Semantic Similarity Methods in WordNet and Their Application to Information Retrieval on the Web

Varelas Ioannis

June 30, 2005

Contents

List of Tables

List of Figures

Abstract

Semantic Similarity relates to computing the similarity between concepts which are not lexicographically similar. This is an important problem in Natural Language Processing and Information Retrieval Research and has received considerable attention in the literature. Several algorithmic approaches for computing semantic similarity have been proposed. We investigate approaches for computing semantic similarity by mapping terms or concepts to an ontology and by examining their relationships in that ontology. Comparing concepts that belong to different ontologies is far more difficult problem. Some of the most popular semantic similarity approaches are implemented and evaluated based on WordNet as the underlying reference ontology. We also propose a method for comparing terms in different ontologies. In this work we examined similarity between terms from WordNet and MeSH (medical) ontologies.

Building upon the idea of semantic similarity we also propose an information retrieval methodology capable of detecting similarities between documents containing semantically similar but not necessarily identical terms. Our proposed Information Retrieval model has been evaluated for retrieval of images and documents on the web. The experimental results demonstrated that our proposed model (although slower) achieves significant performance improvements compared to the state-of-the-art approach based on the Vector Space Model.

Acknowledgements

This research was supported by the Intelligent Systems Laboratory of the Techical University of Crete. I would like to thank Dr. Petrakis for the advice, encouragement and support he provided to me in supervising this thesis effort. We would also like to thank Dr. Millios, for his critical analyses and recommendations. Special thanks go to Epimenidis Voutsakis for his technical advice and recommendations and Rautopoulou Paraskeuh for his invaluable contribution in this thesis.

Chapter 1

Introduction

Information retrieval (IR) in the subject of intensive research efforts during the last twenty years [4]. The purpose of information retrieval is to assist users in locating information they are looking for. Information retrieval is currently being applied in a variety of application domains from database systems to web information search engines. The main idea is to locate documents that contain terms the users specify in their queries. Retrieval, by classical information retrieval models (eg. Vector Space, Probabilistic, Boolean), is based on plain lexicographic term matching between terms (eg. a query and a document term are considered similar if they are lexicographically the same). However, plain lexicographic analysis and matching is not generally sufficient to determine if two terms are similar and consequently whether two documents are similar. Two terms can be lexicographically different but have the same meaning (eg. they are synonyms) or they may have approximately the same meaning (they are semantically similar). The lack of common terms in two documents does not necessarily mean that the documents are irrelevant. Similarly, relevant documents may contain semantically similar but not necessarily the same terms. Semantically similar terms or concepts may be expressed in different ways in the documents and the queries, and direct comparison is not effective (eg. Vector Space Model(VSM) will not recognize synonyms or semantically similar terms). In this work we propose discovering semantically similar terms using ontology, and specifically WordNet [1].

WordNet is a vocabulary and a lexical ontology that attempts to model the lexical knowledge of a native speaker of English into a taxonomic hierarchy. Entries (i.e., terms or concepts) are organized into *synsets* (i.e., lists of synonym terms or concepts), which in turn are organized into senses (i.e., different meanings of the same term or concept). There are also different categorizations corresponding to nouns, verbs, adverbs etc. Each entry is related to entries higher or lower in the hierarchy by different types of relationships. The most common relationships are *Hyponym/Hypernym* (i.e., is-a relationships), and *Meronym/Holonym* (i.e., Part-Of relationships). In the following, we only use the nouns and the Hyponym/Hypernym relationships from WordNet to enhance vector representations.

In this thesis we study several Semantic Similarity methods utilizing ontologies in order for estimating Semantic Similarity between terms. We analyze the use of similarity methods in WordNet ontology and building upon the method proposed by Rodriguez [45], we propose a new method that can be used to compute the semantic

[1] http://wordnet.princeton.edu

similarity between terms that belong to the same or different ontologies. This is a far more difficult problem compared to the one of measuring similarity between terms from a single ontology. We also an architecture that can be used for the evaluation of results by different semantic similarity methods. In addition, we propose the use of such methods for information retrieval on the web by presenting a new IR approach which makes use of Semantic Similarity between terms and can be used in conjunction with any Web Search Engine for enhancing performance of retrieval.

Summarizing, the contributions of the proposed work are

1. A framework for evaluating the performance of various semantic similarity methods is implemented

2. Building upon the method by Rodriguez [45] we propose a new method that can be used for comparing terms from the same or different ontologies

3. We propose an information retrieval model based on the integration of Semantic Similarity methods within a vector like representation of terms

4. We evaluate our proposed IR model in retrieving documents and images on the web. The performance results demonstrated significant improvements in precision and recall over the state-of-the-art IR approach based on the Vector Space Model.

Several methods for determining semantic similarity between terms have been proposed in the bibliography and are divided in four main categories (detailed description of each category and for the methods that belong in each one can be found in Chapter 2)

1. **Edge Counting Methods :** These methods measure the similarity between two concepts c_1, c_2 by determining the path linking the terms in the taxonomy and the position of the terms in the taxonomy

2. **Information Content Methods :** In this category, similarity measures are based on the *Information content* of each concept

3. **Feature based Methods :** Measures that consider also the features of the terms in order to compute similarity

4. **Hybrid methods :** Those methods combine ideas from the above three approaches in order to compute semantic similarity between c_1 and c_2

We also distinguish between methods assuming that the terms which are compared belong to the same ontology (single ontology approaches) and methods capable of comparing terms from a different ontology (cross ontology approaches). Because the structure and information content between different ontologies cannot be compared directly, cross ontology approaches are mainly based on lexicographic content matching between terms and of their relationships with other terms. For example,

two terms are similar if they have similar spelling or definition or they are related with other terms which are similar (with their similarity defined in the same way). Notice also that these methods may also be used to measure semantic similarity between terms from the same ontology.

In this work, we try to determine which method performs better, in terms of providing more reasonable results when compared to result obtained by humans. Another problem studied, is the one of determining similarity of terms that belong in different ontologies. For example how much alike are the terms "alcohol" of the WordNet ontology with "alcohol" of the MeSH [2] ontology? WordNet is a general English language ontology while MeSH is a medical ontology. Cross ontology similarity of terms can be used in order to provide information about a subject that is best described by using terms from another ontology. A common application would be a native user trying to find information about an illness.

As part of this work, we built on the similarity method proposed by Rodriguez [45] and propose a modified version of this method that can be used both for determining similarity between concepts within the same ontology (in our case WordNet) or across ontologies. This method has proved to perform better than the original one both in all the experiments (single and cross ontology) and this is a first contribution of this work.

All methods have been implemented (11 methods total) and a complete prototype system has been developed as part of this work, in order to determine the semantic similarity. Given two terms, the user selects a similarity method and the system returns the semantic similarity of the two terms. Several options are available

- **Sense selection :** The user has the option to select which sense of the term to compare (one specific or all senses)
- **Similarity method selection :** A list of 10 similarity methods is available from all categories with option to add new methods as easy as inserting a java class file in the system
- **Ontology selection :** Our system is Ontology independent. The methods can be used within any ontology that conforms to a specific schema or that can be mapped to this schema
- **Cross ontology similarity :** Select two terms from different ontologies and compare them
- **API :** Furthermore a complete API was developed in order to be used from anyone who wants to make use of the system functionality

The system is available to the public at **http://alchimix.intelligence.tuc.gr/ontoss**

In the second part of this work, we propose a model that can be used in modern IR systems in order to retrieve results, that exploits the issue of semantic similarity for enhancing the performance of retrieval. The main idea behind our approach is that

[2] MeSH is an ontology for medical terms developed by the National Library of Medicine (NLM)

terms in a document or query representation are no longer considered as independent but they are related by virtue of their semantic similarity. This also suggests the idea of enhancing a text representation with other semantic similar terms. Finally to measure similarity between text representation, we abandon the idea of vector similarity and we introduce a model based on a *Similarity Matrix* that better captures the notion of dependency (or similarity) between non-identical terms. In our model, terms of queries and documents are treated as concepts when compared with others and as of this we overcome the binary in nature similarity constrain of lexicographic models, as the Vector Space Model(VSM). We calculate the similarity of each query term which each document term (documents that have been already retrieved by standard methodologies) resulting in a *Similarity Matrix*. Adding all elements of the matrix results in nothing else than the semantic similarity of the user query with the compared document. Furthermore, we make use of the *semantic neighborhood* aspect in order to enrich the user query with terms that possibly interest the user.
Briefly, the proposed model has the following characteristics

1. *Focuses the query in a specific area of interest*
2. *Expands the Query Vector* with other semantic similar terms
3. *Re-weights the Query Terms* based on their semantic similarity. This mechanism also computes the weights for terms that did not originally existed in the text
4. *Ranks the results according to our proposed Similarity Matrix*

The rest of this work in organized in the following way:
Chapter 2 will provide the necessary background and a critical analysis of related work.
Chapter 3 will define the problem semantic similarity problem, introduce our similarity model, describe the developed system and present the experimental results.
Chapter 4 will define the problem of information retrieval on the web, introduce our document model based on semantic similarity, describe the system used in order to make the experiments and present the experiment results.
Chapter 5 will present the criticism and some thoughts for future work.
Appendix A describes some technical issues about the developed software.

Chapter 2

Background and Related Work

This chapter introduces the semantic similarity problem, highlights some of the key technical issues and discusses related work. It concludes by identifying critical questions that have not yet been adequately addressed in the literature.

2.1 Writing and Using Ontologies

Ontologies can be regarded as general tools of information representation on a subject. They can have different roles depending on the application domain and the level of specificity at which they are being used. In general, ontologies can be distinguished into *domain ontologies*, representing knowledge of a particular domain, and *generic ontologies* representing common sense knowledge about the world [51].

There are several examples of general purpose ontologies available including: (a) WordNet[1] [5, 29] attempts to model the lexical knowledge of a native speaker of English. It can be used as both a thesaurus and a dictionary. English nouns, verbs, adjectives, and adverbs are organized into synonym sets, called *synsets*, each representing a concept. (b) SENSUS[2] [19] is a 90,000-node concept thesaurus (ontology) derived as an extension and reorganization of WordNet. Each node is SENSUS represents one concept, i.e., one specific sense of a word, and the concepts are linked in a IS-A hierarchy, becoming more general towards the root of the ontology. (c) The Cyc[3] Knowledge Base (KB) [35, 41] consists of terms and assertions relating those terms, contains a vast quantity of fundamental human knowledge: facts, rules of thumb, and heuristics for reasoning about the objects and events of everyday life. At the present time, the Cyc KB contains nearly two hundred thousand terms and several dozen hand-entered assertions about/involving each term.

Examples of domain specific ontologies include among others ontologies designed around (a) medical concepts such as UMLS[4] [33], SNOMED[5], MESH[6] [32], (b) genomic data such as GO[7] [6, 13] and (c) spatial data such as SDTS[8]. The Unified

[1] http://www.cogsci.princeton.edu/~wn/
[2] http://mozart.isi.edu:8003/sensus2/
[3] http://www.cyc.com/, http://www.opencyc.org/
[4] http://www.nlm.nih.gov/research/umls
[5] http://www.snomed.org
[6] http://www.nlm.nih.gov/mesh
[7] http://www.geneontology.org
[8] http://mcmcweb.er.usgs.gov/sdts/

Medical Language System (UMLS) contains a very large, multi-purpose and multilingual thesaurus concerning biomedical and health related concepts. In particular, it contains information about over 1 million biomedical concepts and 2.8 million concept names from more than 100 controlled vocabularies and classifications (some in multiple languages) used in patient records, administrative health data, bibliographic and full-text databases and expert systems. Furthermore, all the names and meanings are enhanced with attributes and inter-term relationships. UMLS includes other meta thesaurus source vocabularies, such as Medical Subject Headings (MeSH) that is the National Library of Medicine's vocabulary thesaurus. MeSH consists of sets of terms naming *descriptors* in a hierarchical structure. Gene Ontology (GO) is a structured network of defined terms that describe gene proteins and concerns all organisms. The Spatial Data Transfer Standard (SDTS) contains an ontology used to describe the underlying conceptual model and the detailed specifications for the content, structure, and format of spatial data, their features and associated attributes. Concepts in SDTS are commonly used on topographic quadrangle maps and hydrographic charts.

Intensive research efforts during the last few years have focused on providing tools for coherent, unambiguous and easy manipulation of information represented as ontologies. Such tools include languages providing the necessary syntax for the efficient representation of concepts and of their semantics as well as tools in the form of algorithms and graphic interfaces for viewing and manipulating the content of ontologies.

Languages for Writing Ontologies

The Resource Description Framework (RDF[9]) is a language for representing information about resources in the Web [2, 18]. It is particularly intended for representing meta data about Web resources, such as the title, author, and modification date of a document. RDF can also be used to represent information about things that can be identified on the Web, even when they cannot be directly retrieved, as for example information about items available from on-line shopping facilities (e.g., information about specifications, prices, and availability). RDF is intended for situations in which this information needs to be processed by applications, as it provides a common framework for expressing this information so it can be exchanged between applications without loss of meaning. RDF is based on the idea of identifying things using Web identifiers (called Uniform Resource Identifiers, or URIs), and describing resources in terms of simple properties and property values, which enables RDF to represent simple statements about resources as a graph of nodes and arcs representing the resources, and their properties and values. RDF also provides an XML-based syntax (called RDF/XML) for recording and exchanging these graphs. Although, RDF provides a way to express simple statements about resources, using named properties and values, it does not define the terms used in those statements. That is the role of RDF Schema (RDF-S[10]) that provides the facilities needed to describe such classes and properties, and to indicate which classes and properties are expected to be used

[9] http://www.w3.org/RDF

[10] http://www.w3.org/TR/rdf-schema

together [24]. The RDF-S facilities are themselves provided in the form of an RDF vocabulary; that is, as a specialized set of predefined RDF resources with their own special meanings.

DAML+OIL[11], which was the result of an initial joint effort by US and European researchers, is a semantic markup language for Web resources [15, 14]. It builds on RDF and RDF-S, and extends these languages with richer modeling primitives. In particular, DAML+OIL assigns a specific meaning to certain RDF triples. The model-theoretic semantics[12] specify exactly which triples are assigned a specific meaning, and what this meaning is.

The WWW Consortium (W3C) created the Web Ontology Working Group to develop a semantic markup language for publishing and sharing ontologies and the resulting language is Web Ontology Language (OWL[13]). OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and thus OWL goes beyond these languages in its ability to represent content on the Web. OWL is a revision of the DAML+OIL Web ontology language, adding more relations between classes (e.g., disjointness), cardinality (e.g., "exactly one"), equality, more properties, more characteristics of properties (e.g., symmetry), and enumerated classes.

To conclude, if machines are expected to perform useful reasoning tasks on Web resources, some language must be used in order to go beyond raw data, to express the semantics of the data and to extract knowledge from it. A summary of the existent recommendations related to the Semantic Web follows.

- XML provides a syntax for structured documents, but imposes no semantic constraints on the meaning of these documents.
- RDF is a data model describing resources and relations between them and provides simple semantics for this data model. The data models can be represented in an XML syntax.
- RDF-S is a vocabulary for describing properties and classes of RDF resources.
- DAML+OIL assigns specific meaning to certain RDF triples.
- OWL adds more vocabulary for describing properties and classes.

There are also efforts for describing the semantics of Web services, resulting in the DAML-S[14] [46] and OWL-S[15] [17] languages.

[11] http://www.daml.org/language/
[12] http://www.daml.org/2000/12/daml+oil.daml
[13] http://www.w3.org/TR/owl-features
[14] http://www.daml.org/services
[15] http://www.mindswap.org/2004/owl-s/

Tools for Manipulating Ontologies

Examples of tools for manipulating ontologies include Protege-2000[16] [34] and Chimaera[17] [25, 26, 27]. Protege-2000 allows users to construct domain ontologies, contains a platform that can be extended with graphical widgets for tables, diagrams, animation components to access other knowledge-based systems embedded applications, and has a library that other applications can use to access and display knowledge bases. Chimaera is a software system that supports users in creating and maintaining distributed ontologies on the Web. It supports two major functions that is merging multiple ontologies together and diagnosing[18] individual or multiple ontologies. It also provides users with tasks such as loading knowledge bases in different formats, reorganizing taxonomies, resolving name conflicts, browsing ontologies and editing terms.

2.2 WordNet

WordNet [19] is an on-line lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory. It can also be seen as an ontology for natural language since the categories are connected by various kinds of semantic links, e.g. generalization, similar, exclusion, member, part and substance. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets. Examples of these relationships are:

Synonymy: Similarity in meaning of words, which is used to build concepts represented by a set of words. This set of words is called "Synset".

Antonymy: Dichotomy in meaning of words - mainly used for organizing adjectives and adverbs.

Hyponymy: Is-a relationship between concepts. This is-a hierarchy ensures the inheritance of properties from super concepts to sub concepts.

Meronymy: Part-of relationship between concepts.

Morphological Relations: These relations are used to reduce word forms.

It was developed by the Cognitive Science Laboratory [20] at Princeton University. WordNet contains around 100.000 word meanings organized in a taxonomy.

Figure 2.1 contains all relations in WordNet and their representation symbol. Since in this work only nouns are used [21] only noun synsets will be discussed here.

[16] http://protege.stanford.edu

[17] http://www.ksl.stanford.edu/software/chimaera/

[18] Tool used as an ontological sketchpad, and creating classes for example.

[19] http://wordnet.princeton.edu

[20] http://www.cogsci.princeton.edu/

[21] While the words in each grammatical class are used with a particular purpose, it can be argued that most of the semantics is carried by the noun words

The synsets for the other parts of speech are construed in a slightly different fashion. The interested reader may consult [10] chapter 2 for adjectives and adverbs, and chapter 3 for verbs. The lexical items in a synset are not absolute synonyms, rather they are interchangeable within some context. The synsets for the different parts-of-speech are stored in separate files; data.noun, data.verb etc. When using WordNet, the user types in a query, e.g., and then the sense of that query is looked up. The user can then choose to look at synsets related to the one shown.

Relations and pointer symbols

Noun		Verb		Adjective		Adverb	
Antonym	!	Antonym	!	Antonym	!	Antonym	!
Hyponym	~	Troponym	~	Similar	&	Derived from	\
Hypernym	@	Hypernym	@	Relational adj.	\		
Meronym	#	Entailment	*	Also see	^		
Holonym	%	Cause	>	Attribute	=		
Attribute	=	Also see	^	Participle	<		

Figure 2.1: WordNet Relations and their Symbols

2.2.1 A little more about the synsets in WordNet

All noun synsets contain the following data, which can bee seen in Figure 2.2

- A synset ID (eg. ”03013456”). This is used by all other synsets when referring to this one.
- A two-digit code (eg. ”06”) between ”03” and ”28” identifying the synset as descending from one of 25 so called unique beginners. The unique beginners are top nodes in the WordNet hierarchy and can be seen as a sort of semantic prime. For example, unique beginners are ”05 - animal, fauna”, ”20 - plant, flora”, ”27 - substance” etc. In fact, there is another synset for ”crane”, this one descending from unique beginner ”05 - animal, fauna” and with a different description focusing on the animal-sense of the word. This is the case with all polysemous words; they have one synset for every sense of the word, usually descending from different unique beginners.
- A ”part-of-speech” tag. eg. ”n” for nouns.
- A number (eg. ”01”) indicating how many lexical items the synset contains (in our example only ”crane”).
- Pairs containing one lexical item and a number (eg. crane 0) indicating where in the file **index.sense** this sense of the word can be found. **index.sense** provides an alternate way of searching for synsets, but this will not be used in this work.

- A three-digit number specifying how many other synsets the synset points to.
- A number of relation pointers. The first relation is always denotes the hypernym synset (eg. "@ 03528075"), which in this case is "wading bird". Other various relation-type and synset pairs usually follow. In our example the "∼" stands for a hyponym relation.
- A description (also called "gloss") of the synset. It is seldom longer than a couple of sentences. Often the description also contains one or a few example sentences so the user may see how the word is used in a context.

Another type of file, the **index.noun** (and **index.verb** etc.) contains one lexical item per line, paired with the synset ID(s) of the synset(s) in which the lexical item occurs (Figure 2.3). This file is used when a query is issued to WordNet in order for it to find the synsets of the query term. Later in this work, **index.noun** will be used to control whether a word is included in WordNet or not.

```
03013456 06 n 01 crane 0 004 @ 03528075 n 0000 ∼ 03050189 n 0000 ∼
03063444 n 0000 ∼     04300650 n 0000 | lifts and moves heavy objects;
lifting tackle is suspended from a pivoted boom that rotates around a
vertical axis
```

Figure 2.2: Sample data.noun entries of WordNet data file

```
bengali n 3 3 @ #m ; 3 0 09081060 07969251 06544696 benghal_bean n 1 2
@ %p 1 0 11801837 hemp_family n 1 3 @ #m %m 1 0 11648013 hemp_nettle n
1 2 @ #m 1 0 12097519
```

Figure 2.3: Sample index.noun entries of WordNet index file

2.2.2 The Tennis Problem

In WordNet version 1.7.1, the relations between noun synsets are antonymy, hypernymy, hyponymy, holonymy and meronymy. With these, it is possible to relate "tennis" with "badminton" (coordinate term, i.e. both having the same hypernym, both "tennis" and "badminton" is a kind of "court game"), "court game" (hypernym), "doubles" (hyponym, a "double" is a kind of "tennis"), "set point" (meronym, "set point" is a part of "tennis"). There are no antonyms, i.e. opposites to "tennis", but there is an antonymy relation between e.g. "defeat" and "victory". But there is no way to link "tennis" to many things that most humans would agree on really relates to "tennis" in some way ("forehand", "racquet", "serve" etc.). WordNet 2.0 introduced the relation "domain-term" with which these loose relations can be captured. It works in two ways, one can either find the domain(s) in which "serve" occurs, or search for the terms that come with a domain. This relation might prove useful in tasks such as the one undertaken in this work. However, there are yet no semantic similarity tools which can exploit this relation.

2.3 Comparing Concepts

This section presents methods of computing the similarity between entities (eg. concepts or classes) represented in ontologies, or elements (i.e., resources) represented in schemas. These methods, referred to as *semantic similarity methods*, exploit the fact that the entities which are compared may have (in addition to their name) properties (e.g., in the form of attributes) associated with them, taking also into account the level of generality (or specificity) of each entity within the ontology as well as their relationships with other concepts. Notice that, keyword-based similarity measures cannot use this information. Semantic similarity measures methods might be used for performing tasks such as term disambiguation (eg. a user needs the explanation or a definition of a term) as well as retrieving results to user queries, for representation of retrieved resources, and for checking ontologies for consistency or coherency.

2.3.1 Semantic Similarity Measures

We suppose that data in information sources is described by properties and is organized in a taxonomic (subclass-superclass) hierarchy based upon the ontology of the source. When a user issues a query, which is the mechanism for discovering and retrieving answers to the user's inquiry? How are the concepts in the user's query compared with concepts presented in the ontology hierarchies owned by the different information sources? We will give some alternatives to cope with the polysemous nature of natural words, the multiple ways in which the same concept can be described, and the complex terms of source ontologies.

Many measures of semantic similarity have been proposed. In what follows, we present measures of similarity followed by a short discussion of their properties. Semantic similarity measures can be generally partitioned in four categories: those based on how close the two concepts in the taxonomy are, those based on how much information the two concepts share, those based on the properties of the concepts, and those based on combinations of the previous options.

Let $\mathcal{C}$ be the set of concepts in an IS-A taxonomy. We want to measure the similarity of two concepts $c_1, c_2 \in \mathcal{C}$.

Edge-Counting Measures

In the first category we place measures that consider *where* two concepts c_1 and c_2 are in the taxonomy. The following measures are based on a simplified version of *spreading activation theory* [9, 47]. One of the assumptions of the theory of spreading activation is that the hierarchy of concepts is organized along the lines of semantic similarity. Thus, the more similar two concepts are, the more links there are between the concepts and the more closely related they are [39].

Shortest path [39, 7]: The first measure has to do with how close in the taxonomy the two concepts are.

$$sim_{sp} = 2MAX - L \tag{2.1}$$

where MAX is the maximum path length between two concepts in the taxonomy and L is the minimum number of links between concepts c_1 and c_2. This measure is a variant on the *distance* method [39] and is principally designed to work with hierarchies. It is motivated by two observations: the behavior of conceptual distance resembles that of a metric, and the conceptual distance between two nodes is often proportional to the number of edges separating the two nodes in the hierarchy. A measure like this might be implemented in an information retrieval system that is based on indexing documents and queries into terms from a semantic hierarchy, or might be applied to help rank the documents to the query. There are many specific questions about the cognitive realism of shortest path measure, however it is a simple and powerful measure in hierarchical semantic nets.

Weighted links [43]: Extending the above measure, the use of weighted links is proposed to compute the similarity between two concepts. The weight of a link may be affected by: (a) the density of the taxonomy at that point, (b) the depth in the hierarchy, and (c) the strength of connotation[22] between parent and child nodes. Then, computing the distance between two concepts is translated into summing up the weights of the traversed links instead of counting them.

Wu and Palmer [55]: This similarity measure considers the position of concepts c_1 and c_2 in the taxonomy relatively to the position of the most specific common concept c. As there may be multiple parents for each concept, two concepts can share parents by multiple paths. The most specific common concept c is the common parent related with the minimum number of IS-A links with concepts c_1 and c_2.

$$sim_{W\&P}(c_1, c_2) = \frac{2H}{N_1 + N_2 + 2H} \tag{2.2}$$

where N_1 and N_2 is the number of IS-A links from c_1 and c_2 respectively to the most specific common concept c, and H is the number of IS-A links from c to the root of the taxonomy. It scores between 1 (for similar concepts) and 0.

Li et al. [21]: The following similarity measure, which was intuitively and empirically derived, combines the shortest path length between two concepts c_1 and c_2, L, and the depth in the taxonomy of the most specific common concept c, H, in a non-linear function.

$$sim_{Li}(c_1, c_2) = e^{-\alpha L} \cdot \frac{e^{\beta H} - e^{-\beta H}}{e^{\beta H} + e^{-\beta H}} \tag{2.3}$$

where $\alpha \geq 0$ and $\beta > 0$ are parameters scaling the contribution of shortest path length and depth respectively. Based on [21] the optimal parameters are $\alpha = 0.2$ and $\beta = 0.6$. This measure is motivated by the fact that information

[22]The *connotation* of a term is the list of membership conditions for the denotation. The *denotation* of a term is the class of things to which the term correctly applies. For example, the connotation of the general term "square" is "rectangular and equilateral", while its denotation is all squares.

sources are infinite to some extend while humans compare word similarity with a finite interval between completely similar and nothing similar. Intuitively the transformation between an infinite interval to a finite one is non-linear. It is thus obvious that this measure scores between 1 (for similar concepts) and 0.

Leacock and Chodorow [20]: The relatedness measure proposed by Leacock and Chodorow is

$$sim_{lch}(c_1, c_2) = -\log(length/(2 * D)) \tag{2.4}$$

where length is the *length* of the shortest path between the two synsets (using node-counting) and D is the maximum depth of the taxonomy.

The fact that the "Leacock and Chodorow" measure takes into account the depth of the taxonomy in which the synsets are found means that the behavior of the measure is profoundly affected by the presence or absence of a unique root node. If there is a unique root node, then there are only two taxonomies: one for nouns and one for verbs. All nouns, then, will be in the same taxonomy and all verbs will be in the same taxonomy. D for the noun taxonomy will be somewhere around 18, depending upon the version of WordNet, and for verbs, it will be 14. If the root node is not being used, however, then there are nine different noun taxonomies and over 560 different verb taxonomies, each with a different value for D.

If the root node is not being used, then it is possible for synsets to belong to more than one taxonomy. For example, the synset containing turtledove#n#2 belongs to two taxonomies: one rooted at group#n#1 and one rooted at entity#n#1. In such a case, the relatedness is computed by finding the LCS that results in the shortest path between the synsets. The value of D, then, is the maximum depth of the taxonomy in which the LCS is found. If the LCS belongs to more than one taxonomy, then the taxonomy with the greatest maximum depth is selected (i.e., the largest value for D).

The above mentioned measures are based only on taxonomic (IS-A) links between concepts, assuming that links in the taxonomy represent distances. However, the density of terms throughout the taxonomy is generally not constant. Typically, more general terms exist higher in the hierarchy and correspond to a smaller set of nodes than the larger number of more specific terms that populate a much denser space lower in the hierarchy. For example, the distance between *plant* and *animal* is 2 in WordNet (their common parent is *living thing*), and the distance between *zebra* and *horse* is also 2 (their common parent is *equine*). Intuitively *horse* and *zebra* seem more closely related than *plant* and *animal*. Using in our example either the *Wu & Palmer* measure, the measure based on the *Weighted Links* (if the link weights are fixed accordingly) or the *Li et al.* measure, we take into account the fact that the first two terms occupy a much higher place in the hierarchy than the latter two terms and the results will be more realistic. Furthermore, in taxonomies there is wide variability in what is covered by a single taxonomic link. Intuitively, terms adjacent higher in the hierarchy (general terms) should account for higher values of similarity than adjacent

terms lower in the hierarchy (more specific terms). Therefore, edge counting methods weighted by the depth (in the hierarchy) of the terms which are compared, seem more appropriate (eg. "Li et Al [21]). For example, *safety valve* IS-A *valve* seems much narrower than *knitting machine* IS-A *machine*. The *Weighted Links* measure may take into account the strength of links if link weights are computed accordingly. Finally, experimental results presented in [21] have demonstrated that the *Li et al.* measure significantly outperforms previous measures.

In what follows, we present measures involving information content, which seem to perform better that edge-counting measures.

Information Content Measures

The notion of information content of the concept is directly related to the frequency of the term in a given document collection. The frequencies of terms in the taxonomy are estimated using noun frequencies in some large (1,000,000 word) collection of texts [42]. The idea behind semantic similarity information content methods is that the similarity of two concepts is related to information they share in common, as indicated by a highly specific concept that subsumes them both.

Associating probabilities with concepts in the taxonomy, let the taxonomy be augmented by the function $p : \mathcal{C} \rightarrow [0, 1]$, such that for any concept $c \in \mathcal{C}$, $p(c)$ is the probability of encountering an instance of concept c. The concept probability is defined as $p(c) = freq(c)/N$, where N is the total number of terms in the taxonomy, $freq(c) = \sum_{n \in words(c)} n$ and $words(c)$ is the set of terms subsumed by c. This function implies that if c_1 IS-A c_2, then $p(c_1) \leq p(c_2)$, which intuitively means that the more general the concept is, the higher its associated probability.Then, the information content of a concept c can be quantified as the log likelihood, $-\ln p(c)$, which means that as probability increases, informativeness decreases, so the more abstract a concept, the lower its information content.

In order for Information Content measures to perform correct, the limitation of *if* c_1 *IS-A* c_2*, then* $p(c_1) \leq p(c_2)$ must be always satisfied. Although it is implied, the dependency of the above described method on large text corpora, can not guarantee that this will always happen. That's why in this work, we don't use the above described method to measure $p(c)$. We prefer to calculate directly the Information Content (IC) if a concept by using another method proposed by Nuno Seco [49]. In his work, WordNet is used as a statistical resource in order to calculate and innovative exploited in order to produce *ic* values needed for Semantic Similarity calculations.

This method of obtaining IC values rests on the assumption that the taxonomic structure of WordNet is organized in a meaningful and structured way, where concepts with many hyponyms convey less information than concepts that are leaves. As of this, the more hyponyms a concept has the less information it expresses. Likewise, concepts that are leaf nodes are the most informative in the taxonomy. In other words, the Information Content of a WordNet concept is commented as a function of the population of it's hyponyms.

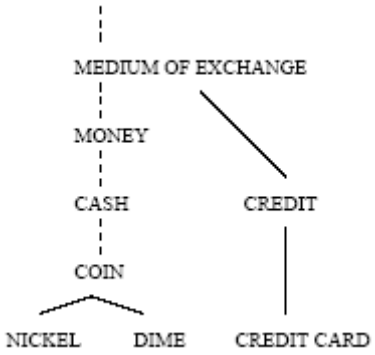

Figure 2.4: A fragment of the WordNet taxonomy

$$ic_{wn}(c) = \frac{\log \frac{hypo(c)+1}{max_{wn}}}{\log \frac{1}{max_{wn}}} \tag{2.5}$$

where the function *hypo* returns the number of hyponyms of a given concept and max_{wn} is a constant that is set to the maximum number of concept that exist in the taxonomy. The denominator which corresponds to the most informative concept and normalizes all IC values in range [0...1]. The above formulation guarantees that the information content decreases monotonically as we traverse from the leaf nodes to the root nodes. Moreover, the information content of the imaginary top node of WordNet would yield an Information Content value of 0.

Given these probabilities (or IC in our case), several measures of semantic similarity have been defined. All these measures use the information content of the shared parents of two terms c_1 and c_2 (see Equations 2.6 and 2.7), where $S(c_1, c_2)$ is the set of concepts that subsume c_1 and c_2. As there may be multiple parents for each concept, two concepts can share parents by multiple paths. We take the minimum $p(c)$ (or maximum ic) when there is more than one shared parents. We call concept c as the *most informative subsumer*.

$$p_{mis}(c_1, c_2) = min_{c \in S(c_1,c_2)}\{p(c)\} \tag{2.6}$$

$$ic_{mis}(c_1, c_2) = max_{c \in S(c_1,c_2)}\{ic_{wn}(c)\} \tag{2.7}$$

For example, in Figure 2.4 *coin* and *cash* are members of $S(nickel, dime)$, but the term that is structurally the minimal upper bound is *coin*, and will also be the most informative subsumer. The information content of the most informative subsumer will be used to quantify the similarity of the two words.

Lord et al. [23]: The first way to compare two terms is by using a measure that simply uses the probability of the most specific shared parent.

$$sim_{Lord}(c_1, c_2) = 1 - p_{mis} \tag{2.8}$$

The probability-based similarity score takes values between 1 (for the very similar concepts) and 0. It is used in order to access the extend to which similarity judgments might be sensitive to frequency per se, rather than information content.

Resnik [42]: The next measure uses the information content of the shared parents.

$$sim_{Resnik}(c_1, c_2) = -\ln p_{mis} \tag{2.9}$$

This measure signifies that the more information two terms share in common, the more similar they are, and the information shared by two terms is indicated by the information content of the term that subsumes them both in the taxonomy. As p_{mis} can vary between 0 and 1, this measure varies between infinity (for very similar terms) to 0. In practice, if N is the number of terms in the taxonomy, the maximum value of p_{mis} is $1/N$ (see Equation 2.6), and the maximum value of the measure is defined by $-\ln(1/N) = \ln(N)$. Thus, this measure provides us with information such as the size of the corpus; a large numerical value indicates a large corpus. Furthermore, the score from comparing a term with itself depends on where in the taxonomy the term is, with less frequently occurring terms having higher scores. This means that when a term is compared with itself, it scores it's Information Content (IC) value rather than "1".

Lin [22]: This measure uses both the amount of information needed to state the *commonality* (ie. sharing of common attributes) between the two terms and the information needed to fully *describe* these terms.

$$sim_{Lin}(c_1, c_2) = \frac{2 \ln p_{mis}(c_1, c_2)}{\ln p(c_1) + \ln p(c_2)} \tag{2.10}$$

As $p_{mis} \geq p(c_1)$ and $p_{mis} \geq p(c_2)$, the values of this measure vary between 1 (for similar concepts) and 0. In this case, a term compared with itself will always score 1, hiding the information revealed by the *Resnik* measure. However, the *Resnik* measure depends solely on the information content of the shared parents, and there are as many discrete scores as there are ontology terms. By using the information content of both the compared terms and the shared parent the number of discrete scores is quadratic in the number of terms appearing in the ontology [23], thus augmenting the probability to have different scores for different pairs of terms. Consequently, this measure offers better similarity ranking of than the *Resnik* measure.

Jiang et al. [16]: Contrary to the above similarity measures, this measure is of *semantic distance*.

$$dist_{Jiang}(c_1, c_2) = -2 \ln p_{mis}(c_1, c_2) - (\ln p(c_1) + \ln p(c_2)) \tag{2.11}$$

Thus, the similarity between two concepts c_1 and c_2, $sim_{Jiang}(c_1, c_2)$, is computed as $1 - dist_{Jiang}(c_1, c_2)$. This measure can give arbitrarily large values, like

the *Resnik* measure, although in practice it has a maximum value of $2\ln(N)$, where N is the size of the corpus. Furthermore, it combines information content from the shared parent and of the compared concepts, as the *Lin* measure does. Thus, this measure seems to combine the properties of the above presented measures (i.e. provides us with both information about the size of the ontology and ranking of different term pairs).

Feature-Based Measure

Up to now, the features of the terms in the ontology are not taken into account. The features of a term contain valuable information concerning knowledge about the term. The following measure considers also the features of terms in order to compute similarity between different concept, while it ignores the position of the terms in the taxonomy and the information content of the term.

Tversky [52]: This measure is based on the *description sets* of the terms (eg. each term may have a set of attributes associated with it or in most cases there are one or more sentenses in plain text explaining the term). We suppose that each term is described by a set of words indicating its properties or features. Then, the *more common* characteristics two terms have (eg. computed as plain text similarity between their associated sets of attributes) and the *less non-common* characteristics they have, the more similar the terms are.

$$sim_{Tversky}(c_1, c_2) = \frac{|C_1 \cap C_2|}{|C_1 \cap C_2| + \kappa|C_1 \setminus C_2| + (\kappa - 1)|C_2 \setminus C_1|} \tag{2.12}$$

where C_1, C_2 correspond to description sets of terms c_1 and c_2 respectively and $\kappa \in [0, 1]$ defines the relative importance of the non-common characteristics. This measure scores between 1 (for similar concepts) and 0, it increases with commonality and decreases with the difference between the two concepts. In reverse to all the above presented measures, it has nothing to do with the taxonomy and the subsumers of the terms, and seems to better exploit the properties of the ontology used.

In the above presented measure, the determination of κ is based on the observation that similarity is not necessarily a symmetric relation: the common, as opposed to the different, features between a subclass and its superclass have a larger contribution to the similarity evaluation than the common features in the inverse direction. Given this assumption, it provides a systematic approach to determine the asymmetry of a similarity evaluation.

Hybrid Methods

The next category of similarity methods,combine ideas from the above presented approaches, considering the path connecting the two terms in the taxonomy, the IS-A links of the terms with their parents in the graph and as well as features of the terms.

Rodriguez et al. [45]: This similarity measure can be used both for single or cross ontology similarities. A similarity function determines similar entity classes by using matching methods over *synonym sets, semantic neighborhoods and distinguishing features* that are further classified into *parts, functions and attributes* (eg. considering the term *college*, a function is *to educate*, its parts may be *roof* and *floor*, and other attributes can be *architectural properties*). The similarity function is a weighted sum of the similarity values for synonys sets, neighborhoods and features.

$$S(a^p, b^q) = \omega_w \cdot S_w(a^p, b^q) + \omega_u \cdot S_u(a^p, b^q) + \omega_n \cdot S_u(a^p, b^q) \tag{2.13}$$

The functions S_w , S_u , and S_n are the similarity between synonym sets, features, and semantic neighborhoods between entity classes **a** of ontology **p** and **b** of ontology **q** and are calculated using Equation 2.14. Weights w_l, w_u, and w_n are the respective weights of the similarity of each specification component.

$$S(a, b) = \frac{|A \cap B|}{|A \cap B| + \alpha|A \setminus B| + (1-\alpha)|B \setminus A|} \tag{2.14}$$

The difference between the above Equation 2.14 and the Tverksy function (Equation 2.12) is in the way κ is computed (in this method α). In this method α is computed according to Equation 2.15. In Tversky 2.12 function, κ defines the relative importance of the non-common characteristics, but here α is computed as a factor of the depth where the two compared concepts are in each taxonomy.

$$\alpha(a^p, b^q) = \begin{cases} \frac{depth(a^p)}{depth(a^p)+depth(b^q)}, & depth(a^p) \leq depth(b^q); \\ 1 - \frac{depth(a^p)}{depth(a^p)+depth(b^q)}, & depth(a^p) > depth(b^q). \end{cases} \tag{2.15}$$

Summary

The key properties of the similarity measures presented in the previous sections are summarized in Table 2.1. We consider whether the similarity measures are affected by the common characteristics of the compared concepts and whether the differences between the concepts cause the measures to decrease. Furthermore, we think the relation of the similarity measures with the taxonomy and the taxonomic relations, i.e. whether the position of the concepts in the taxonomy and the number of IS-A links are considered. It is also presented whether the similarity measures are taking into account the information content of the concepts, whether they are bounded or return infinite values, whether they are symmetric (i.e., $sim(c_1, c_2) = sim(c_2, c_1)$), and whether they give different perspectives.

Property	*Rodriguez*	*Tversky*	*Jiang*	*Lin*	*Resnik*	*Lord*	*Li*	*Wu & Palmer*	*Leacock & Chodorow*	*shortest path*
increase with commonality	yes	yes	yes	yes	yes	yes	yes	yes	no	no
decrease with difference	yes	yes	yes	yes	no	no	yes	yes	yes	yes
information content	no	no	yes	yes	yes	yes	no	no	no	no
position in hierarchy	yes	no	yes	yes	yes	yes	yes	yes	no	no
path length	no	no	no	no	no	no	yes	yes	yes	yes
max value = 1	yes	yes	no	yes	no	yes	yes	yes	no	yes
symmetric	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
different perspectives	yes	yes	yes	yes	no	no	yes	yes	no	no

Table 2.1: Comparison between similarity measures

2.3.2 Ontology Approaches

Ontologies as tools for representing domain knowledge can be used in many different ways. Accordingly, different approaches for comparing concepts within or across ontologies can be defined [54].

Single ontology approach

All information sources are related to one global (unique) ontology providing a common vocabulary for the specification of the entity semantics. A prominent approach is SIMS [3], which includes a hierarchical terminological knowledge base with nodes representing objects, actions and states. Each independent information source is described by relating its objects to the global domain model. Single ontology approach can be applied to cases where all information sources share nearly the same view for a domain (e.g., applications using common sense knowledge may use the Wordnet ontology). Comparing a concept with the ontology is translated into searching for the same or similar concepts within the ontology. How this task can be performed efficiently? How is the notion of "similarity" defined? How close two concepts must be so as to be characterized as "similar"?

Hybrid approach

The semantics of each source is described by its own ontology, but all ontologies are built on one common vocabulary. The shared vocabulary contains basic terms (the primitives) of a domain, upon which the source ontologies are based to built complex[23] terms. As each concept of a source ontology is described by the use of some primitives, the problem of comparing a concept with an ontology can be solved based on methods proposed for the case of the single ontology approach. The drawback however is that existing ontologies cannot be used easily, but have to be redeveloped from scratch as all source ontologies must refer to the shared vocabulary.

In hybrid approaches, the interesting point is how the local ontologies are described, i.e. how the terms of the source ontologies are described by the primitives of the shared ontology. For example, in COIN [11] the local description of the information (called *context*) is an attribute value vector. The terms for the context comes out from the common shared vocabulary. In MECOTA [53] each source information

[23]When the primitives are combined by some operators, as for example the union or intersection operator.

is annotated by a label that indicates the semantics of the information. The label combines the primitives from the shared vocabulary.

Multiple ontology approach

Different information sources (e.g., knowledge about the application) are described by different ontologies. Knowledge within each ontology may be represented without reference to the other information sources or their ontologies. This approach has no common ontology commitment, thus simplifying the adding or modification of information sources. However, the lack of a common vocabulary makes the comparison of different source ontologies a very complicated task.

The multiple ontology approach, although the most general, is the most difficult to handle involving high complexity algorithmic approaches. A straightforward approach is the hard-coded conversion of all data sources into a common ontology which can be stored in a central warehouse. This approach is costly, while it requires substantial efforts from human experts and is not easily extensible to changes of information sources. There are also approaches build around the idea of using wrappers for the automated or semi-automated generation of mappings from the data sources into the global ontology [50]. An attempt to provide intuitive semantics for mappings between concepts in different ontologies is made in [28], where relationships borrowed from linguistics are used to relate terms in various ontologies. In general, the ontology mapping identifies semantically corresponding terms of different source ontologies (e.g., which terms are semantically equal or similar) and has also to consider different views on a domain (e.g., different aggregation of the ontology concepts), becoming thus a non-trivial task.

The problem of comparing concepts between different ontologies could be affronted by borrowing approaches already used in the database community, i.e. schema mapping by discovering semantic correspondences of attributes or instances across heterogeneous sources. The fundamental approach used in this case is "matching", which takes two schemas as input and produces a mapping between elements that semantically correspond to each other [37], or maps concepts to schema elements [44]. Approaches of schema matching can be categorized into *label-based* and *instance-based*, according to the different information on which they rely [40]. Label-based approaches consider only the similarity between schema definitions or attribute labels of two information sources. Instance-based methods rely on the content overlap or statistical properties to determine the similarity of two attributes. Studies concerning the ontology mappings that are based on and extend the schema-matching techniques have been proposed, as for example the development of generic match algorithms [31] and the use of mining techniques [12].

An affinity-based unification method for global mapping construction across ontologies is proposed in [8]. The concept of *affinity* is used to identify terms with semantic relationships in different ontologies. The different ontology terms are firstly analyzed to identify those terms with affinity in different ontologies, which are then identified using a hierarchical clustering procedure [8]. The integration across ontologies is finally achieved by using these clusters.

The advent of peer-to-peer (P2P) systems introduce a different view to the problem by taking a social perspective which heavily relies on self-organization. Mappings between different ontologies are done by special mediator agents which are specialized to translate between different ontologies and different languages in [38]. In this approach agents start from simple one-to-one mappings between classes and continue with mappings between complex expressions. Similarly, data sources introduce their own ontologies and then agents can incrementally come up with a global ontology by exchanging translations between the local ones in [1]. Finally, global semantics are seen in [36] as a matter of continuing negotiation, allowing the creation of global mapping that emerges from peer interactions.

2.4 Information Retrieval (IR)

The term Information Retrieval (IR) identifies all those activities that we can use to choose from a given collection of documents, those documents that are of interest in relation to a specific information need. These activities, allow for reaching a target document or choosing the documents that are probably relevant to the users information need in an automatic way. The problem of translating the users information need is external to the Information Retrieval (IR) system, and the IR system can only answer an information need already formulated as a query. The query is the transformation of an information need in a phrase of a language that the IR system has been programmed to understand and to answer.

In traditional IR, the collection of documents is a set of documents that has been put together, because it is related to a specific context of interest for the users that are going to use it. An IR collection is a set in the mathematical sense, because all the documents of the collection have certain properties or features in common, those features are usually related to a specific subject or thematic area.

The set of all the digital copies of the articles published in the journals and magazines of the ACM (where ACM stands for "Association for Computing Machinery"), of the last 10 years can be an example of collection of documents of interest for the computer scientists of an academic department; These scientists need to have them represented in and managed by an IR system. Another example can be the set of all the published laws of a western country, that can be of interest for the lawyers but also for the citizens of that country. This means that the collection of documents that an IR system manages is clearly identified. When the user receives an answer to a submitted query, this answer is related only to that a-priory identified set.

The IR system uses the complete textual version of each original document in ASCII format to index it and represent its semantic content for matching it with the user's query during retrieval. Retrieval permits choosing some probably relevant documents from the managed collection. This indexing process is often based on full text, since the full text of the document is used in extracting words or terms to represent its semantic content.

2.4.1 Information Retrieval Models

The three classic models in information retrieval are called Boolean, Vector and probabilistic. In the Boolean model, documents and queries are represented as set of index terms, thus this model is *set theoretic*. In Vector Space Model (VSM), documents and queries are presented as vectors in multi-dimensional space, thus we say that the model is *algebraic*. Finally, in the probabilistic model, the framework for modeling document and query representations is based on probability theory, thus we say that the model is *probabilistic*.

2.4.2 The Vector Space Model (VSM)

From the three models described shortly above, Vector Space Model (VSM) is the most popular. This is mostly to its simplicity. Also, the VSM has been shown to perform at least as good as the other two models. The Vector Space Model assigns *non-binary* weights to terms extracted from documents and queries. These term weights are used to compute the *degree of similarity* between each document and the query. The retrieved documents are sorted by similarity with the query. The vector model takes also into consideration documents which match the query terms only partially. The result of this approach is that the answer set is more precise (ie. it better matches the user's information need).

The weight $w_{i,j}$ of term i in document j is associated with a pair (k_i,d_j) is positive and non-binary. The terms of the query are also weighted. Let $w_{i,q}$ be the weight assisted with the pair [k_i,q], where $w_{i,q} \geq 0$. The the query vector $\overrightarrow{q}$, is defined as $\overrightarrow{q}$ = ($w_{1,q}$,$w_{2,q}$,....,$w_{t,q}$) where t is the total number of index terms in the system. The vector for a document d_j is represented by $\overrightarrow{d_j}$ = ($w_{1,j}$,$w_{2,j}$,...., $w_{t,j}$). A document d_j and a user query q are represented as t-dimensional vectors. The Vector Space Model computes the degree of similarity of the document d_j with regard to the query q as based on the inner product between $\overrightarrow{d_j}$ and $\overrightarrow{q}$. This correlation can be quantified by the cosine of the angle between these two vectors.

$$sim(d_j, q) = \frac{\overrightarrow{d_j} \bullet \overrightarrow{q}}{|\overrightarrow{d_j}| \times |\overrightarrow{q}|} = \frac{\sum_t w_{i,j} \times w_{i,q}}{\sqrt{\sum_i w_{i,j}^2} \times \sqrt{\sum_i w_{i,q}^2}} \tag{2.16}$$

Let N be the total number of documents in the system and n_i be the number of documents in which the index term k_i appears. Let $freq_{i,j}$ be the raw frequency of the term k_i in the document d_j. Then, the normalized frequency $f_{i,j}$ of term k_i is given by:

$$f_{i,j} = \frac{freq_{i,j}}{max_l freq_{i,j}} \tag{2.17}$$

where the maximum is computed over all terms which are mentioned in the text of document d_j. If the term k_i does not appear in the document d_j then $f_{i,j} = 0$. Further, let idf_i, inverse document frequency for k_i, be given by

$$idf_i = \log \frac{N}{n_i} \tag{2.18}$$

The best known term-weighting scheme use weights which are given by:

$$w_{i,j} = f_{i,j} \times \log \frac{N}{n_i} \tag{2.19}$$

The main *advantages* of Vector Space Model (VSM) are :

- It's term-weighting scheme improves retrieval performance
- It's partial matching strategy allows retrieval of document that *approximate* the q conditions (don't necessarily match all query terms)
- It's cosine ranking formula sorts the documents by decreasing similarity with the q.

Theoretically, the model has the *disadvantage* that the index terms are assumed to be mutually independent. However, in practice. consideration of term dependencies, might be a disadvantage. Due to the locality of many term dependencies, their indiscriminate application to all the documents in the collection might in fact *hurt* the overall performance.

2.5 Case Study: Information Retrieval on the Web

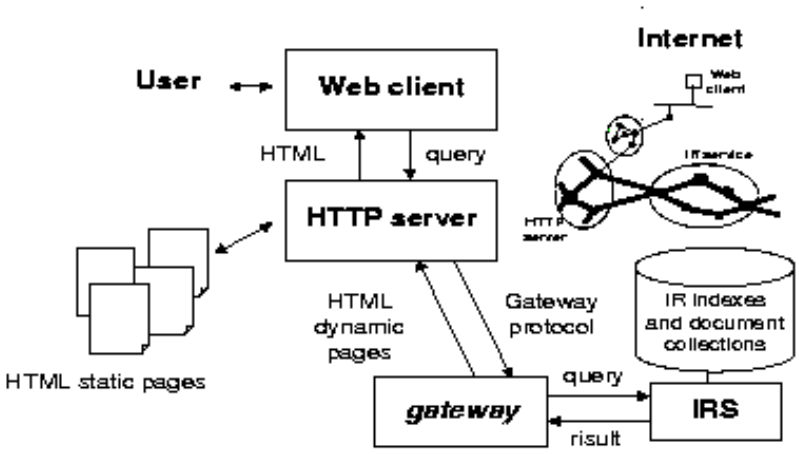

Figure 2.5: Web Information Retrieval System

Information Retrieval on the Web is the process of identifying those documents on the web that are of interest in relation to a specific information need. In the context of IR on the Web, we can consider a Web page as a document. Notice that Web pages can differ in size and in information type and content (ie. they are of

different types of content as opposed to classical IR where all documents belong to a single coherent document collection or document collection of the same subject) since a single page can contain text and many other media, such as graphics, sounds or videos. A Web page also contains hypertext links where a link is an explicit logical association between two Web pages. Then these two Web pages are in some way related (ie. they can be on the same subject). So, the Web page author has decided to relate a page to another to make explicit this relationship. Most of textual parts of Web pages contain links to other Web pages, and each link associates a Web page to a different Web page. The Web pages are related and retrieval or reading can also be performed by navigation or browsing. In fact the Web can be considered as a vast hypertext, and, if the Web is considered in this way, all the studies have been conducted on Hypertext IR (HIR) are of interest.

The size of the web that is public and readable(Lawrence and Giles at mid-1999 made an estimation of about 800 million Web pages, encompassing about 6 terabytes of text data on about 3 million servers), makes the navigation of the web at least hard. So traditional navigation methods, such as direct request of a Web page knowing the correct URL, are not usable.

One main difference between standard IR systems and the Web (SE) is that, in the Web, all queries must be answered without accessing the text (only the indices are available). Otherwise, that would require either storing locally a copy of the Web pages (too expensive) or accessing remove pages through the network at query time (too slow). This difference has an impact on the indexing and searching algorithms, as well as on the query languages made available.

Chapter 3

Problem Definition and Proposed Solution

In this chapter, we present some of the problems involved with semantic similariy and ontologies. We analyze the use of similarity methods in WordNet ontology and building on the method proposed by Rodriguez [45], we adopt a new method for computing semantic similarity between terms that belong in WordNet ontology or between terms from different ontologies. Furthermore, we present the basic characteristics of the software developed as part of this thesis in order to be able to evaluate the performance of semantic similarity methods discussed in previous Chapter 2, both in single or cross ontology experiments.

3.1 Open Problems

During the evolution of the *Phsycolinquistics science* in combination with *Computer Science*, several questions have been brought up. In the computation of the Semantic Similarity between terms, which method better captures the human notion of similariy? Then, in which ontologies can these methods be used and do they perform the same way in all ontologies, or in some ontologies some methods (or families of methods) perform better than others? If so, how can we know which Similarity Methods to use? Now, some methods are based on the so called *Information Content* value of a term. How can we calculate this value for each term? The default way is to calculate it by using large text corpora. But corpus are specific for a domain so, is it safe to use *Information Content* calculated from a specific corpus, in a different domain Ontology? Furthermore, is there a way to compute Semantic Similarity between terms that belong to different ontologies? For example, can we calculate the Semantic Similarity of a term that belongs in WordNet Ontology, ie "car", with a term that belongs in MeSH ontology (ie "anemia")?

3.2 Semantic Similarity Methods in WordNet

WordNet is organized in verb and noun is-a relations. Although WordNet also contains other Part Of Speech (POS) items, such as adverbs and adjectives, those items are not organized in is-a hierarchies. The latest version of WordNet (V2.0) contains 9 separate noun hierarchies that include more that 90.000 different concepts. The is-a relations in WordNet do not cross Part Of Speech (POS) boundaries, so the similarity measures are limited to making judgments between concepts of the same Part

Of Speech (eg. "noun versus noun" but not "noun versus verb"). Similarity between adverbs and adjectives can't be computed because of the non-isa organization.

Except from the is-a hierarchy, WordNet also provides the relations denoted in Figure 2.1 In addition, it provides a small gloss for each concept that describes the concept and may include an example of usage. This information can be used to create more flexible semantic similarity measures, (such as the one we propose in Section 3.3) that can also be applied across different Part Of Speech (POS) and also among different ontologies.

Information Content similarity measures are defined based on the *Information Content* of their Least Common Subsumer (LCS). The fact that the taxonomic structure of WordNet is organized in a meaningful and structured way, where concepts with many hyponyms convey less information from concepts with less hyponyms, makes WordNet a source for computing Information Content (IC). Nuno Seco [49] proposed a method to calculate Information Content (IC) of a concept based on the WordNet taxonomy (rather than on a corpus). The results obtained from the experiments were more than accurate and he proved that WordNet itself is a very good *Information Content*source. Another contribution of his method, is that the IC is normalized in the range [0...1] and also guaranties that the Information Content of a term is less than the information content of its parents ($IC_{c1,c2} < IC_c$)

3.3 Proposed Method

The semantic similarity proposed by Rodriguez [45] relaxes the requirement that both terms are from the same ontology. The proposed similarity function (Equation 3.1) determines similar entity classes, or more simply calculates the similarity between two terms, by using a mathcing process over *synonym sets, semantic neighborhoods and distinguishing features*. Features, are further classified into *parts, functions and attributes*.

$$S(a^p, b^q) = \omega_w \cdot S_w(a^p, b^q) + \omega_u \cdot S_u(a^p, b^q) + \omega_n \cdot S_n(a^p, b^q) \tag{3.1}$$

The functions S_w , S_u , and S_n are the similarity between synonym sets, features, and semantic neighborhoods between entity classes a of ontology p and b of ontology q and are calculated using a Tversky-like (see Equation 2.12) function as can be seen from Equation 3.2. Weights w_l, w_u, and w_n are the respective weights of the similarity of each specification component.

$$S(a, b) = \frac{|A \cap B|}{|A \cap B| + \alpha|A \setminus B| + (1 - \alpha)|B \setminus A|} \tag{3.2}$$

Tversky [52] and the above equation differ in the way α or κ is calculated. In Tversky's method, κ defines the relative importance of the non-common characteristics of the sets, but here is calculated as a function of the position in the hierarchy each term has (Equation 3.3).

$$\alpha(a^p, n^q) = \begin{cases} \frac{depth(a^p)}{depth(a^p)+depth(b^q)}, & depth(a^p) \leq depth(b^q); \\ 1 - \frac{depth(a^p)}{depth(a^p)+depth(b^q)}, & depth(a^p) > depth(b^q). \end{cases} \tag{3.3}$$

Experimental results using different ontologies indicate that the model gives good results when ontologies have complete and detailed representation of entity classes. Also, the combination of word matching and semantic neighborhood matching is adequate for detecting equivalent entity classes and feature matching allows for discriminating among similar but not necessarily equivalent entity classes.

We find their approach very promising, and building upon their method we propose a method that can be used for computing semantic similarity between terms that belong to WordNet ontology as well as comparing terms that belong to different ontologies. We propose modifying the Rodriguez method in the following ways:

- Glosses come with every synonym set (synset) in most ontologies (including WordNet and MeSH) providing us with a description of the term meaning and it is found in most ontologies including WordNet and MeSH. It consists of a descriptive part and an example of use case. This kind of information should also be taken into account for computing the similarity between terms. We propose to replace Feature Mathcing (S_u) with Gloss Similarity that is computed by the following formula

$$S_{gloss}(a^p, b^q) = \frac{|A \cap B|}{|A \cup B|} \tag{3.4}$$

  where A and B are the glosses of concepts a^p, b^q and contain terms extracted from A,B.

- In the original method, *semantic neighborhood matching* is computed in the following way: the similarity function puts all the terms in radius r from the first term into a single set and compares them to another set that contains the all the terms in the same radius of the second term. The similarity between the two sets is computed again using Equation 2.14. We propose the following formula in order to calculate *semantic neighborhood matching*

$$S_{nm}(a^p, b^q) = \max_i \frac{|A_i \cap B_i|}{|A_i \cup B_i|} \tag{3.5}$$

  where i denotes relations (ie. hyponyms, part-meronyms, holonyms) that both a^p and b^q have. A_i, B_i are sets containing all terms derived from the i-th relation of concepts a^p, b^q. In other words, we propose taking the maximum similarity between the two terms in one part of the neighborhood and not in all the area.

- No information regarding the structure of the ontologies should be taken into account when comparing concepts from different ontologies. We propose that the factor α of Equation 2.14 should be always 1 when comparing concepts from different ontologies.

- Combining the above formulas in a linear way we propose 3.6 formula in order to compute Semantic Similarity

$$S_{proposed}(a,b) = \omega_w S_w(a^p, b^q) + \omega_{gloss} S_{gloss}(a,b) + \omega_{nm} S_{nm}(a^p, b^q) \tag{3.6}$$

This is rather an adoptation of Rodriguez [45] model for use with WordNet ontology rather than a new method. We believe that our modified version of Rodriguez method will achieve better perfrormance both when the compared concepts belong to WordNet or to different ontologies.

3.4 Evaluation of Semantic Similarity Methods

In accordance with previous research, we evaluated the results obtained by applying the semantic similarity methods discussed in the previous section, by correlating their similarity scores with the scores obtained by human judgments. These results are provided by Miller and Charles [30]: 38 undergraduate students were given 30 pairs of nouns and were asked to rate similarity of meaning for each pair on a scale from 0 (not similar) to 4 (perfect synonymy). The average rating of each pair represents a good estimate of how similar the two words are. We compute correlation using Pearsons correlation function (Equation 3.7) found in OpenOffice suite and in many other spreadsheet programs. Suppose we have two variables X and Y, with means $\overline{X}$ and $\overline{Y}$ respectively and standard deviations S_X and S_Y respectively. The correlation is computed as

$$r = \frac{\sum_{i-1}^{n} (X_i - \overline{X})(Y_i - \overline{Y})}{(n-1) S_X S_Y} \tag{3.7}$$

Furthermore, a cross-ontology similarity experiment was desinged in order to evaluate the performance of three methods that can be used in such experiments: Rodriguez [45], Tversky [52] and our proposed method. We decided to experiment with WordNet and MeSH ontologies. MeSH is a widelly used medical ontology. Fourty pairs of medical terms, also contained in WordNet ontology, were carefully selected. These pairs were given to 20 doctors in order to calculate the degree of similarity between them. By this way we obtained the human judgement needed in order to evaluate the performance of each method. We decided to perform the experiment in the following way: Each term of a pair belongs to a different ontology (ie. one term is from MeSH and the other from WordNet). Then, we apply the three similarity methods in each pair and retrieve the results. The performance of each method is evaluated again as the correlation of these results with the human judgement.

We developed our own software system that implements all similarity methods described in section 2.3.1 in a way that can be used in conjunction with any ontology that satisfies the WordNet is-a hierarchy. A detailed description of the System Architecture can be found in section 3.5 (More technical details are discussed in Appendix A). Also, an on-line version of the system is available at
http://ultralix.softnet.tuc.gr:8080/similarity.

Options like similarity method selection and ontology selection are available. Furthermore, we can compare one sense of a term with one sense of another term or all the senses of both terms.

3.4.1 Experimental Results

In this section we present the results we obtained by running all semantic similarity methods in our software platform. The results are divided in four tables. Table 3.1 presents results from *edge-counting* similarity methods and their correlation with the human judgment results by Miller and Charles [30]. Table 3.2 present results obtained by *Information Content* similarity methods and their correlation with the human judgment. Table 3.3 present the results we obtained from *Hybrid* methods, including our proposed method and *feature based method* proposed by *Tversky* [52]. Results from the cross ontology experiment are presented in Table 3.4.

The reason why we divide the results in those four categories is because it's not wise to mix methods that are based different models and techniques, for example methods that are completely based on the hierarchy with methods that are completely based on the IC of the concept and ignore the hierarchy, or with methods that are based on feature matching and are based on our proposed techniques. Each category contains methods that are products of long-time research in different divisions of the Linguistics science, so the only secure comments we can make from the observation of the results, is which method performs better from another in each category by just looking at the correlation result. So, the best method, is the one that gives the highest correlation score.
The following is a list of methods tested. Each method is represented by a letter symbol.

- A — Human Judgment
- B — Shortest Path method (Edge Count)
- C — Wu and Palmer method (Edge Count)
- D — Li et al method (Edge Count)
- E — Leacock and Chodorow method (Edge Count)
- F — Shortest Path with Weighted Links method (Edge Count)
- G — Resnik proposed method (IC)
- H — Lin method (IC)
- I — Lord et al method (IC)
- J — Jiang et al method (IC)
- K — Rodriguez proposed method (Hybrid)

- L — Tversky function as similarity method (Feature Based)
- M — Our proposed method based on Rodriguez (Hybrid)

concept one	concept two		A	B	C	D	E	F
car	automobile		3.92	1	1	1	3.58	1
gem	jewel		3.84	1	1	1	3.58	1
journey	voyage		3.84	0.97	0.92	0.82	2.89	0.98
boy	lad		3.76	0.97	0.93	0.82	2.89	0.98
coast	shore		3.7	0.97	0.91	0.81	2.89	0.98
asylum	madhouse		3.61	0.97	0.94	0.82	2.89	0.98
magician	wizard		3.5	1	1	1	3.58	1
midday	noon		3.42	1	1	1	3.58	1
furnace	stove		3.11	0.81	0.46	0.23	1.5	0.8
food	fruit		3.08	0.81	0.22	0.13	1.5	0.67
bird	cock		3.05	0.97	0.94	0.82	2.89	0.98
bird	crane		2.97	0.92	0.84	0.55	2.2	0.96
tool	implement		2.95	0.97	0.91	0.81	2.89	0.98
brother	monk		2.82	0.97	0.94	0.82	2.89	0.98
crane	implement		1.68	0.89	0.67	0.44	1.97	0.9
lad	brother		1.66	0.89	0.71	0.45	1.97	0.91
journey	car		1.16	0	0	0	0.88	0.25
monk	oracle		1.1	0.81	0.59	0.25	1.5	0.88
food	rooster		0.89	0.64	0.13	0.04	0.94	0.57
coast	hill		0.87	0.89	0.67	0.44	1.97	0.89
forest	graveyard		0.84	0.75	0.18	0.09	1.28	0.61
monk	slave		0.55	0.89	0.71	0.45	1.97	0.91
coast	forest		0.42	0.83	0.4	0.25	1.64	0.78
lad	wizard		0.42	0.89	0.71	0.45	1.97	0.91
chord	smile		0.13	0.72	0.44	0.13	1.19	0.79
glass	magician		0.11	0.75	0.31	0.14	1.28	0.7
noon	string		0.08	0	0	0	1.1	0.34
rooster	voyage		0.08	0	0	0	0.59	0.17
Correlation			1	0.59	0.74	**0.82**	**0.82**	0.63

Table 3.1: Performance of Edge Counting Methods

concept one	concept two		A	G	H	I	J
car	automobile		3.92	0.68	1	0.49	1
gem	jewel		3.84	1	1	0.63	1
journey	voyage		3.84	0.66	0.84	0.48	0.88
boy	lad		3.76	0.76	0.86	0.53	0.88
coast	shore		3.7	0.78	0.98	0.54	0.99
asylum	madhouse		3.61	0.94	0.97	0.61	0.97
magician	wizard		3.5	0.8	1	0.55	1
midday	noon		3.42	1	1	0.63	1
furnace	stove		3.11	0.18	0.23	0.17	0.39
food	fruit		3.08	0.05	0.13	0.05	0.63
bird	cock		3.05	0.4	0.6	0.33	0.73
bird	crane		2.97	0.4	0.6	0.33	0.73
tool	implement		2.95	0.42	0.93	0.34	0.97
brother	monk		2.82	0.83	0.91	0.56	0.91
crane	implement		1.68	0.24	0.37	0.21	0.59
lad	brother		1.66	0.18	0.2	0.17	0.28
journey	car		1.16	0	0	0	0.33
monk	oracle		1.1	0.18	0.22	0.17	0.34
food	rooster		0.89	0.05	0.08	0.05	0.4
coast	hill		0.87	0.5	0.63	0.39	0.71
forest	graveyard		0.84	0.05	0.06	0.05	0.19
monk	slave		0.55	0.18	0.23	0.17	0.39
coast	forest		0.42	0.08	0.1	0.08	0.29
lad	wizard		0.42	0.18	0.21	0.17	0.32
chord	smile		0.13	0.25	0.28	0.22	0.35
glass	magician		0.11	0.08	0.21	0.08	0.68
noon	string		0.08	0	0	0	0.18
rooster	voyage		0.08	0	0	0	0.08
Correlation			1	0.79	**0.82**	0.79	**0.83**

Table 3.2: Performance of IC based Methods

concept one	concept two		A	K	L	M
car	automobile		3.92	1	1	1
gem	jewel		3.84	1	1	1
journey	voyage		3.84	0.69	0.86	0.52
boy	lad		3.76	0.6	0.88	0.44
coast	shore		3.7	0.42	0.83	0.66
asylum	madhouse		3.61	0.44	0.89	0.44
magician	wizard		3.5	1	1	0.1
midday	noon		3.42	1	1	1
furnace	stove		3.11	0.21	0.43	0.47
food	fruit		3.08	0.08	0.17	0.09
bird	cock		3.05	0.44	0.89	0.45
bird	crane		2.97	0.36	0.73	0.37
tool	implement		2.95	0.42	0.83	0.48
brother	monk		2.82	0.44	0.89	0.44
crane	implement		1.68	0.29	0.57	0.29
lad	brother		1.66	0.54	0.71	0.45
journey	car		1.16	0	0	0.03
monk	oracle		1.1	0.28	0.56	0.28
food	rooster		0.89	0.04	0.08	0.05
coast	hill		0.87	0.33	0.67	0.34
forest	graveyard		0.84	0.22	0.14	0.07
monk	slave		0.55	0.31	0.63	0.32
coast	forest		0.42	0.17	0.33	0.18
lad	wizard		0.42	0.36	0.71	0.36
chord	smile		0.13	0.2	0.4	0.2
glass	magician		0.11	0.14	0.29	0.14
noon	string		0.08	0	0	0
rooster	voyage		0.08	0	0	0
Correlation			1	0.71	0.73	**0.75**

Table 3.3: Performance of Hybrid Methods

WordNet Term	MeSH term		Human	K	M
Anemia	Appendicitis		0.0312	0	0
Dementia	Atopic Dermatitis		0.0625	0	0
Bacterial Pneumonia	Malaria		0.1562	0.028	0.113
Osteoporosis	Patent Ductus Arteriosus		0.1562	0.0306	0.122
Immunodeficiency Syndrome	Congenital Heart Defects		0.0625	0.021	0.084
Otitis Media	Infantile Colic		0.1562	0	0
Meningitis	Tricuspid Atresia		0.0312	0.006	0.025
Sinusitis	Mental Retardation		0.0312	0	0
Hyperlipidemia	Hyperkalemia		0.1562	0.045	0.182
Hypothyroidism	Hyperthyroidism		0.4062	0.096	0.387
Sarcoidosis	Tuberculosis		0.4062	0	0
Asthma	Pneumonia		0.375	0.026	0.07
Diabetic Nephropathy	Diabetes Mellitus		0.5	0.065	0.205
Lactose Intolerance	Irritable Bowel Syndrome		0.4687	0.01	0.047
Urinary Tract Infection	Pyelonephritis		0.6562	0.0075	0.03
Neonatal Jaundice	Sepsis		0.1875	0	0
Sickle Cell Anemia	Iron Deficiency Anemia		0.4375	0.044	0.14
Psychology	Cognitive Science		0.5937	0.069	0.25
Adenovirus	Rotavirus		0.4375	0.0558	0.16
Migraine	Headache		0.7187	0.011	0.042
Myocardial Ischemia	Myocardial Infarction		0.75	0.1188	0.47
Hepatitis B	Hepatitis C		0.5625	0.118	0.42
Carcinoma	Neoplasm		0.75	0.072	0.17
Failure to Thrive	Malnutrition		0.625	0.0108	0.043
Breast Feeding	Lactation		0.8437	0	0
Antibiotics	Antibacterial Agents		0.9375	0.022	1
Pain	Ache		0.875	0.063	1
Malnutrition	Nutritional Deficiency		0.875	0.13	1
Chicken Pox	Varicella		0.9687	0.25	1
Down Syndrome	Trisomy 21		0.875	0.18	1
Correlation			1	0.5738	**0.6972**

Table 3.4: Cross Ontology experiment results

3.4.2 Discussion of the Results

Table 3.5 demonstrates that all methods have a correlation score between 0.59 and 0.83. *Edge Counting Methods* have a minimum score of 0.59 (Shortest Path Method) and a maximum score of 0.82 corresponding to the method by Li [21]. *Information Content Methods* score a minimum 0.79 and a maximum of 0.83 corresponding to method proposed by Jiang [16]. *Hybrid* methods have a minimum correlation of 0.71 and a maximum of 0.75 which is given by our proposed method 3.3

Information Content Methods seem to perform very well and specially Jiang's [16] method is very close to to what Resnik [42] proposed as a computational upper bound [1]. Reproducing the experiment performed by Jiang and Conrath where they removed the pair of *furnace - stove* from their evaluation claiming that Most Specific Common Abstraction (MCSA) for the pair is not reasonable, we obtain a correlation value of 0.87 for this method.

From the *Edge Counting Methods*, the method by Li [21] performs quite well and we believe it is very promising. It has a correlation score of 0.82, which is close enough to Resnik's upper bound. We also believe that the performance of this method can be explained from the fact that it takes into account some very important facts : 1) the depth of each term in the taxonomy (ie. "how deep in the ontology the terms are?"). The deeper the terms are, the more specific they are. 2) the depth of the Most Common Parent in the taxonomy and 3) the path length between the terms which are compared.

Regarding *Hybrid Methods*, our proposed method 3.3, performed better from other methods in this family in both experiments. Regarding the single ontology experiment, the correlation of (0.75) is not that high compared to the results we obtained by *Jiang et al [16]* and *Li et al [21]* methods, but performs better than the original Rodriguez method and has the best performance in this family. This means that the modifications we propose have a positive effect on it's performance. As for the cross ontology experiment, again we have an increased (12%) performance compared with the original method and an overall correlation of **r=0.6972**. We find this prommising for several reasons. This method is suitable for cross ontology similarity matching, making no apriori assumption on the structure and the properties of the ontologies where the terms belong to. Also, the correlation of 0.75 in the single ontology experiment in not bad compared with all the other methods. By further investigating the subject we believe that it's performance may increase even more.

[1] a correlation of r = 0.83 with a benchmark set of human similarity judgments, against an upper bound of r = 0.90 for human subjects performing the same task

	Min Correlation	Max Correlation	Variation
Edge Counting Methods	0.59	0.82	0.23
Information Content Methods	0.79	0.83	0.04
Feature - Hybrid Methods	0.71	0.75	0.04

Table 3.5: Summary of Performance for each Similarity Method family

3.5 System Architecture

In the following we discuss how the software works in general, and why it works that way. In Appendix A the interest reader can find usage samples of the software plus some technical details. Our system has a key advantage that has to be mentioned. Because of it's nature (accepts XML files as input, simple Concept-IC DB [2]), our system can be used as-is from anyone who wants to calculate Semantic Similarity between two terms based on ANY ontology that can produce XML files that can be validated by the XML-Schema that describes the WordNet ontology, without information loss. For example, if someone wants to compare two terms from the "MeSH" ontology, our system can produce accurate results if the XML files describing the MeSH-terms are valid. Furthermore, *it's plugable architecture*, allowing for expansion with minimum effort. One can write a new Similarity Measure in Java and just plug the Class produced in the system. The use of the new Similarity Measure is now done by just selecting it. Figure 3.1, shows the layout of the system we developed.

The system consists of four parts (see Figure 3.1

Ontology: First we have the the ontology (in our case *WordNet*) itself. We always need to have the Ontology available in order to be able to extract information for each term we want to compare. For example, the web-service that creates the XML files, wouldn't work without the ontology, neither we could calculate the Information Content (IC) of each concept.

XML repository: The second part is the repository that holds the .xml files for all terms. Each file holds all the information extracted from the ontology associated with a term,(ie. car.xml fully describes the term "car", as it is found in WordNet ontology). Notice that each .xml file contains information for all the senses of a term. A detailed description of what the .xml files contains, can be found in Appendix A in the WordNet.dtd schema description. Those XML files were created using the *WordNet XML web-service* [3] created by Bernard Bou [4].

Information Content (IC) database: The third part is the database (postgresql database [5]) holds the *Information Content (IC)* of each term as this is calculated by Equation 2.5. For each sense of a term, a different IC value has been

[2] A database that holds the Information Content (IC) for each concept, as this is computed by [49]

[3] http://wnws.sourceforge.net

[4] bbou@ac-toulouse.fr

[5] http://www.postgresql.org

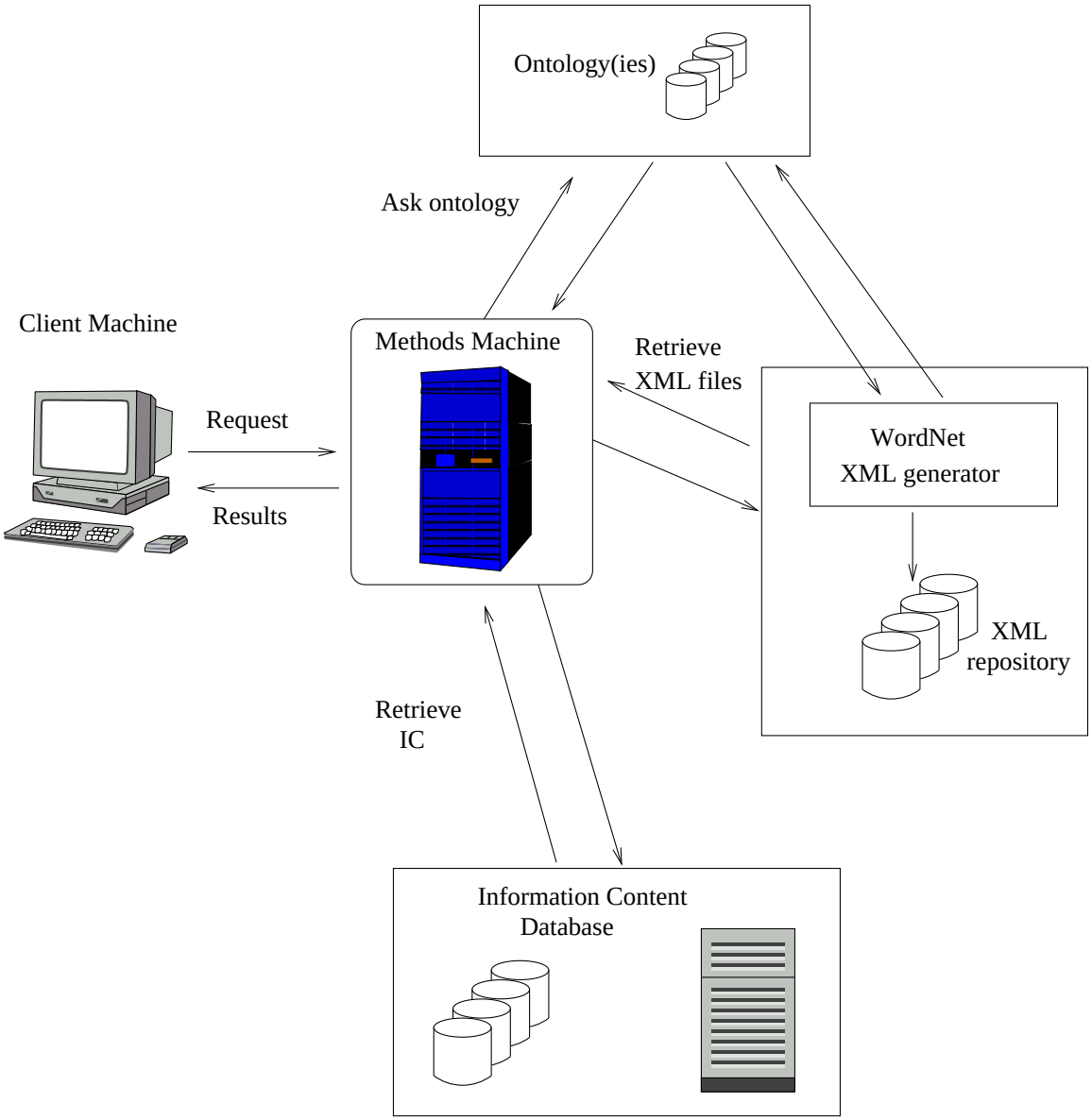

Figure 3.1: Architecture of the Semantic Similarity System implemented in this work

calculated and stored in the database. The database is accessed every time we need to find the IC of a term. The database is relational and all information is stored in one table. The table has two columns: the first one contains the Concept as as string and the second hold the IC of the Concept.

Base System: The machine that calculates the similarity between two given terms. All similarity function are implemented here. Given two XML files and a method, the machine calculates the similarity between them using the given method and returns the result to the user, either by command line or a web interface. User has the option to compare a specific sense of a term with a specific sense of another term or all the senses of the first with all the senses of the other and see how similar they are. All the available options can be found in Appendix A.

Given a request, our system executes the algorithm of the flowchart 3.2
The user of our system has the following options

- *Sense selection* The user has the option to select which sense of the term to compare (one specific or all senses)
- *Similarity method selection* A list of 10 similarity methods is available from all categories with option to add new methods as easy as inserting a java class file in the system
- *Ontology selection* Our system is Ontology independent. The methods can be used within any ontology that conforms to a specific schema or that can be mapped to this schema
- *Cross ontology similarity* by selecting two terms from different ontologies and compare them

A complete API was developed in order to be used from anyone who wants to make use of the system functionality.
A complete working interface of our system, having all the above characteristics, can be found at the web at **h**ttp://ultralix.softnet.tuc.gr:8080/thesis. Additive to the above characteristics, the web user can find information about all similarity methods and the ontologies used.

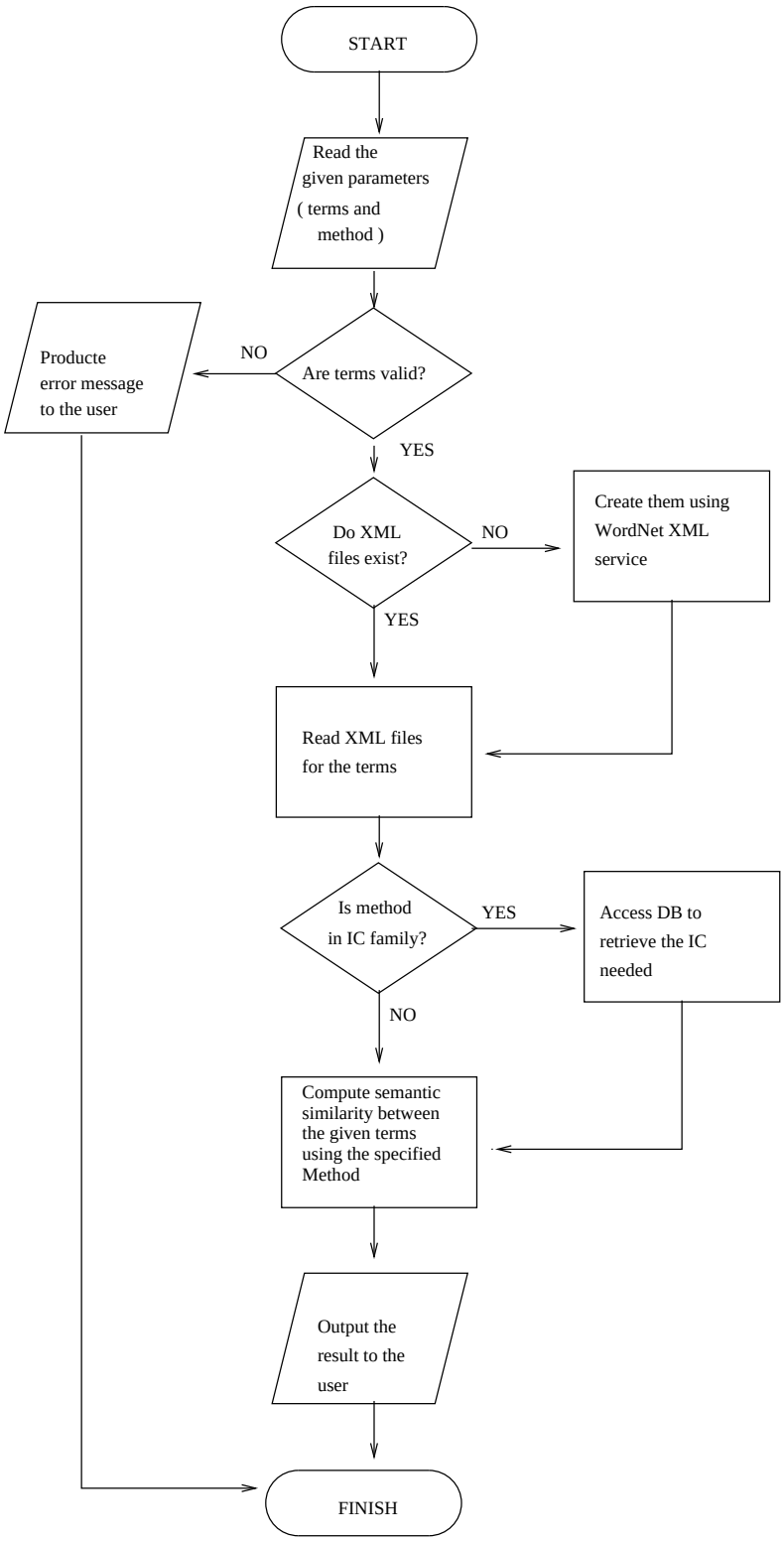

Figure 3.2: Computing Semantic Similarity

Chapter 4

Case Study: Application of Semantic Similarity to Information Retrieval on the Web

In this chapter, we propose a new Information Retrieval Model based on the Semantic Similarity aspect and we present it's application to a Web Information Retrieval System (search engine) that is used to retrieve both Images and Documents. This is another contribution of this thesis. Information Retrieval on the Web is a difficult problem that affects all the users of the Net. Everyone who uses the Web, uses a *Search Engine (SE)*, such as Google [1], to find either pages, pictures, papers, videos etc. We find our model promising and we believe that a further expansion of such IR systems with the use of *Ontologies* and *Semantic Similarity* will result in more accurate and reliable results. Next in this chapter we present our model, the architecture of the system we tested our model, the experimental results we obtained and an evaluation of the results is finally done.

4.1 Proposed method

When the user enters a query the Search Engine (SE) searches all indexed documents to find documents that contain the entered term and returns them to the user ordered by the relatedness of the document with the query as this is calculated by the Vector Space Model (in most cases). However, documents that contain related information but their context in described by other terms, are not returned to the user. For example, let's say that some documents use the term "automobile" instead of "car". Although the two terms are synonyms, if the user's query contains just the term "car", documents that use "automobile" instead, won't be returned. Also documents that contain the entered term can score very low although they might be more related to what the user is searching for.

Traditionally, the similarity between two documents d_i and d_j (e.g., a query and a document) is computed according to (VSM) as the cosine of the inner product between their term vectors

$$Sim(d_i, d_j) = \frac{\sum_t w_{it} w_{jt}}{\sqrt{\sum_t w_{it}^2}\sqrt{\sum_s w_{js}^2}} \tag{4.1}$$

where w_{it} and w_{js} are the weights in the two vector representations. Given a query,

[1] http://www.google.com

all documents are ranked according to their similarity with the query. This model is also known as *bug of words* model and is the state of the art model for document retrieval.

The lack of common terms in two documents does not necessarily mean the documents are irrelevant. Similarly, relevant documents may not contain the same terms. Semantically similar terms or concepts may be expressed in different ways in the documents and the queries, and direct comparison by VSM is not effective. For example, VSM will not recognize synonyms or semantically similar terms (e.g., car - automobile). We propose discovering semantically similar terms using WordNet.

WordNet is a vocabulary and a lexical ontology that attempts to model the lexical knowledge of a native speaker of English into a taxonomic hierarchy. Entries (i.e., terms or concepts) are organized into *synsets* (i.e., lists of synonym terms or concepts), which in turn are organized into senses (i.e., different meanings of the same term or concept). There are also different categorizations corresponding to nouns, verbs, adverbs etc. Each entry is related to entries higher or lower in the hierarchy by different types of relationships. The most common relationships are *Hyponym/Hypernym* (i.e., isA relationships), and *Meronym/Holonym* (i.e., Part-Of relationships). In the following, we only use the nouns and the Hyponym/Hypernym relationships from WordNet to enhance vector representations.

The proposed model relies on discovering semantic similarities between terms algorithmically [42] [45] [22] by relating terms in WordNet. A recent contribution [21] has been shown particularly effective. The proposed approach works in two steps:

Reweighting: The reweighting of the Query terms is done in the following way:

- Re-weight the terms in the vector: the weight of a term is adjusted due to its relationships with other semantically similar terms within the same vector as follows

$$w_i = w_i + \sum_{\substack{j \neq i \\ sim(i,j) \geq t}} w_j sim(i,j) \tag{4.2}$$

  This step help us to understand if the user is trying to emphasize in one specific area of interest by finding semantically similar terms in the vector. The terms in the Query Vector are reweighted as a first step. Semantically similar terms in the query are assigned a greater weight from the initial one and those who don't match an amount of similarity with the other terms, hold their initial weight. In the case the user enters some terms that are semantically similar enough, and some other terms that are not even semantically related with the others (ie. "railway train metro cat spoon"), then with this step the system assigns greater weight to the terms "railway" "train" "metro", that are semantically similar enough. The weights of the terms "cat" "spoon" that are not related will remain unchanged.

Expansion: The Query is expanded using an ontology, (in our case WordNet). Terms that are close in the taxonomy with the Query terms, are useful and

can be used to retrieve information related to the Query term. This is also illustrated by Figure 4.1

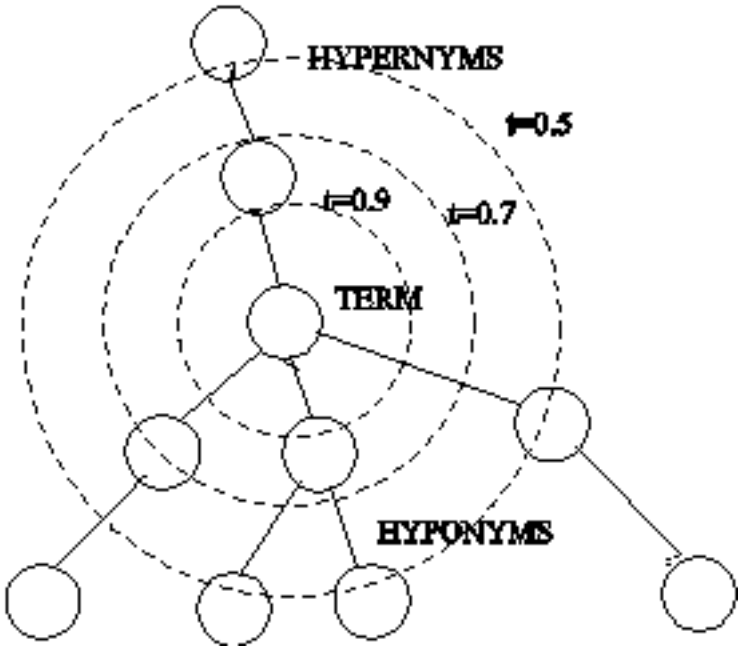

Figure 4.1: Example of a term neighborhood in WordNet taxonomy

- Augment the original vector by synonyms (the most common sense only) then by hyponyms and hypernyms. Each synonym of a term, is assigned the weight of the initial term found in the vector. For each term r in the vector augment the vector by its hypernyms and hyponyms s with $sim_(r, s) \geq t$ (e.g., t=0.8). By this expansion, each new term s in the vector (which now included new terms) is assigned a weight as follows

$$w_s = \begin{cases} \frac{1}{n} sim(r, s), & \text{s is a new term} \\ w_s + \frac{1}{n} sim(r, s) & \text{s had weight } w_s \text{ before expansion} \end{cases} \tag{4.3}$$

  where r is the original term (which is expanded). Notice that one or more of the hypernyms or hyponyms of term s might have existed (and had a weight) before expansion. The above formula suggests taking the weights prior to expansion into account. It also suggests that the contribution of the original term r is normalized by the number n of its hyponyms. For hypernyms, n = 1.

We propose the expansion to be done with terms that are Semantically Similar to the Query Term in a radius t. What we mean by this is that each term is expanded with terms that are Semantically Similar to it, with Similarity values $\geq t$. As the value of t decreases, the radius increases. If $t = 1$ then the

expansion in only done with synonyms. For a given t, the numbers of terms which are added depend also on the term: Very specific terms are expected to be very similar with their direct hypernyms and hyponyms. Very general terms (higher in the hierarchy) are expected to be less similar with their hypernyms and hyponyms. This means that in the expanded Query may contain Synonyms, Hyponyms and Hypernyms of the entered terms.

Also notice that if a big radius is chosen then we face another problem : the retrieved documents might not be focused on the initially specified topic. For example, if the user searches for "car" and we expand the query with all the concepts that are included in a radius of 0.5, then the expanded query will contain terms like "bike, motorbike, vehicle, ... " which although they are semantically related to "car", they refer on another topic. This problem is referred to as "topic drift". Therefore special attention must be given in threshold, with respect to the similarity method for expanding the terms in query and document.

The expansion is done only for the *Most Frequent Term Sense* as this is defined from the Ontology and not for all the senses of each term. This is not exactly intuitively correct as we don't know what sense of the term the user means, but we assume for the purpose of this thesis that the most frequent term is desired. Although in the following steps we can say that we perform a form of "sense disambiguation" [48], we can not state that this step is oriented to do exactly this, so further investigation of this matter should be done. For the purpose of this work we believe that this initial step in enough.

Similarity Matrix: Expanding and re-weighting is fast for queries (queries are short in most cases specifying only a few terms) but not for document vectors with many terms. An approximation would be not to expand and re-weight the document vector. In this case, their similarity function must take into account relationships between semantically similar terms (something that the cosine similarity method cannot do). Then the similarity between an expanded and re-weighted query vector q and a non expanded and non re-weighted vector document vector d is computed as

$$Sim(q,d) = \frac{\sum_k \sum_l w_k w_l sim(k,l)}{\sum_k \sum_l w_k w_l} \tag{4.4}$$

where k and l are terms of the document and the query respectively. Also, w_k is the document k-term weight as it is calculated by the formula of Vector Space Model described in section 2.4.2, using tf and idf. Weight w_l is the query l-term weight as calculated from the above steps. Notice here that the similarity is normalized in the range [0,1] because of the nature of the similarity function (results in the range [0,1]).

Below we present an example of Query Vector and how this is expanded and reweighted. In this example, we choose a threshold $t = 0.9$ of similarity between the

terms to be expanded. This means that the Query is expanded only with terms that have similarity $\geq t$ with the terms of the original Query Vector. Notice that the synonyms, are assigned the same weight as the original term of the Query Vector. The reason we do this is because the *synonymy* relation denotes that either term in the *synonym set* is the same and has the same meaning with the other. In the case we have chosen a bigger threshold and new terms were to be entered in the Query Vector, then their weights would be calculated according to formula 4.3 but the synonym terms would again be assigned the weight of original term of the Query.

Query		Weights
cat		(w_{cat})
cougar		(w_{cougar})
lion		(w_{lion})

Table 4.1: Example of Original Query Vector

Re-Weighted Query		Weights
cat		$(w_{cat} + 1.12)$
cougar		$(w_{cougar} + 0.91\ )$
lion		$(w_{lion} + 0.68)$

Table 4.2: The Re-weighted Query Vector

Re-Weighted Query		Weights
cat		$(w_{cat} + 1.12)$
true cat		$(w_{cat} + 1.12)$
cougar		$(w_{cougar} + 0.91\ )$
puma		$(w_{cougar} + 0.91\ )$
catamount		$(w_{cougar} + 0.91\ )$
mountain lion		$(w_{cougar} + 0.91\ )$
painter		$(w_{cougar} + 0.91\ )$
panther		$(w_{cougar} + 0.91\ )$
Felis concolor		$(w_{cougar} + 0.91\ )$
lion		$(w_{lion} + 0.68)$
king of beasts		$(w_{lion} + 0.68)$
Panthera leo		$(w_{lion} + 0.68)$

Table 4.3: Re-Weighted and Expanded Query Vector

Notice that the expanded query contains only synonyms of the original terms. The weights assigned to the synonym terms is the same as the weight of the original

term. This is because of the large value of threshold we have chosen for the expansion. If we have chosen a threshold of t=0.5 then the vector would be expanded also with hyponyms and hypernyms.

		puma	cat	lion
house		0.0339	0.0415	0.0339
cat		0.5488	1	0.5488
sleeping		0	0	0
near		0	0	0
fireplace		0.0415	0.0569	0.0415
while		0	0	0
heavy		0.0589	0.0730	0.0589
rain		0	0	0
dad		0.0268	0.0400	0.0382
reading		0	0	0
newspaper		0.0186	0.0228	0.0227
smoking		0	0	0
pipe		0.0546	0.0467	0.0678

Table 4.4: Illustration of Similarity Matrix method resulting matrix

In table 4.4, we present an example of the resulting Similarity Matrix while comparing the query vector [2] with the caption of an image, as calculated by formula 4.4. We can see that there are plenty of zeros in the result. This is mainly because of the different POS of each term (we can't calculate semantic similarity between nouns and verbs), the different main sub-tree that each term belongs to, and the assumption that we calculate similarities only for the most common sense of each term. This means that possibly two terms that give us a similarity of "0" in their most common sense, will give us a similarity ≥ 0 while comparing another sense.

4.2 System Architecture

A complete prototype Web image retrieval system, developed as part of the thesis work done by *Epimenidis Voutsakis* [3] was used as the base system in order to be able to complete our experiments. The original system consists of several modules, the most important of them being the *crawler module* for assembling locally a collection of Web pages, the *collection analysis* module for analyzing Web contents and creating appropriate descriptions for Web pages and images, the *storage* module implementing storage structures and indices, and finally, the *query processing* module implementing the search methods for all types of image queries. The modified system, consists of the same modules as the original one, with the addition of some options in the search

[2] We don't include the expanded terms, neither the term weights because this in only an example trying to illustrate how the formula works

[3] http://www.softnet.tuc.gr/p̃imenas

methods menu, plus the "WordNet" ontology part in the *query processing* module. WordNet ontology is used in order to calculate the Semantic Similarity between the Query and the Document vectors, according to Equation 4.4. Figure 4.2 illustrates the architecture of the proposed system. Our search engine is publicly available for use and can be found at *www.ece.tuc.gr/intellisearch*. Notice that this is for experimental use only and can't guarantee up-2-date results, because of the frequency of the web-crawl.

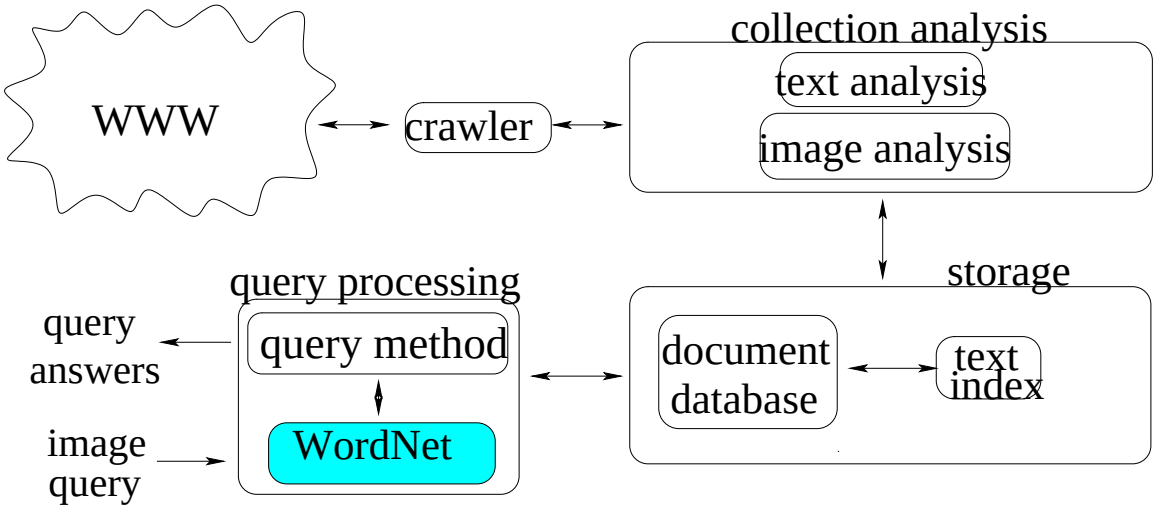

Figure 4.2: Architecture of our Search Engine

Crawler: Implemented based upon Larbin [4], the crawler assembled locally a collection of 2,500,000 pages and a similar number of images. The crawler started its recursive visit of the Web from a set of 200,000 pages which is assembled from the answers of Google image search [5] to 500 queries on topics related to various areas. The crawler worked recursively in breadth-first order and visited pages up to depth 5 links from each origin.

Collection Analysis: The content of crawled pages is analyzed. Text, images, link information (forward links) and information for pages that belong to the same site is extracted. For each image, text descriptions (as term vectors) and image descriptions (as histograms and moment invariants) and its logo-trademark probability are computed.

Storage: Implements storage structures and indices providing fast access to Web pages and information extracted from Web pages (i.e., text, image descriptions and links). For each page, except from raw text and images, the following information is stored and indexed: page URLs, image descriptive text (i.e., alternate text, caption, title, image file name), terms extracted from pages,

[4]http://larbin.sourceforge.net
[5]http://www.google.com/imghp

term inter document frequencies (i.e., term frequencies in the whole collection), term intra document frequencies (i.e., term frequencies in image descriptive text parts), link structure information (i.e., backward and forward links). Image descriptions (i.e., intensity, frequency spectrum and moment invariants, logo-trademark probabilities) are also stored.

The database is implemented in BerkeleyDB [6]. BerkeleyDB is an embedded database engine providing the minimal Application Programming Interfaces (APIs) required to store and retrieve data as efficiently as possible. The mapping of the ERD to database files results into a set of individual files. Each file is associated with one or more indices. There are Hash tables for URLs and inverted files for terms and link information.

Query Processing: Queries are issued by keywords or free text. Query processing starts with the formulation of the initial answer set which is obtained by matching the query term vector with the term vectors extracted from all stored images or documents. The ranking of the results is done according to the selected option of the user. The available options are:

- *Vector Space Model:* This option uses the standard Vector Space Model formula in order to rank the results
- *Vector Space Model with Query Expansion and Term Reweighting:* If this option is selected, the original query is expanded and reweighted according to Equations 4.3 and 4.2 and the ranking is done using again Vector Space Model
- *Similarity Matrix:* With this option, the query is expanded, reweighted according to Equations 4.3 and 4.2 and then we use Equation 4.4 in order to rank the results

4.3 Experiment and Results

An experiment was designed for the evaluation of the performance of our method. We experimented only with queries on images. We believe that *image search* feature of the *Intellisearch* platform is the best environment to compare the methods for many reasons. The main purpose of *Intellisearch* is image searching, the crawl of the web we have in mainly focused on pages with many images and because the text that describes images is more abstract and so it's more difficult to extract information from an image than a web page. Also we believe that *image searching* in a more difficult task than web-page searching, although we have reasons to believe that our method will perform even better in web-page searching.

We searched the web and found a list with the most frequent keywords that users enter in search engines and especially in image search engines like *google images*[7] or *yahoo images*[8]. From those keywords we chose not the 20 first but the 20 we

[6] http://www.sleepycat.com
[7] http://images.google.com
[8] http://www.yahoo.com

believe that are the most representative [9]. When searching for images, queries are more often simple containing one or two keywords, ie "car", "apple", "white house", "baby elephant" as opposed to web-page queries that are often longer than 4 terms. In addition, we decided that the *retrieved entity* is the *image itself* with it's accompanied text in the web page. If the retrieved image is in the topic of the query we mark the answer as correct. Finally, we asked from 5 users to evaluate the results. Each user was given 4 queries and the 50 first results obtained by each method given this query. Each user evaluated the results obtained by all methods on the same queries. The user had the option to choose even "yes" or "no" for each picture indicating correct of wrong result. The methods we tested were the following 5

- *Vector Space Model*
- *Vector Space Model with Query Expansion and Term Reweighting*
- *Similarity Matrix with threshold t=1 for the query expansion*
- *Similarity Matrix with threshold t=0.9 for the query expansion*
- *Similarity Matrix with threshold t=0.8 for the query expansion*

For the *Similarity Matrix Model* and for the *Vector Space Model with Query Expansion and Term Reweighting* the threshold t for bothe the *Expansion and Query Term Re-weighting* step is set to 0.8 indicating that only similar terms would be reweighted and the query is expanded only with very similar terms. Also in the *Vector Space Model with Query Expansion and Term Reweighting* method, the expansion is only done with Synonym terms (t = 1). We also experimented with different thresholds for the *Similarity Matrix Model*. Also note that the Semantic Similarity Method we use in Similarity Matrix Model is the "liEtAl" [21] edge counting method. The reasons we choose this method is firstly because it performed very well achieving very high correlation values and secondly because of it is very fast.

Figure 4.3 we illustrates the precision-recall diagrams for all the methods we tested and in figure 4.4 we present the precision-recall diagram for the *Vector Space Model* and *Similarity Matrix Model with t=1* (which performed better from SMM with a lower threshold). Table 4.3 shows the retrieval times for the most representative methods (VSM and Similarity Mathrix t=1).

[9] we chose keywords that represent objects and not names like "paris hilton" or "ub40" etc

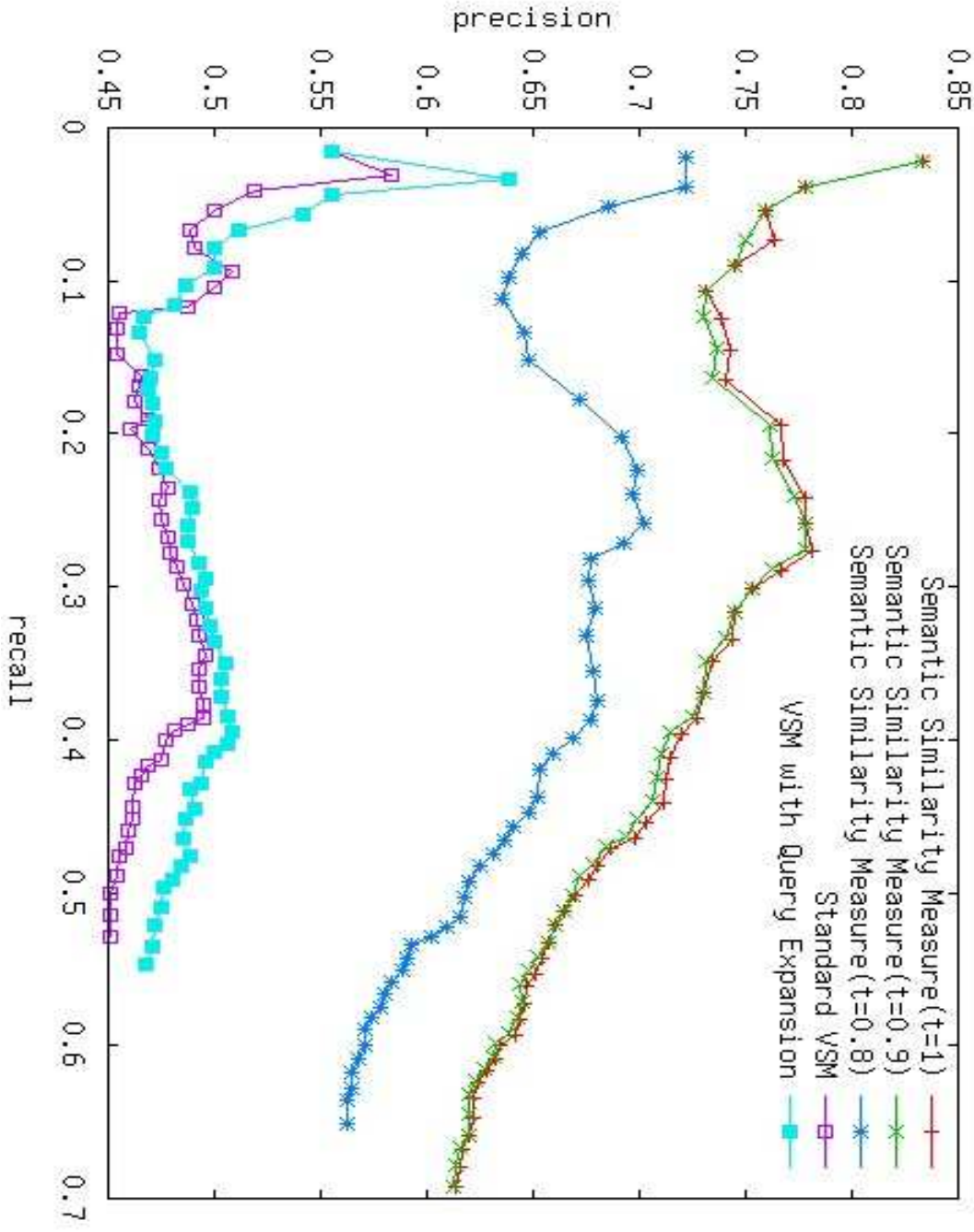

Figure 4.3: Precision-recall diagram of all methods

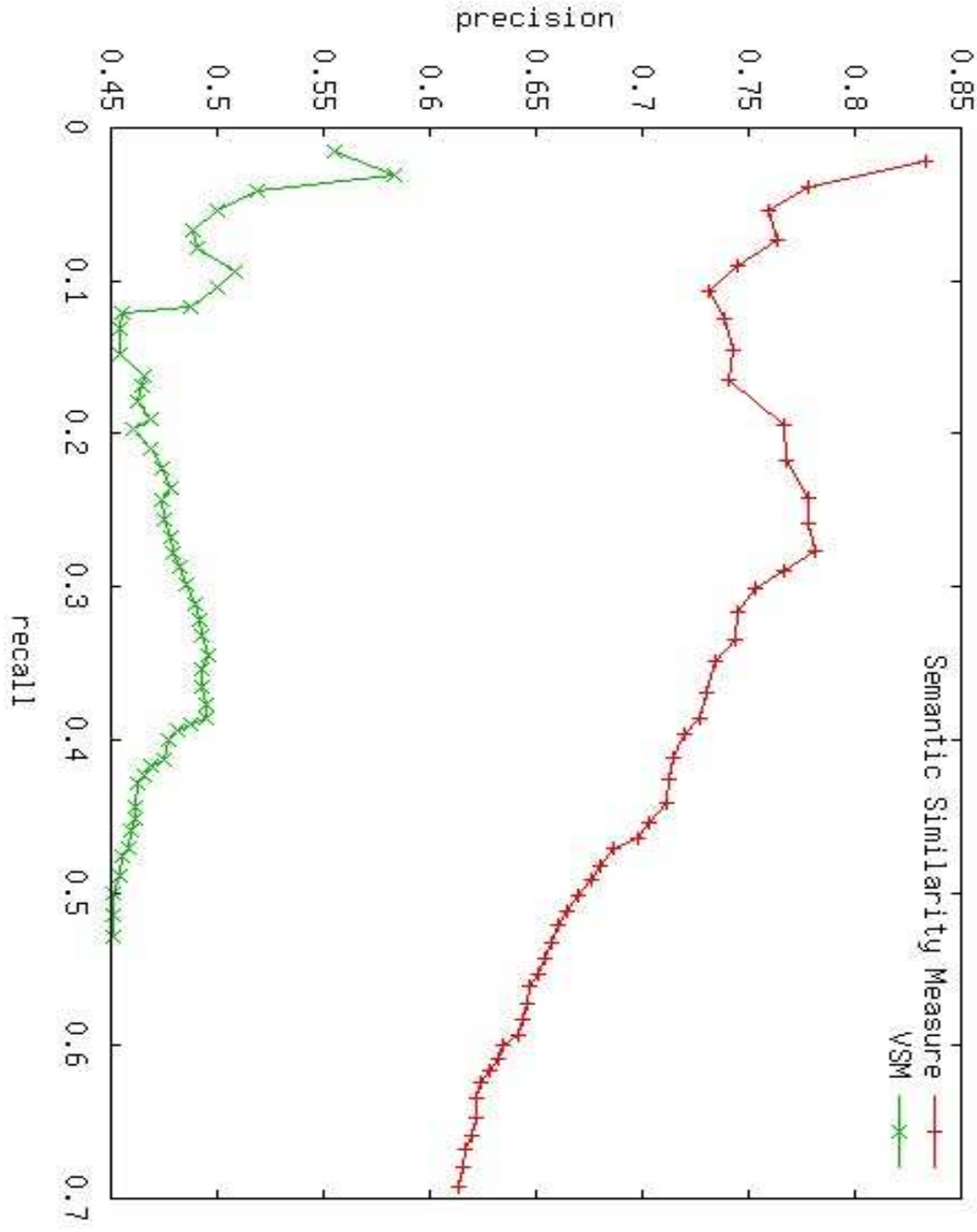

Figure 4.4: VSM vs SMM precision-recall diagram

Method	query	#retrieved results	time (sec)
VSM	airplane	82	0.066
	earth	735	0.189
	woman	597	0.171
	printer	1186	0.190
	house	2546	0.340
	man	2946	0.381
	cpu	3028	0.443
	america	4374	0.581
	car	10862	0.987
	computer	13844	0.999
	greek sunset	652	1.259
	baby elephant	1074	0.947
	linux operating system	33203	1.208
	laptop spare parts	7318	1.1
	rain forest campaign	1774	0.326
SMM	airplane	208	0.975
	earth	16784	27.612
	woman	599	1.692
	printer	1186	2.657
	house	2546	12.056
	man	2951	8.49
	cpu	5366	11.993
	america	52617	70.418
	car	20466	73.521
	computer	13846	21.62
	greek sunset	665	3.31
	baby elephant	1190	6.961
	linux operating system	33424	74.223
	laptop spare parts	7318	25.977
	rain forest campaign	4372	11.514

Table 4.5: Performance in time for VSM and Similarity Matrix t=1 methods

4.4 Evaluation of Results

From Figures 4.3 and 4.4 one can observe the difference in performance between the tested methods. First we have a clear difference in the performance between methods that use Vector Space Model (VSM) and Similarity Matrix Model (SMM). Similarity Matrix Model achieved better performance both in precision and recall compared with Vector Space Model (VSM). This difference touches 30% in the precision of the first result and has an average increment of 20%. Also notice the increased recall in Similarity Matrix Model (SMM) witch has a maximum of 0.7, as opposed to Vector Space Model (VSM) witch has a maximum recall of 0.55. Regarding term expansion, as the threshold increases, meaning that the query vector is expanded with more terms, the performance of the method is decreasing. The best performance (from the three methods that use Similarity Matrix Model) is given by the one with threshold of $t = 1$ and the worst by the one with threshold $t = 0.8$. In the case with $t = 0.9$, we see that the performance is almost the same with the one of $t = 1$. This happens because in order for the query to be expanded with terms other than synonyms, the query terms must be leafs in WordNet taxonomy as the threshold is very high. Also notice that in the case we expand the query with synonyms and use Vector Space Model(VSM) to rank the documents (method WSVSM), the performance is also increased compared with standard Vector Space Model (VSM) method.

In order to have a more complete view for the overall performance of the tested methods, we have to take into account some other factors, like the number of the retrieved documents and the performance in time. In Table 4.3 we present the two most representative methods, the one that uses the standard Vector Space Model (VSM) and the one that uses Similarity Matrix Method (SMM) with threshold $t = 1$, how they perform in time and how many documents they retrieve given a query. It is obviously that Similarity Matrix method is many times slower than Vector Space Model. Methods with smaller thresholds are even slower but in the same scale. his happens for many reasons including the use of WordNet ontology,the increased number of calculations this model needs and the number of documents ranked. Is is also clear that the expansion, even with synonyms, increases the number of the retrieved documents. When we use a smaller threshold, the number of retrieved documents is much higher.

Chapter 5

Epilogue

5.1 Conclusions

In this thesis we experimented with Semantic Similarity Measurement Methods and evaluated their performance both in single and cross ontology experiments. Building upon the method by Rodriguez [45] for measuring semantic similarity among concepts from different ontologies, we proposed a new method for computing semantic similarity between concepts from the same or different ontologies. This method has been shown to perform better than the original method in both single ontology and cross ontology matching.

We concluded that in single ontology experiments using WordNet, *Information Content* family of methods achieve increased performance compared with the other families, as all methods in this category have a correlation (with human judgment) greater than 0.79. For each family, the best performance was given from the methods proposed by Li et al [21] and Leacock-Chodorow [20] for the *Edge Counting Methods*, Jiang et al [16] for the *Information Content Methods* and for *Hybrid and Feature Based Methods*, our proposed method performed best giving a correlation value of 0.75. Regarding cross ontology experiment, we concluded that the method we adopt has an improved performance (an increment of about 12%) compared to the method proposed by Rodriguez [45].

Furthermore, we proposed a model that can be used in modern IR systems in order to retrieve results, that exploits the issue of semantic similarity for enhancing the performance of retrieval. We tested the performance of this model using a complete prototype web and image search engine and compared the results from those obtained by Vector Space Model. From the evaluation of the results, we show that the proposed model achieved at least 20% performance improvement compared to the state-of-the-art approach based on the Vector Space Model.

5.2 Future Work

We consider that experiments using more ontologies (eg. MeSH, Dublin Core) should be performed, in order to see how these methods perform. A bigger variety of results would help us understand which family of methods, and witch specific method, performs better in each situation. Furthermore, detailed investigation of cross-ontology similarity problem, would provide us with information about the difficulties and the

limitations this problem presents and how similarity methods deal with these. This kind of information is critical and can be used in order to design new methods that would overcome those problems and perform better.

Concerning the Similarity Matrix Model, we believe that several things should be further investigated. In order to see more clear how expansion affects the performance of our method, we consider that the same experiments should be performed again but this time using a similarity method (in both Reweight and Similarity Matrix) that has a bigger spreading of results. The method we used to perform the experiments was "liEtAl" method. This method besides its many advantages has a major disadvantage: the results are focused by 70% in the range [1...0.7]. This is a problem in threshold choosing and also in the precision of the calculations. A method like "Leacock and Chodorow [20]", witch has a more wide result spreading and the same correlation with "LiEtAl" method, would let us experiment with even more thresholds and conclude in a more secure way about the query expansion performance.

An evaluation of the tested methods on a larger collection (50G now) with a bigger variety of indexed pages is also a task. We believe that in a larger collection, our method would perform even better. Having an increased number of indexed documents, could help the retrieval of relevant documents that match the expanded terms. Experiments on plain text document instead of image retrieval should also be performed, in order to see how our model performs in larger document vectors (image vectors has a maximum of 200 terms).

Another topic of research would be the further use of Sense Disambiguation techniques in our model. At this time, our model makes the assumption that the user is searching for the most common sense of the entered sense. Sense Disambiguation would help the model to understand which sense of the entered term user is searching for and choose this sense in order to make the calculation needed. By adding an initial Sense Disambiguation step, we believe that the performance of our model would increase even more both in precision and recall.

Furthermore, the expansion step should be investigated. Besides the expansion with semantic similar terms, the expansion with concurrent terms could be a task. Concurrent terms are terms that are often found with some others. For example, we often found term "engine" with term "car". The expansion of the query with such terms could lead in the retrieval of more documents relevant to the user's query.

Although Similarity Matrix Model has an increased performance in both precision and recall, performance in time of this model should be further enhanced. As we show in the results presented in Table 4.3, Vector Space Model (VSM) has a significant advantage in time performance when compared with Similarity Matrix Model. The reasons why this happens are more than easy to understand: more calculations, use of WordNet and increased number of retrieved documents are some of these reasons. In order for our model to be applied in a commercial search engine, time performance should be increased at least at one half the performance of Vector Space Model (VSM). Finetuning the performance of our model can be an even more difficult task than the overall work we presented in this thesis and could demand significant changes in the search engine structure.

Finally, besides the application of such methods in search engines, we believe that

the application in other fields of computer science should be investigated. We believe that the use Semantic Similarity Methods in domains such as document clustering, automatic document categorization or focused crawlers could lead in increased performance of the techniques applied in these fields.

Bibliography

[1] K. Aberer, P. Cudre-Mauroux, and M. Hauswirth. The Chatty Web: Emergent Semanics Through Gossiping. In *Proceedings of the 12th International World Wide Web Conference (WWW'03)*, Budapest, Hungary, 20-24 May 2003. ACM.

[2] S. Alexaki, V. Christophides, G. Karvounarakis, D. Plexousakis, K. Tolle, B. Amann, I. Fundulaki, M. Scholl, and A.-M. Vercoustre. Managing RDF Metadata for Community Webs. In *Proceedings of the ER'00 2nd International Workshop on the World Wide Web and Conceptual Modeling (WCM'00)*, pages 140–151, Salt Lake City, Utah, 9-12 October 2000.

[3] Yigal Arens, Chun-Nan Hsu, and Craig A. Knoblock. Query Processing in the SIMS Information Mediator. In *Advanced Planning Technology*, California, USA, 1996. AAAI Press.

[4] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley Longman, 1999.

[5] R. Beckwith and G. A. Miller. Implementing a Lexical Network. Technical Report 43, Princeton University, 1992.

[6] J.A. Blake and M. Harris. The Gene Ontology Project: Structured vocabularies for molecular biology and their application to genome and expression analysis. In A.D. Baxevanis, D.B. Davison, R. Page, G. Stormo, and L. Stein, editors, *Current Protocols in Bioinformatics*. Wiley and Sons, Inc., New York, 2003.

[7] H. Bulskov, R. Knappe, and T. Andreasen. On Measuring Similarity for Conceptual Querying. In T. Andreasen, A. Motro, H. Christiansen, and H.L. Larsen, editors, *Proceedings of the 5th International Conference on Flexible Query Answering Systems (FQAS'02)*, volume 2522 of *LNAI*, pages 100–111, Copenhagen, Denmark, 27-29 October 2002.

[8] S. Castano, V. Antonelis, and S. De Capitani di Vimercati. Global Viewing of Heterogeneous Data Sources. *IEEE Transactions on Knowledge and Data Engineering*, 13(2):277–297, March/April 2001.

[9] P.R. Cohen and R. Kjeldsen. Information Retrieval by Constrained Spreading Activation in Semantic Networks. *Information Processing and Management*, 23(4):255–268, 1987.

[10] Christianne Fellbaum. *WordNet: An Electronic Lexical Database*. The MIT press, 1998.

[11] Cheng Hian Goh. *Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Sources*. Phd, MIT, 1997.

[12] Bin He, Kevin ChenChuan Chang, and Jiawei Han. Discovering Complex Matchings across Web Query Interfaces: A Correlation Mining Approach. In *Proceedings of the 2004 ACM SIGKDD Conference (KDD'04)*, Seattle, Washington, August 2004.

[13] D.P. Hill, J.A. Blake, J.E. Richardson, and M. Ringwald. Extension and Integration of the Gene Ontology (GO): Combining GO vocabularies with external vocabularies. *Genome Res*, 12:1982–1991, 2002.

[14] I. Horrocks, P.F. Patel-Scheiner, and F. van Harmelen. Reviewing the Design of DAML+OIL: An Ontology Language for the Semantic Web. In *Proceedings of 18th National Conference on Artificial Intelligence (AAAI'02)*, 2002.

[15] Ian Horrocks. DAML+OIL: a Reasonable Web Ontology Language. In *Proceedings of the International Conference on Extending DataBase Technology*, March 2002.

[16] J.J. Jiang and D.W. Conrath. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistic*, Taiwan, 1998.

[17] Lalana Kagal, Grit Denker, Tim Finin, Massimo Paolucci, Naveen Srinivasan, and Katia Sycara. An Approach to Confidentiality and Integrity for OWL-S. In *Proceedings of the 1st International Semantic Web Services Symposium (ISWSS'04)*, AAAI'04 Spring Symposium Series, 22-24 March 2004.

[18] G. Karvounarakis, V. Christophides, D. Plexousakis, and S. Alexaki. Querying RDF Descriptions for Community Web Portals. In *Proceedings of the 17ièmes Journees Bases de Donnees Avancees (BDA'01)*, pages 133–144, Agadir, Maroc, 29 October - 2 November 2001.

[19] K. Knight and S. Luk. Building a Large-Scale Knowledge Base for Machine Translation. In *Proceedings of thr National Conference on Artificial Intelligence (AAAI'94)*, Seattle, WA, 1994.

[20] Claudia Leacock and Martin Chodorow. *Combining local context and WordNet similarity for word sense identification*. In [10], 1998.

[21] Yuhua Li, Zuhair A. Bandar, and David McLean. An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):871–882, July/August 2003.

[22] D. Lin. Principle-Based Parsing Without Overgeneration. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics (ACL'93)*, pages 112–120, Columbus, Ohio, 1993.

[23] P.W. Lord, R.D. Stevens, A. Brass, and C.A. Goble. Investigating Semantic Similarity Measures across the Gene Ontology: the Relationship between Sequence and Annotation. *Bioinformatics*, 19(10):1275–83, 2003.

[24] A. Magkanaraki, S. Alexaki, V. Christophides, and D. Plexousakis. Benchmarking RDF Schemas for the Semantic Web. In *Proceedings of the 1st International Semantic Web Conference (ISWC'02)*, Sardinia, Italy, 9-12 June 2002.

[25] D.L. McGuinness. Conceptual Modeling for Distributed Ontology Environments. In *Proceedings of the 8th International Conference on Conceptual Structures Logical, Linguistic, and Computational Issues (ICCS'00)*, Darmstadt, Germany, 14-18 August 2000.

[26] D.L. McGuinness, R. Fikes, J. Rice, and S. Wilder. An Environment for Merging and Testing Large Ontologies. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR'00)*, Breckenridge, Colorado, USA, 12-15 April 2000.

[27] D.L. McGuinness, R. Fikes, J. Rice, and S. Wilder. The Chimaera Ontology Environment. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI'00)*, Austin, Texas, 30 July - 3 August 2000.

[28] E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. In *Proceedings of the 1st IFCIS International Conference on Cooperative Information Systems (CoopIS'96)*, Brussels, 1996.

[29] G. A. Miller, R. Bechwith, C. Felbaum, D. Gross, and K. Miller. Introduction to WordNet: an on-line lexical database. *International Journal of Lexicography*, 3(4):235–244, 1990.

[30] George Miller and W.G. Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6:1–28, 1991.

[31] P. Mork and P.A. Bernstein. Adapting a Generic Match Algorithm to Align Ontologies of Human Anatomy. In *Proceedings of the 20th International Conference on Data Engineering*, pages 787–790, Boston, USA, 30 March - 2 April 2004.

[32] S.J. Nelson, D. Johnston, and B.L. Humphreys. Relationships in Medical Subject Headings. In C.A. Bean and R. Green, editors, *Relationships in the Organization of Knowledge*, pages 171–184. Kluwer Academic Publishers, New York, 2001.

[33] S.J. Nelson, T. Powell, and B.L. Humphreys. The Unified Medical Language System (UMLS) Project. In A. Kent and C.M. Hall, editors, *Encyclopedia of*

Library and Information Science, pages 369–378. Marcel Dekker, Inc., New York, 2002.

[34] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. Creating Semantic Web Contents with Protege-2000. *IEEE Intelligent Systems*, 16(2):60–71, 2001.

[35] T. O'Hara, N. Salay, M. Witbrock, D. Schneider, B. Aldag, S. Bertolo, K. Panton, F. Lehmann, and et al. Inducing Criteria for Mass Noun Lexical Mappings using the Cyc KB, and its Extension to WordNet. In *Proceedings of the 5th International Workshop on Computational Semantics (IWCS-5)*, Tilburg, The Netherlands, 15-17 January 2003.

[36] A. M. Ouksel. In-Context Peer-to-Peer Information Filtering on the Web. In *SIGMOD Record*, volume 32, pages 65–70, September 2003.

[37] C. Parent and S. Spaccapietra. Issues and Approaches of Database Integration. *Communications of the ACM*, 41(5):166–178, 1998.

[38] A. D. Preece, K.-Y. Hui, W. Gray, P. Marti, Z. Cui, and D. M. Jones. Kraft: An Agent Architecture for Knowledge Fusion. *International Journal of Cooperative Information Systems (IJCIS)*, 10(1-2):171–195, March/June 2001.

[39] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and Application of a Metric on Semantic Nets. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(1):17–30, January/February 1989.

[40] Erhard Rahm and Philip A. Bernstein. A Survey of Approaches to Automatic Schema Matching. *The VLDB Journal*, 10(4):334–350, 2001.

[41] S. Reed and D. Lenat. Mapping Ontologies into Cyc. In *Proceedings of the AAAI'02 Conference Workshop on Ontologies For The Semantic Web*, Edmonton, Canada, July 2002.

[42] O. Resnik. Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity and Natural Language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

[43] R. Richardson, A. Smeaton, and J. Murphy. Using WordNet as a Knowledge Base for Measuring Semantic Similarity Between Words. Technical Report Working paper CA-1294, School of Computer Applications, Dublin City University, Dublin, Ireland, 1994.

[44] N. Rishe, J. Yuan, R. Athauda, S.C. Chen, X. Lu, X. Ma, A. Vaschillo, A. Shaposhnikov, and D. Vasilevsky. Semantic Access: Semantic Interface for Querying Databases. In *Procceedings of the 26th International Conference On Very Large Data Bases*, pages 591–594, 2000.

[45] M.A. Rodriguez and M.J. Egenhofer. Determining Semantic Similarity Among Entity Classes from Different Ontologies. *IEEE Tramsactions on Knowledge and Data Engineering*, 15(2):442–456, March/April 2003.

[46] M. Sabou, D. Richards, and S. van Splunter. An Experience Report on using DAML-S. In *Proceedings of the 12th International World Wide Web Conference Workshop on E-Services and the Semantic Web (ESSW'03)*, Budapest, 2003.

[47] G. Salton and C. Buckley. On the Use of Spreading Activation Methods in Automatic Information. In *Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 147–160. ACM Press, 1988.

[48] Mark Sanderson. Word sense disambiguation and information retrieval. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 49–57, Dublin, IE, 1994.

[49] Nuno Seco, Tony Veale, and Jer Hayes. An intrisic information content metric for semantic similarity in wordnet. Report, University College Dublin, Ireland, 2004.

[50] H. Stuckenschmidt and H. Wache. Context Modeling and Transformation for Semantic Interoperability. In *Knowledge Representation Meets Data Base (KRDB'00)*, 2000.

[51] R. Studer. Knowledge Engineering and Agent Technology. In J. Cuena and et al., editors, *Situation and Perspective of Knowledge Engineering*. IOS Press, Amsterdam, 2000.

[52] A. Tversky. Features of Similarity. *Psycological Review*, 84(4):327–352, 1977.

[53] H. Wache, T. Scholz, H. Stieghahn, and B. Konig-Ries. An Integration Method for the Specification of Rule-Oriented Mediators. In Yahiko Kambayashi and Hiroki Takakura, editors, *Proceedings of the international Symposium on Database Applications in Non-Traditional Enviornments (DANTE'99)*, pages 109–112, Kyoto, Japan, 28-30 November 1999.

[54] H. Wache, T. Vogele, U. Visser, H. Stuckenshmidt, G. Schuster, H. Neumann, and H. Hubner. Ontology-Based Integration of Information - A Survey of Existing Approaches. In *Proceedings of the IJCAI'01 Workshop on Ontologies and Information Sharing*, Seattle, WA, 2001.

[55] Z. Wu and M. Palmer. Verb Semantics and Lexical Selection. In *Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics (ACL'94)*, pages 133–138, Las Cruces, New Mexico, 1994.

Appendix A

Appendix

A.1 WordNet XML web-app

WordNet's XML webapp is a simple web-application that uses the WordNet's database and search machine to represent the data returned from the machine in a form that is both understandable from human and machines. The webapp runs with the help of Tomcat server in any machine having WordNet's database installed. User inputs a concept to find and the webapp returns the results in a well structured form , filtered from any unwanted information , that can be later used from any application that can read XML with a given DTD file. The use of this app is a need, if we want to make efficient use of WordNet's database to extract information regarding a term and use them to make any computation. This application is written by Bernard Bou and is freely available for download at *http://wnws.sourceforge.net/*

A.1.1 WordNet XML Schema

Below we present the XML-Schema used from the WordNet webapp and fully describes a WordNet concept. The schema is in DTD format but it is also available in XSD format. In order to use the developed software with other ontologies than WordNet, the only limitation is that the input files should conform to the following schema.

```
<!-- DTD for WordNet v 1.0 20030430 -->

<!ELEMENT word (#PCDATA | key | pos)*>
<!ATTLIST word
>

<!ELEMENT key (#PCDATA)>
<!ATTLIST key
>

<!ELEMENT pos (sense*)>
<!ATTLIST pos
        name CDATA #REQUIRED
```

```
>

<!ELEMENT sense (synset,links)>
<!ATTLIST sense
        number CDATA #REQUIRED
>

<!ELEMENT synset (item*,defn)>
<!ATTLIST synset
        number CDATA #IMPLIED
>

<!ELEMENT item (#PCDATA)>
<!ATTLIST item
>

<!ELEMENT defn (#PCDATA)>
<!ATTLIST defn
>

<!ELEMENT links (antonym?,hypernym?,hyponym?,entail?,similar?,
member-holonym?,substance-holonym?,part-holonym?,member-meronym?,
substance-meronym?,part-meronym?,meronym?,holonym?,cause?,
participle?,see-also?,pertainym?,attribute?,verb-group?,
(derivation?)*,(classification?)*,(class?)*)>
<!ATTLIST links
>

<!ELEMENT antonym (synset+,antonym*)>
<!ATTLIST antonym
>

<!ELEMENT hypernym (synset+,hypernym*)>
<!ATTLIST hypernym
>

<!ELEMENT hyponym (synset+,hyponym*)>
<!ATTLIST hyponym
>

<!ELEMENT entail (synset+,entail*)>
<!ATTLIST entail
>

<!ELEMENT similar (synset+,similar*)>
```

```
<!ATTLIST similar
>

<!ELEMENT member-holonym (synset+,member-holonym*)>
<!ATTLIST member-holonym
>

<!ELEMENT substance-holonym (synset+,substance-holonym*)>
<!ATTLIST substance-holonym
>

<!ELEMENT part-holonym (synset+,part-holonym*)>
<!ATTLIST part-holonym
>

<!ELEMENT member-meronym (synset+,member-meronym*)>
<!ATTLIST member-meronym
>

<!ELEMENT substance-meronym (synset+,substance-meronym*)>
<!ATTLIST substance-meronym
>

<!ELEMENT part-meronym (synset+,part-meronym*)>
<!ATTLIST part-meronym
>

<!ELEMENT meronym (synset+,meronym*)>
<!ATTLIST meronym
>

<!ELEMENT holonym (synset+,holonym*)>
<!ATTLIST holonym
>

<!ELEMENT cause (synset+,cause*)>
<!ATTLIST cause

<!ELEMENT participle (synset+,participle*)>
<!ATTLIST participle
>

<!ELEMENT see-also (synset+,see-also*)>
<!ATTLIST see-also
>
```

```
<!ELEMENT pertainym (synset+,pertainym*)>
<!ATTLIST pertainym
>

<!ELEMENT attribute (synset+,attribute*)>
<!ATTLIST attribute
>

<!ELEMENT verb-group (synset+,verb-group*)>
<!ATTLIST verb-group
>

<!ELEMENT derivation (synset+,derivation*)>
<!ATTLIST derivation
>

<!ELEMENT classification (synset+,classification*)>
<!ATTLIST classification
>

<!ELEMENT class (synset+,class*)>
<!ATTLIST class
>
```

A.2 Tools we use

A.2.1 Castor

Castor, unlike the other two main XML APIs, DOM (Document Object Model) and SAX (Simple API for XML) which deal with the structure of an XML document, enables one to deal with the data defined in an XML document through an object model which represents that data. Castor XML can marshal almost any "bean-like" Java Object to and from XML. In most cases the marshalling framework uses a set of ClassDescriptors and FieldDescriptors to describe how an Object should be marshalled and unmarshalled from XML. XML Class descriptors provide the marshalling framework with the information it needs about a class in order to be marshalled to and from XML. For those not familiar with the terms "marshal" and "unmarshal", it's simply the act of converting a stream (sequence of bytes) of data to and from an Object. The act of "marshalling" consists of converting an Object to a stream, and "unmarshalling" from a stream to an Object Two main classes are consisted in Castor XML tool, org.exolab.castor.xml.Marshaller and org.exolab.castor.xml.Unmarshaller. The below figure A.1 shows the Castor XML "binding" framework.

Although it is possible to rely on Castor's default behavior to marshal and unmar-

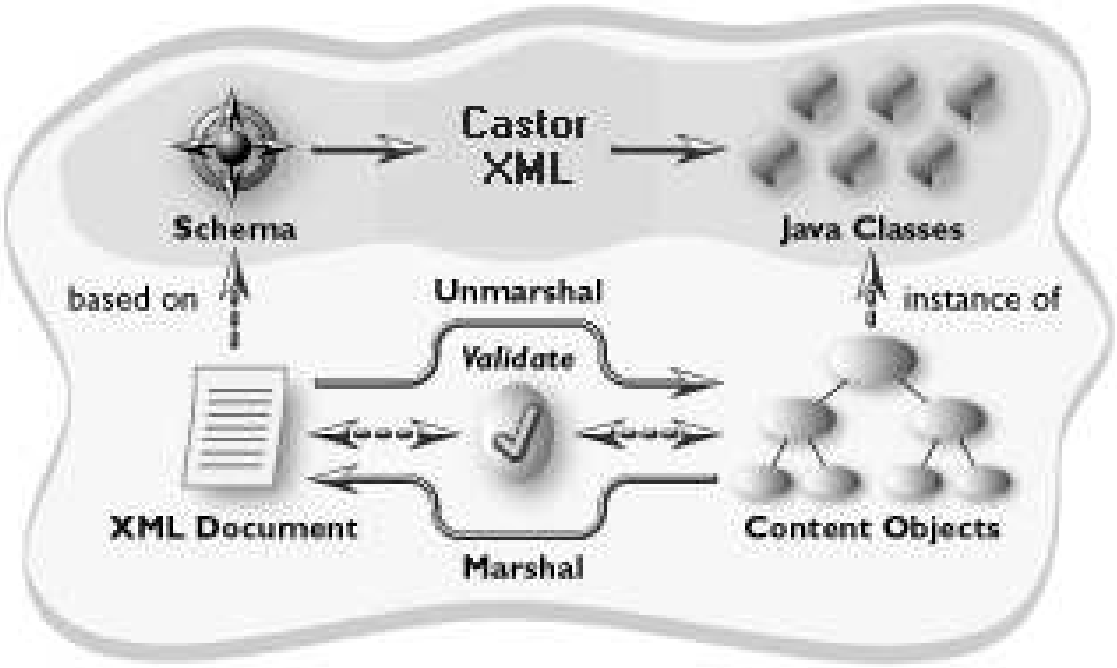

Figure A.1: How Castor XML works

shal Java objects into an XML document, it might be necessary to have more control over this behavior. For example, if a Java object model already exists, Castor XML Mapping can be used as a bridge between the XML document and that Java object model. Castor allows one to specify some of its marshalling/unmarshalling behavior using a mapping file. This file gives explicit information to Castor on how a given XML document and a given set of Java objects relate to each other. The mapping file describes for each object how each of its fields have to be mapped into XML. A field is an abstraction for a property of an object. It can correspond directly to a public class variable or indirectly to a property via some accessor methods (setters and getters). So, with the marshalling and unmarshalling functions plus the mapping file (optional) the "binding" of XML elements - Java objects is rather easy. Castor can be reached at *http://www.castor.org*

A.2.2 Lucene

Lucene is a Java-based open source toolkit for text indexing and searching. It is easy to use, flexible, and powerful – a model of good object-oriented software architecture. Powerful abstractions and useful concrete implementations make Lucene very flexible, and allow new users to get up and running quickly and painlessly. We use lucene in order to perform various operations needed by the search engine we use as part of this work (indexing etc). Lucene if freely available at *http://lucene.apache.org*

A.3 Software Description

The software was developed entirely in Java language for several reasons, including the amenities it provides. A brief description of the API developed and the requirements in order to run the software follow. In order to make use of the software, two things are needed

1. The xml files describing the terms we want to compare
2. The .jar file that contains the developed libraries

To run the software as a standalone program, include the .jar provided in the classpath and set environment variable ONTO_PATH to the directory in which the xml files are. The user has the option to select which senses of the terms to compare and which part-of-speech to use (ie. verb, noun). Some usage samples follow.

Comparing all senses of term "cat" with term "dog" using Li et al method:

```
java gr.tuc.softnet.username.thesiscode.Main cat dog liEtAl

DEBUG [main] (Main.java:180) - cat vs dog  0.44865853939100275
```

Comparing the first sense of noun "car" with the third sense of noun "road" using Jiang et al method:

```
java gr.tuc.softnet.username.thesiscode.Main n#car#0 n#train#2 jiangEtAl

DEBUG [main] (DistanceFactory.java:135) - Comparing sense # : 0 of concept :
car with sense # : 2 of concept : train using method : jiangEtAl
DEBUG [main] (Main.java:180) - car vs train  0.0
```

To get full usage details of the program, just type:

```
java gr.tuc.softnet.username.thesiscode.Main
```

Now, in order to use the functionality of the software in any application, just include the libraries in the code, by writing something like:

```
import gr.tuc.softnet.username.thesiscode.*;
import gr.tuc.softnet.username.thesiscode.tool.*;
```

somewhere in the beginning of the code. For a full list of the API and the provided functionality, the interest reader can refer to the documentation (JavaDoc) of the software. Here, we present just the basic system functionality with an example of usage.

```
double runMain(String first, String second, String method)}
```

The above function, is the basic function of the system. Provided two terms and a method it returns the similarity of the two terms as calculated from the specified method.
Below, we present the code of a servlet that makes use of this functionality in order to compute the semantic similarity between two terms. Figures A.2 and A.3 demonstrate the servlet running.

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
import gr.tuc.softnet.username.thesiscode.*;
import gr.tuc.softnet.username.thesiscode.tool.*;


public class SimilarityServlet extends HttpServlet {
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
      throws ServletException, IOException {
    Main m = new Main();
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println(
                "<BODY>\n" +
                "<UL>\n" +
                "  <LI>first term: "
                + request.getParameter("firstTerm") + "\n" +
                "  <LI>second term: "
                + request.getParameter("secondTerm") + "\n" +
                "  <LI>method: "
                + request.getParameter("method") + "\n" +
                "</UL>\n" +
                "</BODY></HTML>");
   String[] toPass = new String[3];
   toPass[0] = request.getParameter("firstTerm");
   toPass[1] = request.getParameter("secondTerm");
   toPass[2] = request.getParameter("method");
   out.println(m.runMain(toPass[0],toPass[1],toPass[2]));
   }

  public void doPost(HttpServletRequest request,
                     HttpServletResponse response)
      throws ServletException, IOException {
    doGet(request, response);
```

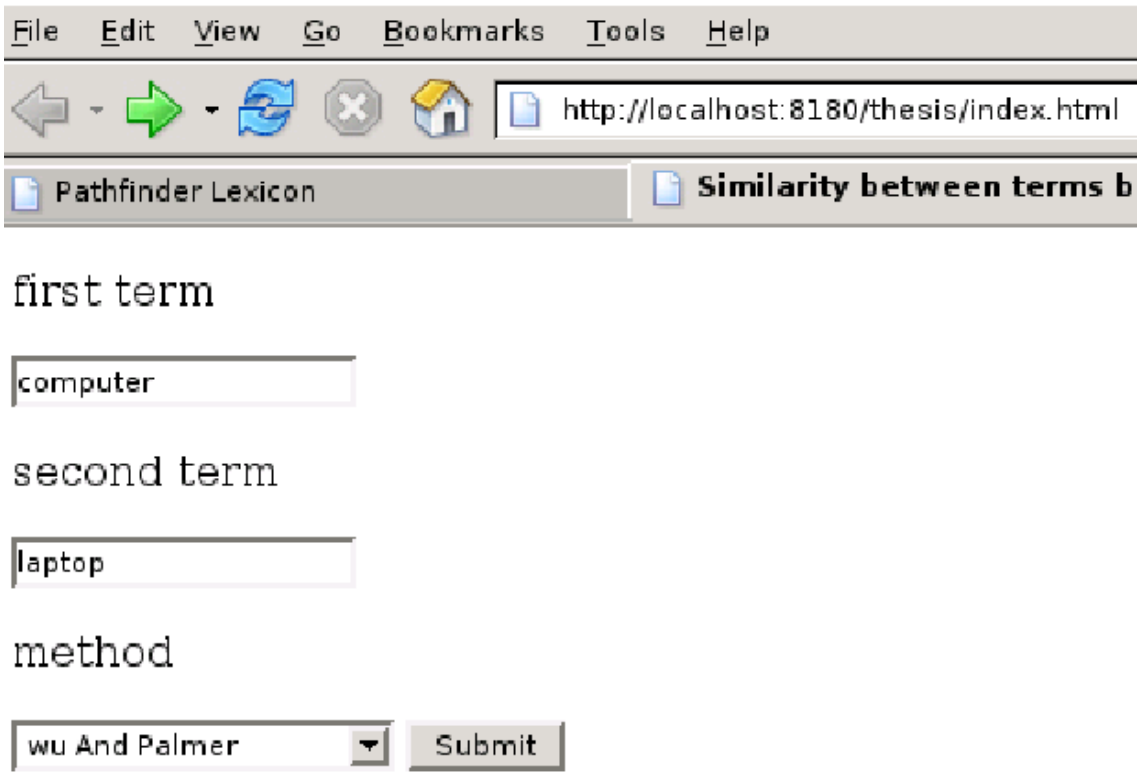

Figure A.2: Example Servlet Request

```
  }
}
```

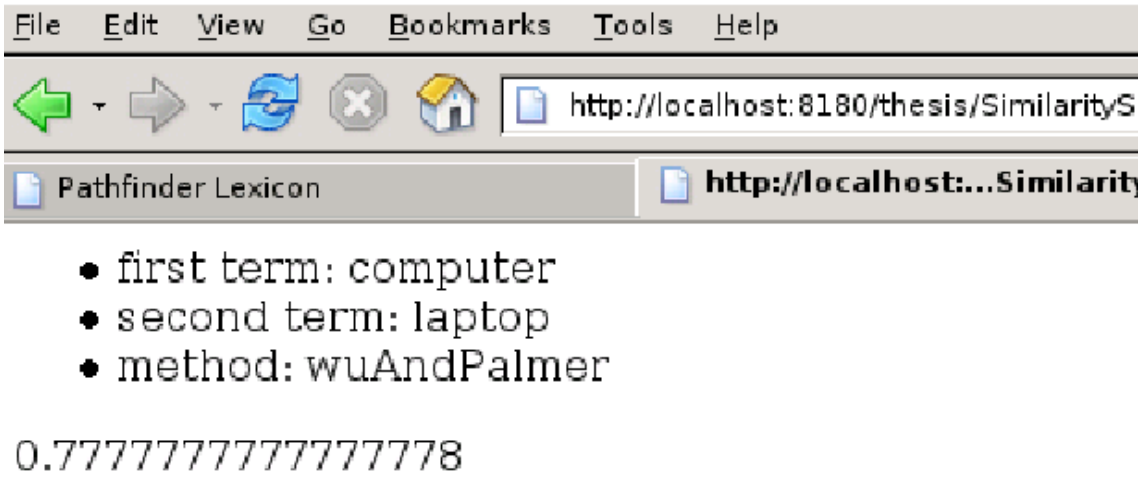

Figure A.3: Example Servlet Result