

# **A Semantic Graph Model for Text Representation and Matching in Document Mining**

by

Khaled Shaban

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2006

©Khaled Shaban 2006

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

# Abstract

The explosive growth in the number of documents produced daily necessitates the development of effective alternatives to explore, analyze, and discover knowledge from documents. Document mining research work has emerged to devise automated means to discover and analyze useful information from documents. This work has been mainly concerned with constructing text representation models, developing distance measures to estimate similarities between documents, and utilizing that in mining processes such as document clustering, document classification, information retrieval, information filtering, and information extraction.

Conventional text representation methodologies consider documents as bags of words and ignore the meanings and ideas their authors want to convey. It is this deficiency that causes similarity measures to fail to perceive contextual similarity of text passages due to the variation of the words the passages contain, or at least perceive contextually dissimilar text passages as being similar because of the resemblance of words the passages have.

This thesis presents a new paradigm for mining documents by exploiting semantic information of their texts. A formal semantic representation of linguistic inputs is introduced and utilized to build a semantic representation scheme for documents. The representation scheme is constructed through accumulation of syntactic and semantic analysis outputs. A new distance measure is developed to determine the similarities between contents of documents. The measure is based on inexact matching of attributed trees. It involves the computation of all distinct similarity common sub-trees, and can be computed efficiently. It is believed that the proposed representation scheme along with the proposed similarity measure will enable more effective document mining processes.

The proposed techniques to mine documents were implemented as vital components in a mining system. A case study of semantic document clustering is presented to demonstrate the working and the efficacy of the framework. Experimental work is reported, and its results are presented and analyzed.

## Acknowledgements

All praise is due to Allah for granting me the strength and knowledge to complete this work. I would like to express my deep and sincere gratitude and appreciation to my advisors Dr. Otman Basir, and Dr. Mohamed Kamel. The opportunity to work with them has been tremendously rewarding – They are always willing to explore new frontiers and have a contagious enthusiasm for excellence. They gave me a great deal of freedom to choose my own course and yet were always there to pull me back when I wandered off on some unproductive track. Throughout my term as a graduate student, they have been a source of trusted advice. I would like to also thank members of my PhD committee for their valuable comments. Many thanks go to the PAMI lab members and colleagues. Last and by far not least, I am indebted to my parents, wife, brothers, and sisters for their continued support, patience, and prayers.

## Dedications

To the Most Merciful...

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ  
الرَّحْمَنُ (1) عَلَّمَ الْقُرْآنَ (2) خَلَقَ الْإِنْسَانَ (3) عَلَّمَهُ الْبَيَانَ (4). سورة الرحمن

In the name of Allah, Most Gracious, Most Merciful.

[1] (Allah) Most Gracious! [2] It is He Who has taught the Quran.

[3] He has created man: [4] He has taught him speech (and

Intelligence). *Quran* 55:1-4.

# Table of Contents

Abstract .....	iii
Dedications .....	v
Acknowledgements .....	vi
Table of Contents .....	vii
List of Figures .....	xii
List of Tables .....	xiv

## CHAPTER 1

Introduction .....	1
1.1 Preface .....	2
1.2 Concepts and Terminologies .....	7
1.2.1 Data Mining .....	7
1.2.2 Text .....	8
1.2.3 Free, Structured and Semi-Structured Text .....	8
1.2.4 Documents and Document Sets .....	9
1.2.5 Document Mining .....	10
1.2.6 Document Mining Processes .....	10
1.2.7 Natural Language Parsers .....	11
1.2.8 Semantic Understanding .....	12
1.2.9 Semantic Similarity Measure .....	14
1.2.10 Semantic-based Document Mining Systems .....	14

1.3 Motivations and Inspirations .....	15
1.4 Challenges .....	17
1.5 Goals .....	19
1.6 Opening of Further Research .....	20
1.7 Organization of the Thesis .....	21
1.8 Conclusion .....	22

## CHAPTER 2

Background and Literature Review .....	24
1.2 Document Mining Main Concerns.....	25
2.1.1 Text Representation Models .....	25
2.1.2 Similarity Measures .....	27
2.1.3 Mining Processes .....	28
2.1.3.1 Document Clustering .....	32
2.1.4 Evaluation Methods .....	37
2.2 Natural Language Understanding .....	39
2.2.1 Morphological Analysis .....	40
2.2.2 Syntactic Analysis .....	41
2.2.3 Semantic Analysis .....	41
2.2.3.1 Lexical Processing .....	42
2.2.3.2 Sentence-level Processing .....	43
2.2.3.3 Discourse Integration .....	44
2.2.3.4 Pragmatic Analysis .....	46
2.2.3.5 World Knowledge Representations .....	46

## CHAPTER 3

The Semantic-based Document Mining Framework: An Overview ..	48
3.1 Architectural Overview .....	49
3.2 Similarity Homomorphism Assumption .....	52
3.3 Semantic-based Text Representation .....	54
3.3.1 Parsing Sample Texts .....	56
3.4 Semantic-based Similarity Measure .....	60
3.4.1 Similarity Estimation: An Example .....	62
3.5 Mining Processes .....	63
3.6 Supplementary Components .....	64
3.7 Conclusion .....	65

## CHAPTER 4

Semantic Graph Model (SGM) A Meaning Representation of Text ..	67
4.1 Document Representation .....	68
4.2 The Semantic Graph Model (SGM) .....	69
4.3 Mapping Text to SGM .....	72
4.3.1 Text Analysis System Architecture .....	73
4.3.1.1 Tokenization .....	77
4.3.1.2 Part of Speech Tagging .....	78
4.3.1.3 Syntactic Parsing .....	78

4.3.1.4 Semantic and Discourse Analysis .....	80
4.4 SGM Desiderata .....	87
4.5 Conclusion .....	89

## CHAPTER 5

Similarity Measure .....	91
5.1 Graph Matching .....	92
5.2 Notations and Terminologies .....	94
5.3 The New Similarity Measure .....	98
5.4 Studying the Metricity of the Measure .....	103
5.5 The Algorithm for Computing the Similarity Measure .....	105
5.6 Measuring Similarity of Two Documents .....	108
5.7 Conclusions .....	110

## CHAPTER 6

Experimental Work: Semantic Document Clustering .....	112
6.1 Document Clustering .....	113
6.2 Experimental Setups .....	114
6.2.1 Text Parser .....	114
6.2.2 Data Sets .....	116
6.2.3 Text Representations .....	117
6.2.4 Similarity Measures .....	119

6.2.5 Clustering Algorithms .....	120
6.2.6 Evaluations .....	120
6.3 Collected Results .....	120
6.4 Analysis of Results .....	124
6.5 Conclusions .....	126

## CHAPTER 7

Summary and Conclusions .....	128
7.1 The Proposed Approach .....	129
7.2 Findings .....	133
7.3 Contributions .....	134
7.4 Future Extensions .....	137
Bibliography .....	141

## List of Tables

Table 6.1 Description of Data Sets Used in the Experiments .....	117
Table 6.1 Experimental Results – Semantic-based - Option 1 .....	121
Table 6.2 Experimental Results – Semantic-based - Option 2 .....	122
Table 6.3 Experimental Results – Semantic-based - Option 3 .....	122
Table 6.3 Experimental Results – Semantic-based - Option 3 .....	122
Table 6.4 Experimental Results – Semantic-based - Option 4 .....	122
Table 6.5 Experimental Results – Semantic-based - Option 5 .....	123
Table 6.6 Experimental Results – Vector Space Model .....	123

## List of Figures

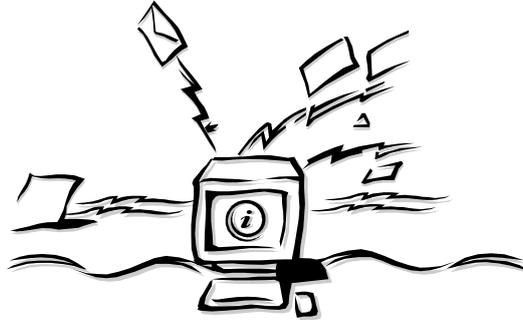
Figure 1.1 (a) Internet Servers Growth Rates .....	5
Figure 1.1 (b) Various Statistical of Internet Growth and Use .....	5
Figure 1.2 Semantic Understanding-based Approach for Document Mining .....	19
Figure 2.1 The Vector Space Model .....	26
Figure 2.2 Information Retrieval .....	30
Figure 2.3 Information Filtering .....	30
Figure 2.4 Information Extraction .....	31
Figure 2.5 Document Classification .....	31
Figure 2.6 Document Clustering .....	32
Figure 2.7 Information Flow in Text Natural Language Understanding Systems	40
Figure 3.1 The Semantic Understanding Approach .....	49
Figure 3.2 An Overall System Architecture .....	51
Figure 3.3 Similarity Homomorphism Assumption .....	53
Figure 3.4 An Example of a Canonical Representation .....	56
Figure 3.5 (a) Parse Trees of Sample Text .....	57
Figure 3.5 (b) Parse Trees of Sample Text .....	57
Figure 3.6 (a) SGM Knowledge Representations of Sample Texts .....	58
Figure 3.6 (b) SGM Knowledge Representations of Sample Texts .....	58
Figure 3.6 (c) SGM Knowledge Representations of Sample Texts .....	59
Figure 3.7 SGM Skeleton for a Document .....	60
Figure 3.8 (a) Similarity Estimation for Sample Texts .....	63
Figure 3.8 (b) Similarity Estimation for Sample Texts .....	63
Figure 4.1 SGM an Example Sentence .....	70

Figure 4.2 Flow Diagram of SGM Mapping Processing Components .....	71
Figure 4.3 The Multi-pass Parser Architecture .....	74
Figure 5.1 Miniature Illustration Trees .....	100
Figure 5.2 Common Similarity Sub-trees .....	101
Figure 5.3 The Algorithm for Computing the Similarity Measure .....	107
Figure 5.4. (a) Two Similar Documents Composed of Different Words .....	109
Figure 5.4. (b) Two Different Documents Contain Similar Words .....	110
Figure 6.1 (a) A Snapshot of the Syntactic Parse Tree .....	115
Figure 6.1 (b) A Snapshot of The SGM Tree .....	116
Figure 6.1 Clustering Results for Data-Set-1 .....	126
Figure 6.2 Clustering Results for Data-Set-2 .....	126

---

# CHAPTER 1

## Introduction



*"Where is the wisdom we have lost in knowledge?"*

*Where is the knowledge we have lost in information?"*

T. S. Eliot (1888 - 1965), *The Rock*.

---

This chapter presents a view of the research work and the problems it addresses. Section 1.1 gives an introduction to the information overload problem and explains how the document mining field has emerged to address this problem. Some important concepts and terminologies are defined in Section 1.2. The chapter summarizes the adopted research approach and contrasts it with current methodologies. Some existing problems in current paradigms are also stated. Section 1.3 discusses motivations and inspirations to pursue the semantic understanding-based framework in document mining. Section 1.4 mentions challenges facing the approach, and Section 1.5 pinpoints specific goals to be accomplished. Further research avenues the work is expected to open are discussed in Section 1.6. Section

1.7 gives the overall structure of the thesis and provides a short overview of each of its seven chapters. The chapter ends with conclusions in Section 1.8.

---

## 1.1 Preface

As the sheer number of documents that are available online increases exponentially, the need to manage these documents also grows. The result of this exponential growth is what has become known as the *information overload* problem. Taking into consideration only the volume of information available via the Internet and the World Wide Web (WWW) presents a non-trivial real problem. The extent of this problem is apparent to anyone who has tapped into the WWW, and attempted to locate specific desired information. Moreover, the acceleration of information change and availability can lead to psychological, physical and social problems, especially to the *knowledge workers* whose jobs mainly involve dealing with and processing information. In a world-wide survey conducted by Reuters News Agency [79], it was found that two thirds of managers suffered from increased tension and one third from ill-health because of information overload. It was also concluded that other effects of too much information can cause anxiety, poor decision-making, difficulties in memorizing and remembering, and reduced attention span [62].

Solving the information overload (or *overkill*) problem involves processes such as *information gathering*, *information filtering*, *information retrieval*, *information extraction*, *document classification*, *document clustering*, and *document summarization*. The goal of these processes is to help users to have better access to documents that satisfy their information needs. The needs can be defined to discover or derive new information, to find patterns across documents, and to separate the

desired information from the noise. These computational processes constitute cornerstone tasks in the ever-developing study field of *document mining*.

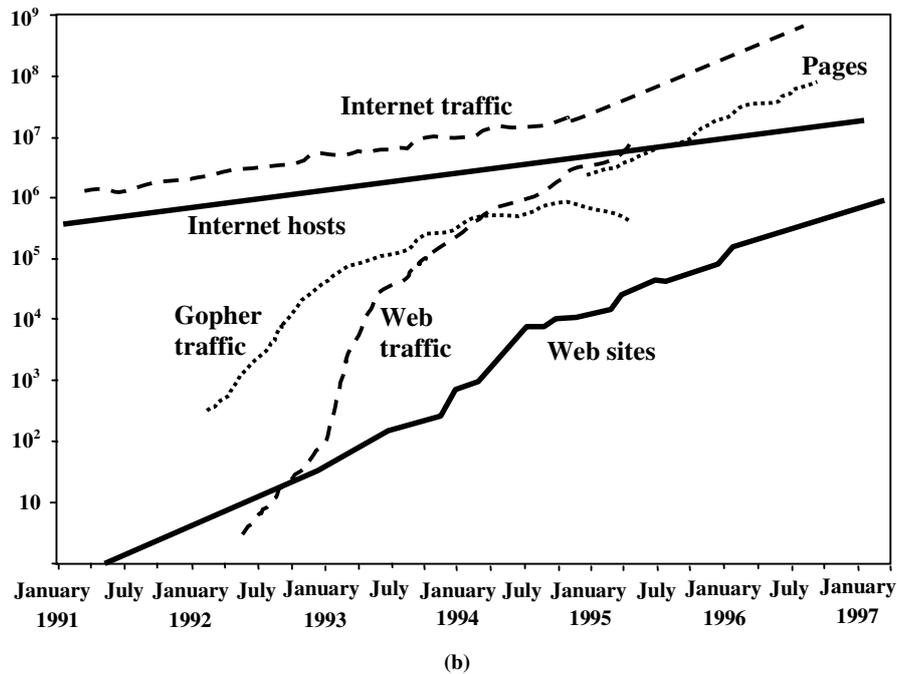
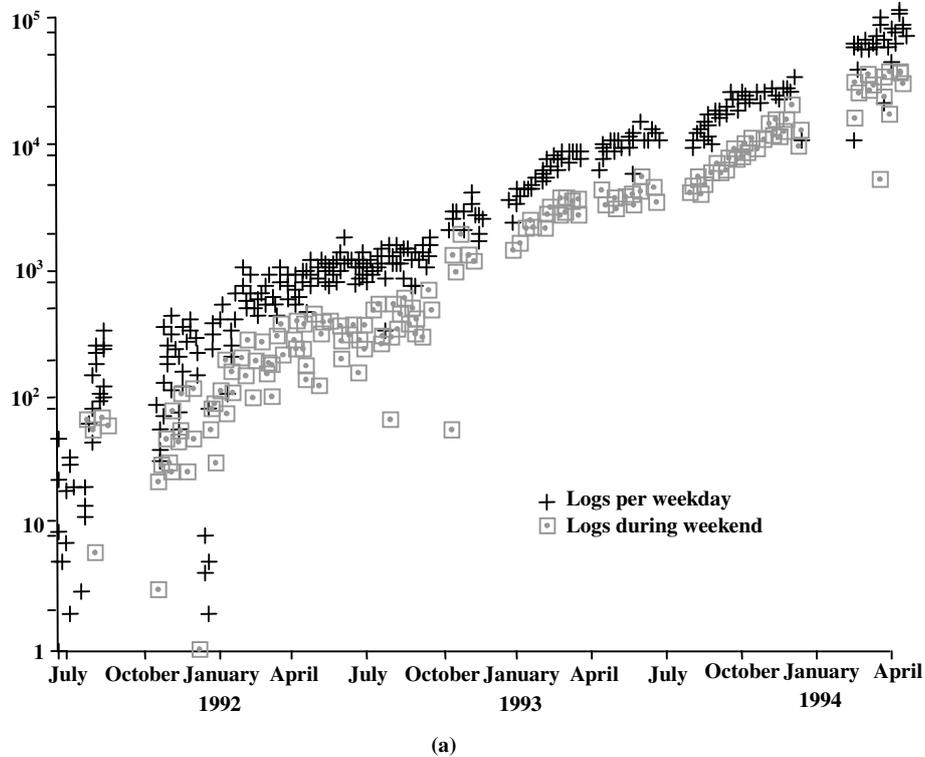
Much of research work has been focused on such document mining aspects for the last few decades [85]. In fact, the concern of mining documents can be traced back in history to as early as the era of the 3000BC Egyptians who invented papyrus scroll ('paper' made from papyrus plants), and then later used by ancient Greeks and Romans to store information in a written form and to easily retrieve it. *Table of Contents, Alphabetization, and Hierarchies of Information* are techniques the Greeks and Romans invented and employed in writing ancient work [89]. Later on, in 105AD, the Chinese invented the modern method of papermaking from silk. The invention spread to the Middle East, where the first paper-manufacturing plant in the Islamic civilization was opened in Baghdad, in 794AD that used cotton rags, and replaced the use of papyrus and parchment (skin of animals). Henceforth, millions upon millions of books were published wherever this invention arrived [3]. The development of paper, mass-writing, *translation, summarization, and indexing* made knowledge and learning easier, as more people were able to access to it.

Throughout modern history, there have been numerous approaches to storing, organizing, analyzing, retrieving, and discovering information from documents. These approaches range from simple statistical approaches to fairly sophisticated content analysis approaches [1]. The former is based on simple exhaustive processing by accumulating statistical details about document contents, and can be done entirely mechanically with probabilistic techniques; the latter depends mainly on intellectual analysis, both by people and/or by machines equipped with artificial intelligence (AI) programs. There have been also hybrid approaches that combine both the statistical and the analytical processing of document contents.

A considerable number of inventions in document mining have been attained throughout the years since the modern information technology (IT) revolution began in the 1950s. First, there was, for example, the use of different indices, concordances, and overlapping codes on edge-notched cards. Then digital systems came to play, and large-scale information systems were built. The development of computer typesetting, word processing, time-sharing technologies, and reliable (yet cheap) storage devices made these mature systems to become more practical, and hence commonly used. Consequently, major experimentations and evaluations of these systems were carried out. In terms of progress in document mining research, there has been a rise and refinement of probabilistic techniques. In the task of information retrieval for instance, this involved measuring the frequency of words in relevant and irrelevant documents, and using term frequency measures to adjust the weight given to different words [98]. On the other hand, there have been AI researchers who tried to do intellectual analysis automatically [75].

Since early 1990s, however, we started to witness an information explosion as a result of the emergence of yet another technological revolution - the Internet. As the number of users exponentially increases (see Figure 1.1), eventually almost everyone will be on the net [73]. The Internet has become a standard medium for publishing after the development of the Web and its different browsers [1]. What is remarkable is not that everyone is accessing information, but the fact that everyone is able to provide information and it is happening entirely on a free basis, without much of industrial support. Admittedly, much of the available information is of low quality, but that does not seem to make it less attractive. Moreover, the availability of multimedia materials (such as, images, graphics, audios, and videos) has provided

attractiveness, and thus a growth rate that none of any other information systems were able to reach.



**Figure 1.1** (a) Internet Servers Growth Rates (the number of servers grows from a few hundred to a million between 1991 and 1997) (b) Various Statistical of Internet Growth and Use. (adopted from [74])

This massive volume of available knowledge ought to be discovered and the tasks of managing, analyzing, searching, filtering, and summarizing information in documents should be automated. Existing document mining systems have shown some limitations in delivering meaningful output, especially when dealing with human languages. The gathering, organization, and handling processes in these systems are focused on the low information (or even data) representations instead of the higher knowledge levels, and that is considered a strong factor that leads to the impreciseness of these systems output. There is a lack in understanding of documents contents, and instead there is a focus on the presence or absence of keywords to mine (i.e., retrieve, filter, extract, classify, or cluster) texts. This simple word counting and frequency distributions do not always capture the meaning behind the words, limiting the ability to distinguish between texts. Breaking through this limitation will require document mining systems to understand the texts they process, and pursue the mining tasks according to the understanding scheme.

The main objective of this work is to advance the state-of-the-art of document mining by introducing a framework for document mining systems that is based on exploiting the semantic information in text. The framework is mainly composed of text analysis and similarity measure components that are utilized to improve mining processes performance. The research work is pursued in three areas; modeling, development and implementation, and experimentation and testing. The underlying theoretical work is centered on introducing semantic notions to represent document contents, and developing similarity measurements to estimate distances between text documents. Developing a system in accordance to the proposed approach requires an excessive use of existing analysis techniques, and the implementation of new components. To validate the proposed concepts, document mining processes are

explored and carried out by the developed models and theoretical schemes. Before elaborating more on the research work and its goals, some terms and concepts are defined in the following section.

---

## **1.2 Concepts and Terminologies**

Below are definitions of concepts and terminologies that characterize the work and give an overall understanding of the research directions.

### **1.2.1 Data Mining**

Data mining (DM) is an information extraction activity to discover hidden, implicit, previously unknown, and potentially useful facts contained in large databases or data warehouses [33]. This is often accomplished using a combination of machine learning, statistical analysis, modeling techniques, and database technology. The goal of DM is to find global patterns, search for relationships in data, and infer rules that allow prediction of future events. Typical DM applications include marketing, profiling, security, and risk analysis. For example, for better supply strategies, a retail business may mine its customers' data and find that people who buy a certain product would also buy another product at the same time. Moreover, mining a medical database may produce new sightings on the outcomes of different treatments or the effects of treatments on different ethnical groups, genders, or population. Another application is in the detection of fraud and crimes, such as credit card scams, where a look across the credit card records can reveal deviations from the normal consumer spending patterns.

### 1.2.2 Text

In this thesis, the terms *message*, *narrative*, *post*, *script*, *story*, *text* and *wire* shall all refer to the input that is stored in a machine readable format (i.e., data consisting of a sequence of characters, as opposed to binary numbers, images, graphics commands, executable programs, and the like). It needs not to be annotated nor structured, yet it should follow some rules of a natural language, such as, Arabic, English, and French (i.e., data consisting of written human language, as opposed to programs, or following the stylistic conventions of human language). It is this kind of format that this work is concerned with the most.

### 1.2.3 Free, Structured and Semi-Structured Text

Text can be categorized into three types: free text, structured text, and semi-structured text. These types are defined as follows:

**a) Free Text** is any short or long natural language (NL) text that adheres to the grammatical rules of the NL and holds semantically valid thoughts. Examples of this type of text are news headlines, articles, research abstracts, and patents. The crucial feature of a free text is that its elements can be arranged in a fixed sequence, so that the phenomena of placement or displacement are relevant, i.e., altering a message component would result in varying its meaning. Free texts may have sections, named sections, paragraphs, and titles.

**b) Structured Text** is defined as textual information stored in a database or a file following a predefined and rigid format. This collection of data is organized so that its contents can easily be accessed, managed and updated automatically. The most prevalent type of database is the relational database; a tabular database in which data is defined so that it can be reorganized and accessed in a number of different ways.

Traditional databases are designed to handle structured data. They require users to use a pre-defined structure and force them to custom their information into this format.

c) **Semi-structured Text** falls between unstructured collections of textual data and fully structured tables of typed data. Semi-structured text is often ungrammatical, telegraphic in style, and does not follow any strict format. Nevertheless, some sort of structuring can be present in what is considered semi-structured text such as HTML-tagged documents.

#### **1.2.4 Documents and Document Sets**

Documents are machine readable files that contain text, and possibly other data formats, such as tables, images, graphics, audios, and videos. Depending on the information domain, a document can refer to anything such as intuitive notions like electronic newspaper articles, encyclopedia entries, or books. Documents can contain logically smaller subunits such as chapters, parts, and sections. In the Web-based applications, for instance, it can refer to a Web page, a part of a page, or to an entire Website. A *document collection* (a *document set*, or a *corpus*) refers to a group of documents being used in a specific task, or a set of documents that are logically related, usually by their contents, target audience, or origin (e.g., a collection of studies produced by a program, project, or an organization). A publication volume is a good example of a document set. In addition, The World Wide Web (WWW) presentation of a document set can consist of several WWW pages.

In many investigations, there has been the use of benchmark data sets such as the Reuters-21578 (and Reuters-22173). This data set consists of a large number of news stories filed and categorized by the Reuters news network during 1987. The topics were marked by professionals to categorize these news stories. RCV1 (Reuters

Corpus Volume 1) is a recently released collection of news stories. It is a large, and a high quality data collection that is expected to become the new standard benchmark [60].

### **1.2.5 Document Mining**

Document mining can be broadly defined as the automated task of discovery and analysis of useful information from documents. Document mining systems encompass processes to solve the information overload problem mentioned above. Examples of these processes are document clustering, and document classification where the goal is to group documents that are similar in meaning or topology. Thus, document mining is fundamentally based on analysing a semantically rich document or a set of documents, understanding their contents and meaning and satisfying some user requirements. It is quite different from regular data mining in the sense that patterns are extracted mainly from natural language text, rather than from structured databases of facts. Databases are designed for systems to process automatically, but text is written by humans for people to read. The research work in this area is rooted heavily in the study fields of artificial intelligent (AI), information retrieval (IR), machine learning (ML), pattern recognition (PR), natural language processing (NLP), natural language understanding (NLU), computational linguistic (CL), and data mining (DM).

### **1.2.6 Document Mining Processes**

Document mining processes are activities conducted by humans, perhaps with the assistance of a machine, to discover and extract useful information from documents. Examples of these processes and their goals are:

- Document clustering: to group documents into topical clusters.
- Document classification: to classify documents into predefined categories.
- Information retrieval (IR): to find and rank relevant documents in response to users' queries.
- Information filtering (IF): to sort through documents and present to users those which are likely to satisfy their information requirements.
- Information extraction (IE): to extract specific data elements from free text and integrate them with structured text.
- Document summarization: to condense the most important information in documents and produce summarized versions for particular user(s) or task(s).

For these processes to perform their expected tasks they often pre-process the documents in order to present them in some tangible forms that enable further processing such as similarity matching.

### **1.2.7 Natural Language Parsers**

A parser is a specialized software program that analyzes textual natural language input (a sentence or more) and converts it to a formal representation that can contain syntactic and/or semantic information not explicitly present in the sentence(s). Parsers can be categorised into three types; morphological, syntactic, and semantic parsers.

A morphological parser takes word forms as input, analyses them, and returns their morpheme structures indicating the different morphemes that constitute the input and how they are related to each other. Morphemes are small meaning-holding units of words. These morphemes can be broadly classified into stems (words roots) and

affixes, which may be prefixes, suffixes, infixes or circumfixes. Prefixes are those morphemes, which may appear before a stem, and postfixes are those that are applied to the end of the stem, circumfixes are those morphemes that can be applied on both sides of the stem. The morphemes categorized under infixes are those that appear inside a stem.

When the parsing program is used to determine a syntactic structure of a sentence according to some language formal grammar, it is called a syntactic (or syntax) parser. The syntactic parser is often provided as input with a language grammar, a lexicon, and a word-string, and will output, if the string is a well-formed sentence, a structural description of it. The structural description is often represented as a parse-tree, also called a phrase-marker, a diagrammatic representation of the sentence's constituent grammatical structure. If the sentence is ill-structured, the parser will reject the string.

Semantic parsers attempt to model the meanings by defining relations between text constituents of phrases, words, and morphemes. Typically, a semantic parser transforms some input text into a data structure that can be processed easily, e.g., for semantic checking, comparison of meanings, knowledge inference, or to ease the further understanding processing of the input. Such a data structure usually captures the semantic relations between concepts of the input and forms a tree or even a full graph.

### **1.2.8 Semantic Understanding**

*"In the main, semantics (from the Greek semantikos, or "significant meaning," derived from sema, sign) is the study of meaning, in some sense of that term. It should not be confused with the general semantics of Alfred Korzybski, a somewhat different*

*discipline. Semantics is often opposed to syntax, in which case the former pertains to what something means while the latter pertains to the formal structure/patterns in which something is expressed (for example written or spoken).*" [Wikipedia encyclopaedia]

Throughout this thesis, the phrase 'semantic understanding' means that a semantic natural language parser generates a sketchy pattern that correctly portrays the formal meaning of the original sentence at a coarse level of details. Hence, in some sentences, some linguistic elements may be lost during the parsing process, but the general meaning is conveyed, and the sentence will have been said to be 'correctly understood'. The process whereby such representations are created and assigned to linguistic inputs is called *semantic analysis*. Initially, the perspective semantic structure should hold the meaning of words through defining atoms or primitives, and relations between them. In addition, this semantic analysis could include determining thematic roles, argument structures, and their linking to syntax. Moreover, the semantic structure could also represent senses, references, certainty conditions, discourse analysis, and pragmatics.

It is worth mentioning that the formal meaning representations derived from linguistics, and processing of natural languages, are different from what philosophically could be considered as meanings behind text. The focus is to get an appropriate, unambiguous, and operational representation from linguistic inputs. Philosophers such as Lewis [61] could see that translating sentences from their original natural form to another artificial structure does not help in getting closer to their meanings nor their truth conditions. It is asserted that these formal representations can facilitate real semantic work. Alone, the representations are not of much interest. That is to say, the importance and the focus of the work are in

developing the functions, or procedures, that determine the mapping from the linguistic inputs to their formal representations, and in the use of these representations in assisting further processing, namely document mining.

### **1.2.9 Semantic Similarity Measure**

Intuitively, the similarity between two entities is a proportional relation to their commonality, and counter proportional to their differences. Finding distances between documents is an opposite expression of measuring similarity. When the process of estimating similarities between documents is defined on the meaning representations of the documents contents, the process is called a semantic similarity measure. This process mimics the intelligent human-like similarity estimation between texts, as the similarity estimator accepts two understood (semantically parsed) texts and determines the semantic distance between them. Algorithmically, there should be a search through the semantic representations of both documents to estimate their levels of commonalities and differences. The measure should preferably be metric, i.e., establish an order over the documents in the document set, to facilitate processes such as comparing, searching, and indexing. Moreover, it should be computationally tractable and can be applied effectively and efficiently.

### **1.2.10 Semantic-based Document Mining Systems**

Semantic-based document mining systems encompass a wide range of components related to storing, organizing, and mining of documents. The focus in these systems is on the analysis of contents, as opposed to the bag-of-word approaches and its subsequent results in response to the mining processes requirements. Of particular interest is the adoption of meaning-based representations

of text, and their use in measuring similarities (or distances) between documents. In these systems, the meaning of documents resides in the structure, constituency, and the reasoning about words/phrases semantics. Similarity measures are defined on these representations to yield meaningful distances assessments. Mining processes, such as document clustering, document classification, and information retrieval, that make use of some or all of the mentioned components are forming what is called the semantic-based document mining processes.

This framework of systems is expected to meet document mining requirements and output more meaningful results than what could be accomplished otherwise.

---

### **1.3 Motivations and Inspirations**

Most conventional approaches to mine documents are based on whether or not documents contain specific keywords, and on statistical information about words appearances and co-occurrence frequencies. As an example, one of the current questions in document mining is which documents belong to which group of documents. That is document clustering to create thematic overviews of document collections. There has been notable success in looking at which words co-occur in articles in order to predict such group belongings. This method can produce good results, even though the meaning of the texts is not being discerned. Rather, the text is treated like a "bag of words". Another example is the extraction of the most frequent words and phrases from a document that, when shown to a human reader, seems to summarize its contents. Moreover, it is relatively easy to extract information from text with somewhat regularized structure, such as reading resumes and extracting people's names, addresses, skills, and so on. However, following this

approach, the mining can only be done on known text. For example, in information retrieval, the user is typically looking for something that he knows and has been previously written by someone else. Generally, considering only words and ignoring meanings of documents contents leads to the deficiency in distinguishing between text passages. This is because they could be similar in meaning and composed from different words, or dissimilar and have same set of words.

There is much room for improvement when one utilizes meanings of the documents contents. The main motivation of this work is the advancement in mining documents after analysing the text semantically rather than considering them as bags-of-words. Humans' ability to discover knowledge from text and to determine similarities between documents are the main inspirations to pursue a semantic-based approach to mine documents. Humans are considered to be the best document analyzers. Their ability lies on the capability to understand the documents contents, and judge similarities based on the understanding. For instance, even with partial understanding, a person can make rough judgments about where to group (classify or assign to a cluster) a document at hand. Similarly, document mining schemes can differ in performance as how much they attempt to understand contents, and how they employ the understanding in finding similarities between documents. Practical trials (as will be discussed later) show clearly that when semantic information clues are augmented in the processing steps, mining results are improved. This gives more motivations to exploit research findings in human-like intelligent text understanding and apply them to mine documents for better accessing, searching, retrieving, organizing, managing, and reasoning about information they contain.

---

## 1.4 Challenges

The potential for mining text collections through semantic understanding are almost unexploited, and there has been little work to date in this direction (See Chapter 2 for details). The reason, perhaps, is that text expresses a vast, rich range of information, and encodes this information in a form that is difficult to interpret automatically. More precisely, the difficulties in choosing the semantic understanding approach to mine documents are tied to the questionable possibility of natural language processing by machines. Many claims have been made about the ability of automating the process of understanding languages, especially after seeing new products such as handwriting, and voice recognition software becoming widely available. The hype increases when some are predicting that within a few years, machines will understand us on our terms; through natural language. Such claims are however far from reality for a few reasons.

A key reason for NLP hardness is the ambiguity that exists in all levels of analysis (Chapter 2 addresses these levels and problems in more details). For example, when humans process language, they continuously make guesses about meanings using a rich knowledge of the world and of the current culture. This knowledge of contexts is assumed to be known to all communicators. Although it is not stated explicitly, it may comprise up to 90% of communications [104]. When a person asks, for instance, "*is there water in the fridge?*" most humans would consider if there was something like a container of water in the fridge. The thought about water molecules in fruits and vegetables would be considered strange and outside the normal cultural use of the question. This ambiguity extends to include cases in the syntactic, morphological, and pragmatic levels of text analysis. Syntactic ambiguity can be found in sentences which can be parsed in more than one way, as in the case of

one word that may have two parts of speech or homonyms as in *"I am going to eat."*, where *"going"* can be a verb with destination *"eat"* or an auxiliary indicating near future. A typical example of the pragmatic ambiguity in NL is the phrase *"Time flies like an arrow."* Although humans unambiguously understand it to mean *"Time passes in a similar speed that an arrow flies,"* it could also mean *"Measure the speed of flies as you would for an arrow"*, *"Measure the speed of flies as an arrow would,"* or even *"A kind of fly, the time fly, likes arrows."* Moreover, the great flexibility and the dynamic characteristics of natural languages make it harder to keep-up with the varieties and new ways of expressing an evolving world. New words and expressions are constantly made such as *"ecotourist"*, *"lol"*, *"to message"*, *"to page"*, and *"to google"*, to mention a few. Stating simple facts and events can be done in many ways as in *"John succeeds Mark as chairperson of Electro Systems"*, *"Electro Systems named John as its new chair-person after Mark"*, and *"Mark was succeeded by John as chair-person of Electro Systems"*.

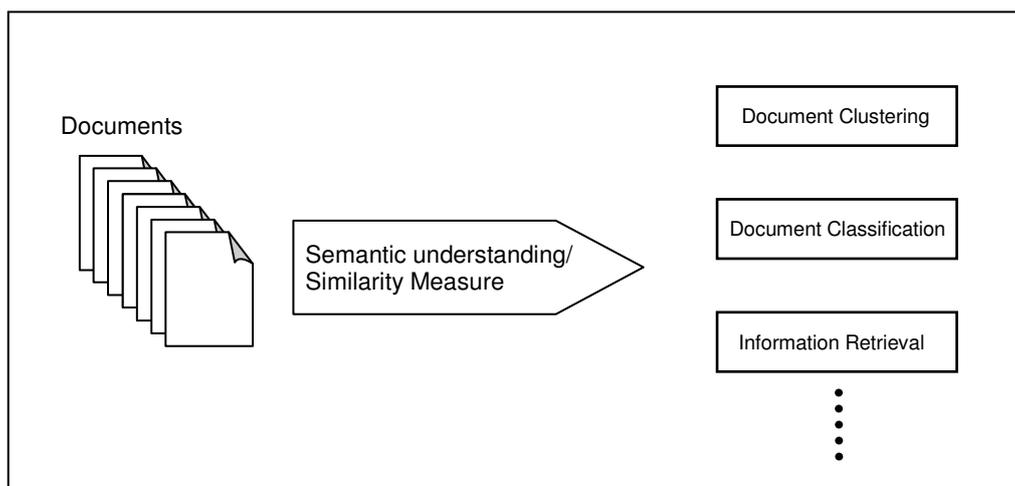
Though the possibility of having full-fledged natural language understanding systems in the near future seems unlikely, and meaning representations can be currently limited, the fields of natural language processing (NLP) and computational linguistics (CL) have a plenty of options to offer. There exist a number of mature and reasonable approaches to understand text that can be utilized and built upon. It seems more likely that the use of NLP and CL is the only choice if meaningful and high quality document mining results will ever to be achieved. By providing mining processes that can deal with rich representations of documents, even through rough and partial understanding, the results are expected to surpass existing document mining approaches. In fact, some of document mining tasks, such as clustering, and classification do not require more than a coarse grasp of the meanings to produce

good results. Moreover, the performance of the semantic understanding-based document mining systems will proportionally improve as the underlying technologies of NLP and CL progress.

---

## 1.5 Goals

The central goal of this research work is to introduce, build, and demonstrate a novel framework for document mining systems that are based on semantic understanding of the documents contents (See Figure 1.2). These systems will be composed of components that facilitate semantic analysis of text in the documents, measuring similarity between documents, and applying the different mining processes. The aim is to show the working of the method and how these systems could provide better performance than it is possible otherwise. Performance improvements including producing high quality mining processes outputs in an efficient time are required. This major goal can be subdivided into the following distinct objectives that are to be accomplished in their order:



**Figure 1.2** Semantic Understanding-based Approach for Document Mining

### **a) Developing Mining Systems Based on the Proposed Framework**

The most basic objective is to develop a document mining system that uses semantic information as the basis of its underlying mechanisms. The main components of this system to be formulated include, semantic representations of text in the documents, similarity measures for these representations, then apply the algorithmic steps of the different mining processes that use both of these features, and evaluation methods to figure the approach merits.

### **b) Demonstrate a Working System on Real Texts**

The developed system has to demonstrate complete document mining tasks on real texts. One of the goals in this succinct statement is that the system must undergo and pass the minimum accepted performance of the implemented processes according to their evaluation criteria. The outcome of the system in this case will play as a proof-of-concept of the developed new framework. Thus, the system is expected to be a prototype that can perform on small to medium-scale document sets.

### **c) Provide Better Performance than Existing Approaches**

The ultimate objective of this work is to have the semantic understanding approach provide a superior mining performance; ideally resulting in a substantial increase in all performance evaluation indices. The system should show maturity and be able to perform the different mining tasks on large-scale document collections. In addition, the system should be efficient and its time and space complexities are expected to be acceptable.

## **1.6 Opening of Further Research**

The adopted semantic approach could be extended and can open new avenues of exploration. In the different development stages of the semantic-based document mining, methodologies can be refined, and advanced. These stages include document representation models, similarity measurement methods, and mining processes application. In the semantic-based data representation stage, there can be many schemes to consider. Identifying advantages and disadvantages of each scheme for mining is a key issue. Furthermore, the idea of considering multiple representations and fusing them or selecting a particular one is worth exploring. Distance measuring techniques between texts represented with specific or fused models could also be investigated and would definitely lead to more interesting findings. The many mining processes can serve as test-beds for the theories developed, and in themselves many avenues could be explored for improvements.

In addition, other documents contents such as multimedia materials are interesting to explore and include in the system. Semantic representation of such contents would further improve the understanding aspect, and enhance the mining results. Considering structured or semi-structured text, where text locations, document layout, etc. adds to the overall understanding of documents contents, and thus is another possible direction for investigation.

---

## **1.7 Organization of the Thesis**

Chapter one presents background materials, defines terms, and discusses some challenges that the work faces. It also states the objectives and goals of the research work. Chapter two reviews relevant literature and provides an introduction to

document mining, and natural language processing and understanding research fields. Chapter three presents an overview of the proposed framework of a semantic-based document mining system including its text representation model, similarity measure, and other supplementary components and their applicability to the different mining processes. Details of the developed representation scheme, and the similarity measure are presented in Chapters four, and five, respectively. The sixth chapter includes a case study of a document mining process, namely, the semantic document clustering. Implementation details and experimental results are discussed and analysed in this chapter. Finally, a summary of the work, and a discussion of the research contributions, findings, and recommendations for future expansions are given in the final chapter; Chapter seven.

---

## **1.8 Conclusion**

Solving the information overload problem requires the development of effective ways to mine available information in documents. A large portion of all existing documents contents are unstructured texts. Books, magazine articles, research papers, product manuals, news feeds, memos, e-mails, and the Web, all contain textual information written in a natural language. Thus, analyzing such information can provide means for automated intelligent mining processing that requires human attention daily. In this first chapter an overview of the research work and the problems it attempts to solve is presented. Inspired and motivated by the ability and potential of processing natural language, a new framework for document mining based on the semantic understanding of text is being proposed. The implementation of this new approach brings us closer to the fulfillment of having

intelligent and more effective knowledge discovery systems. The basic principles of the new approach are outlined, and some of its capabilities, requirements, and challenges, are discussed. It is mainly based on distilling the meaning of text, presenting it in a modular way to facilitate similarity assessments and applying mining processes. The application of this approach can help better navigate, cluster, and carry out semantic information retrieval on a collection.

---

## CHAPTER 2

### Background and Literature Review



*"Don't reinvent the wheel, just realign it."*

Anthony J. D'Angelo, *The College Blue Book*.

---

In this chapter a review is given on various topics that are deemed relevant to the research work. In particular, those that are related to document mining and natural language processing are discussed. Section 2.1 focuses on four basic document mining aspects; (1) text representation models, (2) similarity measurements, (3) document mining processes, and (4) evaluation techniques as found in the literature. Current approaches to carry out these tasks are reviewed, some of their drawbacks are pointed, and alternatives are briefly suggested. Section 2.2 discusses the natural language processing and understanding research fields as they relate to content-based document analysis. The discussion includes morphological, syntactic, and semantic analysis as practiced in these fields.

---

## 2.1 Document Mining Main Concerns

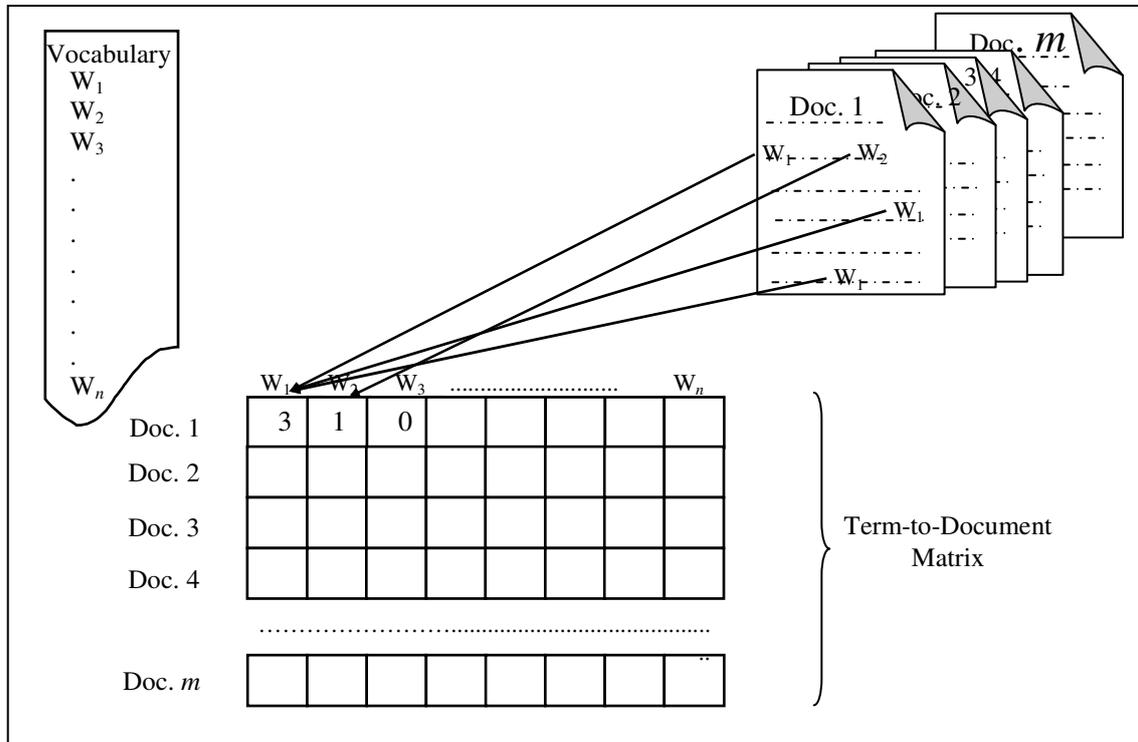
There are four main aspects that pertain to most of document mining approaches: (a) Representation models. (b) Similarity measures. (c) Mining processes. (d) Evaluation methods.

Having an appropriate data representation model is a fundamental activity in data mining. These models dictate how data objects (documents in our case) are thought of, and what features are cared about the most. Similarity measures are used to determine the distances between the objects in the representation space. Mining processes are the algorithms that describe the steps of a specific mining task in order to fulfill some requirements. Finally, the evaluation methods are used to judge the quality of the produced mining processes results. Each of these aspects are reviewed and discussed separately in the following subsections:

### 2.1.1 Text Representation Models

Conventional text representation models focus on whether a document contains specific keywords, or their appearance frequencies. For example, in the vector space model (VSM) [1, 9, 81, 82], documents are represented by vectors containing the frequency of all possible words (*features*) in a document set (see Figure 2.1). Since many words rarely occur in a particular document, many of these features will have zero or low frequencies. Therefore, features are selected to represent documents according to their importance as dictated by criteria such as *Document Frequency-Inverse Document Frequency*, *Information Gain*, *Mutual Information*, a  $\chi^2$ -test, and *Term Strength* [1, 105]. Moreover, before applying feature selection, a common practice is to reprocess text by removing stop-words and applying word-stemming algorithms. Stop-words, such as *the*, *and*, and *a*, are believed to have no significance

in capturing meaningful information. Word-stemming algorithms convert different word forms into a similar canonical form. Two popular stemming algorithms are used; the Porter stemmer [77], and using a lexicon dictionary lookup, such as WordNet [71].



**Figure 2.1** The Vector Space Model

Despite the widespread use of these word-based approaches to represent documents, it is believed that these approaches contribute to the lack of reliable performance of document mining systems. These approaches consider the document as a bag of words, and ignore meanings and ideas the document author wants to convey. It is this deficiency that can cause the similarity measures to either fail to perceive contextual similarity of text passages due to the variation of words the passages contain, or can perceive contextually dissimilar text passages as being similar because of the resemblance of words the passages have. A typical illustration of this deficiency that can cause the failure is the two sentences "John eats the apple

*standing beside the tree*" and *"The apple tree stands beside John's house"*, where despite using the same words, their meanings are different. Moreover, the following two sentences have more or less the same meaning but have been constructed from different sets of words, *"John is an intelligent boy"*, and *"John is a brilliant lad"*.

There has been a number of attempts to improve text representation, beyond what is possible by using keywords alone, including the use of N-grams [90], the collecting of bigrams [66], the extraction of words semantic relations through corpus statistics (e.g., Latent Semantic Indexing [25, 40, 64]), the use of background knowledge by replacing words with their higher concepts in an ontology [37, 47], and the consideration of word sequences [38].

### 2.1.2 Similarity Measures

Similarity measures are used to determine distances between documents, after transforming the textual data into a useable and intelligible format. There are various techniques to measure similarities and they all rely on the chosen model to represent the text. In VSM, for instance, the feature space constitutes a geometric space where documents represented as points in a multidimensional space. Thus, measuring the similarity can be easily calculated. Two measures are often used; they are the *cosine measure*, and the *Jaccard measure* [23, 88]. The cosine measure is defined as:

$$\cos(x,y) = \frac{x \cdot y}{|x| |y|},$$

where  $(x \cdot y)$  denotes the vector dot product of  $x$  and  $y$ , and  $|x|$  and  $|y|$

are the lengths of vectors  $x$ ,  $y$ , respectively. The cosine measure gives high similarity values to documents that share the same set of words with high term frequencies, and lower values to those that do not. The Jaccard measure is defined as:  $\text{sim}(x, y) =$

$$\frac{|x \cap y|}{|x \cup y|}.$$

It finds the overlap between two documents by calculating the number of

terms that are common between them. The Jaccard measure can work for both continuous and binary feature vectors. Another class of distance functions is known as the *Minkowski distances* [18], expressed as:  $\|x - y\|_p = \sqrt[p]{\sum_{i=1}^n |x_i - y_i|^p}$ , where  $x, y \in \mathfrak{R}^2$ . This distance function describes an infinite number of distances indexed by  $p$ , which assumes values greater than or equal to 1. Some of the common values of  $p$  and their respective distance functions are:

$$\begin{array}{lll}
 p = 1 : & \text{Manhattan Distance} & \|x - y\|_1 = \sum_{i=1}^n |x_i - y_i| \\
 p = 2 : & \text{Euclidean Distance} & \|x - y\|_2 = \sqrt{\sum_{i=1}^n |x_i - y_i|^2} \\
 p = \infty : & \text{Tschebyshev Distance} & \|x - y\|_\infty = \max_{i=1,2,\dots,n} |x_i - y_i|
 \end{array}$$

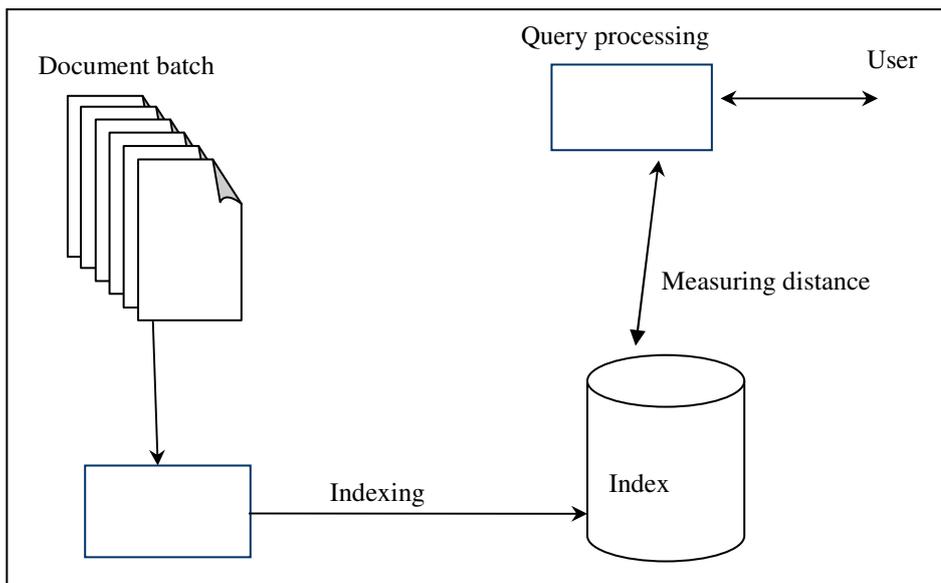
A similarity measure was also developed based on word sequences to measure the similarity between documents [38]. It considers the number of shared sequences between two documents along with their lengths, frequencies, and level of significance in each document.

Although these techniques could be appropriate to measure distances between vectors of numerical features, the inherent inadequacy of word-focused approaches still exist. There is no perfect correlation between comparing words and comparing meanings. As the example sentences in the previous section show, the correlation can be very low indeed. Better mining results can be attained by measuring meanings instead of counting words found in documents. Thus it is imperative to develop semantic representations of text, and distance measures that can determine whether one statement is an instance of, or quite different from, another statement.

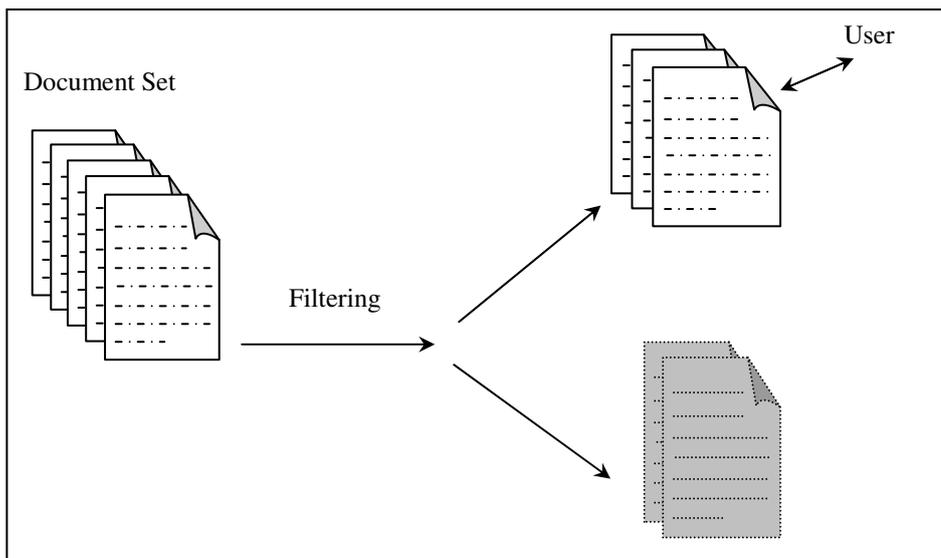
### 2.1.3 Mining Processes

Document mining processes such as document clustering, document classification, information retrieval (IR), information filtering (IF), and information extraction (IE) may vary in their requirements and specifications, yet their goals are to discover and extract knowledge from documents. They facilitate tasks such as classifying documents, discovering relationships or associations between documents, finding relevant documents to queries, routing documents to interested users, and incorporating text with other structured data. IR is concerned with finding relevant documents in response to a user request and ranking them accordingly [5, 57]. This is normally done by measuring the distances between documents and queries in their transformed form in an index (see Figure 2.2). When relevancy and similarity measuring is performed with the intent of transmitting a document to a user, or a set of interested users, it is usually referred to as IF [6, 72] (Figure 2.3). It is also used to either accept or reject an incoming document, as in e-mail filters that attempt to screen for junk mail. The goal of IE is to locate specific information and produce structured format from unstructured or semi-structured documents [87] (Figure 2.4). The output of an extraction system is usually tabular or fixed-format forms that are filled out with unambiguous data. This is done through analyzing those portions of each document that contain relevant information. The relevancy is determined by predefined domain guidelines which specify what types of information the system is expected to find. The aim of the document classification task is to assign a new document to one of a pre-existing set of document classes (Figure 2.5). In this setting, the task of creating a classifier consists of discovering useful characterization of the documents that belong to each class. Although this can be done by hand, the standard approach is to use supervised machine learning. In particular, classifiers can be

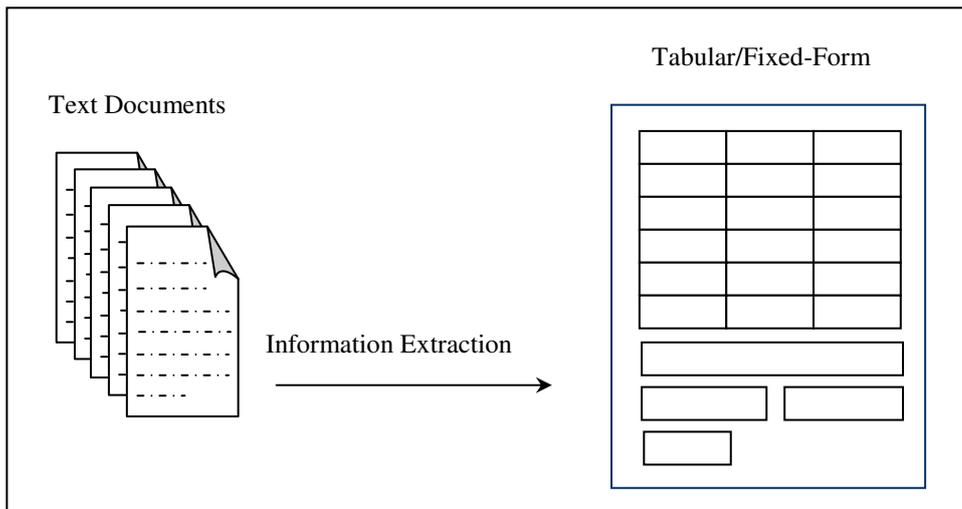
trained on a set of documents that have been labelled with the correct class. The classification task assumes existing categories, or clusters, of documents. By contrast, the task of document clustering is to create, or to discover, a reasonable set of clusters for a given set of documents. A reasonable cluster is defined as one that maximizes the within-cluster document similarity, and minimizes between cluster similarity [7] (see Section 2.1.3.1).



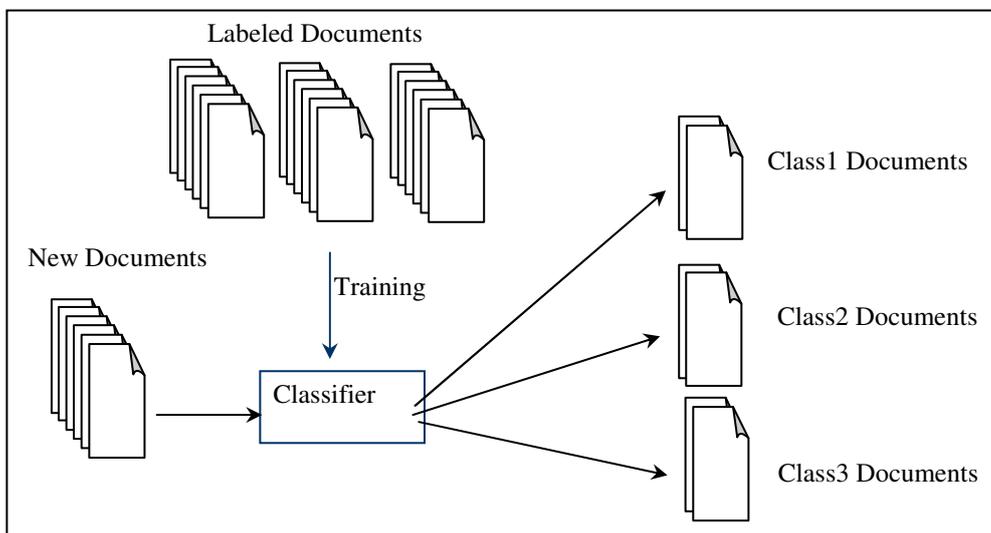
**Figure 2.2** Information Retrieval



**Figure 2.3** Information Filtering



**Figure 2.4** Information Extraction



**Figure 2.5** Document Classification

At a certain level of simplicity, these processes can be looked at as being relatively similar. They all make use of a text representation model, and a similarity measure to perform their specific tasks. Hence, knowledge-rich representations of text combined with accurate similarity measures will definitely result in enhanced mining processes outputs. Since document clustering is used as a case study to illustrate the working and the efficacy of the proposed document mining approach, a more detailed review about it is given in the following subsection.

### 2.1.3.1 Document Clustering

Document clustering aims to automatically divide documents into groups based on similarities of their contents. Each group (or cluster) consists of documents that are similar between themselves (have high intra-cluster similarity) and dissimilar to documents of other groups (have low inter-cluster similarity) (Figure 2.6). Clustering documents can be considered as an unsupervised task that attempts to classify documents by discovering underlying patterns, i.e., the learning process is unsupervised, which means that no need to define the correct output (i.e., the actual cluster into which the input should be mapped to) for an input.

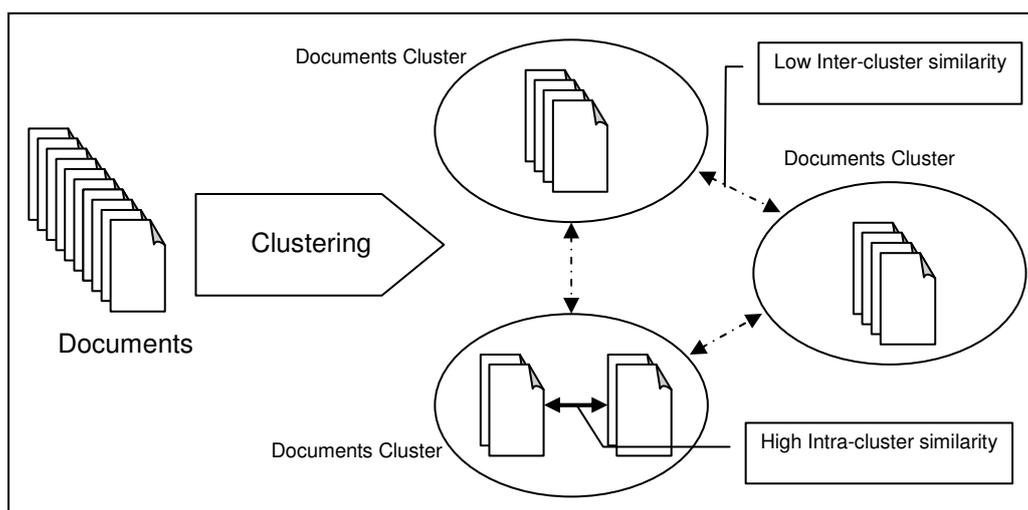


Figure 2.6 Document Clustering

Document clustering is used to disambiguate results of information retrieval systems, by displaying them into specific topics. Aside from visualization of search results, it is used for taxonomy design and similarity search. Topic taxonomy (e.g., Yahoo!, and Open Directory *dmoz.org*) are constructed manually, but this process can be assisted by clustering a large samples of documents. Clustering can also help speed up similarity search, where close-by documents are to be retrieved.

Additionally, in [76] it is argued that sentence-based text clustering can be a key factor for performance improvement of automatic speech recognition systems.

There are many clustering techniques in the literature, each adopting a certain strategy for detecting the grouping in the data, such as K-means algorithm [39], Expectation Maximization [26] and hierarchical clustering [49], and many others surveyed in [7]. They can be divided into three main categories; partitioning, geometric, and probabilistic [17]. The following subsections report some algorithmic approaches under their perspective categories.

#### **a) Partitioning**

In this approach to cluster, objects are partitioned into  $k$  clusters  $C_1, \dots, C_k$  such that the inter-cluster similarity is minimum and the intra-cluster similarity is maximum. Distances and similarities are usually measured with regard to the cluster centroid. Jain and Dubes [49] gave a thorough review regarding clustering techniques including partitioning clustering such as:

***k*-means and Fuzzy C-means** [39]. K-means tries to find  $k$  groups in the data. Basically, it iterates through two steps. The first step is to find the mean of a cluster by averaging all the instances that belong to that cluster. The second step is to update cluster membership according to a distance measure between each instance and all centers of clusters, and choosing the closest one. These two steps are repeated until no more instances are moved between clusters. The time complexity of the algorithms is  $O(kndT)$ , where  $k$  is the number of clusters,  $n$  is the number of documents,  $d$  is the dimension of the feature space, and  $T$  is the number of iterations.

A variant of  $k$ -means that allows overlapping clusters is known as *Fuzzy C-means* (FCM). Instead of having binary membership of objects to their respective

clusters, FCM allows for fuzzy (or degrees) of memberships [50]. Krishnapuram et al [54] proposed a modified version of FCM called “Fuzzy C-Medoids” (FCMdd) where the means are replaced with medoids. They claim that their algorithm converges very quickly and has a worst case of  $O(n^2)$  and is an order of magnitude faster than FCM.

**Hierarchical and Agglomerative** [49]. Hierarchical Agglomerative Clustering (HAC) is performed by assigning each document as cluster, and then (in a greedy manner) combines clusters according to the most similar clusters at each iteration. In this case it is considered as a *bottom-up* approach and its time complexity is  $O(n^2)$ . A *top-down* variant of the approach sets the number of clusters first, and creates a random partition of clusters, then refines the clusters so as to satisfy one of the cost function mentioned above. Notice that HAC accepts a similarity matrix as its input. A similarity matrix is basically a large table representing the distance between each pair of documents in the collection.

***k*-Nearest Neighbour Clustering (*k*-NN)** [18, 24]. This algorithm is used in classification and in clustering. It utilizes the property of nearest neighbours  $k$ , i.e., an object should be put in the same cluster as its nearest  $k$  neighbours. The algorithm accepts a user specified threshold,  $e$ , on the nearest-neighbour distance. For each new document, the similarity is compared to every other document, and the top  $k$  documents are chosen. Accordingly, the new document is grouped with the cluster where the majority of the top  $k$  documents are assigned.

**Single Pass Clustering** [18, 43]. Single pass clustering method also expects a similarity matrix as its input and outputs clusters. The clustering method takes each object sequentially and assigns it to the closest previously created cluster, or creates a new cluster with that object as its first member. A new cluster is created when the similarity to the closest cluster is less than a specified threshold. This threshold is the

only externally imposed parameter. Commonly, the similarity between an object and a cluster is determined by computing the average similarity of the object to all objects in that cluster.

## b) Geometric

Geometric approaches to clustering project the problem space into a two or three dimensional space to aid the user in spotting the clusters. By doing that they allow an easy way to visualize clusters, which is often considered an advantage.

**Self-Organized Maps.** Kohonen Self-Organizing Map (SOM) can be visualized as a sheet-like neural-network array, the cells (or nodes) of which become specifically tuned to various input signal patterns or classes of patterns in an orderly fashion. In the basic version, only one map node (winner) at a time is activated corresponding to each input. The locations of the responses in the array tend to become ordered in the learning process as if some meaningful non-linear coordinate system for the different input features were being created over the network [45, 53]. All vectors are fed to the network, as one epoch, and for a number of iterations the network is trained to map them to a specific number of clusters.

**Multi-dimensional Scaling (MDS).** In MDS the system input is the pair-wise (dis)similarity between documents, rather than the internal vector-space representation of the documents. The algorithm seeks to project the documents onto a low-dimensional space (often 2D or 3D) with minimum distortion of the original pair-wise distances. This is usually done by keeping the Euclidean distance between any pair of points in the low-dimensional space as close as possible to the distance between them specified by the input. Formally; if  $d_{i,j}$  is a (symmetric) user-defined measure of distance (or similarity) between documents  $i$  and  $j$ , and  $\hat{d}_{i,j}$  be the

Euclidean distance between the point representation of the two documents chosen by the MDS algorithm. The *stress* of the embedding to be minimized is given by: *Stress*

$$= \frac{\sum_{i,j} (\hat{d}_{i,j} - d_{i,j})^2}{\sum_{i,j} d_{i,j}^2}.$$

Convergence of this function is often difficult to achieve, and is

usually done using iterative relaxation (hill climbing). Initially, points are assigned random coordinates, and are moved iteratively by small distance in a direction that locally minimizes the stress.

**Latent Semantic Indexing (LSI).** It is an algebraic-based algorithm used to represent documents [25, 40, 64]. LSI assumes that there is some underlying or latent structure in the pattern of word usage across documents, and that statistical techniques can be used to estimate this structure. It considers implicit higher-order structures in the association of terms with documents, i.e., “semantic structure”. The technique successfully takes into account *synonymy* (i.e., words that can express the same meaning) and *polysemy* (i.e., a word that can be used to express different meanings). After factoring the term-document matrix, and decomposing it to compute *singular value decomposition* (SVD), it ranks it so the top  $r$  singular values capture the “signal” in the original matrix, leaving out the lower singular values to account for the “noise”.

### c) Probabilistic

Most of the aforementioned clustering approaches are considered sensitive to the similarity measures. The probabilistic approach assumes that documents follow specific distributions that should be modeled by finding the distributions parameters. Practically, estimating these parameters is the clustering process itself. *Maximization Expectation* algorithm, *Probabilistic LSI*, and *Multiple Cause Mixture Model*

(*MCMM*) are examples of the approach. Some of these techniques are discussed in [70]. A drawback of such approaches is that they are computationally expensive, thus may be appropriate for off-line clustering.

#### 2.1.4 Evaluation Methods

In general, to assess the quality of results produced by a document mining process, two classical indices are used; *precision* and *recall*. Precision  $P$  is the fraction of all correct answers included in a produced set of answers. Recall  $R$  is the fraction of the responses that are actually correct from all possible correct answers.

$$P = \frac{\#Correct\_Answers}{\#Answers\_Produced}, \text{ and } R = \frac{\#Correct\_Answers}{\#Total\_Possible\_Corrects}.$$

These evaluation indices are borrowed from IR literature. In IR, precision is used to indicate the fraction of all relevant documents included in a ranked list of retrieved documents. Recall is the fraction of the top responses that are actually relevant in the whole document set [98, 17].  $P$  and  $R$  have been used to evaluate other mining processes results with slight alteration in their definitions. For instance, in document clustering and document classification, the recall for cluster/class  $c$  is the ratio of processed documents manually classified as  $c$  with regard to the whole document set, and precision is the ratio of these documents clustered/classified as  $c$  by the process that were also manually classified as  $c$  regarding the produced set of results. Moreover, in IE recall is interpreted as the ratio of the information that has been correctly extracted, and precision as the proportion of the extracted information that is correct [29].

Combination methods have been also used such as the *F-measure*, which combines precision,  $P$ , and recall,  $R$ , in a single measurement as follows:

$$F = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

The parameter  $\beta$  influences how much to favour recall over precision. Researchers frequently report the *F1* score of the system where  $\beta=1$ ; weighing precision and recall equally. Thus, using F-measure, the relative performance of systems reporting different values for recall and precision, can be easily observed.

For document clustering, Entropy is used as another external similarity measure that can indicate the quality of clusters with reference to external knowledge. It is used for un-nested clusters or for the clusters at one level of a hierarchical clustering. It is a measure of clusters homogeneity. The entropy  $E_i$  of a cluster  $i$  is calculated using Shanon [86] standard formula:  $E_i = -\sum_j p_{ij} \log(p_{ij})$ , where  $P_{ij}$  is the probability of documents of cluster  $i$  belong to class  $j$ . An overall entropy  $E$  for all clusters can also be calculated as the sum of entropies for each cluster weighted by the size of each cluster as follows:  $E = \sum_i (\frac{N_i}{N} \times E_i)$ , where  $N_i$  is the size of cluster  $i$ , and  $N$  is the total number of documents. A common internal quality measure for clustering is the overall similarity and is used in the absence of any external information such as class labels. Overall similarity measures cluster cohesiveness by using the weighted similarity of the internal cluster similarity:

$$\text{OverallSimilarity}(S) = \frac{1}{|S|^2} \sum_{x,y \in S} \text{sim}(x,y),$$

and  $\text{sim}(x, y)$  is the similarity between the two objects  $x$  and  $y$ .

These evaluation techniques depend on having prior exhaustive knowledge about document contents and the taxonomy of the corpus. The techniques rely on the existent of manually identified relevancies of all the documents to queries, labels indicating documents belonging to pre-specified classes, etc. These requirements are

expensive, if not impossible to have, considering the large document sets (such as the Web) and the limited availability of human expertise. Moreover, there is an inherent deficiency when an evaluation is based on human subjective assessments. No two individuals agree about what each neither expects from the mining process, nor do they think of what is relevant, similar, or proper since their perceptions and assessment capabilities differ [65]. As a result, in [85] user-based relevancy feedback was collected to evaluate a web IR system, and in [76] the quality of clustering was judged based on how it affected the completion of another task, i.e., speech recognition.

More efficient and more meaningful evaluation techniques are still required. Ideally, they should represent the quality of the results regarding all the criteria, including accuracy, usability, speed, scalability, simplicity, and a simple interpretation of results, as well as a resemblance to the manually constructed ideal results, if available.

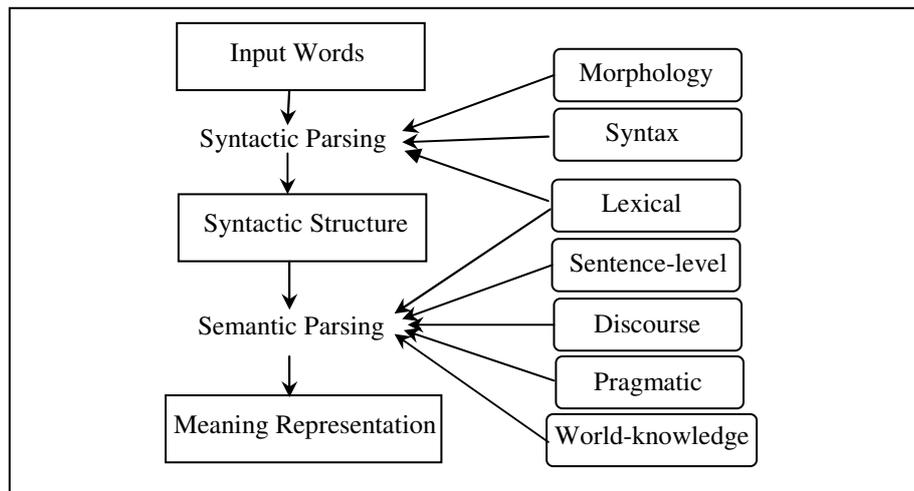
---

## **2.2 Natural Language Understanding**

Natural language understanding (NLU) is a sub-field of the more wide research area of natural language processing (NLP). NLP is concerned with developing computational techniques that process human languages. NLP applications may extend from word counting and automatic hyphenation, to automated question answering systems, and real-time language translation, as long as the knowledge of language is utilized [51].

Language processing can be divided into two areas: (1) Written or textual natural language processing, using lexical, syntactic, and semantic knowledge of the language

as well as any required real world information (See Figure 2.7). (2) Processing of spoken language using all the information needed in processing written natural language plus additional knowledge about phonology as well as enough additional information to handle the further ambiguities that arise in speech. As the focus here is on the first area, the steps in the process of natural language understanding applied specifically to written texts are explained in the following sub-sections:



**Figure 2.7** Information Flow in Text Natural Language Understanding

### 2.2.1 Morphological analysis

Morphological analysis (MA) is mainly concerned with how individual words are analyzed into their components, and non-word tokens (such as punctuations) are separated from the words. For example, in the phrase "*John's house*" the proper noun "*John*" is separated from the possessive suffix "*'s*". It is a fundamental step in every area of NLP, as words are the building blocks of all human languages; spoken, signed, or written. Computational models of the spelling, pronunciation, and morphology of words have been extensively investigated and used in real-world tasks such as: automatic speech recognition (ASR), text-to-speech synthesis (TTS), and the correction of spelling errors.

Some of the most popular computational models used for MA are: the finite-state automata (FSAs) and regular expressions, finite-state transducers (FSTs), weighted transducers, hidden markov model (HMM), and the N-gram model of word sequences. Other heuristics and corpus-based techniques have been used and surveyed in [83].

### **2.2.2 Syntactic Analysis**

In syntactic analysis, or syntax analysis, linear sequences of words are transformed into structures that show the words formal grammatical relationships. This parsing step converts the flat list of words of a sentence to a structure that defines the grammatical units represented by that list. It is done by taking into account several imposed constraints such as word order, number agreement, and case agreement. Several outcomes of this processing stage are gained like: parts-of-speech tags, clauses and phrases chunking, and the way words depends on other words in sentences represented in parse trees.

Considerable advances have been made in syntactic modeling of natural language. Efficient parsers with a broad domain have become available with an accuracy close to human performance (>95%) [14, 63]. Many approaches have been pursued including context-free grammars, lexicalized grammars, feature structures, and metatheoretical issues. Algorithms for dealing with this knowledge have been also introduced like: the Earley and CYK algorithms for parsing and the unification algorithm for feature combination. To deal with ambiguities and multiple interpretations, there have been probabilistic models for this syntactic knowledge including HMM part-of-speech taggers, and probabilistic context-free grammars [46].

### **2.2.3 Semantic Analysis**

In this step of semantic analysis, the structures created by the syntactic analyzer are assigned meanings. In semantic analysis, individual words are mapped into appropriate objects in a knowledge base, to create the correct structures that correspond to the meanings of the individual words combine with each other. This is done after, and sometimes in conjunction with, the syntactic processing. It involves mapping sentences to some formal meaning representations such as logical expressions. These formal representations of sentences meaning are produced when the following steps are usually applied: lexical processing, sentence-level processing, discourse integration, pragmatic analysis, and the integration of world knowledge representation [4]. These steps are explained in the following:

#### **2.2.3.1 Lexical Processing**

This step can be described as looking up individual words in a dictionary. It may not be possible to choose a single correct meaning, since there may be more than one. The term variability problem of natural languages has been a principal reason for the failure of many systems. Individuals choose a variety of words to describe the same object or operation, with little overlap between their choices [10, 36]. The process of determining the correct meaning (sense) of individual words in a text or discourse is called word sense disambiguation (WSD) or lexical disambiguation. The correct meaning needs to be distinguished from other senses which are potentially attributable to that word. This meaning varies significantly according to the context in which it is used. A classical example is the word '*bank*' that has a completely different meaning in financial text than in geological text. There are even harder cases where different senses of this word distinguished, such as, '*bank*' as a financial institution and as a

building. They both may appear in the same context. Hence, more details of their use should be accounted for in order to distinguish between them. The problem is eventually exploding as more words are taken into account.

The WSD has been a problem of interest since the early days of computer treatment of languages. It has been described as *AI-complete* that cannot be solved unless all the difficult problems in artificial intelligence (AI) (such as common sense representation and encyclopedic knowledge encoding) are first solved [48]. Though, there is a large range of approaches and efforts have been put to investigate this problem, "it is a fact that to date no large-scale, broad coverage and highly accurate word sense disambiguation system has been built" [30].

Approaches to build WSD systems fall into two main directions: knowledge-driven methods, and data-driven (or corpus-based) techniques. The knowledge-driven approaches require the use of external knowledge sources, while the knowledge-poor approaches make use of information about the contexts of previously disambiguated instances of the word derived from corpora. Middle grounds between these two approaches have also been pursued.

### **2.2.3.2 Sentence-level Processing**

This part of semantic parsing involves producing an overall analysis of sentences meaning. This is a context-independent analysis, i.e., it considers the meaning of a sentence regardless of the context in which it is used. To do this, precise representation languages are used. Mathematics and logic offer formally specified representation languages constructed from simple building blocks. Several different representations are used including first order predicate logic (FOPL), instant tense logic (ITL), period structures (PS), event structures (ES), conceptual dependency

(CD), and semantic nets (SN). There may be some differences among these approaches, yet at an abstract level they all share the notion that a meaning representation consists of structures composed from a set of symbols and relations between them.

Most of the aforementioned and similar representation schemes were designed to be *Verifiable*<sup>1</sup>, *Unambiguous*<sup>2</sup>, *Canonical*<sup>3</sup>, *Inductive*<sup>4</sup>, and *Expressive*<sup>5</sup> with the aim to produce more sophisticated "intelligent" systems. Systems that are able to, for example, answer essay-like questions, decide on (or learn) an action by reading a text, and/or deduce information not explicitly mentioned in an input passage [51]. Such requirements make these representation methods appropriate and provide what a document mining system necessitates. Most mining processes need representation languages that are able to express distinct reading of a sentence as a distinct formula, and capture its intuitive structure and meaning. Hence, they need to recognize sentences that appear to be structurally similar to produce similar structural representations, and identify the meanings of sentences that are paraphrases of each other and those that are closely related.

### **2.2.3.3 Discourse Integration**

Discourse analysis, also called co-reference resolution, involves determining how context influences the interpretation of a sentence. The meaning of an individual sentence may depend on the sentences that precede it and may influence the sentences

---

<sup>1</sup> Is able to compare the state described by the representation to the state of the world as modeled in a knowledge base.

<sup>2</sup> Can produce a single unambiguous interpretation regardless of the ambiguity in an input sentence.

<sup>3</sup> Should assign the same meaning representation to inputs that mean the same.

<sup>4</sup> Can be used to draw valid conclusions based on the meaning representation of inputs and its store of background knowledge.

<sup>5</sup> Produces a single meaning representation structure that adequately represent the meaning of any sensible inputs

yet to come. Particularly, in a coherent discourse, entities involved in a sentence must either have been introduced explicitly or they must be related to entities that were mentioned before. This step is especially important in understanding the information conveyed by interpreting articles such as pronouns and temporal aspects.

The process includes the task of determining what phrases in a document refer to the same thing, e.g., pronouns. In the sentence "*On Friday, John parked his car beside the tree in the bank's parking lot. He just bought it the day before,*" a successful algorithm would determine that "*it*" refers to "*the car*" rather than "*the tree, the bank*", or "*the parking lot*", and would resolve to what day the phrase "*the day before*" is referring to. More generally, to do successful text analysis and understanding, one needs to identify all noun phrases that co-refer in a discourse. For example, in the sentence "*TD Canada Trust lent John \$100,000 dollars to buy the house; the bank is charging interest on the money,*" it must be figured out that "*TD Canada Trust*" and "*the bank*" co-refer, as do "*\$100,000 dollars*" and "*the money*".

There exist several discourse interpretation algorithms such as those due to Lappin and Leass [55], and Kennedy and Boguraev [52] that try to build evolving representations of the discourse state, called *discourse models*. This discourse model contains symbols of the entities that have been referred to and the relationships they have. Hobbs [44] has developed an algorithm that takes the syntactic representations (parse trees) of the sentences up to the referring expression and performs a search for a referenced noun phrase on these trees. No uses of explicit representation of a discourse model here, but correct syntactic structures are assumed to be available as the input.

The difficulty of performing this task is due to the many ways natural languages refer to entities. Each form of reference has a different way to point to the

referent with respect to the discourse and the set of beliefs about the world. However, under certain assumptions, mid-80% performance range has been achieved by existing algorithms.

#### **2.2.3.4 Pragmatic Analysis**

This step is concerned with mechanisms that try to figure how sentences are used in different situations and how context affects their interpretations as a whole. Here, the structure representing what was said is reinterpreted to determine what was actually meant. To understand the purposeful use of most sentences, it is necessary to know the context in which it was uttered and to possess world knowledge. Examples of such sentences are "*John's new car drinks gasoline like you would not believe it*" and "*Time flies like an arrow*". In general, for a program to intelligently understand a dialog, it must be able to represent its own beliefs about the world, as well as the beliefs of others and their beliefs about its beliefs, and so on .

#### **2.2.3.5 World Knowledge Representations**

Natural Language cannot be fully understood without considering the everyday knowledge about the world. World knowledge representations are techniques that try to relate what was said to some evolving situation in the world. This includes the general knowledge about the structure of the world that languages users must have in order to, for example, maintain a conversation. It includes what each language user must know about the other user's beliefs and goals. General world knowledge is essential for solving many language interpretation problems, one of the most important being disambiguation. For example, the proper understanding of the following two sentences depends solely on the reader's background knowledge of the

appropriate time needed for reading and for evolution: "*John read a book about evolution in ten minutes*" and "*John read a book about evolution in the last million years*".

## The Semantic-based Document Mining Framework: An Overview



*"Imagination is more important than knowledge. Knowledge is limited.*

*Imagination encircles the world."*

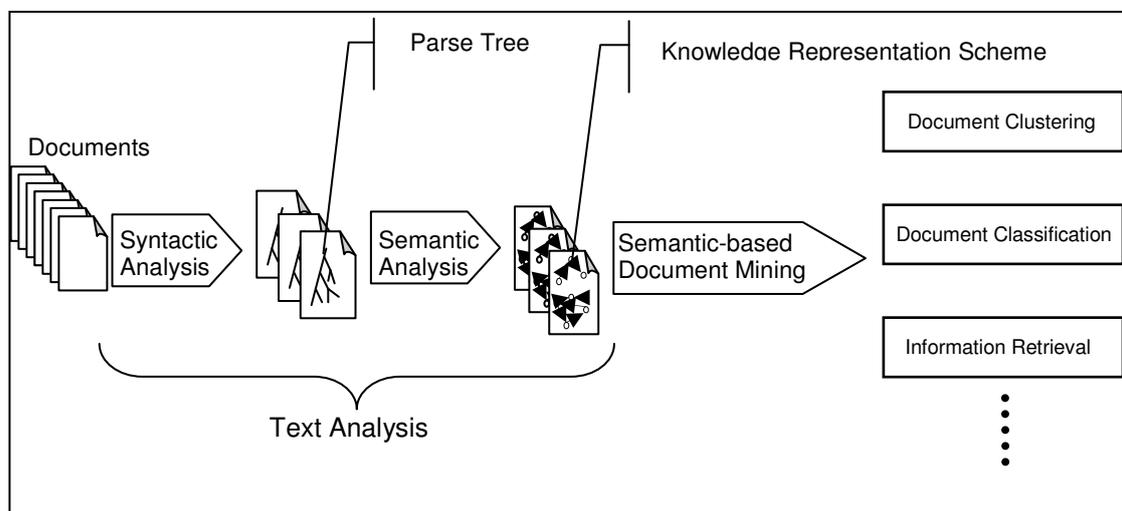
Albert Einstein (1879 - 1955).

---

In this chapter, an introduction is given to the new document mining paradigm based on semantic understanding of text. The system design and its mechanisms to mine texts based on semantic understanding of contents are described. An overview of the system architecture and its main components are introduced in Section 3.1. The system essential components, including the text parser, the similarity estimator, and the mining processes, are introduced in Sections 3.3, 3.4, and 3.5, respectively. This treatment is preceded by a discussion of some assumptions of the work in Section 3.2. Other complementary components that could be utilized in the system are discussed in Section 3.6. Details of the representation scheme, the similarity measure, and the implementation of the system are presented in Chapters 4, 5, and 6, respectively.

### 3.1 Architectural Overview

In this section an introduction is given to a framework for mining documents based on semantic understanding of text. As Figure 3.1 illustrates, a semantic understanding-based document mining system is often provided with a set of documents for which it is expected to produce higher level form of informative depictions that satisfy some user needs. These user needs are acquired through some mining processes such as document clustering, document classification, and information retrieval. The adopted approach is based on analyzing text in documents before proceeding with the different mining processes requirements. The text analysis step comprises syntactic analysis to extract syntax structural descriptions (e.g. part of speech tags, phrasal chunks, and parse trees), and semantic analysis that produces formal knowledge representations of the documents contents.



**Figure 3.1** The Semantic Understanding Approach

The architecture of a generic semantic-based document mining system is illustrated in Figure 3.2. It consists of three major components:

- Text parser
- Similarity estimator

- Mining processes

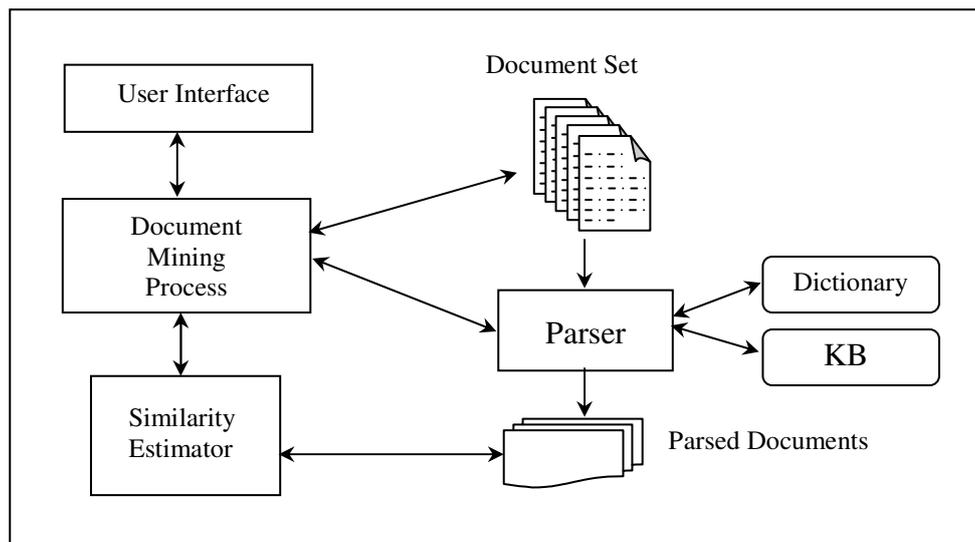
A semantic-based document mining system can be built from these three resources to satisfy the requirements. These components are interconnected in most cases, i.e., outputs of one component can be the inputs of others and vice versa, which makes the system highly integrated, but yet modular.

Assume that there is a mining process for a task  $T$ , and a collection of text documents  $D$  on which the task is to be performed. The subsequent algorithmic steps depict the logical flow of such a system:

1. For each document in  $D$ , run the parser to convert the text to a semantic representation model. Let  $D^s$  be the set of all outcome representations of these documents.
2. If the mining process involves user queries  $Q$ , as in information retrieval, (or new incoming documents, as in document clustering), then use the same parser to convert them to a semantic representation structures  $Q^s$ .
3. Use the similarity estimator to determine the closeness of documents and queries in their transformed forms, i.e.,  $D^s$ , and  $Q^s$ .
4. Output the results by displaying texts in the original  $D$  that correspond to their counterparts in  $D^s$ .

The text parser is responsible for reading input texts and converting them to canonical and symbolic knowledge representations. The first step in a complete parsing procedure would likely be an automatic syntactic analysis. This will gather important structural objects by recognizing text components (such as, phrases, sentences, and paragraphs), tagging text with parts-of-speech annotations, and producing parse trees. The second step is semantic analysis, where high level knowledge is extracted and abstracted in some formal knowledge representation.

Depending on the kind of parser, these representations could be sketchy scripts, or graph-based data structures that match the meaning of the input text. The success of this stage is dependent on advances made in syntactic and semantic modeling of NL, and on the availability of an efficient, broad, and domain-independent of these parsers. An introduction to the semantic-based text representation is given in Section 3.3, and detailed in Chapter 4.



**Figure 3.2** An Overall System Architecture

The similarity estimator takes two understood (parsed) texts and determines the semantic distance between them. It is asserted that when the parser properly converts text to semantic representations and the similarity estimator identifies their closeness with respect to meaning, the parsing and the distance measuring operations are forming a homomorphism with human judgments about documents similarities. That is to say, human judgments about the similarity of two texts are emulated by the process of parsing both texts into knowledge representations and then measuring the distance between these representations. Section 3.2 describes this similarity homomorphism assumption in more details. The similarity estimator algorithm depends heavily on the chosen knowledge representation structure. For instance, when the text is represented as graph structures, the similarity algorithm has to search

through these graphs and estimate their levels of commonalities. The proposed similarity estimator is based on inexact tree matching algorithm; this estimator is introduced in Section 3.4, and detailed in Chapter 5.

Document mining processes (such as, document clustering, document classification, information retrieval, information filtering, and information extraction) vary in their requirements and specifications. Nevertheless, they all require documents to be representing in some formal way, and they all measure similarities in one way or another. Thus, they can all make use of the semantic parser, the similarity estimator, and/or other supplementary components. In information retrieval, for example, documents are first indexed (i.e., formally represented) and a relevancy ranking would be produced based on similarity estimation process. A detailed discussion on one of these different mining processes, namely the semantic document clustering, is presented as a case study in Chapter 6.

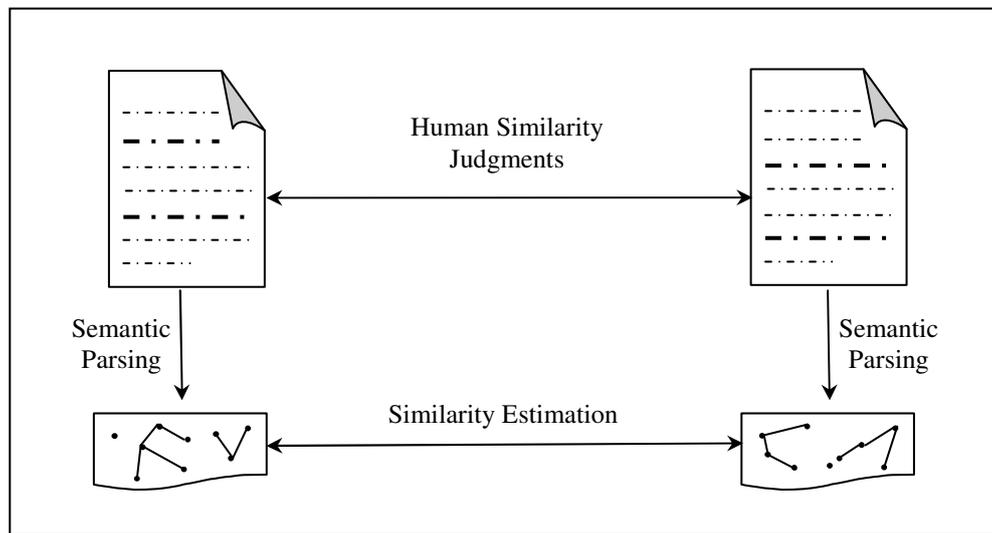
The supplementary components are utility components such as user interfaces, online dictionaries, etc, are mentioned in Section 3.5. These components, if available, could enhance further the mining performance and/or user experience. They provide support in achieving the goals of the main components and the different mining processes, or they can have their own goals that would contribute to the overall functioning of the system.

---

## **3.2 Similarity Homomorphism Assumption**

The proposed approach briefly described in Section 3.1 is based on the assumption that human judgments about the similarity of two texts are approximated by the process of parsing both texts into a formal knowledge representation and then

measuring the distance between them. These operations of parsing and distance measuring form a homomorphism with human intellectual activities of contents understanding and similarity assessments as shown in Figure 3.3. If humans were asked to discover knowledge from text or to assess the similarity between two pieces of text, they would first try to read and understand contents, and then based on the understanding they would determine similarities between the documents. This extends to the case where relations across groups of documents are considered. For instance, through understanding, an individual makes a judgment on to what group of documents he assigns to a new given document. Similarly, document mining tasks should be pursued through attempting to understand contents, and employing the understanding in finding similarity between documents. This should enhance the mining of documents and provide better accessing, searching, retrieving, organizing, managing of their contents, and reasoning about information they contain.



**Figure 3.3** Similarity Homomorphism Assumption

It is not, however, claimed that the way knowledge is represented nor how the understanding process is performed in human mind. It is believed that humans do represent knowledge in some representations and that they do try to understand

contents and use this understanding to search, compare, and reason about information they receive.

Therefore, the goal of this work can now be restated in terms of the assumption. The parsing and similarity estimation can approximate human judgments more closely than word counting and frequency matching. The objective is to provide a concrete example of systems that demonstrate these capabilities and to exploit them in various mining processes.

---

### 3.3 Semantic-based Text Representation

Every data mining system relies on a specific data model upon which it operates. Most text data models that are generally adopted by most document mining approaches are based on word counting and frequencies. A semantic-based data representation that should benefit the objective of document mining is introduced. The basic idea is to convert documents contents to structural and formal representations of meanings. The conversion should enable one to use an estimation component to measure the similarity of meanings between documents.

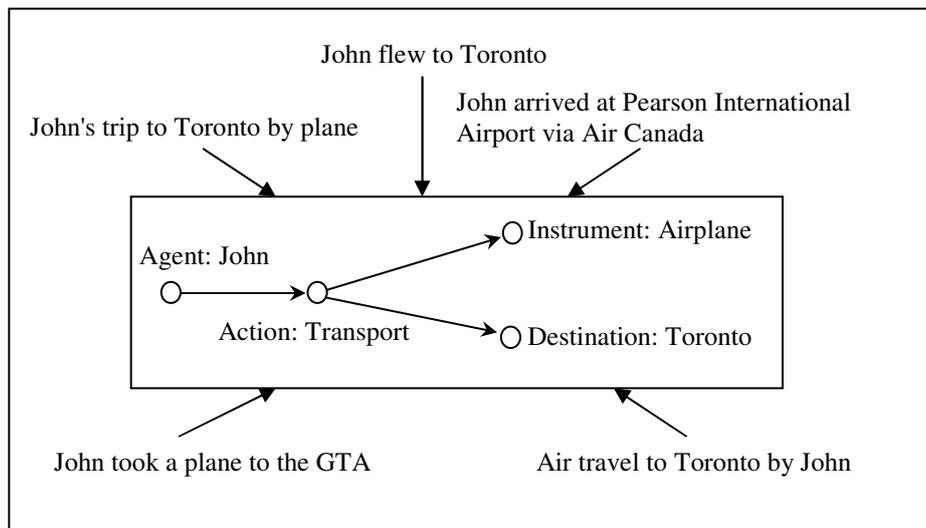
A representation scheme, called the *Semantic Graph Model* (SGM), is being introduced. It is developed to be suitable for document mining processes, where the focus is on the ability to express distinct readings of sentences as distinct formulas that capture their intuitive structures and meanings. The creation of the representation starts by creating predicate structures of sentences, augmenting the structures elements with valuable attributes, and taking all parsed sentences as the document representation. The representation is a graph-based data structure where entities, such as *agents*, *objects*, *states*, *actions*, *events*, *locations* are represented as vertices, and

relations between them are represented as arcs. Each node holds information about the entity it represents that could include its original text, syntactic information, semantic meaning, and relations with other nodes. In addition, each of these entries can have a fuzzy value that reflects the parser confident level regarding it, and incorporates the fuzziness found in the human expressions of perceptions.

Note that the mappings from text to meaning can be many-to-one, i.e., different sentences can express an idea differently but lead to the same interpretation. For example, the following three sentences mean the same: '*X succeeds Y as chairperson of Z.*', '*Z named X as its new chair-person after Y.*', and '*Y was succeeded by X as chair-person of Z.*' This variability can be captured by choosing to use a canonical knowledge representation. That means all NL constructions that have the same basic meaning must be parsed into the same representation. This property simplifies the similarity measuring process and improves the accuracy of similarity estimation for a given level of simplicity. That is because insignificant details and variations in sentences are trimmed early in the parsing process and as a result only condensed meaning representations of texts are used to estimate similarity. Figure 3.4 shows graphically how different sentences with the same basic meaning are given the same representation. This is an example to illustrate what a parser can receive and be required to process. Note that for the parser to determine the sentence '*John arrived at Pearson International*' matches the sentence '*Air travel to Toronto by John*', the parser must be able to determine that Pearson International is in Toronto. Thus, it must have access to world knowledge.

This approach may have to face some difficulties, such as having an efficient and accurate natural language parser. This parser should produce detailed representations of the various nested relations that may be expressed in an input text.

It has to be robust enough to discover partial knowledge from ill-formed or ambiguous and domain independent texts; i.e., capable of processing documents on a variety of topics. Unfortunately, such a parser is currently unavailable. However, a number of systems from natural language processing and understanding research fields are emerging and showing reasonable maturity. These systems offer a variety of designs and implementations representing various options. This research work makes use of such systems that have demonstrated acceptable performance, and extends upon them.



**Figure 3.4** An Example of a Canonical Representation

### 3.3.1 Parsing Sample Texts

Two examples of sentences are used to illustrate the process of converting text to SGM. Assuming these sentences were analyzed by the deep syntactic and semantic text parser described above:

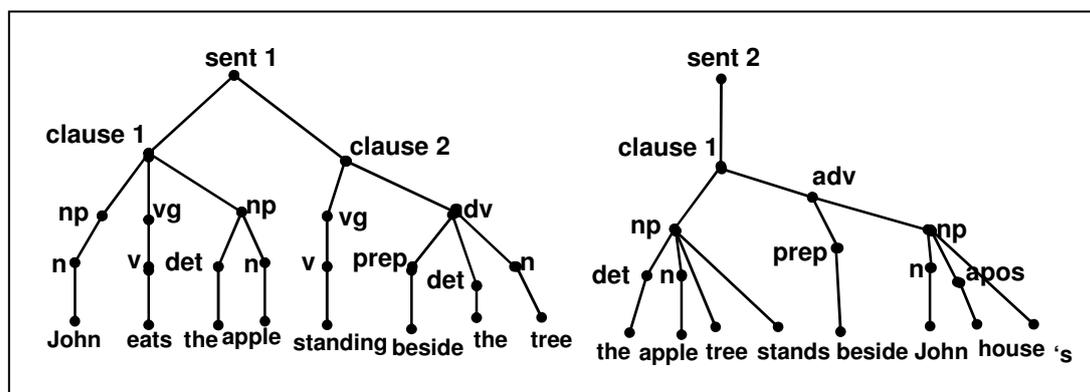
a) *'John eats the apple standing beside the tree,'* and

*'The apple tree stands beside John's house.'*

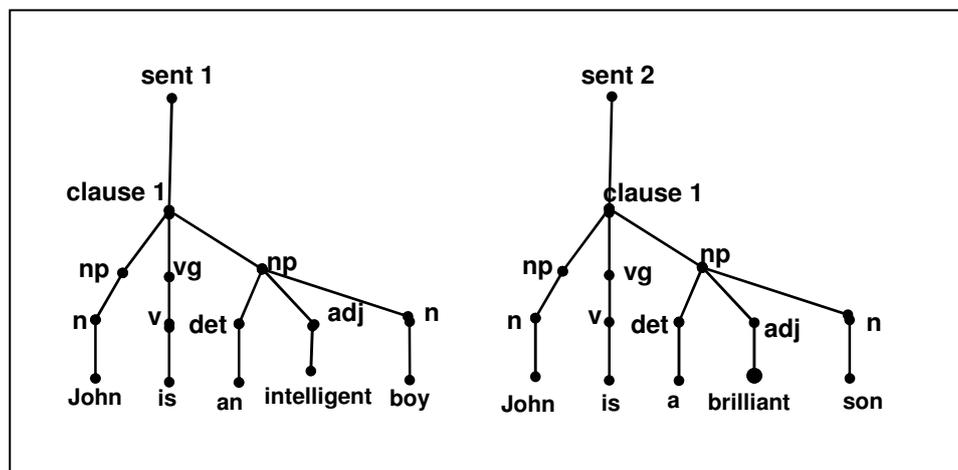
b) *'John is an intelligent boy,'* and

*'John is a brilliant son'*

The purpose of choosing these two samples is to show the advantages of the proposed approach over using traditional methodologies (such as, the-bag-of-words approach). This is manifested in extracting and representing the correct meaning from semantically different sentences even when there is an overlap in words usage, and similar sentences in meaning constructed using different words. The parse trees in Figure 3.5 (a) and (b) are the product of the syntactic analysis stage. In fact, it is clear from the parse trees one can see the dis/similarity of these sentences.



Error! Bookmark not defined. **Figure 3.5 (a)** Parse Trees of Sample Text



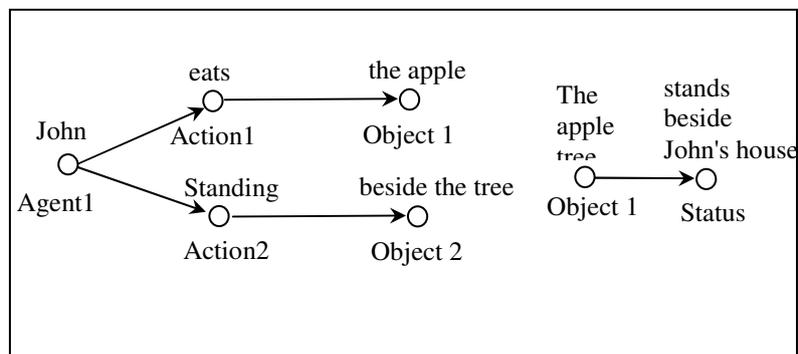
**Figure 3.5 (b)** Parse Trees of Sample Text

A sketch of an SGM knowledge representation scheme is depicted in Figure 3.6 (a) and (b) to show how the sentences could be represented in the higher level of semantic rather than just syntax. Every node in the graph represents a concept and holds some detailed attributable information about it. The concept could correspond to a word or phrases found in the text. According to the word or phrase syntactic tags

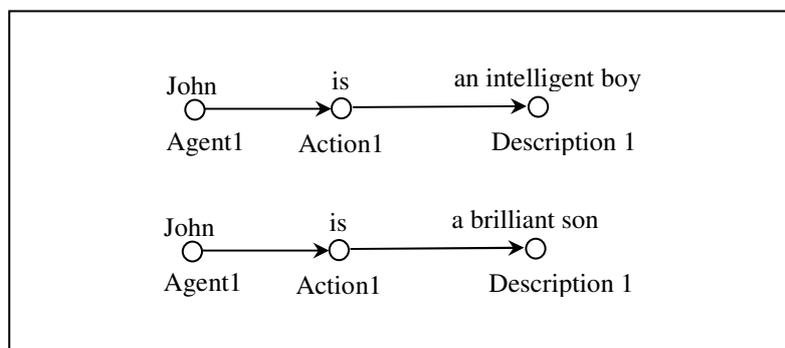
semantic role labeling (Agent, Action, Status, etc.) could be assigned. Examples of the information that could be in a node are illustrated for the first sentence in Figure 3.6 (c). It includes the following fields:

- *Name*: unique identification for the node
- *Type*: classification of the entity (e.g., Agent, Object, Action, and State)
- *Text*: the original text
- *Syn.*: the part of speech tag
- *Sen.*: dictionary senses (synonyms) of the entity
- *Sem.*: disambiguated meaning of the entity
- *Rel.*: relations (i.e., arcs) to other nodes in the graph

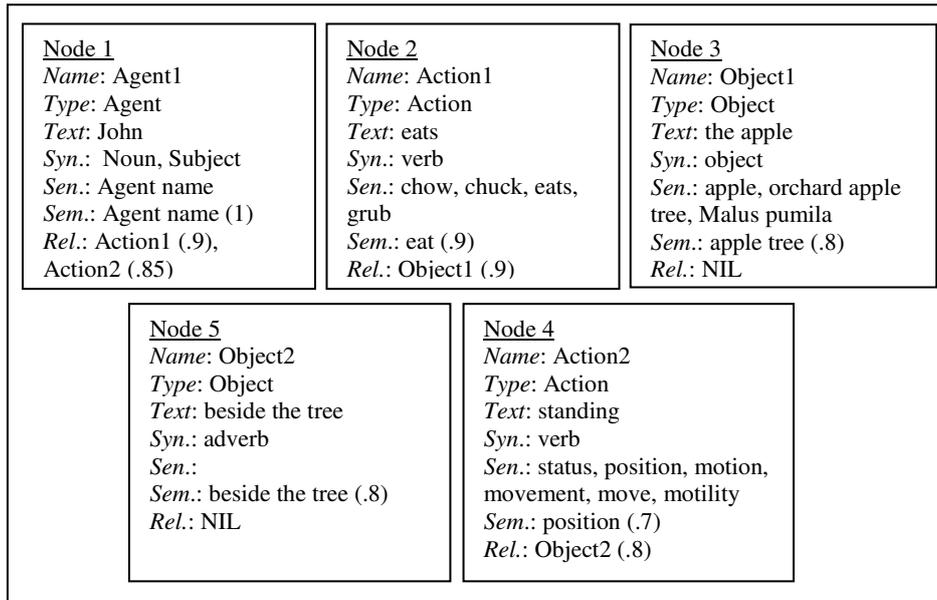
Some fields can have additional fuzzy values (ranges from 0 to 1) to represent the parser level of confidence. This is an important property as in the case of the *Sem.* field where the disambiguated sense is fuzzy, and *Rel.* that could also require a fuzzy function to represent its strength.



**Figure 3.6 (a)** SGM Knowledge Representations of Sample Texts



**Figure 3.6 (b)** SGM Knowledge Representations of Sample Texts



**Figure 3.6** (c) SGM Knowledge Representations of Sample Texts

Building an SGM representation is based on exploiting syntactic and semantic information that could be extracted from text. The representation characterizes sentences by converting them to formulas that reflect their structures and semantics. It identifies relationships between key concepts and accumulates valuable semantic clues about these concepts. Hence, SGMs can precisely represent text contents. SGM for a document is the accumulation of all sentences represented by the meaning representation. Each sentence is represented by an attributed graph of concepts as nodes and relations between them. The union of all graphs represent a whole document.

The sum of all sentences represented in the SGM portrays the knowledge in a document. All graphs (essentially trees) of the produced sentences representations are connected to one node that makes an inclusive rooted tree for the whole document. Figure 3.7 shows a skeleton of an SGM for a document. The SGM provides both the structural and conceptual elements that could be discerned through the syntactic and semantic processing of text.

Moreover, the representation scheme is extendable, and could include outcomes of different analysis and reasoning processes. The structural property of the model allows manipulation efficiently. This is especially important, as the goal of converting text to the SGM is to be used in further mining processes. In the following subsection, a brief introduction is given to the problem of comparing SGM representations and the proposal of a new similarity measure technique. The proposed measure is defined on the SGMs that are in essence tree structures with multiple symbolic attributes, and is based on finding all common similarity sub-trees.

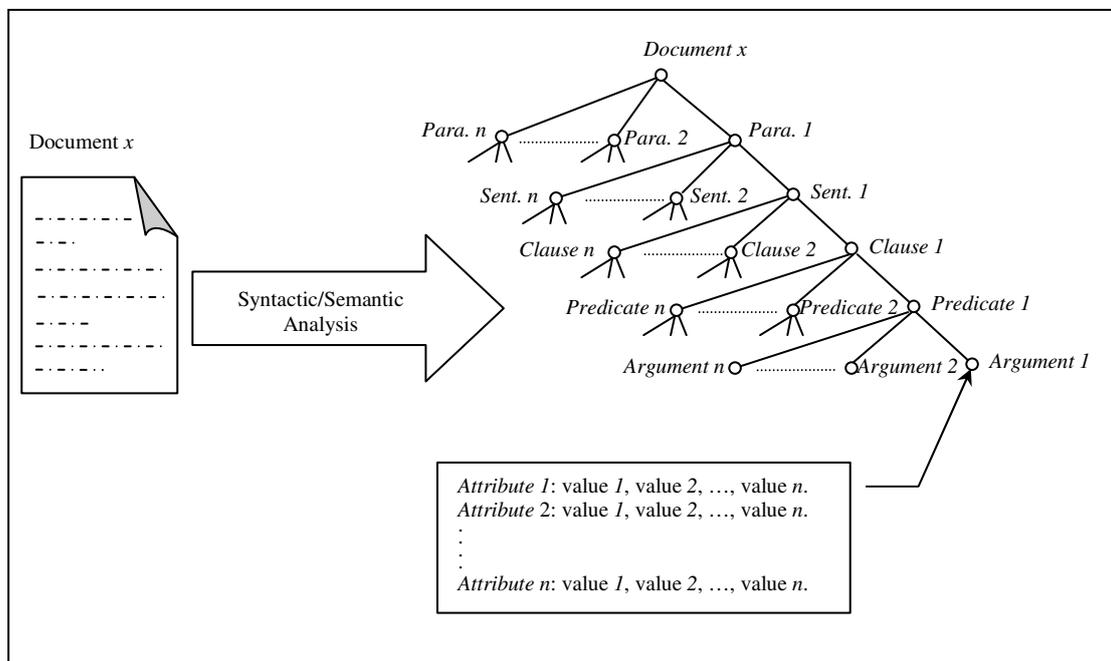


Figure 3.7 SGM Skeleton for a Document

### 3.4 Semantic-based Similarity Measure

Most document mining processes use some measures to assess similarities to determine the distances between documents meanings. Most of these similarity measures are often based on word frequencies to determine whether a given document is similar to or quite different from other documents. They rely on constructing

statistically meaningful frequency vectors to represent documents, and then calculate distances between them in a multi-dimensional geometric space. Examples of such approaches were provided in Section 2.2.1.

As SGM was adopted as the text meaning representation of choice, a proposed effective distance measuring technique for the model is to be developed. The SGM is an abstract representation constructed from symbolic elements rather than multi-dimensional numeric vectors. Thus, the similarity estimator component is responsible for searching the abstract representations of two documents, finding elements that are sufficiently similar, and yielding an overall similarity index. The similarity index should reflect the degree of commonality found between these structures; the more overlapping found between two representations the more similar the documents are, and vice versa.

The proposed approach to represent text data clearly captures meaning proximity, which is crucial for judging similarity of documents. However, the branching factor of searching through documents representations could be huge. In a real-world setting, a document might contain thousands of words and document sets could be very large. Thus, the semantic representation and its manipulation could become complex and expensive in terms of time complexity and memory usage. To overcome these problems, first, the semantic parser will be required to eliminate redundancies, and thus, more condensed meaning structures are to be produced. Second, the specialized distance estimator should be designed to be computationally efficient. Developing a fast similarity estimator is an important part of this research work.

An inexact graph matching technique to approximately match graphs is utilized to calculate semantic distances between documents. Since SGMs are in

essence trees with multiple symbolic node attributes, the semantic distance measure is defined over attributed-trees. At an abstract level, the approach involves the computation of all similarity common sub-trees that do not overlap. These are distinct sub-trees found in the trees that exhibit similarities matching in their node attributes and structures. The eliminating of overlapping sub-trees is stressed to minimize overestimating similarity. The approach is based on a pseudo metric measure that can be computed in a polynomial-time. The development of this similarity measure is detailed in Chapter 5.

### 3.4.1 Similarity Estimation: An Example

Figures 3.8 (a) and (b) demonstrate the proposed distance estimator on the examples of Section 3.3.1. The figures depict the finding of the common similarity sub-trees in the SGMs of the two sentences. The symbol (x) denotes the nodes that are not matched, while ( $\Leftrightarrow$ ) represents the similarity matching of nodes that are included in the common similarity sub-trees. Note that in order to match only distinct sub-trees the overlapping resulting from considering one node in Figure 3.8 (a) (status: '*stands beside John's house*') should be eliminated. Thus, only one matching node (Action 2 or Object 2) would be considered as a match. The consideration for matching could be according to which nodes have higher similarity values as in this case. Also, passing a threshold should be a condition to consider two nodes to be similar. For example, Agent1 and Status were not considered for the matching, even though they are similar to some degree.

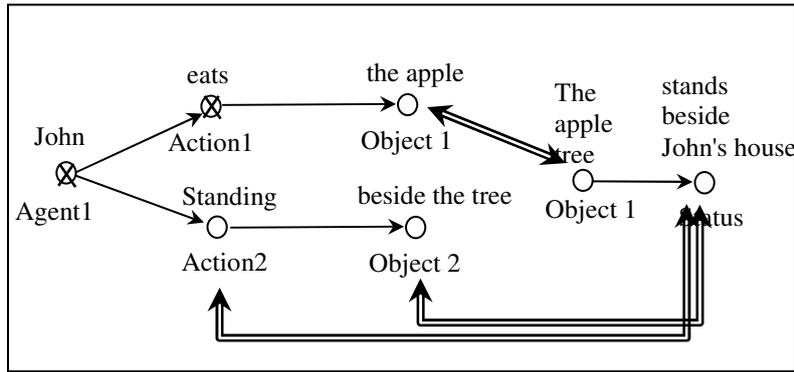


Figure 3.8 (a) Similarity Estimation for Sample Texts

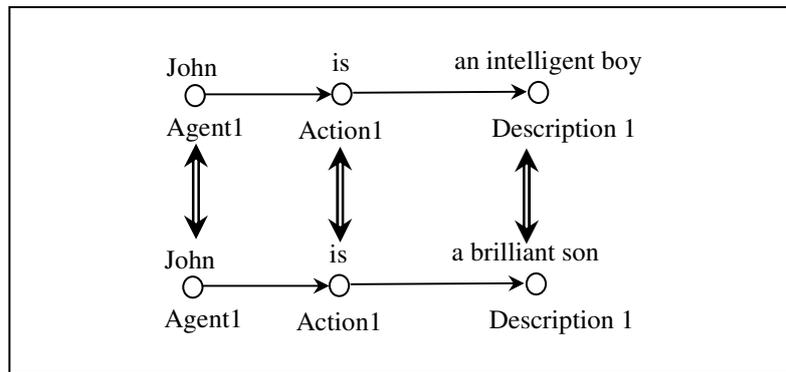


Figure 3.8 (b) Similarity Estimation for Sample Texts

Clearly, the summing up of similarities for the first pair of sentences is lower than the second pair, which reflects the closeness of their meaning distances. An index indicating the similarity between a pair of sentences (or documents) can be determined by normalizing the similarity of all found similarity sub-trees. Applying this on the sample sentences in figure 3.5 (a) and (b), assuming that the similarity values between the sub-trees are estimated through matching of nodes attributes will produce a low similarity for the first sentence pair and a high similarity for the second pair. More details on the proposed similarity measure are given in Chapter 5. The discussion includes formulating the measure as an inexact matching of attributed trees, showing the fulfillments of the measure to metric properties, and providing a polynomial-time algorithm to compute it.

---

### 3.5 Mining Processes

A document mining process is loosely defined as an activity that attempts to analyze, and discover knowledge from documents. Examples of these processes are document clustering, document classification, information retrieval, information filtering, and information extraction. These processes execute tasks such as categorizing documents, discerning relations or associations among a group of documents, retrieving relevant documents to queries, filtering documents for interested users and tasks, and extracting text and integrating it within a structured format. Since these processes differ in their specifications and requirements, they have been investigated, and tackled in various ways. This work is a step to enhance the performance of these processes by manipulating the documents semantically, and involve the semantic similarity measures outcome in the mining processes tasks.

At a certain level of abstraction these processes can be looked at as being relatively similar. They can all benefit from the semantic-based representation (i.e., the semantic graph model (SGM)), produced by a text parser, and can make use of the similarity measure estimator as needed. And then proceed with their specific tasks. The text analysis and the similarity estimation should be as much independent as possible of the specific mining process to allow for modularity and scalability. In Chapter 6, a case study of the proposed approach on one of these document mining processes (a semantic-based document clustering) is presented. Implementations and experimental work have been carried out and results are also presented and analysed.

---

### 3.6 Supplementary Components

What have been mentioned so far are the main components of the semantic-based document mining system. However, depending on the specific applications, other components may be included in the system, such as:

- Intelligent user interfaces that are able to process users' queries and assess their information needs. This component could play a major part along with the parser to understand users' requests. It would be very crucial if systems were to engage in dialogs with users.
- Document pre-processing to identify and clean tags, remove of non-textual objects, e.g., tables, graphs, images, etc.
- Header/footer, and sections titles information extractors that reads messages and put header/footer, and sections titles lines into slots for further analysis.
- Recognizers to determine punctuation marks (such as ":", and ","), numbers (either in numeric form or with the numbers spelled out in English), dates and times (recognized and converted to a canonical format), and others like phone numbers, social security numbers, electronic mail addresses, special identification numbers, etc. The main value in recognizing these special cases is that it prevents the parser from wasting time trying to interpret them.
- Automatic spelling corrector that can suggest alternative spellings to a misspelled words or to words that have slightly different spellings. This information can be added as an attribute in the SGM for later use, especially when matching different documents.
- Dictionary interface component: The parser can make use of a dictionary in the process of understanding and disambiguating words so it can correctly represent meanings of sentences.

- World knowledge base creator and expander: knowledge of the world is essential to correctly interpret text. It should be available to the parser to access and update.
- 

### **3.7 Conclusion**

This chapter has introduced the new approach to build a document mining system that is based on semantic understanding of text. An overview was given about the system architecture and its components and mechanisms. A discussion was given regarding parsing text to the introduced meaning representation model; the SGM, and how can similarity measures be performed effectively and efficiently. A more detailed treatment of the representation scheme, the similarity measure, and the implementation of the system are to be given in the following chapters.

## Semantic Graph Model (SGM): A Meaning Representation of Text



*"If we knew what it was we were doing,  
it would not be called research, would it?"*

Albert Einstein (1879 - 1955).

---

This chapter presents a formal semantic representation that captures the meaning of text and the algorithms for mapping from text documents to this meaning representation. The semantic representation, which can be viewed as expressing relations between concepts of text constituents, is believed to enhance the mining processes performance. The availability of domain independent natural language tools, which are exploited extensively, significantly contributes to the feasibility of this approach. The semantic graph model (SGM) is introduced to model how text constituents (i.e., phrases, words, and morphemes) are related. The process of creating the SGM includes syntactic and semantic analysis. Section 4.1, reviews other attempts to utilize meaning representations in document mining. Section 4.2 outlines the proposed representation model. Section 4.3 and 4.4 detail the steps taken to build the model including the syntactic and semantic analysis algorithms. Both the

nature of the meaning representation and the computational processes that can produce it are discussed. Concluding remarks are given in Section 4.5.

---

## 4.1 Document Representations

There is an increasing interest to enhance document representation modeling for document mining through means of semantic understanding to characterize the contents of text. Document mining research is interested in improving effectiveness of text representations beyond what is possible using keywords alone. There have been the uses of word sequences, term grouping, and phrases in representing documents. These approaches vary with respect to the amount of linguistic analysis in extracting useful representation from free text, ranging from essentially none to a full syntactic and semantic parsing. For instance, in information retrieval, matching the semantic content of queries with the semantic content of documents has been proposed [32]. Croft et. al. [20] suggested using rough parsing to identify sentences and then use them for indexing instead of single terms. The reason for using sentences and/or phrases as oppose to terms is that they carry greater semantic content. In the same direction, Rau and Jacobs [78] suggest grouping the keywords to achieve better precision/recall. Mauldin [67, 68] used a quick parser (FERRET) to create *case frames* that represented documents and used in information retrieval. Compared to a simple keyword system, the method improved the precision/recall performance, although it sometimes offered worse results. Salton et. al. [80] suggested using document vectors as a first stage filtering followed by a comparison of section, paragraph, and sentence vectors. Moreover, Hersh et al. [41] suggested

that adding some form of semantic processing to phrasal-based information retrieval may improve performance.

A text representation method based on exploiting the semantic structure is being introduced to improve documents mining effectiveness. It is believed that more accurate similarity assessments between documents are achieved when performed on the semantic structures. These semantic structures can identify relationships between phrases and thus more precisely represent the contents of the documents. Moreover, having such structures that are extendable and can be enriched with semantic information, can further improve the understanding process of contents. This type of processing shows considerable promise for meaning representation, and to enhance existing mining processes.

In what follows, the approach to the semantic representation is presented. The representation adds to the information available about a text by specifying the relationships that exist among the concepts in the text. The semantic processing is based on a full syntactic analysis and an extensive semantic reasoning, which together support the robust mapping of text constituents to the rich representation of meaning.

---

## **4.2 The Semantic Graph Model (SGM)**

The semantic graph model (SGM) is a representation scheme that is deemed to be suitable for document mining processes. The focus in this modeling process is on the ability to express distinct readings of sentences as distinct formulas that capture their intuitive structures and meanings. SGM represents information in a document sentence by sentence. Each sentence is converted to a directed acyclic graph that has concepts as its nodes and relations as links. Furthermore, the knowledge within a

document is expressed in concepts and their senses in the current context, predicate relations between concepts, and other aspects such as semantic roles, and semantic variations, are captured as concepts attributes. The sum up of all sentences representations makes the document representation model. This can be illustrated with the following simplified sentence example:

1. "John, who is the CEO of the company, has given Mike a raise."

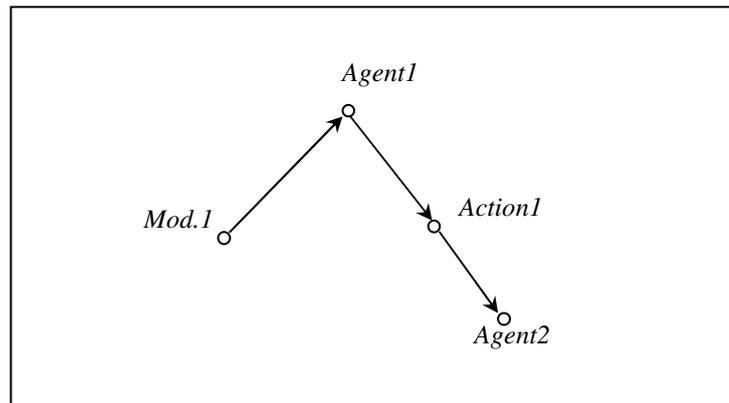
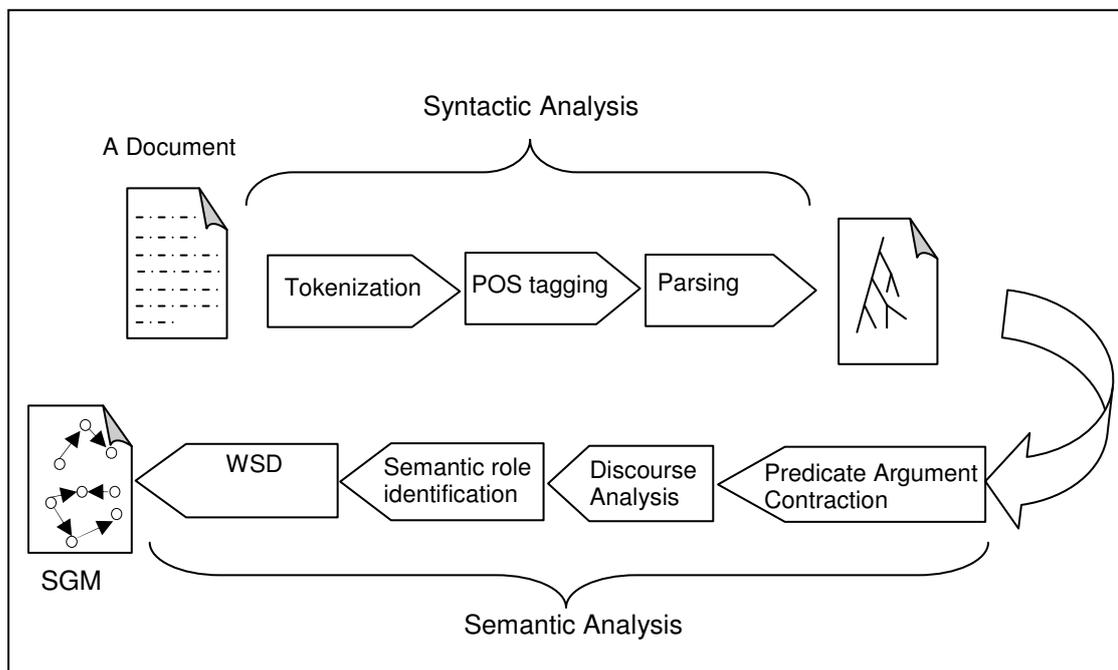


Figure 4.1 SGM of an Example Sentence

In the figure above, *agent1* and *agent2* denote the agent concepts of "John" and "Mike", respectively, *Mod.1* denotes a modifier to *Agent1*, and *Action1* to the action concept of "giving a raise". Clearly, the structure captures some semantic notions and relations found between concepts in the sentence. In addition, the nodes of this structure are attributed and can be augmented with symbolic values that express their syntax and semantic information specifics.

The process to create the SGM semantic representation includes syntactic and semantic analysis stages (Figure 4.2). The first step is to assign the input text to a syntactic parser. This step includes processes such as paragraph detection, tokenization, sentences splitting, morphological decomposition, part of speech tagging, noun/verb/preposition phrase chunking, parsing, and phrases head-words spotting. The semantic processing starts by constructing a predicate argument structure that determines how the elements discovered in the previous phase interact

within a particular linguistic structure. This predicate structure constitutes an acyclic graph model to represent sentences meanings. Further, the semantic process continues by adding different important attributes to the graph nodes. In addition to the attributes discovered through the syntactic analysis phase, some attributes are to be added through further semantic processing. This semantic processing includes, co-reference and anaphoric resolution, semantic role identification, and mapping the graph nodes to their conceptual variants found in an ontology through a word sense disambiguation (WSD) process. The sum of all sentences represented in the SGM characterizes the knowledge in a document. All graphs (essentially trees) of the produced sentences representations are connected to one node that makes a one inclusive rooted tree for the whole document.



**Figure 4.2** Flow Diagram of SGM Mapping Processing Components

In the following example, the syntactic analysis component of the system assigns the structure 2(b) to the input 2(a). The sentence maps to the SGM script shown in 2(c), which represents its semantic interpretation. The SGM provides both the concepts and their associated attributes, and specifies the relationships between

the concepts in the input sentence. For further illustrations, 2(d) gives a list of attributes of one node in the graph.

**2(a).** "*John drove his car yesterday.*"

**2(b).** (S (NP (HEAD *John*))  
(VP (HEAD *drove*)  
(NP *his* (HEAD *car*))  
(NP *yesterday*))  
)

**2(c).** Action1: Drive (Agent1: *John*,  
Object1: *Car*,  
TempMod1: *Yesterday*)

**2(d).** Name: Action1  
Type: Action  
Text: drove  
Syntax: verb (past)  
Synonyms: drive, motor  
Semantic: drive  
Relations: Agent1, Object1, TempMod1

---

### 4.3 Mapping Text to SGM

Mapping text to SGM involves combining the most effective aspects of the various approaches of NLP to semantically represent text in documents. The system first identifies text constituents and annotates them through the syntactic analysis. Such an approach supports the semantic analysis which can effect the accurate representation of meaning. The syntactic analyser includes components such as a tokeniser, a sentence splitter, a part of speech tagger, a morphological analyser, and a bottom-up/top-down parser. After a successful syntactic analysis step, tokens are tagged, noun/verb/prepositional phrases and their head-words are identified, and parse

trees are produced. The production of this syntactic structure, i.e., the parse tree with the identification of head-words of phrases, has significant consequences during the mapping of such phrases to concepts in the SGM. An example of the syntactic structure is given in 3(b) for the input in 3(a).

**3(a).** *"My friend John drove his car yesterday."*

**3(b).** (S  
           (NP (NP My (HEAD *friend*))  
                   (HEAD *John*)  
           )  
           (VP (HEAD *drove*)  
                   (NP *his* (HEAD *car*))  
                   (NP (HEAD *yesterday*))  
           )  
           .)

Note that the structure is fully specified; i.e., all attempts are made to identify the internal structures of the phrases. For example, the noun phrase *"My friend John"* is assigned a syntactic structure, i.e., a noun phrase, which reflects its final interpretation. At the same time, the fact that *"John"*, *"friend"*, *"car"*, and *"yesterday"* are labelled as heads has important consequences during subsequent semantic processing. In particular, the structural information provided by this analysis stage contains the needed information for building the predicate-argument structure. Furthermore, spotting of head-words is crucial in the process of mapping of phrases head-words to concepts in a lexical knowledge-base. The following subsections present a text analysis system and explain the computational processes that produce the predicate-argument structure and the concept mapping in SGM.

### 4.3.1 Text Analysis System Architecture

The design of the text analysis system is based on a multi-pass, and a multi-strategy architecture that can facilitate the building of robust and flexible parsers in an integrated and modular way. In this architecture (Figure 4.1), a single document or a group of documents are processed using a text analyser to populate two data structures, namely syntactic parse trees, and knowledge base (KB).

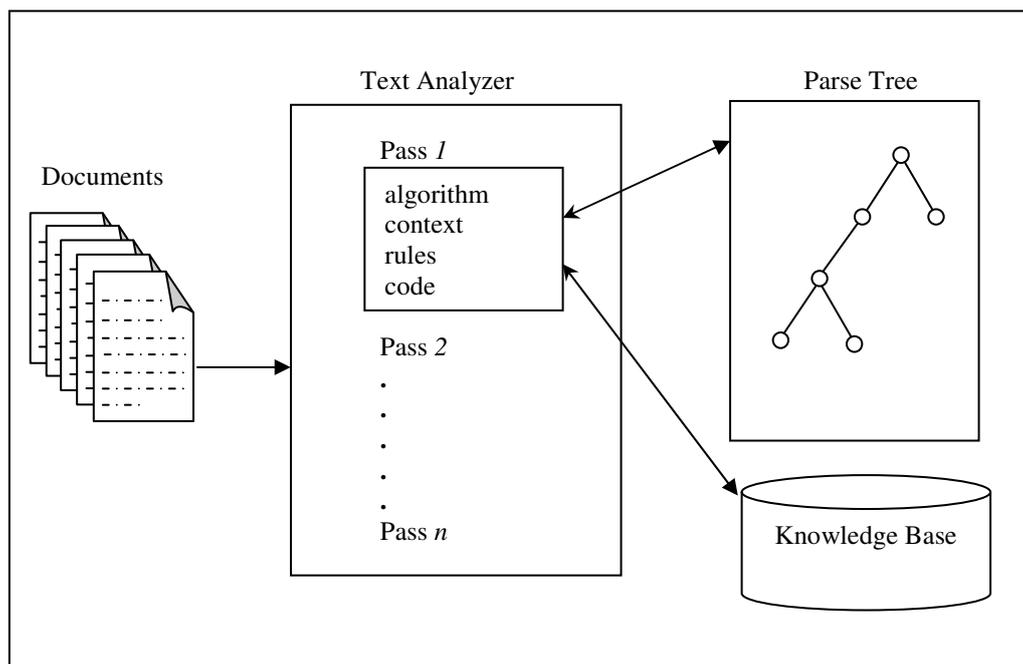


Figure 4.3 The Multi-pass Parser Architecture

The text analyzer comprises multiple passes, where each pass invokes its own processing algorithm and uses its own data as well as shared global data structures such as the parse tree and the data in the KB, as well as general programmatic data structures. Most of the passes algorithms are pattern matching-based passes algorithms, and some are recursive grammar algorithms. The passes are built upon each other to construct a single best-first parse tree and a semantic representation of the text in the KB. Using a multi-pass methodology enables gathering evidence, characterizing text, and even undoing tentative assignments when necessary. For example, a pass can rough out segments in the text and then, later in another pass,

reason about them and their surrounding context, in order to rework those segments with greater confidence. Also, a pattern matching-based pass can use an associated set of rules to recognize specific constructs in a text. This strategy primarily roughs out regions such as sentences, clauses, and uncharacterized segments, and recognizes high-confidence patterns and exploits surrounding context to aid in characterizing ambiguous constructs. The coding of the algorithms and data can be embedded in the passes or can be invoked from external resources. The passes can be generally segregated into four regions as follows: Tokenization, Part of Speech Tagging, Syntactic Parsing, Semantics and Discourse Analysis. Before elaborating more on these regions, a summary is given regarding some advantages of the architecture in the following points:

**1) Integration:** The architecture allows a high degree of coherence between its different components. Different strategies with varies levels of granularity can easily interact within the framework. Integration is mainly achieved through the placing of the input documents, parse tree, associated knowledge, and other global variables available to the different passes. Moreover, the integration of external resources can be easily adopted and implemented.

**2) Modularity:** This feature is illustrated through regarding the priorities existing between resources and functions and their relative independence. For example, passes can be elaborated such that each performs a simple and a modular task. Furthermore, passes can be arranged and put in the order that reflects their importance, while the key data elements are placed and managed separately.

**3) Extensibility:** Adding new features to a developed system can be easily achieved without disturbing any existing components. Elements other than the codes (e.g., the knowledge base) can also be expanded and edited independently.

**4) Maintainability:** The framework can be easily understood, adjusted, adapted and/or enhanced. Different specialized algorithms and codes can be replaced and upgraded without disrupting what has been already developed.

**5) Flexibility:** Changes to existing mechanisms and adoptions of new techniques can be accommodated. Passes or data elements can be added to the framework in any order. This is important as some initially glossed over or missing work can be added in at a later time.

**6) Reusability:** Passes within the architecture can dynamically create and execute other passes and codes. In addition, analyzers and sequences of passes can be configured for reuse as templates or as libraries.

**7) Feedback:** By allowing deferring, rather than making uninformed decisions upfront, passes can be implemented to gather information and make rough assessments, in order to increase confidence in the following processing decisions.

**8) Context:** By breaking the analyzer into multiple passes, each pass (or number of passes) can operate on specific contexts. For example, a syntactic parsing pass can work only within noun phrases or within the header region of the text being analyzed.

It is assumed that an input sentence would pass through the tokenization, part of speech tagging, syntactic parsing and semantic and discourse analysis. This is a pipeline-oriented approach that follows directly from the assumption. An input is first passed through a tokenizer to identify its basic elements, a POS tagger to assign syntactic class categories to words, a syntactic parser to derive its syntactic structure. The output of this analysis stage is then passed as input to a semantic analyzer to produce a meaning representation. Note that although the main intermediate syntactic structure is usually a parse tree, other syntactic representations and information can be used. The remainder of the discussion will assume parse tree-like inputs.

Before moving on, a major assumption about the role of ambiguity of this approach should be addressed. In this syntax-driven semantic approach, ambiguities can arise from the different analysis stages, and can lead to the creation of multiple ambiguous meaning representations. It is not the job of the semantic analyzer, narrowly defined, to resolve these ambiguities. Instead, it is the job of various passes within the different regions when having the knowledge of context to select the best among competing outputs produced.

In the following, the passes regions of tokenization, part of speech tagging, syntactic parsing and semantic and discourse analysis that make the text analyzer are explained:

#### **4.3.1.1 Tokenization**

This is often performed by the first few passes. They execute standard tokenization task that is, segmenting the characters of an input file to tokens or units of alphabetic, numeric, punctuation, white-space characters, etc. Other subsequent passes could also be concerned with some tokenization issues such as, when reasoning about whether two tokens should be joined or not is needed. The tokenization task is fairly straightforward; most punctuation is split from adjoining words and constructions are split into their component morphemes, so that each morpheme is tagged separately. For example, "*I'm*" would be tokenized to "*I*", and "*'m*". This tokenization allows for the further analysis of each component separately, thus, for "*I*" it can be in the subject noun phrase while "*'m*" is the head of the main verb phrase. There is also the tokenization of subtleties for hyphens, dashes, ellipsis, dots, etc.

Moreover, the tokenizer can perform parsing operations and other evidence-gathering on isolated lines of text. It characterizes individual lines and the relationships between them, in order to decide upon the best separation of sentences, paragraphs, headers, and other text regions using a finite state machine lexical analyzer with heuristic rules for finding the boundaries. For example, if a line ends with an English function word such as “the”, this adds evidence for the presence of a prose or sentential region of lines.

#### **4.3.1.2 Part of Speech Tagging**

The Part-of-speech (POS) tagger is responsible for assigning the possible syntax classes to words found in the document (e.g., as known, unknown, spelling errors). The POS tagger is distributed throughout a number of passes. Syntactically unambiguous words are tagged early on by utilizing a list of English words (a lexical lookup), and their possible syntax classes. Tagging of ambiguous words is deferred to passes dealing with clausal patterns, in order to utilize context to enable accurate POS tagging. Distinguishing between unknown and misspelled words can be done through setting a threshold of editing distances between the words and the lexicon.

The recognition process can also be extended to recognize known expressions phrases. Phrasal recognition can occur at various points to recognize relevant idioms and collocations. This can be built as a data element in the knowledge base to enable attaching sequences of words.

#### **4.3.1.3 Syntactic Parsing**

The primary goal of the syntactic parsing region is to recognize a sentence and assign a syntactic structure to it, namely the parse tree. Parse trees are directly useful

and an important intermediate stage of representation for further semantic analysis. The syntactic parsing passes include pattern-based or recursive grammar algorithms, segmenting resolutions, and chunking processing. Parsing algorithms depend on grammars that are declarative formalisms, and they can be computed in many possible ways. The segment resolution passes are interspersed with chunking and the syntactic parsing passes, so as to use feedback to assign segments and their boundaries with greater confidence.

The main and commonly used parsing algorithms are based on the context-free grammar parsing algorithm. Using a grammar and a lexicon, which consist of some of the language rules, the parsing algorithm will search through the space of all possible parse trees to find the correct parse tree for the sentence. Thus, different search metaphors can be utilized, such as top-down (starting with the root and growing trees down to the input words) and bottom-up (starting with the words and growing trees up toward the root). Moreover, the parser should be able to deal efficiently with the important problem of ambiguity; a sentence or words that can have more than one parse. The multi-pass architecture allows using insights from early filtering passes to efficiently handle ambiguous inputs.

Segments are nodes in the parse tree, and the algorithms to create them are primarily based on finding boundaries such as English function words ("*the*", "*is*", "*of*") and prose punctuations. Subsequent passes will then reason about the content and structure of isolated segments, as well as about the context surrounding them.

Chunking processing is essentially re-segmenting, for example to separate a verb phrase from the start of a noun phrase. The chunked text is represented using a parse tree containing tokens and chunks, where each chunk is a sub-tree containing

only tokens. For example, the chunk structure for base noun phrase in the sentence (4) is represented in 4(a):

4. *"I saw the smart boy in the library"*

4. (a) (S

(NP: <I>)

<saw>

(NP: <the> <smart> <boy>)

<in>

(NP: <the> <library>)

)

Chunking algorithms can use regular expressions over tags to chunk a text. Since actions deemed to be erroneous and they may be undone or redone, this is one mean for handling ambiguous constructs. Initial chunks could be first constructed then through applying a sequence of chunk rules to the chunked string, each of which modifies the chunking that it encodes. Chunk rules are transformational rules that update the chunking of a text by modifying its chunk structure. Each of these rules defines the applied method, which modifies the chunking encoded by a chunk structure. Chunk rules can be looked at as a modified version of regular expression patterns. Here, the patterns are used to match sequences of tags.

#### 4.3.1.4 Semantic and Discourse Analysis

The semantic and discourse analysis passes utilize the parse tree, and parse tree semantics (i.e., data layered into the parse tree nodes) in order to perform tasks such as anaphora resolution, semantic reasoning, and correlation of the concepts found in a text. As the syntactic analysis indicates dependencies among the text constituents, these dependencies can be used to represent predicate-argument structures, which are closely related to logical forms. To approximate semantic in

human languages, the predicate-argument structure asserts specific relationships held among the various concepts underlying the constituent words and phrases that make up sentences. In large, it is this underlying structure that allows the formation of a single composite semantic representation from the meanings of the different parts on an input text.

Meaning representation of the formal predicate-argument structures are well known and well specified syntactically and semantically. Other meaning structures do exist; including semantic network, conceptual dependency, and frame-based representations. Though these representation approaches are different, at an abstract level they all share as a common foundation the notion that a meaning representation consists of structures composed from a set of symbols. When appropriately arranged, these symbol structures are taken to correspond to objects, and relations among objects found in some linguistic inputs describing the state of affairs in some world. The approach will be to adopt the foundation of predicate-argument structures; a declarative, compositional semantics that is context-independent and unambiguous, and build upon it a more expressive representation, borrowing ideas from natural language processing while avoiding its drawbacks.

To create predicate argument structures of sentences, rules are utilized that are written to serve the mapping of syntactic generality. In this step, the semantic analysis is integrated into the passes that are responsible for the semantic, and discourse analysis. The rules involve deciding, according to the syntactic attributes of the words, whether to be combined into a single node, or to set a relation between them. The first step in creating a predicate-argument structure is to always attempt to locate the main predicate of the sentence. For example, if there is a proper verb (not an auxiliary or modal verb) and a noun (an object), then the verb is the main predicate

and the noun is one of its arguments. Similarly, if there is an auxiliary verb such as "is" and a noun, then the noun will be the main predicate. This is a syntax-driven semantic analysis, as it depends on the view that verbs impose certain restrictions on the number, grammatical category, and location of the phrases that are expected to accompany them in syntactic structures. To briefly illustrate this idea, consider the following example sentences:

5. "I like Japanese cars."

6. "I like to spend less than five thousand dollars."

7. "I like it to be sport and luxurious."

After syntactic parsing, these sentences can be classified as having the following syntactic parse trees:

5(a). (S

(NP: <I>)

<like>

(NP: <Japanese cars>)

)

6(a). (S

(NP: <I>)

<like>

(Inf-VP: <to spend less than five thousand dollars>)

)

7(a). (S

(NP: <I>)

<like>

(NP: <it>)

(Inf-VP: <to be sport and luxurious>)

)

These parse trees specify the number, position and syntactic category of the arguments that are expected to accompany a verb. For instance, the parse tree for sentence (5) specifies the following facts with regard to the verb *like*:

- This predicate has two arguments.
- Both arguments must be NPs.
- The first argument is pre-verbal and plays the role of the subject.
- The second argument is post-verbal and plays the role of the direct object.

Thus, the predicate-argument structure of the sentence can be formulated as:

**5(b).** *Like(I, JapaneseCar)*

Note that verbs are not the only entities in a syntax that can carry a predicate-argument structure. Prepositions can also be characterized as two-argument predicates if, for example, the first argument is an object that is being placed in some relation to the second argument. This can be seen in the following phrases; (8) and (9):

**8.** *"a Japanese car under five thousand dollars."*

In this phrase the predicate-argument representation associated with the preposition *under* can be expressed in the following structure:

**8(a).** *Under(JapaneseCar, \$5000)*

Another example of the non-verb based predicate-argument structure is illustrated in the following sentence:

**9.** *"Japanese cars are inexpensive, economical, well-made, and well-shaped."*

Here, the predicate argument structure is based on the concept underlying the noun *Japanese cars*. This sentence can be represented in a four argument predicate structure as the following:

**9(a).** *JapaneseCars(Inexpensive, Economical, Well-made, Well-shaped)*

Up to this stage, the meanings assigned to inputs are based only on static knowledge from the lexicon and the grammar, i.e., literal meanings that are context independent and inference-free. The information provided by the predicate-argument structure is quite valuable in capturing a variety of important facts about text constituents. However, further analysis of the semantic information associated with these structures, can bring considerable insights into the meaning representation. This can be done through considering extending these structures in the semantic realm: semantic roles, semantic reasoning, enriching, and discourse analysis. In subsequent stages, these structures will serve as bases to produce richer and useful meaning representations.

After construction the predicate-arguments structure, the arguments (i.e., the nodes) in this structure are labelled with semantic roles (such as *agents*, *objects*, *states*, *actions*, *events*, and *locations*) in order to more clearly specify the relationships among the concepts represented. The notion of a semantic role can be understood by looking at the similarities among the arguments in example sentences (5) through (7). In each of these cases, the pre-verbal argument always plays the role of the entity doing the *liking*, while the post-verbal argument always plays the role of the concept that is *liked*. By noticing these regularities and labelling them accordingly, the surface arguments of a verb with a set of discrete roles in its underlying semantic can associated. Rules can be utilized that depend crucially on the syntactic types obtained from the syntactic analysis phase. For example, the constructed SGM given in 1(c) to represent the semantic interpretation of the sentence "*John drove his car yesterday*". The case labels on the arguments indicate that it is an *agent* (John) performed the *action* (drive) on the *object* (car).

After all heads of phrases have been identified, they are mapped to concepts in a domain independent lexical knowledge-base (or ontology), e.g., the WordNet [71], using a comprehensive word sense disambiguation algorithm. The algorithm measures the semantic relatedness between words found in a context and their correct senses in WordNet. This can be done in a similar way to what has been introduced by Lesk [58] where all the sense definitions of the word to be disambiguated are retrieved from the ontology. Each of these senses is then compared to the ontology definitions of all the remaining words in the context. The sense with the highest overlap with these context words is chosen as the correct sense. Nevertheless, there are other robust algorithms that can achieve high levels of accuracy given certain assumptions. The framework allows seamlessly the substitution and the integration of different algorithms to perform WSD and other tasks as well.

Furthermore, extensive word-variant generation to enrich the representation is employed. Variant generation is determined by the information available for the disambiguated words in WordNet. Variants are obtained by retrieving the available morphological alternatives, synonyms, acronyms and abbreviations for each head word in the input sentence. Such information will be added as attributes of the structure nodes. For example, all senses and variants found in WordNet for the noun "*car*" are listed in (8).

**10.**

- car, auto, automobile, machine, motorcar (a motor vehicle with four wheels; usually propelled by an internal combustion engine) "he needs a car to get to work."
- car, railcar, railway car, railroad car (a wheeled vehicle adapted to the rails of railroad) "three cars had jumped the rails".
- cable car, car (a conveyance for passengers or freight on a cable railway) "they took a cable car to the top of the mountain".

- car, gondola (the compartment that is suspended from an airship and that carries personnel and the cargo and the power plant)
- car, elevator car (where passengers ride up and down) "the car was on the top floor"

After a successful disambiguation of the right sense used in the context for the word, a straightforward retrieval of its variations can be done. This variant generation process makes the representation richer in information, a character that would be beneficial especially when computing similarities between different structures.

Discourse analysis passes are responsible to explicitly resolve co-references (more than one expression refer to the same entity) and anaphora (the use of a linguistic unit, such as a pronoun, to refer back to another unit) according to the surrounding context. This is an important step in understanding contents and it will improve the process of measuring similarities between documents. The result of this process would be added to the resolved words attributes in the SGM. To date, there have not been full-fledged algorithms that can account for all the variety of ways that natural languages provide to refer to entities. However, a number of mature and reasonable algorithms exist and can be utilized accordingly.

Further analysis steps for understanding can also be integrated in the SGM. Pragmatic analysis, for instance, is an interesting aspect to include, but yet hard to accomplish. In fact, the so-far developed presentation of meaning could run into troubles fairly quickly when real language is examined, particularly, when the meaning of a constituent is not based on the meaning of its parts. This phenomenon is prevalent in many phrases and idiomatic constructions. For example, in the phrase "*It is raining cats and dogs*" clearly the concept of rain does not have much to do with cats or dogs. Instead, it roughly means something like "*there is a rain storm*". The most straightforward way to handle idiomatic constructions like these is to introduce

new grammar rules specifically designed to handle them. These idiomatic rules mix lexical items with grammatical constituents, and introduce semantic content that is not derived from any of its parts. These new and more precise concepts can then compose the meaning structures and thus can be handled as before. This is obviously a simplified look at non compositional constructs. Idioms and proverbs are frequent and productive phenomenon that can pose serious difficulties to be generally recognized.

---

#### **4.4 SGM Desiderata**

There are some basic requirements and desirable properties to be fulfilled when designing meaning representations. This section examines the SGM applicability and advantages as a document knowledge representation. Of particular interest are those properties that facilitate assessing similarities and mining of documents. The discussion includes the *verifiability*, *preciseness*, *compactness*, *expressiveness*, *extendibility*, and *tractability* properties.

**1) Verifiability:** This is a very basic requirement for the meaning representation to satisfy. It is the ability to relate the input texts to the world knowledge, so the comparison (or matching) can be performed between the knowledge described by these representations themselves, and between them and the state of the world as modeled in a knowledge base. For example, the representation of sentence (9) can be matched against other similar statements (e.g., "*American cars are inexpensive, luxurious, and fast.*") or against the knowledge base of facts about cars. An affirmative, or a negative answer can be drawn to verify or compare, and a justification can be also provided (in the case of partial matching, or incompleteness

of the knowledge base). The similarity measure that to be detailed in Chapter 5, can be used to verify representations through matching and to what degree is the matching.

**2) Preciseness:** This is to require, regardless of the ambiguity in an input sentence, the representation scheme should produce a single (or a best first) interpretation free from any ambiguity. The representation should allow reasoning and determining meanings of inputs in contexts they are in, which can require the use of intermediate representations that maintain some level of ambiguity. The SGM provides the structure for the disambiguation mechanisms to infer and apply their findings by altering the structure and editing attributes. Moreover, the scheme supports representing uncertainty and vagueness found in human expression, by allowing attaching the parsers confident levels when arriving at their interpretations.

**3) Compactness:** The phenomenon of variability in NL is prevailing. An idea can be expressed in many ways by using different words and/or syntax (see Section 3.3 for examples). Thus, representation models should be able to capture that and compact multiple inputs that mean the same in a context into one form. This can be accomplished using the SGM, as it essentially transforms sentences to a predicate-argument structures that are canonical forms, and allocate the means to perform word sense disambiguation, and word variation retrieval (see Section 4.3.1.4 for details).

**4) Expressiveness:** It is desirable to have a meaning representation scheme that can adequately handle a wide range of subject matters, and present meanings of any sensible inputs. SGM is expressive enough to handle quite a lot of what needs to be represented. SGM does not require many specific obligations as to how things should be represented. The representation consists mainly of concepts, properties of concepts, and relations among concepts.

**5) Extendibility:** Text analysis procedures are many and are expected to grow to meet the needs to fully understand meanings. A good representation scheme should be able to cope with that and provide the means for extensions. The SGM is flexible, easy-to-understand, and open to include various techniques of analysis and reasoning processes outputs.

**6) Tractability:** The SGM is a computationally tractable approach to the representation of knowledge that satisfies many of the requirements raised above. Specifically, it provides a sound computation basis for the verifiability, expressiveness, and extendibility requirements. However, the most attractive feature of SGM is its structural model. SGM are trees that can be manipulated and managed efficiently. This is especially vital, as the aim from building the SGM is to use them for further mining processing.

---

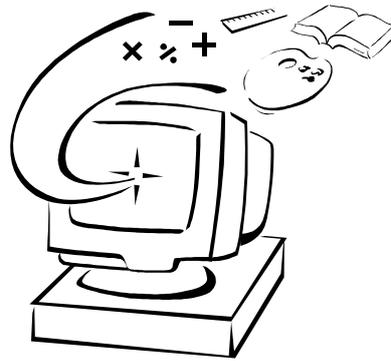
## 4.5 Conclusion

This chapter has introduced the SGM representational approach to meaning and steps to build it. The approach is based on exploiting semantic structures to improve documents mining effectiveness. The representation translates readings of sentences to formulas that encapsulate their structures and semantics. It identifies relationships between phrases and thus more precisely represents the content of text. SGM for a document is the accumulation of all sentences parsed into their meaning representations. Essentially, each sentence representation is a directed acyclic attributed graph of concepts and relations between them. The process to build the presentation starts by identifying predicate-argument structures, and then

disambiguating and making implicit meanings more explicit and embedding them in the structure.

SGM representation can produce unambiguous and canonical interpretations for input sentences that can be ambiguous and verbose. The model produces single meaning representation structures that adequately represent the meaning of sensible inputs. Using SGM representation enables comparison between modeled documents. Moreover, the advantages of this representation scheme are its information richness, and openness to include various analysis and reasoning processes output. More importantly, is the ability of the structural model to be manipulated and managed easily. This is especially vital, as the aim from building the SGM is to use them for further mining process efficiently and effectively. The following chapter addresses the problem of comparing SGM representations and proposes a new similarity measure. The proposed measure is defined on the SGMs that are in essence tree structures with multiple symbolic attributes, and is based on finding common subtrees.

## Similarity Measure



*"There is measure in all things."*

Horace (65BC - 8BC).

---

As the semantic graph model (SGM) to represent text was proposed, the problem of comparing documents converted to SGM representations is being tackled and a new similarity measure is being proposed. This similarity measure is centered on the notion of finding all distinct similarity common sub-trees between trees. The measure is general and defined on tree structures that are augmented with attributes. This chapter explains the proposed similarity measure based on inexact or approximate graph matching. First a review is given regarding existing graph matching techniques in Section 5.1, focusing on the inexact graph matching problem and showing the increasing interest to solve it. Section 5.2 presents some notations and terminologies used to formulate the problem, and to define the measure. Section 5.3 addressed the problem of measuring distances between SGM representations of

text and shows that the novel similarity measure is pseudo metric. Section 5.4 introduces a polynomial-time algorithm for computing the similarity measure. A small test of the effectiveness of the similarity measure is illustrated in Section 5.5, and the chapter ends with conclusions in Section 5.6.

---

## 5.1 Graph Matching

In order to express sentences as formulas that capture their intuitive structures and meanings, a text representation scheme called the semantic graph model (SGM) is introduced. To build this semantic representation, the text goes through several syntactic and semantic analysis stages. The result of these analysis steps is predicate-argument structures that can express the interaction of text constituents within particular linguistic structures. The predicate-argument structure constitutes an acyclic attributed graph structure. Thus, using these structures, the similarity estimation problem of text is transformed into a graph matching problem.

Graph-based representations have been extensively used in diverse areas such as computer vision, pattern recognition, chemistry, molecular biology, and computational linguistics. Graphs are effective and flexible abstract structures that are used to represent objects in images, substances structures, and relations between text constituents. Examples include using graphs to represent shape-skeletons [84], the use of graphs to represent 3D objects [27], and the use of trees to represent grammar parses [4]. The success in using these structures is often attributed to the attractive feature of graph representations as they can concisely describe the relational arrangement of objects and their components.

The problem of similarity (or distance) measure between objects represented as graph abstractions has been widely researched. The traditional approach to graph matching is through edit-distance [31, 106]. This approach works by transforming one graph to another through a sequence of basic edit operations. These operations have costs associated with them. The operations include deletion, insertion, and modification of nodes or edges attributes. The associated costs could vary according to the edit operation, positions of nodes and/or information they contain. The distance is calculated by finding the minimum accumulation of the editing costs that would produce a mapping between the two graphs. Determining the costs of the edit operations can be problematic and application domain dependent. Thus, two graphs may be similar using one cost function and dissimilar using another. Moreover, in general, computing the edit-distance is proven to be an NP-hard problem [107]. Recently, Bunke and Shearer [16] introduced a metric distance measure on un-attributed graphs based on identifying the maximum common sub-graph. Wallis et al. [100] presented a distance measure based on the minimum common super-graph. And, Fernandez and Valiente [36] defined a metric based on the difference in size between maximum common sub-graph and minimum common super-graph. More recently, Hidovic' and Pelillo [42] extended these metric measures to include the case of attributed graphs. Nonetheless, all these methods are guaranteed to find the optimal solution, they are computationally prohibitive. They depend on computing the maximum common sub-graph, which is shown in [15] to be computationally equivalent to the edit-distance, i.e., NP-hard. Therefore, many suboptimal, or approximate solutions have been also investigated using probabilistic relaxation schemes [19, 103], neural networks [34, 91], genetic algorithms [21, 101], and Tabu

search [102]. These approximate methods are computationally attractive, but may fail to find the optimal solution and get trapped in local minima.

In the case of SGM, the graphs at hand have special characteristics of being directed, connected, and acyclic, i.e., they are trees. The nodes of these trees have multiple symbolic attributes. Most similarity measures on trees found in the literature are based on the edit-distance. Zhang and Shasha [106] have used edit-distance on ordered trees, which preserve the order of relation between neighbour nodes in a rooted tree. They provided for this constrained tree matching problem a polynomial time algorithm to solve it. However, in the general case, the problem has been proven to be NP-complete [107, 108]. Recently, Valiente [97] extended the general case of graph metric introduced by Bunke and Shearer [16], and applied a bottom-up distance measure between trees. The measure is based on finding the maximum common sub-tree, and works efficiently both on ordered and unordered trees, but it is limited to rooted and un-attributed trees and not proven to be metric. More recently, Torsello et al. [94, 95, 96] introduced a computationally attractive normalized metric distance measures for attributed-trees that are based on computing the maximum similarity common sub-tree. These latest developments are extended and a new similarity measure is introduced. The approach is based on computing all distinct similarity common sub-trees, and should be able to measure similarity between SGM representations of documents that are trees with multiple symbolic attributes. It will be shown that the proposed measure satisfies some metric axioms, and can be computed in a polynomial-time.

---

## 5.2 Notations and Terminologies

A graph  $G = (N, E)$  is a data structure composed of a finite set of *nodes* (also known as *vertices* or *points*),  $N = \{n_1, n_2, \dots, n_m\}$ , and a set of undirected *edges* (also called *lines* or *links*),  $E \subset N \times N$ . The *order* (or the *size*) of a graph  $G$  is defined as the number of nodes in  $G$ , denoted as  $|N|$ , and the number of edges as  $|E|$ .

Two nodes  $x, y \in N$ , are said to be *adjacent* (or *neighbours*) if they are connected by an edge  $e \in E$  and denoted as  $e = (x, y)$ . Given a subset of nodes  $C \subseteq N$ , the *sub-graph*  $G[C]$  is the graph having  $C$  as its node set, and any two nodes are adjacent in  $G[C]$  if and only if they are adjacent in  $G$ .

When the edges have no directions, they are said to be undirected links, and a graph  $G$  containing only this types of links is called *undirected graph*. When the graph is directed, edges  $(x, y)$  and  $(y, x)$  are distinguished and usually called *arcs*.

A path between two nodes  $x, y \in N$  is a non-empty sequence of  $k$  distinct nodes  $\langle n_1, n_2, \dots, n_k \rangle$  where  $x = n_1$ , and  $y = n_k$  and  $(n_{i-1}, n_i) \in E, i = 1, 2, \dots, k$ . If there is  $(n_1, n_k)$ , the path is called a *cycle*. A graph  $G$  is said to be *acyclic* when there are no cycles between its edges, independently of whether the graph  $G$  is directed or undirected.

A *connected* graph is a graph that has at least one path between any two nodes. Connected graphs that have no cycles are called *trees*. A *rooted* tree is a hierarchical structure with a special node that can be identified as the root. Given two nodes  $x, y \in N$  in a rooted tree,  $x$  is an *ancestor* of  $y$  (and  $y$  is a *descendent* of  $x$ ) if the path from the

root node to  $x$  is a sub-path of the path from the root to  $y$ . Furthermore, if there is  $e = (x, y)$ ,  $x$  is said to be the *parent* of  $y$ , and  $y$  is said to be a *child* of  $x$ .

Graph nodes and edges can contain information attached to them. When this information is a simple label (e.g. a name or a number) the graph is called a *labelled* graph, and if nodes or edges hold more information (symbolic and/or numeric values) the graph is called an *attributed* graph. More often, this concept is further specified by distinguishing between *node-attributed* and *edge-attributed* graphs.

In this dissertation the concern is mainly on measuring similarity between node-attributed trees as the SGM renders to this type of graph. That is, given two graphs –a *pattern* graph  $G_P$  and a *target* graph  $G_T$ – what is the procedure of comparing them to check how similar they are. Generally, the graph matching problem can be stated in different ways as follows: if we are given two graphs  $G_P = (N_P, E_P)$ , and  $G_T = (N_T, E_T)$ , with  $|N_P| = |N_T|$ , the problem could be defined to find a one-to-one mapping (bijection)  $f: N_T \rightarrow N_P$  such that  $(x, y) \in E_T$  iff  $(f(x), f(y)) \in E_P$ . When such a mapping  $f$  exists, it is called an *isomorphism*, and  $G_T$  is said to be isomorphic to  $G_P$ . This type of problems is known to be the *exact graph matching*. An important sub-type of this matching problem is the sub-graph matching problem, in which we have two graphs  $G = (N, E)$  and  $G' = (N', E')$ , where  $N' \subseteq N$  and  $E' \subseteq E$ , and in this case the aim is to find a mapping  $f: N' \rightarrow N$  such that  $(x, y) \in E'$  iff  $(f(x), f(y)) \in E$ . When such a mapping exists, it is called the *exact sub-graph matching* or *sub-graph isomorphism*.

In cases where there is no such isomorphism to be expected between the two graphs, the graph matching algorithm should not only search for the exact way of matching nodes of a graph (or sub-graph) with nodes of the other, but it tries to find the best matching between them. This leads to the class of *inexact graph matching*

problems. In this case, the matching aims at finding isomorphic or similar sub-graphs between a pattern graph and a target graph. This can be due to the differences in number of vertices and edges in both the pattern and target graphs, or due to differences between the attribute values of nodes or edges. In this thesis both types of problems are considered when referring to inexact graph matching. Moreover, exact and inexact graph matching are the terms used in this thesis to differentiate between these two basic types of graph matching problems. However, in the literature these types of graph matching problems are also called *isomorphic* and *homomorphic* graph matching problems, respectively.

A *distance* function is a function that produces a value that is assigned to a pair of points in a space which indicates how far those points are from one another. A distance function is called a *metric* if it is always positive, symmetric, yields zero when measuring the distance from any point to itself, and permits no short-cuts. A metric distance measure is a function that establishes an order over objects in a set, and it is deemed to be an important property when comparing, searching and indexing objects in databases. Formally, a metric distance function  $d$  is defined over a non-empty set of domain objects  $\Omega$  such that  $d: \Omega \times \Omega \rightarrow \mathfrak{R}^+$ , and satisfies the following axioms:

$\forall a, b, c \in \Omega$ :

- 1)  $d(a, a) = 0$  (Minimality)
- 2)  $d(a, b) = d(b, a)$  (Symmetry)
- 3)  $d(a, b) = 0 \Leftrightarrow a = b$  (Identity and Uniqueness)
- 4)  $d(a, c) \leq d(a, b) + d(b, c)$  (Triangular Inequality)

Furthermore, if the function satisfies  $d(a, b) \leq 1$ , it is known to be a *normalized* metric distance.

A *similarity* measure is the reverse of a distance function. Similarity functions take a pair of points and return a large similarity value for nearby points, and a small similarity value for distant points, i.e., similarity is reciprocal of distance. Moreover, if  $d: \Omega \times \Omega \rightarrow \mathfrak{R}^+$ , is a normalized metric distance, then the similarity function derived from  $d$ , defined as  $S(a, b) = 1 - d(a, b)$ , holds the following properties:

- 1)  $0 \leq S(a, a) \leq 1$  (Bounded)
- 2)  $S(a, b) = S(b, a)$  (Symmetry)
- 3)  $S(a, b) = 1 \Leftrightarrow a = b$  (Identity and Uniqueness)
- 4)  $S(a, c) \geq S(a, b) + S(b, c) - 1$  (Triangular Inequality)

These definitions and properties will be instrumental to illustrate the development of the similarity measure, and to prove some of its characteristics in the following sections.

### 5.3 The New Similarity Measure

This section defines a new measure for matching SGM representations that are node-attributed trees. The measure based on finding all distinct common similarity sub-trees. The two miniature trees in Figure 5.1 are used to illustrate the working of the measure. First, to assess similarity between nodes as a comparison of attributes, the function  $Node-Sim(n_1, n_2)$  is used. It allows evaluating the degree of similarity between the graph nodes  $n_1$ , and  $n_2$  by answering the question: 'What is the degree of matching between the set of attributes,  $A$ , defining node  $n_1$ , and the attributes,  $B$ , defining node  $n_2$ ?', thus, the similarity function is valued in the continuous interval  $[0..1]$ . The value 1 means a complete match of attribute values in the two nodes. As

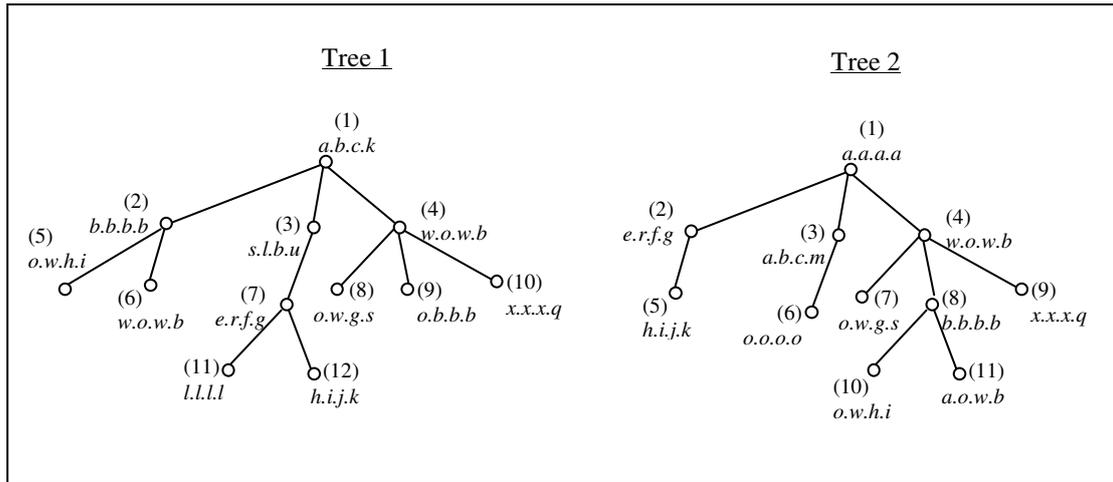
nodes are composed of different sets of pairs of attributes and values, each attribute,  $i$ , is associated with a numerical weight,  $w$  (a positive number), expressing its degree of importance. The node similarity function is formulated as follows:

$$Node-Sim(n_1, n_2) = \frac{\sum_i^{A \cap B} w_i \text{ Att-similarity}_i(n_1, n_2)}{\sum_i^{A \cup B} w_i}, \quad (1)$$

where  $Att-similarity_i(n_1, n_2)$  is the degree of matching for the attribute  $i$  in both nodes  $n_1$ , and  $n_2$ .

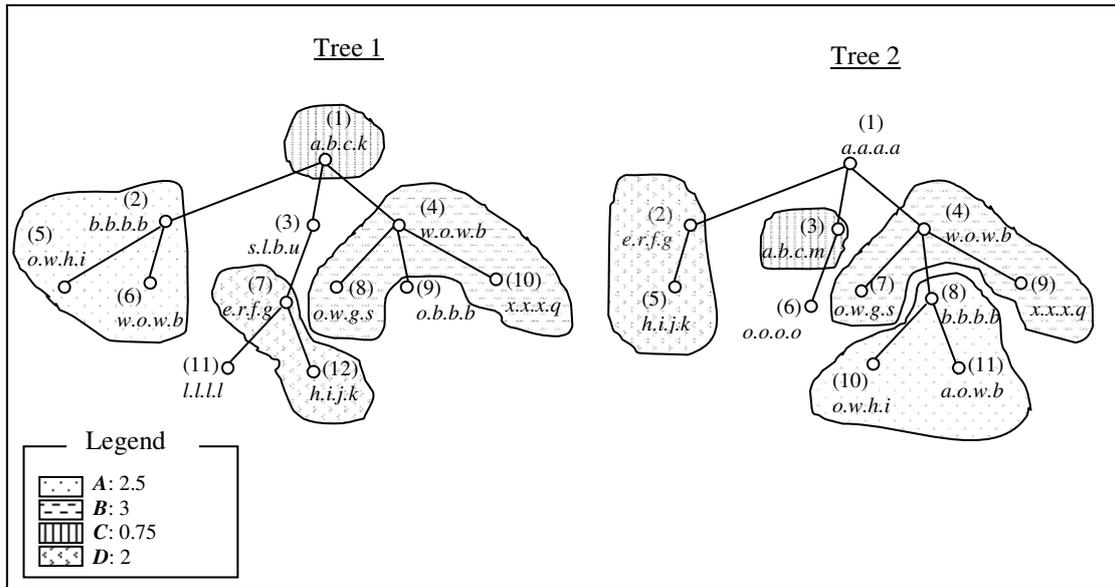
There are two general types of attribute values to be distinguished: values that are singular (such as syntactic tags, or semantic roles attributes of an SGM), and the values that define multiple data elements (such as synonyms, and antonyms). Accordingly, in the singular value case the matching is binary, and it is partial in the multiple values case. In the example trees, the attribute values are denoted with alphabet symbols  $\{a, b, c, \dots\}$ ; each node has a maximum of four single valued attributes. Thus, evaluating the similarity between node (1) in Tree 1, and node (3) in Tree 2, using the above equation and assuming equal weights,  $w_i$ , to all attributes, will yield a value of 75% match.

It is clear that this similarity function is symmetrical, i.e.,  $Node-Sim(n_1, n_2) = Node-Sim(n_2, n_1)$ . Furthermore, the function can be shown that it is a normalized metric as it fulfills the other metric properties, i.e., minimality, identity and triangular inequality. Fulfilling these properties is important, as it will be used to show how metric the proposed similarity measure is.



**Figure 5.1** Miniature Illustration Trees

As the tree structures are used to represent text, the approach to measure similarities between documents has turned into the problem of computing similarity between trees (specifically, inexact tree matching). The solution presented is based on measuring the commonality between trees through finding all distinct common similarity sub-trees. Clearly, the more similar the trees are, the more in number and larger in sizes their common sub-trees will be. Formally, the objective of finding a common similarity sub-tree between two attributed trees is to find an isomorphism which matches nodes having similar attributes. Let  $T_1 = (N_1, E_1)$  and  $T_2 = (N_2, E_2)$  be two trees. A bijection  $f: V_1 \rightarrow V_2$ , with  $V_1 \subseteq N_1$ , and  $V_2 \subseteq N_2$ , such that  $(x, y) \in E_2$  iff  $(f(x), f(y)) \in E_1$  (i.e., it preserves the nodes adjacency and the tree connectedness) is called a sub-tree isomorphism. When  $Node-Sim(n_1, n_2)$  is used as the similarity measure threshold in determining the matching between nodes, the matched trees are called common similarity sub-trees isomorphism. For example, if the similarity threshold is set to 50% match, the four pair of sub-trees matching between Tree 1 and Tree 2 (as marked in Figure 5.2) could be found:



**Figure 5.2** Common Similarity Sub-trees

A similarity index produced by the isomorphism  $f$  can be calculated as follows:

$$S_{1,2}(f) = \sum_{x \in V_1} \text{Node} - \text{Sim}(x, f(x)) \quad (2)$$

There can be many disjoint sub-trees isomorphism,  $i = 1, 2, \dots, l$ , between  $T_1$  and  $T_2$  that satisfy the above conditions. There are four pairs of these sub-trees in Figure 5.2 denoted by  $A$ ,  $B$ ,  $C$ , and  $D$  patterns along with their similarity index values (See Figure 5.2 Legend). When  $S(f)_i$  is the largest similarity index among all existing similarity sub-trees isomorphism, the similarity  $S$  of the isomorphism  $f$  is called the maximum similarity sub-tree isomorphism, and it is  $B$  with 3 similarity index in the example. It is called the minimum similarity super-tree isomorphism if  $S(f)_i$  is the smallest; in our case it is  $C$  that has 0.75 similarity value. The approach, however, is to consider all distinct similarity sub-trees isomorphism, i.e., all common similarity sub-trees that have no overlapping between them. All the sub-trees are taken into account, rather than just the maximum or minimum, as the GSM of documents could be quite large and matching sub-trees could be sparse. Thus, considering all sub-trees

will make the measure reflecting more commonality that can not be accomplished otherwise. Furthermore, the sub-trees overlapping (i.e., considering matching nodes more than once) should be eliminated to reduce overestimating similarities. It can be decided on a greedy manner when there is a node that belongs to more than one similarity sub-tree. Nodes should be included in the sub-trees matching that pass the threshold and have the highest matching. For example, the case of node (9) in Tree 1 and (8) in Tree 2 that has 75% attribute similarity illustrates how the conflict is dealt with. The nodes pass the similarity threshold to be included in  $B$  common similarity sub-tree isomorphism, but node (8) has higher attribute similarity (100%) with node (2) in Tree 1. This is solved by picking node (8) to be included in  $A$  common similarity sub-tree isomorphism instead of  $B$ . In order to be practicable, the algorithm that calculates the similarity measure should be developed in a way that would keep the overlapping eliminated and would have a low time and space complexity. Section 5.5 illustrates how this non-overlapping similarity sub-trees isomorphism can be computed in a polynomial-time.

After finding all distinct common similarity sub-trees, the overall normalized distance measure between  $T_1$  and  $T_2$  can be computed as follows:

$$d(T_1, T_2) = 1 - \frac{W(S_{1,2})}{\min(|T_1|, |T_2|)} \quad (3)$$

where  $W(S_{1,2}) = \sum_{i=1}^l S_{1,2}(f)_i$ ; the overall similarity of sub-trees paired by  $f$ , as defined in (2). Note that when eliminating overlapping sub-trees, we have

$$W(S_{1,2}) \leq \min(|T_1|, |T_2|) \quad (4),$$

thus  $d$  is normalized. In the illustrative example  $W(S_{1,2}) = 8.25$ , thus the distance between the two trees,  $d(\text{Tree 1}, \text{Tree 2}) = 0.25$ , i.e., a similarity of 75%.

Furthermore, with this assumption, we have

$$T_1 = T_2 \Leftrightarrow |T_1| = |T_2| = W(S_{1,2}) \quad (5)$$


---

## 5.4 Studying the Metricity of the Measure

As the developed similarity measure is to be utilized in document mining processes, such as clustering, classification, and retrieval, it is essential to establish that the measure is metric. A metric measure ascertains the order over objects in a set, hence operations such as comparing, searching and indexing of the objects can be performed. In the following prove that  $d$  is a normalized pseudo metric distance measure is given as it fulfills all the required properties:

$$1) 0 \leq d(T_1, T_2) \leq 1 \quad (\text{Minimality})$$

We have  $0 \leq W(S_{1,2}) \leq \min(|T_1|, |T_2|)$ .

$$\text{Thus, } 0 \leq d(T_1, T_2) = 1 - \frac{W(S_{1,2})}{\min(|T_1|, |T_2|)} \leq 1.$$

$$2) d(T_1, T_2) = d(T_2, T_1) \quad (\text{Symmetry})$$

This follows directly from the definition of the node similarity function (1) and that  $\min(|T_1|, |T_2|) = \min(|T_2|, |T_1|)$  in equation (3).

$$3) d(T_1, T_2) = 0 \Leftrightarrow T_1 = T_2 \quad (\text{Identity and uniqueness})$$

Considering the implication direction  $\Leftarrow$  (identity), it is straightforward to show that when  $T_1, T_2$  are identical trees the similarity index value produced by

equation (2) is 1. Since  $T_1 = T_2$ , from (5)  $|T_1|=|T_2|=W(S_{1,2})$ . Hence the distance

between them calculated by equation (3),  $d(T_1, T_2) = 1 - \frac{W(S_{1,2})}{\min(|T_1|, |T_2|)}$ , is 0.

For the opposite implication direction  $\Rightarrow$  (uniqueness), from equation (3), we have  $d(T_1, T_2) = 0 \Rightarrow W(S_{1,2}) = \min(|T_1|, |T_2|)$ . Though it can rarely occur, the similarity  $W(S_{1,2})$  may equal to  $|T_1|$  or  $|T_2|$  even when  $|T_1|$  and  $|T_2|$  are not identical. That leads to conclude that the distance function is *Pseudo Metric*. In our application, however, the chances are minimal to have a 0 distance when trees are not identical. SGM trees representing documents are normally large, and nodes have many attributes.

$$4) d(T_1, T_2) + d(T_2, T_3) \geq d(T_1, T_3) \quad (\text{Triangular Inequality})$$

This is to prove that:

$$1 - \frac{W(S_{1,2})}{\min(|T_1|, |T_2|)} + 1 - \frac{W(S_{2,3})}{\min(|T_2|, |T_3|)} \geq 1 - \frac{W(S_{1,3})}{\min(|T_1|, |T_3|)}$$

After some algebraic manipulations, this triangular inequality becomes equivalent to:

$$\min(|T_1|, |T_2|) \min(|T_2|, |T_3|) \min(|T_1|, |T_3|) \geq W(S_{1,2}) \min(|T_2|, |T_3|) \min(|T_1|, |T_3|) + W(S_{2,3}) \min(|T_1|, |T_2|) \min(|T_1|, |T_3|) - W(S_{1,3}) \min(|T_1|, |T_2|) \min(|T_2|, |T_3|) \quad (4)$$

There are six possible cases to be distinguished and proven:

- a.  $|T_1| \leq |T_2| \leq |T_3|$       b.  $|T_1| \leq |T_3| \leq |T_2|$       c.  $|T_2| \leq |T_1| \leq |T_3|$
- d.  $|T_2| \leq |T_3| \leq |T_1|$       e.  $|T_3| \leq |T_1| \leq |T_2|$       f.  $|T_3| \leq |T_2| \leq |T_1|$

a) When  $|T_1| \leq |T_2| \leq |T_3|$ , equation (4):  $\min(|T_1|, |T_2|) \min(|T_2|, |T_3|) \min(|T_1|, |T_3|) \geq W(S_{1,2}) \min(|T_2|, |T_3|) \min(|T_1|, |T_3|) + W(S_{2,3}) \min(|T_1|, |T_2|) \min(|T_1|, |T_3|) - W(S_{1,3}) \min(|T_1|, |T_2|) \min(|T_2|, |T_3|)$  reduces to  $|T_1||T_2| \geq W(S_{1,2})|T_2| + W(S_{2,3})|T_1| - W(S_{1,3})|T_2|$ .

Given that  $|T_1||T_2| \geq |T_1||T_2| + |T_1||T_2| - |T_1||T_2|$

And  $W(S_{1,2}) \leq \min(|T_1|, |T_2|)$ , thus  $W(S_{1,2}) \leq |T_1|$ ,

$W(S_{2,3}) \leq \min(|T_2|, |T_3|)$ , thus  $W(S_{2,3}) \leq |T_2|$ ,

and  $W(S_{1,3}) \leq \min(|T_1|, |T_3|)$ , thus  $W(S_{1,3}) \leq |T_1|$

When we substitute accordingly, we get:  $|T_1||T_2| \geq W(S_{1,2})|T_2| + W(S_{2,3})|T_1| - W(S_{1,3})|T_2|$ .

b)  $|T_1| \leq |T_3| \leq |T_2|$

The triangular inequality reduces to  $|T_1||T_3| \geq W(S_{1,2})|T_3| + W(S_{2,3})|T_1| - W(S_{1,3})|T_3|$ .

$|T_1||T_3| \geq |T_1||T_3| + |T_1||T_3| - |T_1||T_3|$

Since  $W(S_{1,3}) \leq |T_1|$ ,  $W(S_{2,3}) \leq |T_3|$ , and  $W(S_{1,2}) \leq |T_1|$

Hence:  $|T_1||T_3| \geq W(S_{1,2})|T_3| + W(S_{2,3})|T_1| - W(S_{1,3})|T_3|$ .

c)  $|T_2| \leq |T_1| \leq |T_3|$

We need to prove  $|T_1||T_2| \geq W(S_{1,2})|T_1| + W(S_{2,3})|T_1| - W(S_{1,3})|T_2|$ .

$|T_1||T_2| \geq |T_1||T_2| + |T_1||T_2| - |T_1||T_2|$

Since  $W(S_{1,2}) \leq |T_2|$ ,  $W(S_{2,3}) \leq |T_2|$ , and  $W(S_{1,3}) \leq |T_1|$

Thus:  $|T_1||T_2| \geq W(S_{1,2})|T_1| + W(S_{2,3})|T_1| - W(S_{1,3})|T_2|$ .

The triangle inequality was proved for the three cases: (a)  $|T_1| \leq |T_2| \leq |T_3|$ , (b)  $|T_1| \leq |T_3| \leq |T_2|$ , and (c)  $|T_2| \leq |T_1| \leq |T_3|$ . Similarly, the symmetry attribute can be used to prove the other possible arrangements. Specifically, since the roles of  $T_1$  and  $T_3$  are

symmetric, hence we can use this symmetry to prove the other three cases: (d)  $|T_2| \leq |T_3| \leq |T_1|$ , (e)  $|T_3| \leq |T_1| \leq |T_2|$ , (f)  $|T_3| \leq |T_2| \leq |T_1|$ .

---

## 5.5 The Algorithm for Computing the Similarity Measure

This section presents the algorithm for calculating similarity between two trees through extracting all distinct similarity common sub-trees. When node similarity calculations are ignored (or setting similarity threshold to 100% match), the extraction of common similarity sub-trees becomes equivalent to the problem of finding exact common sub-trees, i.e., merely to verify whether sub-trees from one tree is a sub-tree of the other, or not. Thus, the algorithm is based on a dynamic programming technique and a sub-tree identification algorithm first introduced by Matula and Reyner [69, 99] and further used to extract the maximum similarity common sub-tree by Torsello et al. [96]. The algorithm is extending on the previous algorithm so as to extract all distinctive similarity common sub-trees and use them to calculate the overall similarity between trees as showed in Figure 5.3. The extensions are mainly to use the attribute similarity function, explained above, to generalize the algorithm to consider all common sub-trees, and to eliminate overlapping when nodes are included in the matching of more than one sub-tree. The attribute similarity function is an independent procedure that can be implemented in different ways, and with different considerations of weights and attributes to be considered. The inclusion of all common similarity sub-trees is accomplished through recursive iterations and holding on the value of the similarity index of the matched sub-trees. Eliminating the overlapping is done using a dynamic programming technique that keeps track of nodes visited and their attribute similarity values.

---

```

1:  Similarity( $T_1, T_2$ )
2:  begin
2:      OverallSim = 0;
3:      for each node  $x$  in  $T_1$  do
4:          Sim = Sub-treeSim( $x, \text{root}(T_2)$ )
5:          if Sim > 0 then
6:              OverallSim += Sim;
7:          end if
7:      end for
8:      OverallSim /= min( $|T_1|, |T_2|$ );
9:      return OverallSim;
10: end

```

---

```

1:  Sub-treeSim( $x, y$ )
2:  begin
3:       $C_x = \text{Children}(x)$ ;
4:       $C_y = \text{Children}(y)$ ;
5:      for each  $x_i$  in  $C_x$  do
6:          for each  $y_i$  in  $C_y$  do
7:              if (Node-Sim( $x_i, y_i$ ) > Node-SimTable[ $x_i, y_i$ ])
8:                  and
9:                  (Node-Sim( $x_i, y_i$ ) > MatchingThreshold) then
10:                     Node-SimTable[ $x_i, y_i$ ] = Node-Sim( $x_i, y_i$ );

```

```

11:             Return Node-Sim( $x_i, y_i$ ) + Sub-treeSim( $x_i, y_j$ );
12:         else
13:             return 0;
14:         end if
15:     end for
16: end for
17: end

```

---

**Figure 5.3** The Algorithm for Computing the Similarity Measure

For two trees  $T_1 = (N_1, E_1)$  and  $T_2 = (N_2, E_2)$ , and  $x \in N_1$  and  $y \in N_2$ , the function *Similarity* accepts as input the roots of the two trees and returns the similarity value based on the developed similarity measure stated in Section 5.2; Equation (3). It iteratively calls the recursive function *Sub-treeSim* that calculates, based on Equation (2), the similarity of the sub-tree isomorphism between  $T_1$  and  $T_2$  rooted at nodes  $x$  and  $y$ , respectively. The function utilizes an array, *Node-SimTable*[[[]]] to solve the problem of the overestimating similarity that could result from considering a node in more than one common similarity sub-tree. The function *Node-Sim*( $x_i, y_i$ ) computes the similarity between the two nodes  $x_i, y_i$  as defined in Equation (1). Moreover, a threshold value for node matching is also utilized.

Finding of a common similarity sub-tree of two trees can be computed in a polynomial-time. In [96], it was shown that for two trees with  $N$  and  $M$  nodes respectively, and a maximum branching factor of the two trees  $b$ , an overall computational complexity of  $O(bNM)$  or faster can be achieved. The algorithm is based on this algorithm and extends on it by requiring computing all distinct common similarity sub-trees. Another distinction is that the algorithm deals with trees that

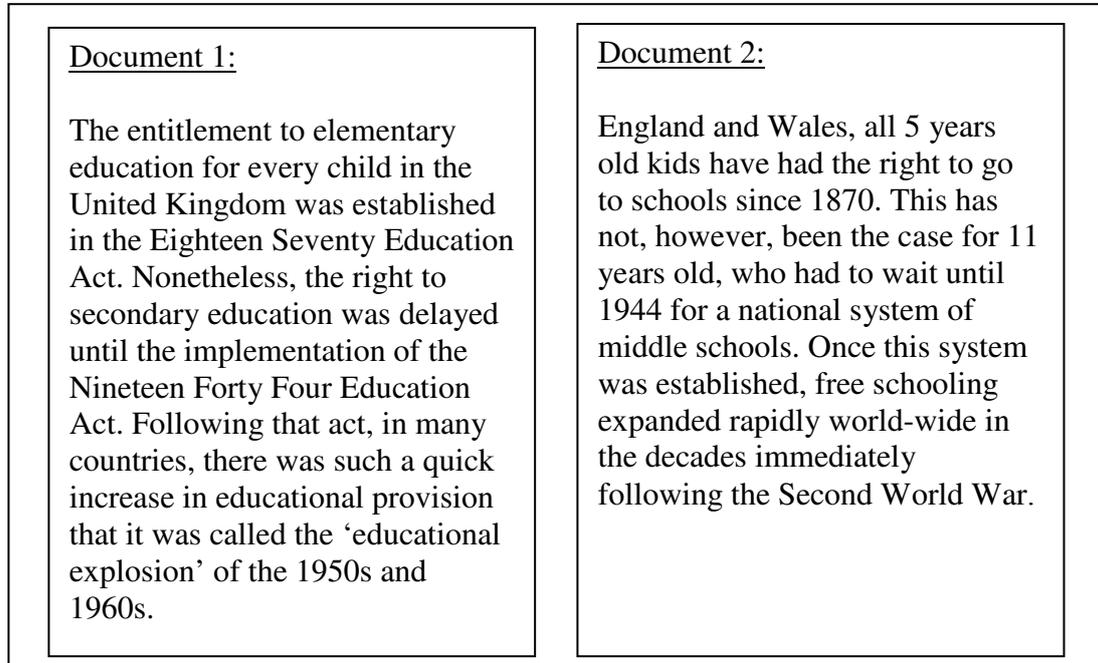
have multiple attributes for their nodes. Thus, if there exists  $W$  of these sub-trees, the complexity would still be polynomial; i.e.,  $O(bkNMW)$ , where  $k$  is the maximum number of attributes a node can hold. The algorithm memory requirement is also polynomial and it is  $O(kNM)$ . The algorithm processed moderate size document sets in matter of minutes. Performance results are discussed in more details in Chapter 6.

---

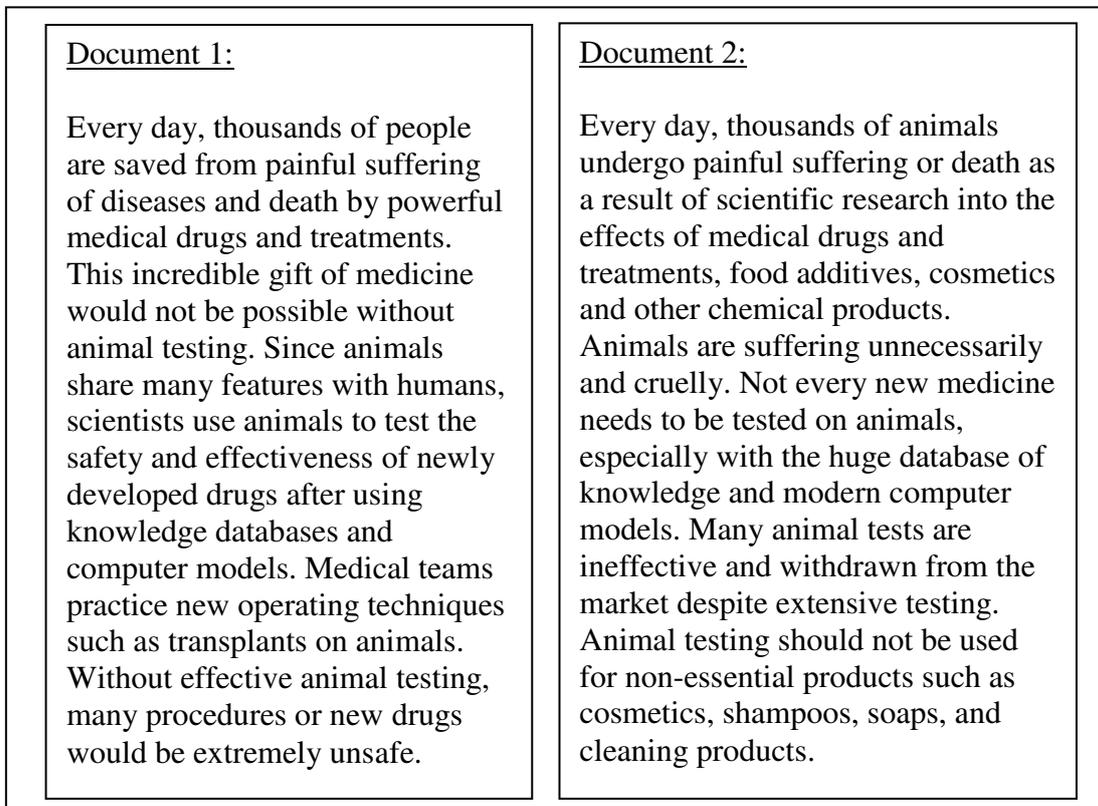
## 5.6 Measuring Similarity of Two Documents

To proof the concepts, an assessment is performed on of the proposed SGM semantic representation of text and the proposed similarity measure in distinguishing between pairs of texts. The proposed formulation is compared against existing approaches, namely the VSM using binary weights with the cosine coefficient similarity measure (introduced in Section 2.1). The two pair of documents listed in Figure 5.4. (a) and (b) are used. These documents are of interest as they exhibit the variability phenomenon typical in humans' expressions. It is easy for humans to assess the similarity of text passages even when they contain variant words and articulation (e.g., Document 1, and Document 2 in Figure 5.4(a)), or to recognize dissimilarity of texts when they may be composed of the same or similar set of words (e.g., Document 1, and Document 2 in Figure 5.4(b)). The similarity assessment of the technique shows clearly more agreement with those that can be concluded by humans. When calculating nodes similarity, only three node attributes are considered (namely the original text, syntactic tags, and semantic variations), and the algorithm produced gave 64% similarity for the first pair of text, and 45% for the second pair. On the other hand, the VSM technique produced 13% and 42% for the pair Document 1 and Document 2 of Figures 5.4 (a) and (b), respectively. This reflects the

effectiveness of the approach in producing meaningful similarity measures for text as it agrees with humans judgments than the conventional word-based technique. More rigorous experimental work is carried out and reported in Chapter 6.



**Figure 5.4. (a)** Two Similar Documents Composed of Different Words



**Figure 5.4. (b)** Two Different Documents Contain Similar Words

---

## 5.7 Conclusions

This chapter presented a new technique and the algorithmic steps for calculating similarities between documents represented semantically as node-attributed trees. The similarity measure is an inexact tree matching method. This measure is deemed to be suitable for measuring similarity between documents represented by the semantic graph model (SGM). The SGMs are essentially trees with symbolic node attributes that could be quite large in size and communality between trees could be sparse. Thus, the measure is based on finding all distinct similarity common sub-trees to reflect the level of commonality between these trees without overrating.

It was shown that the measure is a normalized pseudo metric, as it fulfills most of the metric axioms. In addition, the pseudo-code for the algorithmic steps to compute the measurement is provided, and shown that it is computationally tractable, as it has a polynomial time and space complexity.

Testing was conducted regarding the similarity technique on two different pairs of documents to prove the concept effectiveness. The technique showed more agreement with what humans conclude than other conventional word-based techniques. The following chapter presents further experimental work that utilizes the semantic-understanding approach to mine documents.

## Experimental Work: Semantic Document Clustering



*"Don't go where the path may lead.*

*Instead, go where there is no path and leave a trail"*

By unknown.

---

In this chapter, the application of the semantic understanding-based framework in mining documents is presented. A case study that serves as a test-bed is used to evaluate the approach. A document mining process, namely semantic document clustering, is investigated and tackled in various ways. Implementations and experimental work have been carried out. Results are also presented and analyzed. Section 6.1 gives an introduction to the experimental and implementation choices taken. Section 6.2 explains the experimental setup in terms of the used data sets and their characteristics, the developed text analysis system and its advantages, the document representations that were implemented, the employed similarity measures, the implemented clustering algorithms, and the utilized evaluation techniques. A listing of the experiment collected results is reported in Section 6.3.

The results are discussed and analyzed in Section 6.4. The chapter ends with some concluding remarks in Section 6.5.

---

## **6.1 Document Clustering**

Clustering is a key concept in document mining (see Section 2.1.3.1 for details). A clustering process aims to analyze the similarities between data objects and build groups of them. The grouped objects can then be used to navigate easily through a very large list of data sets. For instance, based on the output of the clustering process, a further reading recommender component can analyze the currently viewed documents, and suggest similar documents (possibly within the same cluster) to the user as related materials.

While a large number of statistical document clustering approaches have become available, relatively little attention has been paid to the clustering of documents represented using symbolic structures. In principle, however, given suitable documents similarity (or dissimilarity) measurements, many of the clustering algorithms originally developed in the context of statistical pattern recognition, can be applied in the symbolic domain. This is the course that has been adapted to cluster documents represented semantically. Text documents were syntactically and semantically parsed and represented by their semantic graph models. Pair-wise similarities were computed using the developed inexact tree matching similarity measure. The similarity matrix was fed to different clustering algorithms to produce clusters for the data sets. Two different data sets were used; each with distinct features. The clustering results were then evaluated using standard evaluation

techniques in document mining. Below are details of the steps taken, followed by an analytical discourse on the findings.

---

## **6.2 Experimental Setup**

The semantic-based mining approach is implemented and evaluated based on its ability to perform high quality document clustering. Text documents are parsed to produce the rich syntactic and semantic representations of SGM. Based on these semantic representations, pair-wise document similarities are estimated using the inexact node-attributed tree matching algorithm that is based on finding all distinct similarity common sub-trees. Clustering algorithms that accept these similarities are utilized to produce clusters of the data sets. The following are details of the experimental work including the text parser implemented, the data sets used, the text representations, the similarity measures, the clustering algorithms, and the collected results.

### **6.2.1 Text Parser**

A parsing system, which performs the syntactic analysis, and builds the semantic structures of SGM is implemented. Components are utilizing from the open source GATE (General Architecture for Text Engineering) project from University of Sheffield in the UK (<http://gate.ac.uk>) [22], and a commercial integrated development environment (IDE), Visual Text [92, 93], from Text Analysis International, Inc. GATE provides an extensible framework for information extraction and text analysis, and Visual Text integrates NLP++ programming language for rapid parsers building. The design of the text analysis system is based on a multi-pass, and a multi-strategy

architecture that could be implemented within Visual Text IDE. The syntactic and semantic parser developments are built upon the TAI Parse general analyser [93] that is provided as an open source from Text Analysis. The analyser contains 123 passes that build syntactic (parse trees), and semantic (the SGM) structures. Many other external processing components and language resources are wrapped and made usable interchangeably within the system. Specifically, every document in the processed data set goes through tokenization, part of speech tagging, syntactic parsing, semantics and discourse analysis. Figure 6.1 (a) and (b) are snapshots of the parser syntactic and semantic output structures; namely the parse tree, and the semantic graph model (SGM) tree.

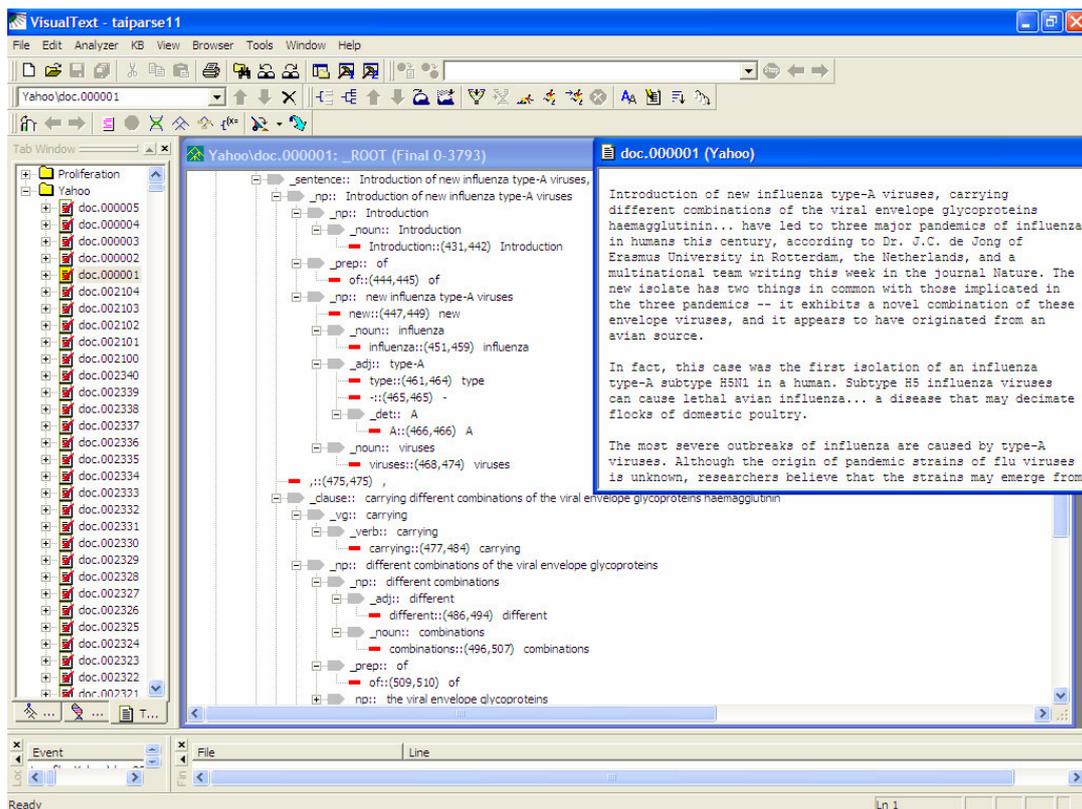


Figure 6.1 (a) A Snapshot of the Syntactic Parse Tree.

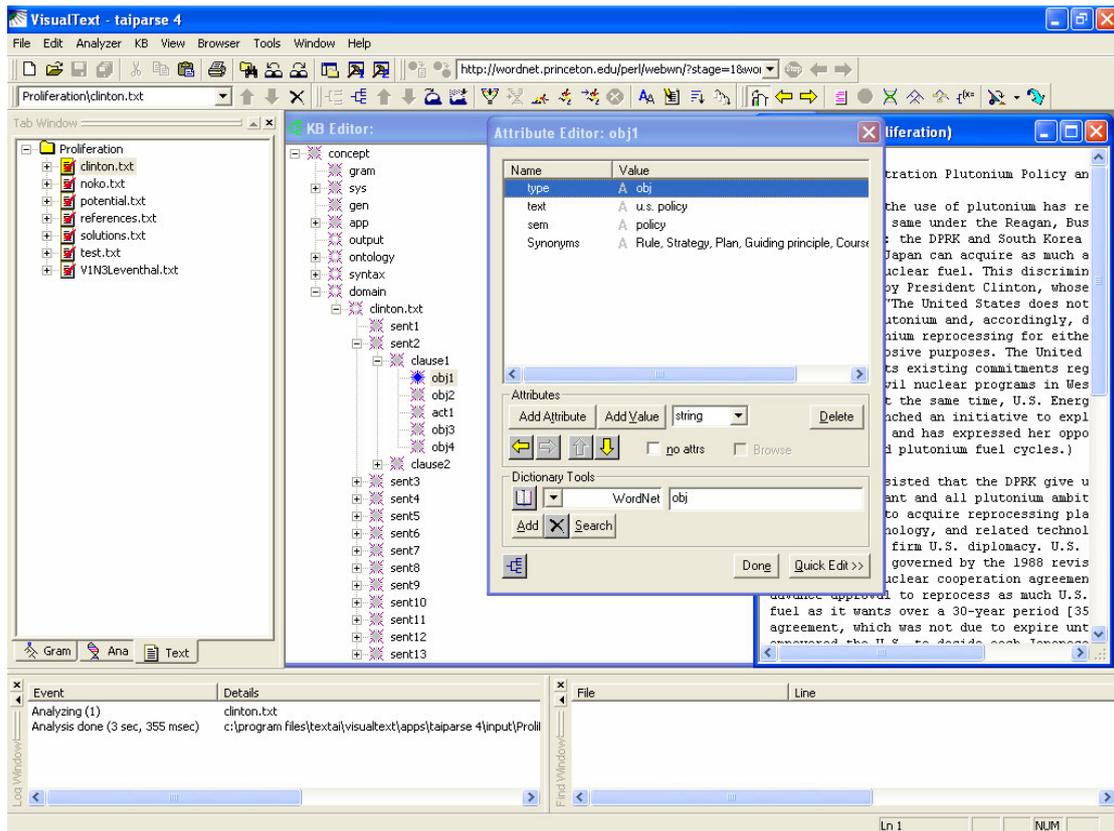


Figure 6.1 (b) A Snapshot of the SGM Tree.

## 6.2.2 Data Sets

For experimentation and testing two document sets are used, a Web documents collection (Data-Set-1), and a learning-objects (LO) data set (Data-Set-2).

The Web document set is a collection of 2340 Reuters news articles posted on Yahoo! News. This corpus is especially interesting for evaluation, as it comes along with a *hand-crafted* classification. All documents have been classified manually by Yahoo experts to one or more of six main categories of Reuter's news feed, namely, business, entertainment, health, politics, sports, and technology. The data set has been collected and used by Boley et al [11, 12, 13] for clustering.

The other data set used for testing is composed of LOs metadata records. This is a data set of metadata entries collected from Canada's SchoolNet website (www.schoolnet.ca). The data described by the metadata is mainly educational

material. The metadata was authored by SchoolNet to make it easy for teachers and learners to find and access educational resources in various subjects. The data set was manually categorized by the authors into 306 hierarchal classes. The purpose of this data set was to facilitate research on learning object metadata by applying current data mining techniques (such as classification and clustering) developed in the pattern analysis and machine intelligence (PAMI) research group at the University of Waterloo ([www.pami.uwaterloo.ca](http://www.pami.uwaterloo.ca)). The techniques developed in relation to this data are to be implemented and integrated into the product of the LORNET ([www.lornet.org](http://www.lornet.org)) project; the TELOS telelearning system. SchoolNet uses an extended set of the Dublin Core metadata element set. Each record in the original data set has 37 different fields. Only text fields in the metadata are being utilized. Specifically, *titles* and *descriptions* fields are used to analyze and explore how the data can be organized or accessed efficiently through clustering. The actual data is available in XML format, thus, all the interesting fields can be extracted and saved in separate files.

The difference between the two data sets (See table 6.1) are in the length of documents and the number of classes they are classified to. Reuter's news feeds are in average one page long articles while SchoolNet description fields are a few and concise sentences that describe the learning objects. The Reuter's documents are classified into 6 main classes, while SchoolNet's are categorized into 306 hierarchal classes.

<b>Data Collection</b>	<b>Source</b>	<b>Type</b>	<b>No. of Doc.</b>	<b>Categories</b>
Data-Set-1	Reuters News	Web Documents	2340	6
Data-Set-2	SchoolNet	Metadata Records	2371	306

**Table 6.1** Description of Data Sets Used in the Experiments

### 6.2.3 Text Representations

All text documents are represented using the proposed semantic graph model (SGM). The SGM representation model constitutes trees that have attributes for their nodes. Each document is represented as a tree rooted in a node describing its file name, and branches to sub-trees that represent paragraphs, sentences, clauses, and predicate-argument nodes with attributes describing the arguments. As applicable, the node attributes include *Name*: a unique identification for the node, *Type*: the semantic role of concept, *Text*: the original text, *Syntax*: the part of speech tag, *Synonyms*: dictionary senses of the concept, *Semantic*: the disambiguated meaning of the concept, and *Relation*: other node IDs that are connected to the node. Different algorithms are employed and integrated in the parser to produce the SGM representations (see Chapter 4 for details).

For comparison purposes, the documents in both document sets are also represented by the vector space model (VSM) as it is commonly practiced in text mining techniques. To convert the data set to the vector space some pre-processing procedures are performed such as normalizing words, analyzing words globally, and word weighing. To normalize words, numeric, and stop words are removed. All words are converted to lowercase, and words of length 2 or less are also removed. Note that no stemming is done. Analyzing words globally is done by building a list of unique words from all documents, and calculating the document frequency of each word. Then, for each document, the word frequency is calculated and weighed using the following formula:

$$w_{i,j} = f_{i,j} \log\left(\frac{N}{df_j}\right),$$

where  $f_{i,j}$  is the term frequency of word  $j$  in document  $i$ ,  $N$  is the number of documents, and  $df_j$  is the document frequency of word  $j$ . Finally, an output of the term-by-document matrix is produced.

#### 6.2.4 Similarity Measures

The distance measure technique is used to measure distances between documents. It follows the similarity calculation approach that is based on finding all distinct common similarity sub-trees introduced in Chapter 6. The algorithm is applied to the semantic trees of the SGMs produced by the semantic parser. Various attributes with different weight settings (as pertain to the node similarity function; Section 5.3 and equation 1) are tested. Attribute consideration starts from taking into account only the original text to represent a node and extends considering other attributes the parser accumulates. Specifically, the experiments were carried out with mainly five options of sets of attributes to include in measuring the similarities. These options are (1) original text only, (2) original text, and syntax tag, (3) original text, syntax tag, and semantic role, (4) original text, syntax tag, semantic role, and semantic disambiguated sense, and (5) original text, syntax tag, semantic role, semantic disambiguated sense, and semantic variation. The weights are tuned heuristically so the best results of clustering are obtained.

To measure similarities of documents represented in the vector space the cosine correlation measure is used. It is a commonly used measure with the VSM, and is defined as:

$$\cos(x, y) = \frac{x \cdot y}{|x| |y|},$$

where  $(\cdot)$  denotes the vector dot product, and  $|x|$  and  $|y|$  are the lengths of vector  $x$ ,  $y$ , respectively. The cosine measure gives high similarity values to documents that share

the same set of words with high term frequencies, and lower values to those that do not.

### **6.2.5 Clustering Algorithms**

The semantic approach to represent text and determine similarity between them is evaluated by allowing clustering algorithms to use the produced similarity matrix of the document set. The test is performed with standard clustering algorithms that accept the pair-wise (dis)similarity between documents, rather than the internal feature-space or SGM representations of the documents. The employed algorithms included: Single-Pass Clustering,  $k$ -nearest neighbour ( $k$ -NN) Clustering, and Hierarchical Agglomerative Clustering (HAC), reviewed in Chapter 2. The different parameters of these algorithms (i.e., the clustering threshold parameters) are tuned so the best results of clustering are obtained.

### **6.2.6 Evaluations**

To evaluate results, a benchmark of manually classified document sets is used; Reuters news feeds, and SchoolNet metadata records. The main aspect of the evaluation is the quality of the clustering task output, which is measured in terms of cluster quality. The widely used evaluation indices, F-measure that combines precision and recall, Entropy, and Overall-Similarity (reviewed in Chapter 2) are used for this purpose as clustering quality evaluation measures.

### 6.3 Collected Results

The above mentioned clustering algorithms were utilized to cluster the two data sets using document-to-document similarities produced by the semantic-based approach, and the vector space model. The following tables summarize the chosen techniques and their results. The different tables report clustering results obtained from the clustering algorithms. Tables from 6.1 to 6.5 show the results of the semantic approach when the similarity measure considers different sets of node attributes, e.g., original text only; original text, and syntax tag; original text, syntax tag, and semantic role; original text, syntax tag, semantic role, and semantic disambiguated sense; and original text, syntax tag, semantic role, semantic disambiguated sense, and semantic variation. Tables 6.2, 6.3, 6.4, and 6.5 also show the accumulative improvement percentage gained by including the different options. Table 6.6 lists the results from the vector space model approach.

	Data-Set-1			Data-Set-2		
Algorithm	F-Measure	Entropy	Overall Similarity	F-Measure	Entropy	Overall Similarity
Single-Pass	0.390832	0.803493	0.655075	0.252542	0.816894	0.620169
KNN	0.378677	0.745176	0.718541	0.368948	0.765487	0.685402
HAC	0.411209	0.730386	0.721991	0.389802	0.753278	0.702591

**Table 6.1** Experimental Results – Semantic-based - Option 1

	Data-Set-1			Data-Set-2		
Algorithm	F-Measure	Entropy	Overall Similarity	F-Measure	Entropy	Overall Similarity
Single-Pass	0.459654	0.769823	0.695402	0.296580	0.762680	0.701258
	+14.97%	-4.19%	+6.156%	+17.44%	-6.64%	+13.08%

KNN	0.402159 +6.20%	0.684596 -8.23%	0.769842 +7.14%	0.395425 +7.18%	0.735846 -3.87%	0.712365 +3.93%
HAC	0.458902 +11.6%	0.671568 -8.05%	0.790235 +9.45%	0.395025 +1.34%	0.693502 -7.94%	0.742540 +5.69%

**Table 6.2** Experimental Results – Semantic-based - Option 2

	Data-Set-1			Data-Set-2		
Algorithm	F-Measure	Entropy	Overall Similarity	F-Measure	Entropy	Overall Similarity
Single-Pass	0.465891 +19.2%	0.612358 -23.79%	0.721545 +10.15%	0.325891 +29.04%	0.735487 -9.97%	0.724581 +16.84%
KNN	0.421568 +11.33%	0.650245 -12.73%	0.778923 +8.43%	0.415987 +12.75%	0.701254 -8.39%	0.732580 +6.88%
HAC	0.488962 +18.91%	0.648912 -11.15%	0.812546 +12.54%	0.412574 +5.84%	0.668923 -11.12%	0.758921 +8.02%

**Table 6.3** Experimental Results – Semantic-based - Option 3

	Data-Set-1			Data-Set-2		
Algorithm	F-Measure	Entropy	Overall Similarity	F-Measure	Entropy	Overall Similarity
Single-Pass	0.478954 +22.55%	0.568951 -29.19%	0.758941 +15.86	0.335890 +33.0%	0.726902 -11.02%	0.758921 +22.37
KNN	0.435879 +15.11%	0.635879 -14.67%	0.786943 +9.52%	0.435910 +18.15%	0.692458 -9.54%	0.761238 +11.06%
HAC	0.544978 +32.52%	0.624589 -14.49%	0.830124 +14.98%	0.446950 +14.66%	0.649850 -13.73%	0.770231 +9.63%

**Table 6.4** Experimental Results – Semantic-based - Option 4

	Data-Set-1			Data-Set-2		
Algorithm	F-Measure	Entropy	Overall Similarity	F-Measure	Entropy	Overall Similarity

Single-Pass	0.628954 +60.93%	0.523980 -34.79%	0.792364 +20.96%	0.382654 +51.52%	0.582349 -28.71%	0.796584 +28.45
KNN	0.652389 +72.28%	0.4698231 -36.95%	0.832569 +15.87%	0.548912 +48.78%	0.601258 -21.45%	0.812365 +18.52%
HAC	0.618950 +50.52%	0.423569 -42.01%	0.862489 +19.46%	0.583691 +49.74%	0.548925 -27.13%	0.789564 +12.38%

**Table 6.5** Experimental Results – Semantic-based - Option 5

	Data-Set-1			Data-Set-2		
Algorithm	F-Measure	Entropy	Overall Similarity	F-Measure	Entropy	Overall Similarity
Single-Pass	0.432591	0.758924	0.789214	0.321456	0.784591	0.675901
KNN	0.395681	0.701265	0.746589	0.385694	0.735405	0.712561
HAC	0.489562	0.689530	0.758962	0.401256	0.798560	0.723548

**Table 6.6** Experimental Results – Vector Space Model

The performance of the model was closely examined to verify that the semantic similarity matching algorithm is scalable enough for moderate to large data sets. The experiments were performed on an Intel Centrino, 1 GHz machine with 512MB main memory. The system was written in NLP++ and run through an interpreter on Windows XP operating system. For both data sets the algorithm performed in a near-linear time to perform the similarity calculation. The longer time was taken when analysing the documents to represent them with SGM. This can lead to a conclusion that, in the current state of NLP, one can use the system for off-line application or in processing small document sets online. Although the two datasets contain a similar number of documents, Data-Set-1 took more time than Data-Set-2. This can be attributed to the fact that Data-Set-1 in average has almost twice as many text contents per document as Data-Set2, so the algorithm ends up matching larger SGM trees that can have larger number of sub-trees matching than in Data-Set-2.

---

## 6.4 Analysis of Results

The experiments revealed some interesting trends in terms of the clustering qualities. Clearly, the results show the effectiveness of the semantic-based approach for clustering. In addition, the experiments illustrate improvements when more semantic clues are included in the process of measuring similarities between SGMs of documents. This affirms the assertion that the performance is to proportionally improve as more semantic understanding of contents is considered. The results listed in Tables 6.1 to 6.5, show the improvement on the clustering quality when more attributes are included in semantic-based similarity measure. The improvements were achieved at a factor of up to 72% for Data-Set-1 and 52% for Data-Set-2 from the base case of considering only the original text of nodes (option 1) to considering all five semantic analysis results, i.e., original text, syntax tag, semantic role, semantic disambiguated sense, and semantic variation (option 5). The threshold parameters for the different clustering algorithms were manually tuned and the ones that produced the best results were reported. The F-measure index showed a noticeable increase. The enhancements of results were consistent among the two data sets. However, for the SchoolNet data set the improvements were less obvious and lower than Data-Set-1 in general, which could be attributed to the nature of its documents contents, where a large percentage of Data-Set-2 texts are short descriptions, and are discussing specific topics that require special knowledge from the parser to perform more accurate semantic analysis. Moreover, the number of classes Data-Set-2 is categorized into is much larger than Data-Set-1 which makes it harder for the clustering algorithm to produce clusters that match these classes. The same can be noticed about the entropy

and the overall similarity indices. The entropy is minimized as a contribution of semantic-based similarity with options going from 1 to 5. The overall similarity also increased from option 1 to option 5. These are interesting observations since, as mentioned earlier, if we rely solely on counting words for measuring similarity we might not get an accurate similarity measurement. The role played by the semantic-based similarity measurement is that it helps assess the similarity measurement by enriching and making the space of communality more explicit.

To better understand the effect of the inclusion of semantic information when calculating similarity on the clustering quality, a plot of the clustering quality profile indices is used against the similarity options in Figures 6.1 and 6.2. The plotted values are the averages of f-measure, entropy, and overall similarity of the different clustering algorithms. The graphs illustrate the effect of semantic-based similarity with the different options to the F-measure, the entropy, and the overall similarity for both data sets. The values of these measures are also plotted with regard to the VSM. It is easy to notice the enhancement of the clustering as more semantic clues are considered. The enhancement is, however, non-linear. Options 2 and 3 seem not to have as much effect on the clustering quality, but they never bring it down. Option 5 had the most effect to bring up the quality of clustering. As it was anticipated, keywords alone cannot capture all the similarity information between documents, thus in both data sets the quality of semantic clustering with option 1 was as low as the conventional clustering technique, i.e., VSM.

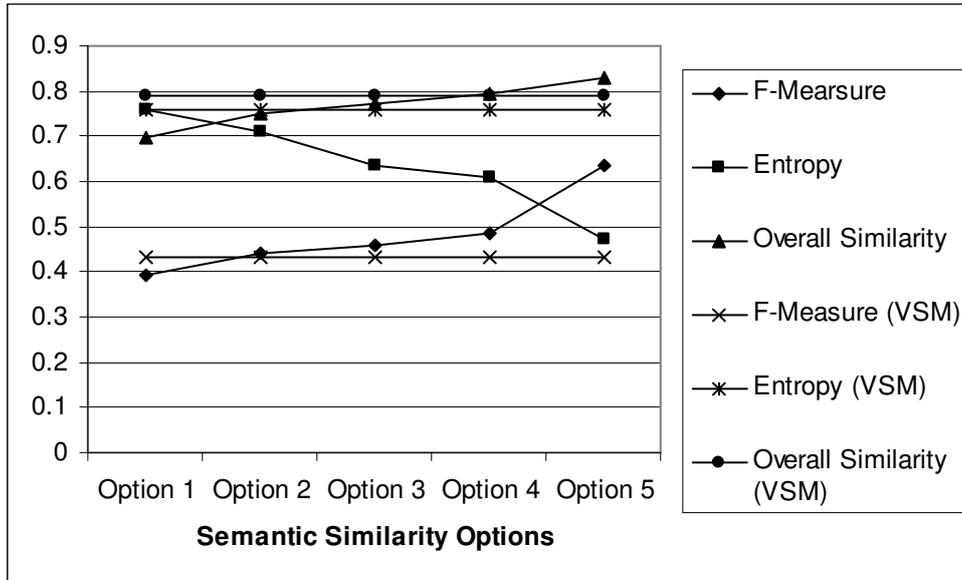


Figure 6.1 Clustering Results for Data-Set-1

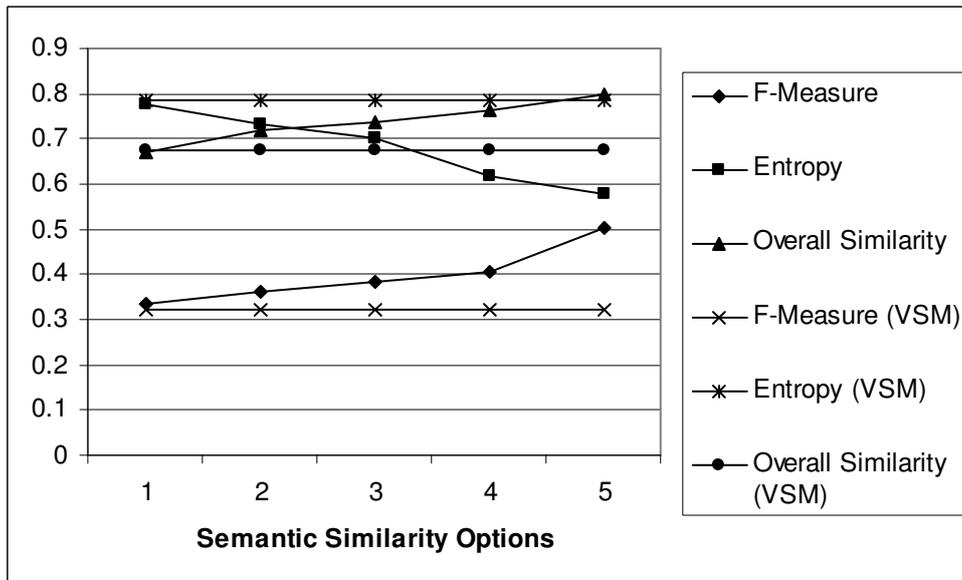


Figure 6.2 Clustering Results for Data-Set-2

## 6.5 Conclusions

A mining system that is composed of semantic components was demonstrated in an attempt to improve the accuracy of measuring the similarity between documents and using the similarity in applying the document clustering problem. By exploiting

the semantic analysis findings, better clustering results could be achieved. The document analysis components that were implemented are capable of identifying the meaning structures of text in documents. The second part, and perhaps the most important, is to measure similarity between parsed documents. The measure has the most impact on the performance as to how much semantic information it considers. Using the maximum amount of the semantic information enables us to perform similarity calculations between documents in a robust and accurate way. The quality of clustering achieved using this model significantly surpasses the traditional vector space model based approach.

The merit of the design is that each component can be utilized and refined independently. However, combining all of these components leads to better outputs, as justified by the results presented in this chapter. The system was tested against different standard clustering techniques and different data sets, and it was found to be very beneficial and rewarding to pursue the semantic understanding direction.

---

# CHAPTER 7

## Summary and Conclusions



*"What you say last in your sales meeting,  
is what your salespeople will remember most"*

By Kevin Davis,  
*"Six Keys To More Powerful Sales Meetings:  
Plan Conclusions That Get Action"*

---

The chapter gives a summary of the research work, points to its contributions and findings, and discusses its possible future extensions. Section 7.1 highlights the uniqueness of the research direction and briefly presents the approach taken to mine documents, i.e., the semantic understanding-based approach to represent text in documents, to measure similarity between documents, and to apply mining processes using the representation and the similarity measure. Section 7.2 cites experimental results and refers to conclusions drawn from these results. It justifies the advantages of adapting the semantic approach as it showed strong merits and the possibility for further improvements. Section 7.3 summarizes contributions of the work in advancing the state-of-the-art research in the area of document mining by introducing and developing a framework based on semantic understanding of text. Section 7.4

suggests further developments to the work and mentions how new avenues can be explored.

---

## **7.1 The Proposed Approach**

The available volume of knowledge in documents must be discovered. To help make that possible, the tasks of managing, analyzing, searching, filtering, and summarizing documents should be automated. This requires an understanding of documents' contents in some fundamental way. Conventional document mining systems mainly use the presence or absence of keywords to mine (i.e., retrieve, filter, extract, classify, or cluster) texts. This typical approach could satisfy some of the mining processes requirements. However, simple word counting and frequency distributions do not capture the meaning behind the words, which results in limiting the ability to distinguish texts and to mine them. Breaking through this limitation requires systems to deeply process the texts they handle. The main objective of the work is to advance the state-of-the-art research of document mining by developing and demonstrating the use of semantic understanding as a basis of its mechanisms.

The NLP field and its developed techniques can now deal with relatively well-formed sentences in many languages. There exist mature systems that can, for instance, disambiguate polysemous words with high accuracy, tag words in text with parts-of-speech information, and represent text documents in canonical machine-usable forms. In addition, similarity measuring techniques can be developed to provide adequate assessments of similarity between represented documents. The work focuses on investigating and building systems that make use of the above mentioned techniques to mine documents. The goal is to achieve the necessary

understanding through representing meaning, measuring similarity, and carrying out mining tasks using this semantic representation. When semantic representations and similarity measures are utilized, mining outcomes are expected to surpass those based on traditional keyword-based approaches.

The research work focused on two areas: (1) theoretical and modeling, and (2) experimental and testing. The underlying theoretical and modeling work is based on introducing semantic notions to represent text and similarity measures to estimate distances between the represented text documents. Document mining processes are carried out by the developed models of the theoretical schemes. The work underwent many revisions in terms of formalization and empirical testing. All attempts are made to have the semantic-based text representation fit the requirements of document mining processes, and be extendable for further refinements. The representation scheme reflects existing relations between concepts and facilitates accurate similarity judgments. Moreover, an appropriate similarity measure has been developed for the semantic representation. The similarity measure is general so that it can be applied on any attributed trees. The approach improves the quality of mining processes output. Experiments were carried out to prove the proposed concepts and thoroughly evaluate the results. Intuitive testing on similarity was observed and a semantic-based document clustering process was implemented in accordance to the proposed approach.

The choice of pursuing this semantic approach is inspired by how humans are able to discover knowledge from text and to assess similarities between documents. Humans can analyze documents and judge similarities based on the analysis of contents. Moreover, when individuals are asked to perform some mining task (such as classifying, or clustering a document), they would do so even with partial

understanding. Thus, similarity assessments and document mining schemes can differ in performance as to how much they attempt to understand contents, and how they employ the understanding in finding similarities between documents. The experiments showed clearly that as more semantic information clues are augmented in the processing steps, the mining results are improved. This gives more motivation to exploit research findings in human-like intelligent text understanding and apply them to mine documents for better accessing, searching, retrieving, organizing, managing, and reasoning about information they contain.

This unique direction, however, has many challenges. Perhaps, the most difficult of all is its assumption that text written for human to read could be interpreted by machines. This assumption is quite questionable as the difficulties to analyze natural languages are immense and there is currently no complete natural language understanding system. NL expresses a vast and rich range of information. It encodes this information in a form that can be very ambiguous. This ambiguity can be found at all levels of analysis including morphological, syntactic, and pragmatic. Moreover, language is in constant change and expansion and that make it harder for the automatic techniques to keep-up with it. Stating this, however, does not mean that there are no successful attempts to build systems that can analysis text. The fields of natural language processing (NLP) and computational linguistics (CL) provide many mature and reasonable approaches to understand text that can be utilized and built upon. The use of NLP and CL seems to be the only choice, if meaningful and high quality document mining results are ever to be achieved. By providing rich representations of documents, even through rough and partial understanding, the document mining results are expected to surpass existing approaches. Moreover, document mining tasks, such as clustering, and classification do not need more than a

rough representation of meanings to produce acceptable results. In addition, the performance of the semantic understanding-based document mining systems is expected to proportionally improve as the underlying technologies of NLP and CL advance.

In this work, NLP and CL techniques were utilized to introduce and build the semantic graph model (SGM) representational scheme to represent meaning. The representation is based on semantic structures that translate readings of sentences to formulas that encapsulate their structures and semantics. It recognizes relationships between concepts and expresses their attributes. SGM for a document is the accumulation of all parsed sentences. Every valid sentence is represented as a directed acyclic attributed graph. The process of building the presentation starts by identifying predicate-argument structures, and then disambiguating and making implicit concepts meanings explicit and adding them to the structure. Thus, SGM can produce unambiguous and canonical interpretations of documents that are adequate to represent the meaning of sensible inputs. Using SGM representation enables comparison between modeled documents. The representation scheme is information-rich, and extendable to include different analysis processes output. Moreover, it could be manipulated efficiently, hence, it can be used for further mining processes efficiently and effectively.

To address the problem of comparing SGM representations a new similarity measure was proposed. The proposed measure is defined on the SGMs that are trees with multiple symbolic attributes, and is based on finding all distinct similarity common sub-trees. It is an inexact tree matching method suitable to compare large sized trees that can have scattered communalities between them. Thus, the measure reflects the level of commonality between trees without overrating. The measure is

normalized pseudo metric, as it was shown that it fulfilled most of the metric axioms. Moreover, the algorithm to compute the measurement is computationally tractable, i.e., has a polynomial time and space complexity.

---

## **7.2 Findings**

This work portrays a semantic understanding-based approach to tackle document mining. It proposes a framework for document mining systems that understand text to provide better performance than the-bag-of-words and the similar techniques. In the semantic-based approach, input texts are converted to a canonical knowledge representation using deep syntactic and semantic parsing. An appropriate similarity measure technique is employed to approximate meaning-based similarity distances between documents.

A case study of a mining task (i.e., semantic clustering of text documents) is considered. The study shows how the proposed approach is workable, and how it can be applied to mining tasks. Experimentally, significant improvements in all performance indices are achieved for the document clustering process. The approach outperforms conventional word-based clustering methods. Moreover, performance tends to improve as more semantic analysis is performed and augmented in the representation.

The developed system is composed of semantic components that use syntactic and semantic analysis output. Through this approach better clustering results can be achieved as more attempts to improve the accuracy of measuring the similarity between documents using semantic information clues. The analysis components are capable of identifying meanings and structures of linguistic inputs of documents. The

second important part of the system is similarity measure to assess distances between represented documents. The similarity measure affects the performance of document mining processes output as to how much of the semantic information it considers. The most accurate similarity can be achieved when the maximum amount of the semantic information is provided. Consequently, the quality of clustering is improved, and when semantic information is considered the approach surpasses traditional word based techniques, i.e., the vector space model.

---

### **7.3 Contributions**

The main contribution of this thesis lies on the introduction of the semantic notion as a basis for text mining, with an emphasis on semantic representation, similarity measure, and document clustering process for illustration purposes. Derived by the search for the most efficient and effective ways to perform document mining, new findings and facts were discovered including the possibility to have better text representation models (i.e., the semantic graph model, SGM), similarity estimation techniques for attributed-trees representation, and proof of effectiveness of the approach on real world application (semantic document clustering).

The SGM representational approach to meaning and the steps to build it were introduced. The approach is based on exploiting semantic structures to improve document mining effectiveness. The representation translates readings of sentences into formulas that encapsulate their structures and semantics. It identifies relationships between phrases and thus more precisely represents the text content. SGM for a document is the accumulation of all sentences parsed into their meaning representations. Essentially, each sentence representation is a directed acyclic

attributed graph of concepts and relations between them. The process to build the representation starts by identifying predicate-argument structures, and then disambiguating and making implicit meanings more explicit and embedding them in the structure. SGM representation can produce unambiguous and canonical interpretations for input sentences that can be ambiguous and verbose. The model produces single meaning representation structures that adequately represent the meaning of sensible inputs. Using SGM representation enables comparison between modeled documents. Moreover, the advantages of this representation scheme are its information richness, and openness to include various analysis and reasoning processes output. More importantly, is the ability of the structural model to be manipulated and managed easily. This is especially vital, as the aim from building the SGMs is to use them for further mining processes efficiently and effectively. One main concern is the problem of comparing SGM representations and measuring similarities.

A new technique and the algorithmic steps for calculating similarities between documents represented semantically as node-attributed trees was also introduced. The similarity measure is an inexact tree matching method. This measure is deemed to be suitable for measuring similarity between documents represented by the semantic graph model (SGM). The SGMs are essentially trees with symbolic node attributes that could be quite large in size and commonality between trees could be sparse. Thus, the measure is based on finding all distinct similarity common sub-trees to reflect the level of commonality between these trees without overrating. The measure is in normalized pseudo metric, as it fulfills most of the metric axioms. In addition, the pseudo-code for the algorithmic steps to compute the measurement was provided, and showed that it is computationally tractable, as it has a polynomial time and space

complexity. The similarity technique was tested on different pairs of documents to prove effectiveness. The technique showed more agreement with what humans conclude than other conventional word-based technique. Further experimental work that utilized the semantic-understanding approach to mine documents was presented.

A system was demonstrated that is composed of semantic components in an attempt to improve the accuracy of measuring the similarity between documents and using the similarity in applying the document clustering problem. By exploiting the semantic analysis findings, better clustering results could be achieved. The document analysis components that are capable of identifying the meaning structures of text in documents were implemented. The second part, and perhaps the most important, is to measure similarity between parsed documents. The measure has the most impact on the performance as how much of semantic information it considers. Using the maximum amount of the semantic information enables us to perform similarity calculations between documents in a robust and accurate way. The quality of clustering achieved using this model significantly surpasses the traditional vector space model based approach. Moreover, the merit of the design is that each component can be utilized and refined independently. Combining all of these components leads to better results, as justified by the results presented in Chapter 6. The system was tested against different standard clustering techniques and different data sets, and the findings encouraged to pursue the semantic understanding direction.

---

## **7.4 Future Extensions**

There can be many ways to extend this work in its different development stages; starting with data representation, going through the process of measuring

similarities, and ending with applying and implementing semantic-based mining processes that are effective and efficient. In all these areas, contributions could be furthered and refined. The following are points to explain some suggestions and incitements to pursue the proposed direction:

- **Extensions to the semantic representation model.** The semantic-based document representation scheme, SGM, is quite extendable to include different NLP techniques outputs. Thus, enriching the information content of the model is easily achievable. One interesting aspect that can be added is fuzzy values to represent the parsers' level of confidence. This is an important property as most text analysis aspects are fuzzy and ambiguous. Moreover, there can be other document representation schemes, all with their pros and cons, for individual investigation and fusion. There has been a fair number of NLP systems that employ text representational schemes to capture the meaning of linguistic inputs. Other schemes than predicate arguments-based can be utilized, such as semantic networks and frames, which are also slot-filler representations. In semantic networks, objects are represented as feature nodes in a graph, with named links to represent relations between these objects. In frame-based systems, objects are represented as feature-structures, which can also be represented as graphs. In these approaches, features are called slots, and values of these slots can either be atomic values or other embedded frames. It is noticeable that meanings represented in any of these approaches are essentially graphs that hold concepts and relations between them, thus, including other schemes is feasible and worth exploring.
- **Considering other data format.** Expressing semantics in documents based on contents other than text, such as images, voices, and videos, is another

interesting point to explore. Nowadays, much of available knowledge is expressed in a multimedia format. Though, advancement in distilling meanings from such data format is still in its early stages, it would be of a great benefit to the overall system performance to add semantic clues resulting from processing images, speech, or video.

- **Improving the similarity measure.** Coupling the chosen representation schemes with techniques to measure distances between documents represented as such, will definitely lead to more interesting findings. The similarity measure could be further enhanced in terms of efficiency and effectiveness. Similarity calculations rely on the significance of the inferred semantic information from the documents. Thus, the semantic-based similarity should consider all semantic clues in order to judge the closeness in meaning between documents.
- **Utilizing structural information of documents.** Considering structured or semi-structured text, where text locations, document layout, etc. adds to the overall understanding of contents is another possible direction to investigate. For instance, HTML-tagged documents are considered to be semi-structured. Though, they are designed to specify the layout of the documents, and to present them in a user-friendly manner rather than specifying the structure of the data in the document; it is still fairly easy to identify key parts of a document based on the tagging. There could be a utilization of the idea that some parts of the document are more informative than others, thus having different levels of significance based on where they appear in the document and the tags that surround them. For example, it would be more informative to treat the title of the document and the text body differently. This structuring

consideration can be incorporated when representing documents using SGM, and in measuring the similarity between them. For example, if we have two matching sentences of high significance in both documents, the similarity should reflect higher similarity than if the match were for low significance sentences. Titles, for instance, are much more informative than text matched in the body.

- **Application to different document mining tasks.** There are many mining processes that can serve as test-beds for the developed theoretical work. In these mining processes many avenues can be explored for improvements. An important part of the mining processes are the mining algorithms. One important benefit of the proposed model is that it can be used with any mining algorithm that can make use of the accurate pair-wise document similarity. Other directions could be to develop algorithms that work directly on the representation in fulfilling mining tasks requirements.
- **Building a large scale semantic-based mining system.** There can be a focus on developing the framework for a complete mining system. A system that is scalable and flexible to deal with real-world problems, such as Web mining. Efficiency and scalability are crucial for real-world applications. Possible directions would be to initially concentrate on small scale instances of problems, consider off-line applications where speed is not a high priority of concern, and then move to larger data sets and more demanding applications. The performance of the proposed system is reasonable for moderate-sized document sets, which could be applied to real-time online applications, such as clustering of Web search engine results. With this performance the

approach can be applied in many other online applications, not necessarily document clustering.

These were a few of the future research directions to extend this work. They all aim at continuing to improve on the semantic understanding approach to mine documents. This could be accomplished through investigating a new direction in document representations, similarity calculations and mining processes algorithms.

## Bibliography

1. Aas, K., and Eikvil, L., "Text categorisation: A survey," Technical Report 941, Norwegian Computing Center, 1999.
2. Ahuja, R. K., Magnanti, T. L., Orlin, J. B., *Network Flows*, Prentice-Hall, Upper Saddle River, NJ, 1993.
3. Ajram, K, *The Miracle of Islamic Science*. Knowledge House Publishers, 1992.
4. Allan, J., *Introduction to Natural Language Understanding*, 2nd edition., Addison-Wesley Publishing Co., 1995.
5. Baeza-Yates, R., and Ribeiro-Neto, B., *Modern Information Retrieval*, ACM Press, New York, 1999.
6. Belkin, N. J., and Croft, W. B., "Information filtering and information retrieval: Two sides of the same coin?," *Communications of the ACM*, vol. 35, no. 12, pp. 29-38, 1992.
7. Berkhin, P., "Survey of Clustering Data Mining Techniques," Technical Report, Accrue Software, 2002.
8. Berry, M. W., and Browne, M., *Understanding Search Engines: Mathematical Modeling and Text Retrieval*, SIAM, Philadelphia, 1999.
9. Berry, M. W., Dunais, S. T., and O'Brien, G. W., "Using Linear Algebra for Intelligent Information Retrieval," *SIAM Review* 37(4), pp. 573-595, 1995.
10. Blair, D. C., and Maron, M. E., "An evaluation of retrieval effectiveness," *Communications of the ACM*, 28: 289-299, 1985.
11. Boley, D., "Principal direction divisive partitioning", *Data Mining and Knowledge Discovery*, 2(4):325-344, 1998.

12. Boley, D., Gini, M., Gross, R., Han, S., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., and Moore, J., "Partitioning-based clustering for web document categorization", *Decision Support Systems*, 27:329-341, 1999.
13. Boley, D., Gini, M., Gross, R., Han, S., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., and Moore, J., "Document categorization and query generation on the World Wide Web using WebACE", *AI Review*, 13(5-6):365-391, 1999.
14. Brill, E., and Mooney, R. J., "An Overview of Empirical Natural Language Processing," *The American Association for Artificial Intelligence*, 18(4): pp. 13-24, 1997.
15. Bunke, H., "On a Relation between Graph Edit Distance and Maximum Common Subgraph," *Pattern Recognition Letters*, vol. 18, no. 8, pp. 689-694, 1997.
16. Bunke, H., and Shearer, K., "A graph distance metric based on the maximal common subgraph," *Pattern Recognition Letters*, 19:255–259, 1998.
17. Chakrabarti, S., *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann Publishers, 2003.
18. Cios, K., Pedrycs, W., and Swiniarski, R., *Data Mining Methods for Knowledge Discovery*. Kluwer Academic Publishers, 1998.
19. Christmas, W., Kittler, J., and Petrou, M., "Structural matching in computer vision using probabilistic relaxation", *IEEE Trans. PAMI*, 8:749-764, 1995.
20. Croft, W. B., Turtle, H. R., and Lewis, D. D., "The use of phrases and structured queries in information retrieval", *Proceedings of ACM SIGIR*, pp. 32-43, October, 1991.

21. Cross, A., Wilson, R., and Hancock, E., "Genetic search for structural matching", In B. Buxton and R. Cipolla, editors, *Computer Vision - ECCV '96*, LNCS 1064, pages 514-525. Springer Verlag, 1996.
22. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., "GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications," *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*. Philadelphia, July 2002.
23. Cutting, D.; Karger, D.; Pedersen, J.; and Tukey, J.: "Scatter/gather: A cluster-based approach to browsing large document collections." In *16th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 126-135, 1993.
24. Dasarathy, B.V.: "Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques." *McGraw-Hill Computer Science Series*. IEEE Computer Society Press, Las Alamitos, California (1991).
25. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer T. K., and Harshman, R., "Indexing by Latent Semantic Analysis," *Journal of the American Society for Information Science*, 1990.
26. Dempster, A. P., Laird N. M., and Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm". *Journal of the Royal Statistical Society, Series B (Methodological)*, 39(1):1-38, 1977.
27. Dickinson, S. J., Pentland, A. P., and Rosenfeld, A., "3-D shape recovery using distributed aspect matching," *PAMI*, 14(2):174-198, 1992.
28. Dumais, Susan T., Furnas, G. W., Landauer, Thomas K., Deerwester, Scott, and Harshman, Richard, "Using Latent Semantic Analysis to Improve Access

- to Textual Information," In Proceedings of Computer Human Interaction '88 , 1988.
29. Eikvil, L., "Information Extraction from World Wide Web – A Survey," Technical Report 945, Norwegian Computing Center, July, 1999.
  30. Escudero, G, Marquez, L., and Rigau, G., "Boosting Applied to Word Sense Disambiguation." In Proceedings of the 12th European Conference on Machine Learning, ECML, Barcelona, Spain, 2000.
  31. Eshera, M. A., and Fu, K.-S. "An image understanding system using attributed symbolic representation and inexact graph-matching," PAMI, 8:604–618, 1986.
  32. Faloutsos, C., and Oard, D., "A Survey of Information Retrieval and Filtering Methods", Technical Report, Information Filtering Project, University of Maryland, College Park, MD, 1996.
  33. Fayyad, U., and Uthurusamy, R., "Data mining and knowledge discovery in databases: Introduction to the special issue," Communications of the ACM, 39(11), November, 1999.
  34. Feng, J., Laumy, M., and Dhome, M., "Inexact matching using neural networks", In E. Gelsema and L. Kanal, editors, Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems, pages 177-184. North-Holland, 1994.
  35. Fernandez, M. L., and Valiente, G., "A graph distance metric combining maximum common subgraph and minimum common supergraph," Pattern Recognition Letters, 22:753–758, 2001.

36. Furnas, G. W., Landauer, T. K., Gornez, L. M., and Dumais, S. T., "The vocabulary problem in human-system communication," *Communications of the ACM*, 30(11): pp. 964-971, November, 1987.
37. Gonzalo, J., Verdejo, F., Ghugur, I., and Cigarran, J., "Indexing with WordNet Synsets can Improve Text Retrieval," In *Proceedings of the COLING/ACL '98 Workshop on Usage of WordNet for NLP*, 1998.
38. Hammouda, K., and Kamel, M. "Phrase-based document similarity based on an index graph model," In *Proceedings of the 2002 IEEE Int'l Conf. on Data Mining (ICDM'02)*, 2002.
39. Hartigan, J. A., and Wong, M. A. "A K-means clustering algorithm". *Applied Statistics*, 28:100-108, 1979.
40. Hasan, M., Matsumoto, Y. "Document Clustering: Before and After the Singular Value Decomposition", Sapporo, Japan, Information Processing Society of Japan (IPJS-TR:99-NL-134.) pp. 47-55, 1999.
41. Hersh, W. R., Hickam, D. H., and Leone, T., J., "Words, concepts, or both: Optimal indexing units for automated information retrieval." In *Frisse ME (ed) Proceedings of the 16th Annual SCAMC*, 644-648, 1992.
42. Hidovic', D., and Pelillo, M., "Metrics for attributed graphs based on the maximal similarity common subgraph," *Int. J. Pattern Recognition Artif. Intell.*, 2004.
43. Hill, D.R.: "A vector clustering technique." In Samuelson, ed.: *Mechanized Information Storage, Retrieval and Dissemination*. North-Holland, Amsterdam (1968).
44. Hobbs, J. R.: "Resolving pronoun references," *Lingua*, 44, 311-338. Reprinted in Grosz et al, 1986.

45. Honkela, T., Kaski, S., Lagus, K., and Kohonen, T.” “WEBSOM self-organizing maps of document collections.” In Proceedings of WSOM'97, Workshop on Self-Organizing Maps, pages 310-315, Espoo, Finland, June 1997.
46. Hopcroft, J. E., Motwani, R., and Ullman, J. D., *Introduction to Automata Theory, Languages, and Computation*, Pearson Addison Wesley; 2nd edition, 2000.
47. Hotho, A., Staab, S., and Stumme, G., “Wordnet improves Text Document Clustering.” In Proceeding of the Semantic Web Workshop at SIGIR-2003, 26th Annual International ACM SIGIR Conference, 2003.
48. Ide, N., and Veronis, J., “Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art”. *Computational Linguistics* 24:1-40, 1998.
49. Jain, A. K., and Dubes, R. C., “Algorithms for Clustering Data”, Prentice Hall, Englewood Cliffs NJ, U.S.A, 1988.
50. Jang, J., Sun, C., and Mizutani, E., *Neuro-Fuzzy and Soft Computing - A Computational Approach to Learning and Machine Intelligence*. Prentice Hall, 1997.
51. Jurafsky, D., and Martin, J. H., *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Prentice Hall, Inc., 2000.
52. Kennedy, C., and Boguraev, B., "Anaphora for everyone: Pronominal anaphora resolution without a parser," In COLING-96, Copenhagen, pp. 113-118, 1996.
53. Kohonen, T.: *Self-Organizing Maps*, Springer, Berlin, Heidelberg, 2001.

54. Krishnapuram, R.; Joshi, A.; and Yi, L.: "A fuzzy relative of the k-medoids algorithm with application to web document and snippet clustering." In IEEE International Conference on Fuzzy Systems, Korea, August 1999.
55. Lappin, S., and Leass, H., "An algorithm for pronominal anaphora resolution", *Computational Linguistics*, 20(4), 535-561, 1994.
56. Lawrence, S., "Online or invisible?". *Nature*, vol. 411, No. 6837, pp. 521, 2001.
57. Lee, D. L., Chuang, H., and Seamons, K., "Document Ranking and the Vector-Space Model," *IEEE Computer*, Theme Issues on Assessing Measurement, 14(2):67-75, April/May, 1997.
58. Lesk, M. E., "Automatic sense disambiguation using machine readable dictionaries: How to tell a pine from an ice cream cone", In proceedings of the Fifth International Conference on Systems Documentation, Toronto, CA, pp. 24-36. ACM.
59. Lesk, M., "The Seven Ages of Information Retrieval," Occasional Paper, International Federation of Library Associations and Institutions, 1996.
60. Lewis, D. D., Yang, Y., Rose, T., and Li, F., "RCV1: A New Benchmark Collection for Text Categorization Research." *Journal of Machine Learning Research*, 5:361-397, 2004.
61. Lewis, David, "General Semantics," In Davidson, D. and Harman, G. (Eds.), *Natural Language Semantics*, pp. 169-218. D. Reidel, Dordrecht.
62. Lewis, David, "Introduction to Dying for Information," 1996, [www.reuters.com/rbb/research/dfiforframe.htm](http://www.reuters.com/rbb/research/dfiforframe.htm).
63. Lin, D., and Goebel, R., "Context-free grammar parsing by message passing," In proceedings of PAACLING 93, 1993.

64. Ljungstrand, P.; and Johansson, H. "Intranet indexing using semantic document clustering." Master Thesis. Department of Informatics, Göteborg University, 1997.
65. Macskassy, S. A., Banerjee, A, Davison, B. D., Hirsh, H., "Human Performance on Clustering Web Pages: A Preliminary Study," In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, 1998.
66. Martinez, A. R., and Wegman, E. J., "Text Stream Transformation for Semantic-Based Clustering", Computing Science and Statistics, 34, 2002 Proceedings.
67. Mauldin, M. L., *Conceptual Information Retrieval: A Case Study in Adaptive Partial Parsing system*, Kluwer Academic Publishers, 1991.
68. Mauldin, M. L., "Performance in ferret: a conceptual information retrieval system", Proceedings of ACM SIGIR, pp. 347-355, October, 1991.
69. Matula, D. W., "An algorithm for subtree identification", SIAM Review, 10:273-274, 1968.
70. Michell, T.: Machine Learning. McGraw Hill, 1997.
71. Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K., "Introduction to WordNet: An On-line Lexical Database". Cognitive Science Laboratory, Princeton University, 1993.
72. Mostafa, J., Mukhopadhyay, S., Lam, W., Palakal, M., "A Multi-level Approach to Intelligent Information Filtering: Model, System & Evaluation," ACM Transactions on Information Systems, 15(4): pp. 368-399, 1997.
73. Murray, B. H.; and Moor, A., "Sizing the Internet," White Paper, Cyveillance, 2000.

74. Nichols, J., *Multimedia and Hypertext: The Internet and Beyond*, Morgan Kaufmann, 1995.
75. Peat, F. D.: *Artificial Intelligence - How Machines Think*, Baen Enterprises, 1985.
76. Podder, S.; Shaban, K.; Sun, J.; Karray, F.; Basir, O.; and Kamel, M. "Performance improvement of automatic speech recognition systems via multiple language models produced by sentence-based clustering," In proceedings of IEEE International Conference on Natural Language Processing and Knowledge Engineering, Beijing, October, 2003.
77. Porter, M. F. "An algorithm for suffix stripping," *Program*, 14(3), pp. 130-137, 1980.
78. Rau, L. F., and Jacobs, P. S., "Creating segmented databases from free text for text retrieval", *Proceedings of ACM SIGIR*, pp. 337-346, October 1991.
79. Reuters. "Dying for Information: An Investigation into the Effects of Information Overload in the USA and Worldwide", Based on research conducted by Benchmark Research. London: Reuters Limited, 1996.
80. Salton, G., Allan, J., and Buckley, C., "Automatic structuring and retrieval of large text files", *Comm. of ACM (CACM)*, 37(2):97-108, February 1994.
81. Salton, G., and McGill, M. J., *Introduction to Modern Information Retrieval*, McGraw-Hill, 1984.
82. Salton, G., Wong, A., and Yang, C., "A vector space model for automatic indexing," *Communications of the ACM*, 18(11), pp.613-620, 1975.
83. Schulze, B. M., Heid, U., Gaschler, J., Grefenstette, G., Rooth, M., Schiller, A., Schmid, H., and Teufel, S., "Comparative State-of-the-Art Survey and Assessment Study of General Interest Corpus-oriented Tools," *Decide*,

- deliverable d-1b i, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, 1994.
84. Siddiqi, K., Shokoufandeh, A., Dickinson, S. J., and Zucker, S. W, "Shock graphs and shape matching," *Int. J. Computer Vision*, 35(1):13–32, 1999.
  85. Shaban, K., *Information Fusion in a Cooperative Multi-Agent System for Web Information Retrieval*, Master of Science Thesis, University of Guelph, 2002.
  86. Shannon, C.E.: "A Mathematical Theory of Communication", *Bell Syst. Tech. J.*, 27, 379-423, 623-656, 1948.
  87. Soderland, S., *Learning Information Extraction Rules for Semistructured and Free Text*. *Machine Learning*, 1999.
  88. Steinbach, M.; Karypis, G.; and Kumar, V. "A comparison of document clustering techniques." *KDD-2000 Workshop on TextMining*, August 2000.
  89. Skydsgard, J. E., *Varro the Scholar*. *Analecta Ramana Instituti Danici*, Copenhagen, 1968.
  90. Suen, C., "N-gram statistics for natural language understanding and text processing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2), pp.164-172, 1979.
  91. Suganthan, P., Tesh, E., and Mital, D., "Pattern recognition by graph matching using Potts MFT neural networks", *Pattern Recognition*, 28:997-1009, 1995.
  92. Text Analysis International, Inc. "Integrated Development Environments for Natural Language Processing", *White Paper*, <http://www.textanalysis.com/>, October 2001.
  93. Text Analysis International, Inc. "Multi-Pass Multi-Strategy NLP ", *White Paper*, <http://www.textanalysis.com/>, October 2003.

94. Torsello, A., Hidovic, D., Pelillo, M., "A new polynomial-time metric on attributed trees", Proceedings of European Conference on Computer Vision, Part 4 (Lecture Notes in Computer Science, vol. 3024), T. Pajdla and J. Matas (eds). Springer-Verlag, pages 414 - 427.
95. Torsello, A., Hidovic, D., Pelillo, M., "Four metrics for efficiently comparing attributed trees", 17th International Conference on Pattern Recognition, Cambridge (UK) 2004. Volume 2, pages 467-470.
96. Torsello, A., Hidovic-Rowe, D., Pelillo, M., "Polynomial-Time Metrics for Attributed Trees", IEEE Transaction on Pattern Analysis and Machine Intelligence Vol. 27, No. 7 (2005), pages 1087-1099.
97. Valiente, G., "An efficient bottom-up distance between trees", in Proc. Int. Symp. String Processing Information Retrieval, pp. 212–219, 2001.
98. van Rijsbergen, C. J.: *Information Retrieval*, Woburn Massachusetts. Butterworths, 1979.
99. Verma, R. M., and Reyner, S. W., "An analysis of a good algorithm for the subtree problem", SIAM Journal on Computing, 18, 5, pp. 906-908, 1998.
100. Wallis, W. D., Shoubridge, P., Kraetz, M., and Ray, D., "Graph distances using graph union," Pattern Recognition Letters, 22:701–704, 2001.
101. Wang, Y.-K., Fan, K.-C., and Homg, J.-T., "Genetic-based search for error-correcting graph isomorphism", IEEE Trans. SMC, 27(4):588-597, 1997.
102. Williams, M., Wilson, R., and Hancock, E., "Deterministic search for relational graph matching" Pattern Recognition, 32: 1255-1271, 1999.
103. Wilson, R., and Hancock, E., "Graph matching by discrete relaxation", In E. Gelsema and L. Kanal, editors, Pattern Recognition in Practice N:

- Multiple Paradigms, Comparative Studies and Hybrid Systems, pages 165-176. North-Holland, 1994.
104. Winograd, T., and Flores, F., *Understanding Computers and Cognition : A New Foundation for Design*, Addison-Wesley Pub Co., 1995.
105. Yang, Y., and Pedersen, J., "A Comparative Study on Feature Selection in Text Categorization," In Proceeding of the 14th International Conference on Machine Learning, ICML, pp. 412-420, Nashville, TN, 1997.
106. Zhang, K, and Shasha, D., "Simple Fast Algorithms for the Editing Distance Between Trees and Related Problems," Society for Industrial and Applied Mathematics, SIAM J. Comput., Vol. 18(6):1245-1262, 1989.
107. Zhang, K., Statman, R., and Shasha, D., "On the editing distance between unordered labeled trees," Inform. Process. Letters, 42:133-139, 1992.
108. Zhang, K., Wang, J., and Shasha, D., "On the editing distance between undirected acyclic graphs," Int. J. Found. Computer Sci., 7(1):43-57, 1996.