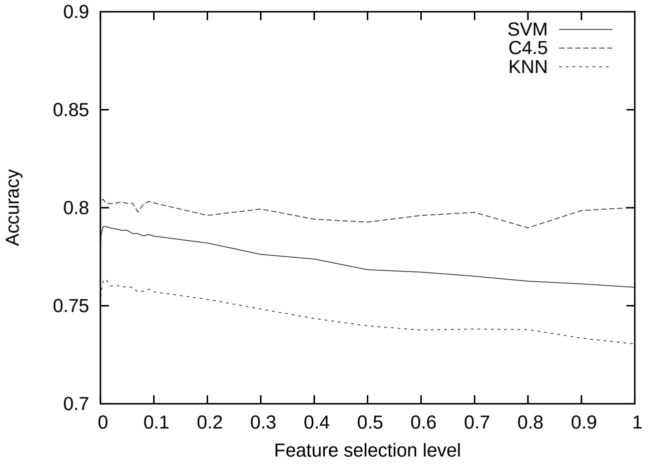

Figure A.6: Classification using an extended feature set

features removed through selection are *redundant*, even though most of them may be considered relevant.

A.4 Discussion

Studies in text categorization usually represent documents as a bag of words, and consequently have to manage feature spaces of very high dimensionality. Most previous works in the field found that these numerous features are relevant for classification, and that in particular the performance of SVM text categorization

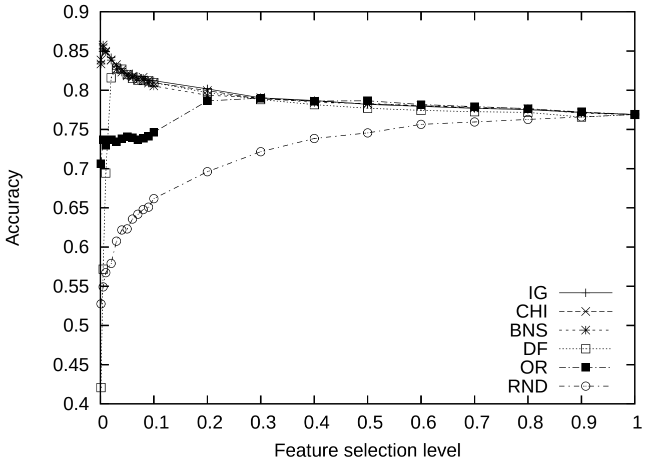

Figure A.7: SVM accuracy vs. FS level

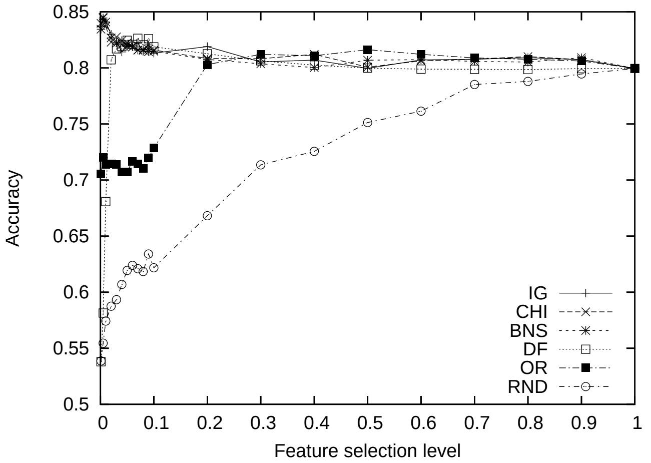

Figure A.8: C4.5 accuracy vs. FS level

peaks when no feature selection is performed.

We described a class of datasets plagued with *redundant* features, such that their elimination significantly boosts categorization accuracy of a host of classifiers. Specifically, we showed that when no feature selection is employed on such datasets, SVMs are significantly outperformed by C4.5. To explain this phenomenon, we analyzed the distribution of features by their information gain, and observed that this effect occurs when a small number of features are sufficient for concisely learning the underlying concept. We defined a measure named Outlier Count that, for a given dataset and fixed representation scheme, estimates the

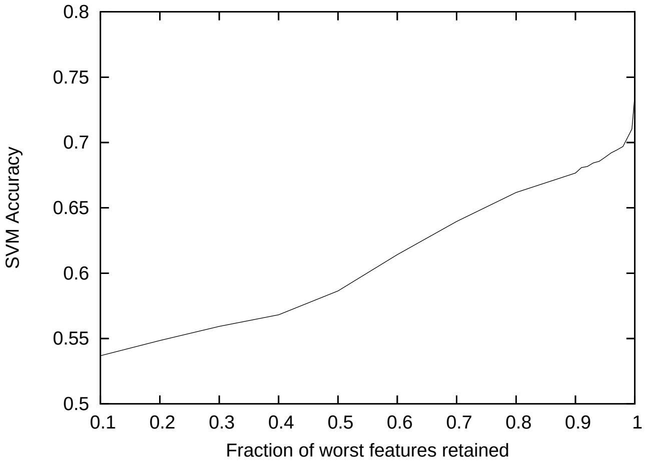

Figure A.9: Removing the best features by IG

amount of feature redundancy through outlier analysis.

In a series of experiments, we demonstrated that Outlier Count reliably predicts the amount of improvement that can be gained through feature selection. These findings are backed by empirical evidence both for the conventional bag of words, and for a representation extended through feature generation based on WordNet. We further performed a controlled ablation study to verify that the redundant features are in fact relevant for classification. To this end, we removed progressively larger fractions of most informative features, and found the remaining ones to suffice for better than random performance. Finally, we analyzed several established benchmarks for text categorization with respect to Outlier Count, and explained why they do not benefit from feature selection.

Following the established practice in text categorization, throughout this paper we used an SVM classifier with a linear kernel. In an ancillary experiment we sought to determine whether a non-linear SVM kernel may fare better than a linear one when dealing with numerous redundant features. Without feature selection, switching from a linear kernel to an RBF one reduced the accuracy from 0.769 to 0.687. Even at the optimal feature selection level, the accuracy achieved with an RBF kernel was slightly below that of a linear one (0.849 vs. 0.853), contradicting our anticipation of better performance by a more sophisticated kernel. However, this experiment should be considered preliminary, and in our future work we plan to conduct a thorough investigation of the ability of non-linear SVM kernels to withstand high rates of redundant features.

In a recent study, Forman (2003) proposed a novel feature selection algorithm named Bi-Normal Separation, which improved the performance of SVM text categorization on a range of datasets. Peak performance was obtained when using 500–1000 features (approximately 10% of all available features on the average). More aggressive feature selection led to sharp degradation of the results—using less than 100 features caused macro-F_1 to decrease by 5%–10% depending on the selection algorithm used.

Our work corroborates the findings that feature selection can help text categorization with SVMs, and describes a class of problems where the improvement due to feature selection is particularly large. We showed that for this class of problems the improvement in accuracy can be twice as high as found by Forman (2003) (namely, 8.4% vs. 4.2%), while optimal performance is achieved when using much fewer features (between 5 and 40, depending on the dataset). We also evaluated several feature selection algorithms on text categorization problems characterized with many redundant features. Our results support earlier findings that Information Gain, Bi-Normal Separation and χ^2 are the most powerful feature selection algorithms, while the differences between them are not significant.

It should be noted that for all the datasets we used, the utility of feature selection could be established by setting aside part of the training data to serve as a validation set. Indeed, the high redundancy level was so pronounced, that the optimal selection level for the testing data could almost always be correctly determined on the validation fold. However, we believe that the introduction of Outlier Count and the use of ablation experiments that systematically eliminate most informative features, allow deeper understanding of the issues of feature redundancy and relevancy.

Appendix B

Parameterized Generation of Labeled Datasets for Text Categorization Based on a Hierarchical Directory

Although text categorization is a burgeoning area of IR research, readily available test collections in this field are surprisingly scarce. We describe a methodology and system (named $\mathcal{A}$CCIO) for *automatically* acquiring labeled datasets for text categorization from the World Wide Web, by capitalizing on the body of knowledge encoded in the structure of existing hierarchical directories such as the Open Directory. We define *parameters* of categories that make it possible to acquire numerous datasets with *desired properties*, which in turn allow better control over categorization experiments. In particular, we develop metrics that estimate the difficulty of a dataset by examining the host directory structure. These metrics are shown to be good predictors of categorization accuracy that can be achieved on a dataset, and serve as efficient heuristics for generating datasets subject to user's requirements. A large collection of automatically generated datasets are made available for other researchers to use.

B.1 Introduction

While numerous works studied text categorization (TC) in the past, good test collections are by far less abundant. This scarcity is mainly due to the huge manual effort required to collect a sufficiently large body of text, categorize it, and ultimately produce it in machine-readable format. Most studies use the Reuters-21578 collection (Reuters, 1997) as the primary benchmark. Others use

20 Newsgroups (Lang, 1995) and OHSUMED (Hersh et al., 1994), while TREC filtering experiments often use the data from the TIPSTER corpus (Harman, 1992).

Even though the Reuters-21578 dataset became a standard reference in the field, it has a number of significant shortcomings. According to Dumais and Chen (2000), "the Reuters collection is small and very well organized compared with many realistic applications". Scott (1998) also noted that the Reuters corpus has a very restricted vocabulary, since Reuters in-house style prescribes using uniform unambiguous terminology to facilitate quick comprehension. As observed by Joachims (1998), large Reuters categories can be reliably classified by virtually any reasonable classifier. We believe that TC performance on more representative real-life corpora still has way to go. The recently introduced new Reuters corpus (Lewis et al., 2004), which features a large number of documents and three orthogonal category sets, definitely constitutes a substantial challenge. At the same time, acquisition of additional corpora for TC research remains a major issue.

In the past, developing a new dataset for text categorization required extensive manual effort to actually label the documents. However, given today proliferation of the Web, it seems reasonable to acquire large-scale real-life datasets from the Internet, subject to a set of constraints. Web directories that catalog Internet sites represent readily available results of enormous labeling projects. We therefore propose to capitalize on this body of information in order to derive new datasets in a fully automatic manner. This way, the directory serves as a source of URLs, while its hierarchical organization is used to label the documents collected from these URLs with corresponding directory categories. Since many Web directories continue to grow through ongoing development, we can expect the raw material for dataset generation to become even more abundant as the time passes.

In what follows, we propose a methodology for *automatic* acquisition of up-to-date datasets with *desired properties.* The *automatic* aspect of acquisition facilitates creation of numerous test collections, effectively eliminating a considerable amount of human labor normally associated with preparing a dataset. At the same time, datasets that possess *predefined characteristics* allow researchers to exercise better control over TC experiments and to collect data geared towards their specific experimentation needs. Choosing these properties in different ways allows one to create focused datasets for improving TC performance in certain areas or under certain constraints, as well as to collect comprehensive datasets for exhaustive evaluation of TC systems.

After the data has been collected, the hierarchical structure of the directory may be used by classification algorithms as background world knowledge—the association between the data and the corresponding portion of the hierarchy is defined by virtue of dataset construction. The resulting datasets can be used for regular text categorization, hypertext categorization, as well as hierarchical text classification. Moreover, many Web directories cross-link related categories with so-called "symbolic links", which allow one to construct datasets for multi-labeled TC experiments.

We developed a software system named $\mathcal{A}$CCIO[1] that lets the user specify desired dataset parameters, and then efficiently locates suitable categories and collects documents associated with them. It should be observed that Web documents are far less fluent and clean compared to articles published in the "brick and mortar" world. To ensure the coherence of the data, $\mathcal{A}$CCIO represents each Web site with several pages gathered from it through crawling, and filters the pages gathered both during and after the crawling. The final processing step computes a number of performance metrics for the generated dataset.

In this paper we describe generation of datasets based on the *Open Directory Project* (ODP, `http://dmoz.org`), although the techniques we propose are readily applicable to other Web directories, as well as to non-Web hierarchies of documents (see Section B.2). A number of previous studies in hypertext and hierarchical text classification used document sets collected from Yahoo! (Mladenic and Grobelnik, 1998b; Labrou and Finin, 1999), ODP (Chakrabarti et al., 2002; Cohen et al., 2002; Meng et al., 2002) and the Hoover's Online company database (Ghani et al., 2000; Yang, Slattery, and Ghani, 2002). To the best of our knowledge, all these studies performed standard acquisition of Web documents pointed at from the explicitly specified directory nodes; specifically, no properties of categories were considered or defined, and no attempt to predict the classification performance was made. Interestingly, a recent study in word sense disambiguation (Santamaria, Gonzalo, and Verdejo, 2003) used ODP to automatically acquire labeled datasets for disambiguation tasks. In this work, a collection of ODP categories were first automatically mapped to WordNet (Fellbaum, 1998) senses, and then the descriptions of links classified under these categories were collected to serve as sentences with sense-labeled words. In contrast to our approach, this mapping only considered category paths, while we also analyze the full text of category and link descriptions (see Section B.2).

The main contributions of this paper are threefold. First, we present a methodology for automatically acquiring labeled data sets for text categorization experiments, which allows parameterized generation of datasets with desired properties. Second, we establish a connection between similarity metrics for document sets and the classification accuracy achieved on these sets. The similarity metrics we developed are shown to be good predictors of classification accuracy, and can therefore be used as efficient heuristics for locating datasets of desired degree of hardness. We also propose to use classification accuracy as a new similarity metric that reflects how separable two document sets are. Finally, we make

[1] *Accio* (Latin - to call to, summon)—incantation for the Summoning Charm, which causes an object called for to fly to the caster (Rowling, 2001).

publicly available a large collection of text categorization datasets that we collected and evaluated in the course of this work, along with a variety of metrics computed for them. Using the same datasets allows different research groups to conduct repeatable experiments and to compare their results directly. This repository, which is similar in purpose to the UCI Repository of machine learning databases (Blake and Merz, 1998), is available for research use and is publicly accessible at `http://techtc.cs.technion.ac.il`. All datasets are available in plain text form and in the form of preprocessed feature vectors; the latter distribution can be used by researchers in machine learning who are less interested in the specifics of text processing. Furthermore, for each dataset we provide baseline performance numbers using SVM, KNN, and C4.5. We also plan to release the software system for automatic generation of datasets. Other researchers will be able to use $\mathcal{A}$CCIO to acquire new datasets subject to their specific requirements.

B.2 Parameterization of Dataset Generation

Throughout this paper we discuss generation of datasets that contain two categories and are single-labeled, that is, every document belongs to exactly one category. In Section B.5 we consider possible relaxations to this rule.

We assume the availability of a hierarchical directory of documents that satisfies the following requirements:

1. The directory is organized as a tree where each node is labeled with a *category*.

2. There is a collection of documents associated with each category (directory node).

3. Categories are provided with text descriptions. Documents associated with the categories may optionally be accompanied by short annotations.

Suitable directories come in a variety of forms. Some are major Web directories that catalog actual Web sites, such as Yahoo! or the Open Directory. The Medical Subject Headings (MeSH) hierarchy (MeSH, 2003) maintained by the U.S. National Library of Medicine is cross-linked with the MEDLINE database, and therefore can be used for automatic generation of labeled datasets of medical texts. Library classification schemes such as UDC and Dewey are hierarchical catalogs of books that can also be used for automatical acquisition of text categorization datasets; samples of books can be used if shorter documents are required. The open content Wikipedia encyclopedia[2] collaboratively developed by Internet

[2] `http://www.wikipedia.org` .

users offers tantalizing opportunities for harnessing high quality datasets. As of this writing, Wikipedia contains over 170,000 articles in English and 150,000 in other languages, thus allowing acquisition of datasets on similar topics in a variety of languages. Yet another option is to use the new Reuters collection (Lewis et al., 2004) that contains over 800,000 documents labeled with categories coming from three distinct hierarchies. In this project we generate datasets based on the Open Directory Project, which is arguably the largest publicly available Web directory.[3]

We employ two kinds of parameters that define the nature of generated datasets: those characterizing the dataset as a whole (i.e., describe *pairs* of categories), and those characterizing individual categories that comprise the datasets. Varying these parameters allows one to create classification tasks with different properties.

B.2.1 Metrics

Metrics quantify conceptual distance between a pair of categories. Intuitively, the larger the distance, the easier it is to induce a classifier for separating the categories. From the machine learning perspective, the difficulty of a dataset for existing categorization algorithms is an important parameter. The ability to create datasets with varying degree of difficulty would be instrumental in the quest for better learning algorithms. In other words, we would like to retain control over *the degree of separability* of the two categories comprising the dataset. In this section we first define an exact but computationally expensive measure of dataset hardness, and then propose two metrics that are highly correlated with it but are much more efficient to compute.

Achievable Categorization Accuracy as a Measure of Dataset Hardness

A straightforward way to assess how difficult a given dataset is for currently available learning algorithms is simply to run these algorithms on it. It is apparently appealing to use the accuracy of a single best classification algorithm as an ultimate measure, especially in the light of the fact that a number of studies showed support vector machines to be the best performing text classifier (Joachims, 1998; Dumais et al., 1998). However, as we show in Section B.4.4, SVM does not necessarily produce the best results for every dataset. Several researchers observed

[3] Although the actual size of Yahoo! has not been publicly released in the recent years, it is estimated to be about half the size of the Open Directory (see `http://sewatch.com/reports/directories.html` and `http://www.geniac.net/odp` for more details).

similar phenomena, and used various learning approaches to decide which classifier should be used for a given category (Lam and Lai, 2001) or for a given document (Bennett, Dumais, and Horvitz, 2002).

We believe that such sophisticated classifier combination schemes might be an overkill for establishing a measure of category separability. We suggest using some function of the accuracy values achieved by a number of classifiers as the "gold standard" of hardness. While there are many ways to define a suitable combination scheme, we propose to use the maximum accuracy among a set of classifiers, as we believe it reflects how difficult the dataset is for the best available algorithm (obviously, without an oracle predicting which classifier to use, this value cannot always be attained in practice). Formally, we define

$$dist_{class\_max}(c_1, c_2) = \max_{alg \in \mathcal{C}} Accuracy_{alg}(c_1, c_2) \ ,$$

where c_1, c_2 are a pair of categories comprising a dataset and $\mathcal{C}$ is a set of classification algorithms. In the sequel we refer to this metric as Maximum Achievable Accuracy (MAA). In the experiments reported in Section B.4 we compute MAA using classifiers based on support vector machines, decision trees and the K-Nearest Neighbor algorithm.

Nothing seems simpler than defining the hardness of a dataset by actual classification accuracy. The only problem with this approach is that it is grossly *inefficient*. When we search for datasets in a certain difficulty range, using MAA as part of "generate-and-test" strategy is too computationally intensive to be practical. Computing MAA requires to actually crawl the Web to download the documents, clean the data and organize it as a dataset, and finally subject it to a number of classifiers. If MAA turns out to be too low or too high compared with the requirements, we have to test another pair of categories, then another one, and so on.

We developed two metrics that estimate the difficulty of a dataset by only examining the *hierarchical structure* of the host directory, *without analyzing the text of actual documents*. In Section B.4 we show that these metrics are strongly correlated with MAA and the accuracies of individual classifiers, and this can serve good predictors of how difficult it is to build a classifier that tells two categories apart.

Historically, the idea of partitioning categories by similarity of meaning (as well as by importance or frequency) was first mentioned by Lewis (1991), when he suggested to group categorization results over different kinds of categories.

In order to develop metrics for computing similarity of categories drawn from a hierarchical directory, let us review a similar setting of assessing similarity of words using a hierarchical dictionary or taxonomy. The metrics we define assign lower values to more similar categories, therefore, in what follows we use the term *distance* metric (rather than similarity metric) to emphasize this fact.

Edge-counting Graph Metric

The edge-counting metric (called *graph metric* below) measures the distance between a pair of categories by the length of the shortest[4] path connecting them in the hierarchy. We conjecture that the closer two categories are in the underlying graph, the closer they are in meaning, and hence the smaller the distance between them is. Formally, we define

$$dist_{graph}(c_1, c_2) = \#edges\ in\ the\ tree\ path\ from\ c_1\ to\ c_2\ .$$

Rada and Bicknell (1989) also used hierarchy path length as a measure of conceptual distance. However, this study focused on estimating the similarity of individual terms rather than entire sets of documents.

WordNet-based Textual Metric

The above metric only uses the graph structure underlying the hierarchy as a sole source of information. We now propose a more elaborate metric (called *text metric* in the sequel) that compares textual descriptions of the categories that are assumed to be provided with the hierarchy.

Our text metric builds upon the similarity metric for individual words suggested by Resnik (1999), which uses the WordNet electronic dictionary (Fellbaum, 1998) as a source of additional background knowledge. Given two words w_1 and w_2 whose similarity needs to be established, let us denote by S_1 the set of all WordNet nodes (called *synsets*) that contain w_1 and by S_2—the set of all synsets that contain w_2. Resnik defined the similarity between two words as

$$sim_{Resnik}(w_1, w_2) = max_{s_j}[-\log p(s_j)]\ , \tag{B.1}$$

where $\{s_j\}$ is a set of synsets that subsume at least one synset from S_1 and one synset from S_2 (i.e., the set of all concepts that subsume both given words), $p(s_j)$ is the probability of synset s_j computed as a function of the frequencies of words that belong to it measured on a reference corpus, and $-\log p(s_j)$ is the information content of this synset. No word sense disambiguation is performed, and all senses of a polysemous word are considered equally probable.

We generalize this metric to make it applicable to entire category descriptions rather than individual words. In the preprocessing phase we represent each category by pooling together (i) the title and description of the category itself and all of its descendants (sub-categories), and (ii) the titles and descriptions

[4] Using the *shortest* path is important when the hierarchy is actually a graph rather than a tree (for example, when symbolic links of the Open Directory are considered).

(annotations) of the links to actual documents classified under this category or one of its sub-categories. We denote the union of all these textual descriptions for category c_i as D_i. Each pooled description D_i is represented as an unordered bag of words.

The (asymmetric) distance between a pair of such descriptions is canonically defined as an average distance from the words of the first description to those of the second one:

$$dist(D_1, D_2) = \frac{1}{|D_1|} \sum_{w \in D_1} dist(w, D_2) \ ,$$

where the distance between a word and a bag of words is defined as the shortest distance between this word and the bag (i.e., the distance to the nearest word in the bag):

$$dist(w, D) = min_{w' \in D} dist(w, w') \ . \tag{B.2}$$

The distance between two words is defined using Resnik's similarity metric, except the score it returns is subtracted from the maximum possible score (sim_{MAX}) to transform the similarity metric into a measure of distance:

$$dist(w, w') = sim_{MAX} - sim_{Resnik}(w, w') \ .$$

To estimate the word frequencies needed for the computation of $p(s_j)$ in (B.1), we used a training corpus composed of the descriptions of all ODP categories; this step effectively tunes the metric to a specific text collection at hand.

Finally, the metric that operates on entire textual descriptions of categories is symmetrically defined as

$$dist_{text}(c_1, c_2) = dist(D_1, D_2) + dist(D_2, D_1) \ .$$

Computing $dist_{text}$ requires some preprocessing computation to build category descriptions D_i, and then use the frequency of words found in these descriptions to train a language model that underlies the computation of $-\log p(s_j)$. Observe that even without the preprocessing phase performed offline, computing the text metric is a computationally intensive process, as it considers every pair of words in the two category descriptions, and for each such pair maximizes the information content of the subsuming synsets.

Budanitsky and Hirst (2001) provide a good survey of other word similarity metrics based on WordNet.

B.2.2 Properties of Individual Categories

The following parameters can be configured for individual categories:

1. The *cardinality* of a category specifies the desired number of documents it should contain. In general, the more examples (documents) are available, the easier the learning task is due to a better representation of the category.

2. Recall that the documents we collect actually represent Web sites they were downloaded from. Exploring Web sites to different depths affects the quality of this representation. However, taking too many documents from each site is not necessarily good, as moving further away from the site's root page may lead to barely related pages. The parameter that controls this fine balance is called *coherence*, and is expressed as a number of pages downloaded from each Web site and concatenated into a single document.

3. Limiting the selection of categories to a certain part of the hierarchy effectively allows to restrict the contents of the documents to a particular topic. For example, generating datasets from the Open Directory `Top/Health` subtree may be useful for testing operational TC systems for the medical domain. The *language* of documents may be restricted in a similar way.

B.3 Methodology for Automatic Dataset Generation

In this section we outline the methodology for automatic generation of datasets.

B.3.1 Acquisition of the Raw Data

Generating a new dataset starts with locating a pair of categories subject to user's specification, which consists of a set of desired parameters (or characteristics) of the dataset to build (see Section B.2). Finding a pair of categories at specified graph distance is easy, as it involves pursuing a corresponding number of edges in the graph underlying the hierarchy. On the other hand, identifying pairs of categories at a specified text distance is far from trivial. Although the experiments presented in Section B.4.3 do show high correlation between the two metrics, in general counting the number of edges can only give a rough estimation of the text distance between two categories.

Since the text metric is much more computationally intensive than the graph one, we cache its values for all pairs of categories considered so far. Given the desired text distance, we first consult the cache to see if a suitable pair of categories was already found. If this simple test fails, we randomly sample the cache and identify a pair in the sample whose distance is closest to the required one. We then perform a hill-climbing search in the hierarchy graph starting from that pair. This search is limited in the number of steps, and if no appropriate pair

```
Algorithm LocateCategoryPair_TextDist($d$)
  if ($\exists (p,q) \in Cache$ s.t. $dist_{text}(p,q) = d$)
    then $return$ $(p,q)$
  $found \leftarrow false$
  while ($\neg found$)
    Draw a random sample $S \subset Cache$
    Let $(p,q) \in S$ s.t. $\forall (p',q') \in S$ :
      $|d - dist_{text}(p,q)| \leq |d - dist_{text}(p',q')|$
    Starting from $(p,q)$, perform $n$-step hill climbing
      until a pair $(p_d, q_d)$ is found s.t. $dist_{text}(p,q) = d$
```

Figure B.1: Locating categories at requested text distance

is found after the limit is exhausted, we randomly sample the cache again, and repeat the entire process until a suitable pair of categories is found. Figure B.1 outlines the pseudocode of the search algorithm.

It is essential to emphasize that the above algorithm only analyzes the hierarchy structure and category descriptions, but *never examines the contents of actual documents*. It is this feature of our methodology that makes finding datasets of configurable difficulty much more computationally tractable than if MAA was to be used (Section B.2.1). In our future work we plan to develop more sophisticated algorithms for efficiently locating pairs of categories at specified conceptual distance (see Section B.5).

After locating an appropriate pair of categories, we collect the documents associated with them. Importantly, if a certain category c has several sub-categories under it in the given hierarchy ($c_1 \ldots c_n$), we collect the documents from the *union* of all these categories. The hierarchy structure allows us to view $c_1 \ldots c_n$ as particular cases of c, and thus we can find many more relevant documents than if looking into category c alone.

When generating datasets from Web directories such as the ODP, where each category contains links to actual Internet sites, we need to construct text documents representative of those sites. Following the scheme introduced in (Yang, Slattery, and Ghani, 2002), each link cataloged in the ODP is used to obtain a small representative sample of the target Web site. To this end, we crawl the target site in the BFS order, starting from the URL listed in the directory. A predefined number of Web pages are downloaded, and then concatenated into a *synthetic document*. We refer to these individual pages as *sub-documents*, since their concatenation yields one document for the categorization task. We usually refer to synthetic documents created by pooling sub-documents simply as *documents* to be consistent with TC terminology; alternatively, we call them *meta-documents* to avoid ambiguity when necessary.

Finally, HTML documents are converted into plain text and organized as a dataset, which we render in a simple XML-like format. It should be noted that converting HTML to text is not always perfect, since some small auxiliary text snippets (as found in menus and the like) may survive this procedure; we view such remnants as a (low) residual noise inherent in automated data acquisition.

B.3.2 Filtering the Raw Data to Cope with Noise

Data collected from the Web can be quite noisy. Common examples of this noise are textual advertisements, numerous unrelated images, and text rendered in background color aimed at duping search engines. To reduce the amount of noise in generated datasets we employ filtering mechanisms before, during, and after downloading the data.

Pre-processing filtering eliminates certain categories from consideration. For example, we unconditionally disregard the entire `Top/World` subtree of the Open Directory that catalogs Web sites in languages other than English. Similarly, the `Top/Adult` subtree may be pruned to eliminate inappropriate adult content.

Recall that for every directory link we download a number of pages whose concatenation represents the corresponding Web site. *Online filtering* performed during the download restricts the crawler to the site linked from the directory, and does not allow it to pursue external links to other sites.

Post-processing filtering analyzes all the downloaded documents as a group, and selects the ones to be concatenated into the final meta-document. In practice, we download more sub-documents than requested by the user, and then decimate them. We developed two post-processing filters:

1. *Weak* filtering discards Web pages that contain HTTP error messages, or only have a few words.

2. *Strong* filtering attempts to eliminate unrelated pages that do not adequately represent the site they were collected from (e.g., legal notices or discussion forum rules). To eliminate such pages, we try to identify obvious outliers. We use the root page of a Web site (i.e., the page linked from the directory) as a "model" deemed to be representative of the site as a whole. Whenever the root page contains enough text for comparison, we use the text metric developed in Section B.2.1 to compute the distance between it and every other page downloaded from the site. We then discard all pages located "further" from the root than one standard deviation above the average.

Comparing weak and strong filtering, we found the latter to improve TC accuracy by about 0.5%–1.5%.

B.4 Empirical Evaluation

In this section we show that the datasets generated using the proposed methodology are sufficiently versatile and allow adequate degree of control over TC experiments.

B.4.1 Data Acquisition

We used the methodology outlined in Section B.3 to automatically generate a collection of datasets based on the Open Directory Project (`http://dmoz.org`). The Open Directory is a public directory that catalogs selected Internet sites. At the time of this writing, ODP covers over 4 million sites organized in more than 540,000 categories. The project constitutes an ongoing effort promoted by non-professional users around the globe; currently, ODP advertises a staff of over 60,500 editors. Being the result of *pro bono* work, the Open Directory has its share of drawbacks, such as non-uniform coverage, duplicate subtrees in different branches of the hierarchy, and sometimes biased coverage due to peculiar views of the editors in charge. At the same time, however, ODP embeds a considerable amount of human knowledge.

Based on the Open Directory, we generated 300 datasets of varying difficulty, by using the metrics defined in Section B.2.1 to find categories located at different graph or text distances. Each dataset consists of a pair of categories with 100–200 documents per category, while each document was created by concatenating 5 sub-documents.

B.4.2 Text Categorization Infrastructure

The following learning algorithms were used to induce actual text classifiers: support vector machines (Vapnik, 1995) (using SVM^{light} implementation (Joachims, 1999a)), decision trees (*C4.5* (Quinlan, 1993)), and K-Nearest Neighbor (Duda and Hart, 1973). The motivation behind this choice of algorithms is that they belong to very different families, and thus allow comprehensive evaluation of the datasets generated.

We used classification accuracy as a measure of text categorization performance. Studies in text categorization usually work with multi-labeled datasets in which each category has much fewer positive examples than negative ones. In order to adequately reflect categorization performance in such cases, other measures of performance are conventionally used, including precision, recall, F_1, and precision-recall break-even point (Sebastiani, 2002). However, for single-labeled datasets all these measures can be proved to be equal to accuracy, which is the measure of choice in the machine learning community. All accuracy values reported in this paper were obtained under the 10-fold cross-validation scheme.

We conducted the experiments using a text categorization platform of our own design and development called $\mathcal{H}$OGWARTS[5]. We opted to build a comprehensive new infrastructure for text categorization, as surprisingly few software tools are publicly available for researchers, while those available only allow limited control over their operation. $\mathcal{H}$OGWARTS performs text preprocessing, feature extraction, construction, selection and valuation, followed by cross-validated classification. $\mathcal{H}$OGWARTS interfaces with SVM, KNN and C4.5, and computes all standard measures of categorization performance. At a later stage we plan to make $\mathcal{H}$OGWARTS publicly available for research use.

B.4.3 Correlation Between Distance Metrics and Text Categorization Accuracy

Recall that our primary aim is to generate datasets with predefined properties. Specifically, one of the most important properties we introduced in Section B.2 is the ability to exercise control over the *difficulty of separation* of two categories comprising a dataset. The experiments reported below were designed to verify whether the metrics we developed in Section B.2.1 can serve as reliable predictors of category separability. We first juxtapose metric predictions with the accuracy of an SVM classifier, and then compare them with the Maximum Achievable Accuracy (MAA).

Figure B.2 shows the correlation between the graph metric and SVM categorization accuracy, while Figure B.3 shows a similar plot for the text metric. Both figures demonstrate that the metrics have strong prediction power for SVM accuracy. The value of Pearson's linear correlation coefficient (Press et al., 1997) that we computed to quantify this dependence is 0.533 for the graph metric and 0.834 for the text one. Interestingly, the two metrics are fairly strongly correlated between themselves, as implied by their correlation of 0.614 (see Figure B.4).

As follows from the experimental results, there is a trade-off between the computational efficiency and the prediction power of the two metrics. The graph metric is much faster to compute, but only offers a rough estimation of the degree of separability of a pair of categories. The text metric is much less efficient to compute, but offers by far more reliable distance assessment.

[5] *Hogwarts school of witchcraft and wizardry* is the educational institution attended by Harry Potter (Rowling, 2001).

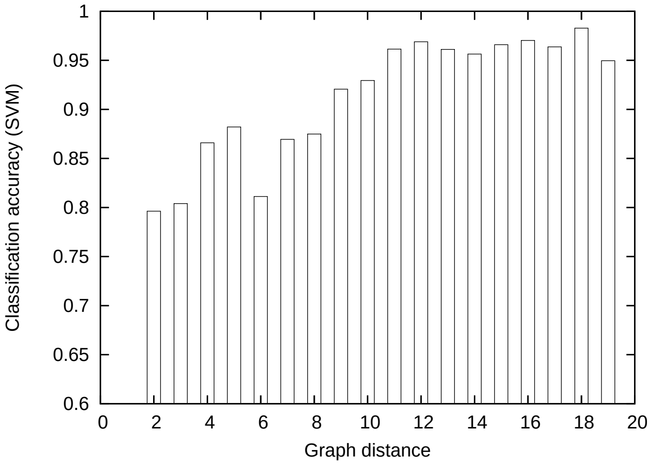

Figure B.2: SVM accuracy vs. graph distance

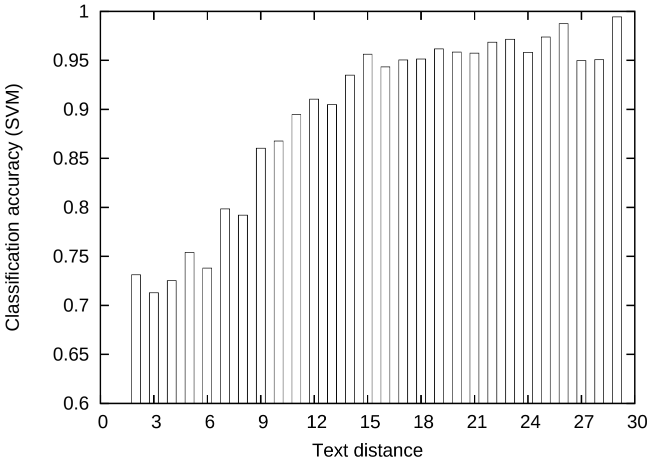

Figure B.3: SVM accuracy vs. text distance

B.4.4 Correlation Between Distance Metrics and MAA

In Section B.2.1 we defined the difficulty of a dataset as a function of performance of a number of classifiers. Instead of using the accuracy produced by any single classifier, we proposed to use the maximum value among several classifiers that were shown to be good performers in previous studies.

Let us first provide empirical support for the choice of MAA as a reasonable measure of conceptual distance between a pair of categories. The

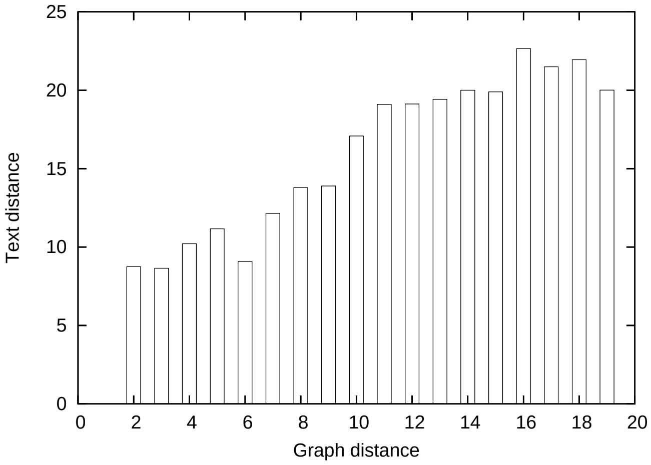

Figure B.4: Text distance vs. graph distance

average accuracy achieved by SVM on the datasets tested is 0.896, KNN—0.874, and C4.5—0.878. These results are consistent with previously published studies (Sebastiani, 2002), and show that the generated datasets exhibit similar performance properties to the manually collected ones used in prior research. However, a closer look at classifier performance on individual datasets reveals that SVM—although a superior technique in the majority of cases—does not always yield the best accuracy compared to other classifiers. Specifically, SVM was outperformed by KNN on 58 datasets (19%) and by C4.5 on 80 datasets (27%). Furthermore, C4.5 outperformed KNN on 119 datasets (40%), even though decision trees are usually deemed an inferior approach to text categorization compared to SVM and KNN. Therefore, the performance of the best currently available algorithm for a particular dataset constitutes a more reliable measure of its true difficulty.

The experiments we conducted prove that the correlation of the graph and text metrics to MAA is consistently high. Specifically, the correlation between $dist_{graph}$ and MAA is 0.550, and between $dist_{text}$ and MAA—0.790. Figures B.5 and B.6 depict these correlations with standard error bars. Based on these findings, we conclude that the metrics we developed are good predictors of dataset difficulty.

B.4.5 Versatility of Dataset Generation

We now show that the proposed methodology can be used to automatically generate a continuum of non-trivial categorization tasks of varying difficulty. Having

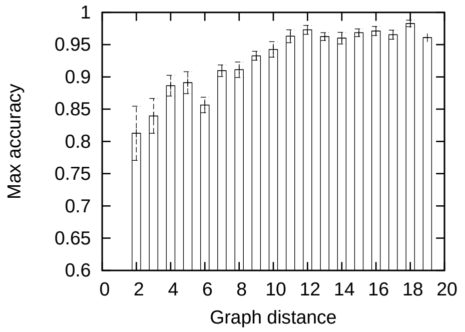

Figure B.5: MAA vs. graph distance

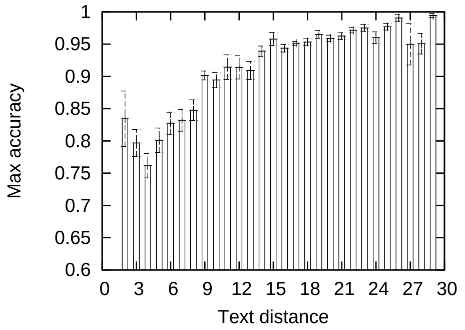

Figure B.6: MAA vs. text distance

established in the previous section that the distance metrics are good predictors of categorization accuracy, we demonstrate that it is possible to find enough category pairs of adequate size at different conceptual distances.

To prove this, we examine two graphs with pertinent ODP statistics. Figure B.7 depicts the number of category pairs that reside at various distances as measured by the graph metric. Since the text metric is much more computationally expensive, showing in full the similar distribution of text distances is not feasible. For machine learning tasks, we are usually interested in categories with a sufficient number of examples to make (statistical) learning meaningful and allow adequate generalization. Figure B.8 shows a sampled distribution of text distances among mid-size category pairs having 100–3000 links. ODP has approximately 13,000 categories in this size range (and therefore $13{,}000^2/2$ pairs); Figure B.8 was built by randomly sampling 3,500 pairs of such categories.

These graphs suggest that the Open Directory is large and versatile enough to produce numerous datasets with desired properties.

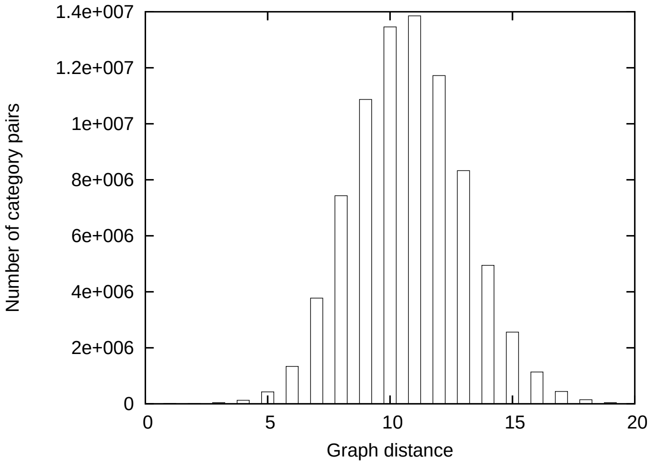

Figure B.7: Distribution of graph distances in ODP

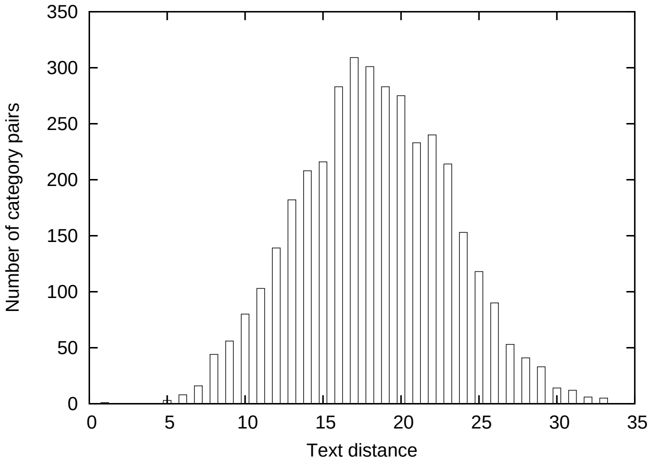

Figure B.8: Distribution of text distances in ODP (sample)

B.5 Conclusions and Future Work

Text categorization is an active area of research in information retrieval, yet good test collections are scarce. We presented a methodology and system named $\mathcal{A}$CCIO for automatically acquiring labeled datasets for text categorization from hierarchical directories of documents. We applied this methodology to generate 300 datasets from the largest Web directory to date—the Open Directory Project—as an example. The datasets thus generated can be used in a variety of learning tasks, including regular text categorization, hypertext categorization, and hierarchical text classification.

To allow acquisition of new datasets with predefined characteristics, we defined a set of properties that characterize datasets as a whole, as well as individual categories that comprise them. We first introduced Maximum Achievable Accuracy (MAA) as an intrinsic measure of dataset difficulty, and then developed two kinds of distance metrics that predict the categorization difficulty of a dataset without actually examining the full text of the documents. These metrics analyze the location of categories in the hierarchy tree, as well as textual descriptions of categories and annotations of documents. We empirically showed that the text-based metric possesses high predictive power for estimating the separability of a pair of categories. The edge-counting graph metric is somewhat less reliable, but is much more efficient computationally. We also observed that MAA can be used as a measure of similarity between sets of documents, quantifying the ease of separating them with a text classifier. Since texts acquired from the WWW are often plagued with noise and are generally quite different in nature from formal written English found in printed publications, we reported specific steps we undertook to filter the data and monitor its quality during acquisition.

Finally, we established a new repository of text categorization datasets, which currently contains several hundred datasets at various levels of difficulty that we generated using the proposed methodology. This collection is available at `http://techtc.cs.technion.ac.il`, along with ancillary statistics and measured classifier performance. The collection continues to grow, and its growth rate is only limited by bandwidth and storage resources. Having a wide variety of datasets in a centralized repository will allow researchers to perform a wide range of repeatable experiments. The $\mathcal{A}$CCIO system that performs parameterized dataset acquisition from the ODP will be released at a later stage. Using a subset of these datasets, we developed a novel criterion that assesses *feature redundancy* and predicts the utility of feature selection for TC (Gabrilovich and Markovitch, 2004).

This research can be extended in several directions. We plan to investigate more sophisticated distance metrics that overcome the drawbacks of the basic metrics we described herein. The graph metric does not account for the fact that two nodes whose common ancestor is close to the hierarchy root are much less related, than two nodes at the same edge distance whose common ancestor resides deep in the tree. The graph metric may also produce unreliable values for extremely long hierarchy paths, which contain too many intermediate generalizations. The WordNet-based text metric is obviously undefined for words not found in WordNet (e.g., neologisms, narrow technical terms, and proper names); currently, if such a word is present in both documents, we take the value in equation (B.2) to be zero, otherwise, we ignore this word. The text metric may also be inaccurate for documents with only a few words. Following standard IR practice, we also tested the conventional cosine metric to compare bag-of-word vectors of categories and documents, but empirically found it to be inadequate. Most of the values of the cosine measure clustered near its extremes (0 and 1), while the mid-range was very sparsely populated; we attribute this phenomenon to the lack of any background knowledge about word semantics (as, for example, provided by WordNet in the text metric).

We intend to investigate additional parameters of categories that will allow to exercise better control over the properties of generated datasets. Of particular interest and practical importance are filtering techniques for cleaning the data downloaded from the Web, and we plan to study this issue in greater depth using focused crawling techniques. We also plan to develop more elaborate algorithms that locate pairs of categories subject to user's requirements.

We further intend to construct larger datasets consisting of more than two categories; to do so, category similarity metrics will need to be generalized appropriately to consider mutual distances in a group of categories. We also intend to generate datasets from additional document directories that contain high quality noise-free articles.

References

Adafre, Sisay Fissaha and Maarten de Rijke. 2005. Discovering missing links in wikipedia. In *Proceedings of the Workshop on Link Discovery: Issues, Approaches and Applications (LinkKDD-2005).*

Ando, Rie Kubota and Tong Zhang. 2005a. Framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, pages 1817–1853.

Ando, Rie Kubota and Tong Zhang. 2005b. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 1–9, Ann Arbor, MI, June.

Baeza-Yates, Ricardo and Berthier Ribeiro-Neto. 1999. *Modern Information Retrieval.* Addison Wesley, New York, NY.

Baker, L. Douglas and Andrew K. McCallum. 1998. Distributional clustering of words for text classification. In W. Bruce Croft, Alistair Moffat, Cornelis J. Van Rijsbergen, Ross Wilkinson, and Justin Zobel, editors, *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 96–103, Melbourne, AU. ACM Press, New York, US.

Ballesteros, Lisa and W. Bruce Croft. 1997. Phrasal translation and query expansion techniques for cross-language information retrieval. In *Proceedings of the 20th ACM International Conference on Research and Development in Information Retrieval*, pages 84–91.

Basili, Roberto, Alessandro Moschitti, and Maria T. Pazienza. 2000. Language-sensitive text classification. In *Proceedings of RIAO-00, 6th International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 331–343, Paris, France.

Bekkerman, Ron. 2003. Distributional clustering of words for text categorization. Master's thesis, Technion.

Bekkerman, Ron, Ran El-Yaniv, Naftali Tishby, and Yoad Winter. 2001. On feature distributional clustering for text categorization. In *Proceedings of the 24th International Conference on Research and Development in Information Retrieval*, pages 146–153, September.

Bekkerman, Ron, Ran El-Yaniv, Naftali Tishby, and Yoad Winter. 2003. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 3:1183–1208, March.

Bennett, Paul N., Susan T. Dumais, and Eric Horvitz. 2002. Probabilistic combination of text classifiers using reliability indicators: Models and results. In *Proceedings of the 25rd ACM International Conference on Research and Development in Information Retrieval*, pages 207–215.

Bennett, Paul N., Susan T. Dumais, and Eric Horvitz. 2003. Inductive transfer for text classification using generalized reliability indicators. In *Proceedings of the ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*.

Bennett, Paul N., Susan T. Dumais, and Eric Horvitz. 2005. The combination of text classifiers using reliability indicators. *Information Retrieval*, 8(1):67–100.

Blake, C.L. and C.J. Merz. 1998. UCI Repository of machine learning databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

Bloehdorn, Stephan and Andreas Hotho. 2004. Boosting for text classification with semantic features. In *Proceedings of the MSW 2004 Workshop at the 10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 70–87.

Brank, Janez, Marko Grobelnik, Natasa Milic-Frayling, and Dunia Mladenic. 2002. Interaction of feature selection methods and linear classification models. In *Workshop on Text Learning held at ICML-2002*.

Brill, Eric. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565.

Buchanan, B. G. and E. A. Feigenbaum. 1982. Forward. In R. Davis and D. B. Lenat, editors, *Knowledge-Based Systems in Artificial Intelligence*. McGraw-Hill.

Budanitsky, Alexander and Graeme Hirst. 2001. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Proc. of the NAACL 2001 Workshop on WordNet and Other Lexical Resources*.

Budanitsky, Alexander and Graeme Hirst. 2006. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.

Cafarella, Michael, Doug Downey, Stephen Soderland, and Oren Etzioni. 2005. Knowitnow: Fast, scalable information extraction from the web. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Vancouver, Canada, October.

Cai, Lijuan and Thomas Hofmann. 2003. Text categorization by boosting automatically extracted concepts. In *Proceedings of the 26th International Conference on Research and Development in Information Retrieval*, pages 182–189.

Callan, James. 1993. *Knowledge-Based Feature Generation for Inductive Learning*. Ph.D. thesis, Department of Computer and Information Science, University of Massachusetts. Also available as CMPSCI Technical Report 93-07.

Caropreso, Maria Fernanda, Stan Matwin, and Fabrizio Sebastiani. 2001. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In Amita G. Chin, editor, *Text Databases and Document Management: Theory and Practice*. Idea Group Publishing, Hershey, US, pages 78–102.

Chakrabarti, Soumen, Byron Dom, Rakesh Agrawal, and Prabhakar Raghavan. 1997. Using taxonomy, discriminants, and signatures for navigating in text databases. In *Proceedings of the 23rd VLDB Conference*, pages 446–455.

Chakrabarti, Soumen, Mukul M. Joshi, Kunal Punera, and David M. Pennock. 2002. The structure of broad topics on the web. In *Proc. of the Int'l World Wide Web Conference*.

Chan, Lois Mai. 1999. *A Guide to the Library of Congress Classification*. Libraries Unlimited, 5th edition.

Chen, Stanley and Joshua Goodman. 1996. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 34th Annual Meeting of the ACL*.

Cohen, Doron, Michael Herscovici, Yael Petruschka, Yoelle S. Maarek, Aya Soffer, and Dave Newbold. 2002. Personalized pocket directories for mobile devices. In *Proc. of the Int'l World Wide Web Conference*.

Cohen, William W. 1995. Fast effective rule induction. In *Proceedings of the 12th International Conference on Machine Learning (ICML-95)*, pages 115–123.

Cohen, William W. 2000. Automatically extracting features for concept learning from the web. In *Proceedings of the 17th International Conference on Machine Learning*.

Cristianini, Nello, John Shawe-Taylor, and Huma Lodhi. 2002. Latent semantic kernels. *Journal of Intelligent Information Systems*, 18(2/3):127–152.

Dagan, Ido, Lillian Lee, and Fernando C. N. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, 34(1–3):43–69.

Davidov, Dmitry, Evgeniy Gabrilovich, and Shaul Markovitch. 2004. Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. In *Proceedings of the 27th ACM International Conference on Research and Development in Information Retrieval*, pages 250–257.

Debole, Franca and Fabrizio Sebastiani. 2003. Supervised term weighting for automated text categorization. In *SAC'03*, pages 784–788.

Deerwester, S., S. Dumais, G. Furnas, T. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407.

Dejong, Gerald and Raymond Mooney. 1986. Explanation-based learning: An alternative view. *Machine Learning*, 1(2):145–176.

Demsar, Janez. 2006. Statistical comparison of classifiers over multiple data sets. *JMLR*, 7:1–30.

Dewdney, Nigel, Carol VanEss-Dykema, and Richard MacMillan. 2001. The form is the substance: Classification of genres in text. In *Workshop on HLT and KM held at ACL-2001.*

Dewey, Melvil, Joan S. Mitchell, Julianne Beall, Giles Martin, Winton E. Matthews, and Gregory R. New, editors. 2003. *Dewey Decimal Classification and Relative Index.* Online Computer Library Center (OCLC), 22nd edition.

Dhillon, Inderjit, Subramanyam Mallela, and Rahul Kumar. 2003. A divisive information-theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3:1265–1287, March.

Do, Chuong and Andrew Ng. 2005. Transfer learning for text classification. In *Proceedings of Neural Information Processing Systems (NIPS).*

Downey, Doug, Oren Etzioni, and Stephen Soderland. 2005. A probabilistic model of redundancy in information extraction. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, Edinburgh, Scotand, August.

Duda, R.O. and P.E. Hart. 1973. *Pattern Classification and Scene Analysis.* John Wiley and Sons.

Dumais, Susan and Hao Chen. 2000. Hierarchical classification of web content. In *Proceedings of the 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 256–263.

Dumais, Susan, John Platt, David Heckerman, and Mehran Sahami. 1998. Inductive learning algorithms and representations for text categorization. In *CIKM'98*, pages 148–155.

Etzioni, Oren, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel Weld, and Alexander Yates. 2004. Webscale information extraction in knowitall (preliminary results). In *Proceedings of the 13th International World Wide Web Conference (WWW'04)*, New York, USA, May. ACM Press.

Fawcett, Tom. 1991. Feature discovery for inductive concept learning. Technical Report COINS Technical Report 91-8, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts.

Fawcett, Tom. 1993. *Feature Discovery for Problem Solving Systems*. Ph.D. thesis, UMass, May.

Fellbaum, Christiane, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Finkelstein, Lev, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002a. Placing search in context: The concept revisited. *ACM TOIS*, 20(1):116–131, January.

Finkelstein, Lev, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002b. WordSimilarity-353 test collection. `http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/wordsim353.html`.

Forman, George. 2003. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, March.

Fuernkranz, Johannes, Tom Mitchell, and Ellen Riloff. 2000. A case study in using linguistic phrases for text categorization on the WWW. In Mehran Sahami, editor, *Learning for Text Categorization: Proceedings of the 1998 AAAI/ICML Workshop*. AAAI Press, Madison, Wisconsin, pages 5–12.

Fuhr, Norbert. 1985. A probabilistic model of dictionary-based automatic indexing. In *Proceedings of RIAO-85, 1st International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 207–216, Grenoble, France.

Gabrilovich, Evgeniy, Susan Dumais, and Eric Horvitz. 2004. Newsjunkie: Providing personalized newsfeeds via analysis of information novelty. In

Proceedings of the Thirteenth International World Wide Web Conference (WWW2004), pages 482–490, New York, NY, May. ACM Press.

Gabrilovich, Evgeniy and Shaul Markovitch. 2004. Text categorization with many redundant features: Using aggressive feature selection to make SVMs competitive with C4.5. In *Proceedings of the 21st International Conference on Machine Learning*, pages 321–328.

Gabrilovich, Evgeniy and Shaul Markovitch. 2005. Feature generation for text categorization using world knowledge. In *IJCAI'05*, pages 1048–1053.

Gabrilovich, Evgeniy and Shaul Markovitch. 2006a. Computing semantic relatedness of words and texts in wikipedia-derived semantic space. Technical report CIS-2006-04, Computer Science Department, Technion—Israel Institute of Technology, Haifa, Israel, July.

Gabrilovich, Evgeniy and Shaul Markovitch. 2006b. Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge. In *AAAI'06*, pages 1301–1306, July.

Gabrilovich, Evgeniy and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, January.

Galavotti, Luigi, Fabrizio Sebastiani, and Maria Simi. 2000. Experiments on the use of feature selection and negative evidence in automated text categorization. In Jose Borbinha and Thomas Baker, editors, *Proceedings of ECDL-00, 4th European Conference on Research and Advanced Technology for Digital Libraries*, pages 59–68, Lisbon, Portugal. Published in the "Lecture Notes for Computer Science" series, number 1923, Springer Verlag.

Ghani, Rayid, Rosie Jones, Dunja Mladenic, Kamal Nigam, and Sean Slattery. 2000. Data mining on symbolic knowledge extracted from the web. In *Workshop on Text Mining at the 6th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

Giles, Jim. 2005. Internet encyclopaedias go head to head. *Nature*, 438:900–901.

Goldberg, Andrew and Xiaojin Zhu. 2006. Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization. In *Workshop on Textgraphs: Graph-based Algorithms for Natural Language Processing, HLT-NAACL 2006*.

Han, Eui-Hong (Sam) and George Karypis. 2000. Centroid-based document classification: Analysis and experimental results. In *PKDD'00*, September.

Harman, Donna. 1992. The DARPA TIPSTER project. In *SIGIR Forum*, volume 26(2). ACM, pages 26–28.

Hayes, Philip J., Peggy M. Andersen, Irene B. Nirenburg, and Linda M. Schmandt. 1990. TCS: a shell for content-based text categorization. In *Proceedings of CAIA-90, 6th IEEE Conference on Artificial Intelligence Applications*, pages 320–326, Santa Barbara, US. IEEE Computer Society Press, Los Alamitos, US.

Hersh, William, Chris Buckley, T.J. Leone, and David Hickam. 1994. OHSUMED: An interactive retrieval evaluation and new large test collection for research. In *SIGIR'94*, pages 192–201.

Hripcsak, George, Carol Friedman, Philip O. Alderson, William DuMouchel, Stephen B. Johnson, and Paul D. Clayton. 1995. Unlocking clinical data from narrative reports: a study of natural language processing. *Annals of Internal Medicine*, 122(9):681–688.

Hu, Yuh-Jyh and Dennis Kibler. 1996. A wrapper approach for constructive induction. In *AAAI-96*.

Hull, David A. 1994. Improving text retrieval for the routing problem using latent semantic indexing. In W. Bruce Croft and Cornelis J. Van Rijsbergen, editors, *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 282–289, Dublin, IE. Springer Verlag, Heidelberg, DE.

Jain, Anil K., Robert P.W. Duin, and Jianchang Mao. 2000. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), January.

Jarmasz, Mario. 2003. Roget's thesaurus as a lexical resource for natural language processing. Master's thesis, University of Ottawa.

Jiang, Jay J. and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *ROCLING'97*.

Joachims, Thorsten. 1998. Text categorization with support vector machines: Learning with many relevant features. In *ECML'98*, pages 137–142.

Joachims, Thorsten. 1999a. Making large-scale SVM learning practical. In B. Schoelkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*. The MIT Press.

Joachims, Thorsten. 1999b. Transductive inference for text classification using support vector machines. In *Proceedings of the 13th International Conference on Machine Learning.*

John, George H., Ron Kohavi, and Karl Pfleger. 1994. Irrelevant features and the subset selection problem. In *Proceedings of the 11th International Conference on Machine Learning*, pages 121–129.

Koller, Daphne and Mehran Sahami. 1997. Hierarchically classifying documents using very few words. In *Proceedings of the 14th International Conference on Machine Learning*, pages 170–178.

Kudenko, Daniel and Haym Hirsh. 1998. Feature generation for sequence categorization. In *Proceedings of the 15th Conference of the American Association for Artificial Intelligence*, pages 733–738.

Kumaran, Giridhar and James Allan. 2004. Text classification and named entities for new event detection. In *Proceedings of the 27th ACM International Conference on Research and Development in Information Retrieval*, pages 297–304.

Labrou, Yannis and Tim Finin. 1999. Yahoo! as an ontology—using Yahoo! categories to describe documents. In *Proc. of the 8th Int'l Conference on Information and Knowledge Management*, pages 180–187.

Lam, Wai and Kwok-Yin Lai. 2001. A meta-learning approach for text categorization. In *Proceedings of the 24th International Conference on Research and Development in Information Retrieval*, pages 303–309.

Lang, Ken. 1995. Newsweeder: Learning to filter netnews. In *ICML'95*, pages 331–339.

Lee, Lillian. 1999. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the ACL.*

Lee, Michael D., Brandon Pincombe, and Matthew Welsh. 2005. An empirical evaluation of models of text document similarity. In *CogSci2005*, pages 1254–1259.

Lenat, D. and R. Guha. 1990. *Building Large Knowledge Based Systems.* Addison Wesley.

Lenat, D. B. 1995. CYC: A large-scale investment in knowledge infrastructure. *CACM*, 38(11), November.

Lenat, D.B. and E.A. Feigenbaum. 1990. On the thresholds of knowledge. *Artificial Intelligence*, 47:185–250.

Lenat, Douglas B. 1997. From 2001 to 2001: Common sense and the mind of HAL. In *HAL's Legacy*. The MIT Press, pages 194–209.

Lenat, Douglas B., Ramanathan V. Guha, Karen Pittman, Dexter Pratt, and Mary Shepherd. 1990. CYC: Towards programs with common sense. *CACM*, 33(8), August.

Leopold, Edda and Joerg Kindermann. 2002. Text categorization with support vector machines: How to represent texts in input space. *Machine Learning*, 46:423–444.

Lewis, David D. 1991. Evaluating text categorization. In *Proc. of the Speech and Natural Language Workshop*, pages 312–318. Morgan Kaufmann, February.

Lewis, David D. 1992a. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th ACM International Conference on Research and Development in Information Retrieval*, pages 37–50.

Lewis, David D. 1992b. *Representation and learning in information rerieval.* Ph.D. thesis, Department of Computer Science, University of Massachusetts, February.

Lewis, David D. and W. Bruce Croft. 1990. Term clustering of syntactic phrases. In *Proceedings of the 13th ACM International Conference on Research and Development in Information Retrieval*, pages 385–404.

Lewis, David D., Robert E. Schapire, James P. Callan, and Ron Papka. 1996. Training algorithms for linear text classifiers. In *Proceedings of the 19th ACM International Conference on Research and Development in Information Retrieval*, pages 298–306.

Lewis, David D., Yiming Yang, Tony Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. *JMLR*, 5:361–397.

Lin, Dekang. 1998. An information-theoretic definition of word similarity. In *ICML'98.*

Liu, Tao, Zheng Chen, Benyu Zhang, Wei-ying Ma, and Gongyi Wu. 2004. Improving text classification using local latent semantic indexing. In *ICDM'04*, pages 162–169.

Manning, Christopher D. and Hinrich Schuetze. 2000. *Foundations of Statistical Natural Language Processing*. The MIT Press.

Markovitch, Shaul and Danny Rosenstein. 2002. Feature generation using general constructor functions. *Machine Learning*, 49(1).

Marlow, Cameron, Mor Naaman, danah boyd, and Marc Davis. 2006. Position paper, tagging, taxonomy, flickr, article, toread. In *Collaborative Web Tagging Workshop, The 15th International World Wide Web Conference*, Edinburgh, Scotland, May.

Maron, M. 1961. Automatic indexing: an experimental inquiry. *Journal of the ACM*, 8(3):404–417.

Matheus, Christopher J. 1991. The need for constructive induction. In L.A. Birnbaum and G.C. Collins, editors, *8th Int'l Workshop on Machine Learning*, pages 173–177.

Matheus, Christopher J. and Larry A. Rendell. 1989. Constructive induction on decision trees. In *Proceedings of the 11th International Conference on Artificial Intelligence*, pages 645–650.

Mcilwaine, I.C. 2000. *The Universal Decimal Classification: Guide to its Use.* UDC Consortium.

Meng, Weiyi, Wenxian Wang, Hongyu Sun, and Clement Yu. 2002. Concept hierarchy-based text database categorization. *Knowledge and Information Systems*, 4:132–150.

MeSH. 2003. Medical subject headings (MeSH). National Library of Medicine. `http://www.nlm.nih.gov/mesh`.

Mihalcea, Rada. 2003. Turning wordnet into an information retrieval resource: Systematic polysemy and conversion to hierarchical codes. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 17(1):689–704.

Mihalcea, Rada, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI'06*, July.

Mikheev, Andrei. 1999. Feature lattices and maximum entropy models. *Information Retrieval.*

Miller, George A. and Walter G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.

Mitchell, Tom, Richard Keller, and Smadar Kedar-Cabelli. 1986. Explanation-based generalization: A unifying view. *Machine Learning*, 1(1):47–80.

Mitra, Mandar, Amit Singhal, and Chris Buckley. 1998. Improving automatic query expansion. In *Proceedings of the 21st ACM International Conference on Research and Development in Information Retrieval*, pages 206–214.

Mladenic, Dunja. 1998a. Feature subset selection in text learning. In *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 95–100.

Mladenic, Dunja. 1998b. Turning Yahoo into an automatic web-page classifier. In *Proceedings of ECAI'98, 13th European Conference on Artificial Intelligence*, pages 473–474.

Mladenic, Dunja and Marko Grobelnik. 1998a. Feature selection for clasification based on text hierarchy. In *Working notes of Learning from Text and the Web, Conference on Automated Learning and Discovery (CONALD-98).*

Mladenic, Dunja and Marko Grobelnik. 1998b. Word sequences as features in text-learning. In *Proc. of ERK-98, 7th Electrotechnical and Computer Science Conference*, pages 145–148.

Murphy, Patrick M. and Michael J. Pazzani. 1991. ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees. In *Proceedings of the 8th International Conference on Machine Learning*, pages 183–188. Morgan Kaufmann.

Nigam, Kamal, Andrew McCallum, and Tom Mitchell. 2006. Semi-supervised text classification using EM. In Olivier Chapelle, Bernhard Schoelkopf, and Alexander Zien, editors, *Semi-Supervised Learning.* MIT Press, Boston, MA.

Nigam, Kamal, Andrew McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2-3):103–134.

2006. Open Directory Project. `http://www.dmoz.org`.

Pagallo, Giulia and David Haussler. 1990. Boolean feature discovery in empirical learning. *Machine Learning*, 5(1):71–99.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *EMNLP'02*, pages 79–86.

Peng, Fuchun, Dale Schuurmans, and Shaojun Wang. 2004. Augmenting naive Bayes classifiers with statistical language models. *Information Retrieval*, 7(3-4):317–345.

Peng, Fuchun and Dale Shuurmans. 2003. Combining naive Bayes and n-gram language models for text classification. In *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR-03)*.

Pereira, Fernando, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of english words. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 183–190.

Pincombe, Brandon. 2004. Comparison of human and latent semantic analysis (LSA) judgements of pairwise document similarities for a news corpus. Technical Report DSTO-RR-0278, Information Sciences Laboratory, Defence Science and Technology Organization, Department of Defense, Australian Government, September.

Porter, M.F. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Press, William H., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1997. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press.

Quinlan, J. Ross. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Rada, Roy and Ellen Bicknell. 1989. Ranking documents with a thesaurus. *JASIS*, 40(5):304–310, September.

Raina, Rajat, Andrew Ng, and Daphne Koller. 2006. Constructing informative priors using transfer learning. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, Pittsburgh, PA.

Raskutti, Bhavani, Herman Ferra, and Adam Kowalczyk. 2001. Second order features for maximizing text classification performance. In L. De Raedt and P. Flach, editors, *Proceedings of the European Conference on Machine Learning (ECML)*, Lecture notes in Artificial Intelligence (LNAI) 2167, pages 419–430. Springer-Verlag.

Resnik, Philip. 1999. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *JAIR*, 11:95–130.

Reuters, 1997. *Reuters-21578 text categorization test collection, Distribution 1.0.* Reuters. `daviddlewis.com/resources/testcollections/reuters21578`.

Riloff, Ellen. 1995. Little words can make a big difference for text classification. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval*, pages 130–136, Seattle, US. ACM Press.

Rocchio, J. 1971. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing.* Prentice Hall, pages 313–323.

Rogati, Monica and Yiming Yang. 2002. High-performing feature selection for text classification. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM'02)*, pages 659–661.

Roget, Peter. 1852. *Roget's Thesaurus of English Words and Phrases.* Longman Group Ltd.

Rorvig, M.E. 1999. Images of similarity: A visual exploration of optimal similarity metrics and scaling properties of TREC topic-document sets. *Journal of the American Society of Information Science*, 50(8):639–651.

Rose, Tony, Mark Stevenson, and Miles Whitehead. 2002. The Reuters Corpus Volume 1—from yesterday's news to tomorrow's language resources. In *Proceedings of the Third International Conference on Language Resources and Evaluation.*

Rowling, J.K. 1997. *Harry Potter and the Philosopher's Stone.* Bloomsbury.

Rowling, J.K. 2001. *Harry Potter and the Goblet of Fire.* Bloomsbury.

Rubenstein, Herbert and John B. Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Ruiz, Miguel E. and Padmini Srinivasan. 2002. Hierarchical text categorization using neural networks. *Information Retrieval*, 5:87–118.

Ruthven, Ian and Mounia Lalmas. 2003. A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review*, 18(2):95–145.

Sable, Carl, Kathleen McKeown, and Kenneth W. Church. 2002. NLP found helpful (at least for one text categorization task). In *Conference on Empirical Methods in Natural Language Processing.*

Sahami, Mehran, Susan Dumais, David Heckerman, and Eric Horvitz. 1998. A bayesian approach to filtering junk e-mail. In *AAAI 98 Workshop on Text Categorization*, July.

Sahami, Mehran and Timothy Heilman. 2006. A web-based kernel function for measuring the similarity of short text snippets. In *WWW'06*. ACM Press, May.

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

Salton, Gerard, editor. 1971. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice Hall.

Salton, Gerard and Chris Buckley. 1988. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

Santamaria, Celina, Julio Gonzalo, and Felisa Verdejo. 2003. Automatic association of web directories to word senses. *Computational Linguistics*, 29(3).

Scott, Sam. 1998. Feature engineering for a symbolic approach to text classification. Master's thesis, U. Ottawa.

Scott, Sam and Stan Matwin. 1999. Feature engineering for text classification. In *Proceedings of the 16th International Conference on Machine Learning*, pages 379–388.

Sebastiani, Fabrizio. 1999. A tutorial on automated text categorisation. In Analia Amandi and Ricardo Zunino, editors, *Proceedings of ASAI-99, 1st Argentinian Symposium on Artificial Intelligence*, pages 7–35, Buenos Aires, Argentina.

Sebastiani, Fabrizio. 2002. Machine learning in automated text categorization. *ACM Comp. Surv.*, 34(1):1–47.

Singhal, Amit. 1998. AT&T at TREC-6. In *Proceedings of the 6th Text REtrieval Conference (TREC-6)*, pages 215–226.

Soucy, Pascal and Guy W. Mineau. 2001. A simple feature selection method for text classification. In *IJCAI*, pages 897–902.

Strube, Michael and Simon Paolo Ponzetto. 2006. WikiRelate! Computing semantic relatedness using Wikipedia. In *AAAI'06*, Boston, MA.

Tishby, Naftali, Fernando Pereira, and William Bialek. 1999. The information bottleneck method. In *Invited paper to the 37th Annual Allerton Conference on Communication, Control,and Computing.*

Toyama, K. and E. Horvitz. 2000. Bayesian modality fusion: Probabilistic integration of multiple vision algorithms for head tracking. In *Proceedings of the 4th Asian Conference on Computer Vision.*

Turney, Peter. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the ACL.*

Turney, Peter. 2005. Measuring semantic similarity by latent relational analysis. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 1136–1141, Edinburgh, Scotland.

Turney, Peter. 2006. Similarity of semantic relations. *Computational Linguistics*, 32(3):379–416.

Turney, Peter and Michael L. Littman. 2002. Unsupervised learning of semantic orientation from a hundred-billion-word corpus. Technical Report ERB-1094, National Research Council Canada, May.

Urena-Lopez, L. Alfonso, Manuel Buenaga, and Jose M. Gomez. 2001. Integrating linguistic resources in TC through WSD. *Computers and the Humanities*, 35:215–230.

van Rijsbergen, C. J. 1979. *Information Retrieval.* Butterworths, London, 2 edition.

Vapnik, V.N. 1995. *The nature of statistical learning theory.* Springer-Verlag.

Voorhees, Ellen M. 1994. Query expansion using lexical-semantic relations. In *Proceedings of the 17th International Conference on Research and Development in Information Retrieval*, pages 61–69.

Voorhees, Ellen M. 1998. Using wordnet for text retrieval. In Christiane Fellbaum, editor, *WordNet, an Electronic Lexical Database.* The MIT Press.

Wang, Bill B., R.I. McKay, Hussein A. Abbass, and Michael Barlow. 2003. A comparative study for domain ontology guided feature extraction. In *Proceedings of the 26th Australian Computer Science Conference (ASCS-2003)*, pages 69–78.

Wermter, Stefan and Chihli Hung. 2002. Selforganizing classification on the Reuters news corpus. In *Proceedings of the International Conference on Computational Linguistics (COLING).*

Widrow, B. and S.D. Stearns. 1985. *Adaptive Signal Processing.* Prentice Hall.

Wikipedia. 2006. Wikipedia, the free encyclopedia. `http://en.wikipedia.org`.

Wong, S.K.M., Wojciech Ziarko, and Patrick C.N. Wong. 1985. Generalized vector spaces model in information retrieval. In *Proceedings of the 8th ACM International Conference on Research and Development in Information Retrieval*, pages 18–25.

Wu, Huiwen and Dimitrios Gunopulos. 2002. Evaluating the utility of statistical phrases and latent semantic indexing for text classification. In *ICDM'02*, pages 713–716.

Xu, Jinxi and W. Bruce Croft. 1996. Query expansion using local and global document analysis. In *Proceedings of the 19th International Conference on Research and Development in Information Retrieval*, pages 4–11.

Xu, Jinxi and W. Bruce Croft. 2000. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems*, 18(1):79–112.

Yang, Yiming. 1999. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69–90.

Yang, Yiming. 2001. A study on thresholding strategies for text categorization. In *Proceedings of the 24th International Conference on Research and Development in Information Retrieval*, pages 137–145.

Yang, Yiming and Xin Liu. 1999. A re-examination of text categorization methods. In *SIGIR'99*, pages 42–49.

Yang, Yiming and Jan Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420.

Yang, Yiming, Sean Slattery, and Rayid Ghani. 2002. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 18(2/3):219–241.

Yu, Shipeng, Deng Cai, Ji-Rong Wen, and Wei-Ying Ma. 2003. Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In *Proceedings of the 12th International World Wide Web Conference (WWW'03)*, Budapest, Hungary, May. ACM Press.

Zelikovitz, Sarah and Haym Hirsh. 2000. Improving short-text classification using unlabeled background knowledge to assess document similarity. In *Proceedings of the 17th International Conference on Machine Learning.*

Zelikovitz, Sarah and Haym Hirsh. 2001. Using LSI for text classification in the presence of background text. In *Proceedings of the Conference on Information and Knowledge Management.*

Zesch, Torsten and Iryna Gurevych. 2006. Automatically creating datasets for measures of semantic relatedness. In *Proceedings of the ACL Workshop on Linguistic Distances*, pages 16–24, Sydney, Australia, July.

Zobel, Justin and Alistair Moffat. 1998. Exploring the similarity space. *ACM SIGIR Forum*, 32(1):18–34.

בנית תכוניות לאיחזור מידע טקסטואלי
בעזרת ידע העולם

יבגני גברילוביץ

בנית תכוניות לאיחזור מידע טקסטואלי
בעזרת ידע העולם

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
דוקטור לפילוסופיה

יבגני גברילוביץ

הוגש לסנט הטכניון — מכון טכנולוגי לישראל

כסלו תשס"ז חיפה דצמבר 2006

המחקר נעשה בהנחיית דר' שאול מרקוביץ
בפקולטה למדעי המחשב

הכרת תודה

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי

השקעה בידע תמיד מניבה את התשואה הגבוהה ביותר.

-- בנג'מין פרנקלין

תוכן ענינים

תקציר 1

1 מבוא 5

1.1 הפתרון המוצע 7

1.2 תרומות מדעיות של המחקר 10

1.3 מבנה החיבור 11

2 רקע 13

2.1 סיווג טקסטים 13

2.1.1 תכוניות של מסמכים 14

2.1.2 בחירת תכוניות 16

2.1.3 קביעת ערכי תכוניות 19

2.1.4 שיטות לאומדן דיוק הסיווג 20

2.1.5 הערות על הערכת ניסויים 22

2.2 בעיות בגישה המבוססת על אוסף מילים 22

2.3 בנית תכוניות 25

3 מתודולוגיה לבנית תכוניות 29

3.1 מבוא 29

3.2 אילוצים על מאגרי ידע מתאימים 31

3.3 יצירת בונה התכוניות 33

3.3.1 בחירת אטריבוטים 34

3.3.2 בנית תכוניות 34

3.4 בנית תכוניות מבוססת הקשר 35

3.4.1 ניתוח הקשרים מקומיים 36

3.4.2 בחירת תכוניות 37

3.4.3 קביעת ערכי תכוניות 38

3.4.4 ניתוח מעמיק יותר של הדוגמא 38

3.5 שימוש במאגרי ידע הירככיים 39

3.6 שימוש במאגרי ידע שמגדירים יחסים בין מושגים 42

4 מימוש המתודולוגיה לבנית תכוניות בעזרת המדריך הפתוח וויקיפדיה 45

4.1 בנית תכוניות בעזרת המדריך הפתוח 45

4.1.1 איסוף ידע נוסף באמצעות האינטרנט 46

4.1.2 סינון רעש ובחירת אטריבוטים 46

4.1.3 פרטי מימוש 50

4.2 בנית תכוניות בעזרת ויקיפדיה 51

4.2.1 ויקיפדיה כמאגר ידע 53

4.2.2 תיכון בונה התכוניות 54

4.2.3 שימוש בקישורים בין מאמרים 55

4.2.4 פרטי מימוש 57

5 הערכה של בנית תכוניות עבור סיווג טקסטים 61

5.1 אוספי נתונים 61

5.1.1 Reuters-21578 61

5.1.2 20 Newsgroups 62

5.1.3 Movie Reviews 62

5.1.4 Reuters Corpus Version 1 63

5.1.5 OHSUMED 64

5.1.6 מסמכים קצרים 65

5.1.7 הרכשה אוטומטית של אוספי נתונים 67

5.2 שיטת הניסויים 68

5.2.1 תשתית לסיווג טקסטים 69

5.2.2 ביצועי המערכת הבסיסית 70

5.2.3 שימוש בבונה התכוניות 70

5.3 שימוש בבונה התכוניות מבוסס המדריך הפתוח 70

5.3.1 ניתוח איכותי של בנית התכוניות 70

5.3.2 ההשפעה של בנית התכוניות 76

5.3.3 ההשפעה של בנית התכוניות מבוססת ההקשר 77

5.3.4 ההשפעה של רוחב הידע 77

5.3.5 ההשפעה של בחירת התכוניות 79

5.3.6 ההשפעה של גודל הקטגוריה 81

5.3.7 ההשפעה של בנית התכוניות על סיווג מסמכים קצרים 82

5.3.8 זמן עיבוד 83

5.4 שימוש בבונה התכוניות מבוסס ויקיפדיה 85

5.4.1 ניתוח איכותי של בנית התכוניות 85

5.4.2 ההשפעה של בנית התכוניות 97

5.4.3 ההשפעה של רוחב הידע 99

5.4.4 סיווג מסמכים קצרים 99

5.4.5 שימוש בקישורים בין מאמרים 100

6 שימוש בבנית תכוניות לאומדן קרבה סמנטית בין טקסטים 103
6.1 אנליזה סמנטית מפורשת 103
6.2 הערכה של אנליזה סמנטית מפורשת 105
6.2.1 אוספי נתונים 106
6.2.2 ההשפעה של הידע החיצוני 107

7 סקירת ספרות 109
7.1 גישות קודמות להרחבת קבוצת התכוניות 109
7.2 בנית תכוניות עבור סיווג טקסטים 109
7.2.1 בנית תכוניות בעזרת מילונים ממוחשבים 111
7.2.2 השוואת מקורות ידע: המדריך הפתוח לעומת WordNet . . . 113
7.2.3 שימוש בדוגמאות לא מתוייגות 116
7.2.4 עבודות רלוונטיות נוספות 117
7.3 דמיון סמנטי וקרבה סמנטית 118

8 סיכום 123

א סיווג טקסטים בנוכחות תכוניות יתירות רבות 127
א.1 מבוא 127
א.2 מתודולוגית ניסויים 129
א.2.1 אוספי נתונים 130
א.2.2 חיזוי ההשפעה של בחירת תכוניות 131
א.2.3 שימוש בתכוניות מבוססות WordNet 132
א.2.4 אלגוריתמים לבחירת תכוניות 133
א.2.5 אלגוריתמים לסיווג טקסטים 133
א.3 הערכה אמפירית 134
א.3.1 ביצועי המערכת הבסיסית 134
א.3.2 חיזוי ההשפעה של בחירת תכוניות 136
א.3.3 השוואת מסווגים 136
א.3.4 ההשפעה של קבוצות התכוניות השונות 138
א.3.5 ההשפעה של האלגוריתמים השונים לבחירת תכוניות 138
א.3.6 בחינת הרלוונטיות של התכוניות 138
א.4 דיון בתוצאות 139

ב יצירת אוספי נתונים מתוייגים לסיווג טקסטים 143
ב.1 מבוא 143
ב.2 פרמטרים של אוספי נתונים 146
ב.2.1 פונקציות לאומדן מרחק 147
ב.2.2 תכונות של קטגוריות 150
ב.3 מתודולוגיה ליצירה אוטומטית של אוספי נתונים 151
ב.3.1 הרכשת הנתונים 151

ב.3.2 סינון הנתונים לבקרת הרעש 153
ב.4 הערכה אמפירית 154
ב.4.1 הרכשת הנתונים 154
ב.4.2 תשתית לסיווג טקסטים 154
ב.4.3 הקורלציה בין מטריקות המרחק לבין דיוק הסיווג 155
ב.4.4 הקורלציה בין מטריקות המרחק לבין MAA 156
ב.4.5 רב-גוניות של יצירת אוספי הנתונים 157
ב.5 סיכום 159

רשימת מקורות 163

תקציר i

רשימת איורים

3.1 הגישה הקיימת לסיווג טקסטים 30
3.2 בנית מסווג טקסטים בעזרת השיטה המוצעת 30
3.3 יצירת בונה התכוניות 35
3.4 בנית תכוניות 35
3.5 בנית תכוניות בשיטת רזולוציה מרובה 36
3.6 דוגצא לבנית תכוניות 39
3.7 יצירת בונה תכוניות בעזרת מאגר ידע היררכי 41
3.8 בנית תכוניות בעזרת אונטולוגיה היררכית 42
3.9 בנית תכוניות בעזרת יחסים בין מושגים 43

4.1 בנית תכוניות תוך שימוש בקישורים 57
4.2 בנית תכוניות תוך שימוש רק בקישורים למושגים כלליים יותר 58

5.1 שינוי אורך ההקשר 79
5.2 בחירת תכוניות (Movies) 81
5.3 בחירת תכוניות (RCV1/Topic-16) 81
5.4 שיפור ממוצע כתלות בגודל הקטגוריה (RCV1/Industry) 82
5.5 שיפור ממוצע כתלות בגודל הקטגוריה (RCV1/Topic) 83

6.1 מנתח סמנטי מבוסס ידע 106

א.1 התפלגות התכוניות באוספי נתונים שונים 132
א.2 ביצועי המערכת הבסיסית 135
א.3 שיפור בדיוק של SVM ברמות השונות של בחירת תכוניות 135
א.4 השוואת המסווגים תוך שימוש בכל התכוניות 137
א.5 סיווג באמצעות אוסף מילים 139
א.6 סיווג באמצעות קבוצת התכוניות המורחבת 139
א.7 התלות בין הדיוק של SVM לבין בחירת התכוניות 140
א.8 התלות בין הדיוק של C4.5 לבין בחירת התכוניות 140
א.9 ביטול התכוניות הטובות ביותר 141

ב.1 איתור קטגוריות הנמצאות במרחק הרצוי 152
ב.2 דיוק SVM כתלות במרחק הגרפי 156

ב.3 דיוק SVM כתלות במרחק הטקסטואלי 156
ב.4 המרחק הטקסטואלי כתלות במרחק הגרפי 157
ב.5 MAA כתלות במרחק הגרפי 158
ב.6 MAA כתלות במרחק הטקסטואלי 158
ב.7 התפלגות המרחקים הגרפיים במדריך הפתוח 159
ב.8 התפלגות המרחקים הטקסטואליים במדריך הפתוח 159

רשימת טבלאות

4.1 דוגמאות לבחירת אטריבוטים 49

5.1 הגדרת קבוצות קטגוריות באוסף הנתונים RCV1 65
5.2 הגדרת קבוצות קטגוריות באוסף הנתונים OHSUMED 66
5.3 ביצועי המערכת הבסיסית 71
5.4 עשר התכוניות הראשונות עבור משפט לדוגמא 76
5.5 עשר התכוניות הראשונות עבור משפט לדוגמא 77
5.6 סיווג טקסטים עם ובלי בנית תכוניות 78
5.7 בנית תכוניות באמצעות חלקים של המדריך הפתוח 80
5.8 סיווג מסמכים קצרים עם ובלי בנית תכוניות 84
5.9 גדלים של אוספי הנתונים 85
5.10 ההשפעה של בנית התכוניות 98
5.11 השוואת גדלים של שתי גרסאות של ויקיפדיה 99
5.12 ההשפעה של הגרסה החדשה יותר של ויקיפדיה 100
5.13 בנית תכוניות עבור מסמכים קצרים 101
5.14 שימוש בקישורים בין מאמרים לבנית תכוניות עבור מסמכים קצרים . 102

6.1 התוצאות עבור מילים בודדות 108
6.2 התוצאות עבור טקסטים שלמים 108

א.1 דיוק המסווגים ברמות השונות של בחירת תכוניות 137
א.2 מובהקות סטטיסטית של הבדלי הדיוק בין המסווגים 137

תקציר

סיווג טקסטים עוסק בתיוג אוטומטי של טקסטים בקטגוריות. למשימה זו ישנם שימושים רבים, כגון זיהוי דואר זבל, ארגון אוספי מסמכים לפי נושא, איתור דיווחים רלוונטיים בתחום המודיעין, ועוד. הגישה המקובלת לסיווג טקסטים מייצגת את הטקסט כאוסף מילים בלתי סדור. סיווג המתבסס על ייצוג זה נותן תוצאות סבירות כאשר מדובר בטקסטים ארוכים ובתנאי שיש הרבה דוגמאות לאימון המסווג. יחד עם זאת, ביצועי הגישה יורדים בצורה ניכרת כאשר מדובר בטקסטים קצרים או בקבוצת אימון קטנה.

גישה המבוססת על אוסף מילים הינה מוגבלת מיסודה, שכן היא יכולה להשתמש רק במידע אשר הוזכר במסמכים באופן מפורש, ובתנאי שהמסמכים השונים משתמשים באותו לקסיקון בצורה עקבית. בפרט, הגישה אינה מסוגלת לעשות הכללות מעל מילים, ולכן תתעלם ממילים שבמסמכי הבדיקה אשר אינן מופיעות במסמכי האימון. הגישה מתקשה לטפל גם במילים נרדפות אשר אינן מופיעות בקבוצת האימון בתדירות גבוהה מספיק. כמו כן, גישה זו אינה מסוגלת לבצע התרת רב-משמעות של מילים, שכן מילים רב-משמעיות אינן מעובדות בהקשר המקורי שלהן.

בעבר נעשו ניסיונות רבים לשפר את ייצוג הטקסט באמצעות טכניקות שונות, כגון שימוש בסדרות מילים (n-grams), תיוג מילים בחלקי דיבר, וכן שימוש במנתח תחבירי. שיטות אחרות השתמשו בצברי מילים (clustering) וכן בגישות לצמצום מימדי הבעיה כמו LSA. לניסיונות אלו היתה לרוב הצלחה מוגבלת. יחד עם זאת, לאנשים קל בדרך

כלל לזהות את נושא הטקסט, וזאת בשל אוצר הידע הרב שיש לאנשים.

בעבודה זו אנו מציעים גישה חדשה אשר מתבססת על מאגרי ידע העולם לשם העשרת ייצוג הטקסט. לפני סיווג הטקסט, נשתמש בבונה תכוניות אשר ישתמש בידע חיצוני על מנת להעשיר את אוסף המילים הפשוט בתכוניות חדשות בעלות כושר הפרדה טוב בין הנושאים שיש לסווגם. בניית התכוניות מתבצעת בצורה אוטומטית, תוך שימוש במאגרי ידע אנושי. אודות התפשטות רשת האינטרנט בשנים האחרונות, ישנם היום מאגרי ידע רבים הנגישים לכל, כגון המדריך הפתוח, מדריך האינטרנט של Yahoo, וכן ויקיפדיה.

בניית תכוניות הינה שיטה ידועה בתחום מערכות לומדות, שמטרתה לשדרג את הייצוג המקורי של הדוגמאות בעזרת תכוניות חדשות, בעלות כושר הפרדה משופר. לשיטה זו שימושים רבים, אך בתחום עיבוד טקסטים נוסתה השיטה עד כה פעמים מועטות, ובהצלחה חלקית.

מטרת העבודה הנוכחית הינה להקנות למערכות לומדות גישה למאגרי הידע שיש לבני אדם. בעבודה זו אנו מגדירים מתודולוגיה כללית לשימוש בידע חיצוני, וממשים אותה עבור שני מאגרי ידע מסוימים - המדריך הפתוח וויקיפדיה. על מנת להשתמש באוצרות הידע האלה, אנו בונים מסווג עזר, אשר מסוגל למפות קטעי טקסט למושגים הרלוונטיים במדריך הפתוח ובויקיפדיה. לאחר מכן, אנו משתמשים במושגים אלה בתור תכוניות שמעשירות את אוסף המילים. ייצוג טקסטים במרחב התכוניות המורחב תורם רבות לשיפור דיוק הסיווג.

המדריך הפתוח וויקיפדיה הינם מאגרי הידע הגדולים ביותר מסוגם. שניהם פרי עבודה של מאות אלפי מתנדבים ברחבי התבל. שני המאגרים ניתנים להורדה חופשית בצורה שקלה לעיבוד במחשב. במדריך הפתוח הידע מאורגן בהירככיה עמוקה ועשירה מאד, אשר מכילה הן ידע כללי והן ידע בתחומים ספציפיים כגון רפואה, אומנות ותחומי המדע השונים. ויקיפדיה הינה האנציקלופדיה הממוחשבת הגדולה ביותר בעולם, והיא מאורגנת בתור אוסף גדול של מאמרים. בעבר דובר רבות על האפשרות להשתמש

באנציקלופדיה להקניית ידע למחשבים, אך רעיון זה נתקל בקושי רב של הבנת השפה על ידי מחשב. בעבודה זו אנו מציעים לראשונה גישה שמאפשרת למחשב להשתמש באנציקלופדיה במישרין, ללא מודולים ייעודיים להבנת השפה וללא מאגר עזר של ידע כללי על העולם.

אנו מאמינים כי לשיטה המוצעת שימושים רבים בתחום עיבוד שפות טבעיות. בעבודה הזאת, בנוסף לסיווג טקסטים אנו מפעילים את שיטה גם בתחום נוסף, דהיינו, אומדן קרבה סמנטית בין קטעי טקסט. משימה זו נפוצה מאד בתחומים השונים של בלשנות חישובית, כגון התרת רב-משמעות של מילים, קיבוץ מילים וטקסטים לפי קרבה במשמעות, זיהוי שגיאות וכו'. בעבר, פתרונות לבעיה זו נעזרו במילונים ממוחשבים ובאנליזה סמנטית חבויה (Latent Semantic Analysis). כאן אנו מציעים גישה חדשה הנקראת אנליזה סמנטית מפורשת (Explicit Semantic Analysis), אשר משתמשת בבונה התכוניות ומייצגת משמעות של טקסטים במרחב הרב-מימדי של מושגים מהמדריך הפתוח ומויקיפדיה.

לעבודה הנוכחית מספר תרומות מדעיות. ראשית, אנו מציעים מתודולוגיה כל-לית ומספר אלגוריתמים לבניית תכוניות בעזרת מאגרי ידע מהגדולים בעולם. בניית תכוניות תוך שימוש במאגרים אלו משתמשת בידע אנושי רב, ומסוגלת לנצל מידע אשר אינו ניתן להסקה מהטקסט המסווג בלבד. כמו כן, אנו מציעים גישה חדשה לניתוח מבוסס הקשר, אשר מנתח טקסטים ברמות רזולוציה שונות ועל ידי כך מבצע התרת רב-משמעות של מילים. מימשנו את המתודולוגיה המוצעת עבור שני מאגרי ידע מסוימים - המדריך הפתוח וויקיפדיה, אשר הינם מאגרי הידע הגדולים ביותר מסוגם. שימוש בשיטה המוצעת שיפר בצורה ניכרת ביצועים של מערכות ממוחשבות בשני תחומים של בלשנות חישובית, דהיינו, סיווג טקסטים ואומדן קרבה סמנטית בין קטעי טקסט. בתחום סיווג טקסטים הצלחנו לפשר את דיוק הסיווג למרות שבשנים הקודמות כמעט ולא היה שיפור בביצועי המערכות הקיימות. כמו כן חשוב לציין, כי הגישה שלנו לאומדן קרבה סמנטית נותנת פתרון אחיד לחישוב הקרבה הן בין מילים

בודדות, והן בין קטעי טקסט ארוכים מאד.