Robust Text Processing in Automated Information Retrieval

Tomek Strzalkowski
Courant Institute of Mathematical Sciences
New York University
715 Broadway, rm. 704
New York, NY 10003
tomek@cs.nyu.edu

Abstract

We report on the results of a series of experiments with a prototype text retrieval system which uses relatively advanced natural language processing techniques in order to enhance the effectiveness of statistical document retrieval. In this paper we show that large-scale natural language processing (hundreds of millions of words and more) is not only required for a better retrieval, but it is also doable, given appropriate resources. In particular, we demonstrate that the use of syntactic compounds in the representation of database documents as well as in the user queries, coupled with an appropriate term weighting strategy, can considerably improve the effectiveness of retrospective search. The experiments reported here were conducted on TIPSTER database in connection with the Text REtrieval Conference series (TREC).[1]

1 Introduction

The task of information retrieval is to extract *relevant* documents from a large collection of documents in response to user queries. When the documents contain primarily unrestricted text (e.g., newspaper articles, legal documents, etc.) the relevance of a document is established through 'full-text' retrieval. This has been usually accomplished by identifying key terms in the documents (the process known as 'indexing') which could then be matched against terms in queries (Salton, 1989). The effectiveness of any such term-based approach is directly related to the accuracy with which a set of terms represents the content of a document, as well as how well it contrasts a given document with respect to other documents. In other words, we are looking for a representation *R* such that for any text items *D1* and *D2*, *R(D1) = R(D2)* iff *meaning(D1) = meaning(D2)*, at an appropriate level of abstraction (which may depend on the types and character of anticipated queries).

The simplest word-based representations of content 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* are important by themselves. In fact, in a 800+ MBytes database, both *joint* and *venture* would often be dropped from the list of terms by the system because their inverted document frequency (*idf*) weights were too low. In large databases comprising hundreds of thousands of documents the use of phrasal terms is not just desirable, it becomes necessary.

To illustrate this point let us consider TREC Topic 104, an information request from which a database search query is to be built. The reader may note various sections of this Topic, with <desc> corresponding to the user's original request, further elaborated in <narr>, and <con> consisting of expert-assigned phrases denoting key concepts to be considered.

<top>
<num> Number: 104
<dom> Domain: Law and Government
<title> Topic: Catastrophic Health Insurance
<desc> Description:
Document will enumerate provisions of the U.S. Catastrophic Health
Insurance Act of 1988, or the political/legal fallout from that
legislation.
<narr> Narrative:
A relevant document will detail the content of the U.S. medicare
act of 1988 which extended catastrophic illness benefits
to the elderly, with particular attention to the financing scheme
which led to a firestorm of protest and a Congressional retreat,
or a relevant document will detail the political/legal
consequences of the catastrophic health insurance imbroglio and
subsequent efforts by Congress to provide similar coverages
through a less-controversial mechanism.
<con> Concept(s):
1. Catastrophic Coverage Act of 1988, Medicare Part B,
Health Care Financing Administration

[1] See (Harman, 1993) for a detailed description of TREC.

2. catastrophic-health program, catastrophic illness, catastrophic care, acute care, long-term nursing home care
3. American Association of Retired Persons, AARP, senior citizen, National Committee to Preserve Social Security and Medicare
</top>

If the phrases are ignored altogether,[2] this query will produce an output where the relevant documents are scattered as shown in the first table below which lists the ranks and scores of relevant documents within the top 100 retrieved documents. On the other hand, if we include even simple phrases, such as *catastrophic-health program*, *acute care*, *home care*, and *senior citizen*, we can considerably sharpen the outcome of the search as seen in the second table.[3]

QUERY:104; NO. RELEVANT:21

REL DOCUMENT	RANK (no phrases)	RANK (phrases)
WSJ890918-0173	2	5
WSJ891004-0119	7	1
WSJ870723-0064	8	8
WSJ870213-0053	10	12
WSJ880608-0121	14	7
WSJ891005-0005	15	4
WSJ891009-0009	35	18
WSJ890920-0115	39	26
WSJ890928-0184	40	61
WSJ880609-0061	53	50
WSJ891009-0188	73	46
WSJ880705-0194	97	95
WSJ870601-0075	-	52
WSJ891005-0001	-	72
WSJ871028-0059	-	93

A query obtained from the fields <title>, <desc> and <narr> will be, as may be expected, much weaker than the one using <con> field, especially without the phrasal terms, because the narrative contains far fewer specific terms while containing some that may prove distracting, e.g., *firestorm*. In fact, Broglio and Croft (1993), and Broglio (personal communication, 1993) showed that the exclusion of the <con> field makes the queries quite ineffective, while adding the <narr> field makes them even worse as they lose precision by as much as 30%. However, adding phrasal terms can improve things considerably. We return to this issue later in the paper.

An accurate syntactic analysis is an essential prerequisite for selection of phrasal terms. Various statistical methods, e.g., based on word co-occurrences and mutual information, as well as partial parsing techniques, are prone to high error rates (sometimes as high as 50%), turning out many unwanted associations. Therefore a good, fast parser is necessary, but it is by no means sufficient. While syntactic phrases are often better indicators of content 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" — the creation of compound terms makes the term matching process more complex since in addition to the usual problems of synonymy and subsumption, one must deal with their structure (e.g., "college junior" is the same as "junior in college").

For all kinds of terms that can be assigned to the representation of a document, e.g., words, syntactic phrases, fixed phrases, and proper names, various levels of ''regularization'' are needed to assure that syntactic or lexical variations of input do not obscure underlying semantic uniformity. Without actually doing semantic analysis, this kind of normalization can be achieved through the following processes:[4]

(1) morphological stemming: e.g., *retrieving* is reduced to *retriev*;

(2) lexicon-based word normalization: e.g., *retrieval* is reduced to *retrieve*;

(3) operator-argument representation of phrases: e.g., *information retrieval*, *retrieving of information*, and *retrieve relevant information* are all assigned the same representation, *retrieve+information*;

(4) context-based term clustering into synonymy classes and subsumption hierarchies: e.g., *takeover* is a kind of *acquisition* (in business), and *Fortran* is a *programming language*.

Introduction of compound terms complicates the task of discovery of various semantic relationships among them. For example, the term *natural language* can often be considered to subsume any term denoting a specific human language, such as *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

[2] All single words (except the stopwords such as articles or prepositions) are included in the query, including those making up the phrases.

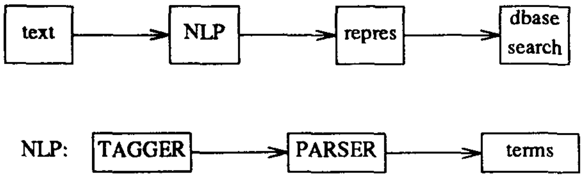
[3] Including extra terms in documents changes the way other terms are weighted. This issue is discussed later in this paper.

[4] An alternative, but less efficient method is to generate all variants (lexical, syntactic, etc.) of words/phrases in the queries (Sparck-Jones & Tait, 1984).

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.

2 Overall Design

We have established the general architecture of a NLP-IR system, depicted schematically below, in which an advanced NLP module is inserted between the textual input (new documents, user queries) and the database search engine (in our case, NIST's PRISE system). This design has already shown some promise in producing a better performance than the base statistical system (Strzalkowski, 1993b). We would like to point out at the outset that this system is completely automated, including the statistical core, and the natural language processing components, and no human intervention or manual encoding is required.



In our system the database text is first processed with a sequence of programs that include a part-of-speech tagger, a lexicon-based morphological stemmer and 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" may be 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".

One of the observations made during the course of TREC-2 was to note that removing low-quality terms from the queries is at least as important (and often more so) as adding synonyms and specializations. In some instances (e.g., routing runs) low-quality terms had to be removed (or inhibited) *before* similar terms could be added to the query or else the effect of query expansion was all but drowned out by the increased noise.

After the final query is constructed, the database search follows, and a ranked list of documents is returned. It should be noted that all the processing steps, those performed by the backbone system, and those performed by the natural language processing components, are fully automated, and no human intervention or manual encoding is required.

3 Fast Parsing with TTP Parser

TTP (Tagged Text Parser) is a full-grammar parser based on the Linguistic String Grammar developed by Sager (1981). It currently encompasses most of the grammar productions and many of the restrictions, but it is by no means complete. Unlike a conventional parser, TTP's output is a regularized representation of each sentence which reflects its logical predicate-argument structure, e.g., logical subject and logical objects are identified depending upon the main verb subcategorization frame. For example, the verb *abide* has, among others, a subcategorization frame in which the object is a prepositional phrase with *by*, as in *he'll abide by the court's decision*, i.e.,

ABIDE: *subject* NP *object* PREP by NP

Subcategorization information is read from the on-line Oxford Advanced Learner's Dictionary (OALD) which TTP uses.

Also unlike a conventional parser, TTP is equipped with a powerful skip-and-fit recovery mechanism that allows it to operate effectively in the face of ill-formed input or under severe time pressure. A built-in timer regulates the amount of time allowed for parsing any one sentence: if a parse is not returned before the allotted time elapses, TTP enters the skipping mode in which it will try to "fit" the parse. While in the skip-and-fit mode, the parser attempts to forcibly reduce incomplete constituents, possibly skimming over portions of input in order to restart processing at a next unattempted constituent; in other words, it will favor reduction over backtracking. The result of this strategy is an approximate parse, partially fitted using top-down predictions. In runs with approximately 130 million words of TREC's Wall Street Journal and San Jose Mercury texts, the parser's speed averaged 30 minutes per Megabyte or about 80 words per second, on a Sun SparcStation10. In addition, TTP has been shown to produce parse structures which are no worse than those generated by full-scale linguistic parsers when compared to hand-coded parse trees.[5]

Full details of TTP parser have been described in the TREC-1 report (Strzalkowski, 1993a), as well as in other works (Strzalkowski, 1992; Strzalkowski & Scheyen, 1993).

As may be expected, the skip-and-fit strategy will only be effective if the input skipping can be performed with a degree of determinism. This means that most of the lexical level ambiguity must be removed from the input text, prior to parsing. We achieve this using a stochastic parts of speech tagger to preprocess the text prior to parsing. In order to streamline the processing, we also perform morphological normalization of words on the tagged text, before parsing. This is possible because the part-of-speech tags retain the information about each word's original form. Thus the sentence *The Soviets have been notified* is transformed into *the/dt soviet/nps have/vbp be/vbn notify/vbn* before parsing commences.[6]

4 Head-Modifier Structures

Syntactic phrases extracted from TTP parse structures are represented as 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 adjuncts or arguments of the head. In the TREC experiments reported here we extracted head-modifier word pairs only, i.e., nested pairs were not used even though this was warranted by the size of the database.

Figure 1 shows all stages of the initial linguistic analysis of a sample sentence from the WSJ database. The reader may note that the parser's output is a predicate-argument structure centered around the main elements of various phrases. For example, BE is the main predicate (modified by HAVE) with 2 arguments (*subject, object*) and 2 adjuncts (*adv, sub_ord*). INVADE is the predicate in the subordinate clause with 2 arguments (*subject, object*). The subject of BE is a noun phrase with PRESIDENT as the head element, two modifiers (FORMER, SOVIET) and a determiner (THE). From this structure, we extract head-modifier pairs that become candidates for compound terms. In general, the following types of pairs are considered: (1) a head noun of a noun phrase 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 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.* We also attempted to identify and remove any terms which were explicitly negated in order to prevent matches against their positive counterparts, either in the database or in the queries.

One difficulty in obtaining head-modifier pairs of highest accuracy is the notorious ambiguity of nominal compounds. 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

INPUT SENTENCE
The former Soviet president has been a local hero ever since a Russian tank invaded Wisconsin.

TAGGED SENTENCE
The/*dt* former/*jj* Soviet/*jj* president/*nn* has/*vbz* been/*vbn* a/*dt* local/*jj* hero/*nn* ever/*rb* since/*in* a/*dt* Russian/*jj* tank/*nn* invaded/*vbd* Wisconsin/*np* ./*per*

TAGGED & STEMMED SENTENCE
the/*dt* former/*jj* soviet/*jj* president/*nn* have/*vbz* be/*vbn* a/*dt* local/*jj* hero/*nn* ever/*rb* since/*in* a/*dt* russian/*jj* tank/*nn* invade/*vbd* wisconsin/*np* ./*per*

PARSED SENTENCE
```
[assert
  [[perf [HAVE]] [[verb [BE]]
      [subject [np [n PRESIDENT] [t_pos THE]
              [adj [FORMER]] [adj [SOVIET]]]]
      [object  [np [n HERO] [t_pos A] [adj [LOCAL]]]]
      [adv EVER]
      [sub_ord [SINCE
             [[verb [INVADE]]
                [subject [np [n TANK] [t_pos A]
                        [adj [RUSSIAN]]]]
                [object [np [name [WISCONSIN]]]]]]]]]
```

EXTRACTED TERMS & WEIGHTS

president	2.623519	soviet	5.416102
president+soviet	11.556747	president+former	14.594883
hero	7.896426	hero+local	14.314775
invade	8.435012	tank	6.848128
tank+invade	17.402237	tank+russian	16.030809
russian	7.383342	wisconsin	7.785689

Figure 1. Stages of sentence processing.

[5] Hand-coded parse trees were obtained from the University of Pennsylvania Treebank Project database.

[6] The tags are read as follows: *dt* is determiner, *nps* is a proper name, *vbp* is a tensed plural verb, *vbn* is a past participle.

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*.

5 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 in a short 100-word abstract, than in a 5000-word article.[7]

A standard tf.idf weighting scheme (see Buckley, 1993 for details) may be inappropriate for mixed term sets, consisting of ordinary concepts, proper names, and phrases, because:

(1) It favors terms that occur fairly frequently in a document, which supports only general-type queries (e.g., "all you know about X"). Such queries were not typical in TREC.

(2) It attaches low weights to infrequent, highly specific terms, such as names and phrases, whose only occurrences in a document are often decisive for relevance. Note that such terms cannot be reliably distinguished using their distribution in the database as the sole factor, and therefore syntactic and lexical information is required.

(3) It does not address the problem of inter-term dependencies arising when phrasal terms and their component single-word terms are all included in a document representation, i.e., *launch+satellite* and *satellite* are not independent, and it is unclear whether they should be counted as two terms.

In our post-TREC-2 experiments we considered (1) and (2) only. 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.

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 selection of a weighting formula was partly constrained by the fact that document-length-normalized tf weights were precomputed at the indexing stage and could not be altered without re-indexing of the entire database. The intuitive interpretation of the $\alpha(N,i)$ factor is as follows. We restrict the maximum number of terms on which a query is permitted to match a document to N highest weight terms, where N can be the same for all queries or may vary from one query to another. Note that this is not the same as simply taking the N top terms from each query. Rather, for each document for which there are M matching terms with the query, only min(M,N) of them, namely those which have highest weights, will be considered when computing the document score. Moreover, only the global importance weights for terms are considered (such as idf), while local in-document frequency (eg., tf) is suppressed by either taking a log or replacing it with a constant.

Changing the weighting scheme for compound terms, along with other minor improvements (such as expanding the stopword list for topics, or correcting a few parsing bugs) has lead to an overall increase of precision of more than 20% over our official TREC-2 ad-hoc results. Table 1 includes statistics of these new runs for 50 queries (numbered 101-150) against the WSJ database. The gap between the precision levels in columns *txt2* and *con* reflects the difference in the quality of the queries obtained from the narrative parts of the topics (*txt2* = *title* + *desc* + *narr*), and those obtained primarily from expert's formulation (*title* + *desc* + *con*). The column *txt2+nlp* represents the improvement of *txt2* queries thanks to NLP, with as much as 70% of the gap closed. Similar improvements have been obtained for other sets of queries.

6 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

[7] This is not always true, for example when all occurrences of a term are concentrated in a single section or a paragraph rather than spread around the article.

Run	txtl	txt2	txt2+nlp	con	con+nlp
Tot number of docs over all queries					
Rel	3929	3929	3929	3929	3929
RelRet	2736	3025	3108	3332	3401
%chg		+9.0	+14.7	+21.8	+24.3
Recall	(interp) Precision Averages				
0.00	0.6874	0.7318	0.7201	0.7469	0.8063
0.10	0.4677	0.5293	0.5239	0.5726	0.6198
0.20	0.3785	0.4532	0.4751	0.4970	0.5566
0.30	0.3060	0.3707	0.4122	0.4193	0.4786
0.40	0.2675	0.3276	0.3541	0.3747	0.4257
0.50	0.2211	0.2815	0.3126	0.3271	0.3828
0.60	0.1765	0.2406	0.2752	0.2783	0.3380
0.70	0.1313	0.1783	0.2142	0.2267	0.2817
0.80	0.0828	0.1337	0.1605	0.1670	0.2164
0.90	0.0451	0.0818	0.1014	0.0959	0.1471
1.00	0.0094	0.0159	0.0194	0.0168	0.0474
Average precision over all rel docs					
Avg	0.2309	0.2835	0.3070	0.3210	0.3759
%chg		+22.8	+33.0	+39.0	+62.8
Precision at N documents					
5	0.5000	0.5240	0.5200	0.5600	0.6040
10	0.4080	0.4600	0.4900	0.5020	0.5580
100	0.2380	0.2790	0.2914	0.3084	0.3346
R-Precision (after Rel)					
Exact	0.2671	0.3053	0.3332	0.3455	0.3950
%chg		+14.3	+24.7	+29.3	+47.9

Table 1. Run statistics for 50 ad-hoc queries against WSJ database with 1000 docs retrieved per query: (1) *txt1* - single terms of <narr> and <desc> fields — this is the base run; (2) *txt2* - <narr> and <desc> fields with low weight terms removed; (3) *txt2+nlp* - <narr> and <desc> fields including syntactic phrase terms using the new weighting scheme; (4) *con* - <desc> and <con> fields with low weight terms removed but with no NLP; and (5) *con+nlp* - <desc> and <con> fields including phrases with the new weighting scheme.

processing can now be done on a fairly large scale and that its speed and robustness can match those of traditional statistical programs such as key-word indexing or statistical phrase extraction. 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 a user's initial specifications, which can be frequently imprecise or vague.

Acknowledgements

The author would like to thank Donna Harman of NIST for making her PRISE system available for this research. 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 Contract N00014-90-J-1851 from the Office of Naval Research, under Contract N00600-88-D-3717 from PRC Inc., under ARPA's Tipster Phase-2 Contract 94-FI57900-000, and the National Science Foundation under Grant IRI-93-02615.

References

Broglio, John and W. Bruce Croft. 1993. "Query Processing for Retrieval from Large Text Bases." Human Language Technology, Proceedings of the workshop, Princeton, NJ. Morgan-Kaufmann, pp. 353-357.

Buckley, Chris. 1993. "The Importance of Proper Weighting Methods." Human Language Technology, Proceedings of the workshop, Princeton, NJ. Morgan-Kaufmann, pp. 349-352.

Harman, Donna (ed.). 1993. *First Text REtrieval Conference*. NIST special publication 500-207.

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

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

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.

Strzalkowski, Tomek. 1993a. "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.

Strzalkowski, Tomek. 1993b. "Robust Text Processing in Automated Information Retrieval." Proc. of ACL-sponsored workshop on Very Large Corpora. Ohio State Univ. Columbus, June 22.

Strzalkowski, Tomek, and Peter Scheyen. 1993. "Evaluation of TTP Parser: a preliminary report." Proceedings of International Workshop on Parsing Technologies (IWPT-93), Tilburg, Netherlands and Durbuy, Belgium, pp. 293-308.