INFORMATION RETRIEVAL USING ROBUST NATURAL LANGUAGE PROCESSING

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

ABSTRACT

We developed a prototype information retrieval system which uses advanced natural language processing techniques to enhance the effectiveness of traditional key-word based document retrieval. The backbone of our system is a statistical retrieval engine which performs automated indexing of documents, then search and ranking in response to user queries. This core architecture is augmented with advanced natural language processing tools which are both robust and efficient. In early experiments, the augmented system has displayed capabilities that appear to make it superior to the purely statistical base.

INTRODUCTION

A typical 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), but it is now widely believed that these traditional methods have reached their limits.[1] These limits are particularly acute for text databases, where natural language processing (NLP) has long been considered necessary for further progress. Unfortunately, the difficulties encountered in applying computational linguistics technologies to text processing have contributed to a wide-spread belief that automated NLP may not be suitable in IR. These difficulties included inefficiency, limited coverage, and prohibitive cost of manual effort required to build lexicons and knowledge bases for each new text domain. On the other hand, while numerous experiments did not establish the usefulness of NLP, they cannot be considered conclusive because of their very limited scale.

Another reason is the limited scale at which NLP was used. Syntactic parsing of the database contents, for example, has been attempted in order to extract linguistically motivated "syntactic phrases", which presumably were better indicators of contents than "statistical phrases" where words were grouped solely on the basis of physical proximity (eg. "college junior" is not the same as "junior college"). These intuitions, however, were not confirmed by experiments; worse still, statistical phrases regularly outperformed syntactic phrases (Fagan, 1987). Attempts to overcome the poor statistical behavior of syntactic phrases has led to various clustering techniques that grouped synonymous or near synonymous phrases into "clusters" and replaced these by single "metaterms". Clustering techniques were somewhat successful in upgrading overall system performance, but their effectiveness was diminished by frequently poor quality of syntactic analysis. Since full-analysis wide-coverage syntactic parsers were either unavailable or inefficient, various partial parsing methods have been used. Partial parsing was usually fast enough, but it also generated noisy data: as many as 50% of all generated phrases could be incorrect (Lewis and Croft, 1990). Other efforts concentrated on processing of user queries (eg. Spack Jones and Tait, 1984; Smeaton and van Rijsbergen, 1988). Since queries were usually short and few, even relatively inefficient NLP techniques could be of benefit to the system. None of these attempts proved conclusive, and some were never properly evaluated either.

† Current address: Laboratoire d'Informatique, Universite de Fribourg, ch. du Musee 3, 1700 Fribourg, Switzerland; vauthey@cfruni51.bitnet.

[1] As far as the automatic document retrieval is concerned. Techniques involving various forms of relevance feedback are usually far more effective, but they require user's manual intervention in the retrieval process. In this paper, we are concerned with fully automated retrieval only.

[2] Standard IR benchmark collections are statistically too small and the experiments can easily produce counterintuitive results. For example, Cranfield collection is only approx. 180,000 English words, while CACM-3204 collection used in the present experiments is approx. 200,000 words.

We believe that linguistic processing of both the database and the user's queries need to be done for a maximum benefit, and moreover, the two processes must be appropriately coordinated. This prognosis is supported by the experiments performed by the NYU group (Strzalkowski and Vauthey, 1991; Grishman and Strzalkowski, 1991), and by the group at the University of Massachussetts (Croft et al., 1991). We explore this possibility further in this paper.

OVERALL DESIGN

Our information retrieval system consists of a traditional statistical backbone (Harman and Candela, 1989) 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'. The conceptual retrieval systems, though quite effective, are not yet mature enough to be considered in serious information retrieval applications, the major problems being their extreme inefficiency and the need for manual encoding of domain knowledge (Mauldin, 1991).

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 user's request into a search query.

The user's natural language request is also parsed, and all indexing terms occurring in them are identified. Next, certain highly ambiguous (usually single-word) terms are 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, "fortran" is added to a query containing the compound term "program language" via a specification link. 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 these performed by the natural language processing components, are fully automated, and no human intervention or manual encoding is required.

FAST PARSING WITH TTP PARSER

TTP (Tagged Text Parser) is based on the Linguistic String Grammar developed by Sager (1981). Written in Quintus Prolog, the parser currently encompasses more than 400 grammar productions. It produces regularized parse tree representations for each sentence that reflect the sentence's logical structure. The parser 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 a severe time pressure. In the recent experiments with approximately 6 million words of English texts,[3] the parser's speed averaged between 0.45 and 0.5 seconds per sentence, or up to 2600 words per minute, on a 21 MIPS SparcStation ELC. Some details of the parser are discussed below.[4]

TTP is a full grammar parser, and initially, it attempts to generate a complete analysis for each sentence. However, unlike an ordinary parser, it has a built-in timer which regulates the amount of time allowed for parsing any one sentence. If a parse is not returned before the allotted time elapses, the parser enters the skip-and-fit mode in which it will try to "fit" the parse. While in the skip-and-fit mode, the parser will attempt to forcibly reduce incomplete constituents, possibly skipping portions of input in order to restart processing at a next unattempted constituent. In other words, the parser will favor reduction to backtracking while in the skip-and-fit mode. The result of this strategy is an approximate parse, partially fitted using top-down predictions. The fragments skipped in the first pass are not thrown out, instead they are analyzed by a simple phrasal parser that looks for noun phrases and relative clauses and then attaches the recovered material to the main parse structure. As an illustration, consider the following sentence taken from the CACM-3204 corpus:

[3] These include CACM-3204, MUC-3, and a selection of nearly 6,000 technical articles extracted from Computer Library database (a Ziff Communications Inc. CD-ROM).

[4] A complete description can be found in (Strzalkowski, 1992).

> The method is illustrated by the automatic construction of both recursive and iterative programs operating on natural numbers, lists, and trees, in order to construct a program satisfying certain specifications *a theorem induced by those specifications is proved,* and the desired program is extracted from the proof.

The italicized fragment is likely to cause additional complications in parsing this lengthy string, and the parser may be better off ignoring this fragment altogether. To do so successfully, the parser must close the currently open constituent (i.e., reduce *a program satisfying certain specifications* to *NP*), and possibly a few of its parent constituents, removing corresponding productions from further consideration, until an appropriate production is reactivated. In this case, TTP may force the following reductions: $SI \rightarrow to\ V\ NP$; $SA \rightarrow SI$; $S \rightarrow NP\ V\ NP\ SA$, until the production $S \rightarrow S$ *and* S is reached. Next, the parser skips input to find *and*, and resumes normal processing.

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 [5] to preprocess the text.

WORD SUFFIX TRIMMER

Word stemming has been an effective way of improving document recall since it reduces words to their common morphological root, thus allowing more successful matches. On the other hand, stemming tends to decrease retrieval precision, if care is not taken to prevent situations where otherwise unrelated words are reduced to the same stem. In our system we replaced a traditional morphological stemmer with a conservative dictionary-assisted suffix trimmer.[6] The suffix trimmer performs essentially two tasks: (1) it reduces inflected word forms to their root forms as specified in the dictionary, and (2) it converts nominalized verb forms (eg. "implementation", "storage") to the root forms of corresponding verbs (i.e., "implement", "store"). This is accomplished by removing a standard suffix, eg. "stor+age", replacing it with a standard root ending ("+e"), and checking the newly created word against the dictionary, i.e., we check whether the new root ("store") is indeed a legal word, and whether the original root ("storage") is defined using the new root ("store") or one of its standard inflexional forms (e.g., "storing"). For example, the following definitions are excerpted from the *Oxford Advanced Learner's Dictionary* (OALD):

> **storage** *n* [U] (space used for, money paid for) the storing of goods ...
> **diversion** *n* [U] diverting ...
> **procession** *n* [C] number of persons, vehicles, etc moving forward and following each other in an orderly way.

Therefore, we can reduce "diversion" to "divert" by removing the suffix "+sion" and adding root form suffix "+t". On the other hand, "process+ion" is not reduced to "process".

Experiments with CACM-3204 collection show an improvement in retrieval precision by 6% to 8% over the base system equipped with a standard morphological stemmer (in our case, the SMART stemmer).

HEAD-MODIFIER STRUCTURES

Syntactic phrases extracted from TTP parse trees are head-modifier pairs: from simple word pairs to complex nested structures. The head in such a pair is a central element of a phrase (verb, main noun, etc.) while the modifier is one of the adjunct arguments of the head.[7] For example, the phrase *fast algorithm for parsing context-free languages* yields the following pairs: *algorithm+fast, algorithm+parse, parse+language, language+context_free*. The following types of pairs were 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 for relating two words (or simple phrases) into pairs carrying compatible semantic content. For example, the pair *retrieve+information* is 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*. An example is shown in Figure 1.[8] One difficulty in obtaining head-modifier

[5] Courtesy of Bolt Beranek and Newman.

[6] We use Oxford Advanced Learner's Dictionary (OALD).

[7] In the experiments reported here we extracted head-modifier word pairs only. CACM collection is too small to warrant generation of larger compounds, because of their low frequencies.

[8] Note that working with the parsed text ensures a high degree of precision in capturing the meaningful phrases, which is especially evident when compared with the results usually obtained from either unprocessed or only partially processed text (Lewis and Croft, 1990).

SENTENCE:
The techniques are discussed and related to a general tape manipulation routine.

PARSE STRUCTURE:

```
[[be],
  [[verb,[and,[discuss],[relate]]],
    [subject,anyone],
    [object,[np,[n,technique],[t_pos,the]]],
    [to,[np,[n,routine],[t_pos,a],[adj,[general]],
         [n_pos,[np,[n,manipulation]]],
         [n_pos,[np,[n,tape]]]]]]].
```

EXTRACTED PAIRS:

```
[discuss,technique], [relate,technique],
[routine,general], [routine,manipulate],
[manipulate,tape]
```

Figure 1. Extraction of syntactic pairs.

pairs of highest accuracy is the notorious ambiguity of nominal compounds. For example, the phrase *natural language processing* should generate *language+natural* and *processing+language*, while *dynamic information processing* is expected to yield *processing+dynamic* and *processing+information*. Since our parser has no knowledge about the text domain, and uses no semantic preferences, it does not attempt to guess any internal associations within such phrases. Instead, this task is passed to the pair extractor module which processes ambiguous parse structures in two phases. In phase one, all and only unambiguous head-modifier pairs are extracted, and frequencies of their occurrence are recorded. In phase two, frequency information of pairs generated in the first pass is used to form associations from ambiguous structures. For example, if *language+natural* has occurred unambiguously a number times in contexts such as *parser for natural language*, while *processing+natural* has occurred significantly fewer times or perhaps none at all, then we will prefer the former association as valid.

TERM CORRELATIONS FROM TEXT

Head-modifier pairs form compound terms used in database indexing. They also serve as occurrence contexts for smaller terms, including single-word terms. In order to determine whether such pairs signify any important association between terms, we calculate the value of the *Informational Contribution (IC)* function for each element in a pair. Higher values indicate stronger association, and the element having the largest value is considered semantically dominant.

The connection between the terms co-occurrences and the information they are transmitting (or otherwise, their meaning) was established and discussed in detail by Harris (1968, 1982, 1991) as fundamental for his mathematical theory of language. This theory is related to mathematical information theory, which formalizes the dependencies between the information and the probability distribution of the given code (alphabet or language). As stated by Shannon (1948), information is measured by entropy which gives the capacity of the given code, in terms of the probabilities of its particular signs, to transmit information. It should be emphasized that, according to the information theory, there is no direct relation between information and meaning, entropy giving only a measure of what possible choices of messages are offered by a particular language. However, it offers theoretic foundations of the correlation between the probability of an event and transmitted information, and it can be further developed in order to capture the meaning of a message. There is indeed an inverse relation between information contributed by a word and its probability of occurrence p, that is, rare words carry more information than common ones. This relation can be given by the function $-\log p(x)$ which corresponds to information which a single word is contributing to the entropy of the entire language.

In contrast to information theory, the goal of the present study is not to calculate informational capacities of a language, but to measure the relative strength of connection between the words in syntactic pairs. This connection corresponds to Harris' likelihood constraint, where the likelihood of an operator with respect to its argument words (or of an argument word in respect to different operators) is defined using word-combination frequencies within the linguistic dependency structures. Further, the likelihood of a given word being paired with another word, within one operator-argument structure, can be expressed in statistical terms as a conditional probability. In our present approach, the required measure had to be uniform for all word occurrences, covering a number of different operator-argument structures. This is reflected by an additional dispersion parameter, introduced to evaluate the heterogeneity of word associations. The resulting new formula $IC(x, [x,y])$ is based on (an estimate of) the conditional probability of seeing a word y to the right of the word x, modified with a dispersion parameter for x.

$$IC(x, [x,y]) = \frac{f_{x,y}}{n_x + d_x - 1}$$

where $f_{x,y}$ is the frequency of $[x,y]$ in the corpus, n_x is the number of pairs in which x occurs at the same position as in $[x,y]$, and $d(x)$ is the dispersion parameter understood as the number of distinct words with which x is paired. When $IC(x, [x,y]) = 0$, x and y never occur together (i.e., $f_{x,y} = 0$); when $IC(x, [x,y]) = 1$, x occurs only with y (i.e., $f_{x,y} = n_x$ and $d_x = 1$).

So defined, IC function is asymmetric, a property found desirable by Wilks et al. (1990) in their study of word co-occurrences in the Longman dictionary. In addition, IC is stable even for relatively low frequency words, which can be contrasted with Fano's mutual information formula recently used by Church and Hanks (1990) to compute word co-occurrence patterns in a 44 million word corpus of Associated Press news stories. They noted that while generally satisfactory, the mutual information formula often produces counterintuitive results for low-frequency data. This is particularly worrisome for relatively smaller IR collections since many important indexing terms would be eliminated from consideration. A few examples obtained from CACM-3204 corpus are listed in Table 1. IC values for terms become the basis for calculating term-to-term similarity coefficients. If two terms tend to be modified with a number of common modifiers and otherwise appear in few distinct contexts, we assign them a similarity coefficient, a real number between 0 and 1. The similarity is determined by comparing distribution characteristics for both terms within the corpus: how much information contents do they carry, do their information contribution over contexts vary greatly, are the common contexts in which these terms occur specific enough? In general we will credit high-contents terms appearing in identical contexts, especially if these contexts are not too commonplace.[9] The relative similarity between two words x_1 and x_2 is obtained using the following formula (α is a large constant): [10]

$$SIM(x_1,x_2) = log(\alpha \sum_y sim_y(x_1,x_2))$$

where

$$sim_y(x_1,x_2) = MIN(IC(x_1,[x_1,y]),IC(x_2,[x_2,y])) * (IC(y,[x_1,y]) + IC(y,[x_2,y]))$$

The similarity function is further normalized with respect to $SIM(x_1,x_1)$. It may be worth pointing out that the similarities are calculated using term co-occurrences in syntactic rather than in document-size contexts, the latter being the usual practice in non-linguistic clustering (eg. Sparck Jones and Barber, 1971; Crouch, 1988; Lewis and Croft, 1990). Although the two methods of term clustering may be considered mutually complementary in certain situations, we believe that more and stronger associations can be obtained through syntactic-context clustering, given sufficient amount of data and a reasonably accurate syntactic parser.[11]

word	head+modifier	IC coeff.
distribute	*distribute+normal*	0.040
normal	*distribute+normal*	0.115
minimum	*minimum+relative*	0.200
relative	*minimum+relative*	0.016
retrieve	*retrieve+inform*	0.086
inform	*retrieve+inform*	0.004
size	*size+medium*	0.009
medium	*size+medium*	0.250
editor	*editor+text*	0.142
text	*editor+text*	0.025
system	*system+parallel*	0.001
parallel	*system+parallel*	0.014
read	*read+character*	0.023
character	*read+character*	0.007
implicate	*implicate+legal*	0.035
legal	*implicate+legal*	0.083
system	*system+distribute*	0.002
distribute	*system+distribute*	0.037
make	*make+recommend*	0.024
recommend	*make+recommend*	0.142
infer	*infer+deductive*	0.095
deductive	*infer+deductive*	0.142
share	*share+resource*	0.054
resource	*share+resource*	0.042

Table 1. IC coefficients obtained from CACM-3204

QUERY EXPANSION

Similarity relations are used to expand user queries with new terms, in an attempt to make the

[9] It would not be appropriate to predict similarity between *language* and *logarithm* on the basis of their co-occurrence with *natural*.

[10] This is inspired by a formula used by Hindle (1990), and subsequently modified to take into account the asymmetry of IC measure.

[11] Non-syntactic contexts cross sentence boundaries with no fuss, which is helpful with short, succinct documents (such as CACM abstracts), but less so with longer texts; see also (Grishman et al., 1986).

final search query more comprehensive (adding synonyms) and/or more pointed (adding specializations).[12] It follows that not all similarity relations will be equally useful in query expansion, for instance, complementary relations like the one between *algol* and *fortran* may actually harm system's performance, since we may end up retrieving many irrelevant documents. Similarly, the effectiveness of a query containing *fortran* is likely to diminish if we add a similar but far more general term such as *language*. On the other hand, database search is likely to miss relevant documents if we overlook the fact that *fortran* is a *programming language*, or that *interpolate* is a specification of *approximate*. We noted that an average set of similarities generated from a text corpus contains about as many "good" relations (synonymy, specialization) as "bad" relations (antonymy, complementation, generalization), as seen from the query expansion viewpoint. Therefore any attempt to separate these two classes and to increase the proportion of "good" relations should result in improved retrieval. This has indeed been confirmed in our experiments where a relatively crude filter has visibly increased retrieval precision.

In order to create an appropriate filter, we expanded the IC function into a global specificity measure called the *cumulative informational contribution function* (ICW). ICW is calculated for each term across all contexts in which it occurs. The general philosophy here is that a more specific word/phrase would have a more limited use, i.e., would appear in fewer *distinct* contexts. ICW is similar to the standard *inverted document frequency (idf)* measure except that term frequency is measured over syntactic units rather than document size units.[13] Terms with higher ICW values are generally considered more specific, but the specificity comparison is only meaningful for terms which are already known to be similar. The new function is calculated according to the following formula:

$$ICW(w) = \begin{cases} IC_L(w) * IC_R(w) & \textit{if both exist} \\ IC_R(w) & \textit{if only } IC_R(w) \textit{ exists} \\ 0 & \textit{otherwise} \end{cases}$$

where (with n_w, $d_w > 0$):[14]

$$IC_L(w) = IC([w,\_]) = \frac{n_w}{d_w(n_w+d_w-1)}$$

$$IC_R(w) = IC([\_,w]) = \frac{n_w}{d_w(n_w+d_w-1)}$$

For any two terms w_1 and w_2, and a constant $\delta > 1$, if $ICW(w_2) \geq \delta * ICW(w_1)$ then w_2 is considered more specific than w_1. In addition, if $SIM_{norm}(w_1,w_2) = \sigma > \theta$, where θ is an empirically established threshold, then w_2 can be added to the query containing term w_1 with weight σ.[14] In the CACM-3204 collection:

$ICW(algol)$	$= 0.0020923$
$ICW(language)$	$= 0.0000145$
$ICW(approximate)$	$= 0.0000218$
$ICW(interpolate)$	$= 0.0042410$

Therefore *interpolate* can be used to specialize *approximate*, while *language* cannot be used to expand *algol*. Note that if δ is well chosen (we used δ=10), then the above filter will also help to reject antonymous and complementary relations, such as $SIM_{norm}(pl\_i,cobol)$=0.685 with $ICW(pl\_i)$=0.0175 and $ICW(cobol)$=0.0289. We continue working to develop more effective filters. Examples of filtered similarity relations obtained from CACM-3204 corpus (and their sim values): *abstract graphical 0.612; approximate interpolate 0.655; linear ordinary 0.743; program translate 0.596; storage buffer 0.622*. Some (apparent?) failures: *active digital 0.633; efficient new 0.580; gamma beta 0.720*. More similarities are listed in Table 2.

SUMMARY OF RESULTS

The preliminary series of experiments with the CACM-3204 collection of computer science abstracts showed a consistent improvement in performance: the average precision increased from 32.8% to 37.1% (a 13% increase), while the normalized recall went from 74.3% to 84.5% (a 14% increase), in comparison with the statistics of the base NIST system. This improvement is a combined effect of the new stemmer, compound terms, term selection in queries, and query expansion using filtered similarity relations. The choice of similarity relation filter has been found critical in improving retrieval precision through query expansion. It should also be pointed out that only about 1.5% of all similarity relations originally generated from CACM-3204 were found

[12] Query expansion (in the sense considered here, though not quite in the same way) has been used in information retrieval research before (eg. Sparck Jones and Tait, 1984; Harman, 1988), usually with mixed results. An alternative is to use term clusters to create new terms, "metaterms", and use them to index the database instead (eg. Crouch, 1988; Lewis and Croft, 1990). We found that the query expansion approach gives the system more flexibility, for instance, by making room for hypertext-style topic exploration via user feedback.

[13] We believe that measuring term specificity over document-size contexts (eg. Sparck Jones, 1972) may not be appropriate in this case. In particular, syntax-based contexts allow for processing texts without any internal document structure.

[14] The filter was most effective at $\sigma = 0.57$.

word1	word2	SIMnorm
**aim*	*purpose*	0.434
algorithm	*method*	0.529
**adjacency*	*pair*	0.499
**algebraic*	*symbol*	0.514
**american*	*standard*	0.719
assert	*infer*	0.783
**buddy*	*time-share*	0.622
committee	**symposium*	0.469
critical	*final*	0.680
best-fit	*first-fit*	0.871
**duplex*	*reliable*	0.437
earlier	*previous*	0.550
encase	*minimum-area*	0.991
give	*present*	0.458
incomplete	*miss*	0.850
lead	**trail*	0.890
mean	**standard*	0.634
method	*technique*	0.571
memory	*storage*	0.613
match	*recognize*	0.563
lower	*upper*	0.841
progress	**trend*	0.444
range	*variety*	0.600
round-off	*truncate*	0.918
remote	*teletype*	0.509

Table 2. Filtered word similarities (* indicates the more specific term).

admissible after filtering, contributing only 1.2 expansion on average per query. It is quite evident significantly larger corpora are required to produce more dramatic results.[15] [16] A detailed summary is given in Table 3 below.

These results, while quite modest by IR standards, are significant for another reason as well. They were obtained without any manual intervention into the database or queries, and without using any other information about the database except for the text of the documents (i.e., not even the hand generated keyword fields enclosed with most documents were used). Lewis and Croft (1990), and Croft et al. (1991) report results similar to ours but they take advantage of Computer Reviews categories manually assigned to some documents. The purpose of this research is to explore the potential of automated NLP in dealing with large scale IR problems, and not necessarily to obtain the best possible results on any particular data collection. One of our goals is to point a feasible direction for integrating NLP into the traditional IR.

[15] KL Kwok (private communication) has suggested that the low percentage of admissible relations might be similar to the phenomenon of 'tight clusters' which while meaningful are so few that their impact is small.

[16] A sufficiently large text corpus is 20 million words or more. This has been partially confirmed by experiments performed at the University of Massachussetts (B. Croft, private communication).

Tests	*base*	*suff.trim*	*query exp.*
Recall	*Precision*		
0.00	0.764	0.775	0.793
0.10	0.674	0.688	0.700
0.20	0.547	0.547	0.573
0.30	0.449	0.479	0.486
0.40	0.387	0.421	0.421
0.50	0.329	0.356	0.372
0.60	0.273	0.280	0.304
0.70	0.198	0.222	0.226
0.80	0.146	0.170	0.174
0.90	0.093	0.112	0.114
1.00	0.079	0.087	0.090
Avg. Prec.	0.328	0.356	0.371
% change		8.3	13.1
Norm Rec.	0.743	0.841	0.842
Queries	50	50	50

Table 3. Recall/precision statistics for CACM-3204

ACKNOWLEDGEMENTS

We would like to thank Donna Harman of NIST for making her IR system available to us. We would also like to thank Ralph Weischedel, Marie Meteer and Heidi Fox of BBN for providing and assisting in the use of the part of speech tagger. KL Kwok has offered many helpful comments on an earlier draft of this paper. In addition, ACM has generously provided us with text data from the Computer Library database distributed by Ziff Communications Inc. This paper is based upon work supported by the Defense Advanced Research Project Agency under Contract N00014-90-J-1851 from the Office of Naval Research, the National Science Foundation under Grant IRI-89-02304, and a grant from the Swiss

National Foundation for Scientific Research. We also acknowledge a support from Canadian Institute for Robotics and Intelligent Systems (IRIS).

REFERENCES

Church, Kenneth Ward and Hanks, Patrick. 1990. "Word association norms, mutual information, and lexicography." *Computational Linguistics*, 16(1), MIT Press, pp. 22-29.

Croft, W. Bruce, Howard R. Turtle, and David D. Lewis. 1991. "The Use of Phrases and Structured Queries in Information Retrieval." Proceedings of ACM SIGIR-91, pp. 32-45.

Crouch, Carolyn J. 1988. "A cluster-based approach to thesaurus construction." Proceedings of ACM SIGIR-88, pp. 309-320.

Fagan, Joel L. 1987. *Experiments in Automated Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods.* Ph.D. Thesis, Department of Computer Science, Cornell University.

Grishman, Ralph, Lynette Hirschman, and Ngo T. Nhan. 1986. "Discovery procedures for sublanguage selectional patterns: initial experiments". *Computational Linguistics*, 12(3), pp. 205-215.

Grishman, Ralph and Tomek Strzalkowski. 1991. "Information Retrieval and Natural Language Processing." Position paper at the workshop on Future Directions in Natural Language Processing in Information Retrieval, Chicago.

Harman, Donna. 1988. "Towards interactive query expansion." Proceedings of ACM SIGIR-88, pp. 321-331.

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.

Harris, Zelig S. 1991. *A Theory of language and Information. A Mathematical Approach.* Cladendon Press. Oxford.

Harris, Zelig S. 1982. *A Grammar of English on Mathematical Principles.* Wiley.

Harris, Zelig S. 1968. *Mathematical Structures of Language.* Wiley.

Hindle, Donald. 1990. "Noun classification from predicate-argument structures." Proc. 28 Meeting of the ACL, Pittsburgh, PA, pp. 268-275.

Lewis, David D. and W. Bruce Croft. 1990. "Term Clustering of Syntactic Phrases". Proceedings of ACM SIGIR-90, pp. 385-405.

Mauldin, Michael. 1991. "Retrieval Performance in Ferret: A Conceptual Information Retrieval System." Proceedings of ACM SIGIR-91, pp. 347-355.

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

Salton, Gerard. 1989. *Automatic Text Processing: the transformation, analysis, and retrieval of information by computer.* Addison-Wesley, Reading, MA.

Shannon, C. E. 1948. "A mathematical theory of communication." Bell System Technical Journal, vol. 27, July-October.

Smeaton, A. F. and C. J. van Rijsbergen. 1988. "Experiments on incorporating syntactic processing of user queries into a document retrieval strategy." Proceedings of ACM SIGIR-88, pp. 31-51.

Sparck Jones, Karen. 1972. "Statistical interpretation of term specificity and its application in retrieval." *Journal of Documentation*, 28(1), pp. 11-20.

Sparck Jones, K. and E. O. Barber. 1971. "What makes automatic keyword classification effective?" *Journal of the American Society for Information Science*, May-June, pp. 166-175.

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

Strzalkowski, Tomek and Barbara Vauthey. 1991. "Fast Text Processing for Information Retrieval." Proceedings of the 4th DARPA Speech and Natural Language Workshop, Morgan-Kaufman, pp. 346-351.

Strzalkowski, Tomek and Barbara Vauthey. 1991. "Natural Language Processing in Automated Information Retrieval." Proteus Project Memo #42, Courant Institute of Mathematical Science, New York University.

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.

Wilks, Yorick A., Dan Fass, Cheng-Ming Guo, James E. McDonald, Tony Plate, and Brian M. Slator. 1990. "Providing machine tractable dictionary tools." *Machine Translation*, 5, pp. 99-154.