A LAYERED APPROACH TO NLP-BASED INFORMATION RETRIEVAL

Sharon Flank
SRA International
4300 Fair Lakes Court
Fairfax, VA 22033, USA
flanks@sra.com

Abstract

A layered approach to information retrieval permits the inclusion of multiple search engines as well as multiple databases, with a natural language layer to convert English queries for use by the various search engines. The NLP layer incorporates morphological analysis, noun phrase syntax, and semantic expansion based on WordNet.

1 Introduction

This paper describes a layered approach to information retrieval, and the natural language component that is a major element in that approach. The layered approach, packaged as Intermezzo^{TM}, was deployed in a pre-product form at a government site. The NLP component has been installed, with a proprietary IR engine, PhotoFile, (Flank, Martin, Balogh and Rothey, 1995), (Flank, Garfield, and Norkin, 1995), at several commercial sites, including Picture Network International (PNI), Simon and Schuster, and John Deere.

Intermezzo employs an abstraction layer to permit simultaneous querying of multiple databases. A user enters a query into a client, and the query is then passed to the server. The abstraction layer, part of the server, converts the query to the appropriate format for each of the databases (e.g. Fulcrum^{TM}, RetrievalWare^{TM}, Topic^{TM}, WAIS). In Boolean mode, queries are translated, using an SGML-based intermediate query language, into the appropriate form; in NLP mode the queries undergo morphological analysis, NP syntax, and semantic expansion before being converted for use by the databases.

The following example illustrates how a user's query is translated.

Unexpanded query natural disasters in New England

Search-engine specific natural AND disaster(s) AND New AND England

Semantic expansion ((natural and disaster(s)) or hurricane(s) or earthquake(s) or tornado(es) in ("New England" or Maine or Vermont or "New Hampshire" or "Rhode Island" or Connecticut or Massachusetts)

The NLP component has been deployed with as many as 500,000 images, at Picture Network International (PNI). The original commercial use of PNI was as a dialup system, launched with approximately 100,000 images. PNI now operates on the World Wide Web (www.publishersdepot.com). Adjustment of the NLP component continued actively up through about 250,000 images, including additions to the semantic net and tuning of the parameters for weighting. Retrieval speed for the NLP component averages under a second. Semantic expansion is performed in advance on the caption database, not at runtime; runtime expansion makes operation too slow.

The remainder of this paper describes how the NLP mode works, and what was required to create it.

2 The NLP Techniques

The natural language processing techniques used in this system are well known, including in information retrieval applications (Strzalkowski, 1993), (Strzalkowski, Perez Carballo and Marinescu, 1995), (Evans and Zhai, 1996). The importance of this work lies in the scale and robustness of the techniques as combined into a system for querying large databases.

The NLP component is also layered, in effect. It uses a conventional search algorithm (several were tested, and the architecture supports plug-and-play here). User queries undergo several types of NLP processing, detailed below, and each element in the processing contributes new query components (e.g. synonyms) and/or weights. The resulting query, as in the example above, *natural disasters in New England*, contains expanded terms and weighting information that can be passed to any search engine. Thus the Intermezzo multisearch layer can be seen

as a natural extension of the layered design of the NLP search system.

When texts (or captioned images) are loaded into the database, each word is looked up, words that may be related in the semantic net are found based on stored links, and the looked-up word, along with any related words, are all displayed as the "expansion" of that word. Then a check is made to determine whether the current word or phrase corresponds to a proper name, a location, or something else. If it corresponds to a name, a name expansion process is invoked that displays the name and related names such as nicknames and other variants, based on a linked name file. If the current word or phrase corresponds to a location, a location expansion process is invoked that, accessing a gazetteer, displays the location and related locations, such as *Arlington, Virginia* and *Arlington, Massachusetts* for *Arlington*, based on linked location information in the gazetteer and supporting files. If the current word or phrase is neither a name nor a location, it is expanded using the semantic net links and weights associated with those links. Strongly related concepts are given high weights, while more remotely related concepts receive lower weights, making them less exact matches. Thus, for a query on *car*, texts or captions containing *car* and *automobile* are listed highest, followed by those with *sedan*, *coupe*, and *convertible*, and then by more remotely related concepts such as *transmission*, *hood*, and *trunk*.

Once the appropriate expansion is complete, the current word or phrase is stored in an index database, available for use in searching as described below. Processing then returns to the next word or phrase in the text.

Once a user query is received, it is tokenized so that it is divided into individual tokens, which may be single words or multiwords. For this process, a variation of conventional pattern matching is used. If a single word is recognized as matching a word that is part of a stored multiword, a decision on whether to treat the single word as part of a multiword is made based on the contents of the stored pattern and the input pattern. Stored patterns include not just literal words, but also syntactic categories (e.g. adjective, non-verb), semantic categories (e.g. nationality, government entity), or exact matches. If the input matches the stored pattern information, then it is interpreted as a multiword rather than independent words.

A part-of-speech tagger then makes use of linguistic and statistical information to tag the parts of speech of incoming query portions. Only words that match by part of speech are considered to match, and if two or more parts of speech are possible for a particular word, it is tagged with both. After tagging, word affixes (i.e. suffixes) are stripped from query words to obtain a word root, using conventional inflectional morphology. If a word in a query is not known, affixes are stripped from the word one by one until a known word is found. Derivational morphology is not currently implemented.

Processing then checks to determine whether the resulting word is a function word (closed-class) or content word (open-class). Function words are ignored. [1] For content words, the related concepts for each sense of the word are retrieved from the semantic net. If the root word is unknown, the word is treated as a keyword, requiring an exact match. Multiwords are matched as a whole unit, and names and locations are identified and looked up in the separate name and location files. Next, noun phrases and other syntactic units are identified.

An intermediate query is then formulated to match against the index database. Texts or captions that match queries are then returned, ranked, and displayed to the user, with those that match best being displayed at the top of the list. In the current system, the searching is implemented by first building a B-tree of *ID lists*, one for each concept in the text database. The ID lists have an entry for each object whose text contains a reference to a given concept. An entry consists of an object ID and a weight. The object ID provides a unique identifier and is a positive integer assigned when the object is indexed. The weight reflects the relevance of the concept to the object's text, and is a positive integer.

To add an object to an existing index, the object ID and a weight are inserted into the ID list of every concept that is in any way relevant to the text. For searching, the ID lists of every concept in the query are retrieved and combined as specified by the query. Since ID lists contain IDs with weights in sorted order, determining existence and relevance of a match is simultaneous and fast, using only a small number of processor instructions per concept-object pair.

The following sections treat the NLP issues in more detail.

2.1 Semantic Expansion, Part-of-Speech Tagging, and WordNet

Semantic expansion, based on WordNet 1.4 (Miller et al., 1994), makes it possible to retrieve words by synonyms, hypernyms, and other relations, not simply by exact matches. The expansion must be constrained, or precision will suffer drastically. The first constraint is part of speech: retrieve only those expansions that apply to the correct part of speech in context. A Church-style tagger (Church, 1988)

[1] In a few cases, the loss of prepositions presents a problem. In practice, the problem is largely restricted to pictures showing unexpected relationships, e.g. a package *under* a table. Treating prepositions just like content works leads to odd partial matches (things under tables before other pictures of packages and tables, for example). The solution will involve an intermediate treatment of prepositions.

marks parts of speech. Sense tagging is a further refinement: the algorithm first distinguishes between, e.g. *crane* as a noun versus *crane* as a verb. Once noun has been selected, further ambiguity still remains, since a crane can be either a bird or a piece of construction equipment. This additional disambiguation can be ignored, or it can be performed manually (impractical for large volumes of text and impractical for queries, at least for most users). It can also be performed automatically, based on a sense-tagged corpus.

The semantic net used in this application incorporates information from a variety of sources besides WordNet; to some extent it was hand-tailored. Senses were ordered according to their frequency of occurrence in the first 150,000 texts used for retrieval, in this case photo captions consisting of one to three sentences each. WordNet 1.5 and subsequent releases have the senses ordered by frequency, so this step would not be necessary now.

The top level of the semantic net splits into events and entities, as is standard for knowledge bases supporting natural language applications. There are approximately 100,000 entries, with several links for each entry. The semantic net supplies information about synonymy and hierarchical relations, as well as more sophisticated links, like part-of. The closest synonyms, like *dangerous* and *perilous*, are ranked most highly, while subordinate types, like *skating* and *rollerblading*, are next. More distant links, like the relation between *shake hands* and *handshake*, links between adjectives and nouns, e.g. *dangerous* and *danger*, and part-of links, e.g. *brake* and *brake shoe*, contribute lesser amounts to the rank and therefore yield a lower overall ranking. Each returned image has an associated weight, with 100 being a perfect match. Exact matches (disregarding inflectional morphology) rank 100. The system may be configured so that it does not return matches ranked below a certain threshold, say 50.

Table 1 presents the weights currently in use for the various relations in WordNet. The *depth* figure indicates how many levels a particular relation is followed. Some relations, like hypernyms and pertainyms, are clearly relevant for retrieval, while others, such as antonyms, are irrelevant. If the depth is zero, as with antonyms, the relation is not followed at all: it is not useful to include antonyms in the semantic expansion of a term. If the depth is nonzero, as with hypernyms, its relative weight is given in the weight figure. Hypernyms make sense for retrieval (*animals* retrieves *hippos*) but hyponyms do not (*hippos* should not retrieve *animals*). The weight indicates the degree to which each succeeding level is discounted. Thus a ladybug is rated 90% on a query for *beetle*, but only 81% (90% x 90%) on a query for *insect*, 73% (90% x 81%) on a query for *arthropod*, 66% (90% x 73%) on a query for *invertebrate*, 59% (90% x 66%) on a query for *animal*, and not at all (more than four levels) on a query for *organism*. A query for organisms returns images that match the request more closely, for example:

Table 1: Expansion depth for WordNet relations

Relation	*Part of Speech*	*Depth*	*Weight*
ANTONYM	noun	0	
ANTONYM	verb	0	
ANTONYM	adj	0	
ANTONYM	adv	0	
HYPERNYM	noun	4	90
HYPERNYM	verb	4	90
HYPONYM	noun	0	
HYPONYM	verb	0	
MEM MERONYM	noun	3	90
SUB MERONYM	noun	0	
PART MERONYM	noun	3	90
MEM HOLONYM	noun	0	
SUB HOLONYM	noun	0	
PART HOLONYM	noun	0	
ENTAILMENT	verb	2	90
CAUSE	verb	2	90
ALSO SEE	verb	1	90
ALSO SEE	adj	1	90
ALSO SEE	adv	1	90
ALSO SEE	noun	1	90
SIMILAR TO	adj	2	90
PERTAINYM	adj	2	95
PERTAINYM	noun	2	95
ATTRIBUTE	noun	0	
ATTRIBUTE	adj	1	80

- An amorphous amoeba speckled with greenish-yellow blobs.

It might appear that ladybugs *should* be retrieved in queries for *organism*, but in fact such high-level queries generate thousands of hits even with only four-level expansion. In practical terms, then, the number of levels must be limited. Excalibur's WordNet-based retrieval product, RetrievalWare, does not limit expansion levels, instead allowing the expert user to eliminate particular senses of words at query time, in recognition of the need to limit term expansion in one aspect of the system if not in another. The depth and weight figures were tuned by trial and error on a corpus of several hundred thousand paragraph-length picture captions. For longer texts, the depth, particularly for hypernyms, should be less.

The weights file does not affect which images are selected as relevant, but it does affect their relevance ranking, and thus the ordering that the user sees. In practical terms this means that for a query on *animal*, exact matches on *animal* appear first, and *hippos* appear before *ladybugs*. Of course, if the threshold is set at 50 and the weights alter a ranking from

51 to 49, the user will no longer see that image in the list at all. Technically, however, the image has not been removed from the relevance list, but rather simply downgraded.

WordNet was designed as a multifunction natural language resource, not as an IR expansion net. Inevitably, certain changes were required to tailor it for NLP-based IR. First, there were a few links high in the hierarchy that caused bizarre behavior, like animals being retrieved for queries including *man* or *men*. Other problems were some "unusual" correlations, such as:

- *grimace* linked to *smile*
- *juicy* linked to *sexy*

Second, certain slang entries were inappropriate for a commercial system and had to be removed in order to avoid giving offense. Single sense words (e.g. *crap*) were not particularly problematic, since users who employed them in a query presumably did so on purpose. Polysemous terms such as *nuts*, *skirt*, and *frog*, however, were eliminated, since they could inadvertently cause offense.

Third, there were low-level edits of single words. Before the senses were reordered by frequency, some senses were disabled in response to user feedback. These senses caused retrieval behavior that users found inexplicable. For example, the *battle* sense of *engagement*, the *fervor* sense of *fire*, and the *Indian language* sense of *Massachusetts*, all were removed, because they retrieved images that users could not link to the query. Although users were forgiving when they could understand why a bad match had occurred, they were far less patient with what they viewed as random behavior. In this case, the rarity of the senses made it difficult for users to trace the logic at work in the sense expansion.

Finally, since language evolves so quickly, new terms had to be added, e.g. *rollerblade*. This task was the most common and the one requiring the least expertise. Neologisms and missing terms numbered in the dozens for 500,000 sentences, a testament to WordNet's coverage.

2.2 Gazetteer Integration

Locations are processed using a gazetteer and several related files. The gazetteer (supplied by the U.S. Government for the Message Understanding Conferences [MUC]), is extremely large and comprehensive. In some ways, it is almost too large to be useful. Algorithms had to be added, for example, to select which of many choices made the most sense. *Moscow* is a town in Idaho, but the more relevant city is certainly the one in Russia. The gazetteer contains information on administrative units as well as rough data on city size, which we used to develop a sense-preference algorithm. The largest administrative unit (country, then province, then city) is always given a higher weight, so that *New York* is first interpreted as a state and then as a city. Within the city size rankings, the larger cities are weighted higher. Of course explicit designations are understood more precisely, i.e. *New York State* and *New York City* are unambiguous references only to the state and only to the city, respectively. And *Moscow, Idaho* clearly does not refer to any Moscow outside of Idaho. Furthermore, since this was a U.S. product, U.S. states were weighted higher than other locations, e.g. *Georgia* was first understood as a state, then as a country.

At the most basic level, the gazetteer is a hierarchy. It permits subunits to be retrieved, e.g. *Los Angeles* and *San Francisco* for a query *California*. An alias table converted the various state abbreviations and other variant forms, e.g.

Washington D.C.; Washington, DC; Washington, District of Columbia; Washington DC; Washington, D.C.; DC; and D.C.

Some superunits were added, e.g. *Eastern Europe*, *New England*, and equivalences based on changing political situations, e.g. *Moldavia*, *Moldova*. To handle queries like *northern Montana*, initial steps were taken to include latitude and longitude information. The algorithm, never implemented, was to take the northernmost 50% of the unit. So if Montana covers X to Y north latitude, northern Montana would be between (X+Y)/2 and Y.

Additional locations are matched on the fly by patterns and then treated as units for purposes of retrieval. For example, *Glacier National Park* or *Mount Hood* should be treated as phrases. To accomplish this, a pattern matcher, based on finite state automata, operates on simple patterns such as:

(LOCATION —
(& (* {word "[A-Z][a-z]*"}) {word "[Nn]ational"} {OR {word "[Pp]ark"} {word "[Ff]orest"}})

2.3 Syntactic and Other Patterns

The pattern matcher also performs noun phrase (NP) identification, using the following patterns for core NPs:

(& {tag deter} [MODIFIER (& (? (& {tag adj} {tag conj})) (* — {tag noun} {tag adj} {tag number} {tag listmark}))] [HEAD_NOUN {tag noun}])

Identification of core NPs (i.e. modifier-head groupings, without any trailing prepositional phrases or other modifiers) makes it possible to distinguish *stock cars* from *car stocks*, and, for a query on *little girl in a red shirt*, to retrieve girls in red shirts in preference to a girl in a blue shirt and red hat.

Examples of images returned for the *little girl in a red shirt* query, rated at 92%, include:

- Two smiling young girls wearing matching jean overalls, red shirts. The older girl wearing a

blue baseball cap sideways has blond pigtails with yellow ribbons. The younger girl wears a yellow baseball cap sideways.

- An African American little girl wearing a red shirt, jeans, colorful hairband, ties her shoelaces while sitting on a patterned rug on the floor.
- A young girl in a bright red shirt reads a book while sitting in a chair with her legs folded. The hedges of a garden surround the girl while a woods thick with green leaves lies nearby.
- A young Hispanic girl in a red shirt smiles to reveal braces on her teeth.

The following image appears with a lower rating, 90%, because the red shirt is later in the sentence. The noun phrase ratings do not play a role here, since red does modify shirt in this case; the ratings apply only to core noun phrases, not prepositional modifiers.

- A young girl in a blue shirt presents a gift to her father. The father wears a red shirt.

Images with girls in non-red shirts appear with even lower ratings if no red shirt is mentioned at all. This image was ranked at 88%.

- A laughing little girl wearing a straw hat with a red flower, a purple shirt, blue jean overalls.

Of course, in a fully NLP-based IR system, neither of these examples would match at all. But full NLP is too slow for this application, and partial matches do seem to be useful to its users, i.e. do seem to lead to licensing of photos.

Using the output of the part-of-speech tagger, the patterns yield weights that prefer syntactically similar matches over scrambled or partial matches. The weights file for NPs contains three multipliers that can be set:

scale noun 200 This sets the relative weight of the head noun itself to 200%.

scale modifier 50 This sets the relative importance of each modifier to half of what it would be otherwise.

scale phrase 200 This sets the relative weight of the entire noun phrase, compared to the old ranking values. This effect multiplies the noun and modifier effects, i.e. it is cumulative.

2.4 Name Recognition

Patterns are also the basis for the name recognition module, supporting recognition of the names of persons and organizations. Elements marked as names are then marked with a preference that they be retrieved as a unit, and the names are expanded to match related forms. Thus *Bob Dole* does not match *Bob Packwood worked with Dole Pineapple* at 100%, but it does match *Senator Robert Dole*.

The name recognition patterns employ a large file of name variants, set up as a simple alias table: the nicknames and variants of each name appear on a single line in the file. The name variants were derived manually from standard sources, including baby-naming books.

3 Interactions

In developing the system, interactions between subsystems posed particular challenges. In general, the problems arose from conflicts in data files. In keeping with the layered approach and with good software engineering in general, the system is maximally modular and data-driven. Several of the modules utilize the same types of information, and inconsistencies caused conflicts in several areas. The part-of-speech tagger, morphological analyzer, tokenizer, gazetteer, semantic net, stop-word list, and Boolean logic all had to be made to cooperate. This section describes several problems in interaction and how they were addressed. In most cases, the solution was tighter data integration, i.e. having the conflicting subsystems access a single shared data file. Other cases were addressed by loosening restrictions, providing a backup in case of inexact data coordination.

The morphological analyzer sometimes stemmed differently from WordNet, complicating synonym lookup. The problem was solved by using WordNet's morphology instead. In both cases, morphological variants are created in advance and stored, so that stemming is a lookup rather than a run-time process. Switching to WordNet's morphology was therefore quite simple. However, some issues remain. For example, *pies* lists the three senses of *pi* first, before the far more likely *pie*.

The database on which the part-of-speech tagger trained was a collection of *Wall Street Journal* articles. This presented a problem, since the domain was specialized. In any event, since the training data set was not WordNet, they did not always agree. This was sorted out by performing searches independent of part of speech if no match was found for the initial part of speech choice. That is, if the tagger marked *short* as a verb only (as in *to short a stock*), and WordNet did not find a verb sense, the search was broadened to allow any part of speech in WordNet.

Apostrophes in possessives are tokenized as separate words, turning *Alzheimer's* into *Alzheimer 's* and *Nicole's* into *Nicole 's*. In the former case, the full form is in WordNet and therefore should be taken as a unit; in the latter case, it should not. The fix here was to look up both, preferring the full form.

For pluralia tantum words (*shorts, fatigues, doubles, AIDS, twenties*), stripping the affix -s and then looking up the root word gives incorrect results. Instead, when the word is plural, the pluralia tantum, if there is one, is preferred; when it is singular, that

Table 2: Conversions from English to Boolean

English	*Boolean*
and	and
or	or
with	and
not	not
but	and
without	not
except	not
nor	not

meaning is ruled out.

WordNet contains some location information, but it is not nearly as complete as a gazetteer. Some locations, such as major cities, appear in both the gazetteer and in WordNet, and, particularly when there are multiple "senses" (*New York* state and city, *Springfield*), must be reconciled. We used the gazetteer for all location expansions, and recast it so that it was in effect a branch of the WordNet semantic net, i.e. hierarchically organized and attached at the appropriate WordNet node. This recasting enabled us to take advantage of WordNet's generic terms, so that *city lights*, for example, would match *lights on a Philadelphia street*. It also preserved the various gazetteer enhancements, such as the sense preference algorithm, superunits, and equivalences.

Boolean operators appear covertly as English words. Many IR systems ignore them, but that yields counterintuitive results. Instead of treating operators as stop words and discarding them, we instead perform special handling on the standard set of Boolean operators, as well as an expandable set of synonyms. For example, given *insects except ants*, many IR systems simply discard *except*, turning the query, incorrectly, into *insects and ants*, retrieving exactly the items the user does *not* want. To avoid this problem, we convert the terms in Table 2 into Boolean operators.

4 Evaluation

Evaluation has two primary goals in commercial work. First, is the software robust enough and accurate enough to satisfy paying customers? Second, is a proposed change or new feature an improvement or a step backward?

Customers are more concerned with precision, because they do not like to see matches they cannot explain. Precision above about 80% eliminated the majority of customer complaints about accuracy. Oddly enough, they are quite willing to make excuses for bad system behavior, explaining away implausible matches, once they have been convinced of the system's basic accuracy. The customers rarely test recall, since it is rare either for them to know which pictures are available or to enter successive related queries and compare the match sets. Complaints about recall in the initial stages of system development came from suppliers, who wanted to ensure their own pictures could be retrieved reliably.

To test recall as well as precision in a controlled environment, in the early phase of development, a test set of 1200 images was created, and manually matched, by a photo researcher, against queries submitted by other photo researchers. The process was time-consuming and frustratingly imprecise: it was difficult to score, since matches can be partial, and it was hard to determine how much credit to assign for, say, a 70% match that seemed more like a 90% match to the human researcher. Precision tests on the live (500,000-image) PNI system were much easier to evaluate, since the system was more likely to have the images requested. For example, while a database containing no little girls in red shirts will offer up girls with any kind of shirt and anything red, a comprehensive database will bury those imperfect matches beneath the more highly ranked, more accurate matches. Ultimately, precision was tested on 50 queries on the full system; any bad match, or partial match if ranked above a more complete match, was counted as a miss, and only the top 20 images were rated. Recall was tested on a 50-image subset created by limiting such non-NLP criteria as image orientation and photographer. Precision was 89.6% and recall was 92%.

In addition, precision was tested by comparing query results for each new feature added (e.g. "Does noun phrase syntax do us any good? What rankings work best?"). It was also tested by series of related queries, to test, for example, whether *penguins swimming* retrieved the same images as *swimming penguins*. Recall was tested by more related queries and for each new feature, and, more formally, in comparison to keyword searches and to Excalibur's RetrievalWare. Major testing occurred when the database contained 30,000 images, and again at 150,000. At 150,000, one major result was that WordNet senses were rearranged so that they were in frequency order based on the senses hand-tagged by captioners for the initial 150,000 images.

In one of our retrieval tests, the combination of noun phrase syntax and name recognition improved recall by 18% at a fixed precision point. While we have not yet attempted to test the two capabilities separately, it does appear that name recognition played a larger role in the improvement than did noun phrase syntax. This is in accord with previous literature on the contributions of noun phrase syntax (Lewis, 1992), (Lewis and Croft, 1990).

4.1 Does Manual Sense-Tagging Improve Precision?

Preliminary experiments were performed on two subcorpora, one with WordNet senses manually tagged, and the other completely untagged. The

corpora are not strictly comparable: since the photos are different, the correct answers are different in each case. Nonetheless, since each corpus includes over 20,000 pictures, there should be enough data to provide interesting comparisons, even at this preliminary stage. Certain other measures have been taken to ensure that the test is as useful as possible within the constraints given; these are described below. Results are consistent with those shown in Voorhees (1994).

Only precision is measured here, since the principal effect of tagging is on precision: untagged irrelevant captions are likely to show up in the results, but lack of tagging will not cause correct matches to be missed. Only crossing matches are scored as bad. That is, if Match 7 is incorrect, but Match 8, 9 and 10 are correct, then the score is 90% precision. If, on the other hand, Match 7 is incorrect and Matches 8, 9 and 10 are also incorrect, there is no precision penalty, since we want and expect partial matches to follow the good matches.

Only the top ten matches are scored. There are three reasons for this: first, scoring hundreds or thousands of matches is impractical. Second, in actual usage, no one will care if Match 322 is better than Match 321, whereas incongruities in the top ten will matter very much. Third, since the threshold is set at 50%, some of the matches are by definition only "half right." Raising the threshold would increase perceived precision but provide less insight about system performance.

Eleven queries scored better in the sense-tagged corpus, while only two scored better in the untagged corpus. The remainder scored the same in both corpora. In terms of precision, the sense-tagged corpus scored 99% while the untagged corpus scored 89% (both figures are artificially inflated, but in parallel, since only crossing matches are scored as bad).

5 Future Directions

Future work will concentrate on speed and space optimizations, and determining how subcomponents of this NLP capability can be incorporated into existing IR packages. This fine-grained NLP-based IR can also answer questions such as who, when, and where, so that the items retrieved can be more specifically targeted to user needs. The next step for caption-based systems will be to incorporate automatic disambiguation, so that captioners will not need to select a WordNet sense for each ambiguous word. In this auto-disambiguation investigation, it will be interesting to determine whether a specialized corpus, e.g. of photo captions, performs sense-tagging significantly better than a general-purpose corpus, such as the Brown corpus (Francis and Kučera, 1979).

References

Church, K. W. 1988. Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, Austin, TX, 1988.

Evans, D. and C. Zhai 1996. Noun-Phrase Analysis in Unrestricted Text for Information Retrieval. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)*, Santa Cruz, CA, 24-27 June 1996, pp.17-24.

Flank, S., P. Martin, A. Balogh and J. Rothey 1995. PhotoFile: A Digital Library for Image Retrieval. In *Proceedings of the International Conference on Multimedia Computing and Systems (IEEE)*, Washington, DC, 15-18 May 1995, pp. 292-295.

Flank, S., D. Garfield, and D. Norkin 1995. Digital Image Libraries: An Innovative Method for Storage, Retrieval, and Selling of Color Images. In *Proceedings of the First International Symposium on Voice, Video, and Data Communications of the Society of Photo-Optical Instrumentation Engineers (SPIE)*, Philadelphia, PA, 23-26 October 1995.

Francis, W. N. and H. Kučera 1979. *Manual of Information to Accompany a Standard Corpus of Present-Day Edited American English, for use with Digital Computers (Corrected and Revised Edition)*, Department of Linguistics, Brown University, Providence, RI.

Lewis, D. D. 1992. An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task. In *Proceedings of ACM SIGIR*, 1992, pp. 37-50.

Lewis, D. D. and W. B. Croft 1990. Term Clustering of Syntactic Phrases. In *Proceedings of ACM SIGIR*, 1990, pp. 385-404.

Miller, G., M. Chodorow, S. Landes, C. Leacock and R. Thomas 1994. Using a semantic concordance for sense identification. In *ARPA Workshop of Human Language Technology*, Plainsboro, NJ, March 1994, pp. 240-243.

Strzalkowski, T. 1993. Natural Language Processing in Large-Scale Text Retrieval Tasks. In *First Text Retrieval Conference (TREC-1)*, National Institute of Standards and Technology, March 1993, pp. 173-187.

Strzalkowski, T., J. Perez Carballo and M. Marinescu 1995. Natural Language Information Retrieval: TREC-3 Report. In *Third Text Retrieval Conference (TREC-3)*, National Institute of Standards and Technology, March 1995.

Voorhees, E. 1994. Query Expansion Using Lexical-Semantic Relations. In *Proceedings of ACM SIGIR* 1994, pp. 61-69.