KeyConcept: Exploiting Hierarchical Relationships for Conceptually Indexed Data

by

Devanand Rajoo Ravindran

B.E. (Computer Science and Engineering)
Bharatidasan University, Trichy, India, 2000

Submitted to the Department of Electrical Engineering and Computer Science and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

_____
Dr. Susan Gauch, Chairperson

_____
Dr. Costas Tsatsoulis, Committee Member

_____
Dr. Jerry James, Committee Member

Date Accepted _____

Acknowledgements

I’m greatly honored to have worked for Dr. Susan Gauch towards my thesis and for KeyConcept. Many thanks are due to her for having helped me hone my skills in programming and inculcate an interest in research. I would also like to thank the committee members of my thesis, Dr. Jerry James and Dr. Costas Tsatsoulis.

I also express my gratitude to Sriram Chadalavada, Juan Madrid and Joana Trajkova for having worked with me in my projects. Without their code, I would not be in a position to have completed my thesis. I’m grateful to all ITTC staff for their timely help in solving any technical problems that I’ve had.

I also thank my roommates and the Adams who make my stay in Lawrence more agreeable. I express my heartfelt gratitude to my parents and brother for having believed in me and helping me in all possible ways to pursue my dreams.

Abstract

As the number of available Web pages grows, users experience increasing difficulty finding documents relevant to their interests. One of the underlying reasons for this is that most search engines find matches based on keywords, regardless of their meanings. To provide the user with more useful information, we need a system that includes information about the conceptual frame of the queries as well as its keywords. This is the goal of KeyConcept, a search engine that retrieves documents based on a combination of keyword and conceptual matching. This paper describes the system architecture of an enhanced KeyConcept, the training of the classifier for the system, and the results of our experiments evaluating system performance. Experiments that check whether result pruning and retrieval based on the hierarchical structure of the ontology help in improving precision are also described. KeyConcept is shown to significantly improve search result precision through its use of conceptual retrieval. Pruning and Hierarchical Retrieval are also shown to yield better results in most cases.

Contents

6. Conclusions and Future Work

List of Figures

List of Charts

1. Introduction

1.1 Motivation

Content on the web has been increasing rapidly over the last few years. A popular search engine reports to index 3.3 billion pages as of September 2002 [SES 03]. As the number of pages available increases, finding relevant information becomes more difficult. Much of this difficulty arises from the ambiguity present in natural language. For instance, two people searching for “wildcats” may be looking for two completely different things (wild animals and sports teams), yet they will get exactly the same results. We can see that searches are made based on pure string matching; the specific meaning wanted by the user is ignored.

There exists a need for a system that can take into account not only the keyword entered for a search, but also the meaning or the *concept* for which the user is searching. Such a system would be able to filter out pages that are not directly related to the user’s desired concept and present only results that match the user’s interests. Current search engines fall short in providing this vital distinction to the user. It is our belief that KeyConcept, a search engine that incorporates search by keyword and concept, can effectively increase search result accuracy.

1.2 Current Problems

The major reason the Internet is hard to search is because there is a lot of data but not enough information. In other words, the data available on the Net is not organized in any particular sense. Current search engines usually search by simply matching the query terms as strings. This results in the user obtaining documents that might have contained the *term* he might have been looking for, but not the *meaning* he was looking for. This is due to the fact that a search term may have multiple meanings [Krovetz 92]. There are very few search engines that even categorize their results in different conceptual categories [NorthernLight], thus guiding users through the result set.

There is already some conceptual information already available on the Internet in the form of directory services such as Yahoo! [YAHOO] or the Open Directory Project [ODP]. Websites do arrange web pages into categories for browsing purposes. However, each site has its own hierarchy and only a small fraction of web pages appear in these browsing hierarchies. The hierarchical information may be used to restrict search to subsets within that site, but this information is not used by web-wide search engines. The semantic web efforts attempt to address this issue by encoding conceptual information inside web pages themselves. These ontological approaches have two main drawbacks. First, where will an agreed-upon conceptual hierarchy of sufficient detail come from [Tirri 03]. Second, only a small percentage of web contributors will ever go through the process of manually annotating their web pages with ontological information. In our approach, we work with regular web pages and try to identify and exploit the latent conceptual information.

1.3 Related Work

1.3.2 Text Classification

Text classification organizes information by associating a document with the best possible concept(s) from a predefined set of concepts. Several methods for text classification have been developed, each with different approaches for comparing the new documents to the reference set. These including comparison between vector representations of the documents (Support Vector Machines, k-Nearest Neighbor, Linear Least-Squares Fit, TF-IDF), use of the joint probabilities of words being in the same document (Naïve Bayesian), decision trees and neural networks. A thorough survey and comparison of such methods is presented in [Yang 03], [Sebastiani 02], [Pazzani 96], [Ruiz 99]. Several systems have been built that incorporate classification to allow users to explore document sets. [Yang 03] particularly examines the complexities involved in different methods of text categorization and especially in hierarchical categorization.

[Kato 99] defines an idea-deriving informational system that uses a combination of a concept-base and normal character-string matching for information retrieval. The concept base in this case is created either based on word co-occurrences (corpus-based) or dictionary based. Document terms are then indexed according to the concept base. A combination of both conceptual and keyword matching rather than either method individually obtains the best results in this system.

As presented by [Chekuri 97] and [Matsuda 99], one of the main uses of text classification is to restrict the search domain. In [Chekuri 97], users have the option of

specifying some concepts of interest when submitting a query. Then, the system only searches for results in the specified concepts. The approach in [Matsuda 99] extends this idea by classifying documents based on other attributes, e.g., size, number of images, presence or absence of certain tags, as well as content. In this system, the user has the option of specifying the type of document he/she is looking for, e.g., a catalog, a call for papers, a FAQ, a glossary, etc., in addition to the search terms. More recently, [Cui 02] uses the internal structure of documents to assign XML classes such as *Nomenclature*, *Description*, *Images*, *References* to domain-specific document collections and the uses these classes for retrieval.

[Glover 01] also discusses the usage of document-specific structure for identifying documents as belonging to a particular category, such as research papers or personal homepages and lets the user choose the type of page he wants. This allows the user to choose the type of page the user wants but not the topic of interest. [Klink 02] approaches conceptual retrieval differently by defining a concept based on the query term(s) used. Previous queries and a known set of relevant documents are used to learn the concept definition for a query term. The key terms in the definition are then used to expand the query that is then sent to a normal search engine.

In the OBIWAN project [Zhu 99], ontologies are used to represent user profiles. Queries are submitted to a traditional search engine, and the results are classified into the ontology concepts based on their summaries. The documents in the result set are re-ranked based on matches between the summary concepts and the highly weighted

concepts in the user profile. While this approach was able to improve rank ordering of the results, it was not able to find more information for users because it could only work on the result set retrieved by a generic search engine. KeyConcept takes this approach one step further by integrating the conceptual matching into the retrieval process itself.

Other methods for Text Classification have been examined in great detail. Some of these approaches include implementing unsupervised learning algorithms like Latent Semantic Analysis [Cai 03], and using AI rule-base trees to compute the conceptual relevancy of search results [Lu 99].

1.3.2 Ontologies

An ontology is an arrangement of concepts that represents a view of the world [Chaffee 2000] that can be used to structure information. Ontologies can be built by specifying the semantic relationships between the terms in a lexicon. One example of such ontology is Sensus [Knight 94], a taxonomy featuring over 70,000 nodes. The OntoSeek system [Guarino 99] uses this ontology for information retrieval from product catalogs.

Ontologies can also be derived from hierarchical collections of documents, such as Yahoo! [YAHOO] and the Open Directory Project [ODP]. [Labrou 99] reports the use of the TellTale [Pearce 97] classifier to map new documents into the Yahoo! ontology. This is done by training the classifier with documents for each one of the concepts in the ontology, and then finding the concept that has a greater similarity with the new document. Furthermore, an ontology can be used to allow users to navigate and search

the Web using their own hierarchy of concepts. The OBIWAN project [Zhu 99] accomplishes this by letting users define their own hierarchy of concepts, then mapping this personal ontology to a reference ontology (Lycos in this case).

Liu [Liu 02] and Pitkow [Pitkow 02] have devised systems that use personalization to identify relevant concepts as related to the individual user. Liu [Liu 02] approaches the problem of personalizing information retrieval according to user preferences by analyzing previous queries made by the user and developing a user profile. The user profile is then used to select the most appropriate categories for a user when a user makes a new query. A general profile for categories is also prepared using the ODP [ODP] structure. In this system, users have to select the relevant documents for a query and the relevant categories for each document too. This lays too much emphasis on the user's decision on whether the document/category is relevant or not. [Pitkow 02] uses *contextualization* to obtain relevant search results from an Internet search engine. The ODP structure is used here to identify the user's search context. The user's past browsing history is also stored. This information is used to augment a query by adding relevant terms from the user's profile and current context.

Authors can also embed ontological information in their documents. SHOE (Simple HTML Ontology Extensions) [Heflin 2000] is a language that serves this purpose, allowing the creation of new custom-made ontologies and the extension of existing ones.

Our work derives from OBIWAN [Zhu 99] by Xiaolan Zhu and Susan Gauch. OBIWAN is a comprehensive system designed for characterizing contents of a website. In this system, a website is classified as belonging to a particular category by taking its component pages and classifying them into concepts in an ontology. This work was extended to provide personalized search and browsing [Gauch 04]. We extend this work by incorporating classification in the indexing process, not merely as a post-process on the result set.

1.4 Proposed Solution

We intend to develop a solution that will, given a query, retrieve information not only based on the query terms, but also on the desired concepts the user is interested in. For example, if the user desires to search for “rock” he enters “rock” as his/her query term and also gives as input to the system, “Music” as the category in which he/she is interested in. This will help avoid the *multiple meaning* problem of words.

To perform the above-mentioned function, the index created needs to be not only based on words in the document, but also indexed based on what concept the document most corresponds to. For this purpose, we intend to simultaneously create an inverted index that indexes both by words and concepts. For *training* the classifier, we use the manually created ODP collection. Using these documents already classified in an ontology, we create a training set that can be used later to classify documents from a different collection. For the ontology, we use the Hierarchical structure obtained from ODP.

We also aim to improve retrieval results by not only searching the exact concepts the user wants to see, but also nearby concepts in the hierarchy. We believe that any document misclassified into a neighboring concept could thus still be selected by the retrieval system.

1.5 Overview

KeyConcept creates an inverted index in a method similar to any other major search engine during the indexing process. The main aspect by which it differs from normal search engines is that KeyConcept also includes information about the concepts to which each document is related. To accomplish this, the traditional structure of an inverted file was extended to include mappings between concepts and documents. The retrieval process takes advantage of this enhanced index, supporting queries that use only words, only concepts, or a combination of the two. The user can select the relative importance of the criteria (word match or concept match).

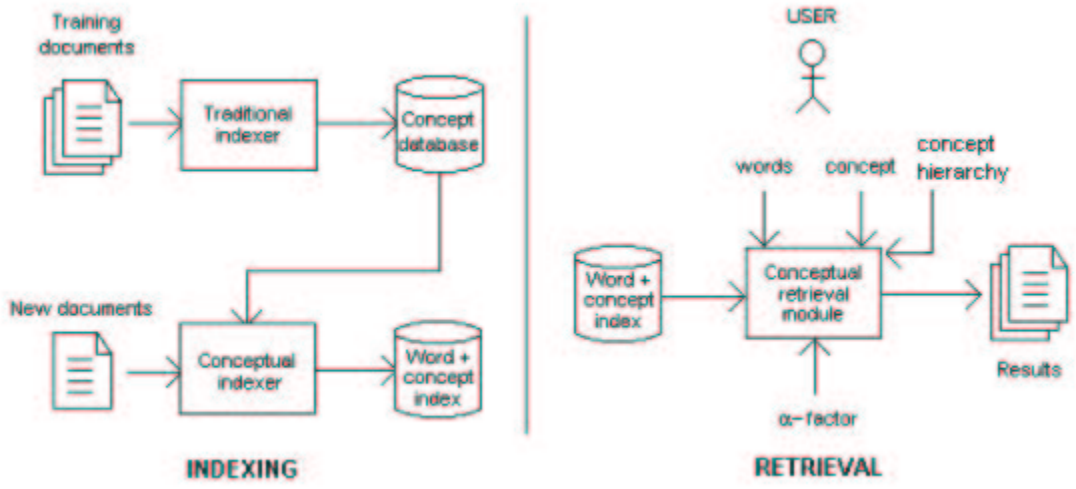

Figure 1 Operation of the conceptual search engine

Indexing

The indexing process is comprised of two phases: Classifier training and collection indexing. During classifier training, a fixed number of sample documents for each concept in the ODP collection are collected and merged, and the resulting super-documents are preprocessed and indexed using the *tf* * *idf* method. This essentially represents each concept by the centroid of the training set for that concept. During collection indexing, new documents are indexed using a vector-space method to create a traditional word-based index. Then, the document is classified by comparing the document vector to the centroid for each concept. The similarity values thus calculated are stored in the concept-based index.

Retrieval

During retrieval, the user provides a query containing words and concept identifiers. Concept matched and keyword matches between the query and documents are combined with a factor (α factor) between 0 and 1, specifying the relative importance of concept matches to word matches. If α is 1, only concept matches are considered. If it is 0, only word matches matter. When α is 0.5, concept and word matches contribute equally.

We used an optimum value of 0.3 for α, the details of the process of deciding optimum α are shown in [ITTC 03]. After receiving user data, the search engine performs the search and stores the results for word and concept matches in separate accumulators. The final document scores are computed using the formula:

$$Document\ score = (\alpha \times concept\ score) + ((1-\alpha) \times word\ score)$$

Hierarchical Retrieval

To obtain results not only in the concept identifiers mentioned by the user, but also those in the vicinity, we intend to reference the ontology used for training. We search for the user-specified concept and then locate the structurally nearby concepts. We investigate the effect of including children, parents, and siblings in the hierarchy. We have performed different experiments with several combinations of relationships and our results are presented in section 5.4.5

2. KeyConcept Architecture

2.1 Indexing

Indexing is the method of storing information about Web pages so they can be retrieved efficiently. KeyConcept uses an inverted index, a common way of organizing data extracted from Web pages. The process of indexing takes a Web page, processes it to extract the words, and calculates statistical information about the word, and stores this information on an index.

In KeyConcept, two indexes need to be created – one for the keywords belonging to a document and one for the conceptual categories to which a document belongs. The process of indexing is shown in Figure 2.

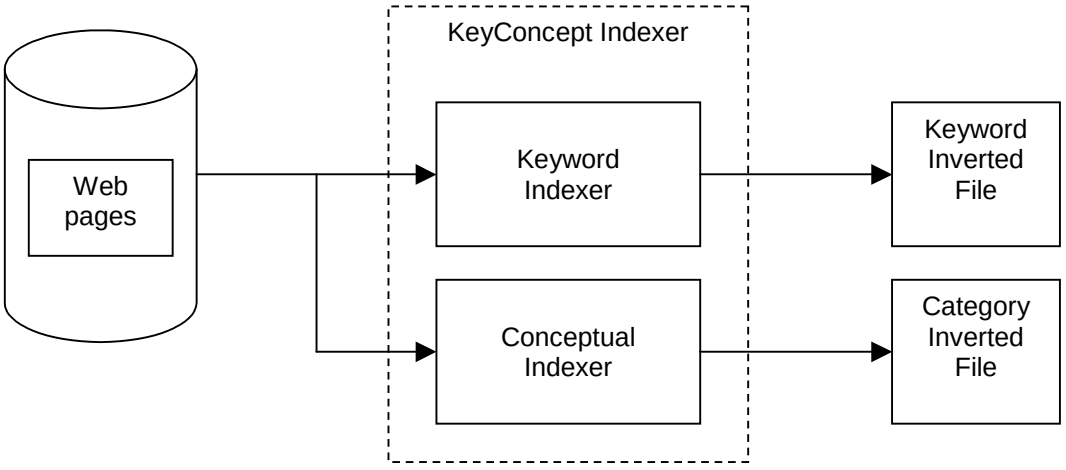

Figure 2. KeyConcept Components

2.1.1 Keyword Indexing

There are three major components of our Inverted Index – the dictionary file, the postings file and the documents file. The dictionary file (DICT) stores each word, its idf (inverse document frequency) and a pointer to the first occurrence of the word in the postings file (POST). The postings file (POST) keeps track of how many times the word has appeared in each document. The documents file (DOCS) maps between the document id and each word contained in the document in DICT and POST. The documents structure keeps track of the normalization value for each document so that search results can be normalized by length. The normalization method followed is one of the most popular normalization methods - the pivoted document normalization method [Singhal 96]. A brief explanation of the elements in each of the three structures of the inverted index – DICT, POST and DOCS are given below:

a) DICT:
    i) key – Term that is being indexed
    ii) docs – Number of docs in collection that contain word
    iii) idf – Inverse Document Frequency of the word over the whole collection (computed at the end of indexing phase)
    iv) post – Link to the first posting record that contains the word

b) POST

i) doc_id – ID of the document that contains the word

ii) count – Number of times the word has occurred in the individual document

iii) word_link – Link to the next occurrence of the word in the collection

iv) doc_link – Link to the DOCS record that contains the IDF of the term

c) DOCS

i) key – Unique Document ID of the document

ii) norm – Normalization value for the document(computed using pivoted normalization technique)

iii) wordpost – Link to the posting record of the first word in the document

iv) catpost – Link to the posting record of the first concept to which the document belongs (updated during classification phase)

v) num – The number of words in the document

In KeyConcept, we use a disk-based mapping system for storing the dictionary and posting files. The disk-based mapping system helps to avoid the common memory-shortage pitfalls experienced in memory-based indexing. The sequence of indexing a collection of files is shown in Figure 3. Each file is processed to obtain its individual tokens. Each token is then processed to its canonical form (i.e. stemming, down casing and removal of stopwords) and entries in DICT, POST and DOCS are updated.

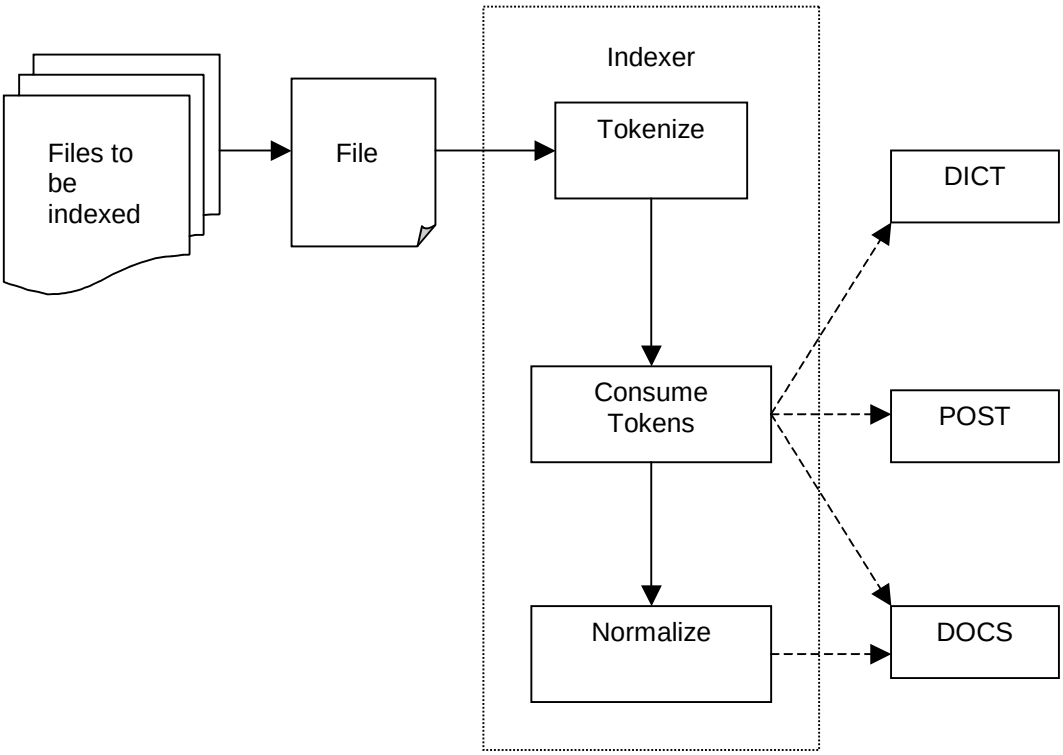

Figure 3. Indexing Process

2.1.2 Conceptual Indexing

KeyConcept provides a unique feature for conceptual searching. The method used to index documents based on concepts is very similar to the method used to index documents based on keywords. The first step is to identify the best concept(s) for document and then index that information. The process of assigning concepts to a document is called *classification.* Our method uses training to create a mapping between concepts and vocabulary based on an initial training set. Then, new documents have their vocabulary compared to each concept's vocabulary to identify the best match(es).

2.1.2.1 Training Phase

In our approach, we use a training corpus in which each concept has a set of documents already assigned to it. We use this to form a training inverted file structure that will later be used to classify the new dataset. The training process is simply an indexing process with training files in the same concept assigned the same ID. Essentially, a super-document is formed by merging all documents in the training set of a concept. This super-document is considered a representative document for that particular category. Figure 4 describes the procedure followed to obtain a training inverted file i.e. the TDICT, TPOST and TDOCS files.

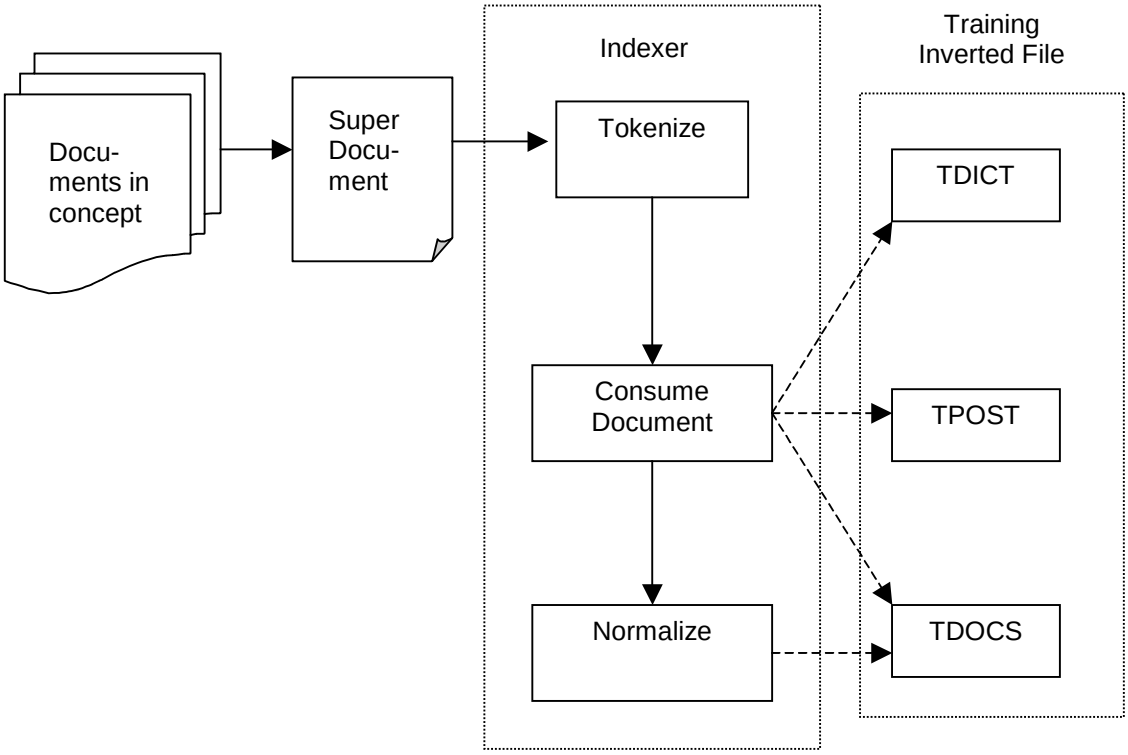

Figure 4. Training Phase

2.1.2.2 Classification Phase

In the classification phase of conceptual indexing, the training inverted file obtained is used to identify the most relevant classes for the set of documents to be indexed. The trained documents serve as an example against which new documents are compared. When a document is classified, the training index is looked up to obtain the top *k* categories for the top *N* words in the documents and the weights are accumulated for each concept. The most relevant concepts from the training index are obtained by sorting the concepts by weight. This process is accomplished via the retrieval function of the vector space search engine code we use. Essentially, the document is treated as a query against the concept super-document stored in the inverted index. The *catpost* field in DOCS file (created during keyword indexing) is updated to point to the first concept to which the document belongs. Figure 5 shows the process of conceptual indexing. The results of this classification are stored in CATDICT, CATPOST files. A brief description of the elements of a CATDICT record and CATPOST record is given below:

a) CATDICT
    i) key – Unique ID of the category
    ii) docs – Number of docs indexed in this category
    iii) catpost – Link to the first CATPOST record corresponding to the category

b) CATPOST

i) doc_id – ID of the category that points to this record

ii) weight – Weight of the category in this document

iii) category_link – Link to the next category for this document

iv) doc_link – Link to the next document in the same category

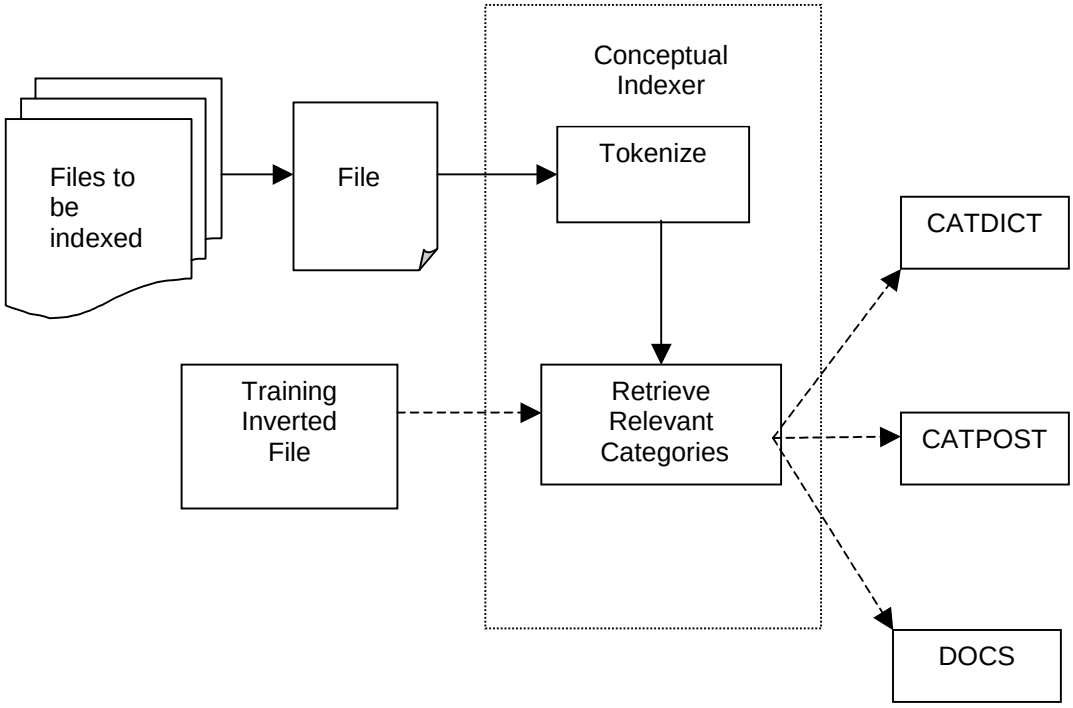

Figure 5. Classification Phase

2.2 Retrieval

There are two main components of retrieval in KeyConcept – keyword-based retrieval and concept-based retrieval. Results from these two different retrieval methods are then combined to give the final result. An α-factor is applied while computing the final scores for documents. Thus, the final scoring formula for documents is:

*document_ score = (α * normalized_concept_score) + ((1-α) * keyword_score)*

where

normalized_concept_score = concept_score for document / *maximum_concept_score*

and

keyword_score = word_score for word in document * *normalization_value* of document

Sections 2.2.1 and 2.2.2 describe each type of retrieval.

2.2.1 Keyword Retrieval

Once the documents have been indexed, search queries can be applied to them to retrieve documents. Once the user enters the query terms, the indexes (DICT, POST, DOCS) are searched to obtain the top N documents for each of them. The terms entered are searched for in DICT. We also get the number of documents (*numdocs*) with that term and the pointer to the first corresponding document is obtained. Next, the posting file POST is accessed at the obtained pointer location and the set of postings records for the word are accessed in order. For each postings record, the frequency of the word in the document (*tf*) is obtained along with a pointer to the DOCS file. The corresponding document information such as normalization factor (*norm*) is found by accessing the DOCS file.

The normalization factor has already been computed during the indexing phase as described in section 2.1.1

A temporary accumulator array to keep track of the score for each document is created and the array elements corresponding to each posting record are updated with the weight of the word in the document calculated from *numdocs*, *tf* and *norm*.

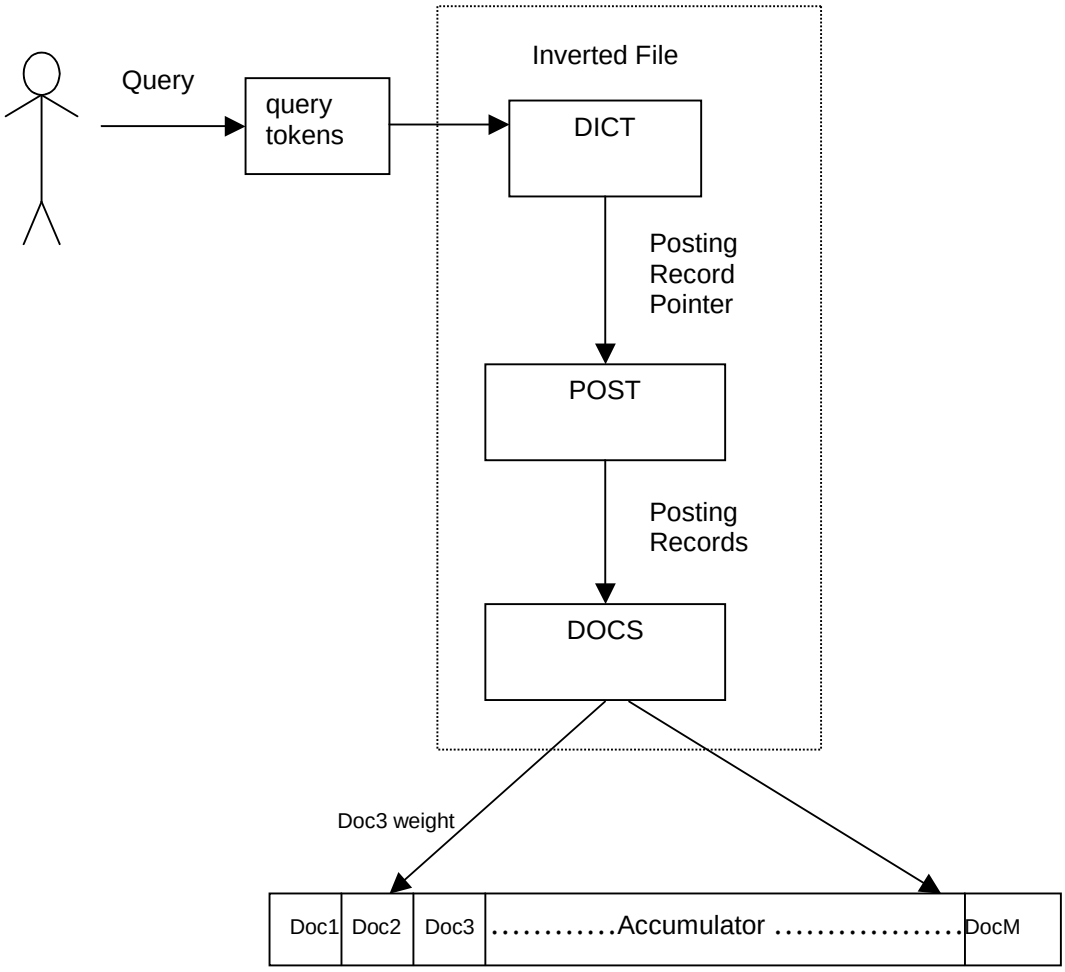

Figure 6. Keyword Retrieval

2.2.2 Conceptual Retrieval

Conceptual retrieval is similar to keyword retrieval, except that instead of accessing the inverted file created for keywords, we access the inverted file created for concepts, i.e. CATDICT and CATPOST. The user enters query terms and selects from a list of concepts the ones that are most related to the topic of interest. For example, if a user were searching for information about the Kansas City Royals, he would choose *Sports >> Teams* as one of his preferred categories. The KeyConcept system uses the unique ID of the category to search CATDICT and find the documents associated with the category present in the category postings file (CATPOST) exactly as it uses words to search DICT to find postings in POST. It updates a concept accumulator with the corresponding weight similar to building the keyword accumulator. Figure 7 shows the operation of the conceptual retrieval process.

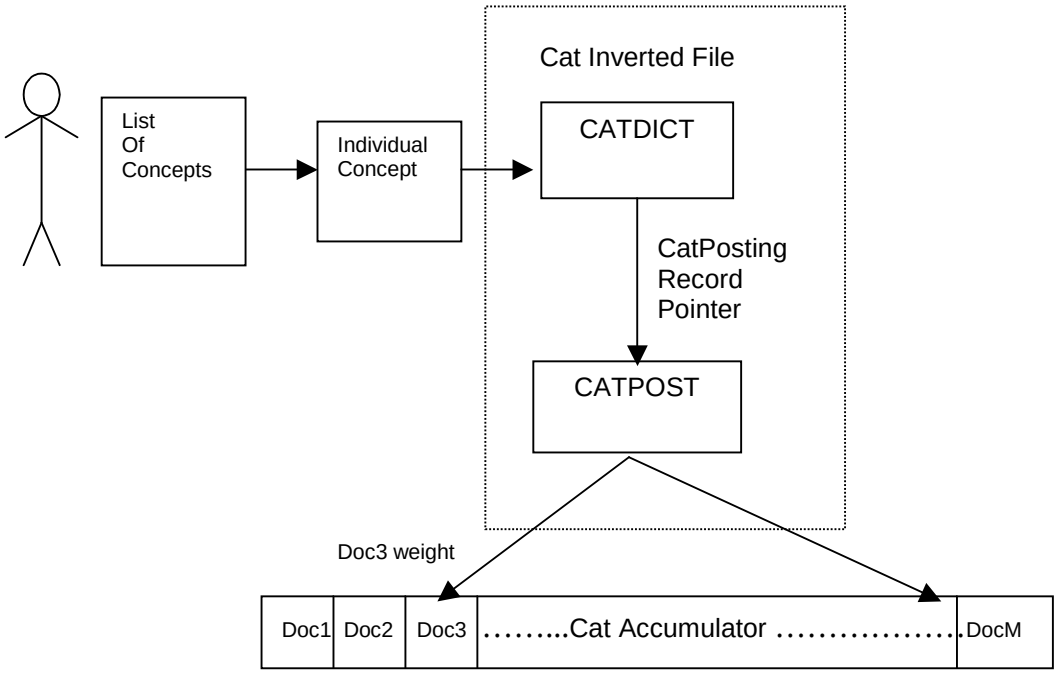

Figure 7. Conceptual Retrieval

2.3.3 Combined Retrieval

The two accumulators, Keyword accumulator and Concept accumulator are now combined using the formula mentioned above. That is,

combined_score[each document]

$$= (\alpha * concept\_score[document]) + (1-\alpha) * keyword\_score[document]$$

Both scores are normalized as described in the beginning of section 2.2. The final scores are sorted in descending order of the combined score. Figure 8 illustrates this operation.

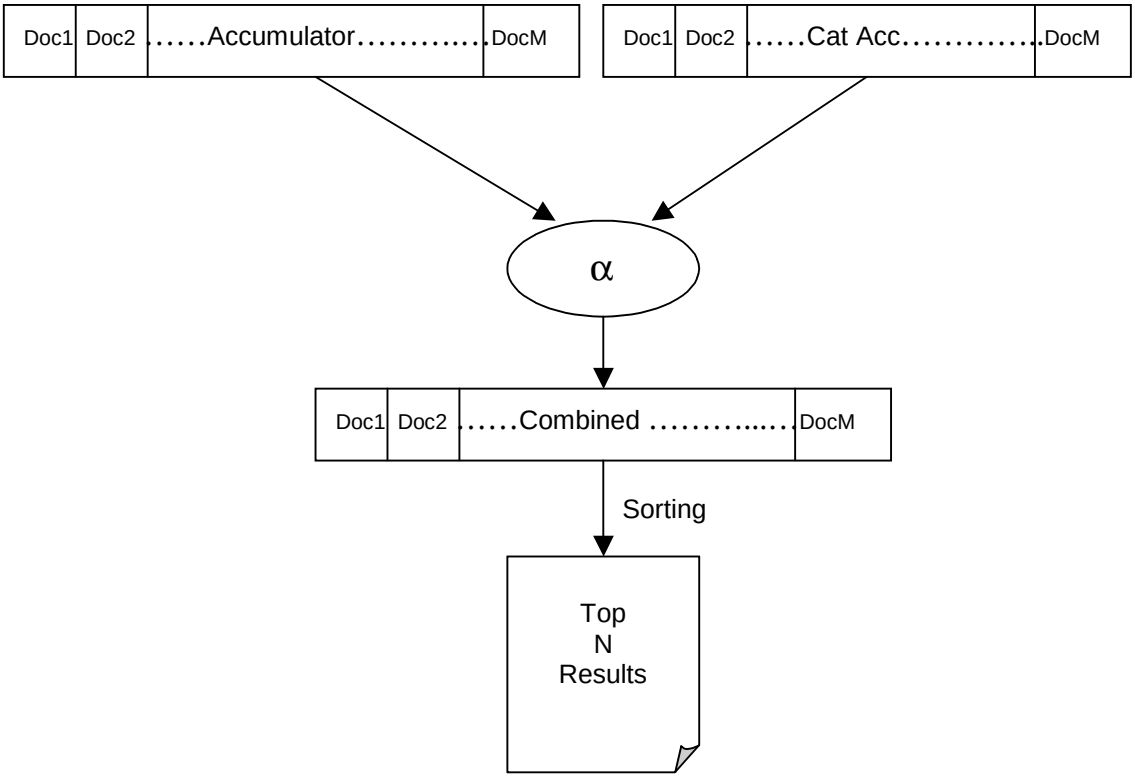

Figure 8. Combined Retrieval

3. Exploiting Hierarchical Relationships

This work is focused on enhancing KeyConcept by making use of the hierarchical nature of concepts in an ontology. This conceptual hierarchy can be exploited in two ways:

3.1 Pruning Results based on Hierarchical Relationships

Obtained results from a search can be "pruned" to remove documents that are completely unrelated to the concepts searched but have managed to find their way into the top *N* results. To do this, the top three concepts of each document in the result set are compared with the concepts the user searched for. We chose the top three concepts that a document belongs to as the cut-off for comparison based on the results obtained in [ITTC 03]. These experiments showed that the increase in precision was not significant with more than three concepts were used for retrieval. The document can generally thus be pruned if none of the top three categories match the user's choice of categories.

For example, the user might be interested in the category *"Arts/Styles/Blues"* for his search. The obtained top 3 categories for a result document might be

1) *Arts/Instruments/Guitar*
2) *Shopping/Music/CDs*
3) *Arts/Style/Classic_Rock*

We can decide to prune at Level1 or Level2, i.e., either at the *"Arts/"* level or the "*Arts/Styles/"* level. Depending on this choice, the document might be allowed to remain in the result set or be pruned.

3.2. Retrieval based on Hierarchical Relationships

Many documents that are been misclassified during training are classified into a concept that is close in distance to the correct concept. For example, if a document belonged to *Sports >> Teams >> Baseball* category, it might have been misclassified either in the *Sports >> Teams* category or in the *Sports >> Teams >> Baseball >> Apparel* category. A segment of the ODP tree hierarchy is shown in Figure 9.

0 Arts
- *Music*
  - Styles
  - Collecting
  - Instruments
  - .
  - .
  - .
  - Trading
- *Television*
  - Networks
  - Video_Production
  - Cable_TV
  - .
  - .
  - .
  - Interactive
- .
- .
- .
- *Directories*

1 Business

.
.
.

2 Sports

Figure 9. Example ODP hierarchy

Enhanced KeyConcept goes to the ODP Tree stored in a plaintext file and searches for concepts that are hierarchically close in distance to the given concept. It includes these concepts as part of the search parameters and performs a combined Keyword/Concept search as in basic KeyConcept. Different experiments have been performed to choose the right kind of neighbors for the given concepts and to estimate the weight each of these hierarchical neighbors should be given. The results of these experiments are detailed in Section 5.2. Figure 10 describes the hierarchical retrieval process.

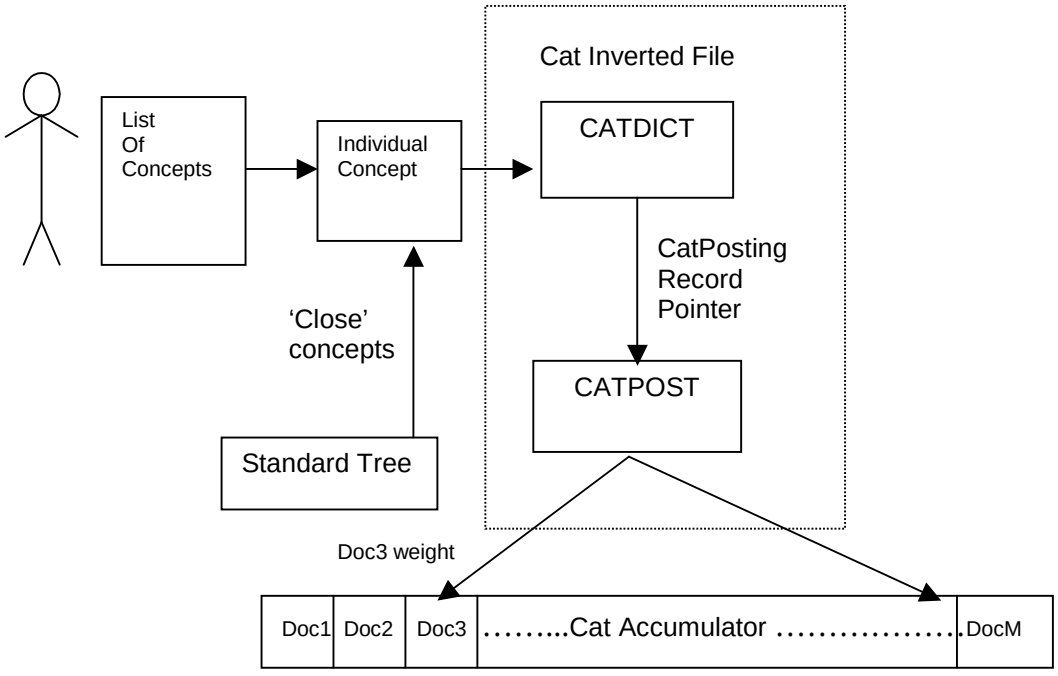

Figure 10. Hierarchical Retrieval

4. Experimental Procedure

This section describes the different experiments we performed to evaluate our enhancements to KeyConcept. Section 5 gives the results of these experiments.

4.1. Data Sets

4.1.1. Training Data

Because it is readily available for download from their Web site in a compact format, we chose to use the Open Directory Project hierarchy [ODP] as our source for training data,. In addition, it is also becoming a widely used, informal standard. As of April 2002, the Open Directory had more than 378,000 concepts. Because of the large volume of data involved, training the classifier would be a long and difficult task, if the full set of concepts were used. In addition, subtle differences between certain concepts may be apparent to a human but indistinguishable to a classification algorithm. Thus, we decided to use the concepts in the top three levels of the Open Directory. This initial training set was composed of 2,991 concepts and approximately 125,000 documents. The first three levels of ODP were used to obtain a list of categories and generate the training set. Documents below level 3 in ODP were propagated upwards to their corresponding level 3 categories.

4.1.2. TREC Data

We built a test set consisting of 100,000 documents chosen from the WT2g collection [Hawking 99]. The test set includes the 2,279 documents having positive relevance judgments and 97,721 randomly selected documents. Because the title section of each of the 50 WT2g's topics resembles a typical search engine query (2 to 3 words in length) [Zien 01], we used the title section as the keyword query for our search engine, Then, we ran the description and narrative paragraphs included with each topic through our classifier and used the obtained concepts as the concept input for the search engine. These descriptions and titles average 50 words in length. It is worth noting that we also manually determined the best matching concepts for each query and input the manually chosen concepts along with the query. However, since the difference between the results obtained using the automatically obtained concepts and the manually obtained ones is not significant (p-value of chi-square test ˜ 1), it is not discussed further. [ITTC 03]

4.1.3. Pruning Query Set

For testing the effectiveness of pruning, queries were chosen manually instead of from the TREC collection. The TREC query collection is a standardized set of query terms, with the relevant documents for each query explicitly indicated. Although these well-defined queries lend themselves to easy verification of search accuracy, the nature of their definition of relevancy eliminates several documents in the TREC collection that might have been considered "relevant" to a casual user of a search engine. Since the concepts for a search have been chosen manually, the well-defined relevancy of the "correct" results is undermined in any case.

Since the purpose is to test the efficiency of conceptual search for a lay user, it was decided to select a set of queries that are typical of a user searching for information on the Internet. A set of 24 queries, with 8 queries each of 1-word, 2-word and 3-word length, was selected for the experiment. This also helped to evaluate the performance of queries of different length with conceptual retrieval.

4.2 Setting the Baseline

We chose our baseline on experiments previously conducted with basic KeyConcept to tune several parameters. For example, the maximum and the minimum bounds of the number of training documents required for training were re-evaluated. Several other experiments were performed with respect to confirming the appropriate scoring formula and the correct α value. A list of the experiments performed is presented below. These results are described in full detail in [ITTC 03].

List of preliminary experiments performed:

1. To determine an upper bound for the number of documents
2. To determine a lower bound for the number of documents
3. To determine if the number of categories chosen affected precision
4. To determine the best weight of term in document and weight of term in concept
5. To determine effect of α on search precision
6. To evaluate per-query precision score comparison

4.3 Retrieval Based on Hierarchy

Once the basic parameters for retrieval, such as α and the number of training documents, were set, we proceeded to investigate which of the concepts in the hierarchy we should include in our conceptual search. It is our assumption that most misclassified data for a concept would be present in a very close neighbor of the concept in the hierarchy. There are several such "close" neighbors of a concept. In Figure 11, if the user-chosen concept is **C,** then the possible concepts that can be included are:

a) Its siblings, **D** and **E**

b) Its parent **B**

c) The children **F**, **G**, **H**, and **I**

d) The grandparent **A**

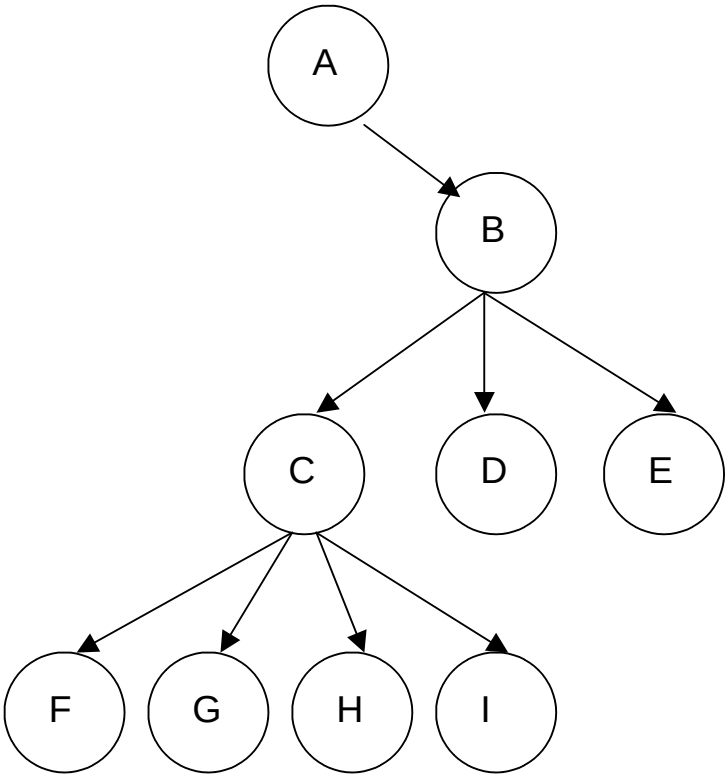

Figure 11. Neighborhood Hierarchy of a node

We decided to set the level of cut-off for enhanced KeyConcept at Level 4, one more level than previously used. This provided a wider range of concepts from which the user could choose and allowed us to expand level 3 concepts to their children in our experiments.

It was decided to test for all these cases by varying the weight for each of the possible expansion while computing the categories' relevance to the search. Combinations of the most promising cases were also tried in order to obtain the best possible results.

5. Experimental Results and Observation

5.1 Baseline Determination

The baseline system set various tuning parameters such as the minimum and maximum number of training documents required, based on results from previous research, which has been detailed in [ITTC 03]. It was found that the best formula for scoring a word in the query was TF * IDF * CDF, where TF refers to the pivoted normalized term frequency of the word in the collection, IDF is the inverse document frequency of the word in the entire collection and CDF is the frequency of the word in the particular concept found during training. We also conducted an experiment to determine the optimum value of α at four levels of the hierarchy (previous work only used three) and we found that $\alpha = 0.3$ was confirmed as the best value.

Figures 12-14 describe the process of retrieval on the online version of KeyConcept, which can be found at [KeyConcept 03]. Figure 12 shows the selection process where three categories the user is interested in are selected. The keywords are also typed for the retrieve. Figure 13 shows the results obtained for the search along with the weight of the document in the final result and a short summary of it. Figure 14 shows the top 10 categories as retrieved by KeyConcept for one of the result documents.

Figure 12. Keyword entry and concept selection in KeyConcept

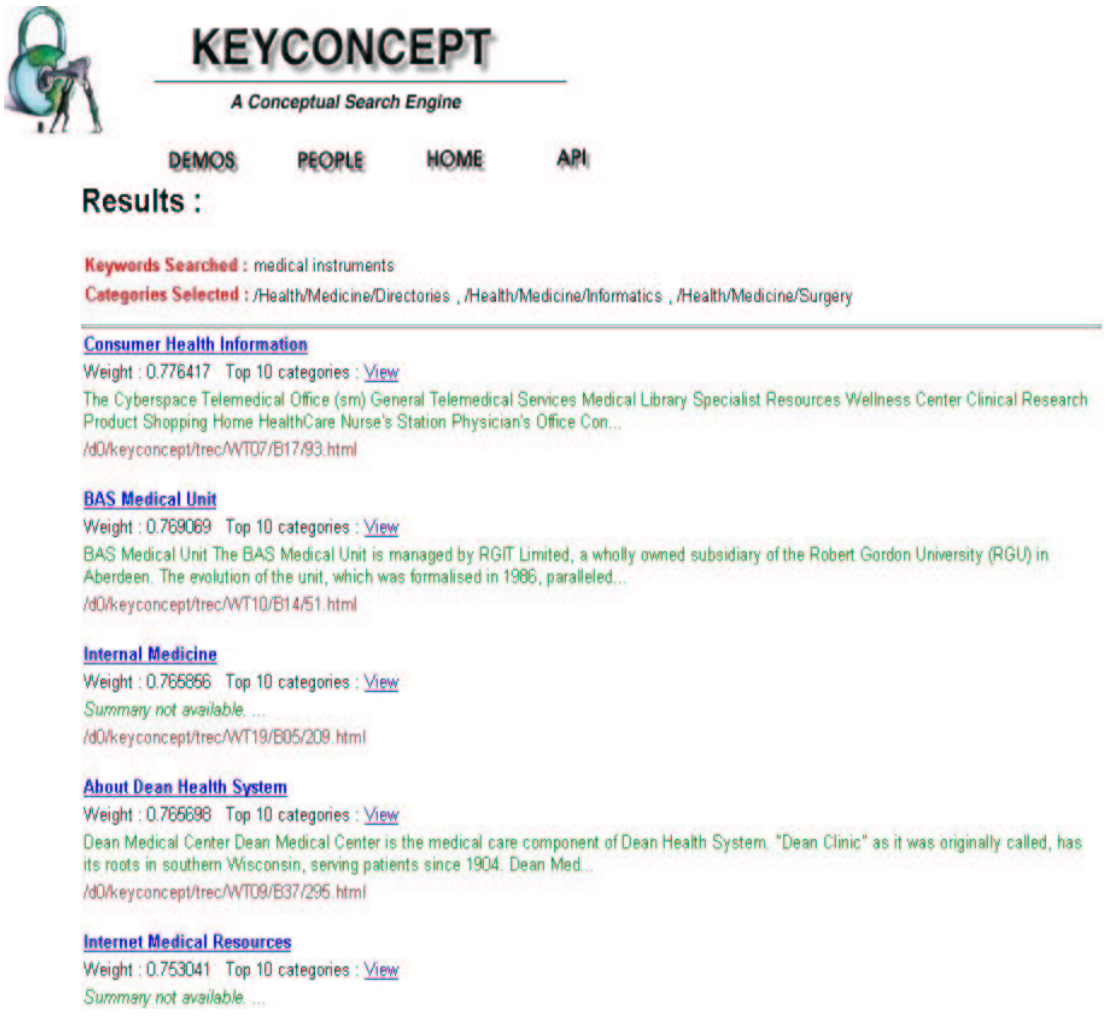

Figure 13. Search results after conceptual search

Figure 14. Top concepts for a document

5.2 KeyConcept Example

In this section, we show a sample illustration of the effectiveness of KeyConcept. For this purpose, we choose a query "rock" that is very ambiguous. Longer queries, involving multiple words, may disambiguate themselves (e.g., 'rock music'), but even they may be ambiguous in some cases (e.g., 'lions football'). In the following example, KeyConcept was used with no enhancements (such as pruning or hierarchical retrieval).

5.2.1. Results of keyword-only search

In this case, no categories were provided for enhancing the query. The query term "rock" was used and the search was performed. Figure 15 shows the results obtained. As seen in Figure 15, most results tend to pertain to rocks of a geological nature rather than any other meaning such as rock music, etc. If the user had been looking for rock music, only 2 of the top 10 results would have been relevant.

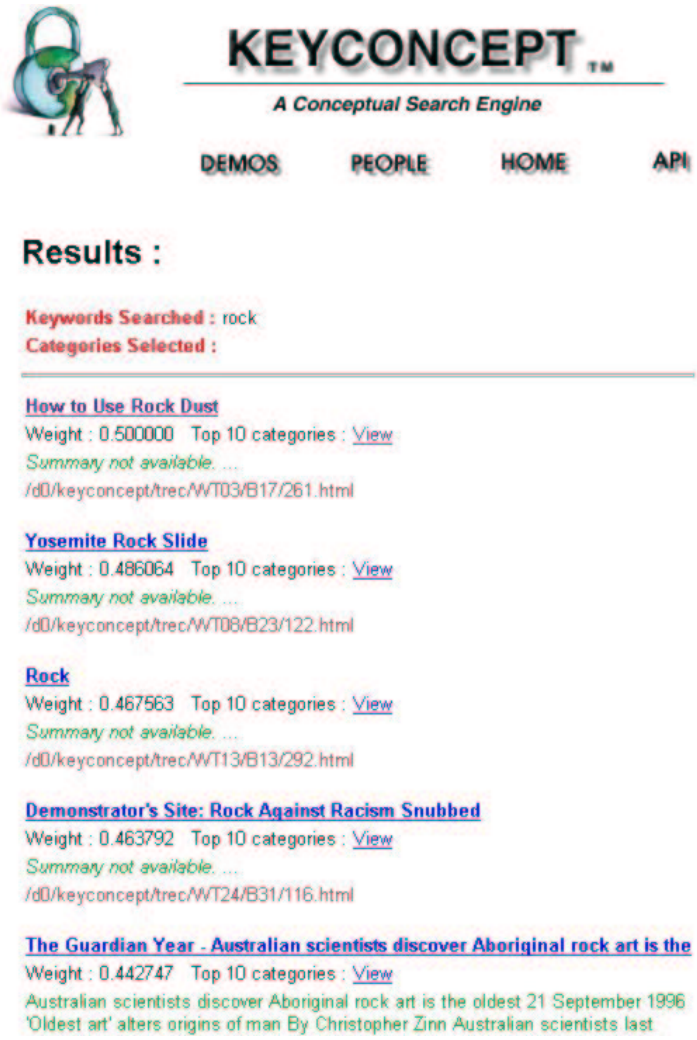

Figure 15. Results for a simple keyword search

5.2.2 Results with keyword and conceptual search

In this case, the same query term was used with in combination with a selected concept that had disambiguated the kind of results the user desired. The concept chosen was “Music Styles” for the query term “rock”. The results are shown in Figure 16. As expected, KeyConcept performs extremely well, retrieving results that are more related to music rather than geology or agriculture. 6 of the top 10 documents were relevant. The average precision increases from 20% for keyword-only search to 60% for keyword and conceptual search, an increase of 200%.

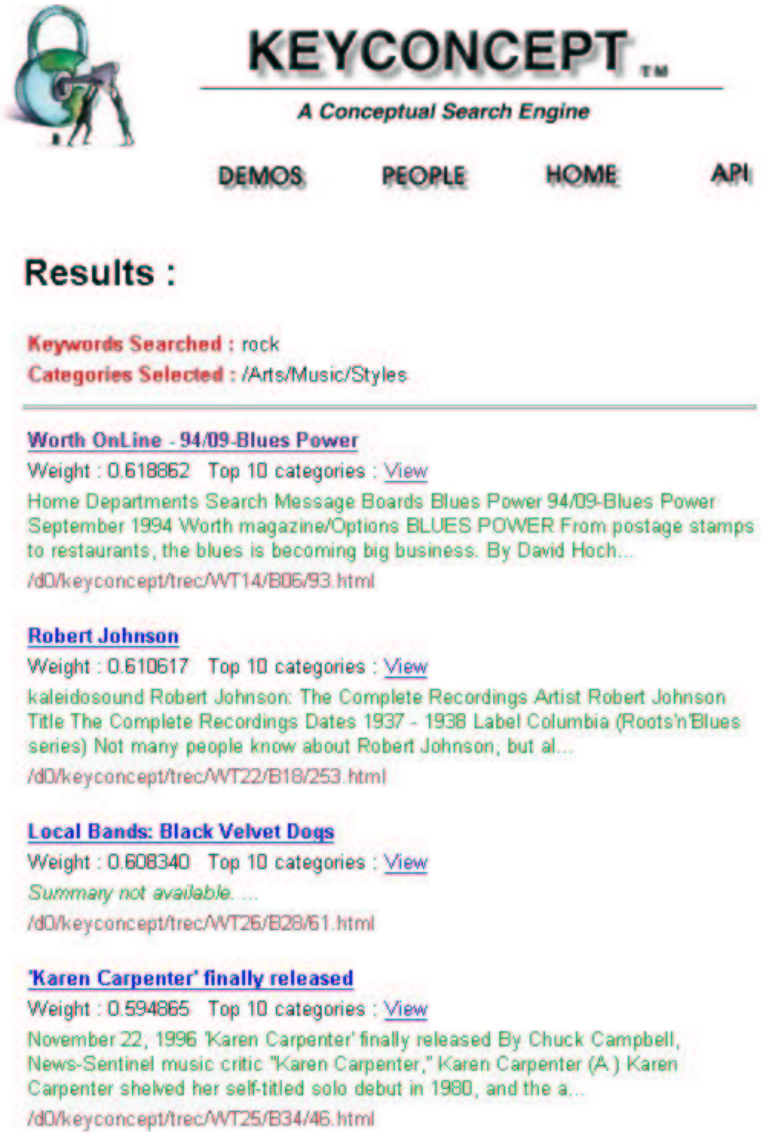

Figure 16. Results of Keyword '*rock*' + Concept '*music styles*'

In another case, the disambiguating power of KeyConcept was reaffirmed by checking the relevancy of the results for the same query term and choosing a concept that is most closely related to the term. That is, for the query term, “rock”, the most related concept of “Geology” was chosen. Here, 9 of the top10 results, or 90%, were relevant. Only one of the results was the same as the case where “Music/Styles” was chosen as the concept. Figure 17 shows the results of such a search.

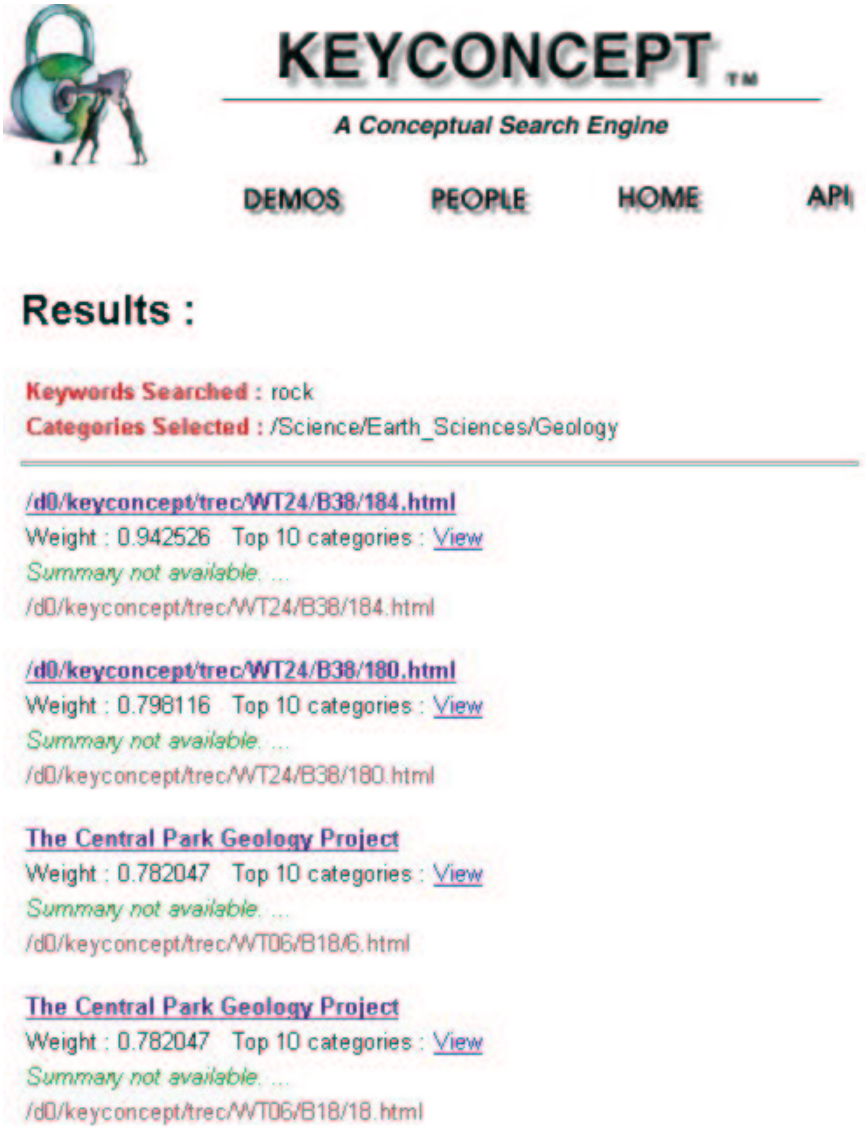

Figure 17. Results of Keyword '*rock*' + Concept '*Earth Science/Geology*'

5.3 Exploiting Hierarchical Relationships for Conceptual Pruning

Because we believe that KeyConcept will have the largest impact on shorter, more ambiguous, queries, we constructed three sets of queries of different lengths. These queries were taken at random from a set of actual user searches, extracted from the log of an online search engine. The first set consisted of query terms of length one, the second set with two query terms, and the third with three. Each set consisted of eight queries for which relevance judgments were provided manually. One relevant concept was chosen manually for each query in each set. The only exception was for the query *"north south korea"* where two categories *"Region/Asia/North_Korea"* and *" Regional/ Asia/South_Korea"* were chosen so that the query might be represented better.

Four sets of results were obtained for each query:

1) Pure Keyword Search
2) Keyword Search with Pruning
    2.1) Pruning at Level 1
    2.2) Pruning at Level 2
3) Keyword + Conceptual Search
4) Keyword + Conceptual Search with Pruning
    4.1) Pruning at Level 1
    4.2) Pruning at Level 2

For all the experiments, the baseline keyword search, with tuning parameter $\alpha = 0.3$ (as described in section 5.1) was used.

Table 1 shows the precision (% of relevant documents) in the top 10 results.

	K*	K+P		K+C	K+C+P	
		Level1	Level2		Level1	Level2
1-word queries	44%	48%	58%	54%	63%	71%
2-word queries	31%	38.4%	55%	49%	58%	65%
3-word queries	35%	49.4%	45%	51%	62%	65.2%
Total	36.7%	45.2%	52.7%	51.3%	61%	67.1%

Table 1. Average Precisions of 1-word, 2-word and 3-word queries

* K = Pure Keyword Based

K+P = Keyword + Pruning

K+C = Keyword + Conceptual Retrieval

K+C+P = Keyword + Conceptual Retrieval with Pruning

We can see that, compared to keyword search alone, we get an improvement of 82.83% (from 36.7% to 67.1%) in total overall precision when conceptual search plus pruning is used. These results will be discussed in more detail in the following sections.

5.3.1 Pure Keyword vs. Keyword with Pruning

Chart 1 shows the performance of KeyConcept when only keyword matching is done compared to the results when the result set is pruned to remove documents that do not contain the user-selected concept within their top three best-matching concepts (as determined at index time).

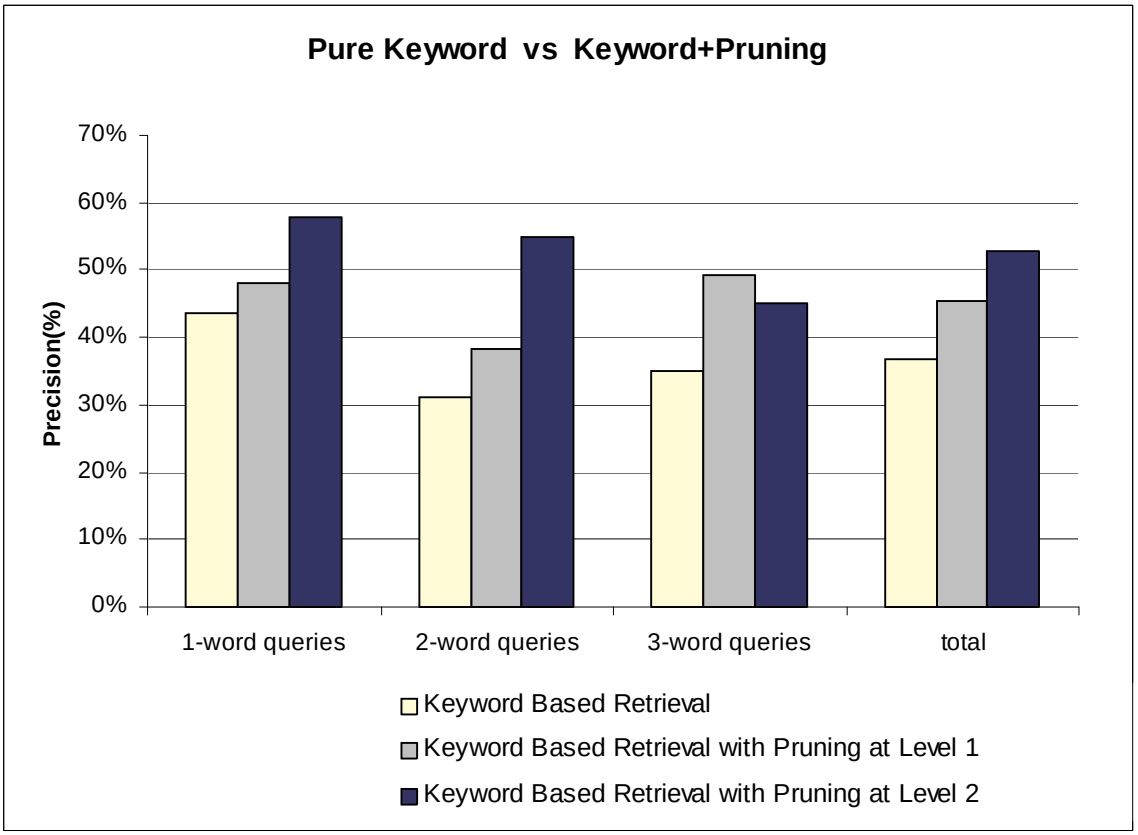

Chart 1. Comparison of precision of pure keyword vs. keyword with pruning

The total average precision increases from 36.7% to 45.24% for level-1 pruning ($p=0.00767$) and from 36.7% to 52.67% ($p=0.0031$) for level-2 pruning. Both the improvements are significant.

5.3.2 Pure Keyword vs. Keyword + Conceptual Retrieval

Chart 2 compares the performance of keyword-only matching to results where a combination of keyword and conceptual matching is used. No pruning is done on any search. The results of the two sets of experiments of pruning at Level1 and Level 2 have been averaged to yield a single precision value.

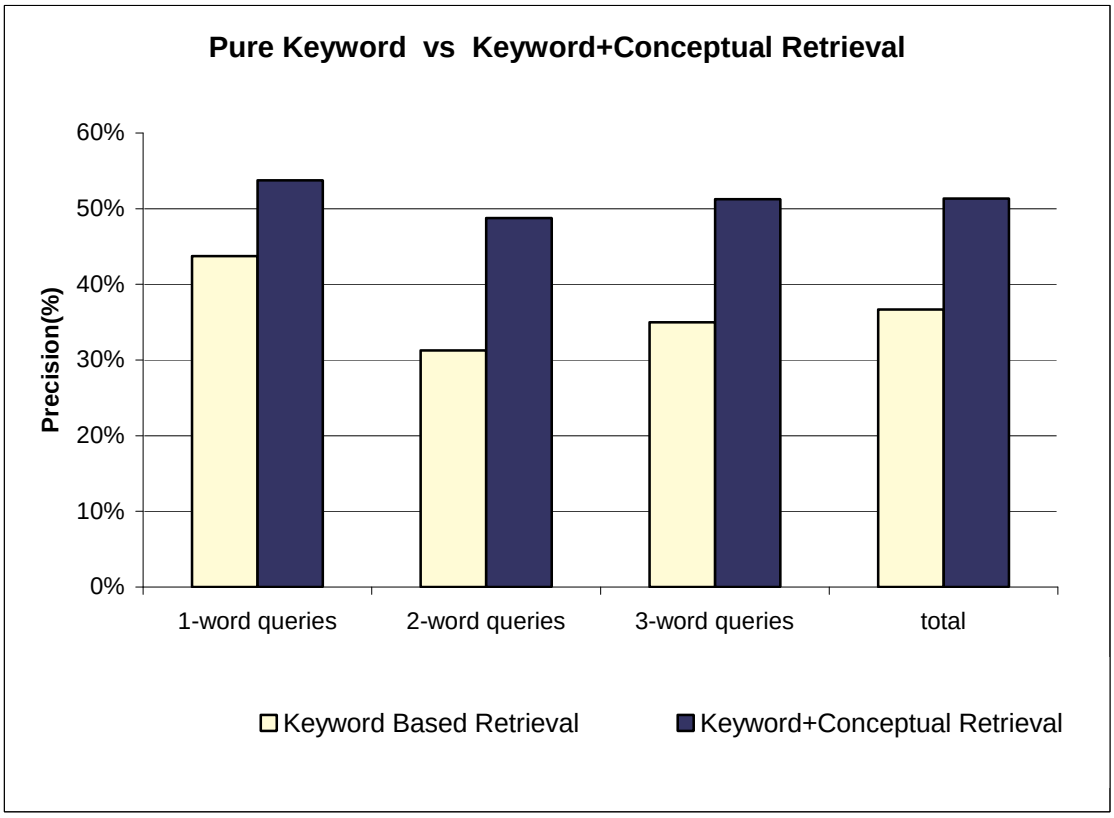

Chart 2. Comparison of precision of pure keyword vs. keyword with conceptual retrieval

Precision within the top 10 results increases significantly ($p=6.81 \times 10^{-5}$) from 36.7% to 51.3% when conceptual search is used.

5.3.3 Pure Keyword vs. Keyword + Conceptual Retrieval with Pruning

A combination of the both the techniques, i.e., conceptual retrieval and pruning yields very high precision values for all sets of queries. Chart 3 shows the effects of combining the two techniques.

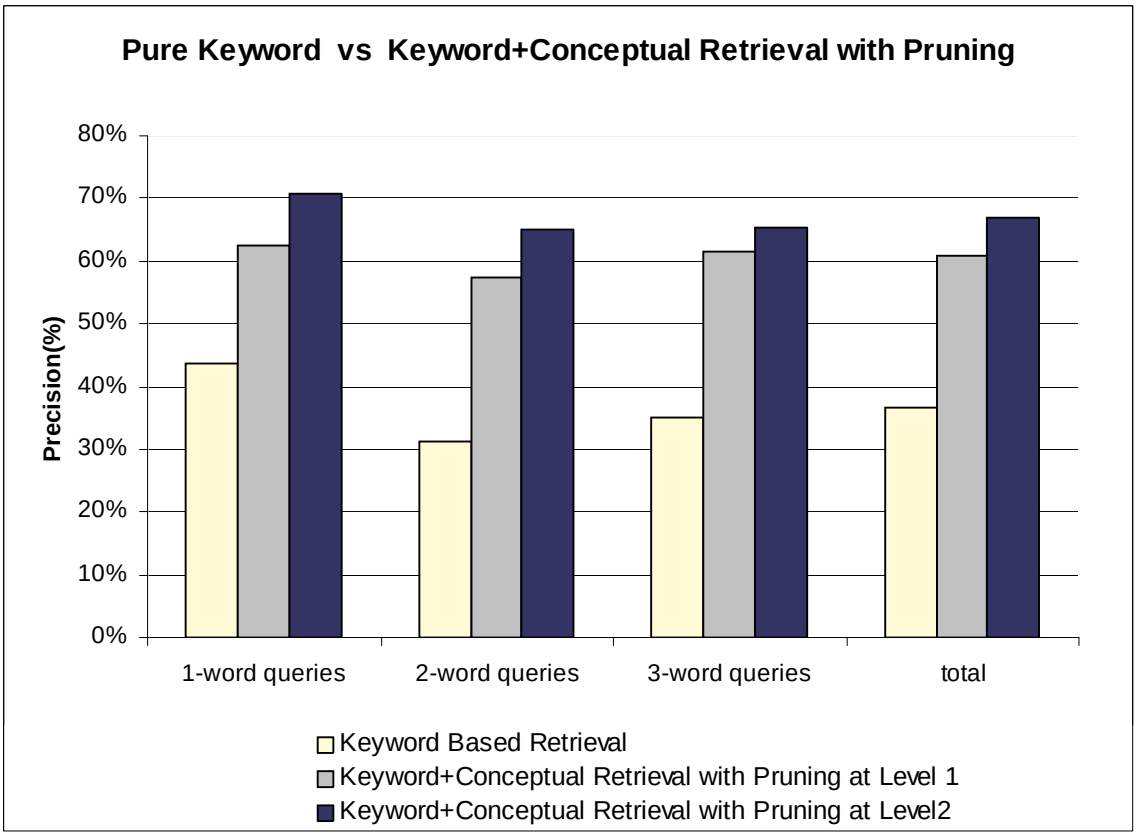

Chart 3. Comparison of precision of pure keyword vs. keyword + conceptual retrieval with pruning

There is an overall significant increase in precision in all three cases. The total average precision increases from 36.7% to 61% for level-1 pruning ($p=4.8 \times 10^{-6}$) and from 36.7% to 67.1% ($p=2.62 \times 10^{-6}$) for level-2 pruning. Both improvements are significant.

5.3.4 Comparison of Retrieval Methods

The overall performance of the above-mentioned systems can be compared now. Results from the previous experiments are combined and shown in Chart 4. Considering simple keyword-based retrieval as the baseline, we see that each subsequent method outperforms it by a significant amount. The most marked increase in precision is obtained in the final case, where keyword retrieval is used along with conceptual retrieval and pruning. Furthermore, from Table 1, we see that the maximum precision among all experiments performed is obtained for single word queries, when pruned at Level 2 (71%). The precision results previously described in section 5.3.3, i.e., keyword + conceptual retrieval with pruning, remain consistently high (above 60%).

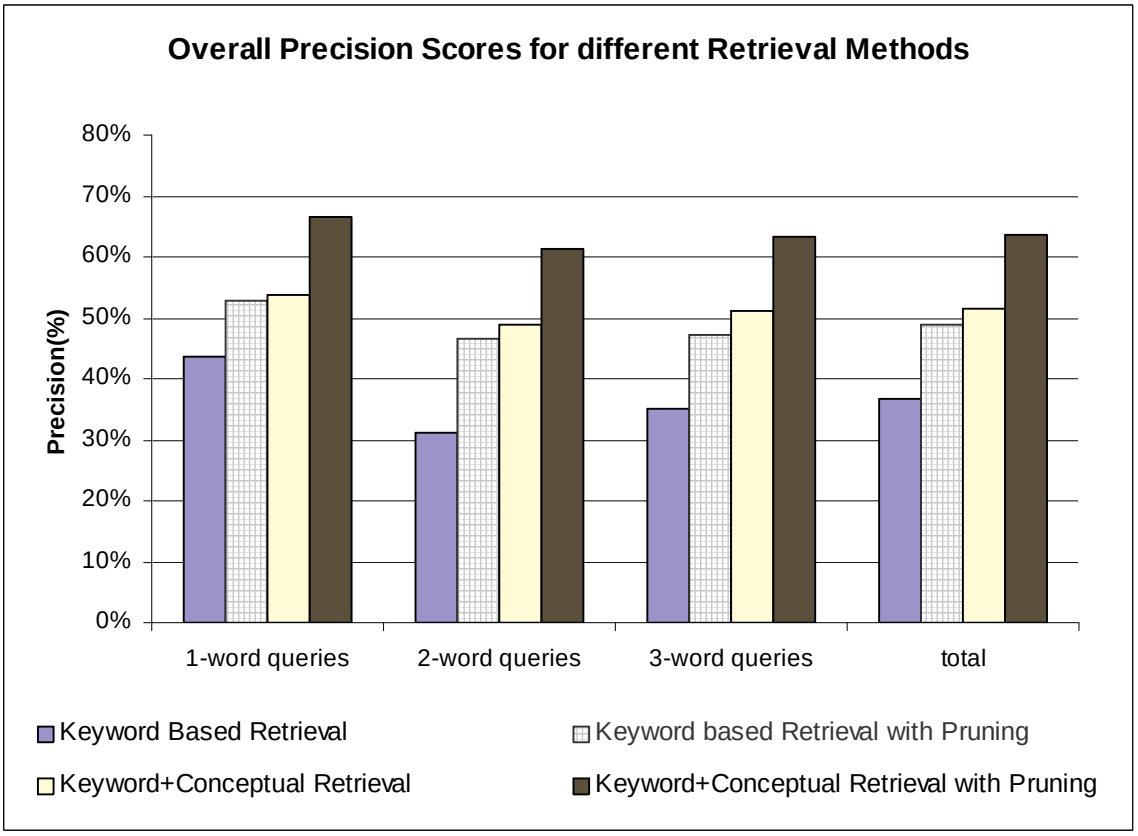

Chart 4. Comparison of retrieval methods

5.4. Exploiting Hierarchical Relationships for Conceptual Retrieval

In this experiment, we tried to estimate the appropriate weight for each kind of hierarchical relation to the chosen concept. It is believed that a suitable combination of suitably weighted hierarchical neighbors, along with the chosen concept, would obtain the best results in conceptual search.

Two factors were varied: a) The weights assigned to the adjacent concept were varied from 0.1 to 1.0 and b) the number of top concepts used for each query was varied from 1 to 4. The following nodes close to the user's concept in the hierarchy were tested:

1) Sibling
2) Parent
3) Children
4) Grandchildren

For all the following experiments, the TREC dataset was used as the test data.

5.4.1 Siblings

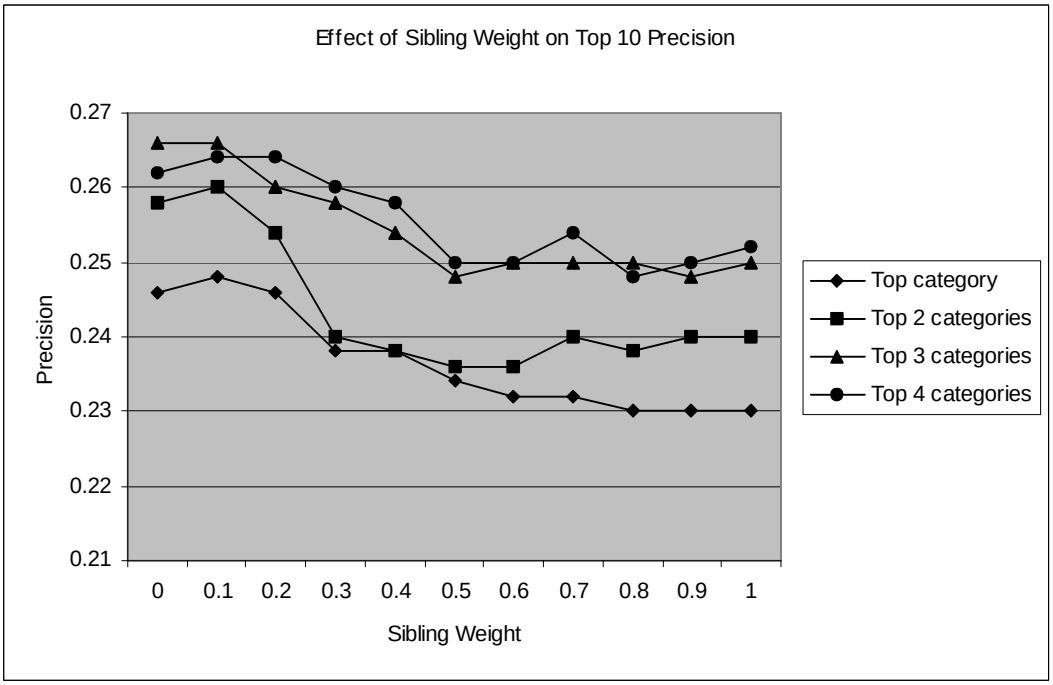

Chart 5. Effect of adding siblings

Chart 5 shows the results of the effect of adding siblings of the concept to the conceptual search. The search precision experiences a slight, non-significant increase at a weight of 0.1 and then begins to decrease. This indicates that the addition of siblings increases the noise in the set of retrieved documents and does not help much in obtaining better results in our collection.

5.4.2 Parent

The effect of adding the parent of a user chosen concept is shown in Chart 6. As before, the α value is set at 0.3 and the number of concepts is set at different values from 1 to 4.

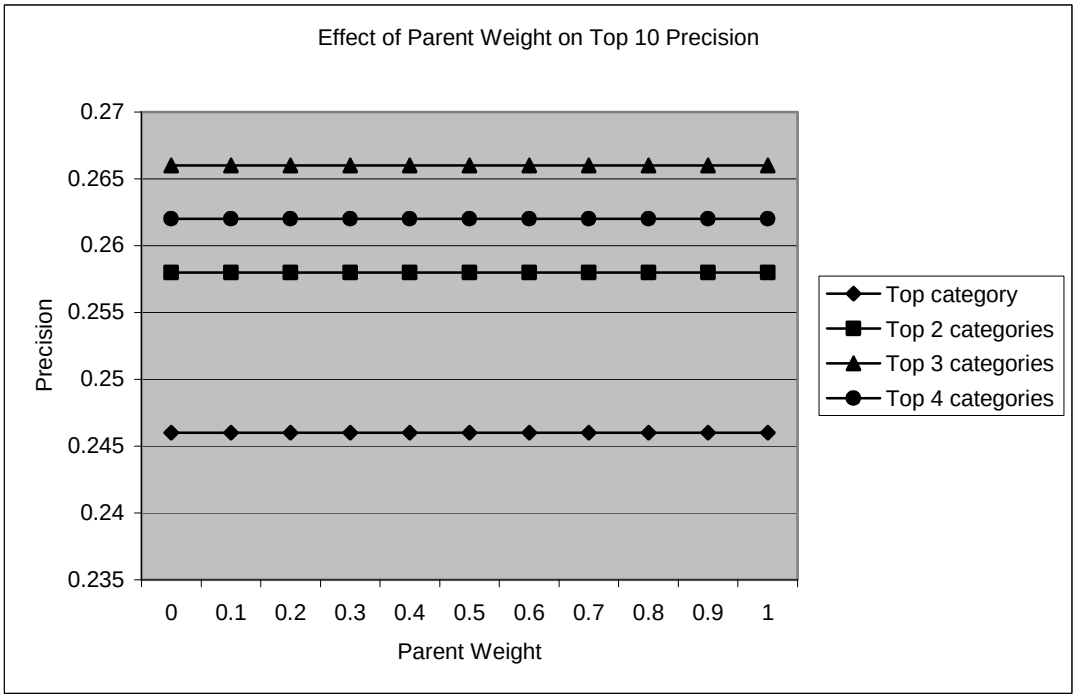

Chart 6. Effect of adding parent

We found from preliminary experiments that there was not enough content for indexing in the first and second levels of the ODP hierarchy. We see our expectations confirmed in the results. The addition of the parent of a concept does not really change the search precision in any way.

5.4.3 Children

Chart 7 shows the effect of adding all the children of a concept to the search. All children are given equal weights and the weights are, as before, varied from 0.1 to 1.0.

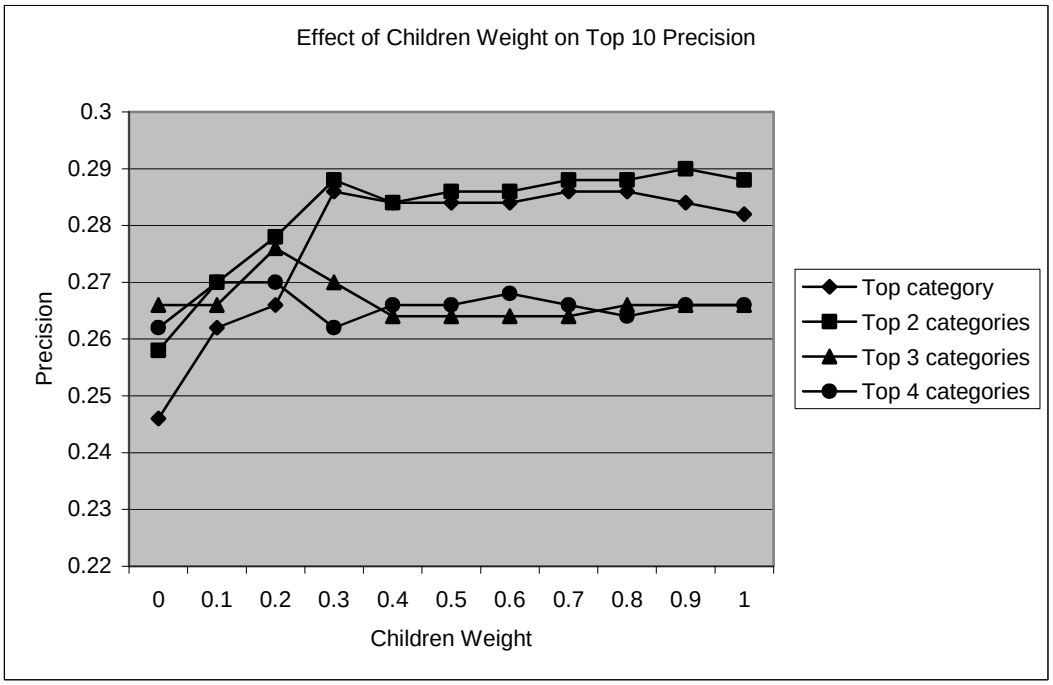

Chart 7. Effect of adding children

A maximum precision of 28.5% is obtained at a weight of 0.3 for one concept. It also shows the maximum increase of precision of 16.32% from 24.5% to 28.5%(p=0.15). This observation hints that adding children may help when the user chooses only one concept. The increase in precision, when 3 and 4 concepts are being used, is not significant. The noise in the set of selected documents increases as more and more related concepts are added to an already large set of chosen concepts

5.4.4 Grandchildren

Grandchildren of a chosen concept were added too, and Chart 8 details the effects of such including them in the search.

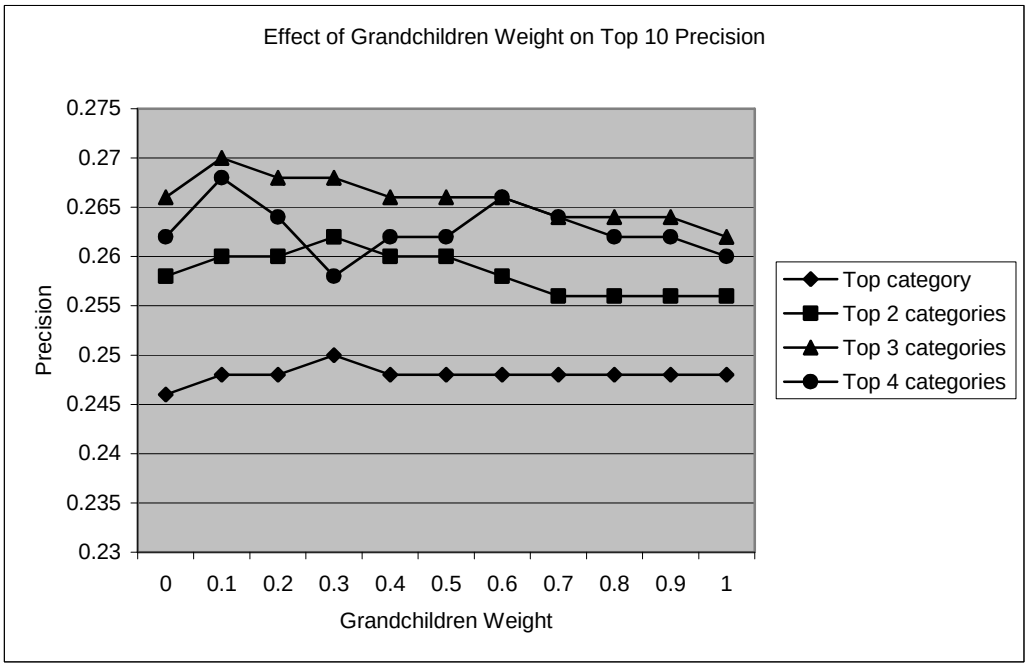

Chart 8. Effect of adding grandchildren

Search precision increases slightly up till a weight of 0.1 and then begins to even out. The maximum precision is obtained for 3 concepts at 27%. Although this increase is not significant, a suitable combination of different concepts could bring about a better precision.

5.4.5 Hierarchical Combinations

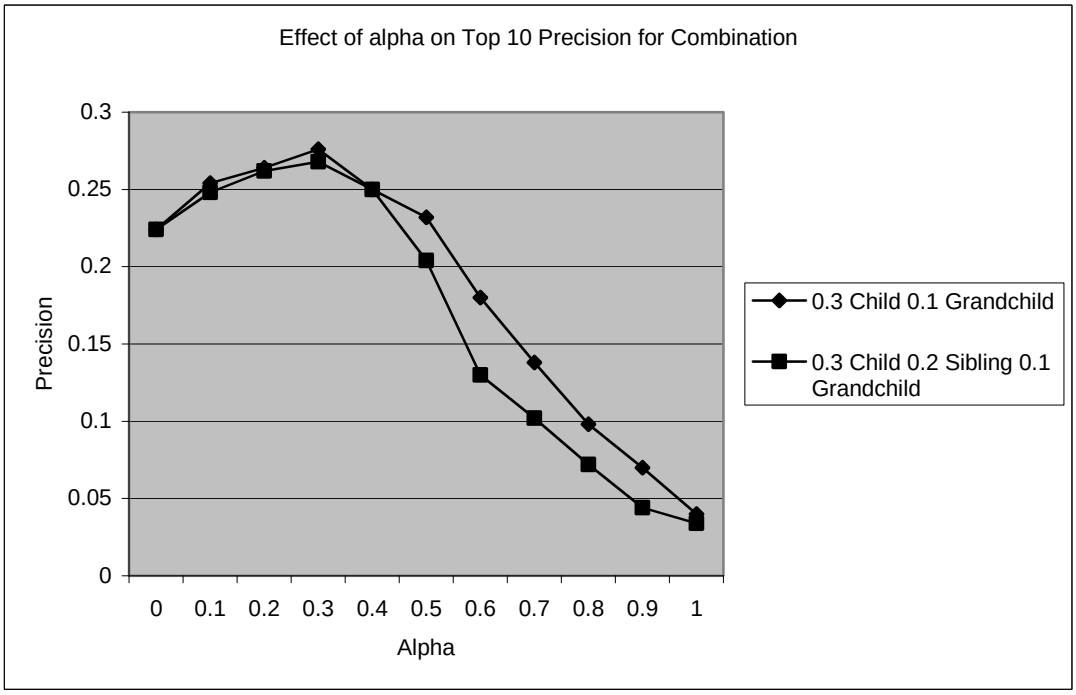

Chart 9. Combinations of related concepts

Two possible combinations are shown in Chart 9. One combination selects the most promising concept-relations i.e. children and grandchildren, at their best weights of 0.3 and 0.1 respectively and varies α from 0.0 to 1.0. Combination 2 includes 3 possible concept relations – child, sibling and grandchildren at weights 0.3, 0.2 and 0.1 respectively and α is varied similarly. The former combination attains a maximum precision of 27.6% at an $\alpha = 0.3$. We also see that the addition of siblings merely increases the noise in the set of documents retrieved for a query.

This maximum precision is higher than that attained while just using grandchildren in the search (27%) but lower than the maximum precision attained in the previous experiment with just children (28.5%). Thus the system that just uses children of a user's concept in the search is found to fare better than combinations of children and grandchildren.

6. Conclusions and Future Work

6.1 Conclusions

This thesis presented the idea that hierarchical information in the ontology might help in obtaining better search precision in conceptual retrieval. It aimed to evaluate different possibilities of using the neighboring nodes in a hierarchy to increase the conceptual search domain. Several combinations of the best hierarchical relatives of the concept were also tried to estimate whether such combinations helped in further increasing precision results. Pruning the obtained results based on the user's selections was also investigated.

Various combinations of pruning, keyword retrieval and conceptual retrieval were tested. The best precision results occurred when conceptual retrieval was performed at $\alpha = 0.3$ with the obtained results being pruned. In fact, the overall precision in the top 10 results jumps from 36.7% to 63.77%, a significant improvement ($p=0.0038$). The increase in precision with the use of the method just mentioned remained consistently above 50% for queries of different word-lengths. This seems to indicate the applicability of the pruning method uniformly to all queries.

Training of the conceptual search engine was done with a 4-level ontology. It was found that the best results were obtained when a single concept's children was selected for search along with key-word search. There was a significant increase of precision for the top-10 documents from 24.6% to 28.6% when the children of the chosen concept were

included for conceptual retrieval at a weight of 0.3. Even for the previously determined value of three chosen concepts, the enhanced system with children increased search precision from 26.6% to 27.6%

Other “relatives” of the chosen concepts, such as siblings, parents and grandchildren were also evaluated. The grandchildren were seen to give a slight increase in precision of 0.4 (from 26.6% to 27%) at a weight of 0.1. Siblings and parents did not increase accuracy but decreased and maintained it, respectively.

Suitable combinations of grandchildren and children were evaluated since they gave the best results in the above experiments. The combination did not seem to improve precision noticeably than just using children. This was probably due to the fact that with the addition of more concepts for search, more noise was also introduced in the retrieved document collection.

6.2 Future Work

The enhanced version of KeyConcept has succeeded in improving the overall efficiency of the conceptual search engine. The availability of such a stable conceptual retrieval system can act as a baseline for further research into alternate or better methods to improve current precision results. A few of the possible enhancements that can be done in the future are listed in the following sections.

6.2.1 Better Data

As seen in the results for hierarchical retrieval using parents, there is at present not enough data in the top two levels of the ODP hierarchy for suitable training. With either the use of a hierarchy with more data in the upper levels, or by somehow training concepts for the upper levels, we believe that the parent nodes in a hierarchy can be used more effectively.

6.2.2 Hierarchical Classification

Instead of performing a hierarchical retrieval, it is possible to start indexing documents hierarchically. In this method, adjustments have to be made while indexing to keep track of hierarchically related concepts. In theory, this method would involve a lengthier indexing process and extra data structures for keeping track of hierarchical relationships between nodes.

6.2.3 Contextualization

An ongoing area of research in the area of information retrieval is contextualization. In contextualization, we take into account the related activity that is occurring while the user is searching for information. This may include the text in the windows that are open on the searcher's desktop, previous websites accessed in a browser before the search was made, etc. For example, if the user has the website for "ESPN SportsCenter" open and is searching using the query term "packers", we may infer that the user is interested in searching about a sports team called "packers". Relevant concepts according to this inference may be chosen and included for the search. In KeyConcept, a relatively

accurate method to accurately capture text/content from adjacent windows and inferring relevant concepts from it needs could be developed and integrated.

6.2.4 Personalization

Currently, the concepts for each query must be provided manually or by running a text related to the query through a classifier. Most users see this as a burdensome task. KeyConcept's main thrust is to provide a comprehensive conceptual search framework in the future. This means that each user does not have to choose concepts manually anymore. The engine would compare the query to a user's profile and choose the best concepts to add to the search. A user's profile can be gathered by keeping track of his/her previous queries and maintaining a record of the concepts he's interested in. The relevant concepts for a user can be arranged in a hierarchical fashion too. Thus an ontology mapping could be made between the user's personal hierarchy and the general conceptual hierarchy. Thus, a completely automated conceptual search engine would, in the future, integrate both personalization and contextualization to obtain results better suited to the user.

References

[Cai 03] Lijuan Cai, Thomas Hofmann. Text Categorization by Boosting automatically Extracted Concepts. In Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, Toronto, Canada, July/August 2003 pp. 182-189

[Chaffee 00] Jason Chaffee, Susan Gauch. Personal Ontologies For Web Navigation. In Proceedings of the 9th International Conference On Information Knowledge Management (CIKM), 2000, pp. 227-234.

[Chekuri 97] Chandra Chekuri, Michael H. Goldwasser, Prabhakar Raghavan, Eli Upfal. Web Search Using Automatic Classification. In Proceedings of the 6th International WWW Conference. Santa Clara (CA), USA, 1997.

[Cui 02] Hong Cui, P. Bryan Heidorn, Hong Zhang. An approach to automatic classification of text for information retrieval. In Proceedings of the second ACM/IEEE-CS, Joint Conference on Digital libraries Portland, Oregon, USA, July 2002, pp. 96 – 97

[Gauch 04] Susan Gauch, Jason Chaffee, and Alexander Pretschner, Ontology-Based User Profiles for Search and Browsing", Web Intelligence and Agent Systems, (in press).

[Glover 01] E. Glover, G. Flake, S. Lawrence, W. Birmingham, A. Kruger, C. Giles, and D. Pennock. Improving Category Specific Web Search by Learning Query Modifications. In Proceedings of the Symposium on Applications and the Internet, SAINT 2001, San Diego, CA, January 2001,pp. 23-31

[Guarino 99] N. Guarino, C. Masolo, and G. Vetere, OntoSeek: Content-Based Access to the Web. IEEE Intelligent Systems, 14(3), May 1999, pp. 70-80.

[Hawking 99] David Hawking, Ellen Voorhees, Nick Craswell, and Peter Bailey. Overview of the TREC8 Web Track. In Eighth Text REtrieval Conference (TREC-7), November 1999.

[Heflin 2000] Jeff Heflin, James Hendler. Dynamic Ontologies on the Web. In Proceedings of 17th National Conference on Artificial Intelligence (AAAI), 2000.

[ITTC 03] Susan Gauch, Devanand Ravindran, Subhash Induri, Juan Madrid, and Sriram Chadalavada, Internal Technical Report ITTC-FY2004-TR-8646-37, Information and Telecommunication Technology Center, University of Kansas

[Kato 99] Tsuneaki Kato, Shigeo Shimada, Mutsumi Kumamoto, Kazumitsu Matsuzawa. Idea-Deriving Information Retrieval System. In Proceedings of the First NTCIR Workshop on Research in Japanese Text Retrieval and Term Recognition, Tokyo, Japan, August 30 - September 1, 1999.

[KeyConcept 03] KeyConcept. http://ittc.ku.edu/keyconcept/

[Klink 02] Stefan Klink, Armin Hust, Markus Junker, Andreas Dengel. Collaborative Learning of Term-Based Concepts for Automatic Query Expansion**.** In Proceedings of ECML 2002, 13th European Conference on Machine Learning, Helsinki, Finland, August 19-23, 2002, pp 195-206

[Knight 94] K. Knight and S.K. Luk. Building a Large-Scale Knowledge Base for Machine Translation. In Proceedings of the 12th National Conference on Artificial Intelligence (AAAI), 1994, volume 1, pp. 773-778.

[Krovetz 92] Robert Krovetz and Bruce W. Croft. Lexical Ambiguity and Information Retrieval. ACM Transactions on Information Systems, 10(2), April 1992, pp. 115-141.

[Labrou 99] Yannis Labrou, Tim Finin. Yahoo! As An Ontology – Using Yahoo! Categories To Describe Documents. In Proceedings of the 8th International Conference On Information Knowledge Management (CIKM), 1999, pp. 180-187.

[Liu 02] Fang Liu, Clement Yu and Weiyi Meng. Personalized web search by mapping user queries to categories. In Proceedings of the 11th International Conference on Information and Knowledge management, McLean, Virginia, USA, 2002, pp. 558 - 565

[Lu 99] Fenghua Lu, Thomas Johnsten, Vijay Raghavan, Dennis Traylor. Enhancing Internet Search Engines to Achieve Concept-based Retrieval. In Proceedings of InForum '99 - Improving the Visibility of R & D information, Oak Ridge, Tennessee, May 1999.

[Matsuda 99] Katsushi Matsuda, Toshikazu Fukushima. Task-Oriented World Wide Web Retrieval By Document Type Classification. In Proceedings of the 8th International Conference On Information Knowledge Management (CIKM), 1999, pp. 109-113.

[NorthernLight] Northern Light. http://northernlight.com

[ODP] Open Directory Project. http://dmoz.org

[Pazzani 96] Michael Pazzani, Jack Muramatsu, Daniel Billsus. Syskill & Webert: Identifying Interesting Web Sites. In Proceedings of the 13th National Conference On Artificial Intelligence, 1996, pp. 54-61.

[Pearce 97] Claudia Pearce, Ethan Miller. The TellTale dynamic hypertext environment: Approaches to scalability. In Advances in Intelligent Hypertext, Lecture Notes in Computer Science. Springer-Verlag, 1997.

[Pitkow 02] James Pitkow, Hinrich Schütze, Todd Cass, Rob Cooley, Don Turnbull, Andy Edmonds, Eytan Adar, Thomas Breuel. The consumer side of search: Personalized search. In the Communications of the ACM, September 2002. pp. 50-55

[Ruiz 99] Miguel Ruiz, Padmini Srinivasan. Hierarchical Neural Networks For Text Categorization. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 1999, pp. 281-282.

[Sebastiani 02] Fabrizio Sebastiani. Machine Learning in Automated Text Categorization – In ACM Computing Surveys 34(1), 2002, pp. 1-47

[SES 03] Search Engine Showdown: Size statistics. http://www.searchengineshowdown.com/stats/sizeest.shtml

[Singhal 96] Amit Singhal, Chris Buckley, Mandar Mitra, Pivoted Document Length Normalization. 19th ACM Conference on Research and Developemnet in Information Retrieval(SIGIR), 1996.

[Tirri 03] H.Tirri, Search in vain: challenges for Internet search. In Computer, Volume 36, Issue 1, Jan 2003, pp. 115-116

[Yang 03] Y. Yang, J. Zhang and B. Kisiel - A scalability analysis of classifiers in text categorization . In Proceedings of 26th Annual International ACM SIGIR Conference,July-Auguest 2003, pp 96-103.

[YAHOO] Yahoo! http://www.yahoo.com

[Zhu 99] Xiaolan Zhu, Susan Gauch, Lutz Gerhard, Nicholas Kral, Alexander Pretschner. Ontology-Based Web Site Mapping For Information Exploration. In Proceedings of the 8th International Conference On Information Knowledge Management (CIKM), 1999, pp. 188-194.

[Zien 01] Jason Zien, Jörg Meyer, John Tomlin, Joy Liu. Web Query Characteristics And Their Implications On Search Engines. In Proceedings of the 10th International WWW Conference. Hong Kong, China, 2001.

Appendix

List of queries in pruning query set and associated concepts

Query terms	Concepts
1-word queries	
diamond	/Shopping/Jewelry/Diamonds
yacht	/Recreation/Boating/Sailing
sitcoms	/Arts/Television/Programs
depreciation	/Business/Investing/Real_Estate
smithsonian	/Reference/Museums/Museum_Resources
chat	/Computers/Internet/Cyberspace
Siamese	/Recreation/Pets/Cats
conservative	/Society/Politics/Conservative
2-word queries	
alien life	/Science/Astronomy/Extraterrestrial_Life
joint pain	/Health/Conditions_and_Diseases/Musculoskeletal_Dis-orders
remove weed	/Home/Gardens/Plants
ancient incas	/Regional/South_America/Peru
aircraft carrier	/Society/Military/Ships
propeller plane	/Recreation/Aviation/Aircraft
amino acids	/Science/Biology/Genetics
fairy tales	/Arts/Literature/Children's_Literature
3-word queries	
national parks animals	/Recreation/Outdoors/Wildlife
north south korea	/Regional/Asia/North_Korea , /Regional/Asia/South_Korea
skills development programs	/Business/Human_Resources/Training_and_Safety
cheap pets drugs	/Recreation/Pets/Health
new york yankees	/Sports/Baseball/Major_League
child day care	/Home/Family/Childcare
cheap airfare deals	/Recreation/Travel/Budget
software life cycle	/Computers/Software/Software_Engineering