NATURAL LANGUAGE INFORMATION RETRIEVAL: TIPSTER-2 FINAL REPORT

Tomek Strzalkowski
GE Corporate Research & Development
Schenectady, NY 12301
strzalkowski@crd.ge.com

ABSTRACT

We report on the joint GE/NYU natural language information retrieval project as related to the Tipster Phase 2 research conducted initially at NYU and subsequently at GE R&D Center and NYU. The evaluation results discussed here were obtained in connection with the 3rd and 4th Text Retrieval Conferences (TREC-3 and TREC-4). The main thrust of this project is to use natural language processing techniques to enhance the effectiveness of full-text document retrieval. During the course of the four TREC conferences, we have built a prototype IR system designed around a statistical full-text indexing and search backbone provided by the NIST's Prise engine. The original Prise has been modified to allow handling of multi-word phrases, differential term weighting schemes, automatic query expansion, index partitioning and rank merging, as well as dealing with complex documents. Natural language processing is used to preprocess the documents in order to extract content-carrying terms, discover inter-term dependencies and build a conceptual hierarchy specific to the database domain, and process user's natural language requests into effective search queries.

The overall architecture of the system is essentially the same for both years, as our efforts were directed at optimizing the performance of all components. A notable exception is the new massive query expansion module used in routing experiments, which replaces a prototype extension used in the TREC-3 system. On the other hand, it has to be noted that the character and the level of difficulty of TREC queries has changed quite significantly since the last year evaluation. TREC-4 new ad-hoc queries are far shorter, less focused, and they have a flavor of information requests ("What is the prognosis of ...") rather than search directives typical for earlier TRECs ("The relevant document will contain ..."). This makes building of good search queries a more sensitive task than before. We thus decided to introduce only minimum number of changes to our indexing and search processes, and even roll back some of the TREC-3 extensions which dealt with longer and somewhat redundant queries.

Overall, our system performed quite well as our position with respect to the best systems improved steadily since the beginning of TREC. We participated in both main evaluation categories: category A ad-hoc and routing, working with approx. 3.3 GBytes of text. We submitted 4 official runs in automatic adhoc, manual ad-hoc, and automatic routing (2), and were ranked 6 or 7 in each category (out of 38 participating teams). It should be noted that the most significant gain in performance seems to have occurred in precision near the top of the ranking, at 5, 10, 15 and 20 documents. Indeed, our unofficial manual runs performed after TREC-4 conference show superior results in these categories, topping by a large margin the best manual scores by any system in the official evaluation.

In general, we can note substantial improvement in performance when phrasal terms are used, especially in ad-hoc runs. Looking back at TREC-2 and TREC-3 one may observe that these improvements appear to be tied to the length and specificity of the query: the longer the query, the more improvement from linguistic processes. This can be seen comparing the improvement over baseline for automatic adhoc runs (very short queries), for manual runs (longer queries), and for semi-interactive runs (yet longer queries). In addition, our TREC-3 results (with long and detailed queries) showed 20-25% improvement in precision attributed to NLP, as compared to 10-16% in TREC-4.

OVERVIEW

A typical (full-text) information retrieval (IR) task is to select documents from a database in response to a user's query, and rank these documents according to relevance. This has been usually accomplished using statistical methods (often coupled with manual encoding) that (a) select terms (words,

phrases, and other units) from documents that are deemed to best represent their content, and (b) create an inverted index file (or files) that provide an easy access to documents containing these terms. A subsequent search process will attempt to match preprocessed user queries against term-based representations of documents in each case determining a degree of relevance between the two which depends upon the number and types of matching terms. Although many sophisticated search and matching methods are available, the crucial problem remains to be that of an adequate representation of content for both the documents and the queries.

In term-based representation, a document (as well as a query) is transformed into a collection of weighted terms, derived directly from the document text or indirectly through thesauri or domain maps. The representation is anchored on these terms, and thus their careful selection is critical. Since each unique term can be thought to add a new dimensionality to the representation, it is equally critical to weigh them properly against one another so that the document is placed at the correct position in the N-dimensional term space. Our goal here is to have the documents on the same topic placed close together, while those on different topics placed sufficiently apart. Unfortunately, we often do not know how to compute terms weights. The statistical weighting formulas, based on terms distribution within the database, such as *tf.idf*, are far from optimal, and the assumptions of term independence which are routinely made are false in most cases. This situation is even worse when single-word terms are intermixed with phrasal terms and the term independence becomes harder to justify.

The simplest word-based representations of content, while relatively better understood, are usually inadequate since single words are rarely specific enough for accurate discrimination, and their grouping is often accidental. A better method is to identify groups of words that create meaningful *phrases*, especially if these phrases denote important concepts in the database domain. For example, *joint venture* is an important term in the Wall Street Journal (WSJ henceforth) database, while neither *joint* nor *venture* is important by itself. In the retrieval experiments with the training TREC database, we noticed that both *joint* and *venture* were dropped from the list of terms by the system because their idf (*inverted document frequency*) weights were too low. In large databases, such as TIPSTER, the use of phrasal terms is not just desirable, it becomes necessary.

The challenge is to obtain ''semantic'' phrases, or ''concepts'', which would capture underlying semantic uniformity across various surface forms of expression. Syntactic structures are often reasonable indicators of content, certainly better than 'statistical phrases' — where words are grouped solely on the basis of physical proximity (e.g., "college junior" is not the same as "junior college") — however, the creation of compound terms makes the term matching process more complex since in addition to the usual problems of lexical meaning, one must deal with structure (e.g., "college junior" is the same as "junior in college"). In order to deal with structure, the parser's output needs to be "normalized" or "regularized" so that complex terms with the same or closely related meanings would indeed receive matching representations. One way to regularize syntactic structures is to transform them into operator-argument form, or at least head-modifier form, as will be further explained in this paper. In effect, therefore, we aim at obtaining a semantic representation. This result has been achieved to a certain extent in our work thus far.

Do we need to parse indeed? Our recent results indicate that some of the critical *semantic* dependencies can in fact be obtained without the intermediate step of syntactic analysis, and directly from lexical-level representation of text. We have applied our noun phrase disambiguation method directly to word sequences generated using part-of-speech information, and the results were most promising. At this time we have no data how these results compare to those obtained via parsing.

No matter how we eventually arrive at the compound terms, we hope they would let us to capture more accurately the semantic content of a document. It is certainly true that the compound terms such as *South Africa*, or *advanced document processing*, when found in a document, give us a better idea about the content of such document than isolated word matches. What happens, however, if we do not find them in a document? This situation may arise for several reasons: (1) the term/concept is not there, (2) the concept is there but our system is unable to identify it, or (3) the concept is not explicitly there, but its presence can be infered using general or domain-specific knowledge. This is certainly a serious problem, since we now attach more weight to concept matching than isolated word matching, and missing a concept can reflect more dramatically on system's recall. The inverse is also true: finding a concept where it really isn't makes an irrelevant document more likely to be highly ranked than with single-word based representation. Thus, while the rewards maybe greater, the risks are increasing as well.

One way to deal with this problem is to allow the system to fall back on partial matches and single word matches when concepts are not available, and to use query expansion techniques to supply missing terms. Unfortunately, thesaurus-based query expansion is usually quite uneffective, unless the subject domain is sufficiently narrow and the thesaurus sufficiently domain-specific. For example, the term *natural language* may be considered to subsume a term denoting a specific human language, e.g., *English*. Therefore, a query containing the former may be expected to retrieve documents containing the latter. The same can be said about *language* and *English*, unless *language* is in fact a part of the compound term *programming language* in which case the association *language - Fortran* is appropriate. This is a problem because (a) it is a standard practice to include both simple and compound terms in document representation, and (b) term associations have thus far been computed primarily at word level (including fixed phrases) and therefore care must be taken when such associations are used in term matching. This may prove particularly troublesome for systems that attempt term clustering in order to create "meta-terms" to be used in document representation.

In the remainder of this paper we discuss particulars of the present system and some of the observations made while processing TREC-4 data. While this description is meant to be self-contained, the reader may want to refer to previous TREC papers by this group for more information about the system.

OVERALL DESIGN

Our information retrieval system consists of a traditional statistical backbone (NIST's PRISE system [2]) augmented with various natural language processing components that assist the system in database processing (stemming, indexing, word and phrase clustering, selectional restrictions), and translate a user's information request into an effective query. This design is a careful compromise between purely statistical non-linguistic approaches and those requiring rather accomplished (and expensive) semantic analysis of data, often referred to as 'conceptual retrieval'.

In our system the database text is first processed with a fast syntactic parser. Subsequently certain types of phrases are extracted from the parse trees and used as compound indexing terms in addition to single-word terms. The extracted phrases are statistically analyzed as syntactic contexts in order to discover a variety of similarity links between smaller subphrases and words occurring in them. A further filtering process maps these similarity links onto semantic relations (generalization, specialization, synonymy, etc.) after which they are used to transform a user's request into a search query.

The user's natural language request is also parsed, and all indexing terms occurring in it are identified. Certain highly ambiguous, usually single-word terms may be dropped, provided that they also occur as elements in some compound terms. For example, "natural" is deleted from a query already containing "natural language" because "natural" occurs in many unrelated contexts: "natural number", "natural logarithm", "natural approach", etc. At the same time, other terms may be added, namely those which are linked to some query term through admissible similarity relations. For example, "unlawful activity" is added to a query (TREC topic 055) containing the compound term "illegal activity" via a synonymy link between "illegal" and "unlawful". After the final query is constructed, the database search follows, and a ranked list of documents is returned. In TREC-4, the automatic query expansion has been limited to to routing runs, where we refined our version of massive expansion using relevenace information wrt. the training database. Query expansion via automatically generated domain map was not usd in offical ad-hoc runs. Full details of TTP parser have been described in the TREC-1 report [8], as well as in other works [6,7], [9,10,11,12].

As in TREC-3, we used a randomized index splitting mechanism which creates not one but several balanced sub-indexes. These sub-indexes can be searched independently and the results can be merged meaningfully into a single ranking.

LINGUISTIC TERMS

Syntactic phrases extracted from TTP parse trees are head-modifier pairs. The head in such a pair is a central element of a phrase (main verb, main noun, etc.), while the modifier is one of the adjunct arguments of the head. In the TREC experiments reported here we extracted head-modifier word and fixed-phrase pairs only. The following types of pairs are considered: (1) a head noun and its left adjective or noun adjunct, (2) a head noun and the head of its right adjunct, (3) the main verb of a clause and the head of its object phrase, and (4) the head of the subject phrase and the main verb. These types of pairs account for most of the syntactic variants [5] for relating two words (or simple phrases) into pairs carrying compatible semantic content. For example, the pair *retrieve+information* will be extracted from any of the following fragments: *information retrieval system; retrieval of information from databases*; and

information that can be retrieved by a user-controlled interactive search process.

The notorious ambiguity of nominal compounds remains a serious difficulty in obtaining head-modifier pairs of highest accuracy. In order to cope with this, the pair extractor looks at the distribution statistics of the compound terms to decide whether the association between any two words (nouns and adjectives) in a noun phrase is both syntactically valid and semantically significant. For example, we may accept *language+natural* and *processing+language* from *natural language processing* as correct, however, *case+trading* would make a mediocre term when extracted from *insider trading case*. On the other hand, it is important to extract *trading+insider* to be able to match documents containing phrases *insider trading sanctions act* or *insider trading activity*.

Proper names, of people, places, events, organizations, etc., are often critical in deciding relevance of a document. Since names are traditionally capitalized in English text, spotting them is relatively easy, most of the time. It is important that all names recognized in text, including those made up of multiple words, e.g., *South Africa* or *Social Security*, are represented as tokens, and not broken into single words, e.g., *South* and *Africa*, which may turn out to be different names altogether by themselves. On the other hand, we need to make sure that variants of the same name are indeed recognized as such, e.g., *U.S. President Bill Clinton* and *President Clinton*, with a degree of confidence. One simple method, which we use in our system, is to represent a compound name dually, as a compound token and as a set of single-word terms. This way, if a corresponding full name variant cannot be found in a document, its component words matches can still add to the document score.

TERM WEIGHTING ISSUES

Finding a proper term weighting scheme is critical in term-based retrieval since the rank of a document is determined by the weights of the terms it shares with the query. One popular term weighting scheme, known as tf.idf, weights terms proportionately to their inverted document frequency scores and to their in-document frequencies (tf). The in-document frequency factor is usually normalized by the document length, that is, it is more significant for a term to occur 5 times in a short 20-word document, than to occur 10 times in a 1000-word article.

In our post-TREC-2 experiments we changed the weighting scheme so that the phrases (but not the names which we did not distinguish in TREC-2) were more heavily weighted by their idf scores while the in-document frequency scores were replaced by logarithms multiplied by sufficiently large constants. In addition, the top N highest-idf matching terms (simple or compound) were counted more toward the document score than the remaining terms. This 'hot-spot' retrieval option is discussed in the next section.

Schematically, these new weights for phrasal and highly specific terms are obtained using the following formula, while weights for most of the single-word terms remain unchanged:

$$weight(T_i)=(C_1*log(tf)+C_2*\alpha(N,i))*idf$$

In the above, $\alpha(N,i)$ is 1 for $i<N$ and is 0 otherwise. The $\alpha(N,i)$ factor realizes our notion of ''hot spot'' matching, where only top N matches are used in computing the document score. This creates an effect of ''locality'', somewhat similar to that achieved by passage-level retrieval. In TREC-3, where this weighing scheme was fully deployed for the first time, it proved very useful for sharpening the focus of long, frequently convoluted queries. In TREC-3 where the query length ranged from 20 to 100+ valid terms, setting N to 15 or 20 (including phrasal concepts) typically lead to a precision gain of about 20%. In TREC-4, the average query length is less than 10 terms, which we considered too short for using locality matching, and this part of the weighting scheme was in effect unused in the official runs. This turned out to be a mistake, as we rerun TREC-4 experiments after the conference, only to find out that our results improved visibly when the locality part of the weighting scheme was restored.

Changing the weighting scheme for compound terms, along with other minor improvements (such as expanding the stopword list for topics) has lead to the overall increase of precision of 20% to 25% over our baseline results in TREC-3.

SUMMARY OF RESULTS

The bulk of the text data used in TREC-4 has been previously processed for TREC-3 (about 3.3 GBytes). Routing experiments involved some additional new text (about 500 MBytes), which we processed through our NLP module. The parameters of this process were essentially the same as in TREC-3, and an interested reader is referred to our TREC-3 paper. Two types of retrieval have been done: (1) new topics 201-250 were run in the ad-hoc mode against the Disk-2&3 database,[1] and (2) topics 3-191

[1] Actually, only 49 topics were used in evaluation, since relevance judgements were unavailable for topic 201 due to an error.

(a selection of 50 topics in this range), previously used in TREC-1 to TREC-3, were run in the routing mode against the Disk-1 database plus the new data including material from Federal Register, IR Digest and Internet newsgroups. In each category 2 official runs were performed, with different set up of system's parameters. Massive query expansion has been implemented as an automatic feedback mode using known relevance judgements for these topics with respect TREC-3 database.

Summary statistics for routing runs are shown in Tables 1 and 2. In general, we can note substantial improvement in performance when phrasal terms are used, especially in ad-hoc runs. Looking back at TREC-2 and TREC-3 one may observe that these improvements appear to be tied to the length and specificity of the query: the longer the query, the more improvement from linguistic processes. This can be seen comparing the improvement over baseline for automatic adhoc runs (very short queries), for manual runs (longer queries), and for semi-interactive runs (yet longer queries). In addition, our TREC-3 results (with long and detailed queries) showed 20-25% improvement in precision attributed to NLP, as compared to 10-16% in TREC-4. At this time we are unable to explain the much smaller improvements in routing evaluations: while the massive query expansion definitely works, NLP has hard time topping these improvements.

CONCLUSIONS

We presented in some detail our natural language information retrieval system consisting of an advanced NLP module and a 'pure' statistical core engine. While many problems remain to be resolved, including the question of adequacy of term-based representation of document content, we attempted to demonstrate that the architecture described here is nonetheless viable. In particular, we demonstrated that natural language processing can now be done on a fairly large scale and that its speed and robustness has improved to the point where it can be applied to real IR problems. We suggest, with some caution until more experiments are run, that natural language processing can be very effective in creating appropriate search queries out of user's initial specifications which can be frequently imprecise or vague. An encouraging thing to note is the sharp increase of precision near the top of the ranking. This indicates a higher than average concentration of relevant documents in the first 10-20 documents retrieved, which can leverage further gains in performance via an automatic feedback process. This should be our focus in TREC-5.

Run	base	xbase	nyuge1	nyuge2
Tot number of docs over all queries				
Rel	6576	6576	6576	6576
RelRet	3641	4967	5078	5112
%chg		+36.0	+39.0	+40.0
Average precision over all rel docs				
Avg	0.1697	0.2715	0.2838	0.2913
%chg		+60.0	+67.0	+72.0
Precision at				
5 docs	0.3760	0.5480	0.5560	0.5680
10 docs	0.3680	0.4840	0.5000	0.5220
15 docs	0.3427	0.4680	0.4880	0.4933

Table 1. Automatic routing with 50 queries from 3-191 range: (1) *base* - statistical terms only, no expansion; (2) *xbase* - massive query expansion, no phrases; (3) *nyuge1* - phrases, names, with massive expansion up to 500 terms; (4) *nyuge2* - expansion limited to 200 terms per query.

Run	abase	aloc	mbase	mloc	iloc
Tot number of docs over all queries					
Rel	6501	6501	6501	6501	6501
RelR	2458	2498	3410	3545	3723
%chg		+1.6	+39.0	+44.0	+51.0
Average precision over all rel docs					
Avg	0.1394	0.1592	0.2082	0.2424	0.2767
%chg		+14.0	+49.0	+74.0	+98.0
Precision at					
5 docs	0.3755	0.4571	0.5020	0.5592	0.6694
10 doc	0.3408	0.3939	0.4510	0.4816	0.6082
15 doc	0.3088	0.3687	0.4082	0.4490	0.5633

Table 2. Ad-hoc runs with queries 202-250: (1) *abase* - automatic statistical terms only; (2) *aloc* - automatic phrases and names, locality N=20; (3) *mbase* - queries manually expanded, no phrases; (4) *mloc* - manual phrases, locality N=20; (5) *iloc* - interactive phrases, locality N=20.

At the same time it is important to keep in mind that the NLP techniques that meet our

performance requirements (or at least are believed to be approaching these requirements) are still fairly unsophisticated in their ability to handle natural language text. In particular, advanced processing involving conceptual structuring, logical forms, etc., is still beyond reach, computationally. It may be assumed that these advanced techniques will prove even more effective, since they address the problem of representation-level limits; however the experimental evidence is sparse and necessarily limited to rather small scale tests.

ACKNOWLEDGEMENTS

We would like to thank Donna Harman of NIST for making her PRISE system available to us. Will Rogers provided valuable assistance in installing updated versions of PRISE at NYU. We would also like to thank Ralph Weischedel and Constantine Papageorgiou of BBN for providing and assisting in the use of the part of speech tagger. This paper is based upon work supported by the Advanced Research Projects Agency under Tipster Phase-2 Contract 94-FI57900-000, and the National Science Foundation under Grant IRI-93-02615.

REFERENCES

[1] Frakes, William, B. and Ricardo Baeza-Yates. (eds). 1992. *Information Retrieval* Prentice-Hall, Englewood Cliffs, NJ.

[2] Harman, Donna and Gerald Candela. 1989. "Retrieving Records from a Gigabyte of text on a Minicomputer Using Statistical Ranking." *Journal of the American Society for Information Science,* 41(8), pp. 581-589.

[3] Meteer, Marie, Richard Schwartz, and Ralph Weischedel. 1991. "Studies in Part of Speech Labeling." Proceedings of the 4th DARPA Speech and Natural Language Workshop, Morgan-Kaufman, San Mateo, CA. pp. 331-336.

[4] Sager, Naomi. 1981. *Natural Language Information Processing.* Addison-Wesley.

[5] Sparck Jones, K. and J. I. Tait. 1984. "Automatic search term variant generation." *Journal of Documentation,* 40(1), pp. 50-66.

[6] Strzalkowski, Tomek and Barbara Vauthey. 1992. "Information Retrieval Using Robust Natural Language Processing." Proc. of the 30th ACL Meeting, Newark, DE, June-July. pp. 104-111.

[7] Strzalkowski, Tomek. 1992. "TTP: A Fast and Robust Parser for Natural Language." Proceedings of the 14th International Conference on Computational Linguistics (COLING), Nantes, France, July 1992. pp. 198-204.

[8] Strzalkowski, Tomek. 1993. "Natural Language Processing in Large-Scale Text Retrieval Tasks." Proceedings of the First Text REtrieval Conference (TREC-1), NIST Special Publication 500-207, pp. 173-187.

[9] Strzalkowski, Tomek and Jose Perez-Carballo. 1994. "Recent Developments in Natural Language Text Retrieval." Proceedings of the Second Text REtrieval Conference (TREC-2), NIST Special Publication 500-215, pp. 123-136.

[10] Strzalkowski, Tomek, Jose Perez-Carballo and Mihnea Marinescu. 1995. "Natural Language Information Retirieval: TREC-3 Report." Proceedings of the Third Text REtrieval Conference (TREC-3), NIST Special Publication 500-225, pp. 39-53.

[11] Strzalkowski, Tomek. 1995. "Natural Language Information Retrieval" **Information Processing and Management**, Vol. 31, No. 3, pp. 397-417. Pergamon/Elsevier.

[12] Strzalkowski, Tomek, and Peter Scheyen. 1993. "An Evaluation of TTP Parser: a preliminary report." Proceedings of International Workshop on Parsing Technologies (IWPT-93), Tilburg, Netherlands and Durbuy, Belgium, August 10-13.