NetSerf: Using Semantic Knowledge to Find Internet Information Archives

Anil S. Chakravarthy and Kenneth B. Haase
MIT Media Laboratory
{anil, haase}@media.mit.edu

Abstract

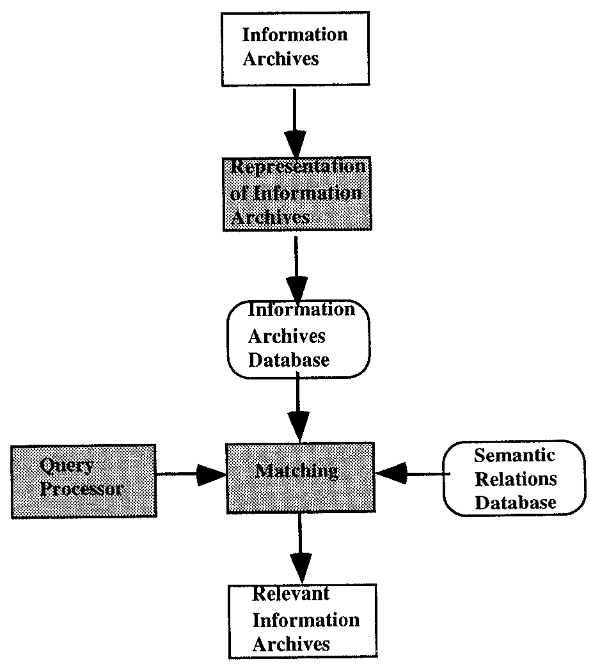
This paper describes the architecture, implementation and evaluation of NetSerf, a program for finding information archives on the Internet using natural language queries. NetSerf's query processor extracts structured, disambiguated representations from the queries. The query representations are matched to hand-coded representations of the archives using semantic knowledge from WordNet (a semantic thesaurus) and an on-line Webster's dictionary. NetSerf has been tested using a set of questions and answers developed independently for a game called Internet Hunt. The paper presents results comparing the performance of NetSerf and the standard IR system SMART on this set of queries.

1 Introduction

The Internet is now one of the world's largest repositories of information. Since the information is widely distributed, we can view the process of finding information on the Internet as involving two steps: locating relevant information archives, and then searching those archives for relevant information items. For example, if we are looking for pictures of birds of the Amazon rainforest on the Internet, the first task of the retrieval system is to identify archives that might contain the desired pictures, e.g., the archive of photographs at the Smithsonian Institution. This paper describes a system named NetSerf that tries to find relevant information archives in response to user queries.

NetSerf can be used to represent any information archive that is organized around a theme, where the description of the archive is a generalization of the descriptions of its contents. Therefore, in NetSerf, an archive is considered relevant to a query if the query can be generalized to the archive's description using semantic knowledge. For example, NetSerf considers the World Factbook archive, whose description is "World facts listed by country," as relevant to the query "What is the primary religion in Somalia?", since its semantic knowledge database contains the fact that Somalia is a country. Thus, NetSerf's mechanism is somewhat analogous to the process of locating a book in a library by searching through a more general section (in contrast to literal pattern-matching tools like Archie, Veronica and WAIS [Schwartz et al. 1992]).

FIGURE 1. Architecture of NetSerf[1].



The architecture of NetSerf is shown in Figure 1. The semantic knowledge database is a combination of semantic relations from WordNet [Miller 1990], and semantic relations extracted automatically from an on-

1. Regular boxes denote external data, shaded boxes denote processes, and rounded boxes denote internal representations.

line Webster's dictionary. Descriptions of information archives are stored in the form of structured, frame-like representations [Minsky 1974]. The query processor turns natural language queries into structured, disambiguated representations, which are then matched to the archive descriptions using the semantic knowledge database.

NetSerf has been evaluated on a set of queries collected from the game *Internet Hunt* [Gates 1992]. Each month, the creator of this game publishes ten questions (e.g., the Somalia question above), which participants are expected to answer using only information available on the Internet. The correct answers are published the following month. Thus, this game provides an independent set of queries with answers for testing NetSerf. Using this query set, we have compared the performance of NetSerf to that of the well-known information retrieval system SMART [Salton 1989].

The organization of the paper is as follows: In Section 2, we will describe in greater detail the semantic knowledge sources used by NetSerf. This section also provides a brief overview of some of the existing research on the use of semantic relations in information retrieval. In Section 3, we will describe the process of hand-coding representations of information archives. Section 4 deals with the query processor and the Internet Hunt questions that are used to evaluate NetSerf. Then, Section 5 describes the mechanism used to match query representations to archive representations, and to rank hits. In Section 6, we will present the results of running NetSerf and SMART on the set of Internet Hunt queries. Section 7 concludes the paper with a summary and suggestions for future work.

2 Semantic Relations in Information Retrieval

Semantic relations are structures which link words to related words, and which often indicate the type of the relationship, e.g. A-KIND-OF(lion, animal). This section provides an overview of techniques for manual and automatic acquisition of *semantic networks,* which are networks composed of semantic relations [Quillian 1968], and an overview of prior work on the use of semantic relations in retrieval. The final part of the section describes how our technique for incorporation of semantic knowledge into retrieval differs from previous approaches.

Many IR systems have used domain-specific semantic networks for text retrieval, e.g. [Cohen & Kjeldsen 1987, Rada & Bicknell 1989]. For instance, [Rada & Bicknell 1989] use a network called MeSH that relates medical topics to more general topics, e.g., "rheumatoid arthritis" to "rheumatism." But, several broad-coverage, domain-independent semantic networks have also been built in recent years, among them, WordNet [Miller 1990], ConText® [Oracle 1993], and CYC [Lenat & Guha 1990]. Since NetSerf uses WordNet extensively, we will describe it in detail here.

WordNet is a large, manually-constructed semantic network built at Princeton University by George Miller and his colleagues. The basic unit of WordNet is a set of synonyms, called a *synset*, e.g., [go, travel, move]. A word (or a word collocation like "rural area") can occur in any number of synsets, with each synset reflecting a different sense (meaning) of the word. WordNet1.3, the version used by NetSerf, is quite large, with well over 30,000 synsets and 60,000 senses. It provides a variety of semantic relations for nouns, verbs, adjectives and adverbs. WordNet is organized around a taxonomy of *hypernyms* (A-KIND-OF) and *hyponyms* (inverse of A-KIND-OF). Other relations used to link synsets in WordNet are ANTONYM-OF, SUBSTANCE-OF, PART-OF, MEMBER-OF (and their inverses), ENTAILS, CAUSES, and PERTAINS-TO, as appropriate.

Turning from manual to automatic acquisition of semantic relations, a distinction can be made based on whether the system only learns which words are related, or also learns the type of the relationship. Many IR systems have acquired and used term co-occurence data that reveal which word pairs typically occur together in a collection. A high degree of co-occurence between terms implicitly indicates that they are related, even though the system does not know the type of the relationship.

In the computational linguistics community, substantial effort has been devoted to the extraction of databases of typed semantic relations from on-line dictionaries, e.g. [Amsler 1980, Chodorow et al. 1985, Fox et al. 1988, Dolan et al. 1993]. Dictionaries are usually very stylized, making it possible to define fairly simple patterns to extract semantic relations from dictionary definitions. For instance, noun definitions usually consist of a *genus* term identifying the kind, followed by *differentiae* that distinguish the noun being defined from its genus, e.g., the definition of "basset hound" in the Webster's dictionary is given as "any of an old French breed of short-legged, slow-moving, hunting dogs with very long ears and crooked front legs." This enables the program to extract A-KIND-OF(basset hound, dog), as well as other semantic relations from the differentiae. NetSerf makes extensive use of semantic relations extracted from an on-line Webster's dictionary using a pattern definition language [Chakravarthy 1994a].

Once the database of semantic relations is available, it can be used to match queries to documents. Traditionally, this has been done through *keyword expansion* techniques, e.g., [Wang et al. 1985, Cohen & Kjeldsen 1987, Rada & Bicknell 1989, Voorhees & Hou 1991]. Expanding a keyword yields new words that are semantically related to it. Keyword expansion is applied either to the query or to the document or both, and the expanded sets are used for matching. However, keyword expansion techniques have not shown significant improvements over other standard techniques, because it is usually very difficult to decide which words to expand [Voorhees 1994], and which semantic relations to apply during the expansion. [Cohen & Kjeldsen 1987] describe an occurence of this problem in a system that was designed to match grant proposals to descriptions of funding agencies. Their system relied on generalization of all keywords (through a hand-coded semantic network) to find relevant matches. But, when the wrong keyword was chosen for expansion, this method yielded poor results. For example, the system matched the query "economic impact of dandelions on landscaping" to the agency description "reproduction in plants," because the keyword "dandelion" was generalized to match "plant" without regard to their semantic contexts. Therefore, as [Voorhees 1994, page 68] puts it, "the challenge now lies in finding an automatic procedure that is able to select appropriate concepts to expand."

Our work is based on the premise that, if a retrieval system deals only with short, structurally predictable descriptions and queries, robust NLP tools can be used to process them into structured representations. These structured representations help the system locate the salient words in the descriptions and queries (and their roles), thereby providing clues for keyword expansion. One such retrieval system is ImEngine [Chakravarthy 1994b], which uses WordNet and dictionary semantic relations to match queries to captions of pictures and video clips. Since captions are usually short, and since they are usually descriptions of actions or situations (i.e., not modal sentences, questions, etc), ImEngine can process them into structured representations. In the next two sections, we will attempt to make the case that structured representations can be obtained for information archives and Internet Hunt queries as well.

3 Representing Information Archives

This section describes how representations of information archives are constructed in NetSerf. The representations are constructed using a text-based editor, and manually disambiguated using WordNet. This section also gives details of a Web site where readers can browse through the list of represented archives and other components of NetSerf.

The representation of an information archive is a list of <relation-type, relation-word> pairs. For each relation-word, NetSerf uses WordNet to identify all of its synsets. The user can disambiguate a relation-word by associating it with a subset of the possible synsets. For example, the World Factbook archive, whose text description is "World facts listed by country," is represented as:

```
TOPIC:      country
            SYNSET: [nation, nationality,
                     land, country, a_people]
            SYNSET: [state, nation, country,
                     land, commonwealth,
                     res_publica,
                     body_politic]
            SYNSET: [country, state, land,
                     nation]
INFO-TYPE: facts
```

Here, the relation-word "country" has been associated with three of its four synsets (the one omitted is [rural_area, country]). Also, the topic of the archive has been separated from the type of information available. This is important because there might be many kinds of information about the same topic, e.g., pictures, text, sound files, etc. about rainforest birds.

In addition to synsets, a relation-word can be associated with other <relation-type, relation-word> pairs, thus creating "parent-child" relationships between pairs. For instance, the archive of "Supreme Court Rulings" is represented as:

OBJECT: ruling
AUTHOR: Supreme Court
SYNSET: [Supreme_Court]

The relation-word "ruling" has not been associated with any synsets since it has only one noun definition in WordNet, [opinion, ruling][1]. We have used a vocabulary of 32 relation-types (including inverses) to construct the archive representations. We started with the vocabulary given in [Fox 1980], but had to add new relation-types in building the representations.

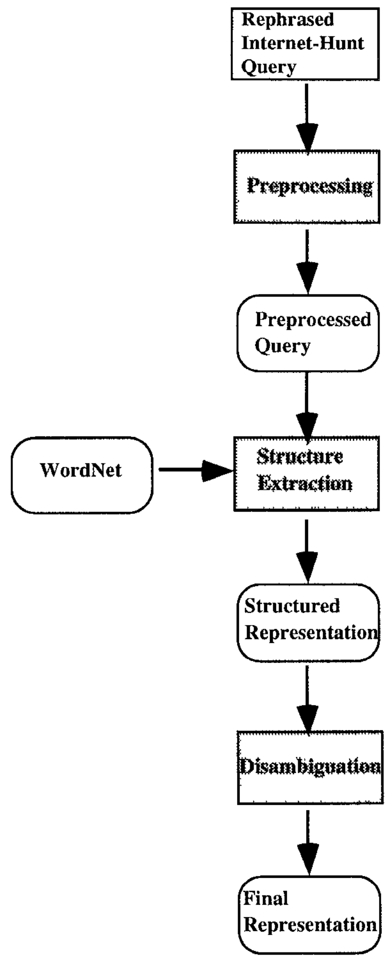
NetSerf's database currently contains representations of 227 Internet archives. Most of these are from two sources, the Whole Internet Catalog [Krol 1992] and the Internet Services List [Yanoff 1993]. In addition, for the purpose of evaluating NetSerf, we added other Internet Hunt archives that were considered to be "correct" answers for the questions used in the experiment. The representations of the archives, the Hunt questions, the correct answers to these questions, and the relation-types used in the representations can all be found at the URL **http://anil.www.media.mit.edu/people/anil/NetSerf/NetSerf.html**.

The need to construct archive representations manually might act as a bottleneck in extending NetSerf. In the future, we will be looking at ways of extracting such representations automatically, either fully or partly, from documents like README files or home pages that typically contain information about the contents of archives. It might be possible to extend the query processor described in the next section to handle this task.

4 Processing Internet Hunt Questions

The query processor makes the assumption that the query, after preprocessing, consists of one or more topic words followed by prepositional phrases and verb clauses that modify either the topic words or preceding modifiers. As in case grammar formalisms [Fillmore 1968], the resulting structured representation assigns roles to various words and word collocations of the query. Figure 2 shows the steps involved in extracting the query representation.

FIGURE 2. Processing an Internet Hunt query



The query processor is not currently capable of handling pronoun resolution or multiple sentences. Therefore, we first manually rephrased Hunt queries that did not fit the processor's format into an equivalent form that it could handle. For instance, we changed the query "A hurricane just blew in! Where can I find satellite photographs of its progress?" to "Satellite photographs of hurricane's progress."

The query is then tagged by the Xerox part-of-speech tagger, which segments the query and assigns a part of speech to each token [Cutting et al. 1992]. A preprocessor then eliminates common query introductions like "Where can I find," "What is" etc. It also extracts

1. "Supreme Court" has two senses, the other one being [supreme_court, state_supreme_court, high_court].

leading information type identifiers like "satellite photographs" (in the query above), or "text" in the query "Text of technology policy proposed by Bill Clinton."

The query processor is then used to locate the topic word(s) and its (their) modifiers. Topic words and modifiers are cast into <relation-type, relation-word> pairs, with the relation-type being based on whether the modifier is a noun modifier or a phrase/clause. For instance, the queries "Satellite photographs of hurricane's progress" and "What is the primary religion in Somalia?" are translated respectively into:

1.	TOPIC:	progress	
		PERTAINS-TO: hurricane	
	INFO-TYPE:	satellite photographs	
2.	TOPIC:	religion	
		IN:	Somalia

The query processor uses WordNet to detect word collocations, e.g., in the query "What is the atomic weight of boron?" the relation-word extracted is "atomic weight," not "weight." Also, if the query does not completely fit the structural patterns expected by the processor, processing continues as far as the structural assumptions allow. For instance, the processor extracts "yen," but not "dollar," as a relation-word from the query "How many yen can I get for a dollar?"

Once the <relation-type, relation-word> pairs are extracted, a word-sense disambiguation program is used to narrow down the set of senses that are associated with a relation-word. This program is described in greater detail in [Chakravarthy 1995]. Here is a brief overview. The disambiguator uses the part of speech assigned to the relation-word by the tagger as the first filter. Then, pairs of neighbouring relation-words are disambiguated using a set of heuristics based on their connecting relationship. To give an example of one heuristic, if two relation-words are joined by an "and" connective, the disambiguator picks those senses that have a common hypernym, e.g., in disambiguating the phrase "slush and snow," the cocaine sense of "snow" is rejected. For a given relation-word, all applicable heuristics are tried, and those senses that are rejected by all heuristics are discarded. The disambiguator uses 44 heuristics based on 12 connecting relationships.

Recent work on the use of disambiguation in IR [Sanderson 1994] suggests that unless disambiguation is very accurate, retrieval performance might be worse than with no disambiguation at all. Therefore, when we evaluate NetSerf in Section 6, we will show results both with and without automatic disambiguation of queries.

The final step of the query processor is to expand the main topic relation-words using semantic relations from the dictionary. Two examples: given the topic word "pub," the pair <PERTAINS-TO "alcoholic beverage"> is generated (from the definition "pub: an establishment where alcoholic beverages are sold or consumed"), and for the topic word "pollution," the pair <HAS-OBJECT "environment"> is generated from the definitions "pollution: the action of polluting" and "pollute: to contaminate (an environment) especially with man-made waste."

5 Matching Query Representations

The two preceding sections showed how representations are constructed for information archives and how structured representations are automatically extracted from queries. In this section, we will describe the successive steps of matching the query representations to the archive representations, and ranking the resulting hits.

The matching step is a straightforward implementation of the generalization principle. A query representation, Q, matches an archive representation, R, if some valid synset of some relation-word in R is a hypernym of some valid synset of some relation-word in Q. We will call each such match a *hypernym-match.* Within R, a valid synset is one that has been explicitly associated to some relation-word. Within Q, if the query has been disambiguated, a valid synset is one explicitly indicated by the disambiguator. If not, any synset of any relation-word in Q is valid.

In the ranking step, a weight is assigned to every hit, i.e., to every $\{Q\ R\}$ match obtained from the previous step. The basic component of this weight is the number of hypernym-matches between relation-words in Q and R. Other components are added or subtracted for every hypernym-match, H, as follows:

- A positive weight is added if the two relation-types of *H* are equivalent, or if there is a hypernym-match between the parents of the two <relation-type, relation-word> pairs of *H,* or if the two relation-words of *H* are both top-level topic words.
- A negative weight is added if the two relation-types of *H* are not equivalent, or if an important child of one pair of *H* does not have a counterpart in the other pair. For instance, consider the example earlier where the query relation-word, "Somalia" was matched to the relation-word "country." A negative weight is added in this case since "Somalia" has the relation-type "IN" which is not equivalent to "TOPIC," the relation-type of the word "country."

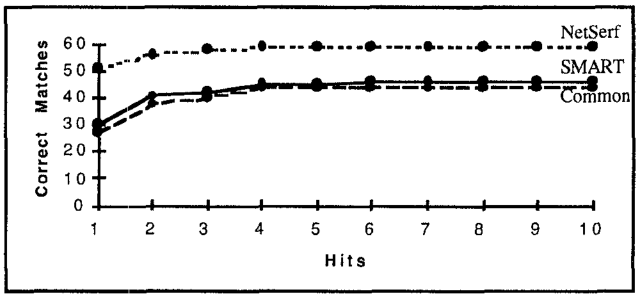
Finally, ties are broken using the average *distance* of all the hypernym-matches between *Q* and *R*. To get the distance between a synset and its hypernym in WordNet, we simply count the number of intervening links between the two.

6 Performance Comparison

This section reports results from the evaluation of NetSerf on a set of 75 questions chosen from the Internet Hunt collection. The questions were selected based on whether the answers suggested by the Hunt followed the principle of generalization. To estimate the effectiveness of NetSerf's techniques, this section also compares the performance of NetSerf to SMART which does not use either structured representations or semantic knowledge. To run SMART on this set of questions, the text descriptions of the sources were gathered to form a single document collection. Lastly, the section looks at the performance of two versions of NetSerf that do not use semantic knowledge and structured representations respectively, in order to judge the individual influence of these components.

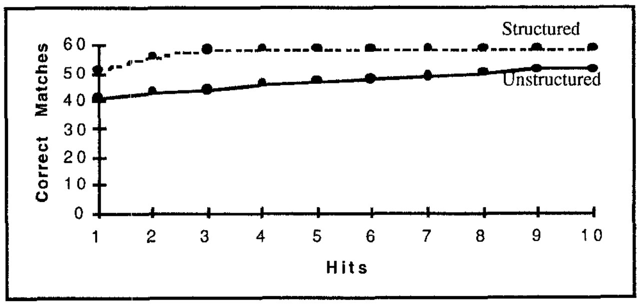
The strategy used to evaluate the two systems is as follows. For each query, the collection provides a set of one or more correct answers (archives). Also, for each query, both systems return a list of ranked matches (archives). As a measure of each system's success rate, we counted the number of queries for which any of the correct answers were found in the first *n* top-ranked hits returned by the system. Figure 3 shows the results for *n* from 1 to 10. For example, when we consider only the top-ranked hit, NetSerf is successful in matching 51 queries (out of 75), while SMART matches 30.

FIGURE 3. Performance of NetSerf and SMART on 75 Internet Hunt queries



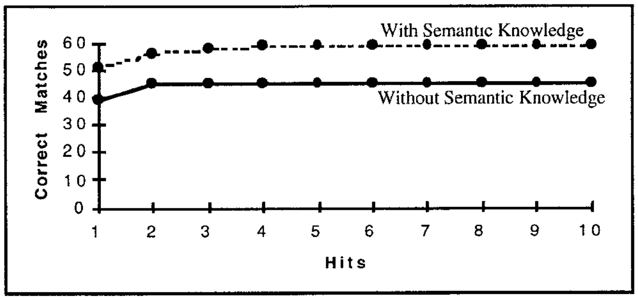
Since NetSerf and SMART use very different retrieval techniques, we were interested in finding out if they were complementary, i.e., successful on different subsets of queries in the collection. The line titled "Common" in Figure 3 shows the number of queries that both systems were successful on. It indicates that NetSerf was successful on almost all the queries on which SMART was successful. The use of different input representations of archives is a significant factor in accounting for the performance differential between NetSerf and SMART.

FIGURE 4. Performance of NetSerf without structured representations of queries



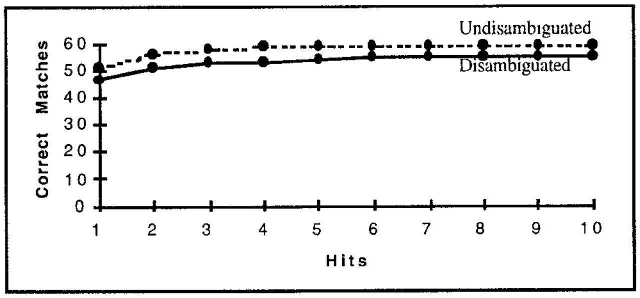
We will now describe the individual influences of the three components of NetSerf: structured representations, use of semantic relations in matching, and disambiguation. To test the effect of structured representations, we built a version of NetSerf which did not use structured representations of the queries. Instead, all the nouns, adjectives and other modifiers in the tagged query were generalized directly to find the hits. The hits were ranked using the method described in Section 5, excluding the formulae that need information about relation-types. Figure 4 presents the results of running this version of NetSerf, showing that the use of structured representations leads to improvements ranging from 15.7% to 24.4%.

FIGURE 5. Performance of NetSerf with/without semantic knowledge



To test the use of semantic knowledge in matching, we ran a version of NetSerf that did not use semantic relations at all in the matching process. Hits were again ranked using the method described in Section 5. As we see from Figure 5, the use of semantic knowledge leads to a clear improvement in NetSerf's performance (between 30.8% and 31.1%).

FIGURE 6. NetSerf with and without disambiguation



Lastly, Figure 6 shows the result of using the disambiguator described in Section 4 (the results shown earlier were derived without using the disambiguator). For all n, the disambiguated version performs slightly worse than the undisambiguated version. Using these two versions, we also found that there were no queries on which only the disambiguated version was successful. At least on this set of queries and archives, mistakes made by the disambiguator seem to drag performance down, while correct disambiguation does not seem to enhance NetSerf's performance.

7 Conclusions

This paper presents NetSerf, a program that finds Internet information archives by generalizing natural language queries. The archives are represented in NetSerf by hand-coded semantic relation structures. The queries are processed by robust NLP tools into structured representations, which are then matched to the archive representations using semantic knowledge from WordNet, a semantic thesaurus, and an on-line Webster's dictionary. NetSerf has been tested on a set of 75 queries selected from Internet Hunt. On this set, NetSerf does better than SMART by 28.3% to 70%, depending on the number of hits considered. The paper also shows that both structured representations and semantic knowledge-based matching lead to significant improvements in NetSerf's performance.

There are many open questions regarding NetSerf. Fundamentally, should we assume that relevant archives need to be located before searching for relevant information items? It would be interesting to explore the idea that all the information items on the Internet could be gathered into a single index, thereby treating the entire Internet as a single gigantic, but "flat," collection [Lewis 1994]. But it should be noted that organizing information into archives (as for WAIS, Gopher, etc.) offers two practical advantages: it enables the search for relevant information to be progressively narrowed, and it makes it easier to get a broader picture of the available information. Therefore, finding relevant archives seems to be a significant first step in finding information on the Internet.

Secondly, to continue with hand-coded archive representations, we have to investigate whether it is reasonable to expect archive providers to create these representations. Alternatively, we have to examine how well NetSerf would work if archive representations are extracted automatically (as discussed in Section 3). The query processor also needs significant extensions to handle more "natural" queries.

The matching process sometimes needs semantic information that cannot be found either in WordNet or in the dictionary. For example, the answer to the query "What is the text of the First Amendment to the Constitution of the United States?" is the archive "Historical Documents," an inference not possible through our semantic relations database. Further, we feel that the matching process needs something analogous to term weighting, which would make it possible to rate some inferences as more valuable than others. The question of what and how much semantic knowledge to use is still mostly unresolved.

References

Amsler, R. 1980. *The Structure of the Merriam Webster Pocket Dictionary*, PhD Dissertation, University of Texas, Austin.

Chakravarthy, A. S. 1994a. "Representing Information Need with Semantic Relations," in *Proceedings of COLING-94*, Kyoto, Japan.

Chakravarthy, A. S. 1994b."Toward Semantic Retrieval of Picture and Video Clips," in *Proc. of RIAO'94*.

Chakravarthy, A. S. 1995. "Sense Disambiguation Using Semantic Relations and Adjacency Information," in preparation for submission to *ACL-95* student session.

Chodorow, M. S., Byrd, R. J., and Heidorn, G. E. 1985. "Extracting Semantic Hierarchies from a Large On-Line Dictionary," in *Proceedings of the 23rd ACL*.

Cohen, P. R., and Kjeldsen, R. 1987. "Information Retrieval by Constrained Spreading Activation in Semantic Networks," in *Information Processing and Management*, 23(4), 255-268.

Cutting, D., Kupiec, J., Pedersen, J. and Sibun, P. 1992. "A Practical Part-of-Speech Tagger," in *Proceedings of the 3rd Conf. on Applied NLP*.

Dolan, W.B., Vanderwende, L., and Richardson, S. D. 1993. "Automatically Deriving Structured Knowledge Bases from On-line Dictionaries," in *Proceedings of the First Conference of the Pacific Association for Computational Linguistics*, Vancouver.

Fillmore, C. J. 1968. "The Case for Case," in *Universals of Linguistic Theory*, edited by E. Bach and R. T. Harms, Holt, New York.

Fox, E. A. 1980. "Lexical Relations: Enhancing Effectiveness of Information Retrieval Systems," in *SIGIR Newsletter*, 15(3).

Fox, E. A., Nutter, J. T., Ahlswede, T., Evens, M., and Markowitz, J. 1988. "Building a Large Thesaurus for Information Retrieval," in *Proceedings of the Second Conf. on Applied NLP*.

Gates R. 1992. *Internet Hunt*. Monthly Internet Manuscripts from ftp.cic.net; directory pub/internet-hunt

Krol, E. 1992. *The Whole Internet: User's Guide and Catalog*, O'Reilly and Associates, Sebastopol, CA.

Lenat, D. B., and Guha, R. V. 1990. *Building Large Knowledge-based Systems*, Addison-Wesley.

Lewis, D. D. 1994. Personal communication.

Miller, G. A. 1990. "WordNet: An On-line Lexical Database," *International Journal of Lexicography*, 3(4).

Minsky, M. L. 1974. *A Framework for Representing Knowledge*, AI Memo 306, MIT AI Laboratory.

Oracle 1993. *ConText®: Introduction to Oracle ConText*, ORACLE White Paper, September, 1993.

Quillian, M. R. 1968. "Semantic Memory," in *Semantic Information Processing*, Minsky, ed., MIT Press.

Rada, R. and Bicknell, E. 1989. "Ranking Documents Based on a Thesaurus," in *Journal of the American Society for Information Science*, 40(5), 304-310.

Salton, G. 1989. *Automatic Text Processing*, Addison-Wesley Publishing Co., Reading, MA.

Sanderson, M. 1994. "Word Sense Disambiguation and Information Retrieval," in *Proceedings of SIGIR'94*.

Schwartz, M. F., Emtage, A., Kahle, B., and Neuman, B. C. 1992. "A Comparison of Internet Resource Discovery Approaches," *Computing Systems*, 5(4).

Voorhees, E. M. and Hou, Y-W. 1991. "Vector Expansion in a Large Collection," in *Proceedings of the First Text Retrieval Conference (TREC-1)*.

Voorhees, E. M. 1994. "Query Expansion Using Lexical-Semantic Relations," in *Proceedings of SIGIR'94*.

Wang, Y-C., Vandenthorpe, J., and Evens, M. 1985. "Relational Thesauri in Information Retrieval," *Journal of the American Society for Information Science*, 36(1), 15-27.

Yanoff, S. 1993. *Internet Services List*, Internet Manuscript 'finger yanoff@csd4.csd.uwm.edu'