Corpus Analysis for TREC 5 Query Expansion

Susan Gauch and Jianying Wang
{sgauch,jwang}@eecs.ukans.edu
Electrical Engineering and Computer Science
University of Kansas

ABSTRACT

Accessing online information remains an inexact science. While valuable information can be found, typically many irrelevant documents are also retrieved and many relevant ones are missed. Terminology mismatches between the user's query and document contents is a main cause of retrieval failures. Expanding a user's query with related words can improve search performance, but the problem of identifying related words remains.

This research uses corpus linguistics techniques to automatically discover word similarities directly from the contents of the untagged TREC database and to incorporates that information in the SMART information retrieval system. The similarities are calculated based on the contexts in which a set of target words appear. Using these similarities, user queries are automatically expanded, resulting in conceptual retrieval rather than requiring exact word matches between queries and documents.

1. INTRODUCTION

Expanding a user's query with related terms can improve search performance. Relevance feedback systems, where related terms come from the contents of user-identified relevant documents, have been shown to be quite effective (Harman 1992). Our earlier work showed that an expert system which automatically reformulated Boolean queries by including terms from an online thesaurus was able to improve search results (Gauch and Smith 1991; Gauch and Smith 1993) without requiring relevance judgments from the user. Some systems (Anick, Brennan et al. 1990) present related terms to the user and allow them to selectively augment the query. However, the latter two approaches require the presence of an online thesaurus whose words closely match the contents of the database.

Where can such a thesaurus come from? In some cases, it is hand-built (Gauch and Smith 1991), a time-consuming and ad hoc process. In other cases, the thesaurus is an online version of a published thesaurus or semantically coded dictionary (Liddy and Myaeng 1993). However, an online published thesaurus or dictionary will have serious coverage gaps if used for technical domains which have their own distinct sublanguages. Because of ambiguity, this type of thesaurus may also be difficult to use with a database of general English documents because they show all possible classifications for a word when only one or a few senses may be actually present in the database.

Our system automatically discovers related words directly from the contents of a textual database and incorporates that information in the modified SMART information retrieval system. We modified and applied one particular technique from the field of corpus linguistics which seemed particularly well-suited for this task. HNC's MatchPlus system (Gallant, Hecht-Nielsen et al. 1993) has a similar approach, however, they use neural networks to identify features which are used to index documents rather than using the words themselves. In contrast, we index documents by their words and identify related words which can be used for query expansion. With our approach, it is possible to provide query expansion on top of pre-indexed collections.

2. SYSTEM ARCHITECTURE

To incorporate the results of the corpus analysis into an existing retrieval engine, SMART was modified to allow it to expand queries based on the similarity matrices, search the database with the expanded queries, and return the top 1000 documents for each query.

We has modified smart.11.0 at the point when a query is passed to the search engine. At this point, the query words are expanded using different techniques according to the word similarity weights generated from the corpus analysis program. If the databases are treated separately, an algorithm is used to select the appropriate similarity matrix to use to expand each query.

We participated in Adhoc category A, which is evaluated based on the combined collection consisting of the documents on both Tipster Disk 2 and TREC Disk 4. The whole database is 2.1 GB in size and contains 524929 documents from 7 different sources (AP, CR, FR88, FR94, FT, WSJ, and ZIFF). Indexing the database as a whole took approximately 5 hours on a shared Sun SPARC center 2000. The retrieval runs on the combined collection used SMART's "lnc" weights for the documents and the modified "ltc" weights for the queries.

3. CORPUS LINGUISTICS TECHNIQUE

Methods that work with entirely untagged corpora have recently been developed which show great promise (Brill and Marcus 1992; Finch and Chater 1992; Hearst, 1992, Myaeng and Li 1992; Schütze 1992). Using a much more fine-grained approach than traditional automatic thesaurus construction techniques, word-word similarities are automatically calculated based on the premise that words which occur in similar contexts are similar. These techniques are particularly useful for specialized text with specialized vocabularies and word-use, for which there are no adequate online dictionaries. They are also appropriate for general English corpora since a general online dictionary may show many senses for a common word where only one or a few actually are used in a given corpus.

We have modified a corpus linguistics approach (Finch and Chater 1992) that takes into account both the relative positions of the nearby context words as well as the mutual information (Church and Hanks 1990) associated with the occurrence of a particular context word. We have applied this to a 15% sample of the TREC5 database to calculate *a priori* the similarities of a subset of the words in the database, called the target words.

3.1 Similarity Calculation

Similar to (Finch and Chater 1992) , the context vector for a word (the *target word*) is a concatenation of the vectors describing the words observed in the preceding two positions and the following two positions (the *context positions*). Each position is represented by a vector corresponding to the occurrence of the 200 highest frequency words in the corpus (the *context words*), giving a 800-dimensional vector describing the context. Initially, the counts from all instances of a word form w_i are summed so that the entry in the corresponding context word position in the vector is the sum of the occurrences of that context word in that position for the corresponding target word form; it is the joint frequency of the context word.

Consider an example in which there are only five context words, {"a", "black", "dog", "the, "very"} and two sentences containing the target word "dog:

(1) The black <u>dog</u> barked very loudly.

(2) A brown <u>dog</u> barked very loudly.

Sentence	Context Position	Observed Word	Context Vector
1	-2	"The"	(0, 0, 0, 1, 0) 4th context word
	-1	"black"	(0, 1, 0, 0, 0) 2nd context word
	+1	"barked"	(0, 0, 0, 0, 0) not a context word
	+2	"very"	(0, 0, 0, 0, 1) 5th context word

Table 1. The context vectors for each of the 4 context positions around the occurrence of the target word "dog" in sentence 1.

The context vector for "dog" in sentence 1 is formed by concatenating the context vectors for each of the 4 context positions:

(0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)

Similarly, the context vector for "dog" in sentence 2 would be:

(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)

and the combined vector for the word "dog" would be:

(1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2)

Using 150 context words and window size 5, 600-dimensional context vectors are created. Subsequently, 600-dimensional vectors of mutual information values, *MI*, are computed from the frequencies as follows,

$$MI(cw) = \log_2\left(\frac{Nf_{cw}}{f_c f_w} + 1\right)$$

This expresses the mutual information value for the context word c appearing with the target word w. The mutual information is large whenever a context word appears at a much higher frequency, f_{cw}, in the neighborhood of a target word than would be predicted from the overall frequencies in the corpus, f_c and f_w. The formula adds 1 to the frequency ratio, so that a 0 (zero) occurrence corresponds to 0 mutual information. When the mutual information vectors are computed for a number of words, they can be compared to see which words have similar contexts. The comparison we chose is the inner product, or cosine measure, which can vary between -1.0 and +1.0 (Myaeng and Li 1992) .

Finally, to make the identification of the most highly similar terms to a given term more efficient, an auxiliary file is produced *a priori* from the similarity matrix. It stores, for each target word, the words and similarity values for all words with similarity above a given threshold. This is called the similarity lists.

3.2 Preprocessing the Database

For use by the corpus analysis program, the TREC5 databases are sampled randomly and uniformly to get a representative sample (roughly 15%) for each database. Then, the sample database is preprocessed. During preprocessing, the required fields of a document are extracted, tokenized and downcased, and some tokens are treated specially (e.g., U.S. --> usa). Then, a sentence tagging algorithm is used to identify sentence breaks. Finally, a word-frequency list for the sample database is generated and sorted by the frequency. Based on this frequency list, target words and context words are automatically selected. Based on earlier experiments, 6000 target words are selected by extending up 500 and down 1500 words from the 4000 word window which captures the maximum number of non-stopped query words. Since only words in the target words will be expanded, any missing important query words are added to the target word file.

The full database goes through a subset of the above steps (downcasing, token replacement) before being indexed by SMART.

4. EXPERIMENTS

4.1 Tuning the calculation on TREC4 category B collection

To tune the corpus analysis program, we ran a series of experiments with TREC4 queries. There are five main parameters which can be adjusted: the sample database size, the list of target words (position and size in the frequency list), the list of context words, the window size (the window around target words that is used to characterize the contexts in which they appear), and the similarity threshold that is used during query expansion. First, we fixed the list of context words (200 highest frequent words in the sample database), the number of target words (4000), and the window size (7). Then, we experimented by shifting the 4000 target words along the frequency list for different sizes of the sample database. We found that the 4000 target words list position that yielded the best 11 point average occurred with a constant ratio of (frequency of first word in target list in sample/(frequency of most frequent word in sample).

sample size	10%	20%
most frequent word	417969	796667
most frequent target word	120	241
(best threshold, 11-pt avg)	(0.34, 0.1839)	(0.45, 0.1840)
ratio	120/417969	241/796667
	0.0002871	0.0003025
average	(0.0002871 + 0.0003025)/2 = 0.000295	

Table 2. Calculate the starting frequency as percentage of the most frequency (Note: unexpanded queries yield 11-pt avg = 0.1791)

More experiments using the following equation:

starting-frequency = most-frequency * 0.000295

to select 4000 target words were conducted to confirm the above relationship. Table 3 shows the pairs of (similarity threshold, 11-pt avg) for some of the experiments shifting the 4000 target words along the frequency list after using the frequency equation to get the initial siting of the 4000 target words. Note, shifting up N words means shifting the target word window N words more frequent (i.e., selecting N more frequent target words and dropping N less frequent target words). Shifting down N words means selecting less frequent target words. This confirmed that the initial area selected by the ratio performed best. For the following table, 200 context words, a context window of 7 and a 10% sample size were used.

Target-shift	(threshold, 11 pt avg)
1000 up	(0.80, 0.1791)
frequency equation	(0.38, 0.1887)
1000 down	(0.38, 0.1807)
2000 down	(0.27, 0.1818)
3000 down	(0.27, 0.1813)
4000 down	(0.32, 0.1800)

Table 3. Sample size 20% (Note: unexpanded queries yield 11-pt avg = 0.1791)

We evaluated another approach for choosing 4000 target words by selecting the area of the frequency list that contained the maximum number of non-stopped query words. We found that the 4000 target words selected in this manner tended to be shifted up (i.e., more frequent) about 1200 positions relative to the ratio method. Because it was independent of sample size, we decided to use the query words to site the target list, but to shift down 1200 to more closely match the ratio results.

Once the target word list was decided, we did a series of experiments with the list of context words, the window size and the sample database size. From the results in Table 4, we know the parameters with window size 7, context words 200 (the most frequent 200 words) provide better performance. However, the best sample size is not clear, although, in general, the 20% sample size seems to produce the best results.

Sample size 10%			
Context\Window	**5**	**7**	**9**
150	(0.39, 0.1823)	(0.37, 0.1834)	(0.30, 0.1830)
200	(0.39, 0.1825	(0.34, 0.1831)	(0.27, 0.1815)
250	(0.37, 0.1850)	(0.30, 0.1830)	(0.27, 0.1826)
Sample size 20%			
Context\Window	**5**	**7**	**9**
150	(0.50, 0.1813)	(0.45, 0.1840)	(0.40, 0.1860)
200	(0.45, 0.1834)	(0.38, 0.1887)	(0.34, 0.1873)
250	(0.45, 0.1811)	(0.38, 0.1852)	(0.32, 0.1861)
Sample size 30%			
Context\Window	**5**	**7**	**9**
150	(0.80, 0.1791)	(0.70, 0.1791)	(0.60, 0.1792)
200	(0.70, 0.1791)	(0.60, 0.1796)	(0.40, 0.1804)
250	(0.60, 0.1799)	(0.45, 0.1795)	(0.45, 0.1799)

Table 4. Experimental results for context words, window size and sample size. (Note: unexpanded queries yield 11-pt avg = 0.1791)

4.2 Query Expansion Techniques

If we use all entries in the similarity matrix above a given threshold to expand queries, query words like "identify" and "create" with many strong matches will be greatly expanded while lower frequency query words (usually important query words) don't get expanded at all. Another problem with using the similarity threshold to control the expansion is that minor changes in the similarity threshold can result in large differences in the number of expansion words for certain query words. So, we experimented with an expansion technique based on using a fixed number of expansion words for each query word appearing in the similarity matrix. Table 5 shows the pairs of (threshold, 11-pt Avg) or (max. expanded, 11-pt Avg) for the same similarity matrices with 20% sample size, 200 context words, and 7 window size:

target-words\method	Similarity threshold	max expanded
4000	(0.38, 0.1887)	(1, 0.1802)
6000	(0.80, 0.1791)	(3, 0.1849)
7000	(0.80, 0.1791)	(1, 0.1836)
8000	(0.80, 0.1791)	(3, 0.1834)
9000	(0.80, 0.1791)	(3, 0.1851)

Table 5. Experiments for the size of target words and expanding techniques. (Note: unexpanded queries yield 11-pt avg = 0.1791).

In this experiment, we also evaluated using more target words. In each of the cases, we expand the original 4000 target words by including 1000 more frequent words and then as many less frequent words to make up the total. For example, the list of 7000 target words is extended from the 4000 target words by adding 1000 more frequent and 2000 less frequent words. Although the best overall performance was provided by the threshold expansion technique, it didn't seem robust. For all of the larger target word lists, expansion based on the threshold led to a degradation in the 11-pt average. However, using a fixed number of expansion words seemed much less sensitive. In particular, adding the three most similar words for each query word seemed to consistently improve performance. With this technique, 6000 target words seemed a reasonable compromise between search quality results and time and space complexity.

4.3 Matrix selection for query expansion

Since the whole collection has seven databases (AP, CR, FR88, FR94, FT, WSJ, and ZIFF) for category A, seven similarity matrices are generated, one for each database. We chose to do this rather than using one big similarity matrix for the whole collection because each of the databases has a different domain of interest, resulting in different word usage. Our experiments for TREC4 also confirm that analyzing the databases separately performs better than treating the corpus as one big database. However, it is very important to select the correct matrix for each query. To do this, for each query, we calculate the frequency of each non-stopped query word normalized by the most frequent word in the database over all 7 databases. Then, we sum the normalized frequencies together for all query words. We expand this query using the similarity matrix built from the database which maximizes this sum.

4.4 TREC5 Results

Based on our experiments, we chose a sample size of 15% for TREC5, the corpus is treated as seven separate databases producing seven separate similarity matrices (200 context words, 6000 target words, window size of

7). The queries are expanded using a combination of the threshold and fixed number of words as follows: A maximum of three words are added per query word. However, only words above a certain threshold will be added (0.28 in KUSG2; 0.20 in KUSG3). The following table summarizes our results compared to the entries with short topics in the ad hoc category. Unfortunately, our expanded queries actually performed worse than the unexpanded queries. This was naturally quite disappointing and we have begun trying out different query expansion techniques. One, KU-new, is presented here. It expands only one word for each concept, and only if the similarity value exceeds 0.28. This does, indeed result in improvement, albeit minor.

	No exp.	**KUSG2**	**KUSG3**	**KU-new**
Precision (20 docs)	0.1980	0.1900	0.1860	0.1950
Precision (100 docs)	0.1276	0.1260	0.1246	0.1290
Non-interp. 11 pt avg	0.1100	0.1073	0.1056	0.1127

Table 6. TREC5, category A results.

4.5 Discussion

Our extensive experiments with the TREC4 corpus leads us to believe that our approach can, indeed, lead to improvements in the 11-pt average of greater than 5%. However, we clearly need to come up with a systematic way of setting the parameters for a new database and/or training on a new database in order to achieve performance improvements. This is the focus of our current work.

BIBLIOGRAPHY

Anick, P.G., Brennan, J.D., Flynn, R.A., Hanssen, D.R., Alvey, B., and Robbins, J.M. (1990). A Direct Manipulation Interface for Boolean Information Retrieval via Natural Language Query. Proc. 13th Ann. International ACM SIGIR Conf., (pp. 135-150). Brussels, Belgium: ACM Press.

Brill, E., & Marcus, M. (1992). Tagging an Unfamiliar Text with Minimal Human Supervision. In AAAI Fall Symposium Series: Probabilistic Approaches to Natural Language (Working Notes), (pp. 10-16). Cambridge, MA.

Church, K. W., & Hanks, P. (1990). Word Association Norms, Mutual Information and Lexicography. Computational Linguistics, 16(1), 22-29.

Finch, S., & Chater, N. (1992). Bootstrapping Syntactic Categories Using Statistical Methods. In W. Daelemans & D. Powers (Ed.), Proc. 1st SHOE Workshop, (pp. 229-235). Tilburg U., The Netherlands.

Gallant, S.I., Hecht-Nielsen, R., Caid, W., Qing, K., Carleton, J., and Sudbeck, C.D. (1993). HNC's MatchPlus System, 1st Text Retrieval Conf. (TREC-1), (pp. 107-111). NIST #500-207.

Gauch, S., & Smith, J.B. (1993). An Expert System for Automatic Query Reformulation. J. of the Amer. Society of Inf. Sci., 44 (3), 124-136.

Gauch, S., & Smith, J.B. (1991). Search Improvement via Automatic Query Reformulation. ACM Trans. on Information Systems, 9 (3), 249-280.

Harman, D. (1992). Relevance Feedback Revisited. Proc. 15th Ann. International ACM SIGIR Conf., (pp. 1-10). Copenhagen, Denmark: ACM Press.

Hearst, M.A. (1992) "Automatic Acquisition of Hyponyms from Large Text Corpora," Proc. 14th Intern'l Conf. on Computational Linguistics, Nantes, France, July.

Liddy, E.D., & Myaeng, S.H. (1993). DR-LINK's Linguistic-Conceptual Approach to Document Detection, 1st Text Retrieval Conf. (TREC-1), (pp. 113-129). NIST #500-207.

Myaeng, S. H., & Li, M. (1992). Building Term Clusters by Acquiring Lexical Semantics from a Corpus. In Y. Yesha (Ed.), CIKM-92, (pp. 130-137). Baltimore, MD: ISMM.

Schütze, H. (1992). Context Space. In AAAI Fall Symposium Series: Probabilistic Approaches to Natural Language (Working Notes), (pp. 113-120). Cambridge, MA:

Chris Buckley (1985). Implementation of the SMART information retrieval system. Technical Report 85-686, Computer Science Department, Cornell University, Ithaca, New York.

Gauch, S. & Chong, M. (1995) Automatic Word Similarity Detection for TREC4 Query Expansion