A Thesis by

John D. King, ID 02550105, B.I.T. (Q.U.T.), M.I.T. (Q.U.T.)

Supervisors: **Associate Professor Yuefeng Li, Professor Peter Bruza, Dr. Richi Nayak**

Search Engine Content Analysis

In Fulfillment
of the Requirements for the Degree
Doctorate of Information Technology

School of Software Engineering and Data Communications
Faculty of Information Technology
Queensland University of Technology, Brisbane, Australia
Submitted on December 15, 2008

j5.king@qut.edu.au
http://kingsonline.com.au/

Keywords

web intelligence, ontology, hierarchal classification, taxonomy, collection selection, search engines, data mining

Search Engine Content Analysis

by

John D. King

Abstract

Search engines have forever changed the way people access and discover knowledge, allowing information about almost any subject to be quickly and easily retrieved within seconds. As increasingly more material becomes available electronically the influence of search engines on our lives will continue to grow. This presents the problem of how to find what information is contained in each search engine, what bias a search engine may have, and how to select the best search engine for a particular information need. This research introduces a new method, search engine content analysis, in order to solve the above problem. Search engine content analysis is a new development of traditional information retrieval field called collection selection, which deals with general information repositories. Current research in collection selection relies on full access to the collection or estimations of the size of the collections. Also collection descriptions are often represented as term occurrence statistics.

An automatic ontology learning method is developed for the search engine content analysis, which trains an ontology with world knowledge of hundreds of different subjects in a multi-level taxonomy. This ontology is then mined to find important classification rules, and these rules are used to perform an extensive analysis of the content of the largest general purpose Internet search engines in use today. Instead of representing collections as a set of terms, which commonly occurs in collection selection, they are represented as a set of subjects, leading to a more robust representation of information and a decrease of synonymy.

The ontology based method was compared with ReDDE (Relevant Document Distribution Estimation method for resource selection) using the standard R-value metric, with encouraging results. ReDDE is the current state of the art collection selection method which relies on collection size estimation. The method was also used to analyse the content of the most popular search engines in use today, including Google and Yahoo. In addition several specialist search engines such as Pubmed and the U.S. Department of Agriculture were analysed. In conclusion, this research shows that the ontology based method mitigates the need for collection size estimation.

Contents

List of Figures

List of Tables

Abbreviations

broker ..... A computer program that takes a query and distributes it to a number of collections, then merges the results from the collections together.

collection ..... A set of documents. In this research the documents are always in electronic format.

Collection Description The analysis and profiling of a collection.

Collection Fusion . The merging together of the results returned by multiple collections.

Collection Selection Collection selection is the selection of an optimal subset of collections (or databases) from a large set of collections for the purpose of reducing costs associated with Distributed Information Retrieval.

data retrieval system A data retrieval system differs from an information retrieval system in the acceptable level of incorrect documents allowed in the set of retrieved items. A data retrieval system is defined on mathematical principles using structured data and thus no incorrect documents are allowed in the set of retrieved items.

Deep Web ..... Any part of the World Wide Web that cannot be reached by following hyperlinks.

Deep Web Collection Selection The selection of an optimal set of collections from the deep web to serve an information need.

disjoint collections A disjoint system occurs when there is little overlapping information between the collections.

Distributed Information Retrieval Distributed Information Retrieval is the querying, ranking and returning of data from a set of information brokers located on different

servers.

document ..... A set of terms. In this research documents are electronic and use natural language.

Document selection Document selection is the selection of a set of documents from the results of each collection.

hyperlink ..... A reference or a pointer to other electronic information. In HTML it is commonly implemented as an anchor tag.

Inverse document frequency The inverse of the number of times that a term occurs in a document or a collection.

inverted index .... An index of all the terms in all the stored documents, along with their statistics and/or locations. Used for fast retrieval of information.

Latent Semantic Analysis A vector space model for analysing relationships between documents.

Localised Information Retrieval Information Retrieval where all the documents reside in a central location.

meta search engine A computer program that takes a query and distributes it to a number of search engines, then merges the results from the search engines together.

mismatch ..... The number of relevant documents not retrieved divided by the total number of relevant documents.

Ontology ..... An ontology can be defined as a shared conceptualisation of knowledge. For the purposes of this research, an ontology is defined to be a hierarchical taxonomy, whereby the nodes correspond to subjects. Each subject is characterized by a set of subject classification terms.

overlapping collections A overlapping system occurs when the same information exists between different collections.

overload ..... The number of irrelevant documents retrieved divided by the total number of irrelevant documents in the collection.

polysemy ..... Polysemy is a word that has multiple meanings.

precision ..... The number of relevant documents retrieved divided by the total number of documents retrieved.

Query Probing .... The process of sending a query to an uncooperative collection. In Web-based search engines this often involves modifying the GET URL and sending this altered URL to the search engine.

Query sampling ... See "Query Probing"

recall ..... The number of relevant documents retrieved divided by the total number of relevant documents in the collection.

relevance ..... A measure of how well a document or a collection fits a user need.

Search engine content analysis Search engine content analysis is the analysis of web based collections which generally only allow access through a HTML Form interface.

search engine selection Search engine selection is the selection of an optimal set of search engines which serve an information need.

Singular value decomposition Singular value decomposition is a matrix factorisation method which draws together related concepts in a matrix.

spider ..... A computer program that traverses the surface web by recursively following hyperlinks.

subject ..... A subject can be difficult to define and abstract to some degree, while still containing a strong taxonomic structure. A subject may have sub and super subjects, with super-subjects being a higher abstraction of the subject.

Surface Web ..... Any part of the World Wide Web that can be reached by following hyperlinks.

synonymy ..... Polysemy is where many words have the same meanings.

term ..... The atomic unit of Information Retrieval. A word or word pair.

term frequency .... The number of times that a term occurs in a document or a collection.

Statement of Original Authorship

The work contained in this thesis has not been previously submitted for a degree or diploma at any other higher education institution. Parts of this thesis have been taken from my earlier writings. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

John D. King

December 2008

Acknowledgments

Many thanks to my principal supervisor Associate Professor Yuefeng Li for his advice and constant support. Thanks to my associate supervisors Professor Peter Bruza and Dr. Richi Nayak for guiding my research. Thanks to my parents for their support. Thanks to Michael Gardner and Rae Westbury for their input and reviews of my writing. Thanks to Blackwell North America Inc for allowing me to use the Q.U.T. library dataset. Thanks to the Q.U.T. Library Staff for helping to acquire the dataset. Also thanks to the Q.U.T. High Performance Computing staff for their assistance with the experiments.

Associated Publications

Journal Papers

J. D. King, Y. Li, X. Tao, and R. Nayak, "Mining world knowledge for analysis of search engine content," In *Web Intelligence and Agent Systems: An International Journal*, Vol. 5, No. 3, pp.233-253, 2007.

J. D. King, Yuefeng Li, P. D. Bruza, and Richi Nayak. "Preliminary investigations into ontology-based collection selection." In *Australian Journal of Intelligent Information Processing Systems*, Vol 9, No. 2 pp 33 - 40, 2006

Conference Papers (Reverse Chronological Order)

J. D. King, Yuefeng Li, P. D. Bruza, and Richi Nayak. "Preliminary investigations into ontology-based collection selection." In *Proceedings of the Eleventh Australasian Document Computing Symposium*, pages 33 - 40, Brisbane, Australia, December 2006.

J. D. King, "Large scale analysis of search engine content," in *The Fourth International Conference on Active Media Technology*, Brisbane, Australia, 2006.

A. Woodley, C. Lu, T. Sahama, J. D. King, and S. Geva, "Queensland University of Technology at TREC 2005," in *Proceedings of Fourteenth Text Retrieval Conference*, Gaithersburg, USA, 2005.

X. Tao, J. D. King, and Y. Li, "Information fusion with subject-based information gathering method for intelligent multi-agent models," in *The Seventh International Conference on*

Information Integration and Web-Based Applications and Services, Kuala Lumpur, Malaysia, 2005.

Chapter 1

Introduction

The internet contains hundreds of thousands of electronic collections that often contain high quality information (Bergman 2001; Chang, He et al. 2004). However when there are many collections available it is often too expensive to send every query to every collection in order to find the most suitable collection to search. Collection selection tries to build a profile on each collection with the aim of reducing the number of collections searched for each query. This profile then can be used to match a query to the best possible collection.

It is fairly easy to profile a collection that publishes detailed statistics of its contents. However most collections on the internet are uncooperative. They generally do not publish detailed information about what they contain.

In an uncooperative environment each collection is effectively a "black box" where nothing is known about its inner workings, so special methods must be used to discover what it contains. Information in the form of a query is sent to the black box, and the information that is returned analysed in order to try to understand what it contains.

Current collection selection methods often select query terms at random from a dictionary and attempt to profile each collection based on these terms (Thomas and Hawking 2007). These methods are problematic, because it is easily possible to miss specialist terms that will help to accurately classify a collection. Instead the careful selection of the query terms is important to understanding what information a collection contains.

Search engine content analysis is a subcategory of *collection selection*, which deals with

collections only accessible through a HTML search interface. Search engine content analysis is similar in important ways to collection selection, in that they both try to find the best set of resources for an information need.

While many search engine usability [JBS07, JSS00], query log [BJC+04, BJC+07], overlap [LG98, Dog07, Din96], and selection [LYM+98, LMYR00, SH03] studies have been published there has been little work on analysing and comparing the content of the major search engines. This is because of the unavailability of a method which can handle the huge range of subjects contained in the major search engines. Traditional information retrieval methods which require direct access to a collection cannot be used on large search engines as generally the only interface to the search engine is through a HTML search form.

1.1 Problem Statement

This research tries to solve the problem of how to select the best collection of information for a particular information need. By itself this is a difficult problem which involves a great deal of communication and cooperation between the collections and the broker which controls the analysis and selection of the collections. To make things even more difficult, the method of collection selection used in this thesis is performed within the constraints of an uncooperative search engine environment where there is no communication and cooperation between the collections and the broker.

There are a number of problems facing efficient collection selection and search engine content analysis. These problems will be briefly addressed here and covered in more detail later. The first problem is that there are too many electronic search engines available to track by hand. The massive growth of the search engines is parallel to the growth in information becoming available on the deep web. Search engines often contain dynamic data, which constantly grows and changes. Another problem is that many search engines are not indexable by traditional methods. Only surface web documents have a hyperlinking system that supports easy indexing. A further problem is that the search engine selection system needs to be able to respond quickly to user queries. Users expect near instantaneous responses to their queries. This research en-

ables a fast and simple way of finding a search engine's content, and can quickly suggest the best search engine for a user need.

A major problem with previous collection selection research is that it often required cooperative collections or full knowledge of each collection. While they work well when supported, these collection selection techniques are rarely supported by search engines. Even when there is cooperation, there is little protocol standardisation between collections and the collection descriptions are often incomparable. CORI (Collection Retrieval Inference Network) [CLC95] and GlOSS (Glossary of Servers Server) [GGMT99] both rely on cooperation between the search broker and the collections. Meta data is gathered, term statistics retrieved, messages passed, and software is distributed. However, in reality few search engines are willing to co-operate by giving detailed statistics about themselves, allowing full access to their data, or allowing foreign software to reside on their servers. The methods compared in this thesis do not require co-operation between the search broker and the collections. Other more recent collection selection methods such as ReDDE [SC03] rely on collection size estimation, which is difficult, error prone, and can be computationally expensive.

In this research the phrases "collection selection" and "search engine selection" are used interchangeably. "Collection" and "search engine" are also used interchangeably. This is because this research can be used for both "collection selection" and "search engine selection". A traditional collection often differs from a search engine by the interface that is presented to the user, and a collection is generally accepted to cover a wider scope of information sources than a search engine. A traditional collection may be a database or a collection of PDF documents. A search engine in this research is defined as system that offers an interface to a database of HTML documents that have been retrieved from the World Wide Web.

Also in this research "deep web search engine" and "general purpose search engine" are used interchangeably. A deep web search engine is generally a specialised source of information, for example the deep web search engine PubMed covers only medical subjects, while the general purpose search engine like Google covers many different subjects.

1.2 Contributions

This multi-disciplinary thesis draws from the fields of web intelligence, collection selection, taxonomies, ontologies, ontology learning, data mining, and singular value decomposition. Each of these areas will be briefly outlined throughout this thesis when relevant to the topic at hand. Several contributions are made for the fields of ontologies, web intelligence(WI), information retrieval (IR) and search engine analysis. The first is the ability to create a large ontology for representation of world knowledge. The second contribution is the method of selecting query probe terms from the ontology. The third is the analysis of collections and large search engines using an ontology. The experiments performed show that this analysis is effective. This is the first time that an ontology based collection selection method has produced good results. The feature of the third contribution is that we make use of expert human knowledge. The fourth contribution is a method for ontology-based collection selection which does not rely on estimation of collection sizes. The fifth is a method for the analysis of the similarity between the content of collections motivated from linear algebra. This method calculates how similar different collections are to each other.

The collections and search engines discussed in this research are web based, with the only interface being a HTML form where a query is entered, and a set of HTML web pages which comprise the results of the query.

1.3 Motivations and Aims

There are several motivations for this work. The first motivation is that little is known about the subjects contained in the deep web search engines, especially the bias towards or against certain subjects each engine may have. The second motivation is that statistical search engine selection models [LYM+98, LMYR00, SH03] do not take into account world knowledge. The third motivation is the difficulty of manually creating a large ontology for representation of world knowledge which is broad and deep enough to cover the subjects encountered in the

major search engines. The fourth is that current statistical methods for selecting search engines are not able to find hidden patterns of similarity between the search engines.

The aim of this research is to produce results using noncooperative collections that approach the results of working with cooperative collections. Another aim of this research is to take a user query and automatically select the best search engines to search. This research uses a combination of web intelligence, data mining, distributed information retrieval, and ontologies to solve this problem. The anticipated outcome of this research is a system that can work with multiple search engines and perform at levels close to a centralised system. Another aim of this research is to provide a method of judging a relevance of a search engine to a query.

1.4 Research Approach

Firstly an ontology based collection selection method is introduced. This method is then compared with the leading collection selection method, which the ontology based collection selection method matches or improves on in the majority of cases. Having proven that the ontology-based collection selection method is effective, it is then applied to the real world application of search engine content analysis.

A subject based approach to search engine and collection selection is taken. Items in the ontology are classified into subjects. Traditional information retrieval represents documents and collections as terms and their related statistics. However instead of representing search engines as a set of terms, this research represents them as a set of subjects, which is a highly efficient representation, and leads to a more robust representation of information and a decrease of synonymy. Representing search engine descriptions as subjects, rather than terms has considerable advantages. For example, consider a computing dictionary from the 1970's and a computing dictionary from today; they both cover the same subject, yet there are wildly different sets of terms within each dictionary. By representing search engines by their subjects instead of their terms, a more robust system results, that is more adaptive in the face of technological and social change. In addition, by representing collections as a hierarchy of subjects instead of terms, users can easily navigate an environment containing many thousands of collections. This is

significant because many other information retrieval systems use terms to describe documents and collections, and are therefore significantly larger and less agile.

Table 1.1 shows the search engines analysed in this research. A comparison of the search engines across hundreds of subjects was performed, and the similarities and differences between the engines were analysed.

1.5 Proposed Solution

This research takes an ontology based approach to collection selection. An ontology is trained and mined in order to extract classification rules. The ontology contains a broad set of terms and contains many highly technical terms and acronyms that do not exist in standard English dictionaries. The rules are then used to classify each collection using a system that finds clusters of high quality information in the collections by sending groups of highly specific classification query terms to each search engine and transforming the results into metadata. The metadata is subject-based, making it short, concise, understandable by humans, and easy to maintain. Each entry is in the format of subject code, support and confidence. The subject code will specify exactly what part of the hierarchy portion occurs in. The support and confidence values will specify the belief in the subject classification. Once the collections have been profiled it is simple to return the best collection or set of collections for an information need.

An example of a low level analysis is pictured in Figure 1.2, which shows the search engines most related to *Medical Sciences and Medicine*. The y-axis represents normalised relevance to

Title	Abbreviation	URL
Google Search	Google	Google A.P.I.
Wikipedia	Wikipedia	http://en.wikipedia.org/
Microsoft Live Search	Live	http://www.live.com/
U.S. Department of Commerce	Commerce	http://www.commerce.gov/
USA Search.gov	FirstGov	http://usasearch.gov
Accoona Search	Accoona	http://www.accoona.com/
PubMed	PubMed	http://www.ncbi.nlm.nih.gov/sites/entrez
U.S Department of Agriculture	Agriculture	http://www.usda.gov/
Yahoo Search	Yahoo	Yahoo A.P.I.

Table 1.1: The Search Engines Analysed In This Thesis

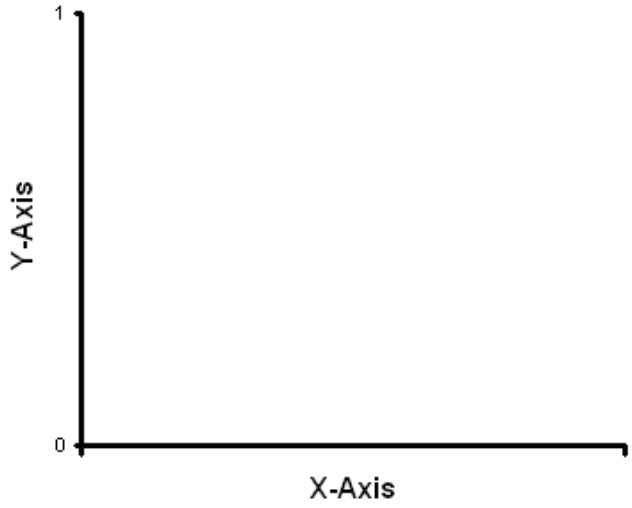


Figure 1.1: X-axis and y-axis.

the subject, with a score of 1.0 being perfect relevance. The x-axis shows the search engines compared. It is easy to see that PubMed is the best search engine for this subject. (For each of the graphs in this thesis the x-axis and y-axis are defined by Figure 1.1)

The subject taxonomy used in this research is multi-level, allowing both a meta and micro analysis of a search engine. Using a high level analysis it was found that PubMed was best for the subject group *Natural Sciences & Mathematics* which includes some subjects related to medicine and biology, the U.S. Department of Agriculture was best for the subject group *Applied Sciences* which contains some subjects related to agriculture and produce, and the U.S. Department of Commerce was best for the subject group *Social Sciences* which contains some subjects related to commerce and statistics. An example of the high level analysis is pictured in Figure 1.3. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance. The x-axis shows the search engines compared. This analysis covers the one hundred subjects in the third level of the taxonomy. It is clear that PubMed is the best search engine by for this high level subject group.

As part of the ontology based solution Singular Value Decomposition was used to show how similar search engines are to each other. Singular Value Decomposition can quickly and easily find hidden relationships between the search engines. In previous experiments, search engines using Google's results for their own results were automatically identified. In this research each

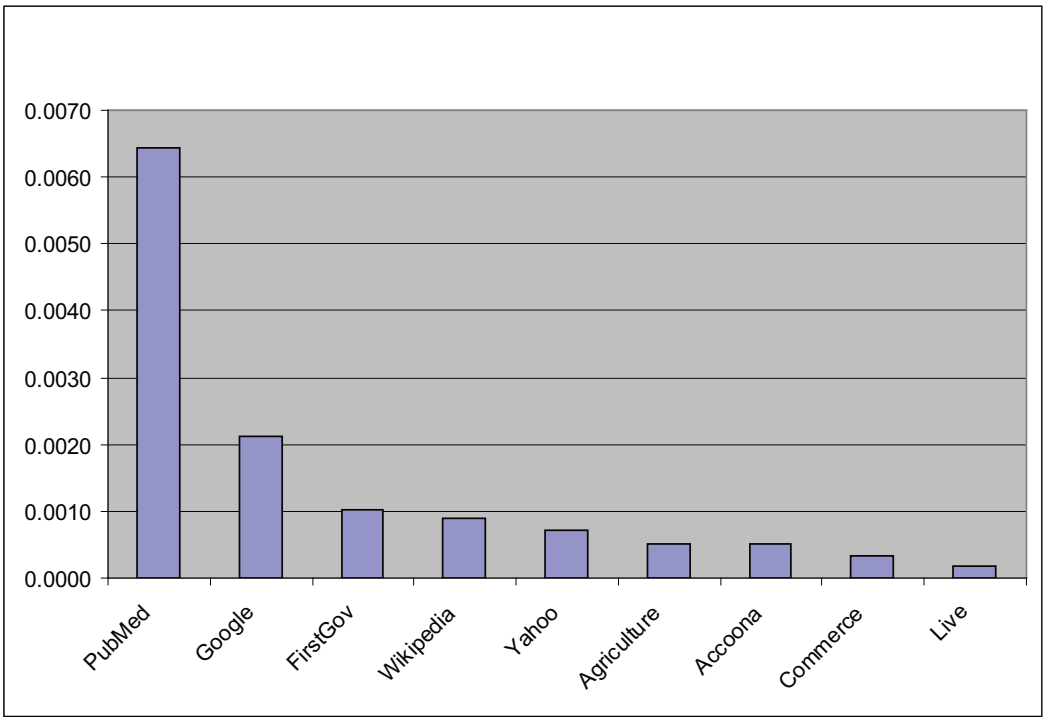


Figure 1.2: 610 Medical Sciences - Medicine

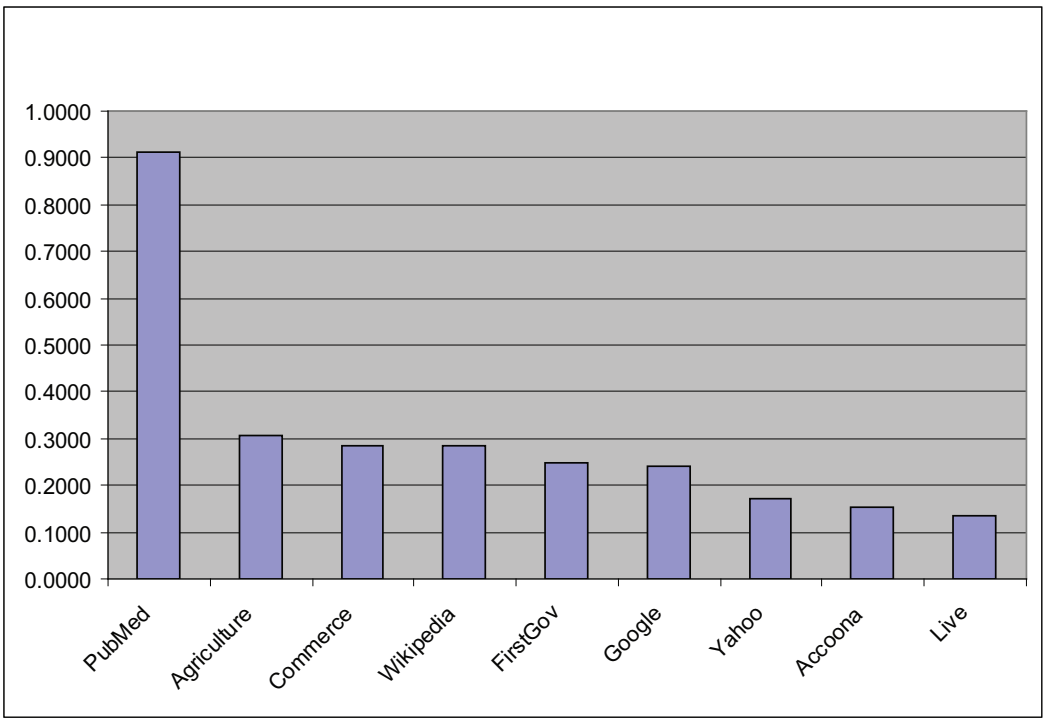


Figure 1.3: 500 Natural sciences and mathematics

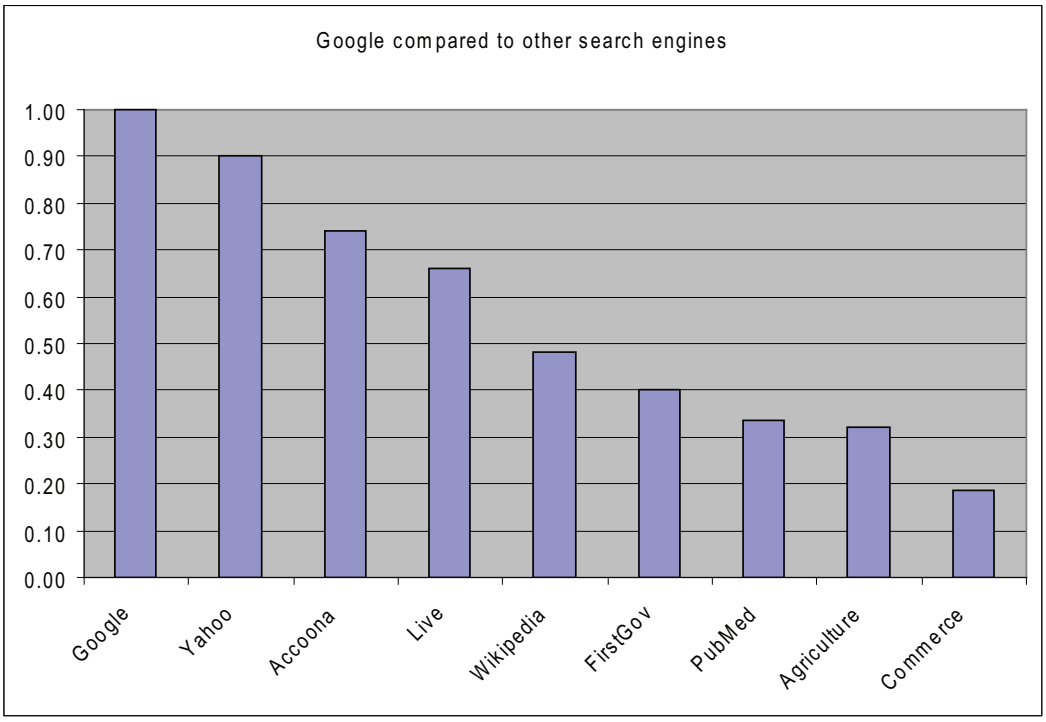


Figure 1.4: Similarity measure of Google to other search engines

search engine was compared to Google, revealing that Yahoo and Accoona were most similar to Google while the U.S. Department of Commerce was the most different. This result can be seen in Figure 1.4 which shows a comparison of Google to the other search engines.

1.6 Thesis Outline

The rest of this thesis is organised as follows. Chapter 2 gives an introduction to the concepts used in this thesis. Chapter 3 gives a review of the literature related to the work presented in this thesis. Chapter 4 details the ontology mining process used by the search engine content analysis method. Chapter 5 gives a detailed description of the resource discovery process used to find the subject matter of large search engines. Chapter 5 presents objectives and algorithms used to analyse search engines, and gives a detailed description of a pattern matching method used to group related search engines by subject matter. Chapter 7 shows the experimental evaluation of the analysis method, and shows the results of the search engine analysis. Chapter 8 concludes the thesis.

Chapter 2

Definitions and Background

This chapter defines the terms and concepts used throughout the rest of this thesis and gives a background of the research.

2.1 World Wide Web

The World Wide Web can be divided into two major areas, the *surface web* and the *deep web*.

2.1.1 Surface Web

The surface web consists of a massive interlinked collection of electronic information. It is often visualised as a large graph similar to a spider's web. *Hyperlinks* give the web its highly inter-connected web structure. Hyperlinks are pointers to other documents and collections on the web. Each HTML web page typically has outgoing hyperlinks to other HTML pages, and incoming hyperlinks from other HTML pages, creating a spider-web like system of interconnected data. The information from these maps of hyperlinks may also be mined, and in some systems each hyperlink is treated as a measure of popularity, and is used for off-the-page ranking of a document [BP98]. Often hypertext documents point to other hypertext documents. The surface web is highly distributed. There exist hubs and authorities [Kle99] which identify central, respected documents. The surface web can be indexed using computer programs that recursively follow these hyperlinks. These programs, known as *spiders*, retrieve information

from the surface web. This information can then be processed into a format which allows fast and efficient information retrieval.

2.1.2 Deep Web

Many people think that a large search engine like Google covers most of the information available on the internet. However, a comparison of search engines such as PubMed [1] or IEEE Explore [2] with a popular search engine reveals that most of the collection contents are not in the Google results. There are many collections on the internet that contain a rich sources of information [Ber01, CHLZ03] that are not available to traditional search engines. These collections are collectively called the *deep web*. Examples of deep web collections are phone directories, subject directories, patent collections, news articles, and holiday booking interfaces. Why is this rich and authoritative information not available to search engines? The simple answer is that there are generally no hyperlinks directly pointing to the information contained in these collections. The only practical way to access and evaluate the information contained in these collections is to enter a query into a search box on the collection's HTML interface, and then to browse the results.

In 2003 it was estimated that there were over 450,000 deep web collections [CHLZ03], containing over 550 billion documents [HC02, Ber01] whereas the largest search engine index of the surface web contained less than 9 billion documents. The sixty largest deep web sites cumulatively contained 750 terabytes of data, while the surface web cumulatively contained only 19 terabytes [Ber01]. Search engine crawlers (or spiders) cannot find these deep web pages. Deep web collections most commonly occur as front ends to databases such as Oracle, mySQL, SQL Server, or as structured documents such as XML or SGML. These collections often contain good quality, structured, well-maintained data, but as there are no hyperlinks pointing to them, they are inaccessible to search engines like Google. Deep web collections often have differing query interfaces, schema, and constraints. Some of these collections may stay static, while other collections may change frequently. The quality of deep web content is

[1] http://www.ncbi.nlm.nih.gov/entrez/query.fcgi
[2] http://ieeexplore.ieee.org/Xplore/login.jsp?url=/search/advsearch.jsp

also much higher than surface web content in the sense that the information is more precise and reliable. As stated above, there are hundreds of thousands of these deep web collections available, and they cover a vast number of subjects. This makes it difficult for a novice user to find the right collection to use for their information need. This thesis presents an answer to this problem, presenting an automatic method for selecting the best collection or set of collections for an information need.

Many deep web collections contain highly subject-specific data, leading to high-quality clusters of information. There are no hyperlinks, hubs or authorities. Traditional Information Retrieval methods cannot be used on the deep web. Searching the deep web requires new methods. Often the only way to access deep web information is to use a search query interface. A query is sent to each interface and an ordered set of documents is returned. The search interface will also generally hide all but the first few hundred results from the user. It is difficult to find detailed term usage statistics about deep web collections.

French [FPC+99] claimed that in the case of no direct access to a database, it is in principle possible to build an index by issuing a single-term query for each word. This method is only useful when a very small number of results are returned. In addition, this method would be very difficult to use when working with queries of two or more terms.

2.1.3 Deep Web Collection Selection

Deep web collection selection is significant because deep web collections often contain high quality, structured information not accessible to modern search engines. The need for deep web collection selection will keep growing as more and more collections are added to the deep web.

A problem with deep web collection selection is the difficulty in estimating the size of such collections and most do not allow direct access to their information. This also means that standard information retrieval methods and metrics cannot be used with deep web collections.

Deep web collection selection is typically performed using a *broker* or a *meta search engine*, which queries a number of search engines and combines the results into a single result set

ordered by relevancy. Brokers sometimes use collection selection to improve results and cut down bandwidth usage.

There are a number of problems facing efficient deep web collection selection. The first problem is that there are too many electronic collections available to track by hand. The massive growth of the deep web is parallel to the growth in electronic collections becoming available on the deep web. This is dynamic data, with constant growth and change. The second problem is that many deep web collections are not indexable by traditional methods. Only surface web documents have a hyperlinking system that supports easy indexing. The third problem is that the collection selection system needs to be able to respond quickly to user queries. This is important as collection selection is only the first part of the distributed information retrieval process. Based on their experience of querying the surface web using search engines like Google, users would expect near instantaneous responses to their queries. Finally very short queries are common in deep web systems. This makes it difficult to make exact matches to documents and can produce poor results.

Considering these problems, the aim for this work is to build a system that can automatically, efficiently and accurately profile a large number of deep web collections with no dependance on human intervention at any stage of the process.

2.1.4 Search Engines

Due to the massive growth of the world wide web, applications called “search engines” have emerged as a fast and efficient method of finding electronic information. Information on almost any subject imaginable can be rapidly found. And as information and knowledge continues to grow, search engines will become increasingly important to our daily life.

Most large search engines are based on an “inverted index” model. This is similar to a concordance, where all information is broken into terms and stored in a sorted index of terms called the “index”. This enables searches to be quickly and efficiently performed. For each term in each document, statistical information is stored about it, such as the number of times it occurred in the document and where it was placed in relation to other terms. Information

lookup is performed by taking a query, breaking it into terms, and then doing a search of the index for the query terms. Once the terms have been found in the index, the search engine then runs a series of algorithms to select the most relevant document based on the query terms. Words which occur frequently, *stopwords*, are discarded. A stopword is any word which has no semantic content. Common stopwords are prepositions and articles, as well as high frequency words that do not help retrieval. These words can be removed from the internal model of the query, document, or collection without causing loss of precision and recall.

There are two main types of search engines. The *traditional search engine* is simply an HTML interface to some sort of database or collection. An example might be an search engine with a HTML interface which connects to an Oracle database. The *spidering web search engine* is an interface to a database where the contents are actively updated using information from the web. An example of this kind of search engine is Google, which also the most popular spidering search engine in use today.

2.1.5 Spidering Web Search Engine

Spidering web search engines are comprised of an *spider*, an *indexer* and an *retrieval engine*. The spider is a program that is used to gather information from the world wide web. The spider follows hyperlinks across the web collecting information from HTML web pages. Search engines such as Google [3] start from a small set of pages on the World Wide Web and recursively follow all the hyperlinks to other pages. This makes it possible to reach most of the surface web in a relatively short time. Some of the major search engines attempt to index the entire surface web into a centralised index. [4] Each major search engine has a different implementation of a spider, each with a different search strategy. For example, many spiders refuse to collect information from pages generated by database enabled scripting languages such as PHP and ASP, leaving massive amounts of high quality data unindexed. Some spiders limit the depth to which they crawl a website, sometimes only collecting the index page of the site, other times only

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

[4] this index is also often distributed, but that is over far fewer machines for the purpose of faster search and load sharing

following to a depth of two hyperlink levels within a website. Other spiders index document formats other than HTML, such as PDF and Microsoft Word files, and image search engines are also becoming increasingly prevalent. The frequency of spidering also varies widely. Some spiders return daily, others return yearly. Spiders often refuse to visit area of the web known as "bad neighborhoods", because they contain material known to be harmful or offensive. Because of these different implementations of spiders, the content collected by the spiders can vary widely from search engine to search engine.

The *indexer* takes the web pages collected by the spiders and parses them into a highly efficient index. In the index the terms are given an importance weighting by the search engine's ranking algorithm. Each major search engine has its own proprietary weighting methods and algorithms [PBMW98, Kle99]. In fact most of the major search engines are secretive about their weighting methods and algorithms because they represent significant intellectual and financial investment. Search engines may exclude documents that they consider irrelevant or of little value. The indexer may delete certain materials such as advertising, offensive material, or websites which have attempted to manipulate their rankings, from the search engine's index. These materials are known as "spam". Because of the large profits possible from having a good ranking in a major search engine, "search engine optimisation" has become a lucrative industry [HMS02]. Search engine optimisation involves manipulating a search engine into ranking a website higher in the search engine results page than other related websites. Generally the higher a website appears on the search engine results page the more traffic it attracts. In certain subject areas on certain search engines having a high ranking can lead to a large payoff. Some engines[5] also use a pay-for-inclusion model where a web page is guaranteed inclusion only after payment is made, and may accept payment to display pages higher in the results. Some search engines also apply "filters", which use pattern matching to improve the quality of their indexes. It can also be argued that some types of information are harder to find in some search engines when compared to other search engines. For example, some search engines give precedence to

[5] such as Altavista

commercial information[6], while other search engines have an academic focus[7].

Finally, once the data has been indexed, the *retrieval engine* returns results of a query to the user. Because of these different term weighting and document selection methods, bias is introduced in the search engines (even though the bias may simply be against unwanted advertising materials). Different search engines also have different ranking algorithms and apply run-time filters to their results. The quality of the search engine's algorithms also make a large difference to the order of which information is returned from a search. All these things combine to make the quality and quantity of information returned by a search engine different to other search engines(See [Dog07] for a recent comparison of search engine overlap).

2.2 Query Sampling

Search engine collection selection is significantly different to traditional Distributed Information Retrieval collection selection. With traditional collection selection, the search broker has full access to collection statistics and is able to communicate with the collection. The communications can include term statistics, message passing and sometimes there is software that must run on both search broker and collection for the collection selection to work. The information is largely static, and indices are often built to make retrieval faster. Queries are often large and highly specific. After searches are performed, full access is given to the results. There are standard, widely used metrics to measure how well a search performed.

With search engines, this kind of cooperation rarely exists. There are no term statistics available to the search broker. The information is often dynamic. There is no communication between engine and the search broker. Query sizes are often limited. After searches are performed, limited access is given to the results, sometimes with only the first few pages or results available. Precision and recall metrics are useless [MB00] because there are no statistics about the size and distribution of the data in the collections. Search engine selection requires the use of several unusual web intelligence approaches. New web intelligence methods that work-around

[6]See LookSmart: http://search.looksmart.com/
[7]See CiteSeer: http://citeseer.ist.psu.edu/

the lack of cooperation must be created. This chapter describes a method which allows access to the contents of a search engine without direct access to its index.

2.2.1 Query Sampling Solution

To solve some of the above problems, collection selection researchers use *query sampling*, which involves sending query terms to a search engine and analysing the results in order to find the best search engine for an information need [CCD99, CC01, Fuh99, IGS01c]. While some collection selection techniques require the implementation of cooperative interfaces between the search broker and the server, query samples do not require special protocols between broker and server. The two main types of query sampling are *Query Based Sampling* [CC01] and *Hierarchal Sampling* [IG02][8]. Query Based Sampling involves sending random terms to a collection until it returns a document. The terms from the found documents are then used for further queries of the search engine until a threshold number of documents is reached. In Hierarchal Sampling, a set of queries for the top level of the hierarchy are generated and sent to the collection. This continues down the hierarchy until a threshold limit number of returned documents is reached. A content summary for the engine and the place of the engine in the hierarchy is returned.

A modified query based sampling method is used in this research; and the results are converted into a hierarchal format. In this method, query sampling terms are carefully selected from an ontology and sent to each collection. The terms from the retrieved documents are not reused. As an example of query sampling terms mined from the ontology for the subject *004 Data processing Computer science* are “ccna”, “ipv6”,“ethernet”, and “dreamweaver”. These are the terms that tend to cluster around the Dewey Decimal Number 004. Note that few of these terms would occur in a common dictionary, yet they are excellent for using as query samples to pinpoint that the content in the collection is related to the subject *004 Data processing and computer science*. Classifying a collection requires that query terms are carefully selected which makes this method a form of sampling.

[8] Otherwise known as *Focused Probing*

In this research the query sampling method was chosen over the hierarchal method because with the Dewey Decimal System it is possible to classify a level and then move upwards in the hierarchy without having to regenerate query probe terms for each level. If a collection is profiled using level three of the Dewey Decimal System then to get the results of upper level it is calculated by summing the results of each subtree together. For example, once the subject nodes 301-309 are profiled then to get the value of the supernode 300 for this collection it is a simple process of getting the sum of the results from 301-309.

Manually interfacing with uncooperative search engines is often a clumsy process. In order to search a collection, an interface for communicating with the collection must be configured, followed by configuring a *sample query* of the collection, then a *screen scraper* must be trained to extract the results and then parse them into a format that can be used by the search broker. In related work an "automatic wrapper generator" was written. This generator takes ten snapshots of search engine results from ten different queries and inducts what areas change and what areas stay the same. Then a regular expression can be automatically generated to extract the relevant information from the page.

2.2.2 Lightweight Probes

Hawking and Thistlewaite [HT99] propose *Lightweight Probes*(LWP) for server selection. These probes use a minimal amount of communication between the search broker and the servers, and operate without global server information and descriptions. Lightweight Probes aim to reduce probe processing and reduce costs while still gathering enough information to give good results.

Probes for server S_i with terms t_1 and t_2 are as follows:

$$S_i = c_1 f'_1 + c_2 f'_2 + c_3 f'_{cooccur} + c_4 f'_{prox} \tag{2.1}$$

The primes indicate that a normalised frequency was used. The best results obtained under training for $c_1..c_4$ were $c_4 = 100, c_3 = 10, 0 \leq c_2 \leq 1$, and $0 \leq c_1 \leq 1$.

The servers are sorted in order of S_i, where

f_i the number of documents containing each individual term t_i (i=1 or 2),

$f_{cooccur}$ the number of documents in which a specified number of the terms occur near each other,

f_{prox} the number of documents containing a specified number of the terms within a specified proximity of each other, which often indicates a relationship between the terms.

Lightweight Probes are a two term subset of the user query that is sent to all collections for all queries. Statistics are retrieved using a special communication protocol between broker and server.

2.2.3 Incremental Probe Queries

Craswell [Cra01] and Callan et al [CCD99] use *incremental probe queries* for server selection. The search broker periodically broadcasts a query to all the servers. From the returned documents, the search broker can apply server ranking methods and select the best servers. Multi-term probe queries are taken from a query log, rather than on the fly, and are not run at query time.

Probe Queries are periodically sent to each server in batches. While some collection selection techniques require the implementation of cooperative interfaces between the search broker and the server, probe queries do not require special protocols between broker and server.

The previous literature references show the past work on query probing for the purpose of collection selection. This current work makes use of and extended this research for the purpose of search engine analysis.

2.3 Information Retrieval Definitions

Information Retrieval(IR) is a widely studied discipline traversing artificial intelligence, data mining, natural language processing, user profiling, and many other fields.

The atomic unit of Information Retrieval is the *term*. A term is a word and it can be of any language. A *query* is a term or set of terms entered by the user which provides clues to the user's information need. A *document* is generally a larger set of ordered terms, usually in

sentence format. A *set of documents* is often called a *collection*. The count of how many times a term appears in a document is known as *term frequency*.

In information retrieval, a document is often represented as a matrix of terms and their statistics. A *document vector* contains the term frequency of every word in the document. To compare short and long document vectors with each other, we often have to *weight* the term frequencies of each document vector. Weighting is also applied to increase the value of more descriptive terms. Previously one of the most popular term weighting methods was $TF \times IDF$ where *TF* stands for Term Frequency and it is the number of times that a term occurs in a document. *IDF* stands for Inverse Document Frequency, which is a measure of how common a word is in a collection. Common words will have a low *IDF* and rare words will have a high *IDF*. *TF* is multiplied by *IDF* for each term in the document vector. The method for calculating *IDF* and then *TFIDF* is given as:

$$IDF = log \frac{collection\ size}{number\ of\ documents\ containing\ the\ term} \tag{2.2}$$

$$TFIDF = TF \times IDF \tag{2.3}$$

The goal of Information Retrieval is to take a query and return a set of documents that are most *relevant* to the query. *Relevance* is a measure of how well a document fits a user need. Relevance is difficult to quantify because different users will often have different information needs, and what might be relevant to one user may be irrelevant to another. Some relevance judgements use a boolean value, so that a score of 0 is given to a irrelevant document, and a score of 1 is given to a relevant document. Other relevance judgements use a scale, for example assigning each document a floating point relevance score between zero and one.

There are a number of ways for measuring performance. Four standard Information Retrieval (IR) measures of performance are *precision*, *recall*, *overload* and *mismatch*.

Precision is a standard IR measure of performance. It is defined as the number of relevant documents retrieved divided by the total number of documents retrieved:

$$Precision = \frac{Number\ of\ relevant\ documents\ retrieved}{Total\ number\ of\ documents\ retrieved} \tag{2.4}$$

The goal of a IR system is to achieve 100% precision, however as this can be achieved by returning only one document the system should also try to maximise *recall* (see below). Giving more weight to common documents is one method of improving precision [NK98].

Recall is a standard IR measure of performance. It is defined as the number of relevant documents retrieved divided by the total number of relevant documents in the collection:

$$Recall = \frac{Number\ of\ relevant\ documents\ retrieved}{Total\ number\ of\ relevant\ documents\ in\ the\ collection} \tag{2.5}$$

The goal of a IR system is to achieve 100% recall. However as this can be achieved by returning all the documents, the system should also try to maximise *precision* as well. Recall can be improved by selecting collections that retrieve different relevant documents [NK98].

Overload is where returned documents are not relevant. Overload is defined as the number of irrelevant documents retrieved divided by the total number of irrelevant documents in the collection:

$$Overload = \frac{Number\ of\ irrelevant\ documents\ retrieved}{Total\ number\ of\ irrelevant\ documents\ in\ the\ collection} \tag{2.6}$$

Overload is made worse if the irrelevant documents returned are highly rated.

Mismatch is where relevant documents are not retrieved from the collection. Mismatch is defined as the number of relevant documents not retrieved divided by the total number of relevant documents:

$$Mismatch = \frac{Number\ of\ relevant\ documents\ not\ retrieved}{Total\ number\ of\ relevant\ documents\ in\ the\ collection} \tag{2.7}$$

Mismatch is difficult to avoid when working with short or unspecific queries.

2.3.1 Subject Based Web Intelligence

While many information retrieval systems use terms to describe documents and search engines, the ontology-based collection selection method presented here uses subjects to describe collections and search engines. The power of a subject based approach is better understood through the following example. If a user types "matrix factorisation methods" into a search engine, they would expect "singular value decomposition" to be returned as a result. Both phrases belong to the same subject, yet there is no overlap of terms. By identifying the subject matter instead of using terms it is possible to return items where the terms do not overlap yet they are still highly relevant. This research argues that it is more efficient to classify collections with a small number of subject classification codes, rather than a large number of term statistics. It is also easier to group related collections together based on subjects rather than terms. The result is easier for humans to understand, and it is more robust in the face of technical and social change. This research attempts to exploit this capability in the following way: an arbitrary query is mapped into the ontology, yielding a set of subjects, then the subject classification terms of each subject are accumulated into a query which is used to rank collections.

A subject can be difficult to define and abstract to some degree, while still containing a strong taxonomic structure. A subject may have sub and super subjects, with super-subjects being a higher abstraction of the subject. Many subjects can be defined by a domain vocabulary as can be easily observed if for example one compared the terms used in a movie review to the terms used in a computer science paper.

2.3.2 Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a vector space model based on singular value decomposition which analyses relationships between documents. Developed by Deerwester [DDL+90] and Berry [BDJ99], it is a statistical method of reducing the dimensionality of a matrix and associating related concepts. Used with a term-document matrix, Latent Semantic Analysis takes the background structure of word usage and removes the noise. This noise reduction allows the higher order relationship between terms and documents to be clearly seen [DDL+90]. It

considers documents which have many terms in common as being closer to each other than documents which have few words in common. This helps to solve two major information retrieval problems: *polysemy* and *synonymy* [DDL+90]. *Polysemy* is where a word has multiple meanings. An example of polysemy is that a student may be looking for information about a "jaguar" car, and so enters the term "jaguar". However this term can also mean jaguar "cat", and the Mac OS X "Jaguar" operating system. In computational linguistics polysemy is called *word sense disambiguation*. Term matching, while a partial solution to this problem, often returns irrelevant documents, reducing precision. *Synonymy* is multiple words having the same meaning. For example the term "cat" can be also referred to as "feline", "lion", "kitten", and so on. Synonymy can be partially solved by human generated thesaurus, however this causes the problem of inconsistent human judgments occurring in human generated indexes. Based on context, the human reader has little problem inferring meaning about these words, however it is difficult for machines to infer meaning about these words. Latent Semantic Analysis helps with these problems; in fact the term itself does not have to occur within the collection for Latent Semantic Analysis to find that the collection is relevant. This is a great feature, with off-the-page rankings being so popular with search engines at present. With Latent Semantic Analysis it is possible to fingerprint each collection and measure a sample of each collection against a query.

Chen et al [CMC01] showed that Latent Semantic Analysis consistently improves both recall and precision. The Latent Semantic Analysis method has equalled or outperformed standard vector retrieval methods in almost every case, and gives up to 30% better recall [Din99a].

A popular use of LSA is to analyse relationships between documents - however this research uses it to analyse relationships between collections. LSA is very suitable for collection selection because of its ability to match related terms and concepts.

Dimensionality reduction is another important part of Latent Semantic Analysis, allowing dimensions to be reduced while still keeping relative distances. This reduction pulls together related documents and terms, while removing the background noise.

Ding [Din99b] observes that there is no rule for selecting the number of dimensions to represent in Latent Semantic space. This means that the method can be used on different languages

and text formats with no loss of recall.

2.4 Distributed Information Retrieval

This work builds on previous research in distributed systems, server selection and information retrieval. From the field of distributed systems comes the necessity of dealing with different text formats and languages. From the field of server selection comes the issue of having to be efficient in the selection of the collection servers in order to reduce costs. From the field of information retrieval comes the issue of consistently delivering relevant information to the user.

There are two forms of Information Retrieval: Localised Information Retrieval and Distributed Information Retrieval. In *Localised Information Retrieval*, all documents reside in a central location. In Distributed Information Retrieval, the documents are distributed across multiple servers and the servers co-operate in order to return as many relevant documents as possible. There are several levels of cooperation, ranging from none to full. The levels of co-operation will determine how much information gets passed between the search broker and the server, and will determine the quality of results returned. *Distributed Information Retrieval* is the searching of multiple servers and the merging of the results. There are three components in a Distributed Information Retrieval system: *Collection Description*, *Collection Selection* and *Collection Fusion*. In collection description a profile of each collection is created. This may include statistical and other metadata. In collection selection, a *search broker* acts as an interface between the user and the collections. The search broker takes a query and sends it to the collections. The best collection is then selected. In collection fusion, the resulting documents are merged into a single ordered list of documents and returned to the user.

Distributed Information Retrieval is the querying, ranking and returning of data from a set of information brokers located on different servers. Distributed information retrieval draws on two major fields: information retrieval systems and distributed systems [Cra00]. Information retrieval systems perform the task of returning relevant results to a query. Distributed systems consist of a collection of autonomous yet connected computers which cooperate to perform a task. For this research there will be some collections distributed across various servers (Dis-

tributed Infrastructure Component), and each collection will support an Information Retrieval Architecture (Information Retrieval Component) that returns an ordered set of documents for each query to a Search Broker. The aim of Distributed Information Retrieval is to combine the results from querying multiple collections into a result such that it appears to be from a single collection.

It is important to make the distinction between a data retrieval system and an information retrieval system. A data retrieval system differs from an information retrieval system in the acceptable level of incorrect documents allowed in the set of retrieved items. A data retrieval system is defined on mathematical principles using structured data and thus no incorrect documents are allowed in the set of retrieved items [BYRN99]. An information retrieval system deals with unstructured data with ambiguous natural language information with possibly many meanings and is thus allowed a medium level of incorrect documents in the set of retrieved items.

The Distributed Information Retrieval process is as follows. A query is taken from the user, the collections relevant to the query are selected using *collection selection*, the number of documents to take from each collection is calculated using *document selection*, the documents taken from each selected collection are sorted using *collection fusion*, and the results are returned to the user as a set of documents ordered by relevance to the query. This thesis will only discuss Collection Selection, leaving Collection Fusion for future work.

Sometimes distributed collections are *overlapping*, which occurs when different collections contain some of the same information. If the collections are not overlapping, they are referred to as *disjoint*.

In a recent paper, Tsikrika et al [TL01] separated the Distributed Information Retrieval process into three stages, including *Document Selection* as a new step between collection selection and collection fusion. *Document selection* is the selection of a set of documents from the results of each collection. This can be as simple as selecting an arbitrary number of documents from each collection, or alternatively selecting documents based on a relevancy score. Intuitively, the more relevant the collection is, the larger the number of documents that should be taken from it.

2.4.1 Collection Selection

Alongside the massive growth of the internet has come the massive growth of the number and size of electronic collections of information available on the internet. However when there are hundreds of thousands of collections available it difficult to select the best one to use. Sending a query to each collection is too expensive and time consuming. This is when collection selection becomes important. *Collection selection* is the selection of an optimal subset of collections from a large set of collections for the purpose of reducing costs associated with Distributed Information Retrieval [CLC95, FPC+99, HT99, LCC96, CBH00, DTZ00, GGMT99, MLY+99, GGM95, CPFC00a]. Collection Selection aims to be efficient with respect to bandwidth and computation, and to decrease both resource usage and time taken to return a set of results for a query. Well-planned collection selection can have a large influence on the efficiency of a query, Lu et al [LM00] and Choi et al [CY99] state that there is a high correlation between the search performance and the distribution of queries among collections. Collection selection is becoming increasingly important as the number of electronic collections grows daily. The central goal of collection selection is to make searching multiple collections appear as seamless as searching a single collection [VGJL95], and to reduce the number of overall search requests needed. By searching a small, high quality, relevant subset of the available collections savings can be made in time, bandwidth, and computation. Collection selection should not reduce information retrieval effectiveness when compared to centralised search.

A common problem with traditional collection selection techniques is that they require communication between the search broker and collections, or that they are limited by the number of subjects that they are able to cover. This research introduces a method which does not need communication between the search broker and collection, and can cover a large range of subjects. Other common problems encountered during collection selection include reducing expenses, increasing search speed, learning to adapt to change in the search environment, and increasing precision and recall.

There are two main types of collection selection in an uncooperative environment. The

first query samples the database at runtime. The second relies on pre-generated metadata [9] about each collection to choose the best collection. In previous research [Kin03, KL03] the author used query sampling at runtime. This research uses an ontology to help create a resource description of each collection which is then stored as metadata for each collection.

As an example of collection selection, three web collections and a query are taken from the user and collection selection is performed. PubMed has been previously classified into Medicine, IEEE Explore into Computing, and the CIA World Factbook into Geography and Economics. The user then enters the query term "algorithms". This query is then mapped to a subject set. Because the term "algorithms" commonly occurs in Computing and rarely in other subjects, the query is mapped into the Computing subject. IEEE Explore is then returned as a suitable search engine because it also belongs to the Computing subject.

Current solutions to the collection selection problem are Distributed Information Retrieval and Web Intelligence. Distributed Information Retrieval is concerned with search across many distributed servers and requires complete knowledge of the collections being searched. The Web Intelligence commonly uses data mining techniques to find high quality information [HC02].

In collection selection, a *broker* is the intermediator between the user and the collections in Distributed Information Retrieval. The broker takes the query from the user, parses it, passes it to the collections in parallel, takes the results from the collections, parses them into a suitable format, and returns the results to the user. Other terms for brokers are *meta search engine*, *search broker*, or *search agent*. In some research, it is common for large amounts of information and statistics to be passed between the broker and the collections.

Figure 2.1 shows the collection selection and fusion process. For a set of collections a query is taken from the user and classified into a subject code. The best collections for the subject code are then selected. The documents are then extracted from each selected collection and fused together in order of relevance. The ordered set of retrieved documents are returned to the user.

Collection Selection Metrics are different to the precision and recall document selection

[9] sometimes called *resource description*

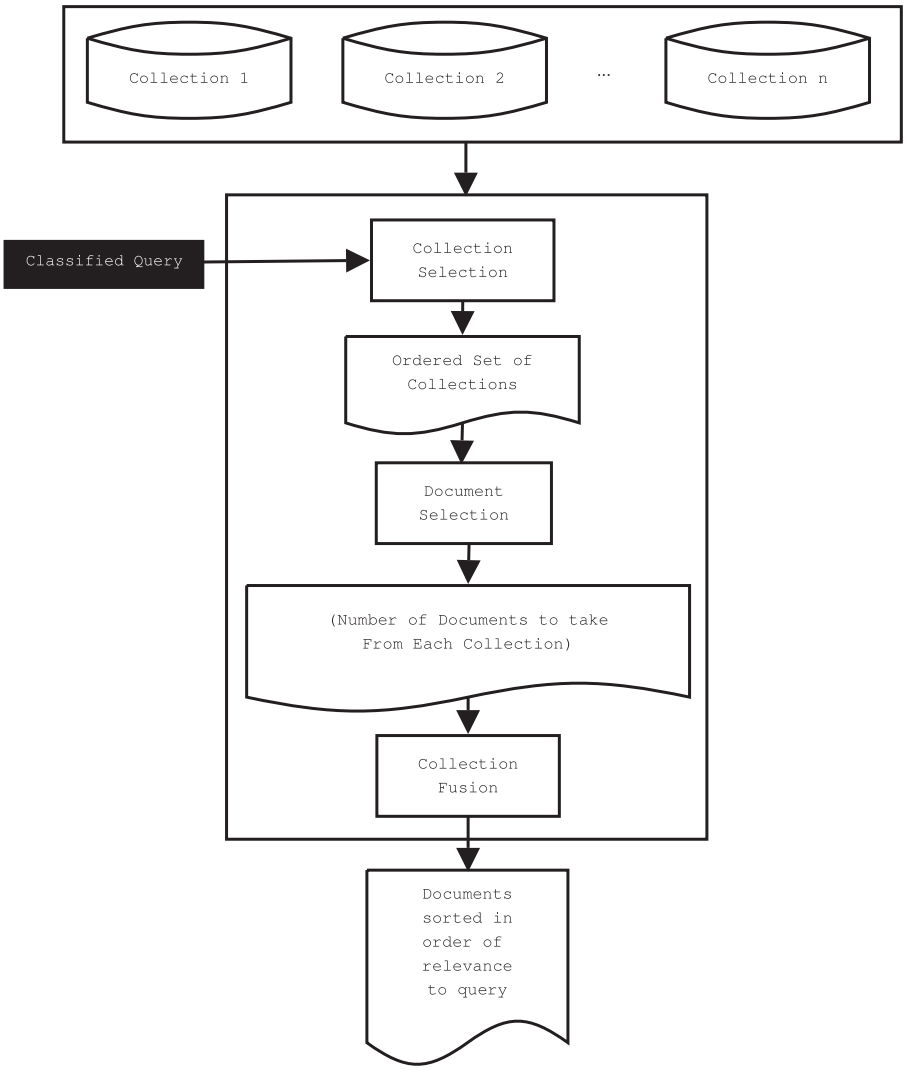


Figure 2.1: Collection Selection and Fusion

metrics presented above. Historically collection selection has used precision and recall measurements to evaluate the effectiveness of the collection selection technique. This collection selection metric set is an extension of the document precision and recall metrics, first mentioned in Gravano *et al* [GGM95] and more recently in Callans [CPFC00b] paper. They measure the percentage of relevant collections in relation to the number of collections retrieved, and compare the amount of relevance in all the collections with the relevance in the top *n* collections for a query.

The metrics are presented below.

$$Precision_n = \frac{\sum_{i=1}^{n} \begin{cases} 1 \ if \ NumRel(db_{ei})>0 \\ 0 \ otherwise \end{cases}}{n}$$

$$Recall_n = \frac{\sum_{i=1}^{n} NumRel(db_{ei})}{\sum_{i=1}^{n} NumRel(db_{bi})}$$

where

- *e* is the estimated collection ranking
- *b* is the corresponding relevance based collection ranking (*baseline*)
- *n* is the rank
- db_{ji} is the i^{th} collection of collection ranking j
- NumRel(db_{ji}) is the number of relevant documents in db_{ji}
- *db* is the collection

2.4.2 Collection Fusion

The second component of distributed information retrieval is *Collection Fusion* (or *result merging*), where the documents taken from the selected collections are ranked and fused [Li01]. A collection fusion method retrieves information about documents D_n and then generates a ranking r_n with respect to query q. Documents are returned in order of relevance to the query. On

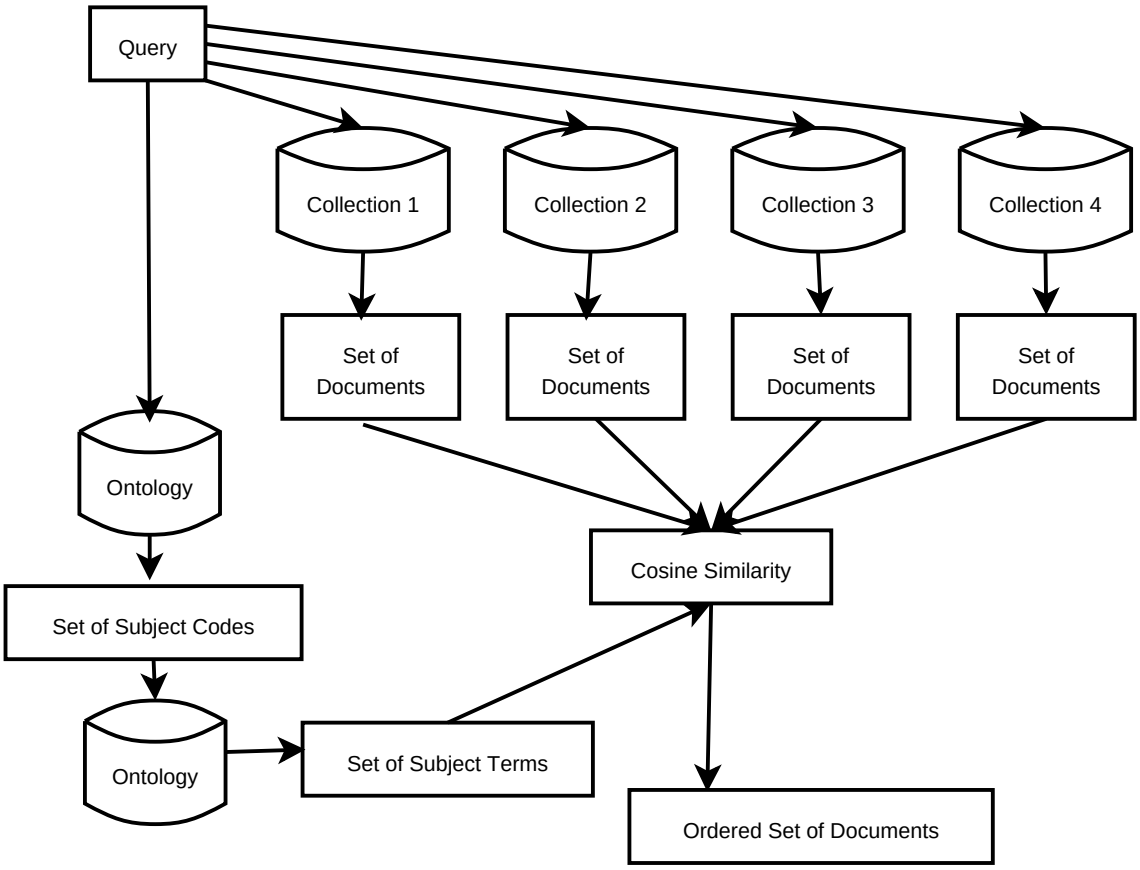


Figure 2.2: Ontology Based Collection Fusion

the web, collection fusion is most commonly used by *meta search engines*, which query a number of search engines in parallel and interleave the results. There are two types of collection fusion: cooperative collection fusion and uncooperative collection fusion. Cooperative collection fusion uses some protocol which enables message passing between the search broker and the collections. Uncooperative collection fusion uses what little information each search engine returns such as document rankings and scores. Note that search engines are uncooperative, and even the collections which are cooperative use different communication protocols. Previous collection fusion research has concentrated on normalising local and global weighting systems, or on sequentially interleaving results.

Because collection fusion relies heavily on each collection's search application it is difficult to achieve the same results that a centralised system would produce. This is because fusion takes only the top results returned from each collection and may miss relevant documents which are not in the top results from a collection.

Collection fusion is a difficult problem because different collections use different methods to index, calculate document weight, calculate query weight, and compare document similarity. Commercial search engines are understandably secretive about the methods they used for information retrieval because they represent significant development time. Releasing detailed information about them would allow unscrupulous competitors to exploit them for commercial gain. It requires some skill to merge the documents together when dealing with such different collection ranking methods. In addition, most search engines do not return the document weights in the results, which makes merging the results of different queries on different search engines a difficult task. Figure 2.2 shows the collection fusion process.

Collection fusion is important for Distributed Information Retrieval because merging must take place in such a way that there is an increase in retrieval quality when compared to centralised information retrieval.

2.4.3 Collection Size Estimation

Because larger collections tend to have a greater term distributions (Zipf's law) many collection selection methods tend to treat larger collections as being more relevant than smaller collections, even though smaller collections may actually have more relevant information relative to their size. In traditional cooperative collection selection, collection size statistics are used to reduce the bias that most collection selection methods have towards larger documents. However uncooperative collections do not publish details of their total size or the total number of documents that they contain. Collection size estimation is a recent addition to the field of collection selection in uncooperative environments [SC03]. It attempts to make an estimation of the size of each collection, then uses this estimate to normalize the results from the collection selection process. Collection size estimation can be difficult and error prone. For example, imagine trying to guess the size of a large search engine such as Google, which has many billions of documents without having access to any detailed statistical information about it. This is very difficult and expensive, and can involve the downloading, processing and analysis of many documents.

In recent work, Shokouhi *et al.* [SZST06] presented probabilistic methods for collection size estimation that were drawn from a technique of animal population estimation used by ecologists where animals are captured and recaptured and the recapture rate is used to guess the population size. Various methods for analysis of the resulting data exist, and the quality of the results is effected by the analysis method used. The experiments showed that the methods were more accurate and efficient than previous size estimation methods.

The ontology-based collection selection method does not use collection size estimation, yet it still performs better overall than methods that uses collection size estimation. With this research gives a baseline as to how well the method performed in its simplest configuration, without any sophisticated pre and post-processing of the results. Collection size estimates can easily be used to enhance the method, this is left to later work.

2.4.4 Coverage and Specificity

The Dewey Decimal subject hierarchy contains many layers. The top layer contains ten subjects, the second layer contains one hundred subjects, and the third level contains one thousand subjects. This pattern continues downwards. Note that not all subject nodes are used, especially below the third level of the hierarchy.

Coverage and specificity are measures commonly used in collection selection. For example, consider the category *630 Agriculture*. It contains the subjects:

- 631 Techniques, equipment, materials
- 632 Plant injuries, diseases,pests
- 633 Field & plantation crops
- 634 Orchards, fruits, forestry
- 635 Garden crops (Horticulture)
- 636 Animal husbandry
- 637 Processing dairy & related products
- 638 Insect culture
- 639 Hunting, fishing, conservation

Collection A may contain a large amount of general agriculture information, and will perform well on the second level of the ontology. Collection B may contain little general agriculture information and be much smaller than Collection A, yet perform well on subject 634 Orchards, fruits, forestry on the third level of the ontology. Searchers who want general information on agriculture would choose Collection A which has more coverage. Searchers who only want information about bee keeping would prefer Collection B which has more specificity.

A collection is denoted as C, the Dewey Decimal taxonomy as DD, and a subject $DD_i \in DD$. The coverage of C for DD_i is the number of documents in C for the subject DD_i.

$Coverage(C, DD_i)$ = the count of the documents in C for the subject DD_i

$Coverage(C, DD_i)$ is the total amount of information that collection C has about the subject DD_i

The specificity of C for DD_i is the number of relevant documents in C divided by the number of the documents in C.

$Specificity(C, DD_i) = \frac{Coverage(C,DD_i)}{|C|}$

The $Specificity(C, DD_i)$ measures the relative number relevant documents in the collection.

For a search engine, the coverage is the sum of the number of documents returned for each term. SE denotes a search engine.

$Coverage(SE, DD_i) = \sum(t_1..t_n, DD_i)$

This equation means that the coverage of a search engine for a subject is the sum of the terms for each subject.

2.5 Ontologies

An ontology can be defined as a shared conceptualisation of knowledge [DFvH+00]. Ontologies are used across a number of domains [vHSW97, McG98, UG96]. Ontologies often contain a model of a domain, its taxonomy the relationships between its entities. Some ontologies are detailed enough to allow some level of reasoning. Ontologies are usually built by domain experts, who have a high degree of knowledge about their specific domain. Motta [MS06] suggests that one way to overcome knowledge sparseness is to enrich dictionaries (such as WordNet) and thesauruses. This research has taken the opposite path, taking a large training set and reducing it into a subject based classification system. The aim of this research was to quickly, cheaply and simply build an ontology which has both a wide range of knowledge and capabilities across many different domains.

Examples of large ontologies are Cyc [Len95], ConceptNet [LD04], and ThoughtTreasure [Mue8a]. These contain a large number of rules which are intended to allow some form of reasoning.

The term "ontology" has a number of conceptions. For the purposes of this research, an ontology is defined to be a hierarchical structure, whereby the nodes correspond to subjects. This study makes use of world knowledge stored in an ontology and applies it to collection selection.

Two strengths of using an ontology in Information Retrieval (IR) are listed in [GVCC98, Mar82]. Firstly, an Information Retrieval system using an ontology may improve its precision performance. The concepts embodied in the given query and documents can be identified and extracted along with their relationships from the ontology, which helps an IR system to restrict the search scope so that the retrieval precision can be improved. Secondly, an IR system using an ontology may improve its recall performance. The synonymous terms with the same concepts as the query terms can be extracted from the ontology. The IR system can use the synonymous terms to search instead of the original terms in the given query. As a result, the IR system does not have to rely on the exact terms occurred in both of the query and documents. Although some documents do not contain the query terms, as long as they discuss the same concepts as the given query, they may be identified and extracted. However, while the recall is improved by searching using the synonymous term, the precision may be hurt as the synonymous terms normally refer to a broader concept area than the original terms [Sim03]. This concept is similar to "query expansion", which has been researched for many years with little success.

Berners-Lee[BLHL01] considers that the next major step in the evolution of the World Wide Web will be the adoption of ontology-driven technology, resulting in what is termed the "Semantic Web".

Ontologies will allow computers to understand and reason about the meaning of information, and software programs known as "agents" will be able to reason about the information in ontologies. Ontologies will providing the reasoning framework, allowing the sharing of knowledge and a formal definition of relationships between entities [ADC+00]. Semantic Web ontologies commonly consist of a set of inference rules and a taxonomy. The taxonomy defines relationship between entities. The taxonomy is often hierarchal, allowing "children" to inherit attributes of the "parents". The inference rules allow computers to reason about the entities in

the ontology, deducing information from a set of logical relationships. Before this happens, much development must occur to bring about the ontology based understanding of text. One emerging standard is the Web Ontology Language (OWL), which allows ontologies to be created, standardised, and shared using a XML language. It is also designed to allow a level of ontology reasoning.

Ontologies are frequently subject specific, covering only a small number of subjects. This causes a problem when working with multiple distributed ontologies because a concept may be defined and formulated in many different ways between the different ontologies, or there may be many different concepts with the same definition or formulation. The merging of overlapping ontologies is still a difficult research problem because different entities often have different meanings both in definition and use [CGL01]. Current generation semantic web tools either use one well defined domain ontology, or are ontology independent but are limited to using one ontology at a time [MS06]. This research makes use of a large centralised ontology with a standardised taxonomy in order to give a computer program access to a large amount of world knowledge created by human experts.

2.5.1 Building an Ontology

There are currently three popular ways to build an ontology, each offering a trade-off between speed and accuracy. The three methods are:

1. Domain experts populate the ontology by manually entering rules (slow but arguably accurate)

2. Generate rules from expert created and/or classified materials such as dictionaries and encyclopaedia texts

3. Generate rules from free text (fast but currently inaccurate)

In the first method, domain experts are used to populate the ontology with rules and information. However building an ontology using this method can be a slow, difficult, and expensive process. Even for experts, identifying relevant documents or classifying documents is a difficult

process that involves deep knowledge of a wide variety of background information. Another major problem associated with using domain experts to build an ontology is also the large number of human-hours required to construct it. This problem is called the "knowledge acquisition bottleneck" [BW93].

The second and third methods use a computer program to build an ontology. This is known as "ontology learning" [MS01b]. A major component of the semantic web is expected to be a high quality computer generated ontology which will allow reasoning based on free text information. A problem with the third method is that current natural language processing systems are unable to automatically extract high level knowledge from free text because of the many ambiguities in natural languages such as English.

The second method of building an ontology, generating it from expert created and classified materials, is adopted in this research as it provides the most balanced approach. This research uses a large-scale expert classified taxonomy training set of over 400,000 documents which cover almost every major subject of human endeavour. This eliminates the need to have human experts build the ontology, while still providing high quality rules. While the ontology presented here is taxonomy based, it does not contain any actual inference rules; however with further development it is expected that will be easy to generate clustering rules. It is expected that the ontology presented here will be of use for annotation of free text with subject groupings, allowing a computer program to reason about the subjects contained within the text. Also this ontology covers a large range of subjects which are, however, clearly defined and standardised, so there is no problem of merging different overlapping ontologies. The aim of this research was to use a computer program to quickly, cheaply, and simply build an ontology which has world knowledge of a large range of subjects.

Figure 2.3 gives an overview of the ontology building and mining process.

2.5.2 Classification Rules

Term usage patterns within each subject are mined and used for classification of each subject. For example, the following rule is extracted:

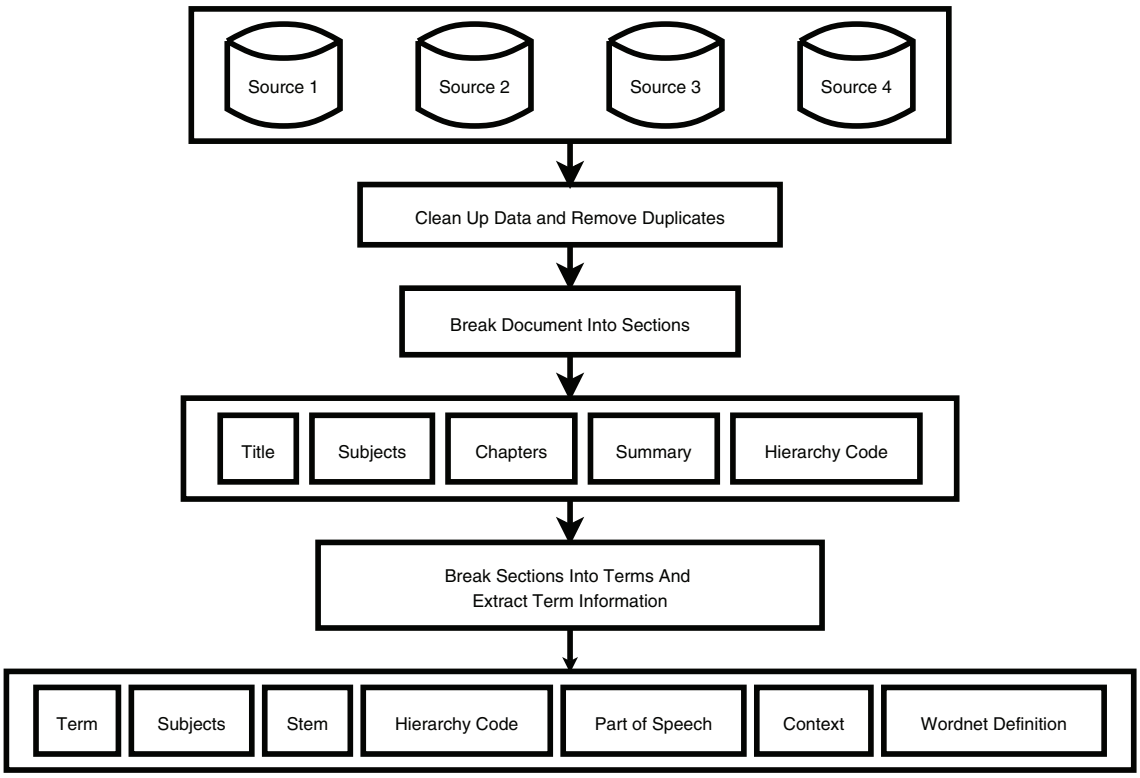


Figure 2.3: Building The Ontology From Multiple Sources

(C#,Microsoft) $\Rightarrow$ *.Net Programming Languages*

This means that if the terms "C#" and "Microsoft" occur in the same document, it probably belongs to the subject ".Net Programming Languages". This is a form of automatic classification. Terms with a strong relationship to a Dewey Decimal classification code are extracted from the ontology and used for classifying collections. This subject based term association method benefits from being easily understandable and modifiable by humans.

2.5.3 Collection Selection vs. Document Selection

Collection selection is significantly different to document selection in a number of ways. Collection selection uses different methods to document selection for scoring relevance. It uses different ways of calculating term weighting, where terms distributed across all documents in a collection are worth more than terms clustered in one document of a collection. Collection selection commonly uses partial collection sampling while document selection uses full document indexing. In collection selection it is often too expensive to download every single document

from a large collection. These differences mean that collection selection requires a significantly different approach to document selection.

2.5.4 Collection Selection vs. Search Engine Selection

Search engine selection is the selection of the best search engine or engines for a query. There are a number of differences between collection selection and search engine selection. In search engine selection it is difficult to estimate the size and term statistics of search engines because most do not provide a way of finding the frequency of terms and the number of available documents. This also means that standard information retrieval metrics cannot be used with search engines. Most search engines are not indexable by traditional indexing methods as they only present a search interface and no way of indexing the contents. Some search engines also change frequently, meaning that traditional indexing methods have to be run frequently in order to keep index information valid.

Search engine selection presents additional problems of:

- Dealing with uncooperative search engines
- Being efficient with respect to resource usage. Many search engine providers do not like having their servers being hit with a large number of automated queries or document download
- Learning to adapt to change in the search environment

Collections are usually searched in parallel. This is because some collections may have a higher response time than other collections due to network load, database access times, and other factors not under the control of the searcher. Figure 2.4 shows the collection selection process where a search broker is used. Some search engines are more effective than others, and the quality of the results returned will often vary across search engines. The methods used in this research do not depend on the quality of the documents returned from the search engines, just on the extent of the subject coverage.

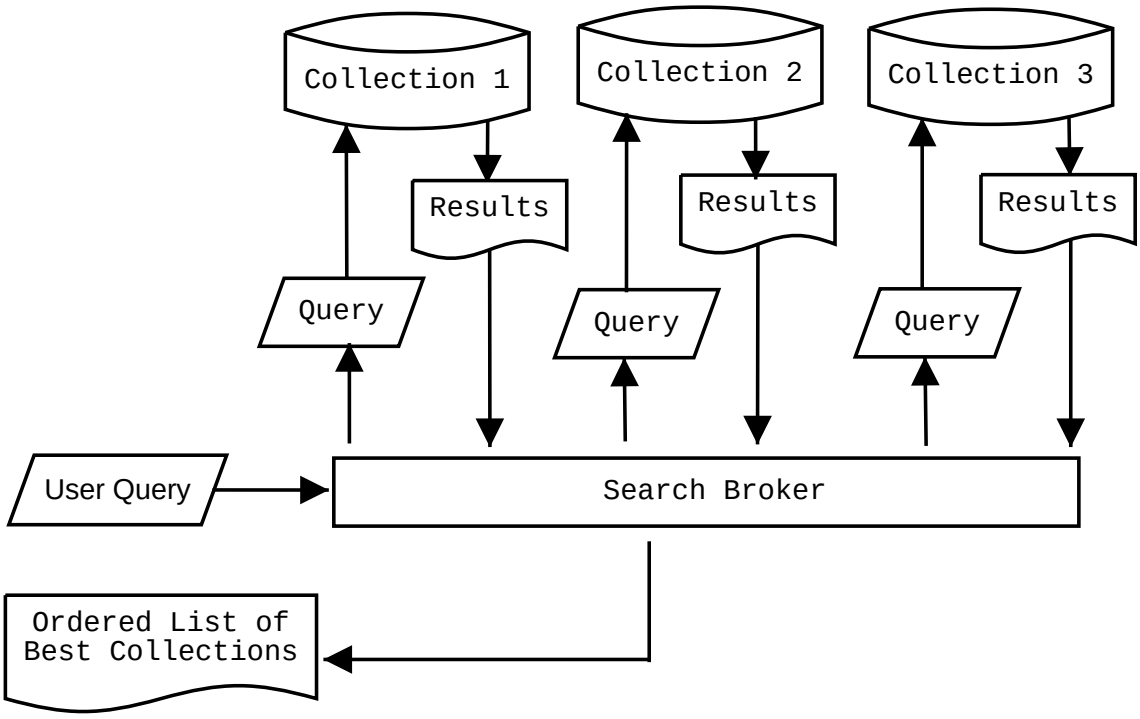


Figure 2.4: Collection Selection Using A Search Broker

An important collection selection problem is selecting which engine to search based on a user need. The ontology-based search engine analysis method tries to solve this problem by suggesting the best search engines to search based on a query. The ontology-based search engine analysis method is also able to do a profile of each search engine over hundreds of different subjects. Using this profile, a query is mapped to the best possible set of collections.

2.5.5 Collection Description - Profiling Collections

Another requirement of a collection selection system is that it must accurately summarise the contents of a collection. This summary is known as a "Collection Description" or a "Collection Profile". Being able to accurately describe the subject matter of a collection is an important feature of good collection selection system.

For instance, if the collection selection method was used for business intelligence, being able to identify the business related collections from a large list of tens of thousands of heterogenous collections will give the user the power to identify the most relevant and important information sources. This ability to judge and measure the relevance of a collection requires a large amount of background knowledge. This is the job of the *collection description* component.

Collection description is the process of describing the contents of each collection in the distributed system. Creating a full index of every term in every collection is the more accurate (and expensive) approach. However a constraint on collection description is that it should be relatively easy to obtain, and that it does not overload the servers which host the collections, which in many cases rules out creating a full index.

An ontology based collection description component is useful because a pre-existing taxonomy can be used to map a collection. This allows a multiple level view of the collection to be created, which can also be used to group collections together based on subjects.

2.6 Search Engine Content Analysis (SECA)

2.6.1 A Search Engine as a Black Box

Most search engines on the world wide web are uncooperative. They do not publish information about what they contain, and do not allow access to their indices. An uncooperative search engine can be treated as a black box with no knowledge of its contents. All that is known is that when some information is sent to the search engine, some information is returned from it.

This research makes heavy use of an idea known as ‘domain vocabulary’. This idea is based on the concept that specific domains often include specialised terms that occur frequently within the domain and rarely occur outside of it. For example terms common in a movie review might be ‘actor’, ‘theater’, ‘dolby’. Terms common in a computer science paper may be ‘algorithm’,‘C#’, ‘J2EE’.

It is possible to infer what information a search engine contains by carefully selecting what is sent to the black box, and by performing detailed analysis of what is returned from the black box. Based on the results of the analysis some knowledge is gained about the contents of the black box. There are two main questions that need to be answered in order to find information about the contents of the black box.

1. How to decide what to send to the black box? (Ontology based method)

2. How to decide what to do with the information returned from the black box? (Ontology and Singular Value Decomposition)

An ontology answers the first question. The ontology used in this thesis contains information on topics such as philosophy, psychology, religion, social sciences, language, natural sciences, mathematics, technology, the arts, literature, geography, and history. This knowledge can be used to help select the best information to send to the black box. Classification terms from a wide range of subjects are selected from the ontology. Collection selection is enhanced by using these classification terms, and automatic generation of the ontology is used to help the system scale to a very large range of subjects.

Taxonomy answers the second question. By transforming the results returned from the black box into a taxonomy, a combined high-level and a detailed view of the information contained in the black box is achieved. The taxonomy used in the ontology is hierarchal, and becomes more specific in the lower levels of the hierarchy.

In this thesis search engine content analysis is introduced as a new field. Search engine content analysis focuses on discovering what information an individual search engine contains. It has previously been too difficult to analyse the major search engines as they contain too many different subjects for smaller ontologies to be cover. Using the method presented here, it is simple and efficient to get a profile of what subject bias an engine has across many different subjects. And because the taxonomy has many levels it is possible to move up or down the hierarchy to show a meta-view or detailed view of the subjects. Search engine content analysis is useful for a number of fields such as research, advertising and search engine optimisation. Because the ontology-based collection selection method has been designed for uncooperative search engines it can also be applied to almost any information retrieval system.

For an example of search engine content analysis using the above black-box method, a set of classification terms for each of the subjects is generated from the world knowledge contained in the ontology. These terms are then sent to a search engine, the results measured and then grouped by subject.

It is important to note that this approach does not rely on estimating the collection size. Even though collection size is acknowledged as being an important feature determining collection selection effectiveness [SC03, SZST06], it is also acknowledged that acquiring reliable estimates can be a costly and challenging problem. The hypothesis behind this research is to examine whether a subject based approach may compensate for not having collection size estimates. In other words, the problem is tackled from the query side, rather than the collection side.

2.6.2 Search Engine Selection

As the number of search engines available on the internet increases, so does the difficulty of finding the right search engine for a user need. The array of search engines available often overwhelms the first time user, who will waste time hunting through pages of search engine names, followed by time reading results pages after doing an ad-hoc search. Automatic search engine selection methods try to solve this problem by suggesting the best subset of search engines to search based on a query. A central aim of search engine selection is to accurately profile each search engine. This is of importance to fields containing a large number of electronic search engines that undergo frequent change. This research aims to develop a solution to the problems of selecting the best search engines and eliminating search engines that do not contain relevant data. Once the content of each engine has been determined, the best subset of engines can be returned to serve an information need.

Search engine selection is important as the number of search engines on the internet is growing far too quickly for human experts to classify them. It is now impossible to manually track and index all search engines as they number in the hundreds of thousands. As more and more specialist search engines are becoming available electronically, the need for this research will increase. The importance of search engine selection has grown rapidly as more and more databases are migrated to the web each day.

2.6.3 Search Engine User Analysis

The field of *search engine user analysis*, or 'information behavior' has been well established over many years [JSBS98, SMHM99, Kir98]. Search engine user analysis studies user interaction with search engines through data mining of query logs or conducting focus groups and interviews. Search engine IR is different to traditional IR [JSBS98]. In a 1988 analysis of Altavista's query logs [SHMM98] is was found that users generally used short queries, rarely looked past the first page of results, and rarely explored modifications of the query. Little has changed in recent years. This highlights the importance of having the most relevant search engine for a query.

Spink et. al. [SWJS01] analysed how users search a major search engine. They found that most searchers use the shortest path route, using few query terms, making few query changes, and viewing few query result pages. Jansen et. al. [JSS00] confirms this study. Zwol [vZvO04] evaluated the usability and retrieval effectiveness of major search engines concentrating on user satisfaction.

2.6.4 Problems with Analysing Search Engines

Search engine content analysis introduces many problems.

The first is that there is usually no direct access to a search engine's statistics, and that there is rarely cooperation between the search engine and the collection broker.

Another is that search engines cannot be accessed using traditional information retrieval methods. Traditional information retrieval methods frequently require communication between a search broker and collections, full knowledge of each collection, or large amounts of metadata about each collection. However the only access to the major search engines is through a web based query interface that generally accepts only short queries. One proposed solution to this is to use a 'dictionary attack' and send every possible term to each engine. However it is too expensive in terms of both time and bandwidth to send hundreds of thousands of queries to each search engine through a web interface, and it is difficult to even know what queries to send as there are many rare terms and acronyms which although they do not occur in the dictionary are

valuable for establishing the subject matter of a collection. For instance, the term "servlet" [10] does not occur in most dictionaries, yet if it occurs in a collection it is a strong indication that the search engine probably contains at least some information related to computing.

A further problem is that many information retrieval metrics require knowledge of the exact size and contents of an information store. Search engines rarely allow this. This prevents many traditional information retrieval techniques from being used in search engine content analysis. It is very difficult to estimate the size of search engines, meaning that measuring performance is difficult. Work has been done on estimating the size of search engines [SC03] but it is still difficult and error prone.

Another significant problem is that search engine content analysis is heavily dependant on the implementation and quality of the search algorithm of each search engine. Some search engines allow boolean queries and other protocols while other systems do not. [11] Other search engines use PageRank type algorithms which tend to deliver superior performance when working with web based information, also making it difficult to compare search engines to each other. Because of the probable high financial returns of good ranking on the major search engines in a competitive area, there are many other search engine algorithms used which are kept secret. Also, because of the high development costs of quality ranking algorithms they are kept secret as they provide competitive advantage over other search engines.

A further problem is that many search engines only permit viewing of a small number of search results. They may report many hundreds of thousands of results, but actually only allow access to a fraction of these results. This means that detailed term usage statistics cannot be used and another method of finding results needs to be found.

French [FPC+99] claimed that in that case of no direct access to a database, it is in principle possible to build an index by issuing a single-term query for each word. Indexing a search engine would involve sending a massive set of queries to each engine in the hope of retrieving all possible data from the collection. Creating these indexes would consume large amounts

[10] "*A Java application that, different from applets, runs on the server and generates HTML-pages that are sent to the client*"
http://www.softwareag.com/xml/about/glossary.htm

[11] Boolean queries allow the use of operators like *AND*, *OR*, or *NOT* in a query to improve performance.

of computation, bandwidth, and memory. While generating a lexicon for each collection is possible by querying every single word in the dictionary it is too slow and inefficient. Also, many standard English dictionaries only include terms that can be called "multi-purpose", and omit specialist terms and acronyms that are of great importance for classification purposes. In addition, the process would have to be repeated if there is ever a change in the collection such as a new addition or update of data. Another constraint is that the metadata should be relatively short in comparison to the database size.

In previous work Ipeirotis et al. used the Yahoo classification scheme as an ontology in order to analyse deep web search engines [IGS01d, GIS02, IG02, IGS01a]. However they only used a small number of subjects, while this research uses a thousand, which can get better results in large search engines such as Google which have a far greater subject distribution then a smaller specialised deep web search engines. Also the Q.U.T. library data set used looks to be superior to the Yahoo dataset. The quality of the Q.U.T. library data set can be demonstrated in that only single terms are used for query samples, while still returning good results, while Ipeirotis et al. used multiple terms. Converting the ontology-based collection selection system to use multiple terms would further improve the performance of the system.

2.7 Summary

This chapter defined terms and concepts related to the World Wide Web, the surface web and deep Web, search engines, Web Intelligence, Information Retrieval, Distributed Information Retrieval, ontologies, collection selection and search engine content analysis. This research covers a broad range of topics, and the topics introduced here will be covered in greater depth in the following chapters.

Chapter 3

Literature Review

This chapter reviews some of the literature related to the work presented in this thesis. Research covered includes automatic ontology learning, query sampling, and a survey of traditional and modern collection selection literature.

3.1 Automatic Ontology Learning Related Work

Automatic Ontology Learning is employed to train the ontology used in this thesis. Automatic Ontology Learning aims to build an ontology without the input of human experts. There is a growing body of work covering automatic and semi-automatic ontology learning. Automatic ontology learning has emerged as a separate entity from other ontology research, drawing from data mining, machine learning and psychology. However, automatic ontology learning is still very difficult to achieve other than in specialised domains.

Viezzer [Vie01] identified four main approaches to ontology research.

1. Development of formal languages such as OIL
2. Building of domain specific ontologies and the tools which aid the building of ontologies.
3. Ontology learning and ontology discovery
4. Development of meta ontologies

This work automates the *ontology learning and ontology discovery* approach, and can be considered similar to thesaurus extraction, which has been studied since the 1960s.

Maedche et. al. [MS00] presents methods for semi-automatically extracting ontologies from domain text. This includes methods for determining the measure of relationship between terms and phrases. Some ontology mining algorithms have been mentioned in [MS01a, MS01b], which are the discoveries of the *backbone taxonomy* and the non-taxonomic relation.

While Maedche has made significant advances in ontology learning, the authors have doubts about whether his work can be ported to the English language. The English language may be considered to be a difficult language to reason with. An examination of natural language literature will reveal a wealth of examples of ambiguous English phrases and sentences which even an semi-intelligent computer would have trouble understanding.

Esposito et al. [EFFS00] provided semi-automatic ontology learning based methods for transforming raw text into a computer readable representation, enabling a computer to learn a language from training examples.

Faure et. al. [FN98] claims to have built a machine learning clustering system which learns subcategorization frames of verbs and ontologies from unstructured technical natural language texts. Unfortunately, in this example the methods were only tested within a single limited domain of cooking recipes which is itself highly structured (for example ingredients and cooking methods are fields common to all recipes).

Another prominent researcher in ontologies and library systems, Welty [Wel98b] uses description logics to allow reasoning and subject based classification within a library catalogue ontology. The user's search experience is improved by allowing search by description logic. In further work [Wel98a] he develops XML markup tagging for description logics with a library catalogue, an important development in the improvement of ontology reasoning. Weldt et. al. [WKCC03] also demonstrated how the use of an improved ontology can significantly improve information retrieval by 19%.

Buitelaar [Bui98] selected 126 classification types and used WordNet as an ontology to assign almost forty thousand polysemic noun terms to one or more types in an automatically generated ontology. Each term could be disambiguated by what set of categories it belongs to or is excluded from. These groupings could then be used to tag corpora to aid automatic

processing of data.

Suryanto et. al. [SC00] applied ontology learning to an existing well structured ontology allowing rapid extraction of rules. Kietz et. al. [KMV00] applied semi-automatic ontology learning tools to a company intranet environment where natural language was mined for information.

The objective of ontology mining is different to semi-automatic ontology engineering. Ontology mining concerns the automatic representation of discovered knowledge. Semi-automatic ontology engineering mainly develops tools to map, merge and align existing ontologies [SS04]. A concept of association sets was presented in [LZ06] to formalize the correlations between classes.

Ontology based world knowledge has previously been used for natural language processing [Whi92], machine translation and sociology. CYC and WordNet are examples of systems which have been trained with world knowledge by domain experts. Significant effort is required to train them, although as a result they contain high quality information. MindNet [LA05] is an example of a system which has been trained by automatic methods such as parsing of dictionaries, encyclopedias and free text.

Li et. al. [LZ04, LZ06] presented a method of automatic ontology learning and refinement which can be used to model web user information needs. Stojanovic [Sto05] used an ontology to refine search queries by removing term ambiguity. Queries were taken and mapped to their neighborhoods in a controlled vocabulary, then the neighborhoods were presented to the user for assessment. Gauch [GCP03] uses an hierarchal weighted ontology to create a personalised user profile and to assist web browsing. The ontology was taken from Yahoo, Magellan, Lycos and the Open Directory Project. The ontologies are used to classify web pages and user browsing habits into different categories, and the user profiles are matched to spidered web pages. Gandon [Gan03] provided methods for managing distributed knowledge and assisting corporate activities by using ontologies.

The above references all contain examples of ontology generation and ontology learning. However many of the above examples use only a small, domain specific ontology with limited

application. This research automatically creates a large ontology potentially covering thousands of different domains.

3.1.1 Search Engine Selection Related Work

Most previous search engine analysis research involved evaluating search engines using meta-data in such areas as size, change over time, overlap, and usage patterns.

In 1998 Lawrence et. al. [LG98] analysed the coverage of search engines in proportion to the total size of the web. They found that even the largest general purpose search engine covered fewer than one-third of the total web. Unfortunately the World Wide Web grows and changes so fast that surveys such as these become quickly outdated.

In the field of search engine performance evaluation [HCTH99, LS99] Hawking et. al. [HCBG01] compared search engines using web search query logs and methods learned from TREC[1]. Hawking et. al. [HCG01] further compared the retrieval performance of eleven search engines based on usefulness of search results for finding online services. Chowdhury et. al. [CS02] compared search engines using known search results, a set of web search query logs, and a corpus with relevance judgements. Beitzel et. al. [BJC+03] uses ODP and Looksmart along with a set of web search query logs to evaluate search engines. Gordon [GP99] and Chowdhury [CS02] show that some search engines perform better than others for some queries. However, overall the search engines returned statistically similar results.

The previous references illustrate the past work in the field of search engine analysis. However none of them perform a full analysis of the content of large, general purpose search engines. The work in this thesis extends this by doing a large scale analysis of the largest search engines in common use today.

3.2 Query Sampling Related Work

Callan [CCD99] proposed that only three hundred query probes are needed to evaluate the contents of a collection; however during this research it was found that it takes many more query

[1]TREC stands for Text REtreival Conference. Website http://trec.nist.gov/

probes to accurately assess the content distribution of a large general purpose collection because of the broad range of subjects they cover. In the same work Callan presented a system that learnt a content summary through query probing a search engine. In other work Callan et. al. [CC01] postulated that "increases in document sample size do not tend to result in comparable improvements in content summary quality".

Fuhr [Fuh99] used a cost based method aimed at reducing selection cost. Methods for estimating the number of results in a collection and the precision-recall statistics for the collection were presented.

Ipeirotis et. al. [IGS01c] trained a document classifier, generated rules from the classifier, then used these rules to perform query probing. The probes that returned the highest number of matches were then used to classify the collection. They [IG02] also used hierarchal "focused query probes" to adapt to the collection contents and try to find the depth of the collection, and estimated document frequencies for each term. Information was stored in a content summary for each collection as a result. However they also argued that because of Zipf's law, query probing cannot find many low-frequency terms in an collection, which leads to incomplete content summaries. Further, they argue that since collections that are topically similar often share the same lexicon, they share content summaries across topically similar collections. They hierarchically categorised and used smoothing for the content summaries to improve content summaries and collection selection.

Panagiotis et. al. [IGS01b] claimed to have trained a system with a set of documents pre-classified into taxonomy topic areas. This paper presents no method, no experimental method or results and does not give a detailed discussion of the training set used. The research also appears to use a different rule generation method to the one presented in this thesis and is not hierarchal in nature.

Gravano et. al. [GIS03] used a document classifier for query probing collections. The authors used machine learning to generate document classifiers, followed by creating rules from the classifiers. The system only used the number of returned results, rather than the actual results. Additionally, they defined coverage and specificity and applied them when selecting

which place in the taxonomy to assign a collection.

Craswell [Cra01] and Callan et al [CCD99] use incremental query probes for server selection. The search broker periodically broadcasts a query to all the servers. From the returned documents, the search broker can apply server ranking methods and select the best servers. Multi-term probe queries are taken from a query log, rather than on the fly, and are not run at query time. Query probes are periodically sent to each server in batches.

Suk et al [CY99]uses a hierarchy of collections. This enables the search to scale and reduces the cost of the search. (However Suk uses a neural net to complete queries).

A common problem with query probing is that metadata about the collection tends to be sparse because it is not efficient to query probe each collection with every possible term and bi-gram. To help solve this a method called *shrinkage* [IG04] (Or *smoothing*) to even out the differences between the collections. Shrinkage involves sharing classification terms across related collections in order to reduce the sparse data problem. However as this research uses a subject based approach the sparse-data problem is not nearly acute as with a term based approach.

3.3 Older Collection Selection Literature

Two of the most popular traditional collection selection methods are CORI and GLOSS. CORI assumes that the best collections are the ones that contain the most documents related to the query. GlOSS uses a server which contains all the relevant information of other collections.

3.3.1 CORI

CORI(*Collection Retrieval Inference Network*) is a Bayesian probabilistic inference network which is commonly used for document selection. Callan [CLC95] and later Lu [LCC96] apply CORI to the collection selection problem. Callan's solution is elegant, and is a popular collection selection method.

CORI assumes that the best collections are the ones that are estimated to contain the most documents related to the query. These collections are therefore ranked in order of number of

documents relevant to the information need. Once ranked, the top *n* collections are presented to the user in a list. Methods are available for calculating the number of documents about a particular term in a collection, for calculating normalised document frequency, inverse collection frequency, and the importance of a collection for a particular term.

CORI uses document frequency and inverse collection frequency to rank each document. Because of this CORI uses minimal storage size for a large collection of documents. Also because an inference network is used for both collection selection and document selection the one system can rank both collections and documents within the same framework. However there are distinctions to be made between collection selection and document selection using an inference network. Collection selection is actually looking for as many documents as possible about a topic. Care must be taken to not discard small sets of relevant documents.

3.3.2 GlOSS

Gravano's [GGMT99] solution to the collection selection problem is to use a server which contains all the relevant information of other collections. Users query this *Glossary of Servers Server* (or GlOSS for short) which then returns a ordered list of the best servers to contact to send the query to.

GlOSS is used to evaluate and rank collections. Collections are evaluated by the usefulness for each query. This usefulness is measured by the estimated number of documents in the collection that are similar to the query. The collections most likely to be selected contain many documents relevant to the current query. Collections are ranked based on information about each collection. However full data on each collection cannot be stored due to size restrictions. Instead each term and each collection is given a number based on the weight of the term and the number of documents in the collection that contain the term. These numbers are periodically updated by a collector program which gets this information from each collection. GlOSS works best with a large collection of heterogeneous data sources.

3.3.3 Comparison of CORI and GlOSS

Craswell [CBH00] compares the collection selection methods described above, CORI and GlOSS. The paper evaluates six different collection selection methods across different data and configurations. The selection methods used in the comparison are:

- CORI
- CORI plus E'i
- CORI plus Ei
- vGlOSS Max(0)
- CVV
- centralised 100% index
- centralised 50% index
- centralised 25% index

In a comparison of CORI and GlOSS [CBH00] it was found that CORI was the best collection selection method, and that a selection of a small number of collections could outperform selecting all the servers and a central index. Probe queries are a good method for evaluating collections without having full knowledge of the collection contents, with 50,000 documents evaluated instead of 250,000 documents.

These conclusions are important to this research because they show that a high quality subset of collections will be as effective as a full set of collections, and that probes of a collection are an effective method of ranking an entire collection.

This is related to Chen's [CM00] Yarrow system in which a number of servers are used to search the world wide Web. By selecting only the most relevant collections for each query the system will be faster and use less bandwidth.

CORI and GlOSS are not deep web collection selection techniques because they require full knowledge of each collection, require communication between search broker and collection, and keep metadata on each collection. In this thesis a method which does not require communication between the search broker and search engines is used.

3.3.4 bGlOSS and vGlOSS

Two variations of GlOSS are available, *bGlOSS* [GGMT94] is a boolean version of GlOSS, and *vGlOSS* [GGMT99](also know as *gGlOSS*) which is a vector-space version of GlOSS. A decentralised version of GlOSS is also available, called hGlOSS.

Each server contributing to bGlOSS uses a Boolean evaluation system, which communicates document and server statistics back to the broker. Each server is ranked by an estimate on how many documents in the collection satisfy the query. However this method assumes that query distribution is independent across the collections.

vGlOSS uses a vector to represent each document, with each term in the space given a weighting based on frequency and normalisation. A query is also represented as a sparse vector, and is compared to the other document vectors.

Server ranking methods *Sum(l)* uses a vector sum of the server's normalised document vectors, while *Max(l)* uses the document frequency statistics about each server to compare the vectors. They estimate the summed scores of documents which score above l, and use the inner product to rank collections. The goodness value of each collection c_i with respect to query *q* is:

$$G_{i,q} = \sum_{j=1}^{M} W_{i,j} \tag{3.1}$$

In Equation 3.1 $W_{i,j}$ is the sum of document weights contributed by term q_j in collection c_i and *M* is the total number of terms for collection c_i. A threshold is used to ignore terms that are less than a certain weight.

The problem with this is that unless each collection uses the same ranking method, comparing the results becomes difficult. A solution can be found through message passing between

collections, but in the real world this is difficult to implement unless each collection has the same interface and methods.

3.4 Recent Collection Selection Literature

Si et. al. [SC03] present a web based modification of CORI called *ReDDE* which performs as well as or better than CORI by using an collection size estimate to supplement selection. They introduce a collection size estimation technique which is more efficient than in other estimation techniques such as the capture-recapture method [LYM02].

Hawking et al [HT05] presented a method which used both centralised and distributed collection selection techniques. They also made use of anchor text to extract information on collections that have not been indexed.

Si et. al. [SC05] presented a method for minimalising the poor quality results returned by collections which have not implemented good information retrieval methods. By including the retrieval performance of each collection in the collection ranking, this problem can be reduced. A method for approximating the retrieval effectiveness of a collection, known as RUM, was presented. The RUM method was compared to CORI and outperformed CORI in all the experiments conducted.

Meng et al [MYL02] divides collection description into three approaches: rough representative, statistical representative, and leaning based. Rough representations use a few keywords or short sections of text to represent a collection. There are serious problems with this model, including its inability to match a query to a collection where the metadata does not contain any of the query terms. Statistical methods represent the collection as a set of term frequencies and weights. Although suitable for collections with direct SQL access, statistical representation is not suitable for the deep web, as it takes too many queries to generate the term index. The learning-based approach uses a form of relevance feedback. Previous queries results measure how relevant a collection is to the current query. One problem with this approach is that if the collection is updated previous queries may be rendered useless. Furthermore if a new query is encountered the system is unable to suggest a suitable collection.

Experiments with language modeling and US Patent data showed that the document scores on systems that use IDF ranking (such as INQUERY [CCH92]) become skewed when used on collections ordered by subject [Cal00]. This means that subject specific collections receive low scores when ranked on their own subject matter. Normalisation methods have been used reduce this but it is still a large problem with such systems.

Nottelmann and Fuhr [NF03a] used a decision theoretic framework for collection selection. The decision theoretic framework tries to minimise costs such as retrieval effectiveness, time and money that are associated with distributed information retrieval. They compared their system with CORI [NF03b] and found that is was comparable to CORI, except when using short queries when it performed worse than CORI. In later work [NF04] they showed that combining CORI with a decision theoretic framework can produce improvements.

Si and Callan proposed a method called "Unified Utility Maximization Framework"(UUM) [SC04]. A database is built based on the results of query samples, and the database is then used to produce relevance probabilities for each new query. The system then optimizes collection selection based on whether the collections provide better results returning either precision or recall.

In later work Si and Callan introduced a "Returned Utility Maximization" (RUM) [SC05] method. The central idea is that the algorithms used by some search engines are not as effective as they should be. RUM builds on UUM by estimating the retrieval effectiveness of each collection and uses these estimations for collection selection. The RUM method outperforms CORI in all the experiments

Hawking et al [HT05] presented a method which used both centralised and distributed collection selection techniques. They also made use of anchor text to extract information on collections that have not been indexed. Collections are ranked based on how closely the anchor text was to the page they referred to. The method performed well against CORI in some experiments.

This researchers previous work [Kin03, KL03] in deep web based collection selection used query sampling methods that did not require communication with the broker or metadata about

each collection. Singular value decomposition was then used on the results of the queries to select the best collection. These techniques were tested on the INEX collection with satisfactory results. [KLBN06] presented the first comparison with ReDDE using a smaller version of the ontology than the one given in this work. In [KLTN07] this method was used to compare the major search engines against each other. Again a smaller version of the ontology than the one presented here was used.

Shokouhi et al [SZTS07] use query logs for collection selection. They also introduce a pruning method that removes terms judged to be less important to users.

3.5 Summary

This thesis covers a wide range of concepts related to information retrieval and web intelligence. This chapter gave a review of the academic literature related to the concepts presented in this thesis. Automatic ontology learning is a web intelligence method used for extracting high level rules from an ontology without the use of a human expert. Query sampling is a collection selection method used for finding the information contained in a collection with only limited access to the collection data store. Collection selection is a method for comparing and selecting the best information collection for a user need. The academic literature related to these concepts was reviewed and discussed in this chapter.

Chapter 4

Ontology Mining

Human experts are currently better at identifying relevant documents than the most advanced information retrieval methods. Human experts are also currently better at classifying documents than the most advanced automatic classification methods. One factor that makes human experts superior to computer programs is '*world knowledge*'. Ontologies can be used to give computer programs access to the world knowledge and experience of human experts. This research generates a large ontology which is then mined to extract classification rules. This chapter gives a description of how the ontology used in this thesis was built and mined.

4.1 Planning the Ontology

Usually a large ontology is manually constructed by a group of experts working with an ontology construction framework. However manual ontology construction by experts is slow, as entering rules into an ontology takes much skill and background knowledge. Also human experts often only have expert knowledge of a small number of domains. An important property of an ontology is that it covers many subjects and is of good quality. Automatically building such an ontology is still the focus of much research. Automatic ontology learning will be a great improvement, enabling technologies to facilitate the creation of the semantic web.

The method presented in this thesis uses an *automatic* ontology learning method which results in an expert trained system with a knowledge of a large set of domains. The ontology

can then be used to classify collections, find term usage patterns, subject classification patterns and other rules. The stages involved in the ontology construction process are:

1. Selecting a classification taxonomy
2. Identifying a training set
3. Populating the ontology
4. Cleaning up the ontology

4.1.1 Selecting a Classification Taxonomy

There are several desirable properties in a good taxonomy. The taxonomy should cover a wide number of subjects, be carefully constructed, be standard across the world, and be available in different languages.

It was decided to use the Dewey Decimal System, a widely used library classification system[1]. The system has been used for the classification of a large number and wide range of materials, and is reportedly used in over 200,000 libraries worldwide and in over 130 countries[2].

The Dewey taxonomy is based on a pattern of ten root nodes, with each root node having ten child nodes. Each child node has another ten child nodes with this pattern continuing downwards. There can be many different levels of the taxonomy, depending on how precise the subject match is. There are 1,110 classification nodes at the top three levels of the taxonomy, with many more nodes in the lower levels. There are some subject nodes that are unused because of depreciation or limited coverage. In this research only the top three levels of the Dewey Decimal system are used.

The top Dewey level contains general subject classifications, while the bottom Dewey level contains specific subject classifications. Each subject node covers the entire set of subject nodes below it. Moving down in the taxonomy focuses on a specific subject. Moving up in the

[1]For a full listing of the classifications see http://www.tnrdlib.bc.ca/dewey.html. For an example of the classification system in use see the Q.U.T. university's library web site. http://libcat.qut.edu.au/

[2]http://www.oclc.org/dewey/

DDC	Subject Description
000	Generalities
100	Philosophy and Psychology
200	Religion
300	Social Sciences
400	Language
500	Natural Sciences and Mathematics
600	Technology (Applied Sciences)
700	The Arts
800	Literature and Rhetoric
900	Geography and History

Table 4.1: The Top Level of the taxonomy

taxonomy gives more general coverage. Table 4.1 shows the subject descriptions of the ten root nodes of the taxonomy.

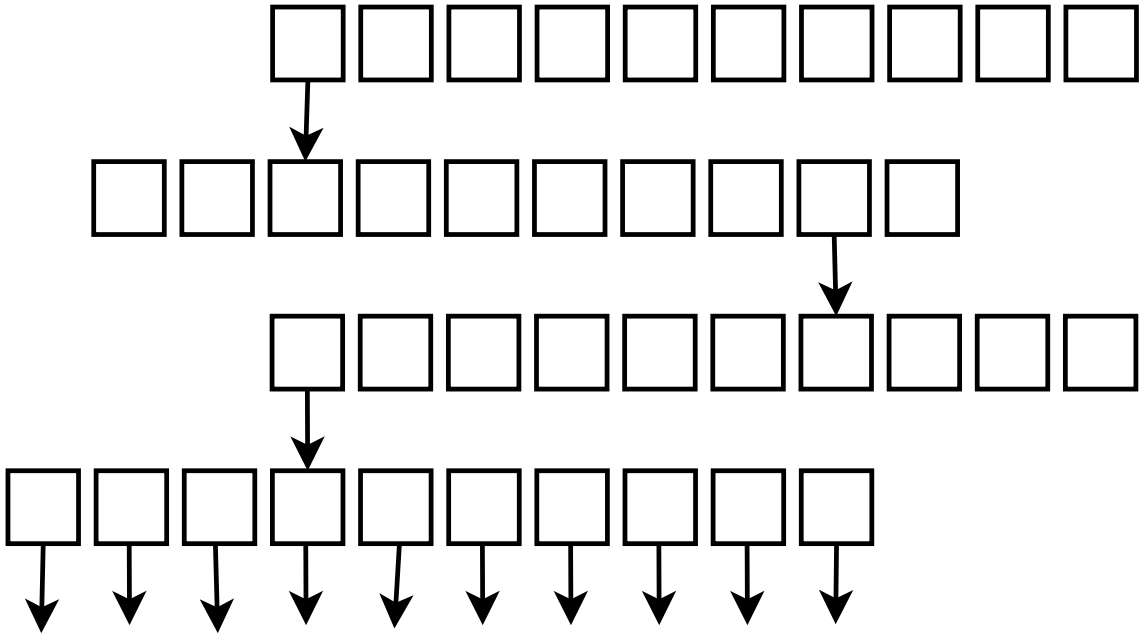


Figure 4.1: The Dewey Decimal taxonomy

Figure 4.1 shows an abstraction of the Dewey hierarchy, where each node has ten nodes below it, continuing downwards for as many levels as necessary. Many subjects overlap with other subjects, meaning that a collection can belong to more than one subject. Figure 4.2 shows how mlutiple subjects are interlinked within the hierarchy. Each Dewey Decimal Code (DDC) provides the best possible classification for each item. Figure 4.3 shows a portion of the taxonomy. The meta subjects are at the top of the taxonomy, as the taxonomy moves downwards the

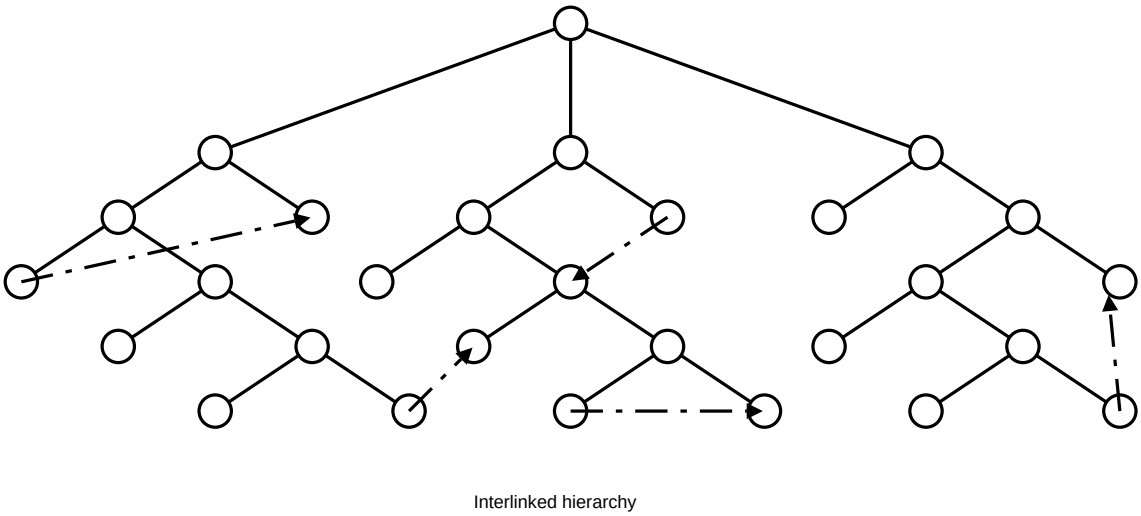


Figure 4.2: Interlinked Subjects

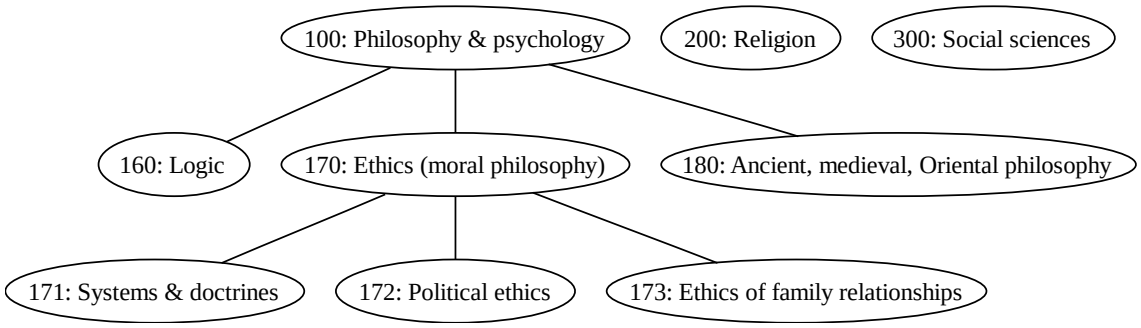


Figure 4.3: A Portion of the taxonomy

subjects become more specific.

In its simplest form the Dewey taxonomy consists of terms and subjects in the form of a *backbone taxonomy*. Each node in the taxonomy represents a subject. Each subject node can be represented by a description of the subject, a set of terms and term weights which are associated with the subject, and the inter-relations between it and other subject nodes. One term can be assigned to many different subject nodes; however if a term belongs to only a few subject nodes it is often better for classification than a term that belongs to many subject nodes.

Compared to many other web taxonomies such as the Open Directory Project (ODP)[3], the Looksmart directory[4], and the Yahoo directory[5], it may be argued that the Dewey taxonomy is substantially better planned. Web directories are notorious for misclassified documents aris-

[3]http://dmoz.org/
[4]http://www.looksmart.com/
[5]http://dir.yahoo.com/

ing from sub-standard data handling that reflects a poor understanding of taxonomy. Another problematic issue with web directories is the presence of ad-hoc subject nodes. An advantage of the Dewey Decimal system is that the taxonomy is designed by trained classification experts, and almost all the additions are made by classification experts. The Dewey system is also multilingual, and it is possible to train the IntelliOnto ontology with other languages.

It would be possible to use other classification schemes (such as the Library of Congress Classification scheme[6] or the Universal Decimal Classification Scheme) however the Dewey System was chosen because it is one of the more universally used schemes, it is better planned from a hierarchal perspective, and has more multilingual training data available. The authors also investigated using WordNet as a training set, however there was too little training data in the WordNet examples and it did not contain enough subject-specific terms for it to be of use.

Previous use of ontology based library classification schemes as an automatic classifier include [GS01] which uses the Library of Congress Classification scheme to automatically classify web documents, and [JGR00] which organises web materials into a machine readable format with RDF syntax.

The Dewey Decimal System is not perfect. For example, take the category *200 Religions of the World*.

- 210 Natural theology
- 220 Bible
- 230 Christian theology
- 240 Christian moral & devotional theology
- 250 Christian orders & local church
- 260 Christian social theology
- 270 Christian church history

[6] http://www.loc.gov/catdir/cpso/lcco/lcco.html

- 280 Christian sects & denominations
- 290 Other religions

This category system was designated in 1876 and there was obviously no plan for future developments in religious culture.

4.1.2 The Training Set

The desirable properties of a training set are that it is large, of high quality, available in electronic form, and cover a wide range of subjects. A data set reflecting these requirements is the Queensland University of Technology(Q.U.T.) Library Catalogue[7], which contains over 400,000 usable items each classified with a Dewey Decimal code. This library catalogue is excellent for use as a training set because most of the entries have extra meta-information such as descriptions and summaries. It should be noted the Queensland University of Technology Library Catalogue is but an exemplar of an ontology which can be employed. The method presented in this thesis is by no means tied to this particular source of knowledge.

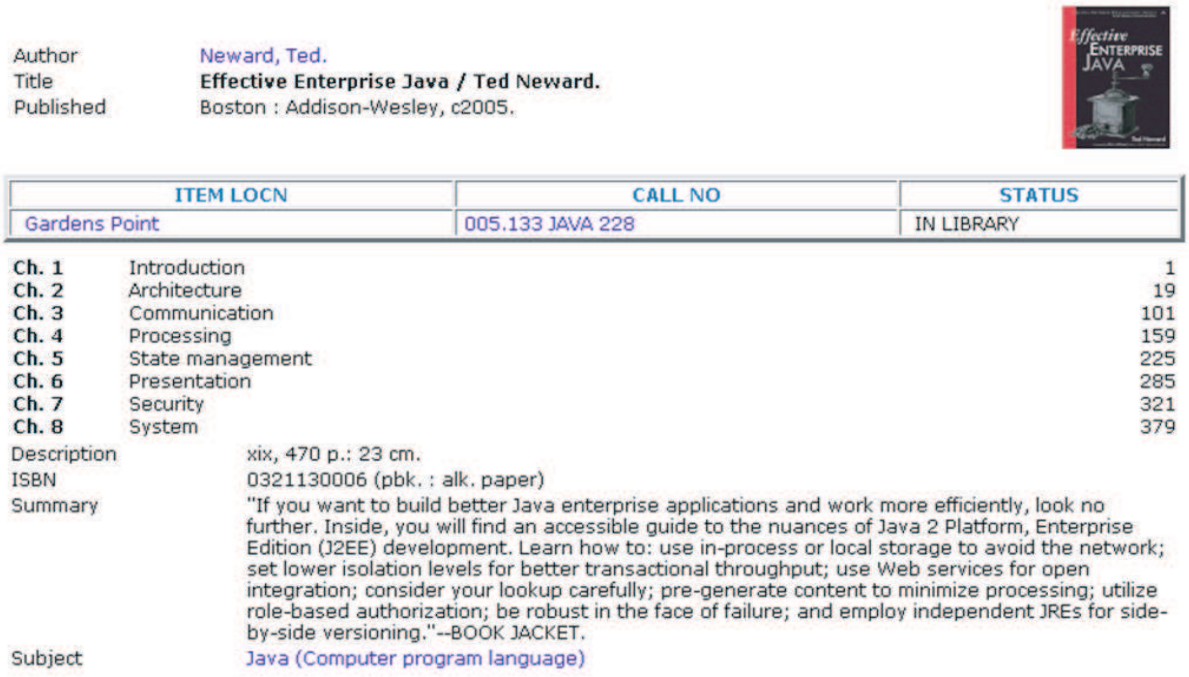
This data set was used to populate the ontology with world knowledge. This is referred to as the "IntelliOnt" construction process. (See [KLTN07] for an introductory paper on this method). Figure 4.4 shows an example item from the training set. Data extracted from this page includes the title, Call Number (Dewey Decimal Code), chapter titles and summary. Note that many of the terms are highly descriptive. Stopwords are discarded. Each document in the training set is assigned a Dewey Decimal Call Number. These documents have been carefully classified by experts in the field, and the information is of superior quality to other web based directories. However if anyone wishes to replicate this research, any university library with a wide and deep representation of knowledge would suffice as a training set.

As a note, requiring good quality training data rules out automatically classified information, because the classification quality of such information generally cannot compare to the quality of expert classified information. For example *k-means* clustering [Mac67] is often inaccurate and error prone.

[7] See http://libcat.qut.edu.au/

All the items from the Q.U.T. library database are used for training data. The data extracted includes the document title, chapter headings, description, notes, Dewey Decimal Number, and subject groups. This metadata can be treated as a compression of the document. Many of the terms contained in the metadata can help to distinguish the exact subject matter of the document. Each piece of metadata we extract is rich in keywords and may contain summary and description information. The document itself is not indexed as few library documents are currently in full text form. However in coming years it is expected that many library items will be available in electronic form, improving the ontology even further. [8]

With the ontology, the more document metadata available for training it the better it will perform. Performance increases as a factor of the training data used. This ontology can easily be trained for other subject domains as required. Also, if the user enters a query term that is not currently in the ontology, the ontology can set out to retrieve relevant training data from other libraries that use the Dewey Decimal system from around the world.



Author	Neward, Ted.
Title	**Effective Enterprise Java / Ted Neward.**
Published	Boston : Addison-Wesley, c2005.

ITEM LOCN	CALL NO	STATUS
Gardens Point	005.133 JAVA 228	IN LIBRARY

Ch. 1	Introduction	1
Ch. 2	Architecture	19
Ch. 3	Communication	101
Ch. 4	Processing	159
Ch. 5	State management	225
Ch. 6	Presentation	285
Ch. 7	Security	321
Ch. 8	System	379

Description	xix, 470 p.: 23 cm.
ISBN	0321130006 (pbk. : alk. paper)
Summary	"If you want to build better Java enterprise applications and work more efficiently, look no further. Inside, you will find an accessible guide to the nuances of Java 2 Platform, Enterprise Edition (J2EE) development. Learn how to: use in-process or local storage to avoid the network; set lower isolation levels for better transactional throughput; use Web services for open integration; consider your lookup carefully; pre-generate content to minimize processing; utilize role-based authorization; be robust in the face of failure; and employ independent JREs for side-by-side versioning."--BOOK JACKET.
Subject	Java (Computer program language)

Figure 4.4: Example Training Set Page.

[8] See http://books.google.com/ for a prototype full text book repository.

Term	DDC	Document ID	Frequency
approach	001.39	36	2
approach	101.1	46	1
interdisciplinary	001.39	36	1
psychoanalysis	001.40	37	1
domain	001.40	37	1
domain	001.70	39	1

Table 4.2: A Sample of the IntelliOnto Ontology Base

4.1.3 Populating the IntelliOnto Ontology

The Q.U.T. Library data set was used to populate the ontology with world knowledge. Each document in the training set is assigned a Dewey Decimal Call Number. These documents have been carefully classified by experts in the field, and the classification quality is superior to web based directories.

The raw training set data was taken and transformed into a large set of low-level classification terms which were used as the base of the IntelliOnto ontology. Table 4.2 shows a sample of the base of the IntelliOnto ontology. This base is built by iterating through all the training set items from the library catalogue. Each item has a Dewey Decimal code associated with it. The metadata for each item such as title, chapter headings, summary and description is retrieved. Once the metadata for the item is retrieved from the library database it is parsed into terms, and each term and its Dewey Decimal code are saved in the IntelliOnto ontology. This information is then used to generate classification rules.

In this thesis only the third level of the Dewey Decimal system is used to populate the ontology. However it is expected that even better results could be produced if the fourth level (10,000 possible subjects), fifth level (100,000 possible subjects), or sixth level (1,000,000 possible subjects) were used with an improved mapping system. It should be noted that not all the subjects on each level are used. There are large gaps in certain places. This is due to depreciation of older subjects or areas set aside for later expansion.

Term	Part Of Speech	Count	ISF
history	NOUN	5910	-1.77664583141801
social	ADJ	4378	-1.47659199910067
congresses	NULL	4206	-1.43651207728051
book	NOUN	3166	-1.15246896145882
art	NOUN	3151	-1.14771986277514
study	NOUN	3041	-1.11218640869523
great	ADJ	2578	-0.947013904516996
language	NOUN	2547	-0.93491619599732

Table 4.3: Terms with low Inverse Subject Frequency.

4.1.4 Cleaning Up the IntelliOnto Ontology

Stopwords are terms that are too common to be of any use in the classification process. Stopwords are removed to improve performance. Most information retrieval systems use a static stopword list. However, the IntelliOnto system uses a dynamic stopword list which is created by identifying terms in the ontology which cover too many subject areas to be of any value. This is useful in situations where a static stopword list is not readily available, for example if this system was implemented in another language such as French it might be easier to generate a dynamic stopword list than to search for a static stopword list. The *Inverse Subject Frequency*(ISF) metric (as defined in the Definitions section) was used for the purpose of identifying and removing these stopwords. Table 4.3 shows the terms from the IntelliOnto ontology with the lowest Inverse Subject Frequency. These are terms that occur across so many subjects and have so many senses that they are of no use for classification. Note that all of these terms will occur in a common English dictionary, yet they are of little or no use of identifying the subject matter of an item because they are so widely used. This leads to the notion that for a term to be included in a common English dictionary, it must be multi-purpose or reusable across different subjects.

The ISF metric is a variation of the *Inverse Document Frequency*(IDF) metric commonly used in information retrieval. The IDF metric is represented as:

$$IDF = log \frac{collection\ size}{number\ of\ documents\ containing\ the\ term} \tag{4.1}$$

The ISF weighting system means that the value of a term increases when it occurs in few subjects, and decreases when it occurs in many different subjects. The ISF metric is represented as:

$$ISF = log\frac{number\ of\ subjects}{number\ of\ subjects\ containing\ the\ term} \tag{4.2}$$

All terms which score below a user specified threshold are removed from the ontology. As the number of subjects can change, the threshold is not a constant. The calculation is based on the third level of the taxonomy. If the calculation is made on higher levels, the number of stopwords may be reduced as some terms belong in many subject groupings under the one super-subject grouping.

4.2 Formalization of IntelliOnto Ontology

In this chapter the formalisation of the proposed IntelliOnto Ontology is presented along with the related automatic ontology learning methods. Firstly the ontology structure is described, followed by the lexicon level of term-subject matrix of the proposed ontology. Next the first step of ontology learning is introduced for how the candidate terms representing a subject are selected. Finally the ontology is built based on the relationships existing in the subjects.

4.2.1 The Ontology Structure and Term-Subject Matrix

Definition 1 *Let $OntoBASE$ be the ontology base of the taxonomy, the ontology base is formally defined as a 2-tuple $OntoBASE :=< S, R >$, where*

- *S is a set whose element is called subject;*
- *R is a set whose element is called relation.*

Definition 2 *A subject s in the subject set $S := \{s_1, s_2, \cdots, s_n\}$ is a 3-tuple $s :=< code, termset, abs >$, where*

- *$code$ is an unique identification code assigned by the Dewey Decimal Code system to the subject s;*

- $termset$ *is a set of terms representing the subject* s*;*
- abs *is an abstract name of the subject* s*.*

With regard to each s in this paper the taxonomy code of a subject s is denoted by $code(s)$, the set of terms representing s is denoted by $termset(s)$, and the name of s is denoted by $abs(s)$.

Definition 3 *A relation* r *in the Relation set* $R := \{r_1, r_2, \cdots, r_m\}$ *is a 3-tuple* $r :=< type, x, y >$*, where*

- $type$ *is a set of relationship types, which has two elements* $type := \{kindOf, partOf\}$*;*
- x *and* y *are the subjects or terms that hold the relation* r*.*

Usage: $< kindOf, s_1, s_2 >$ means *subject* s_1 *is a kind of* s_2. And $< partOf, t_1, s_1 >$ means *term* t_1 *is a part of subject* s_1. Regarding to r, the $type$ of r is denoted by $type(r)$, and x by $x(r)$, and so as $y(r)$.

A lexical level for the ontology named *term-subject matrix* is defined as a quadruple and described as follows:

Definition 4 *A term-subject matrix* $M(O)$ *in the ontology structure* $O :=< S, H(S), R, \sigma >$ *is a quadruple* $M(O) :=< T, S, TS, \eta >$*, where*

- T *is the set of terms assigned to all subjects* S*;*
- TS *is a* $m \times n$ *zero-one matrix, where* $n = |T|$ *and* $m = |S|$*. For example,* $TS(t_i, s_j) = 1$ *means term* $t_i \in termset(s_j)$*, and* $TS(t_i, s_j) = 0$ *means* $t_i \notin termset(s_j)$
- η *is called a reference, a mapping* $(\eta : T \rightarrow 2^S)$*, that defines the associated terms to a subject* O_s*. For a term* $t \in T$

$$\eta(t) = \{s \in S | TS(t, s) = 1\} \tag{4.3}$$

and its reverse is a set of terms, which satisfies

$$\eta^{-1}(s) = \{t \in T | TS(t, s) = 1\}. \tag{4.4}$$

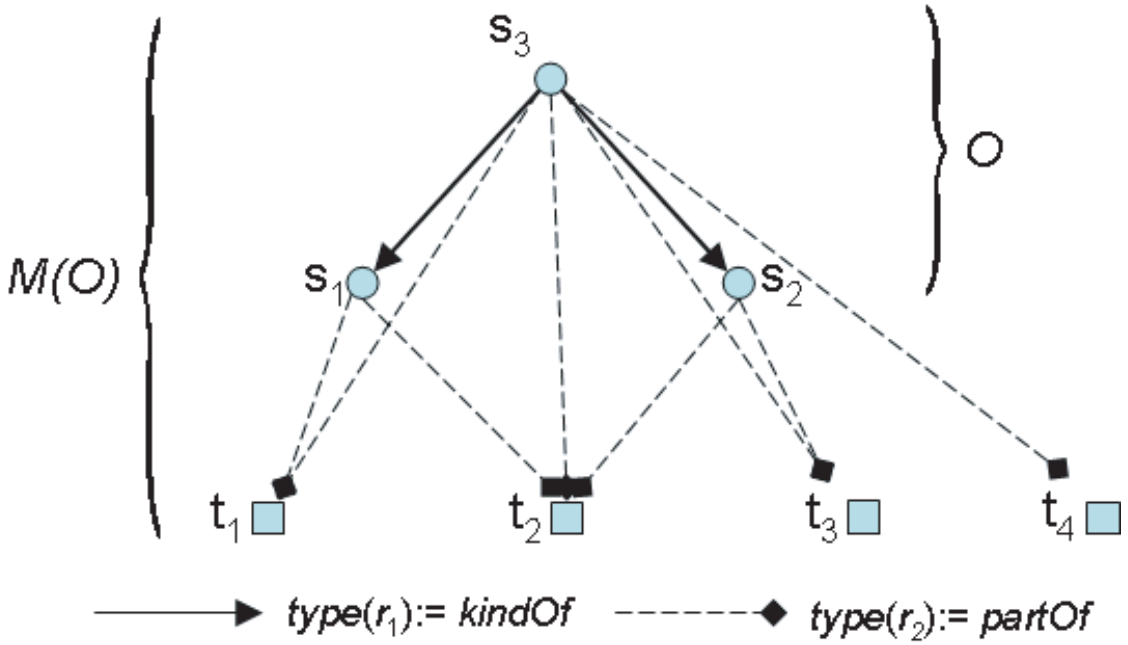


Figure 4.5: A Illustration of the Simplified Ontology.

	s_1	s_2	s_3
t_1	1	0	1
t_2	1	1	1
t_3	0	1	1

Table 4.4: A Simplified Term-Subject Matrix

Based on the term-subject matrix $M(O)$, one term may refer to multiple subjects, and one subject may be referred to by multiple terms. Table 4.4 illustrates a simplified sample of term-subject matrix. Given a term t_1, a set of relevant subjects $\{s_1, s_3\}$ that t_1 refers to can be identified. On the other side, given a subject s_2, a set of relevant terms $\{t_2, t_3\}$ that s_2 is referred by can be identified as well. By using the term-subject matrix 4-tuple $M(O) :=< T, S, TS, \eta >$, the sample matrix can form a sample ontology which is illustrated in Figure 4.5 where s denotes a subject and t denotes a term. One may see in the figure that the subject s_3 is the parent of s_1 and s_2, as all the terms including those referring to s_1 and s_2 refer to s_3. s_3 covers broader conceptual area than s_1 and s_2. This shows that a subject can be a kind of a higher level subject, and that a term can be a part of any subject.

With the term-subject matrix, the proposed ontology can be further defined as a pair of $(O, M(O))$, where O is the ontology structure and $M(O)$ is the term-subject matrix. Given two

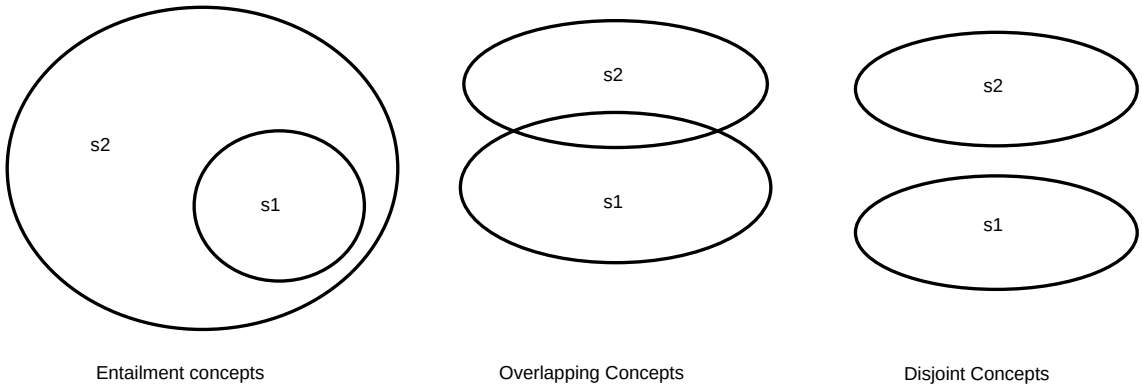


Figure 4.6: Various Situations of Conceptual Areas Referred by the Different Subjects

subjects s_1 and s_2, if $termset(s_1) = termset(s_2)$, it is said that $s_1 = s_2$. If $termset(s_1) \subset termset(s_2)$, it is said that s_1 is a $kind-of s_2$, since every term referring to s_1 also refers to s_2, but s_2 has more term referred. The conceptual area referred by s_1 is entailed in s_2 in this case. If $termset(s_1) \cap termset(s_2) \neq \emptyset$ and $s_1 \neq s_2$, it is said that the conceptual areas referred by s_1 and s_2 are overlapping. However, if $termset(s_1) \cap termset(s_2) = \emptyset$, the conceptual areas referred by s_1 and s_2 are disjointed. The situations of containing/contained, overlapping, and disjointed concept areas are illustrated in Figure 4.6.

4.2.2 Assigning Candidate Terms to a Subject

Let $|D|$ denote the length of the training document set D and m be the number of subjects. Each document $d \in D$ in the set D is assigned a subject s. Each document $d \in D$ is represented by a set of terms $termset(d)$, $df(t)$ is the number of documents in D with $t \in termset(d)$, and $sf(t)$ is the number of subjects in S with $t \in termset(s)$. Instead of the traditional *inverse document frequency*(idf) [SB87] presented as Equation (4.1), *inverse subject frequency*(isf) is presented as Equation (4.2).

isf shows terms that occur across too many different subjects to be of any use. The inverse subject frequency method is used for term pruning. Terms with low isf value are considered "stopwords" and are subsequently removed from *T*.

A term-subject pair $p(t \rightarrow s)$ in $M(O)$ with their confidence and support values mined from the ontology is referred to as a classification pattern $p(t \rightarrow s) :=< t, s, conf(t \rightarrow s), sup(t \rightarrow s) >$ in this research, where $t \in T, s \in S, conf(t \rightarrow s) \in [0, 1]$ and $sup(t \rightarrow s) \in [0, 1]$.

A modified support and confidence method is used for the ontology-based system, in order to accommodate the taxonomy. The $conf(t \rightarrow s)$ and the $sup(t \rightarrow s)$ in the pattern describe the extent to which the pattern is discussed in the training set. The $conf(t \rightarrow s)$ and $sup(t \rightarrow s)$ are defined as follows:

$$conf(t \rightarrow s) = \frac{sf(t,s)}{sf(t)} \tag{4.5}$$

$$sup(t \rightarrow s) = \frac{sf(t,s)}{m} \tag{4.6}$$

where $sf(t,s)$ is the number of child subjects under s (including s) with t occurred in the $termset$. The greater $sup(t \rightarrow s)$ and $conf(t \rightarrow s)$ are, the more important the term t is to the subject s.

Algorithm 1 shows the process for selecting the candidate terms which represent a subject and generating the rules which specifies the level of confidence of the candidate terms. Only the top k terms with the highest confidence and support values are selected as the candidates in the $termset(s)$ to represent the subject s. To avoid rare terms and spelling mistakes only terms that occur more than twice in the ontology are used for the generation. It returns a set of subjects, and a set of classification terms for each subject. These terms will then be sent to each search engine in order to classify the subject coverage of each search engine.

4.2.3 Build Taxonomy from Lowest Level

Based on Definition 4, for any child subject s_c on the lower level of taxonomy and its parent subject s_p on the upper level, there is $termset(s_c) \subset termset(s_p)$. For any subjects on the lowest level which have no child, its candidate term set may be its final term set. However, for the subjects on the upper levels which have children, another method is needed. The final terms assigned to $termset(s_p)$ may consist of its children subjects' term sets and its own candidate term set, which may be formalized as:

$$termset(s_p) = \langle \bigcup_{<kindOf,s_c,s_p>} termset(s_c) \rangle \cup termset(s_p^{'}). \tag{4.7}$$

Algorithm 1 Term-subject pattern miner

Input 1: Given raw $AssociationTable$ from Algorithm 4 in the form of $< DDC, term, termcount >$
Input 2: k - the experimental coefficient.

Outputs: $SubjectAssociationTable$ in the form of $< DDC, term, termcount, confidence, support >$ - which is a table for a set of subject codes, a set of all terms in each subject code, and termcount and support and confidence for each term in each subject code.

For a definition of support and confidence see Chapter 4.

LET $SubjectAssociationTable = \emptyset$;
LET S be the set of subjects in $AssociationTable$;
LET T be the set of terms in $AssociationTable$;
for each $s \in S$ **do**
  let $P = \{p|p = (t, s, conf(t \rightarrow s), sup(t \rightarrow s)), t \in T\}$//$P$ is the pattern set of s;
  sort P by $conf(t \rightarrow s)$ values in descending order;
  for each group of the patterns with the same $conf(t \rightarrow s)$ value **do**
    sort the patterns by $sup(t \rightarrow s)$ values in descending order;
  end for
  for the top k pattern $< DDC, term, termcount, confidence, support >$ **do**
    add the $< DDC, term, termcount, confidence, support >$ to the $SubjectAssociationTable$;
  end for
end for

Subject	Term set	Terms
s_1	$termset(s_1)$	{computer, information, system}
s_2	$termset(s_2)$	{system, organisation}
$s_k^{'}$	$termset(s_k^{'})$	{information, technology, system}
s_k	$termset(s_k)$	{computer, information, system, organisation, technology}

Table 4.5: Subject examples

where $termset(s_p^{'})$ is the candidate term set assigned to s_p by using the algorithm introduced in "term-subject pattern miner". If a subject s is on the lowest level and has no child, the $termset(s) = termset(s^{'})$. The $termset(s)$ that the subject s consists of would be only its own $termset(s^{'})$. Table 4.5 illustrates an example of that, where s_k is the parent of subjects s_1 and s_2, $termset(s_k^{'})$ is the candidate term set assigned to subject s_k by the valuable patterns, and $termset(s_k)$ is the final term set representing the subject s_k.

Hierarchal taxonomy knowledge allows us to make set based inferences such as **isa** which is defined as *set inclusion* and **hasa** which represents attributes of the items in the set. Each item in the set has properties which defines how strongly the item belongs to the set. An item can belong to more than one set, however items that belong to one and only one set are used in this thesis. A **hasa** confidence property with a value of 1.0 shows that an item belongs to one and only one set [Bui98].

4.2.3.1 Decision Rules in Collections

Let C be a set of collections, where each collection is a set of documents. A document d is a set of terms (may include duplicate terms). Let $V^C = \{t_1, t_2, \ldots, t_n\}$ be a set of selected terms for all documents in C.

A set of terms X is referred to as a *termset* if $X \subseteq V^C$. A collection c can be organized into a taxonomy of subjects. Given a *termset* X and a set of subjects Z, the association between X and Z can be described as a decision rule $X \Rightarrow Z$. X is called the condition of the rule and Z the decision of the rule. More formally a decision rule can be represented as a tuple $(X \Rightarrow Z, support, confidence)$, where *support* is the probability that the collection contains

both X and Z; and *confidence* is the conditional probability that the collection having X also contains Z.

For example, a rule for the collection "computer" in Q.U.T. library might be $(\{Computer, Java, Programming\} \Rightarrow JavaProgramming, 0.82, 0.55)$.

This means that if the terms *Java* and *programming* occur together, the document can be classified as including the subject *Java programming* with support of .82 and confidence of .55.

4.3 Mining From the IntelliOnto Ontology

There are many different classification rules that can be mined from the IntelliOnto ontology by using the terms, the subjects, and the taxonomy. By finding patterns between subject nodes and terms classification rules can be discovered. These rules can then be made more useful by applying the taxonomic nature of the Dewey Decimal system.

When mining classification rules from the IntelliOnto ontology the correlation between the nodes of the taxonomy can be described in an association table. In its simplest form the association table shows the terms and the subjects that they belong to.

The classification terms need to be carefully selected. These terms should preferably be subject-specific (occurring within few or no other subjects) and should occur frequently within the subject and infrequently in other subjects. It is difficult to decide which terms to select as there are many possible terms to describe a subject. Many terms may not occur in common English dictionaries yet are still valuable for classification. These may include technical or subject specific terms such as conference names, acronyms and names of specialist technology. Some examples from computing are *RMI* [9], *SMIL* [10], and *XSLT* [11]. Few standard English dictionaries include these terms, yet if any of these acronyms occur in a document it is highly likely the document covers a subject related to computing.

[9] Remote Method Invocation.
[10] Synchronized Multimedia Integration Language
[11] Extensible Stylesheet Language Transformation.

term	term count
software	281
programming	205
security	200
program	191
web	152
object	117
database	117
programs	105
applications	101
microsoft	99
processing	92
user	85
databases	83
server	81
windows	81
oriented	78
electronic	76

Table 4.6: Terms that occur most frequently in *005 Computer programming, programs, data*

For example, the term *servlet*[12] does not occur in most standard English dictionaries, yet it is a valuable term for identifying that a document contains information related to Java applications.

The first term selection method, *highest term frequency*, involves selecting the most frequent terms from each subject. Table 4.6 shows the most frequent terms for the subject *005 Computer programming, programs, data*.

The second term selection method, *highest support and confidence*, involves finding the most distinguishing (or unique) terms from each subject based on confidence and support. These are terms that occur frequently within the subject, yet infrequently in other subjects. Support in this work is defined as the number of times the term occurs in a subject divided by the number of terms in the ontology. Table 4.7 shows the most distinguishing terms for the same subject. These terms cluster around the Dewey Decimal code "005". The nodes are grouped based on the third level of the taxonomy, any subject groupings below this level are not considered. Note that few of these terms would occur in a standard English dictionary, yet they are excellent for

[12]"*A Java application that, different from applets, runs on the server and generates HTML-pages that are sent to the client*"
http://www.softwareag.com/xml/about/glossary.htm

Term	Count	Support	Confidence
c#	55	0.00003840	1
j2ee	48	0.00003351	1
javabeans	43	0.00003002	1
fedora	27	0.00001885	1
sax	27	0.00001885	1
awt	25	0.00001745	1
xsl	23	0.00001606	1
jdbc	23	0.00001606	1
oo	20	0.00001396	1
unicode	20	0.00001396	1
ejb	20	0.00001396	1
overloading	19	0.00001327	1
servlet	18	0.00001257	1
steganography	17	0.00001187	1
serialization	16	0.00001117	1
isso	16	0.00001117	1
ecoop	16	0.00001117	1
odmg	16	0.00001117	1
schulmeyer	15	0.00001047	1

Table 4.7: Terms for *005 Computer programming, programs, data* with a confidence score of one.

classification purposes.

When using the frequency metric it was found that some of the more frequent terms were so common across different subjects that they could be considered stopwords. For example terms like “book” and “computer” occurred in many different subjects. It was found that terms that were frequent across many subjects were more likely to appear in a standard English dictionary. From this comes the concept that for a term to appear in a standard dictionary it is likely to be multi-purpose. It can then be shown that many terms that appear in a standard dictionary are of little use for classification purposes, which exposes another flaw in the method of randomly selecting query sample terms from a dictionary. Terms that are useful for classification are generally highly subject specific and can often only be recognised by people with a good knowledge of the subject.

The results presented in this research therefore only use the support and confidence method. The experimental results were obtained by using the top four terms for each Dewey Decimal code based on descending order of first confidence and then support where there were more than

twenty terms in the ontology. So the terms were first selected in descending order of confidence, and then within each confidence group they were sorted in descending order of support. It was also found that terms that occurred less than a relative percentage of times in the ontology were too rare for classification purposes and were not found in many of collections used in the experiments. This means that some subjects may have less than four terms, but this did not seem to affect the quality of the results. In addition, some subjects had no classification terms at all which could be rectified by adding more training data from other library catalogues.

In future research, a probability code can be assigned to each term pattern and subject relation. For example the term "C#" and the subject ".Net Programming Languages" could be given a 80% chance of being correct, while the same term with the subject "Operating Systems" could be give a 20% chance of being correct.

A single term-subject association rule is an implication of the form $term \Rightarrow subject$. The rule $term \Rightarrow subject$ holds in the set of all term occurrences in the ontology with confidence c if $c\%$ of documents in $ontology\ base$ that contain $term$ also contain $subject$. The rule $term \Rightarrow subject$ has support sup in the ontology base $ontology\ base$ if $sup\%$ of rules in $ontology\ base$ contain both $term$ and $subject$.

Algorithm 2 shows the process for generating the rule which specifies the confidence and support that a term belongs to a Dewey Decimal code. As only the top three levels of the taxonomy are used each Dewey Decimal code is truncated to three digits. The algorithm takes a set of raw training data imported from the library catalogue, and converts it into a set of terms and their associated support and confidence. The highest weighted terms for each subject can then be used for classification purposes. This method generated 1,437 classification terms from 458 subjects which were then used to classify each search engine.

4.4 Summary

This chapter gave an overview of ontologies, a formalisation of ontologies, and a method for automatically extracting classification rules from an ontology. This method is innovative because it removes the need for experts to manually create classification rules for each subject in

Algorithm 2 Generate term association table

Given a training set of library documents and their associated Dewey Decimal Codes.

Return a term association table containing the set of tuples <term,DDC,confidence,support>

```
1: Take training set and break into terms. Save each term and it's associated Dewey Decimal
   Code in the ontology.;
2: Select all terms from the ontology that occur more than once; (To avoid rare terms and
   spelling mistakes only terms that occur more than twice in the ontology are used.)
3: for Each term do
4:     Select the set of Dewey Decimal Codes that the term occurs with;
5:     Calculate the confidence and support for the set of Dewey Decimal Codes;
6:     for Each Dewey Decimal Code in the set do
7:         Write the term, the Dewey Decimal Code, the confidence and support for this code to
           the ontology's term association table;
8:     end for
9: end for
```

the taxonomy. It is also significant because the rules generated can easily be extended to other subjects and languages. The experiments in Chapter 7 show that the rules produced are of high quality and perform well across a wide variety of subjects.

Chapter 5

Resource Discovery for the Deep Web

This chapter describes the ontology based method used in this thesis to profile collections and search engines. A step-by-step overview of the method used in this thesis is given.

5.1 Ontology based Collection Selection

In this work world knowledge in the form of an ontology is used to help profile the collections. An ontology is trained with world knowledge and classification rules are extracted from it. The rules are then used to profile collections. Once the collections have been profiled, a query can be taken from a user and the best collections for that query can be returned.

In this research subjects are used to profile collections, as using a term histogram to describe each search engine is inefficient. Considering that the Oxford English Dictionary (Second Edition) contains 291,500 words [Pre05], it is possible that a large general purpose search engine could have at least several times that number of terms in its index due to the many specialist acronyms and technical terms that exist. This would necessitate sending a minimum of 291,500 queries to each search engine to assess its content distribution, and the same number of terms would be required to describe each engine. This research takes a different approach by using subjects to profile and describe collections. This means that large search engines can be compactly described with a small set of subject classification codes instead of a large set of terms. This is far more compact and robust than the standard collection selection method of describing collections with a term histogram [XC98, CLC95].

The following methods can also be used with no modification for large internet search engines, deep web search engines, and almost any text based database. The need for effective and efficient collection selection will keep growing as more and more search engines are added to the web, and as the size and subject coverage of web search engines continues to grow.

5.1.1 Method

There are three stages in the ontology-based collection selection method. In the first stage, the collections are profiled and the results are stored as metadata. In the second stage, a query is taken from the user and matched with the most relevant collections' collection description. In the third stage the collections are grouped together using a pattern matching method.

5.1.2 Stage 1

The following method is used for the first stage, profiling the collection:

1. Extract query probe terms from the ontology;
2. Query probe collection with the extracted terms;
3. Extract the results from the collection;
4. Convert collection results into taxonomy format; and
5. Store results as metadata;

Extract Query Sample Terms From The Ontology

For each subject in the Dewey Decimal System, the most distinguishing (or unique) terms are extracted from the ontology. The support and confidence of each term in each subject is calculated, and then the terms are ordered first in descending order of confidence, and then within each confidence value group they are sorted in descending order of support. The top n terms for each subject are then selected. Table 4.7 shows an example of the top terms for the subject *005 Computer programming, programs, data*.

Query Sample Collection With The Extracted Terms

In collection selection, *query sampling* [Cra01, CCD99] is commonly used to discover the contents of uncooperative collections. Query sampling involves sending a set of query terms to a collection and using the results to form conclusions about the collection's content.

The ontology-based collection selection method uses a *Lowest-Level-First* query sampling approach. The query sample terms from each subject node of the lowest level of the taxonomy are extracted. While it was difficult to decide how many classification terms to extract for each subject node, the use of more terms allows better results for search engines which have a wider but more shallow coverage of a subject. However these engines may not have as high quality results as ones that provide deeper results for part of a subject. The use of fewer terms would result in better results for search engines which have a deeper coverage of some aspects of a subject but poor results for engines which have a wider coverage of a subject. In the experiments presented in this thesis the best results were obtained by using the top four results from the highest confidence and support for each subject node.

The selected terms are then taken and sent to each collection one-by-one as query samples. For a deep web search engine, this is done by altering the URL and then sending this modified URL to the server. For example, for the "Microsoft Live" search engine, the URL for the query "servlet" looks like:

http://search.live.com/results.aspx?q=**servlet**&go=Search&mkt=en-gb&scope=&FORM=LIVSOP

To send another query to Microsoft Live, the "q=servlet" text is replaced with another term, such as "Java", and the URL will then look like:

http://search.live.com/results.aspx?q=**Java**&go=Search&mkt=en-gb&scope=&FORM=LIVSOP

This query is then submitted to the Microsoft server, which will then return a HTML page which contains the results.

Once the query sample terms have been sent to each search engine the results are extracted and saved[1]. The detailed set of steps in the query probing process are shown in Chapter 4.

Extract The Results From The Collection

[1] A HTML parser was used for all the search engines except Google and Yahoo, where their respective APIs were used

Depending on the type of collection being analysed, the format of the results from the query probe may vary widely. From a deep web search engine, the results will almost always be in HTML format, although some do use XML.

The following shows the result of sending the query sample “servlet” to the Microsoft Live search engine:

1. Header

```
<!DOCTYPE html PUBLIC ``-//W3C//DTD XHTML 1.0 Transitional//EN
''http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd``><html
xmlns=''http://www.w3.org/1999/xhtml``
xmlns:Web=''http://schemas.live.com/Web/`` lang=''en``
xml:lang=''en``><head><meta http-equiv=''content-type``
content=''text/html; charset=utf-8`` /><!-- TN:
F54DA6E0DB112215C00D4F57A2E90E05 --><title>Live Search:
servlet</title>''
```

...

2. Results Area

```
<div id=``content''><div id=``results_area'' class=``results_area
results_area_flank''><div class=``sc_rslth1''> <h2>Web results <span
id=``count''>1-10 of 6,620,000</span></h2><div>See also:<ul
class=``sc_scp'' id=``sch_scopes''><li><a
href=``/images/results.aspx?q=servlet&amp; FORM=BIRE''
gping=``&amp;POS=7&amp;CM=SBI&amp;CE=1&amp;CS=SCP&amp;
SR=1&amp;sample=0''><span>Images</span></a>, </li><li><a
href=``/news/results.aspx?q=servlet&amp; FORM=BNRE''
gping=``&amp;POS=8&amp;CM=SBN&amp;CE=1&amp;CS=SCP&amp;
SR=2&amp;sample=0''><span>News</span></a>, </li>
```

...

3. Footer

```
<script type=``text/javascript''>//<![CDATA[
RMS.RegisterResources(['\/sa\/3_10_0_146641\/
```

```
live2_c.css'],['\/sa\/3_10_0_146641\/live2_c.js']);
//]]></script></body></html>
```

The important text is in the **Results Area**. The actual number of results is "6,620,000". To extract this number from the page the page is stripped of HTML tags leaving something similar to:

```
Web results 1-10 of 6,620,000
```

A regular expression is then used to extract the final number "6,620,000" from the results.

```
Web results 1-[0-9,]{1,25} of [0-9,]{1,25}
```

This number is then stripped of commas and saved in the database along with the query probe term "servlet".

Table 5.1 shows the results of query probes on search engines for the subject *004 Data processing Computer science*. Each engine has a column (vector) to show the term frequencies. However due to Zipf's law, which states that the frequency of a term follows the power law function, the larger search engines are probably going to contain a larger variety and distribution of terms, which will then bias them to the ontology-based collection selection method which tends to use rarer terms. Terms can also be given weights of how important they are.

Table 5.2 shows some sample query probe classification terms generated from the ontology and their results from the search engines analysed.

Convert Collection Results Into Taxonomy Format

Once all the query samples terms for each Dewey Decimal code have been sent to the collection, and the results gathered, the resulting information is then converted into a Dewey Decimal format. For each Dewey Decimal code on the third level, the terms with highest support and confidence are extracted. For each Dewey Decimal code, the results of all the terms for that code are summed together. For example, if four terms from a subject are used to query probe a search engine, the results for each of the four terms will be added together and this result recorded as the result for this subject code. This number is then stored as the value for this Dewey Decimal code in this collection in a database.

Term	Search engine	DDC	Term hits
ccna	PubMed	004	0
ccna	Live	004	7170000
ccna	Wikipedia	004	581
ccna	Accoona	004	504617
ccna	FirstGov	004	28700
ccna	USDA	004	8
ccna	Commerce	004	1340
ccna	Yahoo	004	10100000
ccna	Google	004	8440000
ipv6	Live	004	8590000
ipv6	PubMed	004	0
ipv6	Wikipedia	004	3831
ipv6	Accoona	004	575518
ipv6	FirstGov	004	49900
ipv6	USDA	004	225
ipv6	Commerce	004	5030
ipv6	Yahoo	004	18200000
ipv6	Google	004	17800000
ethernet	PubMed	004	138
ethernet	Live	004	35100000
ethernet	Wikipedia	004	16331
ethernet	Accoona	004	3738573
ethernet	FirstGov	004	271000
ethernet	USDA	004	284
ethernet	Commerce	004	9590
ethernet	Yahoo	004	86600000
ethernet	Google	004	75800000
dreamweaver	PubMed	004	20
dreamweaver	Live	004	45000000
dreamweaver	Wikipedia	004	1791
dreamweaver	Accoona	004	1842846
dreamweaver	FirstGov	004	98600
dreamweaver	USDA	004	917
dreamweaver	Commerce	004	95
dreamweaver	Yahoo	004	48400000

Table 5.1: Sample query probe results for *004 Data processing Computer science*.

term	Google	Wikipedia	Live	Commerce	FirstGov	Accoona	PubMed	Agriculture	Yahoo
abraham	46100000	140581	31700000	6300	1410000	2797818	9002	1170	4020000
absenteeism	3410000	7731	1170000	126	180000	162259	1741	385	6920000
accelerators	6740000	136421	2140000	6300	130000	268283	639	262	73500000
accountability	41100000	704701	18700000	492000	13100000	3123154	4332	7290	37300000
accountants	60300000	704011	25300000	165000	1510000	2091464	948	1110	50900000
accreditation	30800000	96211	17000000	108000	3460000	2417532	5567	5760	21200000
acoustics	17000000	170821	8890000	22800	179000	1178833	1452	385	38400000
acronyms	19000000	87181	6190000	821000	2860000	1019187	651	2230	20700000
actionscript	26600000	1241	2590000	13	1140	711590	0	0	7200000
actuarial	9260000	6371	3450000	3880	879000	338042	2847	18200	20600000
additives	24300000	1787501	10600000	13100	444000	1225160	5777	7850	36300000
adobe	36300000	28291	12900000	441000	17000000	13498942	6623	111000	4270000
adoptees	3360000	4981	656000	48	30900	72452	192	2	31100000
adventure	19600000	425051	22200000	4290	2300000	16113541	1418	410	18700000
advertisers	11900000	321791	48800000	1060	111000	5608388	1107	144	5350000
aesop	4780000	5711	2290000	66	98700	257771	77	181	35900000
aesthetic	29500000	82341	13800000	4980	1040000	1871052	1365	2270	20800000
afghanistan	131000000	112531	68700000	22600	1960000	7962199	1093	6820	1510000

Table 5.2: A sample of the query probe results from the search engines for each of the terms selected from the ontology.

The following formula is used for each subject in each search engine (where k is the number of terms selected):

$$\sum_{i=1}^{k} count(term_i) \tag{5.1}$$

Algorithm 3 shows the method for calculating the Dewey Decimal subject classifications for each search engine. It generates a table which contains a list of search engines and their associated subject classifications:

Algorithm 3 Calculate third level D.D.C. subject classifications for each search engine

The inputs are the results from query probing each search engine with the classification terms selected from the ontology.

The output is a table of search engines and the frequencies of each D.D.C. subject code. This table can then be normalised to decrease the effect of search engine size on the results.

```
1: for Each search engine do
2:     for Each top-level subject from DDC code 000 to 999 do
3:         Calculate the result value for each subject node using equation 5.1;
4:         Write the subject node value to the result table;
5:     end for
6: end for
```

Table 5.3 shows a sample of the raw results of the third level of the hierarchy. These results have not been normalised so may be relative to the size of the search engine. Table 5.4 shows a sample of the normalised results of the third level of the hierarchy. The highest values have been highlighted. There are many more results which cannot be shown here due to space limitations.

The Dewey Decimal codes in the table stand for:

	Google	Wikipedia	Live	Commerce	FirstGov	Accoona	PubMed	Agriculture	Yahoo
003	70250000	269944	25760000	113620	504400	5120784	8015	1404	61100000
004	104950000	22534	95860000	16055	448200	6661554	158	1434	163300000
005	49970000	13134	786980000	7071	154215	2114320	143	159	47420000
006	53880000	238484	50160000	11543	202440	3212791	129	264	83850000
300	135500000	1903564	50040000	158568	2615900	5981843	90787	14656	133490000
301	10810000	9604	3010000	396	67196	311475	653	192	6790000
302	19685000	213324	6343000	68	448250	889083	1847	178	22714000
303	31340000	4273324	13986000	1179	612500	3494069	92	82	87560000
304	83860000	201614	37020000	55740	1944000	4737067	80384	13227	95620000
305	44330000	31864	71000000	61	198604	1936717	6129	241	34560000
306	3868000	6226	828628000	315	183000	210315	7097	122	4301000
307	55160000	105034	9487000	408794	6162100	2220138	4768	9776	58812000

Table 5.3: Raw results from highest confidence and support query probes on the third level of the hierarchy.

	Google	Wikipedia	Live	Commerce	FirstGov	Accoona	PubMed	Agriculture	Yahoo
003	0.0102	**0.0119**	0.0027	0.0027	0.0018	0.0074	0.0040	0.0011	0.0055
004	**0.0152**	0.0010	0.0100	0.0004	0.0016	0.0096	0.0001	0.0011	0.0148
005	0.0073	0.0006	**0.0818**	0.0002	0.0005	0.0031	0.0001	0.0001	0.0043
006	0.0078	**0.0105**	0.0052	0.0003	0.0007	0.0046	0.0001	0.0002	0.0076
300	0.0197	**0.0837**	0.0052	0.0038	0.0092	0.0086	0.0452	0.0115	0.0121
301	**0.0016**	0.0004	0.0003	0.0000	0.0002	0.0004	0.0003	0.0002	0.0006
302	0.0029	**0.0094**	0.0007	0.0000	0.0016	0.0013	0.0009	0.0001	0.0021
303	0.0046	**0.1878**	0.0015	0.0000	0.0022	0.0050	0.0000	0.0001	0.0080
304	**0.0122**	0.0089	0.0038	0.0013	0.0069	0.0068	0.0400	0.0104	0.0087
305	**0.0064**	0.0014	0.0074	0.0000	0.0007	0.0028	0.0031	0.0002	0.0031
306	0.0006	0.0003	**0.0861**	0.0000	0.0006	0.0003	0.0035	0.0001	0.0004
307	0.0080	0.0046	0.0010	0.0098	**0.0218**	0.0032	0.0024	0.0077	0.0053

Table 5.4: Normalised results from highest confidence and support query probes on the third level of the hierarchy.

003 Systems

004 Data processing Computer science

005 Computer programming, programs, data

006 Special computer methods

... ...

300 Social sciences

301 Sociology & anthropology

302 Social interaction

303 Social processes

304 Factors affecting social behavior

305 Social groups

306 Culture & institutions

307 Communities

5.1.2.1 Store Results As Metadata

The above process results in database table which contains all the collections, and the value for each Dewey Decimal code in each collection. This table is called a Collection Description and is a profile of each collection, which can then be used to find the subject matter of each search engine for each Dewey Decimal code.

5.1.3 Stage 2

Once the Collection Description has been generated it can then be used to return the best collection or set of collections for an information need. Once a query has been submitted by the user it is then classified into a Dewey Decimal Code. So for example the query “Haskell function” might be classified into 006 and 005.

Each query is mapped to a set of subjects. The intersect of the query subject codes and collection subject codes is taken and a weighted set of relevant collections is returned to the user.

The set of steps involved is:

1. Accept a query from the user
2. Convert the query into a set of subject codes
3. Look up collection metadata to find matching collection(s)
4. Return the collections to the user

Accept A Query From The User

In the first step the user enters a query into an engine interface. The interface accepts boolean queries using AND, AND NOT, and OR. Where a query contained multiple terms the AND operator between the terms did not return any results the OR operator was used between the terms instead. The search engine also supports sorting by date, title, or relevance. For this thesis all results were sorted by relevance.

Convert The Query Into A Set Of Subject Codes

Once the query has been submitted, it is then sent to the Q.U.T. library catalogue where it is transformed into a set of Dewey Decimal Codes. The Dewey Decimal codes for the most relevant documents are selected and extracted. For the third level of the ontology all codes were stripped to three digits.

Look Up Collection Metadata To Find Matching Collection(S)

The collection description is then searched for the subject codes and the most similar collections selected. If the query contained more than one subject, a disambiguation can be performed by presenting the user with a list of possible subjects, which the user can then then select to narrow down the search.

Return The Collections To The User

The most relevant collections in the collection description can then returned to the user in a page with a list of links to collections and the subjects that they contain. Because the taxonomy is multi-level, the user can then browse further down the taxonomy or move upwards to get a high level view of the subjects contained in the taxonomy.

5.1.4 Stage 3

Singular value decomposition (SVD) [Kin03] is performed on the data from Table 5.4 to add further metadata to the collection description. Singular value decomposition is a matrix factorisation and dimension reduction method that has many uses such as information retrieval; time series analysis; stock market analysis; and pattern matching. In this research, SVD is used for determining how related the subject matter of each collection is to each of the other collections. The advantage of SVD is that it can quickly show the latent relationship patterns between the collections, and this method is able to identify patterns in the results which are difficult for a human to see. The result of the SVD calculation is a "collection-collection" matrix with values between 1.0 and -1.0. In this case, a score of one means an exact match of content between the search engines and a score of zero means that there is no overlap of content between the collections. Table 5.5 shows some example results of SVD.

5.2 Summary

This chapter gave a step-by-step overall picture of the ontology based collection selection method. Each of the three steps involved in the process were outlined. In the first stage, the collections are profiled and the results are stored as metadata. In the second stage, a query is taken from the user and matched with the most relevant collection descriptions. In the third stage the collections are clustered together by subject using an advanced pattern matching method. Sample results from the collection selection method were shown.

	Google	Wikipedia	Live	Commerce	FirstGov	Accoona	PubMed	Agriculture	Yahoo
Google	1.0000								
Wikipedia	0.4821	1.0000							
Live	0.6619	0.5080	1.0000						
Commerce	0.1847	0.3009	0.3240	1.0000					
FirstGov	0.4033	0.4953	0.4678	0.8360	1.0000				
Accoona	0.7412	0.4572	0.6741	0.3957	0.5670	1.0000			
PubMed	0.3354	0.2600	0.2173	0.1118	0.2304	0.2614	1.0000		
Agriculture	0.3223	0.3816	0.3032	0.3116	0.5377	0.3518	0.3099	1.0000	
Yahoo	0.9007	0.5440	0.8271	0.3642	0.5894	0.8185	0.3131	0.4068	1.0000

Table 5.5: Singular value decomposition results from highest confidence and support query probes. These indicate how closely related the subject content of each search engine is to the subject content of each of the other search engines.

Chapter 6

Search Engine Content Analysis

We are living in an "information age" where knowledge is growing at an enormous rate. Search engines are of vital importance because they enable us to almost instantly find information about almost any subject imaginable. Yet very few people are allowed direct access to the secret inner workings of the largest search engines, so little is known about what they contain and the algorithms they use to return information. This chapter describes the problems associated with analysing search engine content and gives an overview of the method used to analyse the content of the largest search engines.

6.1 Definition of Search Engine Content Analysis

Search Engine Content Analysis is the analysis and mapping of a search engine to a set of subjects. It is expected that Search Engine Content Analysis will empower users to find the best search engine for a specific information need. For example, many novice research students may find that there are hundreds of specialist and academic search engines available through their university library and that finding the best search engine to use for their research task is a daunting process of trial and error. Search Engine Content Analysis will enable the profiling of a large number of search engines across a large set of specialised subjects. This research uses many hundreds of subjects, compared to earlier work that only used a small set of highly specialised subjects.

Search Engine Content Analysis uses a set of steps to take a large set of raw data and convert it into a set of specialised classification rules for each subject. The rules are then used to profile the subject distribution of each search engine. Once the search engines have been profiled the profile is then used to match an information need to a set of relevant search engines.

6.2 Objectives of Search Engine Content Analysis

The main objective of this research is to select the best search engine(s) for an information need. This is a difficult task as commercial search engines tend to be very large and cover a massive range of subjects which require a great amount of expert knowledge to classify. Another problem with analysing the content of search engines is that it is important to classify the search engine using as few query probes as possible, and as few retrieval hits as possible must be used to classify each search engine. It must be possible to classify an engine using as few requests and as little bandwidth as possible, because the search engines do not like others using their bandwidth and resources, as well as the fact that there are now so many specialist search engines that the time it takes to analyse them all using a large set of terms would be prohibitive.

There are two approaches to analysing search engines. In the first, a resource description of each search engine is built at the system initialisation. In the second, the queries are broadcast at runtime. This system uses the first method, as when a user enters a query, broadcasting the query to all the search engines on the web is far too expensive in terms of resource use and bandwidth. Instead of broadcasting a query to all available search engines for each query a resource description of the subject distribution and content of each search engine must be built previously. A search engine can be found for an information need quickly and efficiently without the need for any large bandwidth usage. The resource description can also be periodically updated to keep an accurate profile of each search engine. As the IntelliOnto method of generating a resource description is very efficient, using less than four thousand query samples per engine, a collection profile can be built over a long period of time with little impact on the search engine server(s).

The main contributions of this research can be described as follows:

- the ability to create a large ontology for representation of world knowledge
- the method of selecting query probe terms from the ontology
- the analysis of collections and large search engines using an ontology
- a method for ontology-based collection selection which does not rely on estimation of collection sizes

The collections and search engines discussed in this research are web based, with the only interface being a HTML form where a query is entered, and a set of HTML web pages which comprise the results of the query.

6.3 Ontology Based Algorithms

The ontology-based collection selection and search engine content analysis algorithms are exactly the same, allowing this system to be used for a wide variety of information sources including search engines, email servers, and text databases. For the remainder of the chapter, each algorithm in the search engine content analysis process is given, along with an example of it's use.

The Search Engine Content Analysis process is described as follows:

First generate the base of the ontology using Algorithm 4, where the term frequency for each $< DDC, term >$ pair is calculated. This is the lowest level of the ontology, and this data will later be mined to extract classification rules and terms. The complexity of this algorithm is linear as only one pass of the data is needed to create the association table. Table 6.1 shows some example documents and their classifications which will be used throughout this chapter. There are over four hundred thousand documents in the library catalogue dataset but for this example only six documents are used.

So to generate the base of the ontology from Table 6.1 we split each document into terms and add each term to the ontology along with it's Dewey Decimal classification code. Table 6.2 shows the raw ontology base where term frequency is the number of times that the term occurs in the DDC subject.

Document DDC and Subject Name	Example Document
004 - Data processing Computer science	Oracle Development
004 - Data processing Computer science	Oracle Data Mining Applications Comparison
005 - Computer programming, programs, data	The C# Applications
005 - Computer programming, programs, data	The C# Programming Language
111 - Ontology	The Kant and Heidegger Comparison
111 - Ontology	Kant, Heidegger and the Ontology

Table 6.1: Example Dewey Decimal Codes and their subject description, along with example documents

DDC	term	term frequency
004	oracle	2
004	development	1
004	data	1
004	mining	1
004	applications	1
004	comparison	1
005	the	2
005	C#	2
005	applications	1
005	programming	1
005	language	1
111	the	1
111	kant	2
111	ontology	1
111	and	1
111	heidegger	2
111	comparison	1

Table 6.2: Example Term Association Table including stopwords

term	DDC	term frequency
004	oracle	2
004	data	1
004	mining	1
005	C#	2
005	programming	1
005	language	1
111	kant	2
111	heidegger	2
111	ontology	1

Table 6.3: Example Term Association Table excluding stopwords

Secondly, after Table 6.2 is generated, the Inverse Subject Frequency method is used to identify terms that occur in too many different subjects to be of use. These terms are then removed from the ontology in order to reduce it's size. This leaves Table 6.3.

Algorithm 4 Generate the ontology base

Inputs: $DocumentSet$ - a set of library documents, each with a Dewey Decimal Code classification.

Outputs: $AssociationTable$ - a table of terms, their DDC codes, and their frequencies in the form of $< DDC, term, termcount >$

LET $AssociationTable = \emptyset$;
for each $document \in DocumentSet$ **do**
  let DDC be the subject classification code for the $document$;
  for each $< term, DDC > \in document$ **do**
    if the pair $< term, DDC >$ does not exist in the association table **then**
      add the term and the DDC to the association table in the form of a tuple $< DDC, term, termcount >$ where termcount is 1;
    else if if the pair $< term, DDC >$ exists in this table **then**
      increment the $termcount$ for the pair $< term, DDC >$ in the Association Table tuple $< DDC, term, termcount >$;
    end if
  end for
end for

In Algorithm 1, the classification rules are generated from the ontology base. These rules help to identify terms which best represent each subject. These are terms that occur frequently within a subjects, and occur rarely outside the subject. These terms are often highly specific and difficult for non-experts to identify. This algorithm is the most computationally expensive as the

DDC	term	term frequency in this DDC
004	oracle	2
005	C#	2
111	kant	2
111	heidegger	2

Table 6.4: $SubjectAssociationTable$ - Terms with highest support and confidence

entire ontology needs to be searched for each term encountered. Table 6.4 shows the terms with highest support and confidence. These are the terms that are the best for classification purposes. They usually occur only within one subject, and they occur frequently within that subject. For a definition of support and confidence see Chapter 4.

In Algorithm 5 the content of the search engines are analysed using the classification rules generated in Algorithm 1. This algorithm takes a set of classification terms from the ontology, and query probes each search engine using the classification terms. The results from each of the search engines are then analysed and ranked for each DDC subject. The complexity of Algorithm 5 is $O(n)$ as it can be performed with linear passes. However this algorithm may take some time to execute as only a small number of terms can be sent to each search engine every minute and only a small number or results can be extracted from each search engine every minute, otherwise the search engine server(s) may be overloaded. This is of importance to the smaller specialist search engines which operate with limited funds and server processing power, which frequently includes many of the smaller, specialised search engines such as CiteSeer and PubMed.

To do this we now need to wrap each classification query probe term in a HTML request. For example in Search Engine 1, the HTML request might be:

```
http://www.searchengine1.com/search?source=ig&hl=en&rlz=&=&q=oracle
```

In this example this would result in the following data (example result data taken from MSN Live):

1. Header

```
<!DOCTYPE html PUBLIC ``-//W3C//DTD XHTML 1.0 Transitional//EN
''http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd``><html
```

```
xmlns=''http://www.w3.org/1999/xhtml``
xmlns:Web=''http://schemas.live.com/Web/`` lang=''en``
xml:lang=''en``><head><meta http-equiv=''content-type``
content=''text/html; charset=utf-8`` /><!-- TN:
F54DA6E0DB112215C00D4F57A2E90E05 --><title>Live Search:
servlet</title>''

...


<div id=``content''><div id=``results_area'' class=``results_area
results_area_flank''><div class=``sc_rslth1''> <h2>Web results <span
id=``count''>1-10 of 100</span></h2><div>See also:<ul class=``sc_scp''
id=``sch_scopes''><li><a href=``/images/results.aspx?q=servlet&amp;
FORM=BIRE'' gping=``&amp;POS=7&amp;CM=SBI&amp;CE=1&amp;CS=SCP&amp;
SR=1&amp;sample=0''><span>Images</span></a>, </li><li><a
href=``/news/results.aspx?q=servlet&amp; FORM=BNRE''
gping=``&amp;POS=8&amp;CM=SBN&amp;CE=1&amp;CS=SCP&amp;
SR=2&amp;sample=0''><span>News</span></a>, </li>

...

<script type=``text/javascript''>//<![CDATA[
RMS.RegisterResources(['\/sa\/3_10_0_146641\/
live2_c.css'],['\/sa\/3_10_0_146641\/live2_c.js']);
//]]></script></body></html>
```

A regular expression is then used to extract the data from the page, which results in the number '100' being extracted.

Table 6.5 shows the results of sending each of the terms to each of the three example search engines:

Now we sum the DDC code values together to produce Table 6.6

Table 6.7 shows the final normalised results.

DDC	term	Search Engine 1	Search Engine 2	Search Engine 3
004	oracle	100	23	170
005	C#	120	13	193
111	kant	3	30	2
111	heidegger	4	43	5

Table 6.5: Example results of query probing search engines

DDC	Search Engine 1	Search Engine 2	Search Engine 3
004	100	23	170
005	120	13	193
111	7	73	7

Table 6.6: Search engine classifications grouped by DDC codes

From this table it is possible to see that Search Engine 3 is best for '004 Data processing and Computer science' , Search Engine 1 is best for '005 Computer programming, programs, data' , and Search Engine 2 is best for '111 Ontology'.

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} \tag{6.1}$$

Equation 6.1 shows the Frobenius norm(or vector norm) that is used to normalise the results of Algorithm 5.

In Algorithm 6 the user query is matched to the most relevant set of search engines. As all the heavy computational processing has been done by Algorithm 1 and the time consuming search engine analysis has been done by Algorithm 5 this algorithm can quickly return it's results to the user. The algorithm complexity is $O(n)$.

Using the above example, the user query might be 'Immanuel Kant'. These terms are then retrieved from the ontology. The ontology classification algorithm then decides that the term 'kant' occurs frequently in the subject '111 Ontology' which results in a classification of the

DDC	Search Engine 1	Search Engine 2	Search Engine 3
004	0.6395	0.2963	**0.6607**
005	**0.7675**	0.1675	0.7501
111	0.0448	**0.9403**	0.0272

Table 6.7: $SearchEngineRankingTable$ - Normalised search engine classifications grouped by DDC codes

Algorithm 5 Search Engine Content Analysis for one search engine

Input: Given $SubjectAssociationTable$, which is a set of subject codes, each term that occurs in each subject, and the support and confidence values for each term.

Output: Return a table $SearchEngineRankingTable$ where search engines are the columns and DDC codes are the rows, and each cell contains the normalised relevance number of the search engine for that DDC code.

1: **for** each $ddc \in S$ **do**
2: Take the terms with highest support and confidence values from the $SubjectAssociationTable$ associated with DDC, let the set be $\{[t_1, t_2, ...t_n]\}$;
3: Send the set of terms to the search engine one-by-one and extract the number of results for each term from the search engine results and sum the results together. Let the search result be $\{(t_1, f_1), (t_2, f_2), ..(t_n, f_n)\}$ where f is the frequency of the term in the search engine.;
4: Use the $\{(t_1, f_1), (t_2, f_2), ..(t_n, f_n)\}$ to rank the collections in relation to the DDC code, where $Rank(< SE, DDC >) = \sum_{i=1}^{n} f_i]$;
5: normalise the collection ranks $< SE, DDC >$ by using the Frobenius norm;
6: **end for**

query subject to '111 Ontology'. Looking up the Table 6.7, the DDC '111 Ontology' is then mapped to Search Engine 2. This search engine is then returned to the user as being the best search engine to use to find information on the subject of Ontologies from. By using this term-subject mapping method we overcome the problems of synonymy and polysemy. Also it has another benefit in that it is possible to return relevant results which do not contain the words 'Immanuel Kant'. For example, the user might be interested in the writings of the philosopher Martin Heidegger, a major contributor to the subject '111 Ontology', yet be unable to recall his name. Using the standard information retrieval method of only matching query terms with other terms, relevant results may be excluded from the result set because they do not contain the terms 'Immanuel Kant'. However using the ontology based subject retrieval method given in this thesis important related results are returned, making it easier for users to find information related to what they are searching for even if the information does not contain their query terms.

Figure 6.1 and Figure 6.2 show an example of the Q.U.T. Library search interface and results. These results are screen scraped in order to extract the DDC codes for each user query.

Algorithm 6 Return the best set of search engines for a user query

Input 1: $SearchEngineRankingTable$ for each search engine where search engines are the columns and DDC codes are the rows, and each cell contains the normalised relevance number of the search engine for that DDC code
Input 2: a query from a user.

Return: an ordered set of search engines, with a relevance value for each search engine in the range 0.00 to 1.00. The relevance value is the relevance of the user query q to the search engine SE.

1: Accept a query q from the user. Assume $q = [t_1, t_2, .., t_n]$;
2: Map each query into a set of Dewey Decimal codes using the data from the ontology, such as $q \rightarrow [DDC_1, DDC_2, ...DDC_n]$. This is done using the current library search engine.;
3: Using the set of DDC codes generated for each query, find the search engine(s) which score the highest for these DDC codes in Algorithm 5;
4: SVD can then be used to group similar search engines based on content. See details of SDV in Appendix B;

Figure 6.1: QUT Library Search Interface

Figure 6.2: QUT Library Search Results

6.4 Summary

This chapter gave an introduction to the objectives of the search engine content analysis method used in this thesis. A series of major contributions were given, followed by a listing of the main algorithms used in this thesis. Each algorithm was accompanied by an example of the algorithm in use. Although the examples are simple, the real search engine content analysis method deals with many hundreds of subjects, hundreds of thousands of documents, and uses a large amount of processing power to generate the classification rules for each subject, followed by a time consuming profiling of each search engine. In the next chapter this method will be compared to the current state of the art collection selection method.

Chapter 7

Evaluations

The ontology-based collection selection method is compared to ReDDE, the current state of the art collection selection method, and the results are given in this chapter. The ontology-based collection selection method is then used for search engine content analysis and the results across the third level and first levels are given. An evaluation of the Singular Value Decomposition is then given.

7.1 Experiments for Collection Selection

In the previous chapters the IntelliOnto automated ontology learning method was presented. It uses a taxonomic ontology based approach, and it also covers a large range of domains. In this section the method is compared to ReDDE, which relies on estimates of collection size to rank collections. The experiments are conducted on the standard TREC collection selection testbed in a controlled setting with relevance judgements from human experts.

7.1.1 Data

For the experiments, four well-known collection selection testbeds were derived from the TREC Tipster collection; Trec123-100col-bysource, Trec123-2ldb-60col, Trec123-AP-WSJ-60col, and Trec123-FR-DOE-81col [SC03].

Trec123-100col-bysource: Taken from TREC CD's 1, 2, and 3. Partitioned into 100 somewhat heterogeneous databases of the same average size which are grouped by publication date

Collection	Num of documents (x 1000)	Size (MB)
LDB1	231.3	665
LDB2	199.7	667
Apall	242.9	764
WSJall	173.3	533
Frall	45.8	492
DOEall	226.1	194

Table 7.1: Statistics for the large folded databases. Taken from Si et al. [SC03]

and source. This collection is 3.2GB in size, contains a minimum of 700 documents in its smallest database, a maximum of 39,700 documents in its largest database, and an average of 10,800 documents across all its databases. Its minimum database size is 28MB, and its maximum database size is 42MB, with an average database size of 32MB.

Trec123-2ldb-60col ("representative"): Trec123-100col-bysource is sorted alphabetically and forty of its databases are iteratively folded into two large databases known as LDB1 and LDB2 using every fifth database in the sorted list (See Table 7.1). LDB1 and LDB2 are roughly 20 times larger than the other databases but have a similar density of relevant documents when compared to the other databases.

Trec123-AP-WSJ-60col ("relevant"): Trec123-100col-bysource is taken and the 16 Wall Street Journal and 24 Associated Press databases are each folded into two large databases, WSJall and APall (See Table 7.1). The databases are large when compared to the unfolded databases, and contain a much higher density of relevant documents than the unfolded databases.

Trec123-FR-DOE-81col ("nonrelevant"): Trec123-100col-bysource is taken and the 6 Department of Energy and 13 Federal Register collections are folded into two databases DOEall and FRall (See Table 7.1). The databases are large when compared to the unfolded databases; however they contain few relevant documents when compared to the unfolded databases.

7.1.2 Parameters and measures

For the queries, the TREC Topic Queries 51-100 are taken from the Tipster collection. The TREC Topic Queries are included in the query samples. A comparison of sending the actual TREC Topic Queries as both a phrase and as a single term was performed and found that the

queries performed better as a phrase than as single terms.

To convert a query into a Dewey Decimal code the following process is used:

1. Extract the query phrase
2. Send the query phrase to the Q.U.T. library catalogue search engine using the sort by relevance option and setting terms to "OR"
3. Select the top documents from the library catalogue results page.
4. Extract the Dewey Decimal code for each of the documents from the results page and save as the subjects most relevant to this query phrase.

For example "South African Sanctions" translated to International Economics(337), "Satellite Launch Contracts" translated to Applied Physics (628), and "Insider Trading" translated to Private Law(346). Note that this search engine is not optimal and better results could be obtained using an improved search engine. Also note that it is possible for a query phrase to have more than one subject, but this has not been exploited in this work. A problem encountered with the TREC Topic Queries 51-100 was that some were U.S. centric, which lead to problems with the Q.U.T. Library dataset as it was Australian centric. Better results may have been attained if the ontology included more U.S. related data.

The method for evaluating each collection for each query is shown in Algorithm 7:

The experiments evaluating the method were performed on four testbeds. The relevance judgements for each of the TREC Topic Queries 51-100 were taken from TREC website, the file names were qrels.51-100.disk1.disk2.parts1-5.tar.gz and qrels.51-100.disk3.parts1-5.tar.gz. From these the "ideal" baseline was computed; for each query, collections were ranked on decreasing order of the number of relevant documents they contain. The ideal calculated a baseline for each collection using these relevance judgements. To shed light on the question of collection size estimation, the ReDDE system was used as it employs a well-motivated collection ranking

Algorithm 7 Collection Ranking Method

Given a set of TREC Topic Queries 51-100, an ontology, and a set of electronic collections

Return a ranking score for each Topic Query for each collection

1: Extract $< title >$ queries from TREC Topic Queries 51-100. The average query size was three terms;
2: Convert each query into third-level Dewey Decimal codes using the Q.U.T. Library catalogue search engine;
3: Convert each Dewey Decimal code into a set of query sample terms by taking the terms with highest support and confidence values for each Dewey Decimal code from the ontology. As this research is exploratory and little is known about how information is distributed the number of terms selected was found by performing a set of experiments;
4: Send each set of query sample terms to each of the collections one-by-one using the Zettair [BCC+04] search engine. (Zettair is a compact open source search engine from the R.M.I.T. University, which supports both TREC and HTML formats);
5: Extract the number of results for each query sample term from the Zettair results;
6: Sum the results from Zettair together and use them to rank each collection;

algorithm, whereby collection size estimation is a crucial feature. The method used for ranking the results of the experiments is:

$$R_k = \frac{\sum_{i=1}^{k} E_i}{\sum_{i=1}^{k} B_i} \tag{7.1}$$

Where B is the baseline ranking (in this case the TREC relevance judgements), and E is what the collection selection method selected for that position. Bi and Ei are the count of relevant documents at position i. The greater the value R at position k is, the more accurate the collection selection method E.

7.1.3 Results

For each of the graphs in this thesis the x-axis and y-axis are defined by Figure 1.1.

The top 20 collections for each testbed were selected. Figure 7.1 shows 2LDB-60col, Figure 7.2 shows 100col-bysource, Figure 7.3 shows APWSJ-60col, and Figure 7.4 shows FR-DOE-81col. The method performed best on AP-WSJ-60col ("relevant"), where the folded databases were large and had a high concentration of relevant documents, and worst on FR-

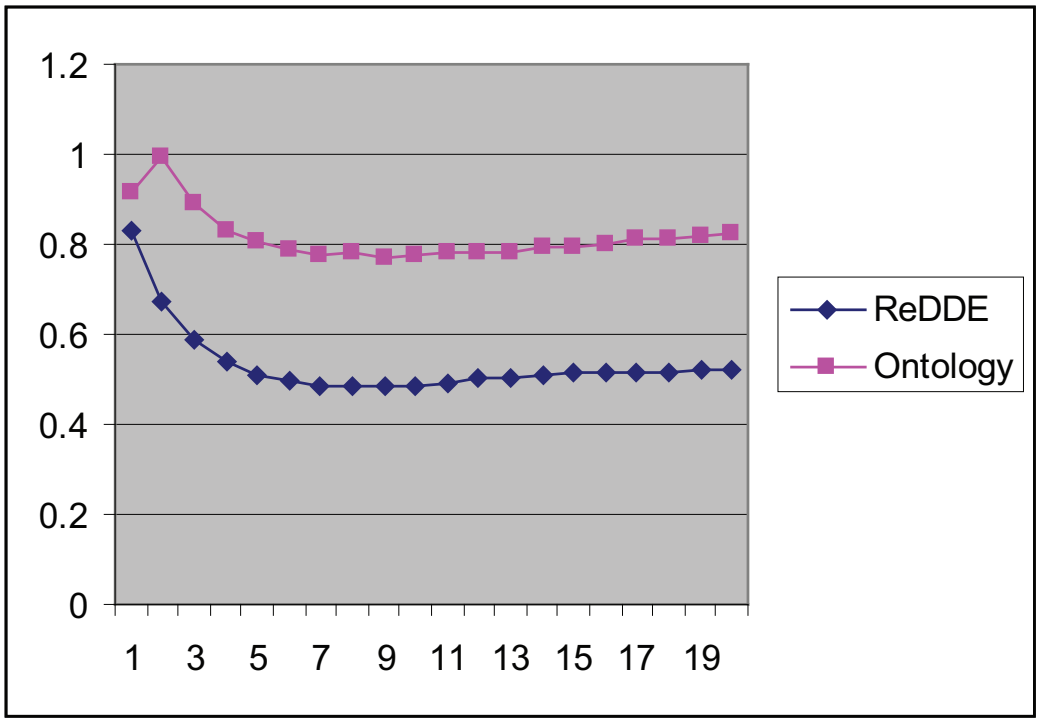


Figure 7.1: 2LDB-60col. The y-axis shows the R value, and the x-axis shows the position.

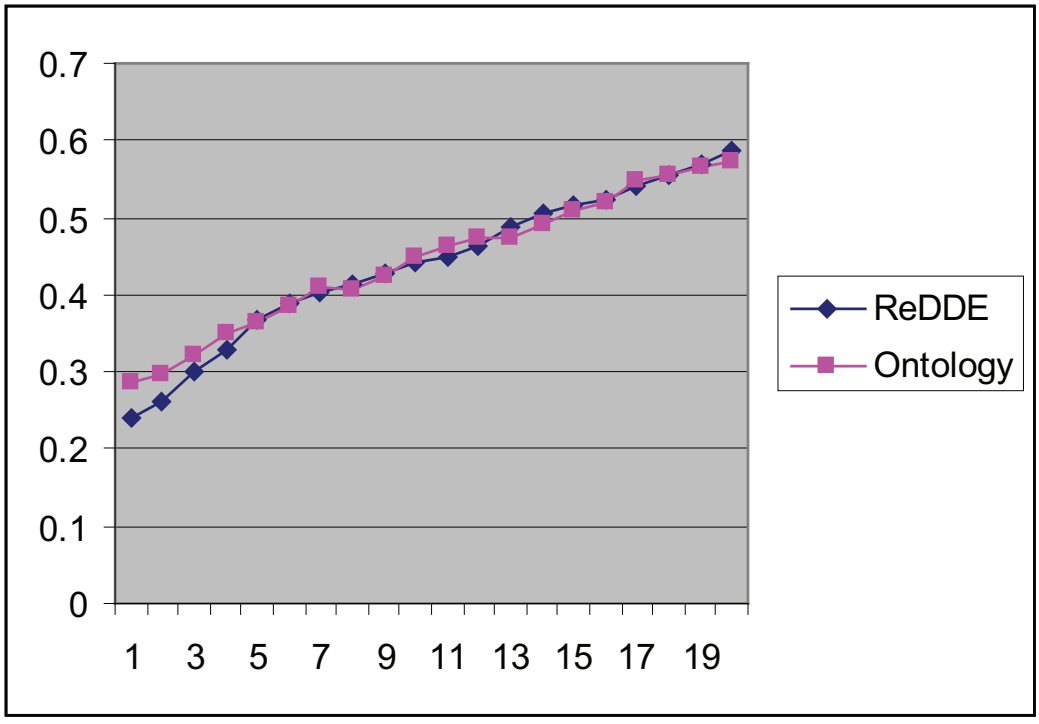


Figure 7.2: 100col-bysource. The y-axis shows the R value, and the x-axis shows the position.

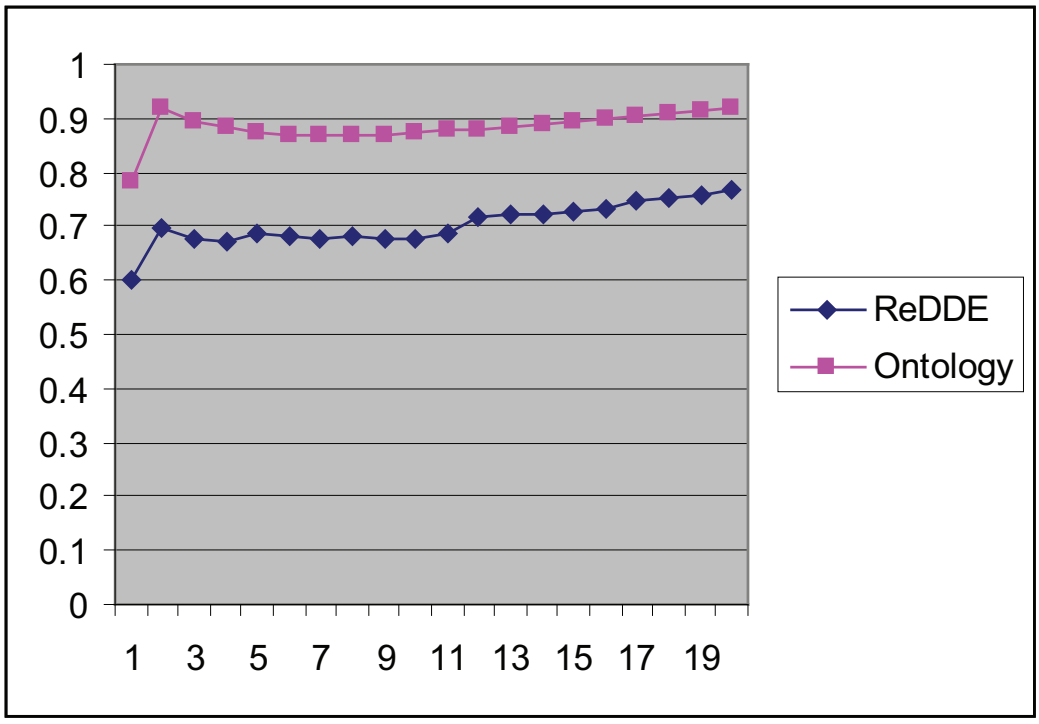


Figure 7.3: APWSJ-60col. The y-axis shows the R value, and the x-axis shows the position.

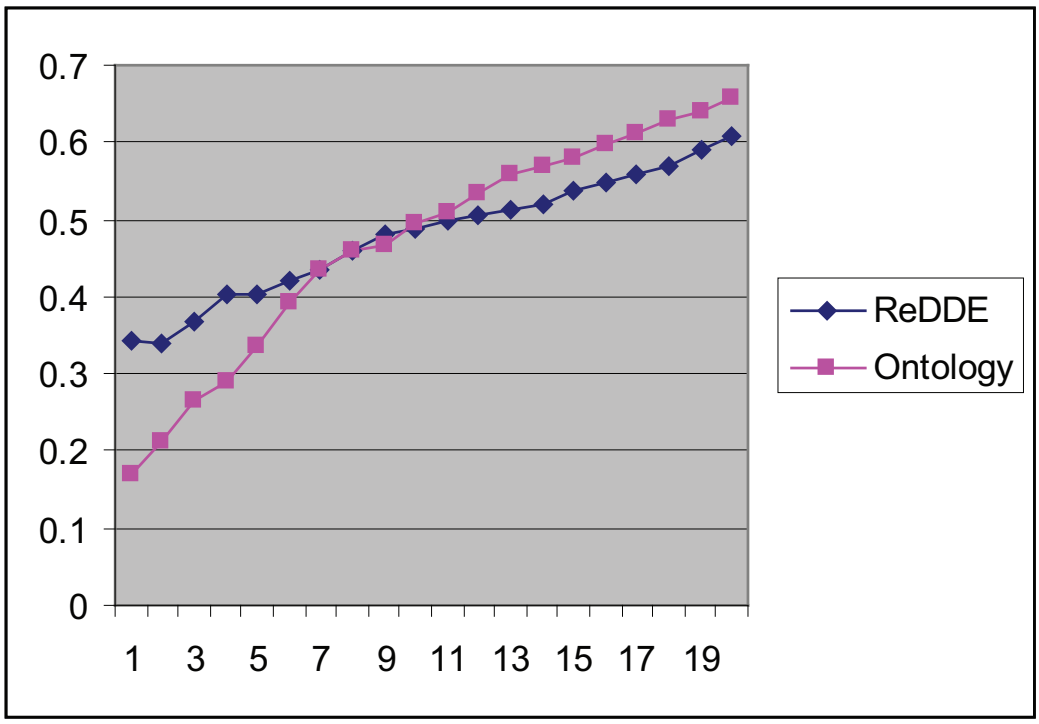


Figure 7.4: FR-DOE-81col. The y-axis shows the R value, and the x-axis shows the position.

Collection	P-Value
2LDB-60col	6.22798×10^{-17}
100col-bysource	0.083746572
APWSJ-60col	7.05735×10^{-20}
FR-DOE-81col	0.289978

Table 7.2: Outcomes of T-Test on Experimental Results

DOE-81col ("non-relevant") where the folded databases were smaller and had a low concentration of relevant documents. This is because the query sample terms selected using the method are highly subject specific, which makes them more suitable for selecting collections with more high quality subject specific information, and less suitable for collections containing more general information. Collections containing more general information often use less specialist terms and acronyms when compared to collections containing more technical information, and thus are harder to classify because the terms they contain often are more multi-purpose, occurring in many different subjects. There was a noticeable favouring of large collections in the experimental results, which means that smaller collections were biased against. However there was also a favouring of collections which had high concentrations of quality data which gave this method an advantage against methods which only used collection size estimation. It is expected that if phrases were mined from the ontology it would be possible to more accurately classify collections containing more general information because when multi-purpose terms are used in conjunction they often refer more explicitly to a specific subject or small set of subjects.

7.1.4 Measuring the Significance of the Results

A T-Test was performed on the results of the previous experiments in order to find out how significant they were. The T-Test calculates the Null Hypothesis, which is the suggestion that there is no relation between two different models. The t-test measure of the two sets of data results in a P-Value, where a p-value less than 0.05 shows substantial significance in the compared models, a p-value in the range 0.05 to 0.1 shows some significance, and a p-value greater than 0.1 shows no significance.

Table 7.2 shows the resulting p-values. The t-test showed that the experimental results from APWSJ-60col ($\times 10^{-20}$) and 2LDB-60col ($\times 10^{-17}$) were highly significant, the results

	Single Queries		Pool Based		Random walk		Multiple queries		Ontology	
Collection	**q**	**d**	**q**	**d**	**q**	**d**	**q**	**d**	**q**	**d**
calendar	1	0	1	1	1000	1000	34	0	5	0
zsh-list	1	0	7	2	1123	1000	5	0	5	0
procmail	1	0	8	2	1137	1000	6	0	5	0
email	1	0	8	2	1173	1000	5	0	5	0
WSJ	1	0	25	4	1161	1000	8	0	5	0
.GOV	2	0	11	1	2140	1000	23	0	5	0

Table 7.3: Estimated cost in queries "q" and document downloads "d", per document sampled, for each query sample method. Taken from Thomas [TH07]

from 100col-bysource were of some significance, and the experimental results from FR-DOE-81col showed no significance. This t-test result provides evidence that the results from the APWSJ-60col, 2LDB-60col, and 100col-bysource experiments are of significant value and can be viewed as important. The t-test setup used one tail with paired results.

7.1.5 Limitations

Zipfs law also shows that the larger a corpus is, the more likely it is to have a wider distribution of terms, which for this method explains the higher ranking for larger collections as some of the query sampling terms are highly subject specific, rarely occurring outside of a specialist domain. This query sampling method also favours larger collections.

It was also found that the main reason FR-DOE-81col performed poorly in the initial seven results was that there was no direct analysis of the documents returned from each of the collections. The results could be improved by combining analysis of the result documents with this method of using the number of results returned.

The ontology-based collection selection method was very efficient when compared to other high performing collection selection methods. Table 7.3 shows the estimated cost in queries and document downloads for four of the most popular query sample methods as calculated by Thomas [TH07] in his evaluation of query sample methods. Compared to the best query sample methods "Random Walk" and "Multiple Queries" the ontology-based collection selection method is more efficient as it only uses five queries (where one query is the title and other queries consist of one term only) and zero document downloads for each collection. It must be

noted that to fully profile a collection on the third level of the Dewey Decimal hierarchy across all 1,000 subjects would take 4,000 terms. This could be reduced to 400 terms by simply using the second level of the Dewey Decimal hierarchy and reducing the number of meta-subjects covered to 100, which would cause a drop in precision and an increase in recall. This reduced set would still cover more subjects than most current collection selection research.

The TREC collection selection testbed is limited because it does not simulate collection overlap, a common occurrence in internet collections. Collection overlap can cause large numbers of similar collections to be returned and a loss of effectiveness [SZ07].

The ontology-based collection selection method is limited because it relies too heavily on the results returned by the library catalogue search engine, where a miss-classification of the subject of query can have a large effect on the quality of the results returned. This could be improved by implementing a better mapping algorithm, which is left for later work.

Unfortunately, due to the coverage of the Q.U.T. library, and the subjects offered at Q.U.T. being technical in nature, a lot of subjects had two terms or less, with a number of subjects having no terms at all. This could be rectified by combining the documents from the Q.U.T. library with other university libraries, thus extending the number of subjects covered and the depth of the terms mined from the ontology. Religion is one example of a Q.U.T. subject that has no coverage in the extracted results.

Another problem found was that some of the TREC queries were U.S. centric, while the Q.U.T. library catalogue is more focused on Australian issues. This impacted the quality of some of the results, better results would have been achieved if U.S. library data had been included in the ontology.

7.2 Experiments for Search Engine Content Analysis

After proving that the ontology-based collection selection system is an effective collection selection method on a standard testbed in a controlled setting with relevance judgements, it was applied to large real world search engines and deep web collections. Table 7.4 shows the search engines used in this analysis. These are some of the largest and most popular search engines

in use today. In addition Wikipedia was included as an example of a centralised repository of information. To test that the method works, three specialist search engines were also included; the U.S. Department of Commerce, PubMed (A medical search engine), and the U.S. Department of Agriculture. This section shows the results from the third level of the Dewey Decimal taxonomy comparison.

7.2.1 Data

The IntelliOnto ontology was mined for extract terms which were then used to analyse some of the largest search engines in common use today. As exactly the same set of terms is sent to each search engine it is possible to compare differences *within subjects*, ie comparing difference between Google and Yahoo for the Dewey Decimal code 106. However, note that in the current form differences in content *between subjects* cannot be accurately compared, ie between Dewey Decimal Code 100 and Dewey Decimal Code 200 for Google. This is because it is possible that highly subject specific terms exist in Google but do not exist in the classification set. The same term can be used to measure the same subject across two different collections, but different terms cannot be used when comparing the same subject across different collections. It may be possible to normalise the subject results but this is left for future research.

Each term from the ontology is sent to each search engine, and the number of results from each search engine is then used for the data in this experiment.

Title	Abbreviation	URL
Google Search	Google	Google A.P.I.
Wikipedia	Wikipedia	http://en.wikipedia.org/
Microsoft Live Search	Live	http://www.live.com/
U.S. Department of Commerce	Commerce	http://www.commerce.gov/
USA Search.gov	FirstGov	http://usasearch.gov
Accoona Search	Accoona	http://www.accoona.com/
PubMed	PubMed	http://www.ncbi.nlm.nih.gov/sites/entrez
U.S Department of Agriculture	Agriculture	http://www.usda.gov/
Yahoo Search	Yahoo	Yahoo A.P.I.

Table 7.4: The Search Engines Analysed In This Thesis

7.2.2 Parameters and measures

The results from the different levels of the taxonomy are all normalised using a Frobenius norm in Matlab.

7.2.3 Results

7.2.3.1 Third Level Analysis

Once the terms are mined from the ontology the terms are then sent to each search engines. The results from the terms are then grouped together by Dewey Decimal code to get the subject distribution of the third level of the taxonomy.

U.S. Department of Agriculture Results

The U.S. Department of Agriculture search engine performed well in the following subjects. Note that each subject description is preceded by it's Dewey Decimal Code:

- 630 Agriculture. Figure 7.5.
- 632 Plant injuries, diseases, pests. Figure 7.6.
- 633 Field & plantation crops. Figure 7.7.
- 635 Garden crops (Horticulture). Figure 7.8.
- 636 Animal husbandry. Figure 7.9.
- 637 Processing dairy & related products. Figure 7.10.

PubMed Results

The PubMed search engine performed well in the following subjects. Note that each subject description is preceded by it's Dewey Decimal Code:

- 610 Medical sciences Medicine. Figure 7.11.
- 176 Ethics of sex & reproduction. Figure 7.12.
- 536 Heat (Natural sciences). Figure 7.13.

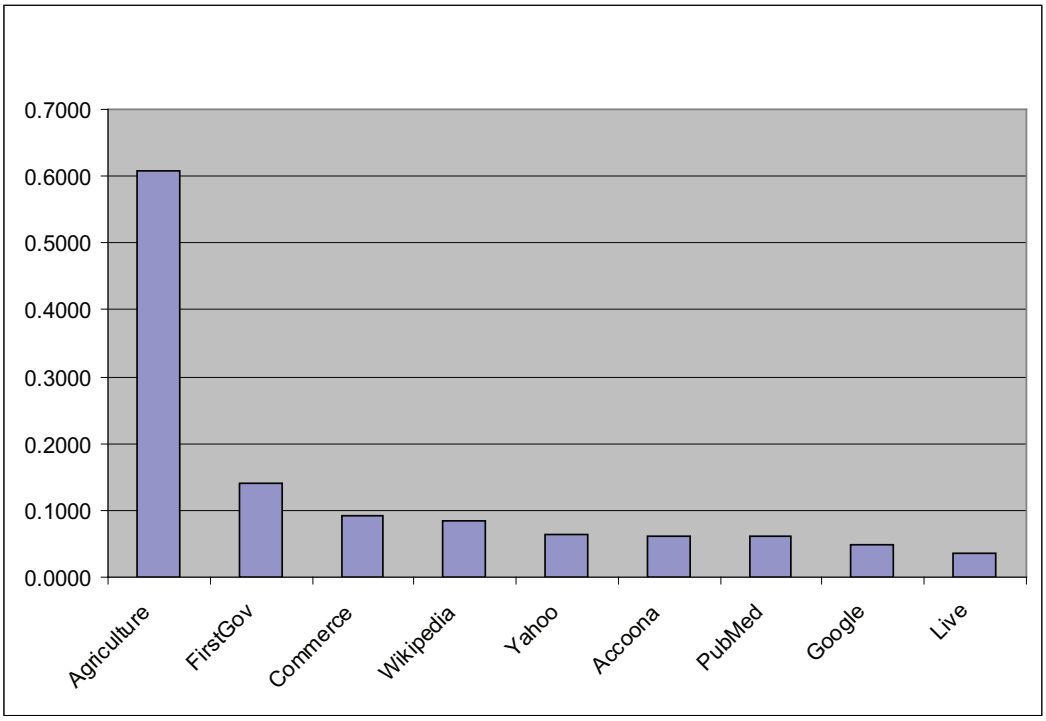


Figure 7.5: 630 Agriculture. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

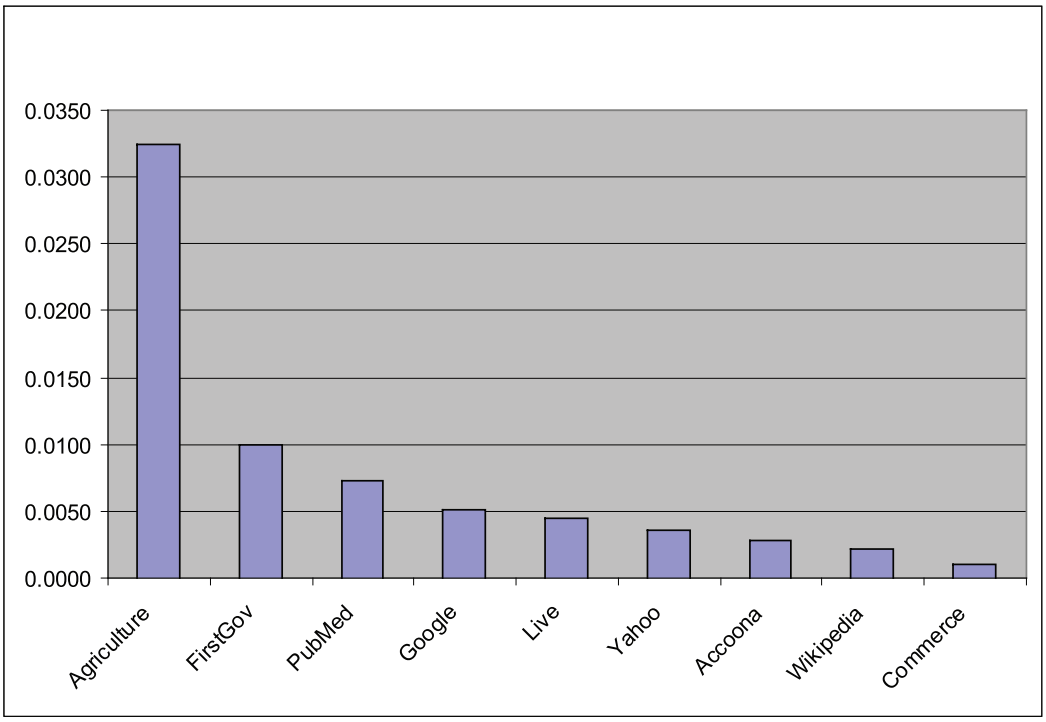


Figure 7.6: 632 Plant Injuries, Diseases, Pests. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

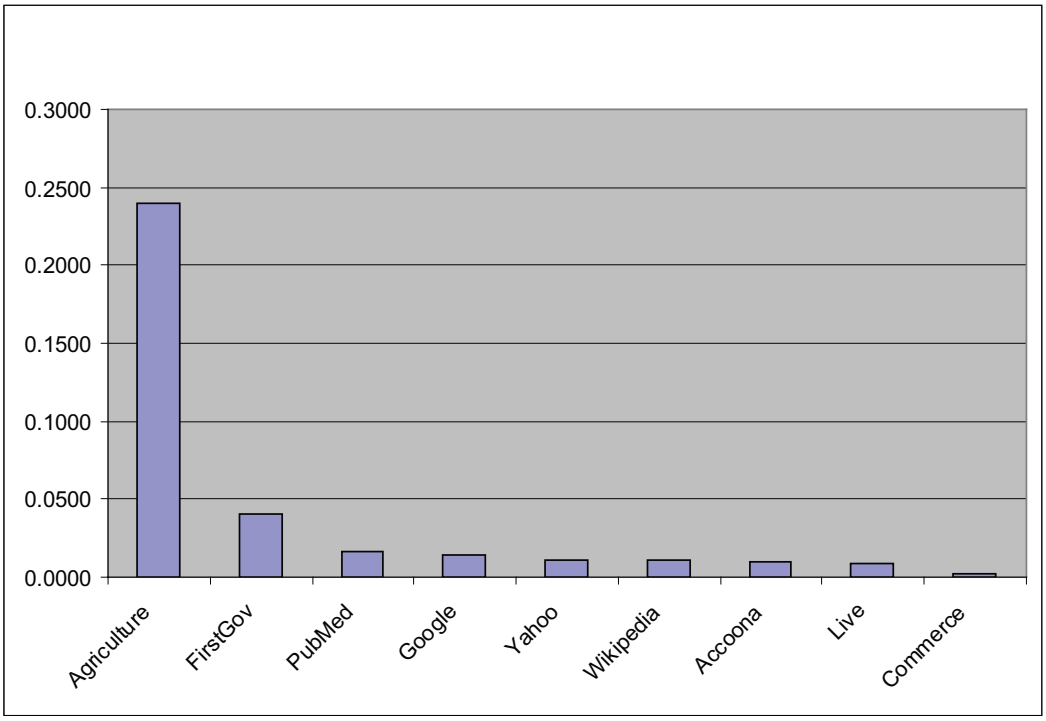


Figure 7.7: 633 Field & Plantation Crops. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

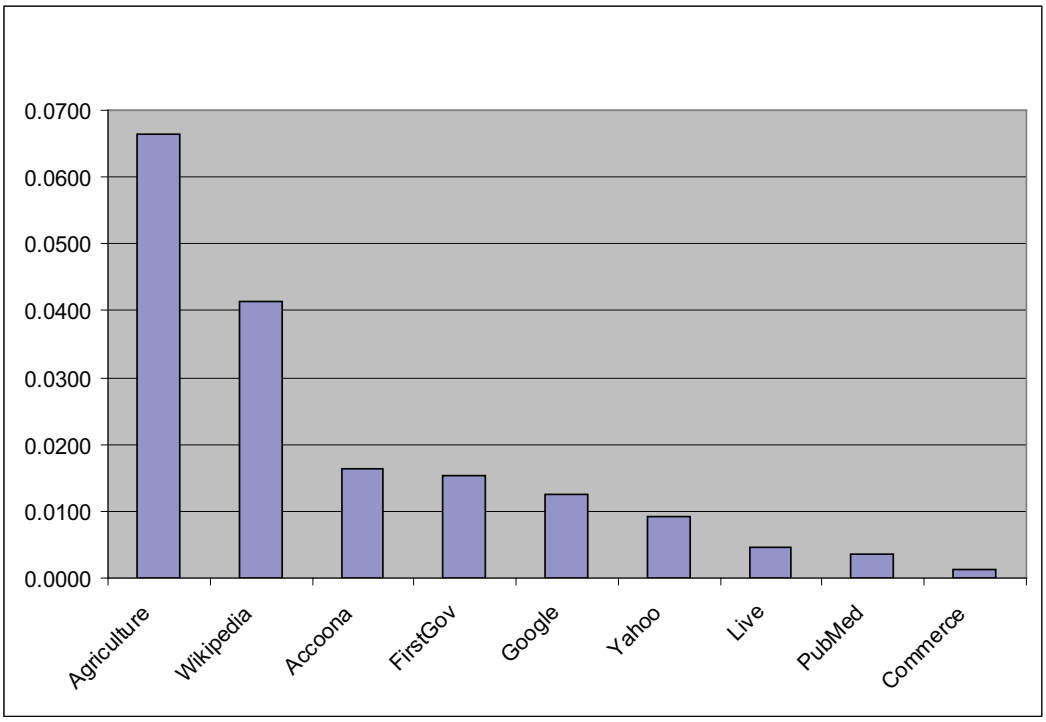


Figure 7.8: 635 Garden Crops (Horticulture). The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

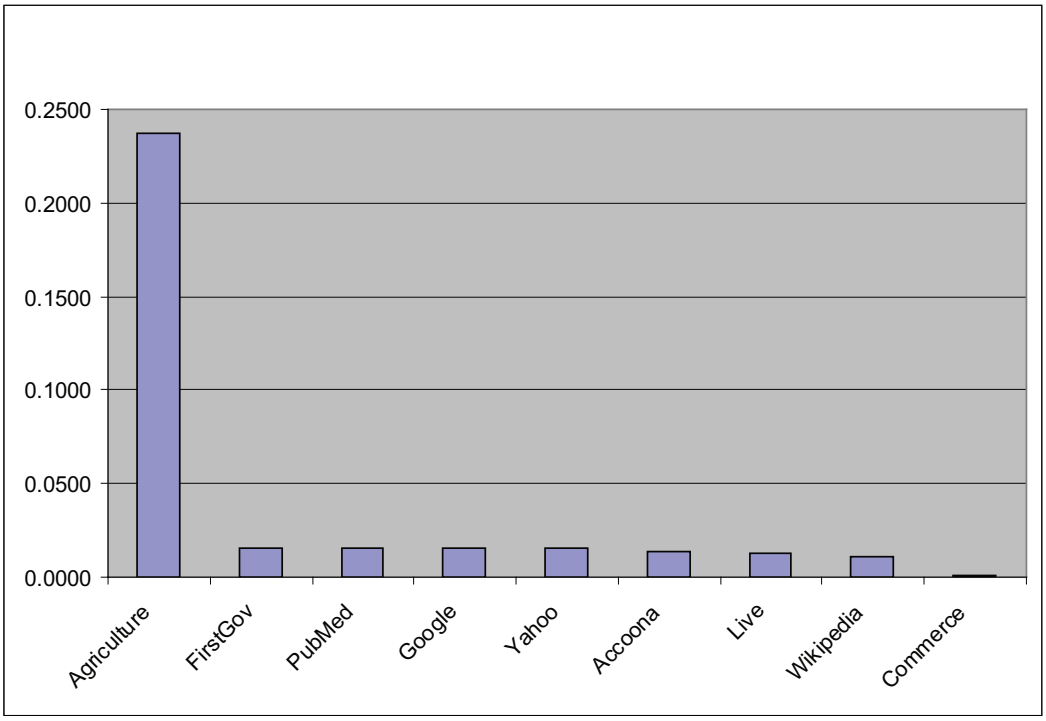


Figure 7.9: 636 Animal Husbandry. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

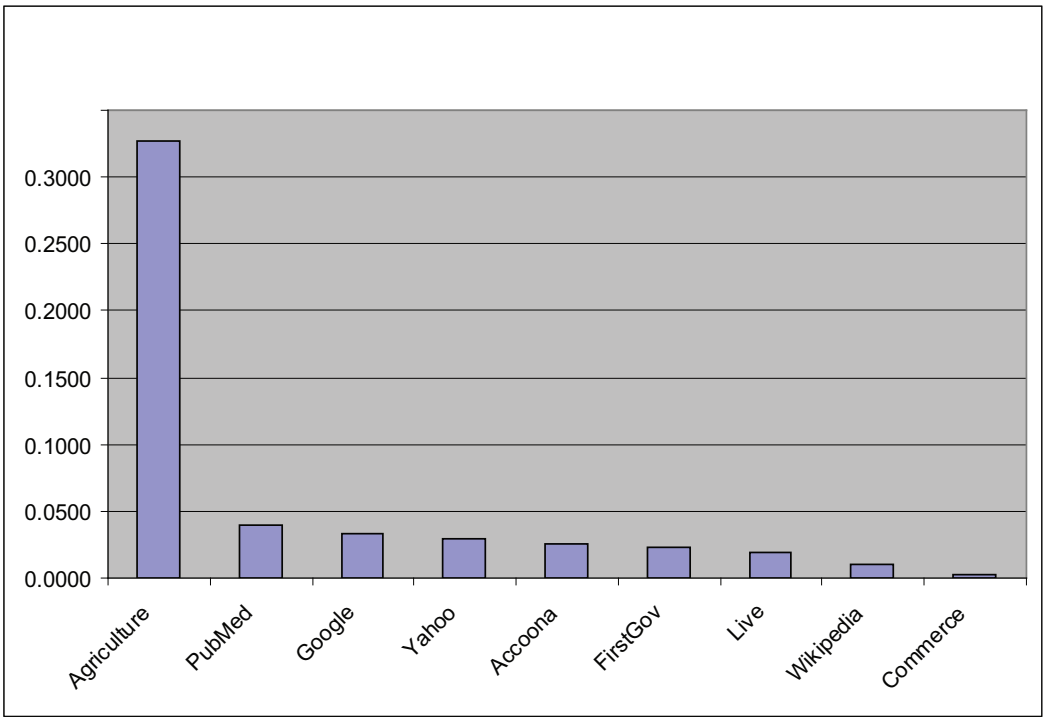


Figure 7.10: 637 Processing Dairy & Related Products. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

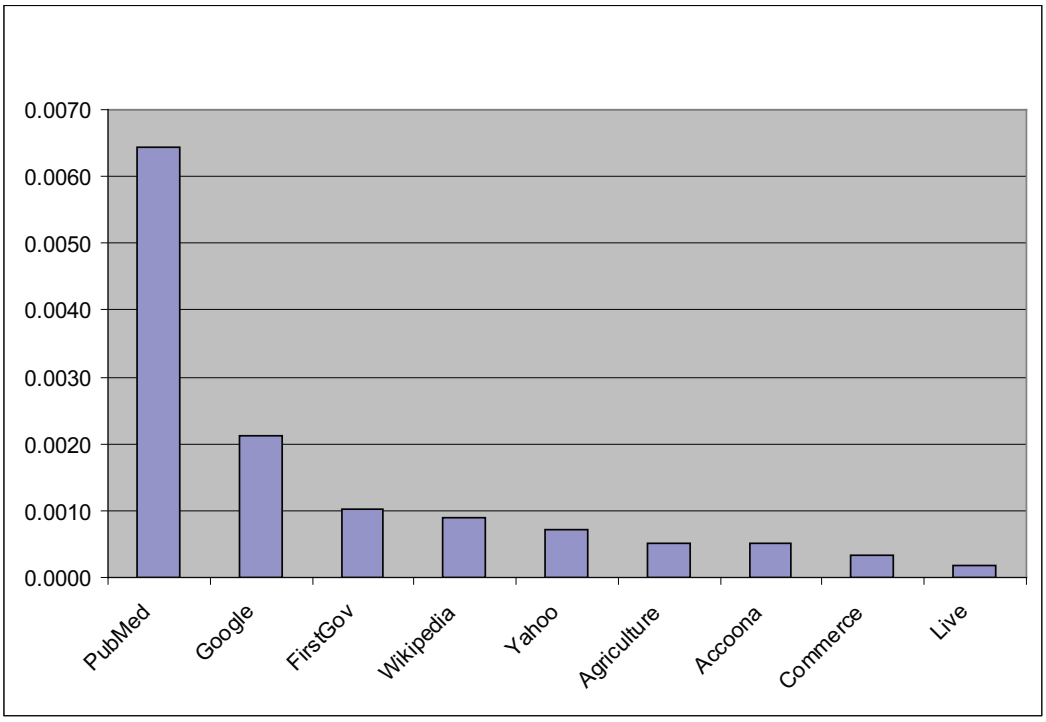


Figure 7.11: 610 Medical Sciences - Medicine. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

- 575 Evolution & genetics. Figure 7.14.
- 596 Vertebrata (Craniata, Vertebrates). Figure 7.15.

U.S. Department of Commerce Results

The U.S. Department of Commerce search engine performed well in the following subjects. Note that each subject description is preceded by it's Dewey Decimal Code:

- 023 Personnel administration. Figure 7.16.
- 174 Economic & professional ethics. Figure 7.17.
- 342 Constitutional & administrative law. Figure 7.18.
- 511 General principles (Mathematics). Figure 7.19.

7.2.3.2 Top Level Analysis

The highest level of the Dewey Decimal system consists of the following subject groups:

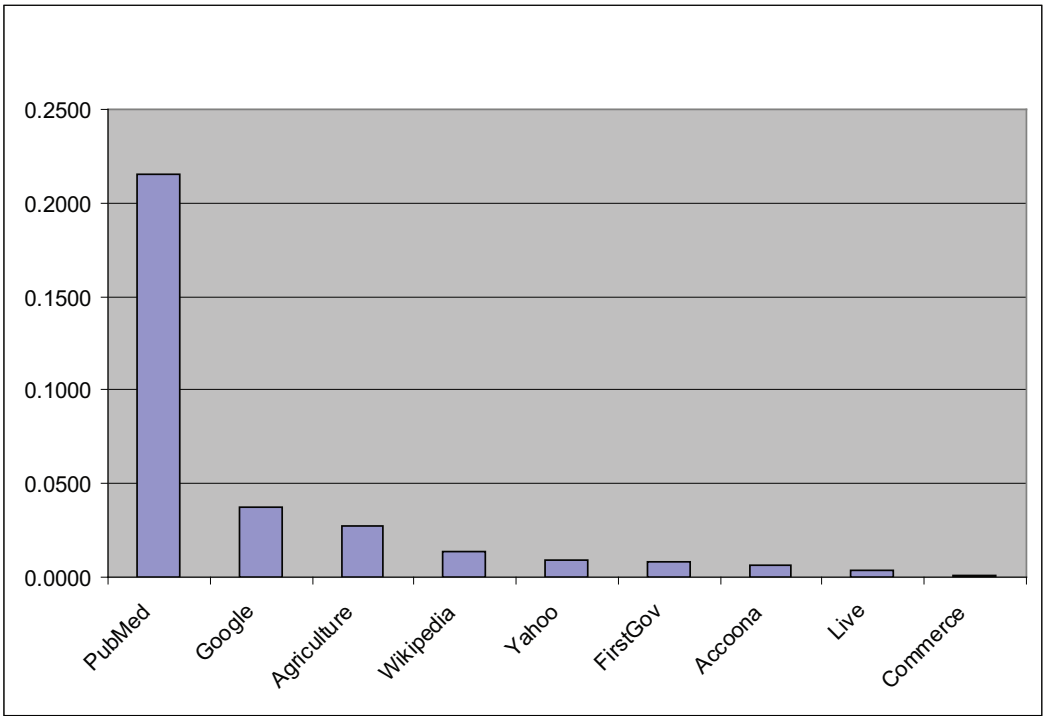


Figure 7.12: 176 Ethics of Sex & Reproduction. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

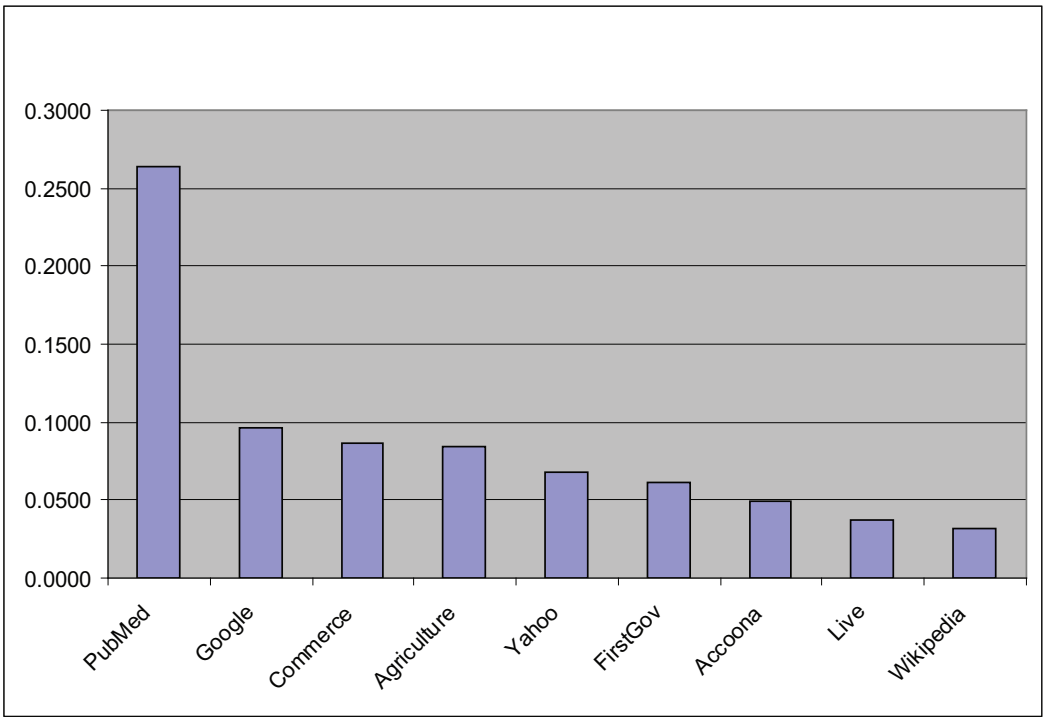


Figure 7.13: 536 Heat (Natural Sciences). The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

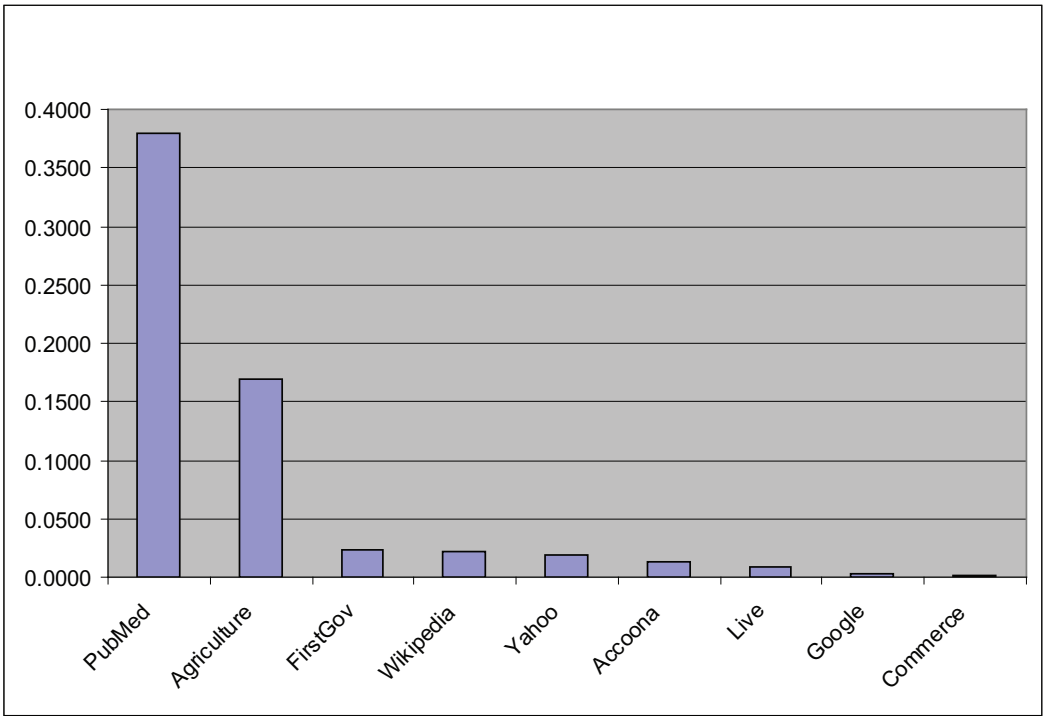


Figure 7.14: 575 Evolution & Genetics. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

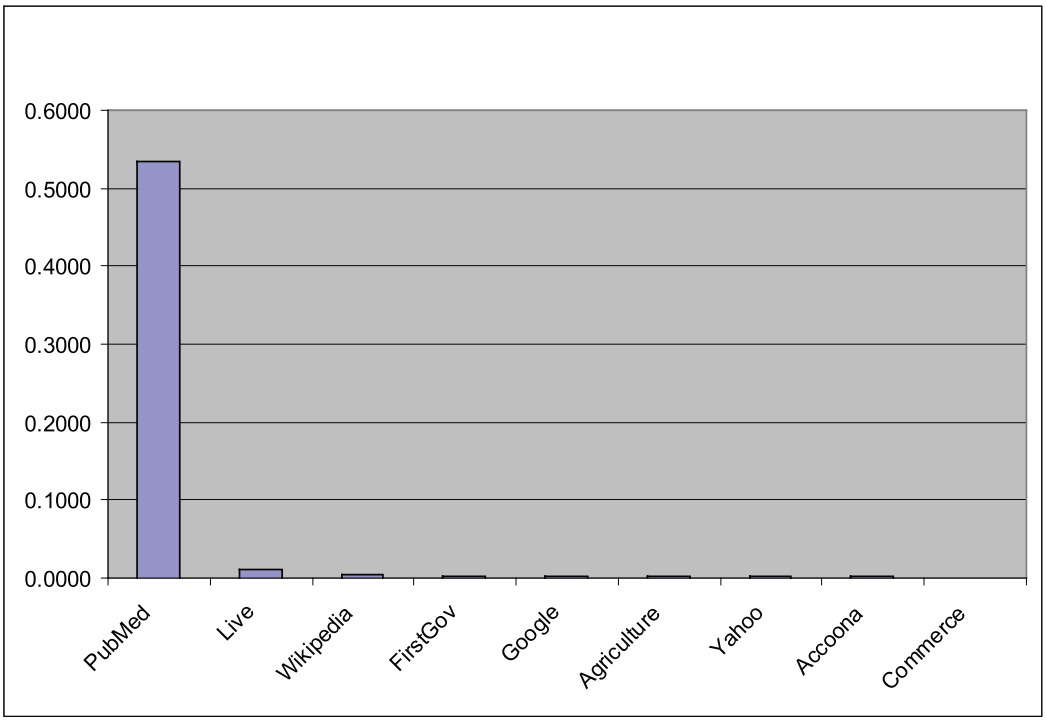


Figure 7.15: 596 Vertebrata (Craniata, Vertebrates). The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

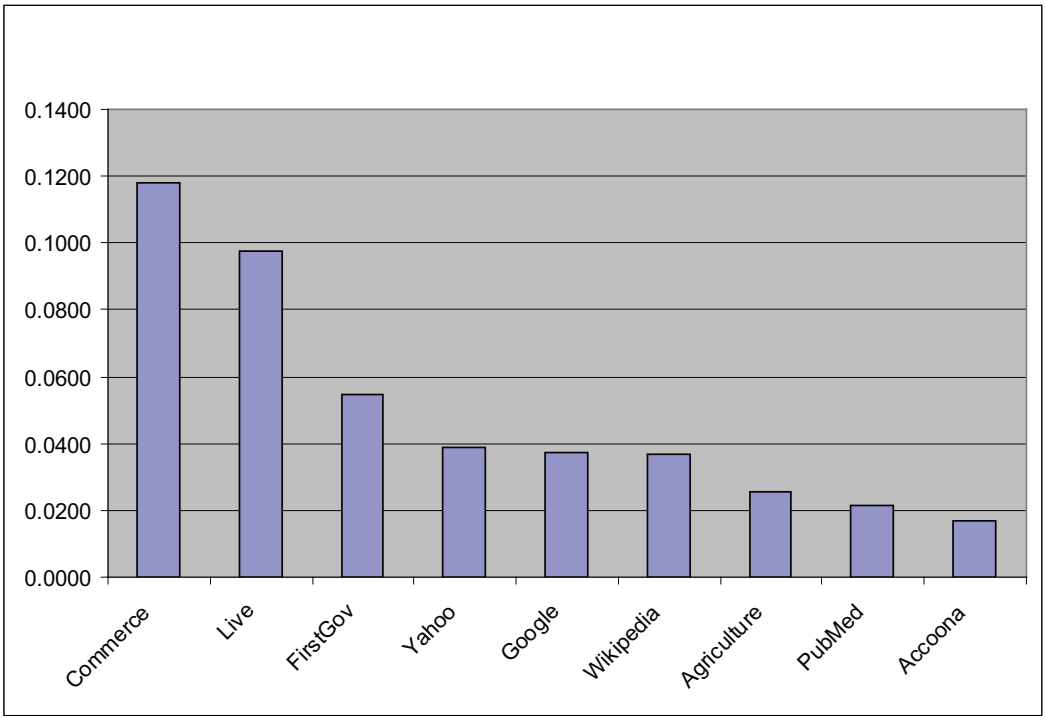


Figure 7.16: 023 Personnel Administration. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

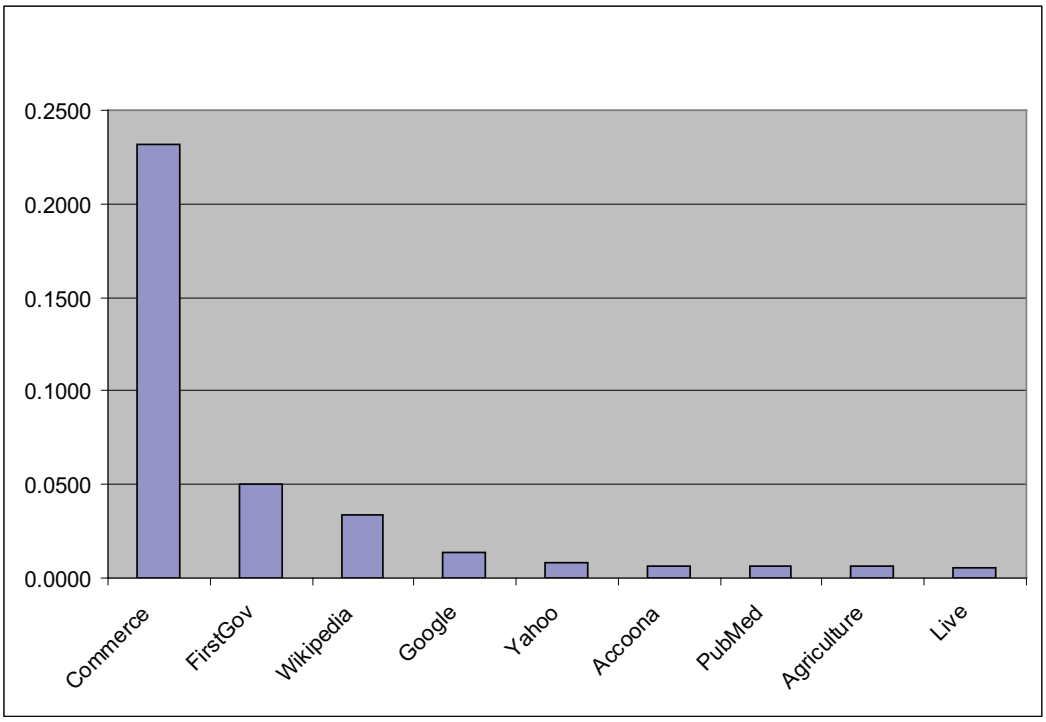


Figure 7.17: 174 Economic & Professional Ethics. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

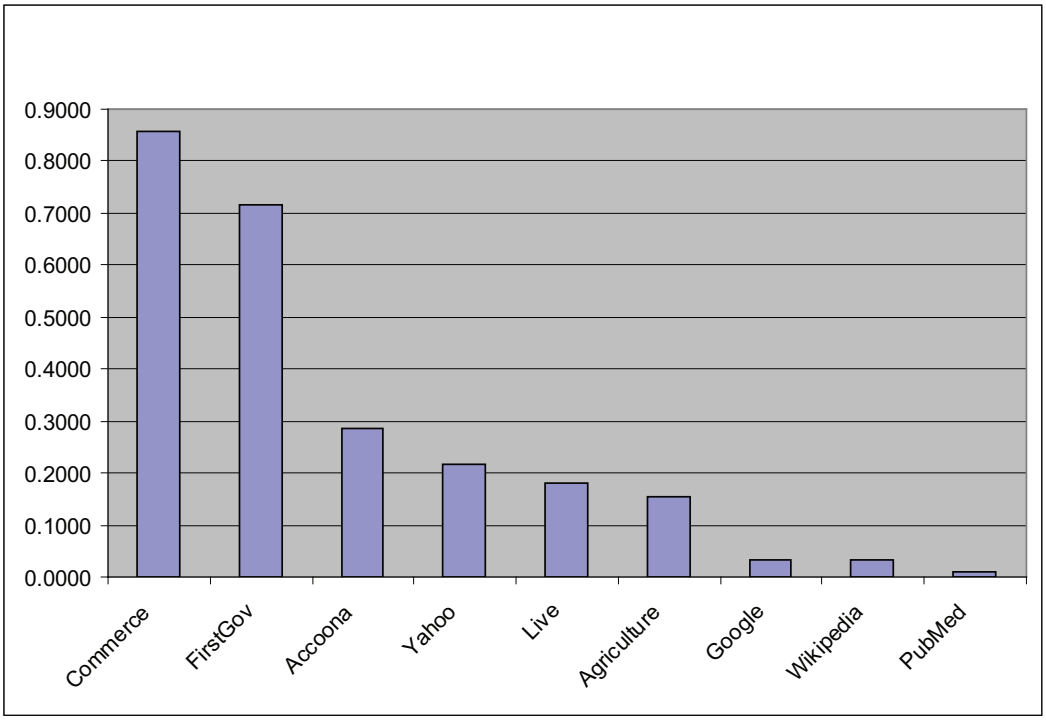


Figure 7.18: 342 Constitutional & Administrative Law. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

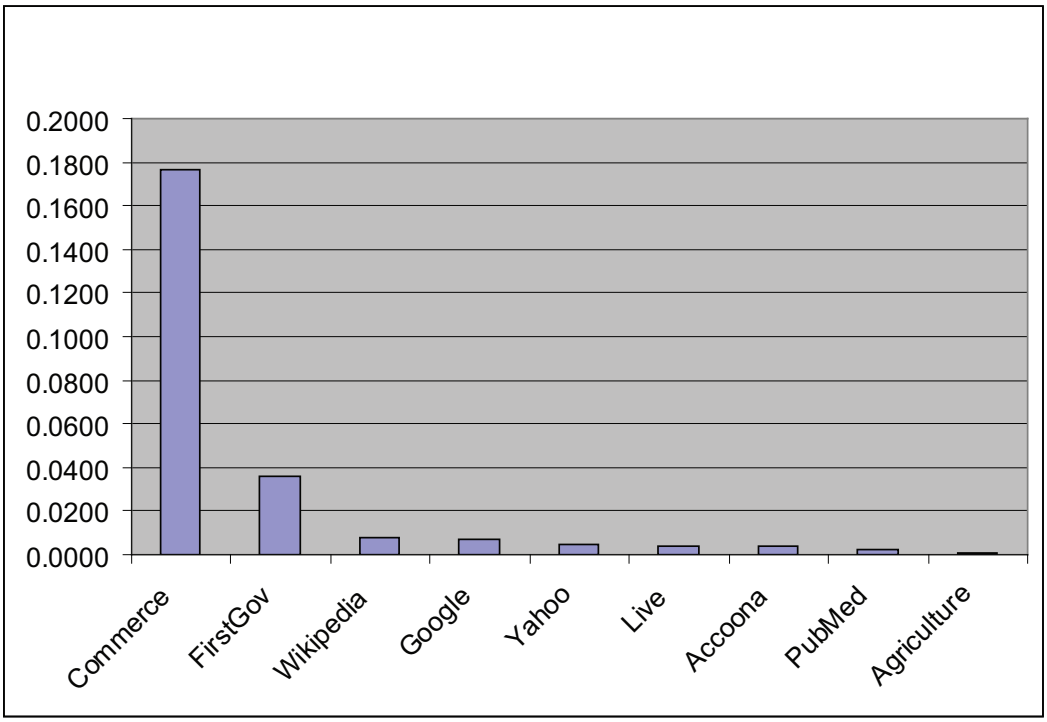


Figure 7.19: 511 General Principles (Mathematics). The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

- 000 Generalities
- 100 Philosophy & psychology
- 200 Religion
- 300 Social sciences
- 400 Language
- 500 Natural sciences & mathematics
- 600 Technology (Applied sciences)
- 700 The arts
- 800 Literature & rhetoric
- 900 Geography & history

After calculating the third level of the taxonomy, the results are grouped together to view the top level of the taxonomy. For example, for the top level Dewey code 100 the sum of the results of Dewey codes 100 to 199 are taken. The process for creating each of these is:

1. Query probe the search engines using the selected terms and extract the results
2. Sum the results based on which subject code the terms belonged to across the third level of the Dewey Decimal system
3. Sum the results of the third level of the system together to create the top level of the Dewey Decimal system

Table 7.5 shows the raw data from the top level of the hierarchy. Table 7.6 shows the normalised data from the top level of the hierarchy. The graphs are based on the normalised data.

The following figures 7.20, 7.21, 7.22, 7.23, 7.24, 7.25, 7.26, 7.27, 7.28, and 7.29 show how the search engines performed for each of the highest level subjects of the Dewey Decimal

	Google	Wikipedia	Live	Commerce	FirstGov	Accoona	PubMed	Agriculture	Yahoo
000	3.55E+09	2.43E+07	3.87E+09	1.36E+07	2.12E+08	2.28E+08	3.66E+05	8.86E+05	6.39E+09
100	3.73E+09	1.34E+07	4.38E+09	1.15E+07	5.69E+07	2.60E+08	8.93E+05	1.20E+05	4.77E+09
200	2.07E+09	1.32E+07	2.32E+09	4.23E+06	9.70E+07	2.42E+08	3.06E+05	1.74E+05	5.07E+09
300	9.05E+09	4.72E+07	1.02E+10	5.63E+07	6.28E+08	9.11E+08	9.77E+05	1.05E+06	1.59E+10
400	7.78E+09	1.13E+07	5.60E+09	3.80E+06	9.63E+07	5.35E+08	6.37E+05	3.52E+05	1.05E+10
500	5.31E+09	2.52E+07	4.25E+09	2.19E+07	2.16E+08	3.43E+08	7.03E+06	1.15E+06	6.42E+09
600	9.10E+09	3.14E+07	1.31E+10	3.83E+07	3.94E+08	1.09E+09	2.28E+06	3.14E+06	1.55E+10
700	9.43E+09	4.09E+07	2.25E+10	9.24E+06	2.27E+08	1.15E+09	1.14E+06	6.44E+05	2.08E+10
800	6.97E+09	1.44E+07	6.04E+09	3.67E+06	6.31E+07	4.94E+08	7.37E+05	1.25E+05	9.32E+09
900	8.34E+09	3.29E+07	8.13E+09	1.92E+07	1.78E+08	9.10E+08	7.40E+05	5.98E+05	1.27E+10

Table 7.5: Results of the top level of the Dewey hierarchy.

	Google	Wikipedia	Live	Commerce	FirstGov	Accoona	PubMed	Agriculture	Yahoo
000	0.1603	0.2736	0.1240	0.1760	0.2450	0.1020	0.0475	0.2368	0.1698
100	0.1684	0.1509	0.1405	0.1493	0.0657	0.1165	0.1158	0.0322	0.1269
200	0.0932	0.1490	0.0743	0.0549	0.1121	0.1082	0.0396	0.0466	0.1348
300	0.4081	0.5312	0.3281	0.7308	0.7248	0.4077	0.1267	0.2798	0.4225
400	0.3509	0.1273	0.1795	0.0493	0.1111	0.2395	0.0827	0.0942	0.2794
500	0.2396	0.2834	0.1364	0.2846	0.2491	0.1533	0.9122	0.3066	0.1705
600	0.4107	0.3542	0.4207	0.4976	0.4553	0.4886	0.2957	0.8386	0.4125
700	0.4254	0.4612	0.7201	0.1200	0.2622	0.5135	0.1483	0.1721	0.5534
800	0.3144	0.1622	0.1937	0.0477	0.0729	0.2211	0.0956	0.0334	0.2477
900	0.3764	0.3702	0.2607	0.2489	0.2050	0.4073	0.0960	0.1598	0.3370

Table 7.6: Normalised results of the top level of the Dewey hierarchy.

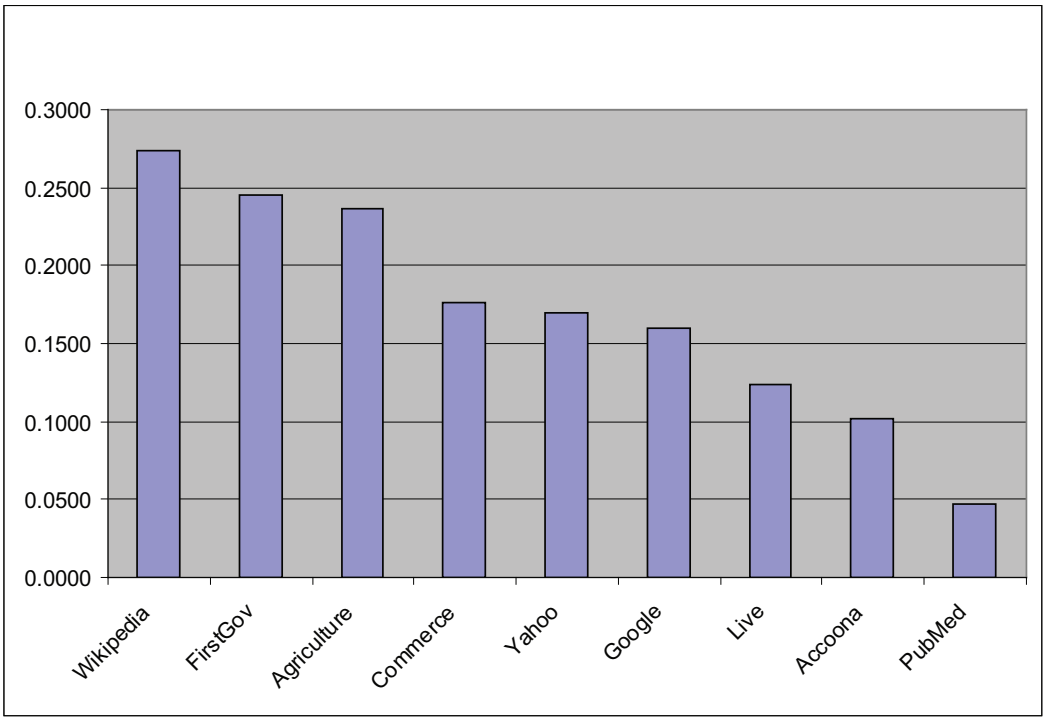


Figure 7.20: 000 Generalities. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

system. Note that PubMed performed very well on *500 Natural sciences & mathematics*, which covers subjects related to biology and medicine. Also the Agriculture search engine performed well on *600 Technology (Applied sciences)* which covers subjects related to agriculture, and the Commerce search engine performed well on *300 Social sciences* which contains some subjects related to commerce.

7.2.4 Graphical Representation

Figure 7.30 shows a graphical representation of the correlation of the search engines. The Matlab PlotFreq function was used to generate the data, and then "figure;y=x*x';imagesc(y);" was run to generate the graph. The Jet colourmap was used for this graph. Blue colours indicate low correlation, and red colours indicate high correlation. The numbers correspond to:

1. Google
2. Wikipedia
3. Live

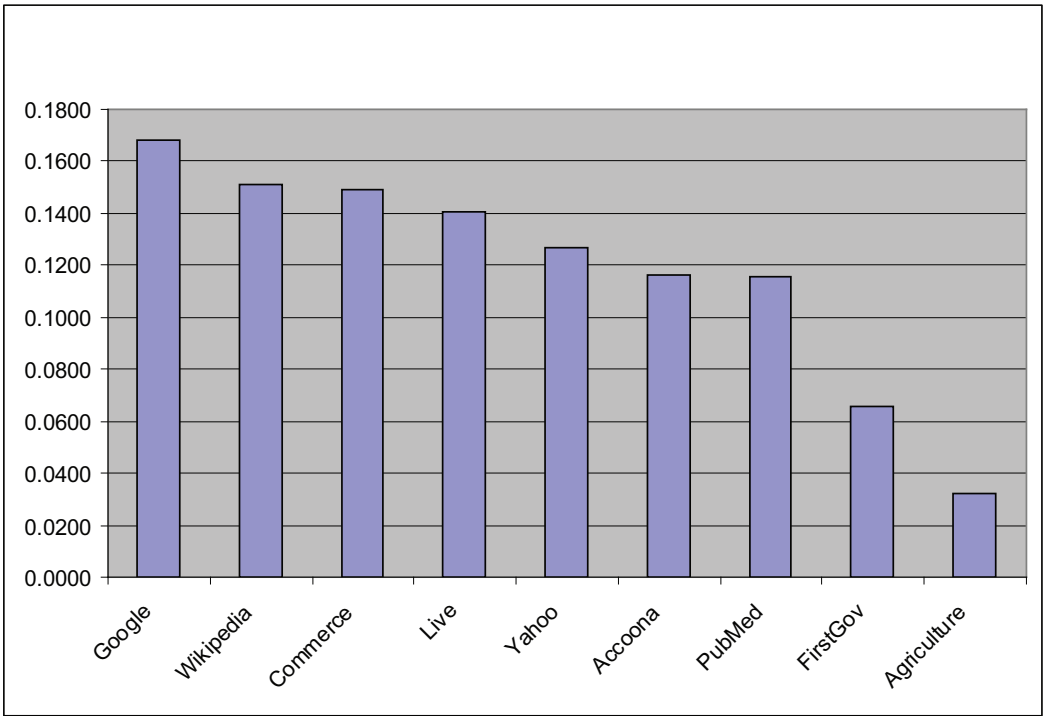


Figure 7.21: 100 Philosophy & psychology. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

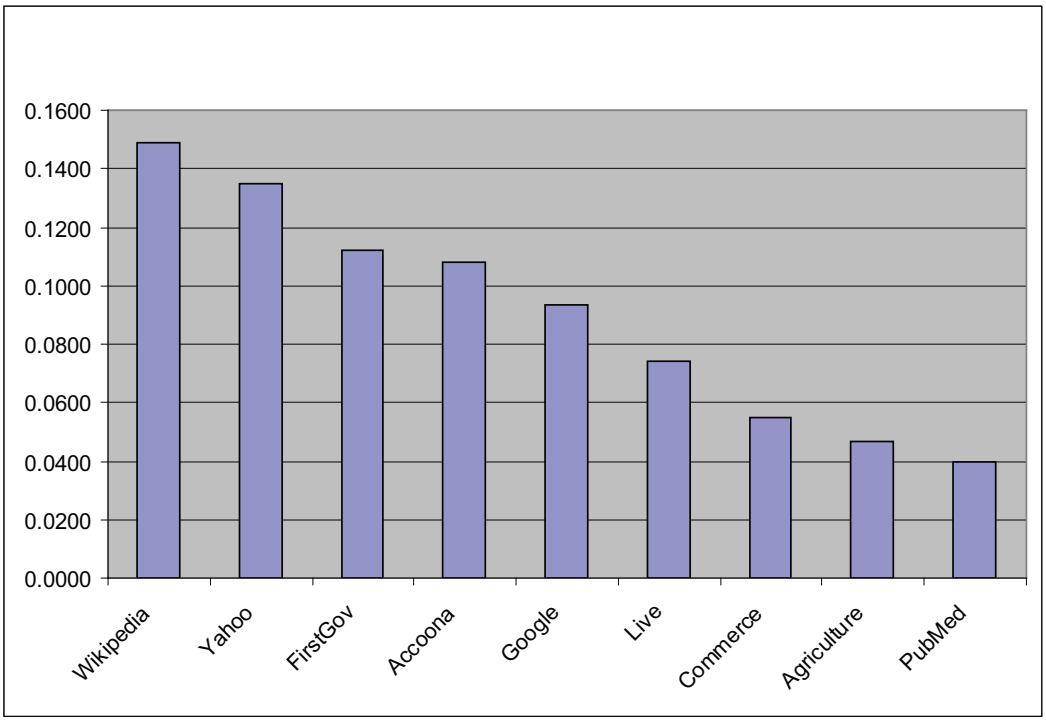


Figure 7.22: 200 Religion. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

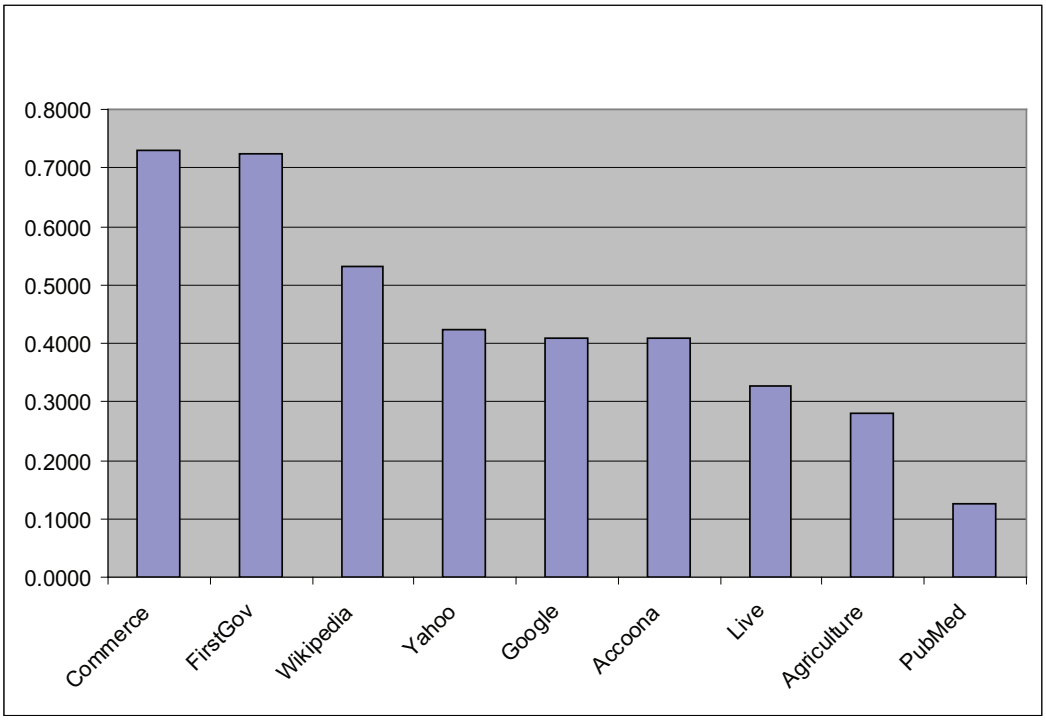


Figure 7.23: 300 Social sciences. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

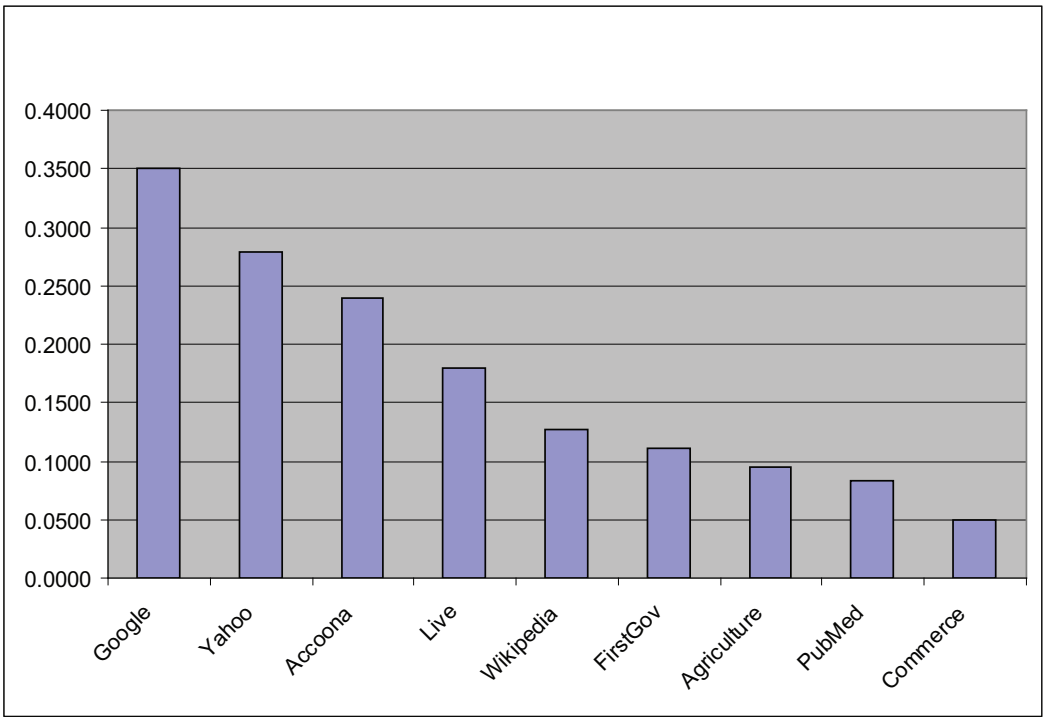


Figure 7.24: 400 Language. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

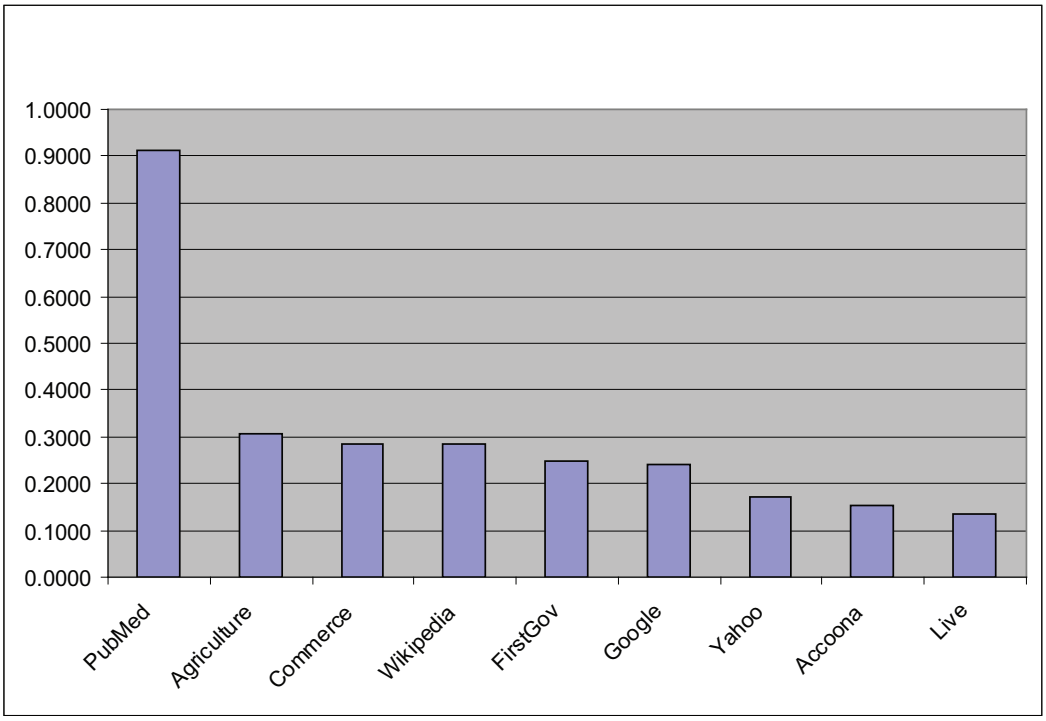


Figure 7.25: 500 Natural sciences & mathematics. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

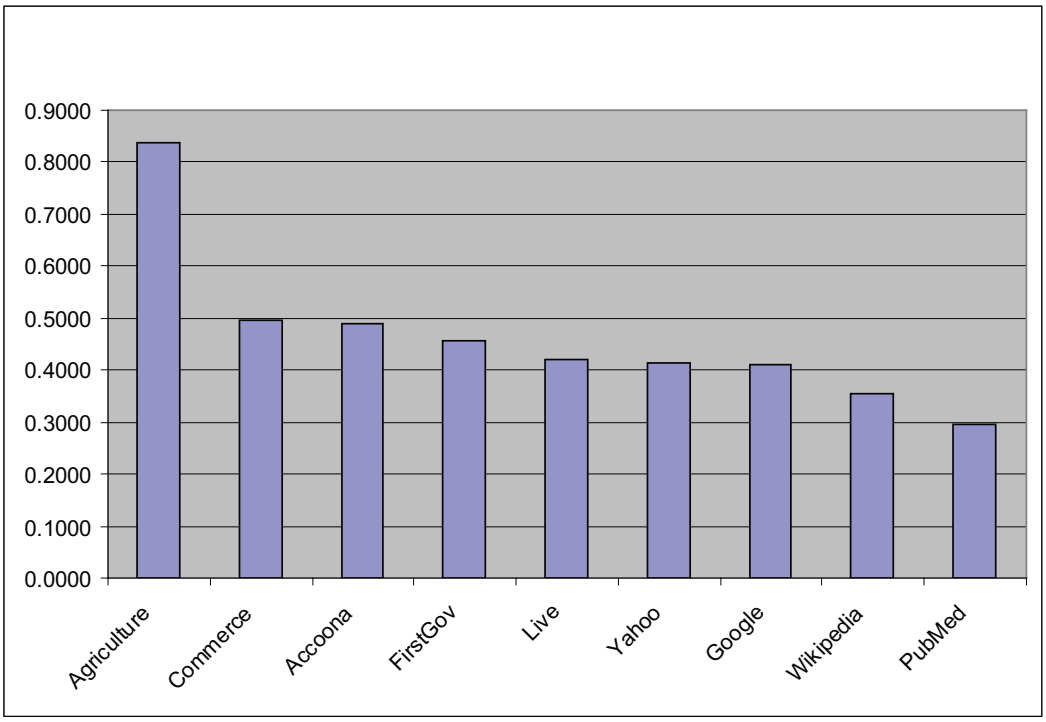


Figure 7.26: 600 Technology (Applied sciences). The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

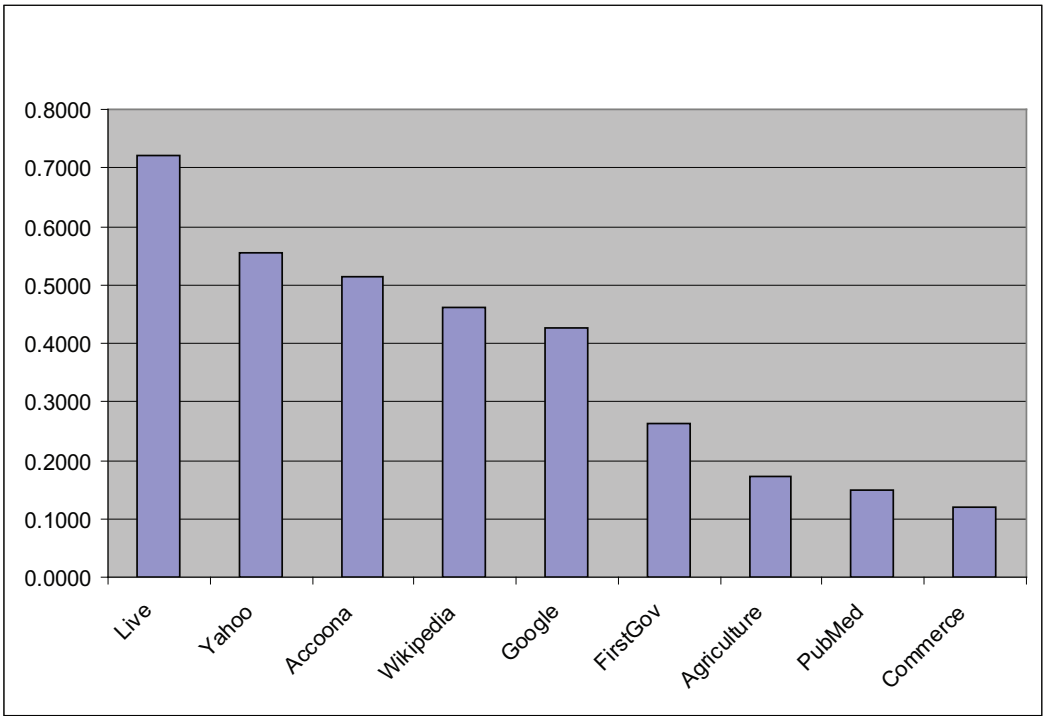


Figure 7.27: 700 The arts. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

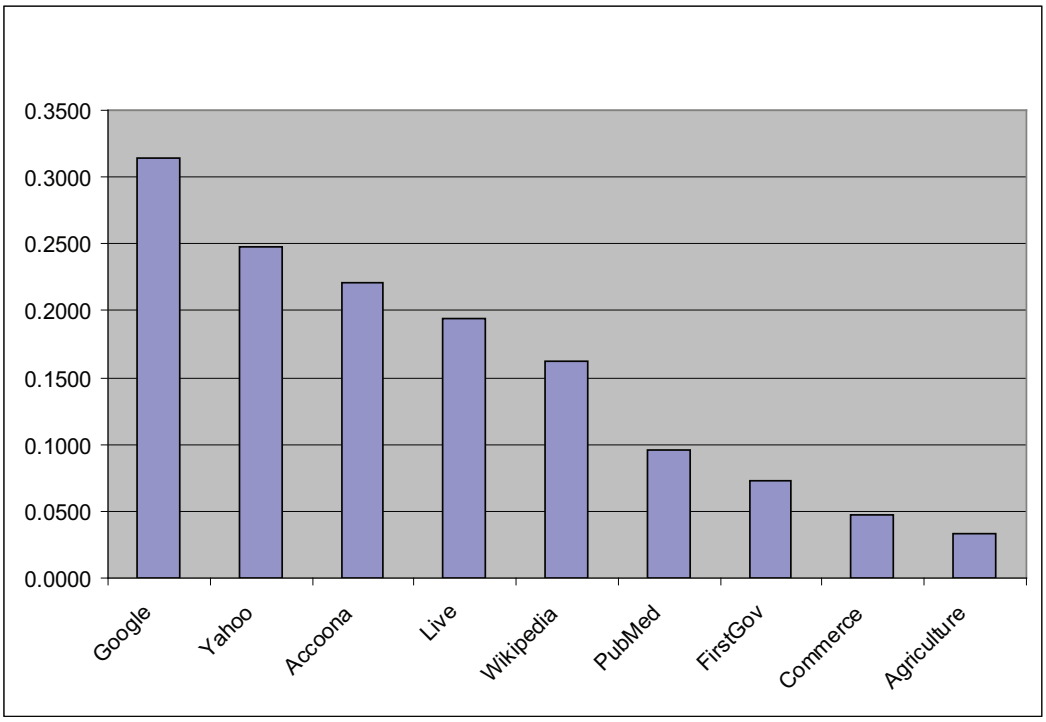


Figure 7.28: 800 Literature & rhetoric. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

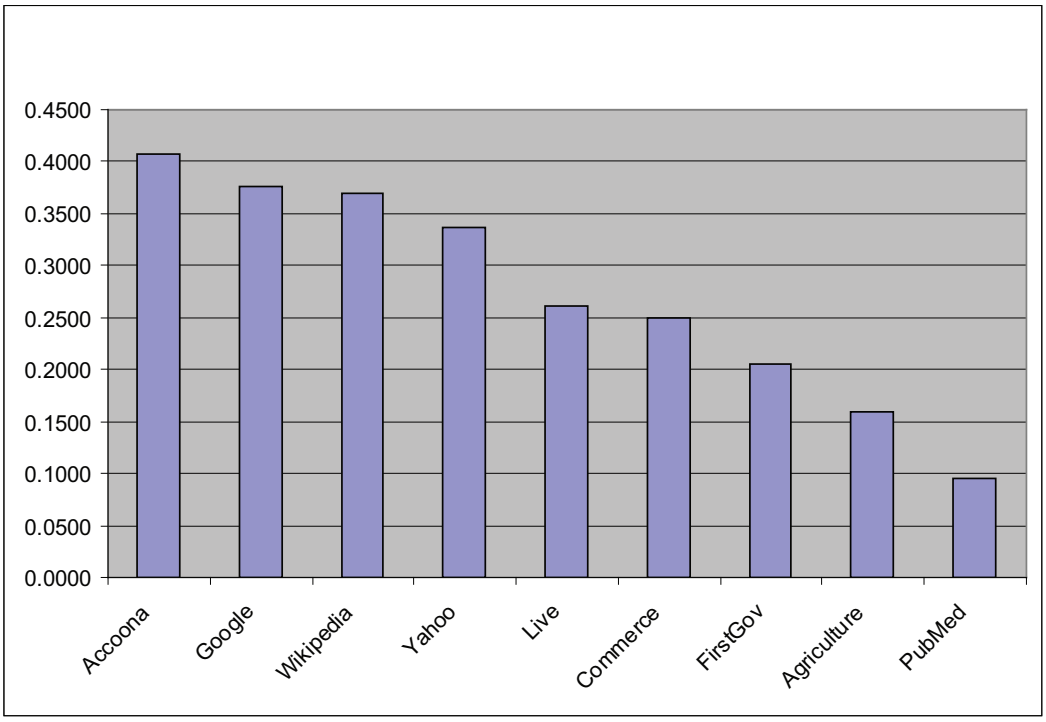


Figure 7.29: 900 Geography & history. The y-axis represents normalised relevance to the subject, with a score of 1.0 being perfect relevance.

4. Commerce
5. FirstGov
6. Accoona
7. PubMed
8. Agriculture
9. Yahoo

7.2.5 Summary

Overall, the ontology-based method performed well, and shows a lot of potential as both a search engine content analysis method and a search engine selection method. Only the notable results from the third level of the hierarchy are shown. There are many other results that performed just as well but only the highest ranking ones are shown here.

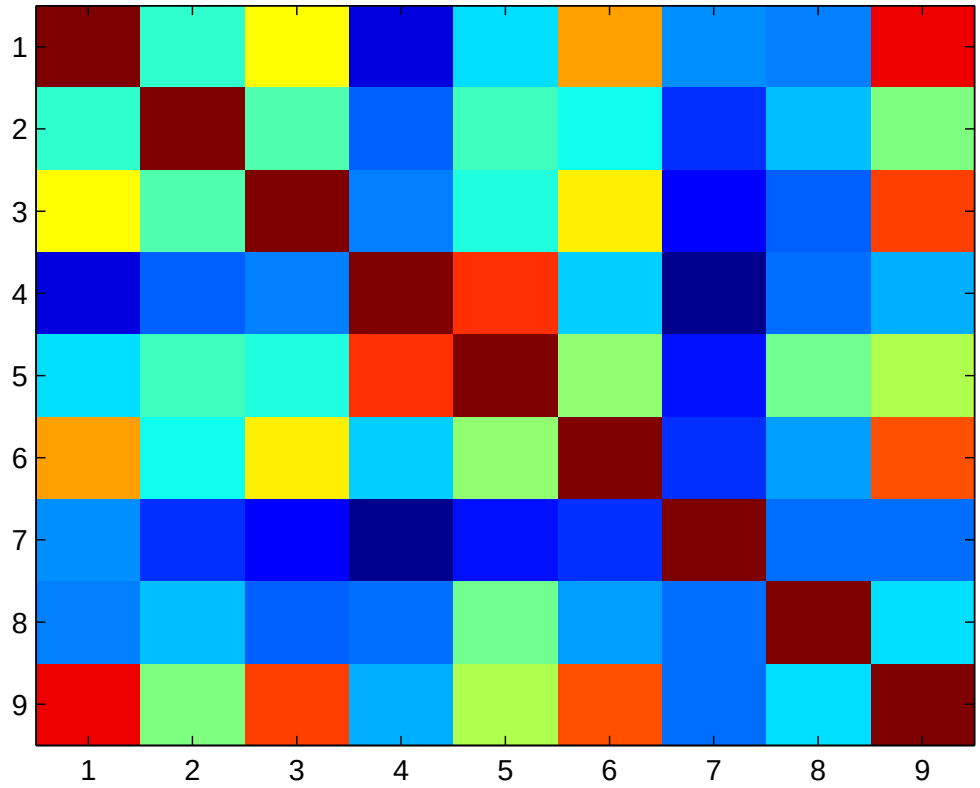


Figure 7.30: Graphical representation of the data

After examining the above results, and reviewing many others, it can be concluded that the ontology-based method performs just as well in a real world situation as is does in controlled experiments. And because it uses a much deeper set of subjects in the hierarchy (three levels deep) than other methods it can arguably produce superior results. The early ontology-based method only used a few subjects [KLBN06], whereas this later work uses many hundreds of subjects.

7.3 Experiments for Singular Value Decomposition

Singular Value Decomposition is used to compare the search engines with each other. It is a useful method for quickly finding similar search engines. This information can then be used to decide on what search engines to return for an information need. This method was also used in my masters thesis.

7.3.1 Data

The Singular Value Decomposition was performed directly on the number of results for each term from each search engine. A rectangular matrix was built with the search engines as the columns and the terms as the rows.

7.3.2 Results

Table 7.7 shows the singular value decomposition of the results of the query probes for each search engine. Each value in the table has a value between -1.0 and 1.0, with a 1.0 indicating an exact match of the data. Singular value decomposition was used because it is able to find latent patterns between different sets of data. It is able to show how related each of the search engines are to each other, which is useful when selecting the search engines most relevant to a subject.

Table 7.7 shows that the major search engines Google, Yahoo, Accoona and Live were all closely related to each other.

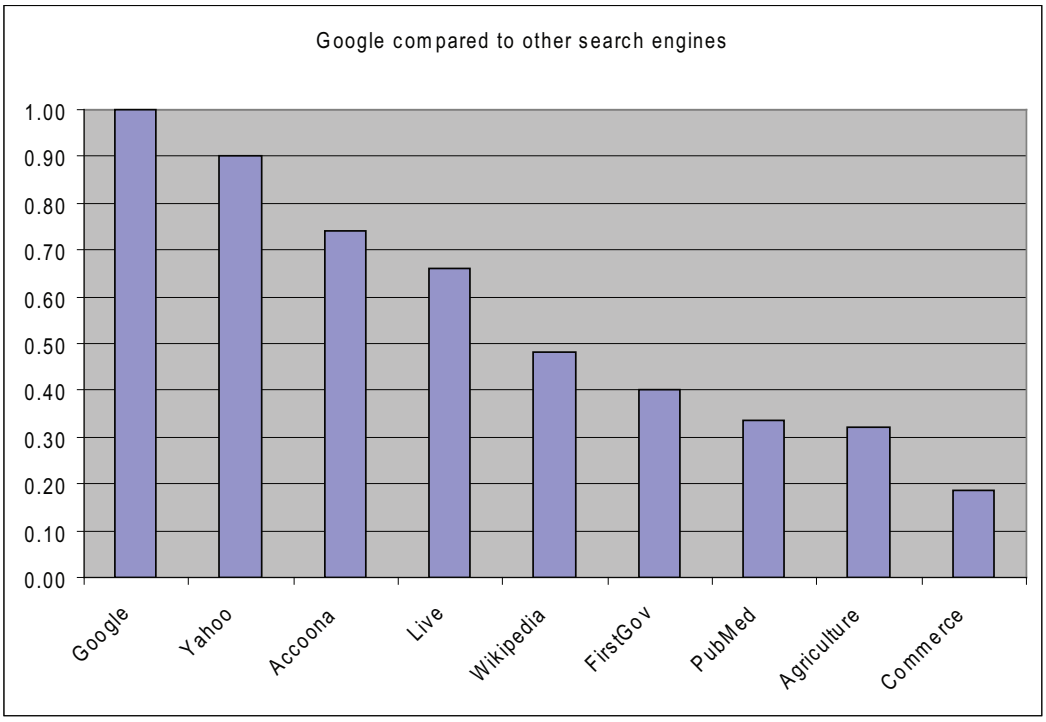


Figure 7.31: Similarity measure of Google to other search engines. The y-axis represents how related the search engines are to each other.

7.3.2.1 Singular Value Decomposition Analysis of Google

Figure 7.31 shows a comparison between Google and the other search engines using singular value decomposition. This figure shows that:

- The search engine that was most similar to Google was Yahoo, which is Google's closest competitor.
- The search engine that was the most different to Google was the Commerce search engine

7.3.2.2 Singular Value Decomposition Analysis of Microsoft Live Search, PubMed, and Yahoo

Figures 7.32, 7.33, and 7.34 show a singular value decomposition comparison of the Microsoft Live, Pubmed, and Yahoo search engines with the other search engines.

	Google	Wikipedia	Live	Commerce	FirstGov	Accoona	PubMed	Agriculture	Yahoo
Google	1.0000								
Wikipedia	0.4821	1.0000							
Live	**0.6619**	0.5080	1.0000						
Commerce	0.1847	0.3009	**0.3240**	1.0000					
FirstGov	0.4033	0.4953	0.4678	**0.8360**	1.0000				
Accoona	**0.7412**	0.4572	0.6741	0.3957	0.5670	1.0000			
PubMed	**0.3354**	0.2600	0.2173	0.1118	0.2304	0.2614	1.0000		
Agriculture	0.3223	0.3816	0.3032	0.3116	**0.5377**	0.3518	0.3099	1.0000	
Yahoo	**0.9007**	0.5440	0.8271	0.3642	0.5894	0.8185	0.3131	0.4068	1.0000

Table 7.7: Singular value decomposition results from highest confidence and support query probes.

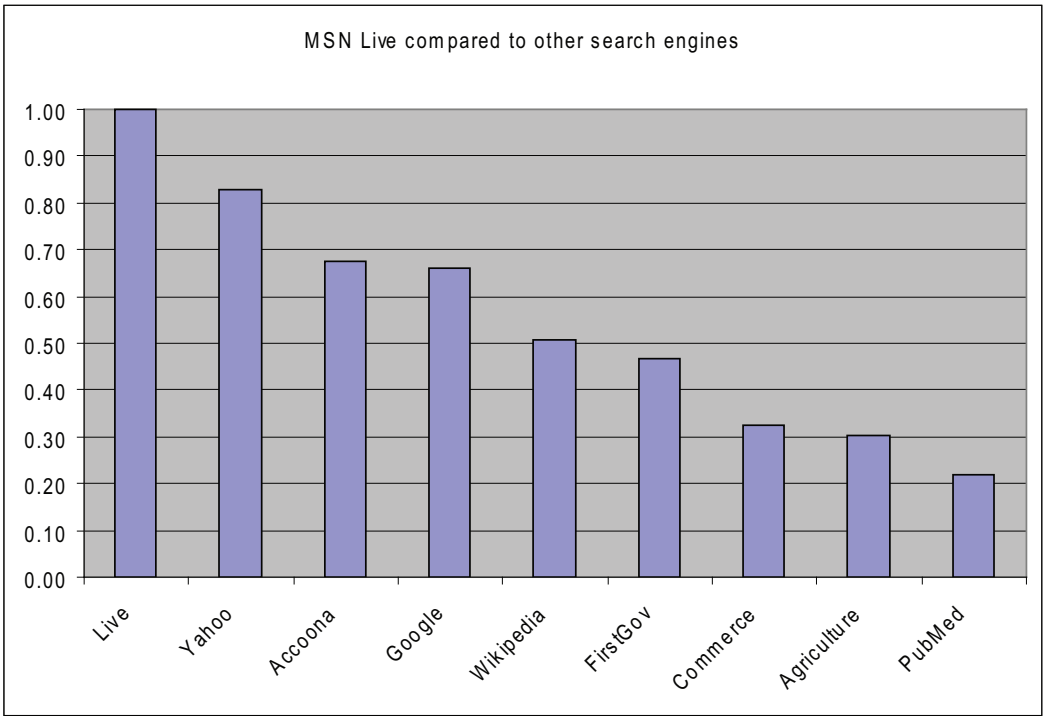


Figure 7.32: Similarity measure of MSN to other search engines. The y-axis represents how related the search engines are to each other.

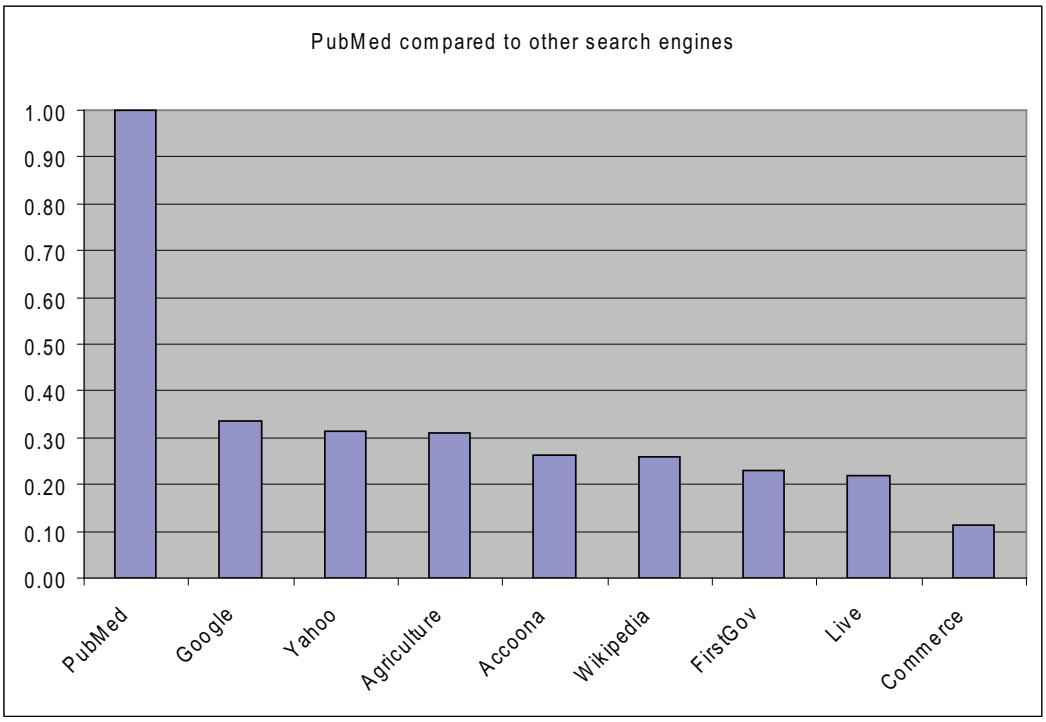


Figure 7.33: Similarity measure of PubMed to other search engines. The y-axis represents how related the search engines are to each other.

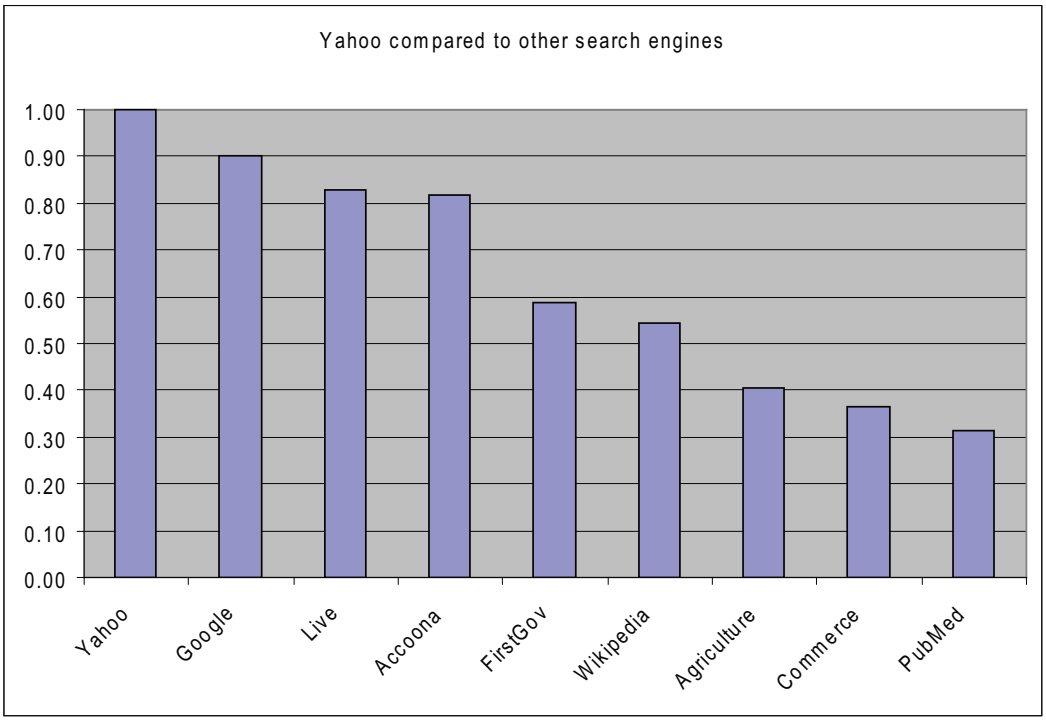


Figure 7.34: Similarity measure of Yahoo to other search engines. The y-axis represents how related the search engines are to each other.

7.3.3 Summary

Singular value decomposition was used to analyse the correlation of the search engines. A graphical representation of the results was shown. This method can be used to give a user a quick and simple way of selecting the most relevant collections for an information need. This method can also be extended to include the query string, which will then show the most relevant collections for a query.

7.4 Summary

This chapter gave the results of the experiments. The method performed well in experiments against ReDDE, the current state of the art collection selection method. A set of the largest and most popular search engines in common used was selected, and the method was used to perform a detailed taxonomy based comparison of the engines. Singular value decomposition was also used to perform a high level pattern comparison of the search engines. For the analysis

of the third level only a small number of results could be shown as there were many hundreds of results returned. Overall the method performed well and gave excellent results.

Chapter 8

Conclusion and Future Work

The deep web contains a vast number of electronic collections which cannot be indexed by traditional search engines. Due to the number of collections available, methods need to be found which will automatically identify the best collections to use for a particular query. Previous collection selection methods such as GlOSS and CORI required communication between search broker and collections, or software running concurrently on broker and collection. These methods are effective when dealing with cooperative collections, however few deep web collections can be considered cooperative. In this thesis a collection selection technique which can be used to rank uncooperative collections based on relevance to a query was presented. A novel form of collection selection was introduced where queries were enhanced by mapping them to subjects in an ontology; the associated subject classification terms were then employed to retrieve collections. This ontology-based method was compared to ReDDE, currently the most advanced collection selection method, and the ontology-based method performed well across most of the experiments, and was also more efficient than all the other high performing methods. This gives support to the hypothesis that an ontology-based approach mitigates the need for collection size estimates in collection selection.

A number of contributions to the field of ontologies, web intelligence(WI) and search engine content analysis were made. The first being the creation of a large multi-domain ontology for representation of world knowledge for Web Intelligence. Another contribution is a method for mining classification terms from an ontology. This is the first time that an ontology based

collection selection method has produced good results. Another contribution is a taxonomy based method for analysing search engine content across multiple levels. Another contribution is a novel method of evaluating search engines using singular value decomposition.

After proving that the ontology-based collection selection method works in a controlled setting, it was then used to analyse some of the largest search engines in use today. In this researchers' previous work it was proven that singular value decomposition is an effective method for analysing collections [Kin03]. However the previous method used was based on statistics rather than on an ontology, and it was also performed at run-time rather than in a previous iteration.

This thesis has drawn from the fields of collection selection, taxonomies, ontologies, ontology learning, data mining, and singular value decomposition. Each of these areas were briefly outlined throughout this thesis. This research developed and applied of a set of methods which allows analysis of large collections across hundreds of different subjects by using the IntelliOnto ontology. Analysing a collection without having direct access to it's index is difficult, particularly if the collection covers a broad range of subjects. The experiments performed showed that this analysis is effective. A feature of the research is that it made use of expert human knowledge for the purpose of classification of resources. A method for ontology-based collection selection which does not rely on estimation of collection sizes was presented. A method for the analysis of the similar patterns between the content of collections using singular value decomposition was also given. The singular value decomposition method calculates how similar different collections are to each other.

The collections and search engines discussed in this research were all web based, with the only interface being a HTML form where a query is entered, and a set of HTML web pages which comprise the results of the query. This ontology based method can also be applied to text documents for the purpose of automatic classification of subject matter.

An interesting finding was that terms which occur in standard dictionaries tend to be multi-purpose and were not much use for classification, while terms that are good for classification are single purpose and were generally too specialised for inclusion in standard English dictionaries.

A major cause of difficulty with the method was reliance on the Q.U.T. Library search engine to produce good results. If the search engine produced a bad set of results in the top four returned items, the entire experiment was tainted.

In further work, the fourth level of the Dewey Decimal hierarchy will be used with the goal of improving precision. Another problem was the precision of the system which maps the query to its Dewey Decimal codes. Both of these tasks are interrelated; as the subjects become more constrained, the importance of a precise mapping of the query to its Dewey Decimal code increases. The query-based sampling method presented in this thesis also supports phrases, which if exploited would almost certainly improve the quality of the results. Another improvement would be exploiting the multi-subject feature of the library dataset, as some documents were assigned to more than one subject. This may result in more precise classifications.

Appendix A

Query Sample Terms

This chapter contains a sample of some of the query sample terms used to profile each search engine. Note that in a few cases a term may be repeated across different subjects. Also note that the optimal settings used for the collection selection experiments were used for the search engine analysis so many subjects did not have enough data in order be included in this list. Some subjects had terms that were too rare to be of use in the classification of documents. Also note that the English language is constantly changing and evolving, terms that are widely used within subjects now may be replaced by other terms within the space of a few short years. As an example of this, compare a computing dictionary from the 1970 with a computing dictionary from 2008. The terms used are hugely different but they still belong to the same subject. This property of ontology based collection selection is why we dynamically generate classification terms, rather than having an expert select a set of classification terms which would be too expensive as the list of classification terms would have to be frequently updated as each subject evolves and grows. In future work the algorithm used to generate these terms could be refined to modify itself based on the different document sizes and depths of each subject and to give more weight to terms which occurred in more recent publications (library catalogues almost always includes the publication date of each document). This would keep the index more up to date and would allow better classification terms to be generated.

The format of the list is:

Dewey Decimal code : sample term

The description of each Dewey Decimal code can be found online at:

http://www.tnrdlib.bc.ca/dewey.html

001:cryptography
001:cybernetics
001:nonverbal
001:consulting
003:cryptography
003:cybernetics
003:dynamical
003:fuzzy
004:ccna
004:ipv6
004:ethernet
004:dreamweaver
005:javabeans
005:javaserver
005:j2ee
005:applets
006:3ds
006:actionscript
006:shockwave
006:lightwave
011:lists
011:publications
011:indexes
015:imprints
015:publications
015:indexes
020:librarianship
020:librarian
020:librarians
020:acronyms
021:librarianship
021:librarians
021:bibliographic
023:librarians
023:technicians
023:descriptions
025:aacr
025:aacr2
025:oclc
025:interlibrary
026:librarianship
027:librarianship
027:librarian
027:parliamentary
027:handicapped
028:illustrators

028:librarians
028:lists
029:indexing
069:museums
070:photojournalism
070:correspondents
070:reporters
070:journalistic
071:newspapers
071:newspaper
072:newspapers
079:newspapers
082:quotations
091:manuscripts
100:introductions
110:metaphysics
111:heidegger
111:sublime
111:ontology
111:kant
121:hermeneutics
121:epistemology
121:belief
121:truth
126:consciousness
128:consciousness
128:reason
128:philosophical
133:parapsychology
133:occultism
133:shui
133:witches
142:existentialism
142:phenomenology
144:humanism
149:structuralism
149:semantics
150:psychoanalysts
150:behaviorism
150:psychometrics
150:skinner
152:neuropsychological
152:psychophysiology
152:neuropsychology
152:auditory
153:metacognition
153:intelligences
153:neuropsychology
153:psycholinguistics
154:hypnotism
154:hypnosis
154:sleep
154:dreaming
155:enneagram
155:rorschach
155:erikson

155:psychobiology
158:neurolinguistic
158:assertive
158:counselor
158:burnout
160:reasoning
170:kant
170:morals
170:morality
171:utilitarianism
172:morality
174:bioethical
174:accountants
174:euthanasia
174:confidentiality
176:fertilization
176:vitro
176:cloning
176:reproductive
179:euthanasia
179:virtue
179:suicide
179:die
183:socrates
184:plato
185:aristotle
190:philosophers
192:wittgenstein
192:ludwig
192:russell
192:philosophers
193:hegel
193:nietzsche
193:wittgenstein
193:heidegger
194:levinas
194:deleuze
194:descartes
194:sartre
200:religions
200:christianity
220:bible
221:bible
222:noah
230:theology
232:christ
232:jesus
241:catholic
253:pastoral
261:theology
261:catholic
261:christianity
266:missionaries
266:missions
266:mission
270:reformation

270:ca
274:reformation
282:catholic
291:cults
291:religions
291:sacred
291:mythology
292:mythology
294:buddha
294:hinduism
294:zen
294:hindu
296:judaism
296:jewish
297:koran
297:jihad
297:muhammad
297:muslims
299:mythology
300:spss
300:researcher
300:interviewing
300:regression
301:talcott
301:giddens
301:durkheim
301:bourdieu
302:mcluhan
302:intergroup
302:literacies
302:bullying
303:qaida
303:qaeda
303:americanism
303:jihad
304:newborn
304:fertility
304:migrants
304:demography
305:transsexuals
305:housewives
305:geriatrics
305:kakadu
306:stepparents
306:stepfamilies
306:stepchildren
306:remarriage
307:waterfronts
307:seq
307:slums
307:sprawl
320:anarchism
320:hobbes
320:machiavelli
320:authoritarianism
321:republicanism

321:totalitarianism
321:utopias
321:federalism
322:armed
322:protest
322:poland
323:timur
323:timor
323:luther
323:maori
324:electioneering
324:voters
324:votes
324:suffrage
325:detention
325:asylum
325:seekers
325:decolonization
327:antinuclear
327:disarmament
327:spies
327:espionage
328:legislators
328:electoral
328:commons
328:districts
330:schumpeter
330:keynesian
330:cointegration
330:microeconomic
331:telecommuting
331:federated
331:absenteeism
331:conciliation
332:arbitrage
332:swaps
332:bretton
332:keynesian
333:daintree
333:shoalwater
333:riparian
333:landcare
334:cooperative
335:anarchism
335:engels
335:gramsci
335:marxian
336:eligible
336:gst
336:superannuation
336:debts
337:apec
337:enlargement
337:wto
337:asean
338:jabiluka

338:spillovers
338:fdi
338:oligopoly
339:keynesian
339:macroeconomics
339:macroeconomic
339:wage
340:jurisprudence
340:lawyers
340:rule
341:peacekeeping
341:treaties
341:ec
341:treaty
342:constitutions
342:judicial
342:privacy
342:liability
343:gst
343:superannuation
343:antitrust
343:arbitration
344:conciliation
344:arbitration
344:dismissal
344:malpractice
345:bail
345:sentencing
345:jury
345:offenders
346:wik
346:debtor
346:creditor
346:mabo
347:arbitration
347:jury
347:judges
347:judicial
348:statutes
348:indexes
350:bureaucracy
351:cape
351:peninsula
352:contracting
352:municipal
352:budget
352:accountability
353:presidents
353:bureaucracy
353:dept
354:auditor
354:appropriations
354:intergovernmental
354:expenditures
355:militarism
355:bomb

355:defenses
355:soldiers
359:navy
359:naval
361:casework
361:generalist
361:voluntarism
361:philanthropy
362:respite
362:adoptees
362:hospices
362:sniffing
363:nohsc
363:fatalities
363:policewomen
363:ohs
364:recidivism
364:criminologists
364:criminological
364:burglary
365:inmates
365:correctional
365:penal
365:imprisonment
368:actuarial
368:medicare
368:pension
368:pensions
370:microteaching
370:piaget
370:freire
370:intelligences
371:truancy
371:nongraded
371:principalship
371:summerhill
372:preprimary
372:kodaly
372:reggio
372:phonics
373:dropouts
373:principals
373:oecd
373:schooling
374:numeracy
374:tafe
374:compulsory
374:learner
375:folios
375:specification
375:module
375:syllabus
377:catholic
378:dropouts
378:lecturing
378:scholarships

404:bilingualism
404:bilingual
410:noam
410:chomsky
410:linguistic
411:alphabet
412:semantics
414:phonology
414:phonetics
415:generative
415:syntax
415:semantics
415:linguistic
418:l2
418:proficiency
418:esl
418:communicative
419:deaf
419:sign
421:phonics
421:punctuation
421:orthography
421:phonetic
422:etymology
423:antonyms
423:synonyms
423:idioms
423:thesaurus
425:generative
425:syntax
427:slang
427:dialects
427:variation
428:toefl
428:ielts
428:headway
428:tesol
438:fremdsprache
438:als
438:deutsch
438:fur
443:dictionnaire
448:francais
448:spoken
448:en
448:le
489:athanasioy
489:ioannoy
489:barbara
491:russian
495:nihongo
495:phrase
495:vietnamese
495:thai
499:bahasa
499:indonesian

526:photogrammetric
526:hydrographic
526:geodetic
526:photogrammetry
529:clocks
530:schrodinger
530:astrophysics
530:amorphous
530:angular
531:rheology
531:elasticity
531:elastic
531:vibration
532:viscous
532:hydrodynamics
532:turbulence
532:fluids
534:acoustics
534:waves
535:interferometry
535:raman
535:diffraction
535:scattering
536:thermodynamics
536:thermal
536:temperature
536:cold
537:electrodynamics
537:dielectrics
537:superconductivity
537:superconductors
538:paramagnetic
538:magnets
538:magnetism
538:electron
539:quarks
539:neutrons
539:accelerators
539:dosimetry
540:chemists
540:alchemy
541:orbitals
541:stereochemistry
541:valence
541:colloid
542:laboratories
543:chemometrics
543:supercritical
543:hplc
543:reagents
544:chromatographic
544:chromatography
544:analytic
545:spectrometry
545:analytic
545:quantitative

546:phosphorus
546:complexes
546:inorganic
546:periodic
547:heterocyclic
547:alkaloids
547:stereochemistry
547:organometallic
548:crystallography
548:diffraction
548:crystals
548:rays
549:determinative
549:petrology
549:mineralogy
549:rocks
550:geophysics
550:geological
550:sensing
550:planet
551:tsunamis
551:glaciers
551:paleoclimatology
551:volcanism
552:metamorphism
552:diagenesis
552:sedimentology
552:metamorphic
553:sedimentary
553:stratigraphic
553:ores
553:ore
559:stratigraphic
559:geological
559:basin
560:paleoecology
560:fossils
560:paleontology
560:fossil
561:paleobotany
562:fossil
567:dinosaur
567:paleontology
567:dinosaurs
567:fossil
570:bioinformatics
570:microscopy
571:apoptosis
571:transduction
571:cytology
571:membrane
572:microarrays
572:proteomics
572:rna
572:bioinformatics
573:origin

574:bioorganic
574:histochemistry
574:immunochemistry
574:bioinorganic
575:heredity
575:naturalists
575:genes
575:gene
576:virology
576:rna
576:microbes
576:microbiological
577:wetland
577:biotic
577:wetlands
577:freshwater
578:biogeography
578:staining
578:microscope
578:microscopy
579:fungi
579:bacterial
579:microorganisms
579:bacteria
580:botanic
580:botanical
580:botany
580:gardens
581:phytogeography
581:ethnobotany
581:medicinal
581:photosynthesis
582:wildflowers
582:angiosperms
582:flowering
582:seeds
583:eucalypts
583:eucalyptus
584:orchids
584:grasses
589:mycology
589:mushrooms
589:lichens
589:yeast
590:zoology
591:camouflage
591:parasitology
591:poisonous
591:embryology
592:invertebrate
592:invertebrates
593:thorns
593:protozoa
593:reef
593:crown
594:mollusks

594:shells
595:arthropoda
595:beetles
595:termites
595:moths
596:retina
596:vertebrate
596:vertebrates
597:amphibians
597:lizards
597:sharks
597:crocodiles
598:parrots
598:birds
598:bird
599:marsupials
599:lymphocytes
599:bats
599:platypus
602:coke
602:coal
602:specification
604:drafting
604:hazardous
604:recycling
604:substances
608:inventors
608:patent
608:patents
608:inventions
609:inventors
609:inventions
610:perioperative
610:oncologic
610:biomaterials
610:hematologic
611:ankle
611:cytogenetics
611:neuroanatomy
611:spinal
612:neurochemistry
612:menopause
612:lactation
612:cytokines
613:dietitians
613:lactation
613:contraceptives
613:vegetarianism
614:measles
614:immunisation
614:nosocomial
614:influenza
615:brachytherapy
615:infusions
615:analgesics
615:pharmacodynamics

616:myeloproliferative
616:brachytherapy
616:immunodiagnosis
616:neoplasm
617:phacoemulsification
617:lasik
617:biomicroscopy
617:keratotomy
618:strabismus
618:neonatology
618:cesarean
618:gestational
620:abaqus
620:autocad
620:maintainability
620:prestressed
621:pspice
621:verilog
621:microstrip
621:switchgear
622:ranger
622:geochemical
622:ores
622:ore
623:radar
623:boats
623:naval
623:orlando
624:geomechanics
624:girder
624:tunnelling
624:eurocode
625:subgrades
625:skidding
625:embankments
625:skid
627:breakwaters
627:spillways
627:dredging
627:hydraulics
628:incineration
628:bioremediation
628:effluent
628:landfill
629:servomechanisms
629:astrodynamics
629:crashworthiness
629:supersonic
630:crops
630:farming
630:extension
630:farm
631:hydroponics
631:dryland
631:salinization
631:burdekin

632:insecticides
632:herbicides
632:feral
632:pesticide
633:forage
633:sugarcane
633:wheat
633:crops
634:agroforestry
634:rirdc
634:eucalyptus
634:olive
635:shrubs
635:ornamental
635:horticulture
635:planting
636:chickens
636:goats
636:poultry
636:beef
637:cheese
637:dairy
637:milk
638:honeybee
638:honey
639:whaling
639:aquaculture
639:fishery
639:fisheries
641:baking
641:cholesterol
641:herbs
641:seafood
642:caterers
642:catering
643:dwellings
646:dressmaking
646:sewing
646:tailoring
646:fabrics
647:foodservice
647:restaurant
647:restaurants
647:hotels
649:lactation
649:breastfeeding
649:toddler
649:parenthood
650:sap
650:rã©sumã©s
650:resumes
650:resume
651:secretarial
651:filing
651:secretaries
651:telephone

652:typewriting
652:keyboarding
652:wordperfect
652:penmanship
653:pitman
653:shorthand
657:myob
657:fasb
657:gaap
657:bookkeeping
658:controllership
658:buyouts
658:pert
658:14000
659:stash
659:commercials
659:typography
659:advertisers
660:catalysis
660:polymerization
660:fermentation
660:reactors
661:solvents
661:oils
661:petroleum
661:chemicals
662:liquefaction
662:explosives
662:biomass
662:fuels
663:brewing
663:wine
664:flavoring
664:essences
664:flavor
664:beverages
665:lubricants
665:oils
665:hydrogen
665:fuels
666:portland
666:additives
666:ceramic
666:ceramics
667:dyeing
667:coating
667:dyes
667:coatings
668:epoxy
668:surfactants
668:extrusion
668:polymerization
669:metallography
669:asm
669:steels
669:metallurgical

670:cad
670:manufactures
670:robotics
670:robots
671:plating
671:castings
671:welded
671:machining
672:welding
672:iron
673:alloys
673:aluminum
673:copper
674:timber
674:wood
676:papermaking
676:pulp
676:wood
677:fibres
677:wool
677:fibers
677:textiles
678:elastomers
678:rubber
681:transducers
681:sensor
681:vessels
681:detectors
684:woodworking
684:woodwork
684:furniture
686:typesetting
686:bookbinding
686:latex
686:typography
687:merchandising
687:clothing
688:packaging
688:toys
690:superintendence
690:roofing
690:roofs
690:masonry
691:timber
691:wood
691:stone
692:surveyors
692:contractors
692:quantity
692:specification
693:soundproofing
693:fireproof
693:dampness
693:brickwork
694:joinery
694:carpentry

694:wooden
694:timber
695:roofing
695:roofs
696:plumbing
696:drainage
696:heating
697:ventilating
697:hvac
697:refrigerating
697:refrigeration
700:viola
700:surrealism
700:installations
700:patronage
701:avant
701:garde
701:aesthetic
701:appreciation
702:collage
702:artist
704:nude
704:arnhem
704:buddhist
704:symbolism
708:galleries
708:gallery
708:museums
709:rauschenberg
709:dadaism
709:nauman
709:beuys
711:ptrc
711:malls
711:pedestrian
711:urbanism
712:landscaping
712:planting
712:gardening
712:gardens
715:trees
720:neoclassicism
720:skyscraper
720:rohe
720:mies
721:tall
721:aided
721:walls
721:glass
722:roman
723:romanesque
723:gothic
724:gaudi
724:aalto
724:alvar
724:rohe

725:pools
725:theaters
725:stores
725:opera
726:cathedrals
726:temples
726:cathedral
726:gothic
727:museums
728:apartments
728:apartment
728:hotels
728:vernacular
729:ornament
729:acoustics
729:decorative
730:gormley
730:rodin
730:auguste
730:sculptor
731:masks
738:glazes
738:potters
738:kilns
738:porcelain
739:silverwork
739:ironwork
739:jewellery
739:jewelry
741:typography
741:pencil
741:comics
741:illustrators
743:figure
745:calligraphy
745:lettering
745:deco
745:puppet
746:rugs
746:applique
746:quilting
746:chanel
747:interiors
748:glassware
748:glass
749:chairs
749:chair
749:furniture
750:paintings
751:watercolor
759:namatjira
759:schiele
759:titian
759:bruegel
760:printmaking
760:prints

761:engraving
761:wood
763:lithography
769:ukiyoe
769:etchings
769:printmakers
769:engraving
770:photographer
770:photographers
770:photograph
770:camera
771:realia
771:photographic
776:photoshop
776:adobe
776:maya
776:videos
778:cinematographers
778:avid
778:dv
778:photojournalism
779:photographers
779:photograph
779:photographic
779:portraits
780:sibelius
780:kodaly
780:britten
780:ethnomusicology
781:reggae
781:montreux
781:vocals
781:musicianship
782:oratorios
782:requiems
782:oratorio
782:librettos
783:vocal
783:singing
783:sight
783:song
784:concerti
784:orchestre
784:konzert
784:grossi
785:clarinets
785:violinen
785:woodwind
785:violine
786:marimba
786:pianists
786:klavier
786:chopin
787:violine
787:violoncello
787:concertos

787:sonatas
788:sonaten
788:trombone
788:saxophone
788:oboe
790:playgrounds
790:recreational
790:toys
790:outdoor
791:kurosawa
791:godard
791:cirque
791:scorsese
792:ailey
792:rambert
792:fonteyn
792:bournonville
793:ballroom
793:salsa
793:tango
793:dancer
794:chess
794:3d
794:players
794:adventure
796:netball
796:squash
796:karate
796:badminton
797:canoes
797:canoeing
797:swim
797:sailing
799:archery
799:fishing
801:structuralism
801:derrida
801:deconstruction
801:semiotics
803:terminology
808:scriptwriting
808:screenwriting
808:screenwriters
808:playwriting
809:romanticism
809:horror
809:detective
809:comedy
810:bio
810:canadian
811:plath
811:whitman
811:ezra
811:dickinson
812:albee
812:mamet

812:playwrights
812:dramatists
813:steinbeck
813:nabokov
813:hemingway
813:poe
820:illustrators
820:romanticism
820:imperialism
820:bio
821:poetical
821:slessor
821:tennyson
821:donne
822:andronicus
822:stoppard
822:coriolanus
822:shrew
823:brontã≪
823:wuthering
823:trollope
823:jolley
826:letters
827:wit
827:humor
828:wit
828:prose
828:humor
832:bertolt
832:brecht
832:translations
833:kafka
833:mann
833:franz
833:roman
839:strindberg
839:ibsen
839:henrik
839:translations
841:baudelaire
842:godot
842:moliere
842:beckett
842:translations
843:les
843:du
843:le
843:roman
848:beckett
848:samuel
851:alighieri
851:dante
852:pirandello
882:aristophanes
882:euripides
882:sophocles

882:tragedy
883:iliad
883:homer
891:dostoyevsky
891:fyodor
891:chekhov
891:tolstoy
895:haiku
895:korean
895:translations
899:indonesian
901:historiography
904:disasters
907:historiography
907:historians
909:crusades
909:mediterranean
909:arab
909:islamic
910:arcview
910:arcgis
910:geographers
910:cartography
911:cartographic
912:topographic
912:landforms
912:cartography
912:tourist
915:mekong
915:tibet
915:bali
915:guidebooks
919:sketchbook
919:polynesia
919:antarctic
919:reef
920:autobiography
920:biographical
929:heraldry
929:flags
929:genealogy
929:names
930:civilizations
930:archaeological
930:prehistoric
930:antiquities
932:antiquities
932:egypt
936:antiquities
936:roman
937:emperors
937:caesar
937:augustus
937:romans
938:peloponnesian
938:greeks

938:athens
938:antiquities
940:kokoda
940:regimental
940:battalion
940:gallipoli
941:stuarts
941:ulster
941:churchill
941:ministers
942:tudors
942:tudor
942:kings
942:anglo
943:prussia
943:bismarck
943:reunification
943:weimar
944:napoleon
944:emperor
944:rulers
944:xiv
945:mussolini
945:medici
945:fascism
945:venice
946:spanish
947:sergeevich
947:gorbachev
947:lenin
947:stalin
948:vikings
949:hercegovina
949:kosovo
949:balkan
949:serbia
951:tiananmen
951:tse
951:tibet
951:tung
952:tokugawa
952:meiji
952:tokyo
953:saudi
953:arabia
954:mahatma
954:gandhi
954:bangladesh
954:pakistan
955:iran
956:palestine
956:israeli
956:persian
956:iraq
958:afghanistan
959:angkor

959:malayan
959:viet
959:hanoi
962:egypt
963:ethiopia
966:nigeria
967:rwanda
968:apartheid
970:columbus
970:antiquities
970:christopher
970:native
971:canadian
972:aztecs
972:panama
972:nicaragua
972:guatemala
973:roosevelt
973:nixon
973:terrorist
973:abraham
977:chicago
979:calif
979:angeles
979:los
979:francisco
981:brazil
983:chile
985:incas
985:peru
993:maori
994:stockade
994:squatters
994:tasmanians
994:menzies
995:bougainville
995:vanuatu
995:caledonia
995:solomon
996:tuvalu
996:hawaiians
996:micronesia
996:tonga
998:antarctica

Appendix B

Singular Value Decomposition - Tool of SECA

This section explains *SVD (singular value decomposition)*, a powerful matrix factorization method. In this research SVD is used as a minor part of the ontology based method to compare how similar collections and search engines are to each other. This allows related collections to be grouped together based on the concepts that they contain.

B.0.1 Definitions

This section introduces some definitions used in calculating the singular value decomposition of a matrix.

B.0.1.1 Transpose

The *transpose* of a matrix is obtained by exchanging a matrix's rows with its columns. This is achieved by replacing all elements a_{jk} with a_{kj}. The superscript letter $T(^T)$ is used to denote the transpose of a matrix.

B.0.1.2 Identity Matrix

An *identity matrix* is a simple diagonal square matrix of the form:

$$I = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

We use the letter I to denote a identity matrix.

B.0.1.3 Orthogonal

A matrix A is *orthogonal* if:

$$AA^T = I \tag{B.1}$$

where I is an identity matrix.

Two vectors are **orthogonal** if their dot product is 0, c.e. $x^T y = 0$.

Vectors are called **pairwise orthogonal** if any two of them are orthogonal.

B.0.1.4 Determinants

Determinants can be used for solving systems of linear equations, and can be used to test for matrix singularity, where the determinant will be zero if the matrix is singular.

To calculate the determinant of a 2 by 2 matrix:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

we use:

$det(A) = a_{11}a_{22} - a_{12}a_{21}$

For a general *n-by-n* matrix, the determinant is calculated using Leibniz formula:

$$\det(A) = \sum_{\sigma \in S_n} sgn(\sigma) \prod_{i=1}^{n} A_{i,\sigma(i)} \tag{B.2}$$

Where S_n is the set of all permutations, sgn(σ) denotes the signature of the permutation σ: +1 if σ is an even permutation and -1 if it is odd. The sum is computed over all permutations σ of the numbers 1,...,n.

det is used to to denote the determinant of a matrix.

B.0.1.5 Eigenvectors and Eigenvalues

Eigenvalues and *eigenvectors* are used in physics, mechanics, and many other fields. Eigenvalues and eigenvectors are also important for calculating Singular Value Decomposition.

Eigenvectors are the vectors that are preserved in direction under matrix multiplication. This is an important property that helps users to see the base components of the matrix.

Given a matrix A, its eigenvalue λ can be determined using the following equation.

$$det(A - \lambda I) = 0 \tag{B.3}$$

where I is the identity matrix.

The number λ in Equation B.3 is called the *eigenvalue* of *A* [Wat91]. To calculate the eigenvalues the equation is solved for λ. Eigenvalues are sometimes also known as characteristic roots, proper values, or latent roots. Each eigenvalue is paired with a corresponding eigenvector. The set of all eigenvectors corresponding to an eigenvalue A, including 0, forms a vector space called the *eigenspace* of A.

Given a λ, its eigenvector X can be determined using the following equation.

$$(A - \lambda)X = 0 \tag{B.4}$$

To calculate the eigenvectors the equation is solved for λ. The decomposition of a matrix into eigenvalues and eigenvectors is known as eigen decomposition.

For example the following shows how the eigenvalues and eigenvectors of a matrix are calculated.

Let:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$det(A - \lambda I) = det \begin{bmatrix} 1-\lambda & 2 & 3 \\ 4 & 5-\lambda & 6 \\ 7 & 8 & 9-\lambda \end{bmatrix}$$

It can be verified that there are three eigenvalues. They are:

λ_1 = 16.1168, λ_2 = -1.1168, and λ_3 = 0.

The corresponding eigenvectors of *A* are:

$$v_1 = \begin{bmatrix} -0.232 \\ -0.5253 \\ -0.8187 \end{bmatrix} v_2 = \begin{bmatrix} -0.7858 \\ -0.0868 \\ 0.6123 \end{bmatrix} v_3 = \begin{bmatrix} 0.4082 \\ -0.8165 \\ 0.4082 \end{bmatrix}$$

B.0.1.6 Gram-Schmidt Orthonormalisation

A set of vectors $v_1, v_2, ..., v_k \in R^n$ is called *orthonormal* if they are pairwise orthogonal, and each vector has a Euclidean norm 1:

$$v_i^T v_j = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

It is often convenient and simpler to use an orthonormal basis when doing some matrix calculations.

The Gram-Schmidt algorithm is used to find the orthogonal basis of a matrix. After the orthogonal basis have been found, the orthonormal basis are calculated by normalising each vector in the orthogonal basis.

To calculate u_i, we project v_i orthogonally onto the subspace generated by $u_1, ..., u_{i-1}$.

To do this start with linearly independent vectors $v_1, ..., v_k$ and find mutually orthogonal vectors $u_1, ..., u_k$ which occupy the same subspace as the vectors $v_1, ..., v_k$.

In the following Gram-Schmidt algorithm, the dot product of *v* and *u* is represented as $(v.u)$:

$u_1 = v_1$

$u_2 = v_2 - [(v_2.u_1)/(u_1.u_1)]u_1$

$u_3 = v_3 - [(v_3.u_1)/(u_1.u_1)]u_1 - [(v_3.u_2)/(u_2.u_2)]u_2$

...

$u_k = v_k - [(v_k.u_1)/(u_1.u_1)]u_1 - [(v_k.u_2)/(u_2.u_2)]u_2 - ... - [(v_k.u_{k-1})/(u_{k-1}.u_{k-1})]u_{k-1}$

B.0.2 Singular Value Decomposition

Singular Value Decomposition is a matrix factorisation and dimension reduction method that has many uses: eg., information retrieval (where it is known as "latent semantic analysis", time series analysis, and pattern matching).

B.0.2.1 Properties of Singular Value Decomposition

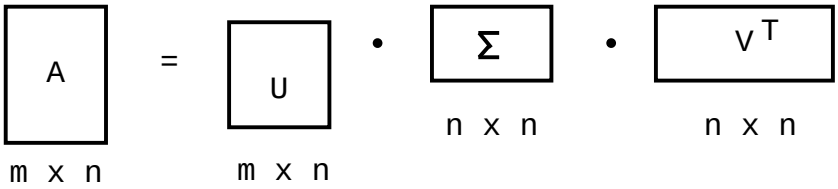


Figure 5.1: Matrix Decomposition

Figure 5.1 shows the decomposition of the $m \times n$ matrix *A*, where $m \geq n$.

A is a matrix that represents collections such that the rows are terms and columns are collections (i.e. vectors). The *Singular Value Decomposition* of *A* is said to be the factorisation:

$$A = U \sum V^T \tag{B.5}$$

where the diagonal of $\sum$ is said to be the singular values of the original matrix, *A*:

$$\sum = \begin{bmatrix} w_0 & 0 & 0 & 0 \\ 0 & w_1 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & w_{n-1} & 0 \\ 0 & 0 & 0 & w_n \end{bmatrix}$$

and

$$U^T U = V^T V = I \tag{B.6}$$

	C1	C2	C3	C4	C5
appear	1	4	4	4	0
brief	0	4	3	2	0
extensive	3	4	2	2	0
language	4	4	0	1	1
object	0	2	0	0	1
self	0	0	4	0	0
year	1	1	3	0	0

Table B.1: The Term-Frequency Table

$$w_1, w_2, ..., w_{n-1}, w_n \geq 0 \tag{B.7}$$

In Equation B.5, matrices *U* and *V* are orthogonal. The columns of U are called left singular values (terms) and the rows of V^T are called right singular values (collection vectors). In this research, only the right singular values, which are also commonly represented as V^T, are used.

The orthogonal matrices *V* and *U* are formed by the eigenvectors of the matrices $A^T A$ and AA^T respectively. The dimensions of *A* are reduced to a smaller set of eigenvectors that are closest to the original matrix. This dimension reduction produces a clustering effect in the reduced matrix, removing noise from the matrix and bringing together related concepts.

B.0.2.2 Singular Value Decomposition Example

Table B.1 illustrates an example term-collection matrix. There are five collections; each collection has a column(vector) to show the term frequencies.

The following is the procedure of computing SVD (see [ITL]).

The collection correlation matrix is generated first by calculating SVD on the Matrix A shown in Table 5.1, then by normalising and transforming the resulting matrix.

(I) Calculating the eigenvalues of the matrix $A^T A$ and arrange them in descending order, assigning each one to a lambda variable λ_1 to λ_5 gives $\lambda_1 = 134.4504$, $\lambda_2 = 29.394$, $\lambda_3 = 8.9824$, $\lambda_4 = 3.6183$, and $\lambda_5 = 0.5548$

(II) Calculating the number of nonzero eigenvalues of the matrix $A^T A$ results in $r = 5$.

(III) Finding the orthogonal eigenvectors of the matrix $A^T A$ corresponding to the obtained eigenvalues, and arranging them in the same order to form the column-vectors of the matrix $X \in R^{n \times n}$ give the following vectors v_1 to v_5

$$v_1 = \begin{bmatrix} 0.0356 \\ -0.2939 \\ 0.0477 \\ 0.3251 \\ 0.8969 \end{bmatrix} \quad v_2 = \begin{bmatrix} 0.4014 \\ -0.4873 \\ -0.0645 \\ 0.6553 \\ -0.4097 \end{bmatrix} \quad v_3 = \begin{bmatrix} 0.6785 \\ -0.2899 \\ 0.3768 \\ -0.5568 \\ 0.0598 \end{bmatrix}$$

$$v_4 = \begin{bmatrix} -0.5203 \\ -0.3417 \\ 0.7669 \\ 0.0469 \\ -0.1491 \end{bmatrix} \quad v_5 = \begin{bmatrix} -0.3264 \\ -0.6894 \\ -0.5133 \\ -0.3908 \\ -0.044 \end{bmatrix}$$

Hence the matrix V is the result $[v_1, v_2, v_3, v_4, v_5]$, resulting in:

$$V = \begin{bmatrix} 0.0356 & 0.4014 & 0.6785 & -0.5203 & -0.3264 \\ -0.2939 & -0.4873 & -0.2899 & -0.3417 & -0.6894 \\ 0.0477 & -0.0645 & 0.3768 & 0.7669 & -0.5133 \\ 0.3251 & 0.6553 & -0.5568 & 0.0469 & -0.3908 \\ 0.8969 & -0.4097 & 0.0598 & -0.1491 & -0.044 \end{bmatrix}$$

(IV) Then form the diagonal matrix $\sum$ by taking square roots of the eigenvalues and sorting in descending order in a diagonal direction of $\delta_i = \sqrt{\lambda_i}$.

$$\sum = \begin{bmatrix} 11.5953 & 0 & 0 & 0 & 0 \\ 0 & 5.4216 & 0 & 0 & 0 \\ 0 & 0 & 2.9971 & 0 & 0 \\ 0 & 0 & 0 & 1.9022 & 0 \\ 0 & 0 & 0 & 0 & 0.7448 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(V) Then find the first 5 column-vectors of the matrix U using the following equation:

$$u_i = \delta_i^{-1} A v_i (i = 1 : r) \tag{B.8}$$

$$u_1 = \begin{bmatrix} -0.5779 \\ -0.438 \\ -0.4782 \\ -0.3879 \\ -0.1227 \\ -0.1771 \\ -0.2204 \end{bmatrix} \quad u_2 = \begin{bmatrix} -0.2523 \\ -0.1896 \\ 0.2398 \\ 0.6548 \\ 0.1535 \\ -0.5658 \\ -0.2654 \end{bmatrix} \quad u_3 = \begin{bmatrix} -0.4008 \\ -0.3813 \\ 0.1721 \\ 0.3528 \\ -0.1735 \\ 0.5029 \\ 0.5068 \end{bmatrix}$$

$$u_4 = \begin{bmatrix} 0.4286 \\ -0.4375 \\ 0.2294 \\ -0.0516 \\ -0.7278 \\ -0.1356 \\ -0.1469 \end{bmatrix} \quad u_5 = \begin{bmatrix} -0.4711 \\ 0.5136 \\ 0.4342 \\ -0.2532 \\ -0.4148 \\ -0.2561 \\ 0.1548 \end{bmatrix}$$

(VI) Then add to the matrix U the rest of $m - r$ vectors u_6 and u_7 using the Gram-Schmidt orthagonalisation algorithm described in Subsection B.0.1.6 and using $u_1, ..., u_5$.

$$u_6 = \begin{bmatrix} -0.1321 \\ -0.1945 \\ 0.6018 \\ -0.2862 \\ 0.2862 \\ 0.3734 \\ -0.5284 \end{bmatrix} \quad u_7 = \begin{bmatrix} 0.1368 \\ -0.3657 \\ 0.2843 \\ -0.3842 \\ 0.3842 \\ -0.4149 \\ 0.5471 \end{bmatrix}$$

Hence U = $[u_1, u_2, u_3, u_4, u_5, u_6, u_7]$

$$U = \begin{bmatrix} -0.5779 & -0.2523 & -0.4008 & 0.4286 & -0.4711 & -0.1321 & 0.1368 \\ -0.438 & -0.1896 & -0.3813 & -0.4375 & 0.5136 & -0.1945 & -0.3657 \\ -0.4782 & 0.2398 & 0.1721 & 0.2294 & 0.4342 & 0.6018 & 0.2843 \\ -0.3879 & 0.6548 & 0.3528 & -0.0516 & -0.2532 & -0.2862 & -0.3842 \\ -0.1227 & 0.1535 & -0.1735 & -0.7278 & -0.4148 & 0.2862 & 0.3842 \\ -0.1771 & -0.5658 & 0.5029 & -0.1356 & -0.2561 & 0.3734 & -0.4149 \\ -0.2204 & -0.2654 & 0.5068 & -0.1469 & 0.1548 & -0.5284 & 0.5471 \end{bmatrix}$$

and the singular value decomposition of A is:

$$U = \begin{bmatrix} -0.5779 & -0.2523 & -0.4008 & 0.4286 & -0.4711 & -0.1321 & 0.1368 \\ -0.438 & -0.1896 & -0.3813 & -0.4375 & 0.5136 & -0.1945 & -0.3657 \\ -0.4782 & 0.2398 & 0.1721 & 0.2294 & 0.4342 & 0.6018 & 0.2843 \\ -0.3879 & 0.6548 & 0.3528 & -0.0516 & -0.2532 & -0.2862 & -0.3842 \\ -0.1227 & 0.1535 & -0.1735 & -0.7278 & -0.4148 & 0.2862 & 0.3842 \\ -0.1771 & -0.5658 & 0.5029 & -0.1356 & -0.2561 & 0.3734 & -0.4149 \\ -0.2204 & -0.2654 & 0.5068 & -0.1469 & 0.1548 & -0.5284 & 0.5471 \end{bmatrix}$$

$$\sum = \begin{bmatrix} 11.5953 & 0 & 0 & 0 & 0 \\ 0 & 5.4216 & 0 & 0 & 0 \\ 0 & 0 & 2.9971 & 0 & 0 \\ 0 & 0 & 0 & 1.9022 & 0 \\ 0 & 0 & 0 & 0 & 0.7448 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$V = \begin{bmatrix} 0.0356 & 0.4014 & 0.6785 & -0.5203 & -0.3264 \\ -0.2939 & -0.4873 & -0.2899 & -0.3417 & -0.6894 \\ 0.0477 & -0.0645 & 0.3768 & 0.7669 & -0.5133 \\ 0.3251 & 0.6553 & -0.5568 & 0.0469 & -0.3908 \\ 0.8969 & -0.4097 & 0.0598 & -0.1491 & -0.044 \end{bmatrix}$$

U is orthagonal because:

$$U^TU = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

V is orthagonal because:

$$V^TV = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Next multiply $\sum$ by the transpose of V:

$$\sum V^T = \begin{bmatrix} -6.5613 & -4.2029 & -3.6 & -4.1065 & -0.9723 \\ 0.9076 & -3.0511 & -1.9023 & 3.3503 & -0.0421 \\ 0.5269 & -1.4221 & 1.391 & -0.6459 & 0.1692 \\ 0.9011 & -0.0607 & -0.6355 & -0.6683 & -0.6431 \\ -0.1906 & 0.0171 & 0.2871 & 0.2204 & -0.7815 \end{bmatrix}$$

Now normalise the columns so they all have length 1

$$E = \begin{bmatrix} -0.9781 & -0.7805 & -0.8259 & -0.7626 & -0.6875 \\ 0.1353 & -0.5666 & -0.4364 & 0.6221 & -0.0298 \\ 0.0785 & -0.2641 & 0.3191 & -0.1199 & 0.1196 \\ 0.1343 & -0.0113 & -0.1458 & -0.1241 & -0.4547 \\ -0.0284 & 0.0032 & 0.0659 & 0.0409 & -0.5526 \end{bmatrix}$$

Finally, calculating $F = E^T * E$ gives the collection correlation matrix:

	C1	**C2**	**C3**	**C4**	**Query**
C1	1	0.6644	0.7524	0.8028	0.6325
C2	0.6644	1	0.8094	0.2759	0.5252
C3	0.7524	0.8094	1	0.3408	0.6489
C4	0.8028	0.2759	0.3408	1	0.5252
Query	0.6325	0.5252	0.6489	0.5252	1

The final matrix is the collection correlation matrix and can be used to find the most and least similar collections.

B.0.3 The Meaning Of Singular Value Decomposition

When singular value decomposition is applied to a matrix, the lesser patterns and the background noise are removed, allowing the main patterns of the matrix to be seen. This is caused by selecting the highest singular values, and reducing the lengths of the vectors in space. This can even have the effect of bringing related eigenvectors closer to other eigenvectors which do not use the same terms. The closeness of the patterns of occurrence of words with similar meanings is what helps to resolve polysemy and synonymy.

Calculation of Singular Value Decompositions in MATLAB

Algorithm 8 is the code used to calculate the singular value decomposition of a matrix A in MATLAB. The matrix A should contain subject classifications in the rows and search engines in the columns. SVD is performed on the matrix (the U matrix is ignored in the following calculations). The matrix Sigma and matrix V are then transformed and the columns are normalised so they all have length 1, leaving a matrix containing a diagonal of the number 1 and a triangular of mirrored values which describe how the corresponding eigenvectors relate to each other.

B.1 Summary

This chapter gave a detailed description of Singular Value Decomposition. This research uses Singular Value Decomposition to analyse and compare the latent patterns between the search

Algorithm 8 Apply SVD to matrix A

where A is a n by m matrix where n is the search engine and m is the term frequency

Outputs an n x n matrix which shows the relations of the search engines

```
1: [U,Sigma,V] = svd (A);
2: // Multiply singular values by the transpose of matrix V'
3: B = Sigma * V';
4: // Normalise columns using Frobenius norm;
5: normalised_columns=normc(B);
6: // Multiply normalised search engine matrix transposed by search engine matrix again to
   give matrix with all values between -1 and 1;
7: final_matrix = normalised_columns' × normalised_columns;
```

engines and collections. This is significant to web intelligence because the pattern analysis can be used on any level of on the entire taxonomy, increasing precision as lower levels are used. The pattern analysis can also be used to analyse relationships between search engines on selected nodes or node groups of any level of the taxonomy, allowing a detailed view of the relationships for only a limited set of subjects. This targeted information is very valuable for finding hidden relationships which human experts cannot easily see.

Bibliography

[ADC+00] T. Adams, J. Dullea, P. Clark, S. Sripada, and T. Barrett. Semantic integration of heterogeneous information sources using a knowledge-based system, 2000. 35

[BCC+04] B. Billerbeck, A. Cannane, A. Chatteraj, N. Lester, W. Webber, H. E. Williams, J. Yiannis, and J. Zobel. RMIT University at TREC 2004. In E. M. Voorhees and L. P. Buckland, editors, *Proceedings Text Retrieval Conference (TREC)*, Gaithersburg, MD, November 2004. National Institute of Standards and Technology Special Publication 500-261. 108

[BDJ99] Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup. Matrices, vector spaces, and information retrieval. volume 41, pages 335–362. Society for Industrial and Applied Mathematics, 1999. 22

[Ber01] Michael K. Bergman. The deep web: Surfacing hidden value. *The Journal of Electronic Publishing*, 7(1), 2001. 11

[BJC+03] Steven M. Beitzel, Eric C. Jensen, Abdur Chowdhury, David Grossman, and Ophir Frieder. Using manually-built web directories for automatic evaluation of known-item retrieval. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 373–374, New York, NY, USA, 2003. ACM Press. 50

[BJC+04] Steven M. Beitzel, Eric C. Jensen, Abdur Chowdhury, David Grossman, and Ophir Frieder. Hourly analysis of a very large topically categorized web query log. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference*

on Research and development in information retrieval, pages 321–328, New York, NY, USA, 2004. ACM Press. 2

[BJC+07] Steven M. Beitzel, Eric C. Jensen, Abdur Chowdhury, Ophir Frieder, and David Grossman. Temporal analysis of a very large topically categorized web query log. *J. Am. Soc. Inf. Sci. Technol.*, 58(2):166–178, 2007. 2

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web: Scientific american. *Scientific American*, May 2001. 35

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998. 10

[Bui98] P. Buitelaar. *CoreLex: Systematic Polysemy and Underspecification*. PhD thesis, Computer Science Department, Brandeis University, 1998. 48, 74

[BW93] Bruce G. Buchanan and David C. Wilkins, editors. *Readings in knowledge acquisition and learning: automating the construction and improvement of expert systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. 37

[BYRN99] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. Modern information retrieval. ACM Press / Addison-Wesley Boston, MA, 1999. 25

[Cal00] J. Callan. *Distributed information retrieval*, chapter 5, pages 127–150. Kluwer Academic Publishers, 2000. 57

[CBH00] Nick Craswell, Peter Bailey, and David Hawking. Server selection on the world wide web. In *Proceedings of the fifth ACM conference on Digital libraries, San Antonio, Texas, United States*, pages 37–46. ACM Press, 2000. 26, 54

[CC01] Jamie Callan and Margaret Connell. Query-based sampling of text databases. *ACM Trans. Inf. Syst.*, 19(2):97–130, 2001. 17, 51

[CCD99] Jamie Callan, Margaret Connell, and Aiqun Du. Automatic discovery of language models for text databases. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 479–490. ACM Press, 1999. 17, 19, 50, 52, 82

[CCH92] James P. Callan, W. Bruce Croft, and Stephen M. Harding. The INQUERY retrieval system. In *Proceedings of DEXA-92, 3rd International Conference on Database and Expert Systems Applications*, pages 78–83, 1992. 57

[CGL01] D. Calvanese, G. De Giacomo, and M. Lenzerini. A framework for ontology integration, 2001. 36

[CHLZ03] K. Chang, B. He, C. Li, and Z. Zhang. Structured databases on the web: Observations and implications, 2003. 11

[CLC95] J. P. Callan, Z. Lu, and W. Bruce Croft. Searching Distributed Collections with Inference Networks . In E. A. Fox, P. Ingwersen, and R. Fidel, editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–28, Seattle, Washington, 1995. ACM Press. 3, 26, 52, 80

[CM00] Z. Chen and X. Meng. Yarrow: A real-time client-side meta-search learner. In *Proceedings of the 2000 AAAI Workshop on Artificial Intelligence for Web Search (AAAI'00), Austin, Texas*, pages 12–17, 2000. 54

[CMC01] Ned Stoffel Chung-Min Chen. Telcordia lsi engine: Implementation and scalability issues. page 51, Heidelberg, Germany, April 01 - 02 2001. 11th International Workshop on research Issues in Data Engineering, Heidelberg, Germany. 23

[CPFC00a] J. Callan, A. Powell, J. French, and M. Connell. The effects of query-based sampling on automatic database selection algorithms. In *Technical Report CMU-LTI-00-162*, Carnegie Mellon University, 2000. Langauage Technologies Institute, School of Computer Science. 26

[CPFC00b] J. Callan, A. L. Powell, J. C. French, and M. Connell. The effects of query-based sampling on automatic database selection algorithms. Technical Report Technical Report CMU-LTI-00-162, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, 2000. 29

[Cra00] N. Craswell. Methods for distributed information retrieval, 2000. 24

[Cra01] Nicholas Eric Craswell. *Methods for Distributed Information Retrieval*. PhD thesis, The Australian National University, 2001. 19, 52, 82

[CS02] Abdur Chowdhury and Ian Soboroff. Automatic evaluation of world wide web search services. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 421–422, New York, NY, USA, 2002. ACM Press. 50

[CY99] Yong S. Choi and Suk I. Yoo. Neural net agent for discovering text databases on the web. In Johann Eder, Ivan Rozman, and Tatjana Welzer, editors, *Advances in Databases and Information Systems, Third East European Conference, ADBIS'99, Maribor, Slovenia, September 13-16, 1999, Proceedings of Short Papers*, pages 221–231. Institute of Informatics, Faculty of Electrical Engineering and Computer Science, Smetanova 17, IS-2000 Maribor, Slovenia, 1999. 26, 52

[DDL+90] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990. 22, 23

[DFvH+00] Stefan Decker, Dieter Fensel, Frank van Harmelen, Ian Horrocks, Sergey Melnik, Michel C. A. Klein, and Jeen Broekstra. Knowledge representation on the web. In *Description Logics*, pages 89–97, 2000. 34

[Din96] Gary Ding, Wei; Marchionini. A comparative study of web search service performance. In *Proceedings of the ASIS Annual Meeting*, pages 136–42, 1996. 2

[Din99a] Chris H. Q. Ding. A similarity-based probability model for latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, Berkeley, California, United States*, pages 58–65. ACM Press, 1999. 23

[Din99b] Chris H. Q. Ding. A similarity-based probability model for latent semantic indexing. In *Research and Development in Information Retrieval*, pages 58–65, 1999. 23

[Dog07] Dogpile. Different engines, different results. available from: "http://compareseachengines.dogpile.com/overlapanalysis.pdf, 2007. 2, 16

[DTZ00] Daryl J. D'Souza, James A. Thom, and Justin Zobel. A comparison of techniques for selecting text collections. In *Proceedings of the 11th Australasian Database Conference(ADC'2000)*, pages 28–32, Canberra, Australia, 2000. 26

[EFFS00] F. Esposito, S. Ferelli, N. Fanizzi, and G. Semeraro. Learning from parsed sentences with INTHELEX. In Claire Cardie, Walter Daelemans, Claire Nédellec, and Erik Tjong Kim Sang, editors, *Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop, Lisbon, 2000*, pages 194–198. Association for Computational Linguistics, Somerset, New Jersey, 2000. 48

[FN98] D. Faure and C. N'edellec. A corpus-based conceptual clustering method for verb frames and ontology acquisition. In *LREC workshop on Adapting lexical and corpus resources to sublanguages and applications, Granada, Spain*, pages 1–8, 1998. 48

[FPC+99] James C. French, Allison L. Powell, James P. Callan, Charles L. Viles, Travis Emmitt, Kevin J. Prey, and Yun Mou. Comparing the performance of database selection algorithms. In *Research and Development in Information Retrieval*, pages 238–245, 1999. 12, 26, 45

[Fuh99] Norbert Fuhr. A decision-theoretic approach to database selection in networked IR. *ACM Transactions on Information Systems*, 17(3):229–229, 1999. 17, 51

[Gan03] Fabien Gandon. Agents handling annotation distribution in a corporate semantic web. *Web Intelligence and Agent Systems, IOS Press*, 1(1):23–46, 2003. 49

[GCP03] Susan Gauch, Jason Chaffee, and Alexander Pretschner. Ontology-based personalized search and browsing. *Web Intelligence and Agent Systems, IOS Press*, 1(3):219–234, 2003. 49

[GGM95] Luis Gravano and Héctor García-Molina. Generalizing GlOSS to vector-space databases and broker hierarchies. In *International Conference on Very Large Databases, VLDB*, pages 78–89, 1995. 26, 29

[GGMT94] Luis Gravano, Héctor García-Molina, and Anthony Tomasic. The effectiveness of gioss for the text database discovery problem. In *Proceedings of the 1994 ACM SIGMOD international conference on Management of data*, pages 126–137. ACM Press, 1994. 55

[GGMT99] Luis Gravano, Héctor García-Molina, and Anthony Tomasic. GlOSS: text-source discovery over the Internet. *ACM Transactions on Database Systems*, 24(2):229–264, 1999. 3, 26, 53, 55

[GIS02] Luis Gravano, Panagiotis G. Ipeirotis, and Mehran Sahami. Query- vs. crawling-based classification of searchable web databases. IEEE Data Engineering Bulletin, March 2002. 46

[GIS03] Luis Gravano, Panagiotis G. Ipeirotis, and Mehran Sahami. Qprober: A system for automatic classification of hidden-web databases. *ACM Trans. Inf. Syst.*, 21(1):1–41, 2003. 51

[GP99] Michael Gordon and Praveen Pathak. Finding information on the world wide web: the retrieval effectiveness of search engines. *Inf. Process. Manage.*, 35(2):141–180, 1999. 50

[GS01] Jean Godby and Jay Stuler. The library of congress classification as a knowledge base for automatic subject categorization. In *Presented at the IFLA Preconference*, Dublin, Ohio, 2001. 63

[GVCC98] Julio Gonzalo, Felisa Verdejo, Irina Chugur, and Juan Cigarran. Indexing with wordnet synsets can improve text retrieval. In *ACL/COLING Workshop on Usage of WordNetfor Natural Language Processing.*, 1998. 35

[HC02] Jiawei Han and Kevin Chang. Data mining for web intelligence. *Computer*, 35(11):64–70, 2002. 11, 27

[HCBG01] David Hawking, Nick Craswell, Peter Bailey, and Kathleen Griffihs. Measuring search engine quality. *Information Retrieval*, 4(1):33–59, 2001. 50

[HCG01] David Hawking, Nick Craswell, and Kathleen Griffiths. Which search engine is best at finding online services? In *WWW Posters*, 2001. 50

[HCTH99] David Hawking, Nick Craswell, Paul Thistlewaite, and Donna Harman. Results and challenges in Web search evaluation. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1321–1330, 1999. 50

[HMS02] Monika R. Henzinger, Rajeev Motwani, and Craig Silverstein. Challenges in web search engines. *SIGIR Forum*, 36(2):11–22, 2002. 15

[HT99] David Hawking and Paul Thistlewaite. Methods for information server selection. *ACM Trans. Inf. Syst.*, 17(1):40–76, 1999. 18, 26

[HT05] David Hawking and Paul Thomas. Server selection methods in hybrid portal search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 75–82, New York, NY, USA, 2005. ACM Press. 56, 57

[IG02] P. Ipeirotis and L. Gravano. Distributed search over the hidden web: Hierarchical database sampling and selection. Technical report, Columbia University, Computer Science Department, 2002. 17, 46, 51

[IG04] P. Ipeirotis and L. Gravano. When one sample is not enough: Improving text database selection using shrinkage, 2004. 52

[IGS01a] P. Ipeirotis, L. Gravano, and M. Sahami. Qprober: A system for automatic classification of hidden-web resources, 2001. 46

[IGS01b] P. G. Ipeirotis, Luis Gravano, and Mehran Sahami. Persival demo: categorizing hidden-web resources. In *JCDL '01: Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries*, page 454, New York, NY, USA, 2001. ACM Press. 51

[IGS01c] Panagiotis G. Ipeirotis, Luis Gravano, and Mehran Sahami. Automatic classification of text databases through query probing. In *Selected papers from the Third International Workshop WebDB 2000 on The World Wide Web and Databases*, pages 245–255, London, UK, 2001. Springer-Verlag. 17, 51

[IGS01d] Panagiotis G. Ipeirotis, Luis Gravano, and Mehran Sahami. Persival demo: categorizing hidden-web resources. In *JCDL '01: Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries*, page 454, New York, NY, USA, 2001. ACM. 46

[ITL] Seppo Pohjolainen Ivar Tammeraid, Juri Majak and Tero Luodeslampi. Linear algebra. http://www.cs.ut.ee/ toomas_l/linalg/. 175

[JBS07] Bernard J. Jansen, Danielle L. Booth, and Amanda Spink. Determining the user intent of web search engine queries. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 1149–1150, New York, NY, USA, 2007. ACM Press. 2

[JGR00] Eric Miller Jean Godby and Ray Reighart. Automatically generated topic maps of world wide web resources. In *The Ninth International World Wide Web Conference*, May 2000. 63

[JSBS98] Bernard J. Jansen, Amanda Spink, Judy Bateman, and Tefko Saracevic. Real life information retrieval: a study of user queries on the web. *SIGIR Forum*, 32(1):5–17, 1998. 44

[JSS00] Bernard J. Jansen, Amanda Spink, and Tefko Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing and Management*, 36(2):207–227, 2000. 2, 44

[Kin03] John D. King. Deep web collection selection. Master's thesis, School of Software Engineering, Queensland University of Technology, 2003. 27, 57, 92, 141

[Kir98] Steven Kirsch. Infoseek's experiences searching the internet. *SIGIR Forum*, 32(2):3–7, 1998. 44

[KL03] John King and Yuefeng Li. Web based collection selection using singular value decomposition. In *IEEE/WIC International Conference on Web Intelligence (WI'03)*, pages 104–110. IEEE, 2003. 27, 57

[KLBN06] J D King, Yuefeng Li, P D Bruza, and Richi Nayak. Preliminary investigations into ontology-based collection selection. In *Proceedings of the Eleventh Australasian Document Computing Symposium*, pages 33–40, Brisbane, Australia, December 2006. 58, 134

[Kle99] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999. 10, 15

[KLTN07] John D King, Yuefeng Li, Xiaohui Tao, and Richi Nayak. Mining world knowledge for analysis of search engine content. *Web Intelligence and Agent Systems: An International Journal*, October 2007. 58, 64

[KMV00] Joerg-Uwe Kietz, Alexander Maedche, and Raphael Volz. A method for semi-automatic ontology acquisition from a corporate intranet. In *Proceedings of EKAW-2000 Workshop "Ontologies and Text", Juan-Les-Pins, France, October 2000*, number 1937 in Springer Lecture Notes in Artificial Intelligence (LNAI), 2000. 49

[LA05] H.Suzuki L.Vanderwende, G.Kacmarcik and A.Menezes. Mindnet: An automatically-created lexical resource. In *Proceedings of HLT/EMNLP 2005 Interactive Demonstrations, Vancouver, British Columbia, Canada*, 2005. 49

[LCC96] Z. Lu, J.P. Callan, and W.B. Croft. Applying inference networks to multiple collection searching. Technical Report TR96–42, University of Massachusetts at Amherst. Department of Computer Science, 1996. 26, 52

[LD04] Hugo Liu and Glorianna Davenport. Conceptnet: a practical commonsense reasoning toolkit. *BT Technology Journal*, 22(4):211–226, 2004. 34

[Len95] Douglas B. Lenat. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995. 34

[LG98] Steve Lawrence and C. Lee Giles. Searching the World Wide Web. *Science*, 280(5360):98–100, 1998. 2, 50

[Li01] Yuefeng Li. Information fusion for intelligent agent based information gathering. In Ning Zhong, Yi Yu Yao, Jiming Liu, and Setsuo Ohsuga, editors, *Web Intelligence: Research and Development, First Asia-Pacific Conference, WI 2001, Maebashi City, Japan, October 23-26, 2001, Proceedings*, volume 2198 of *Lecture Notes in Computer Science*, pages 433–437. Springer, 2001. 29

[LM00] Z. Lu and K. S. McKinley. *Advances in Information Retrieval*, chapter The Effect of Collection Organization and Query Locality on Information Retrieval System Performance and Design. Kluwer, New York, New York, 2000. 26

[LMYR00] King-Lup Liu, Weiyi Meng, Clement T. Yu, and Naphtali Rishe. Discovery of similarity computations of search engines. In *CIKM*, pages 290–297, 2000. 2, 4

[LS99] H. Vernon Leighton and Jaideep Srivastava. First 20 precision among world wide web search services (search engines). *J. Am. Soc. Inf. Sci.*, 50(10):870–881, 1999. 50

[LYM+98] K. Liu, C. Yu, W. Meng, W. Wu, and N. Rishe. A statistical method for estimating the usefulness of text databases, 1998. 2, 4

[LYM02] King-Lup Liu, Clement T. Yu, and Weiyi Meng. Discovering the representative of a search engine. In *CIKM*, pages 652–654, 2002. 56

[LZ04] Y. Li and N. Zhong. Capturing evolving patterns for ontology-based web mining. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 256–263, Beijing, China, 2004. 49

[LZ06] Y. Li and N. Zhong. Mining ontology for automatically acquiring web user information needs. *IEEE Transactions on Knowledge and Data Engineering*, 18(4):554–568, 2006. 49

[Mac67] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967. 64

[Mar82] M. E. Maron. Probabilistic approaches to the document retrieval problem. In *Proceedings of the 5th annual ACM conference on Research and development in information retrieval*, pages 98–107. Springer-Verlag New York, Inc., West Berlin, Germany, 1982. 35

[MB00] Filippo Menczer and Richard K. Belew. Adaptive retrieval agents: Internalizing local context and scaling up to the web. *Machine Learning*, 39(2/3):203–242, 2000. 16

[McG98] D. L. McGuinness. Ontological issues for knowledge-enhanced search. *Proceedings of the First International Conference on Formal Ontology in Information Systems*, pages 302–316, 1998. 34

[MLY+99] Weiyi Meng, King-Lup Liu, Clement T. Yu, Wensheng Wu, and Naphtali Rishe. Estimating the usefulness of search engines. *Proceedings of the 15th International Conference on Data Engineering, 23-26 March 1999, Sydney, Austrialia*, pages 146–153, 1999. 26

[MS00] Alexander Maedche and Steffen Staab. Discovering conceptual relations from text. In W. Horn, editor, *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, pages 321–325, 2000. 48

[MS01a] Alexander Maedche and Steffen Staab. Learning ontologies for the semantic web. In *SemWeb*, 2001. 48

[MS01b] Alexander Maedche and Steffen Staab. Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2):72–79, 2001. 37, 48

[MS06] E. Motta and M. Sabou. Language technologies and the evolution of the semantic web. In *The fifth International Conference on Language Resources and Evaluation*, Genoa, Italy, 2006. 34, 36

[Mue8a] Erik T. Mueller. Natural language processing with thoughttreasure. new york: Signiform, 1998a. 34

[MYL02] Weiyi Meng, Clement T. Yu, and King-Lup Liu. Building efficient and effective metasearch engines. *ACM Computing Surveys*, 34(1):48–89, 2002. 56

[NF03a] H. Nottelmann and N. Fuhr. Decision-theoretic resource selection for different data types in MIND. 2003. 57

[NF03b] Henrik Nottelmann and Norbert Fuhr. Evaluating different methods of estimating retrieval quality for resource selection. In *SIGIR '03: Proceedings of the 26th*

annual international ACM SIGIR conference on Research and development in informaion retrieval, pages 290–297, New York, NY, USA, 2003. ACM. 57

[NF04] Henrik Nottelmann and Norbert Fuhr. Combining cori and the decision-theoretic approach for advanced resource selection. In *In Proc. ECIC 2004*. Springer, 2004. 57

[NK98] Kwong Bor Ng and Paul P. Kantor. An investigation of the preconditions for effective data fusion in information retrieval: A pilot study, 1998. 21

[PBMW98] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998. 15

[Pre05] Oxford University Press. Dictionary facts. http://www.oed.com/about/facts.html, 2005. 80

[SB87] Gerard Salton and Chris Buckley. Term weighting approaches in automatic text retrieval. Technical report, Ithaca, NY, USA, 1987. 71

[SC00] H. Suryanto and P. Compton. Learning classification taxonomies from a classification knowledge based system. In S. Staab, A. Maedche, C. Nedellec, and P. Wiemer-Hastings, editors, *Proceedings of the Workshop on Ontology Learning, 14th Conference on Artificial Intelligence (ECAI'00)*, Berlin, 2000. Conference on Artificial Intelligence (ECAI'00). 49

[SC03] Luo Si and Jamie Callan. Relevant document distribution estimation method for resource selection. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 298–305, New York, NY, USA, 2003. ACM Press. xi, 3, 32, 43, 45, 56, 105, 106

[SC04] L. Si and J. Callan. Unified utility maximization framework for resource selection, 2004. 57

[SC05] Luo Si and Jamie Callan. Modeling search engine effectiveness for federated search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 83–90, New York, NY, USA, 2005. ACM Press. 56, 57

[SH03] LaQuiesia S. Smith and Ali R. Hurson. A search engine selection methodology. In *ITCC '03: Proceedings of the International Conference on Information Technology: Computers and Communications*, page 122, Washington, DC, USA, 2003. IEEE Computer Society. 2, 4

[SHMM98] Craig Silverstein, Monika Henzinger, Hannes Marais, and Michael Moricz. Analysis of a very large altavista query log. Technical Report 1998-014, Digital SRC, 1998. http://gatekeeper.dec.com/pub/DEC/SRC/technical-notes/abstracts/src-tn-1998-014.html. 44

[Sim03] Kwang Mong Sim. Web agents with a three-stage information filtering approach. In *2003 International Conference on Cyberworlds, 2003. Proceedings.*, pages 266–273, 2003. 35

[SMHM99] Craig Silverstein, Hannes Marais, Monika Henzinger, and Michael Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12, 1999. 44

[SS04] S Staab and R Studer. *Handbook on ontologies*. Berlin; New York: Springer, 2004. 49

[Sto05] Nenad Stojanovic. Information-need driven query refinement. *Web Intelligence and Agent Systems, IOS Press*, 3(3):155–170, 2005. 49

[SWJS01] A. Spink, D. Wolfram, B. Jansen, and T. Saracevic. Searching the web: The public and their queries. *Journal of the American Society for Information Science*, 53(2):226–234, 2001. 44

[SZ07] Milad Shokouhi and Justin Zobel. Federated text retrieval from uncooperative overlapped collections. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 495–502, New York, NY, USA, 2007. ACM Press. 113

[SZST06] Milad Shokouhi, Justin Zobel, Falk Scholer, and S. M. M. Tahaghoghi. Capturing collection size for distributed non-cooperative retrieval. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 316–323, New York, NY, USA, 2006. ACM Press. 32, 43

[SZTS07] Milad Shokouhi, Justin Zobel, Saied Tahaghoghi, and Falk Scholer. Using query logs to establish vocabularies in distributed information retrieval. *Inf. Process. Manage.*, 43(1):169–180, 2007. 58

[TH07] Paul Thomas and David Hawking. Evaluating sampling methods for uncooperative collections. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 503–510, New York, NY, USA, 2007. ACM Press. xi, 112

[TL01] Theodora Tsikrika and Mounia Lalmas. Merging techniques for performing data fusion on the web. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 127–134. ACM Press, 2001. 25

[UG96] Mike Uschold and Michael Grüninger. Ontologies: principles, methods, and applications. *Knowledge Engineering Review*, 11(2):93–155, 1996. 34

[VGJL95] E. Voorhees, N. Gupta, and B. Johnson-Laird. The collection fusion problem. In *Text REtrieval Conference*, pages 95–104, 1995. 26

[vHSW97] G. van Heijst, A. Th. Schreiber, and B. J. Wielinga. Using explicit ontologies in kbs development. *Int. J. Hum.-Comput. Stud.*, 46(2-3):183–292, 1997. 34

[Vie01] Manuela Viezzer. *Dynamic Ontologies or How to Build Agents That Can Change Their Mind*. PhD thesis, The School of Computer Science, The University of Birmingham, 2001. 47

[vZvO04] Roelof van Zwol and Herre van Oostendorp. Google's "i'm feeling lucky", truly a gamble? In *WISE*, pages 378–389, 2004. 44

[Wat91] David S. Watkins. *Fundamentals of Matrix Computations*. John Wiley and Sons, Inc, 1991. 172

[Wel98a] C. Welty. Dls for dls : Description logics for digital libraries. In *Proc. of the 1998 Int. Workshop for Description Logics*, pages 8–10, Trento, Italia, 1998. 48

[Wel98b] C. Welty. The ontological nature of subject taxonomies. In *Proceedings of the 1998 International Conference on Formal Ontology in Information Systems (FOIS'98).*, pages 317–327, 1998. 48

[Whi92] John S. White. Lexical and world knowledge: Theoretical and applied viewpoints. In *Proceedings of the First SIGLEX Workshop on Lexical Semantics and Knowledge Representation*, pages 171–183, London, UK, 1992. Springer-Verlag. 49

[WKCC03] Chis Welty, Ruchi Kalra, and Jennifer Chu-Carroll. Evaluating ontological analysis. In A. Doan, A. Halevy, and N. Noy, editors, *Proceedings of the ISWC-03 Workshop on Semantic Integration.*, 2003. 48

[XC98] J. Xu and J. Callan. Effective retrieval with distributed collections. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 112–120, 1998. 80