Improving the Effectiveness of Informational Retrieval with Local Context Analysis[*]

Jinxi Xu and W. Bruce Croft
Computer Science Department
University of Massachusetts, Amherst, MA 01003
{xu, croft}@cs.umass.edu

Abstract

Techniques for automatic query expansion have been extensively studied in information retrieval research as a means of addressing the word mismatch between queries and documents. These techniques can categorized as either global or local. While global techniques rely on analysis of a whole collection to discover word relationships, local techniques emphasize analysis of the top ranked documents retrieved for a query. Both types of techniques have advantages and limitations. In this paper we propose a new technique, called *local context analysis*, which combines the advantages of a global technique called *Phrasefinder* and a local technique known as *local feedback*. Experiments on a number of collections, both English and non-English, show that local context analysis offers more effective and consistent retrieval results.

Categories and Subject Descriptors: H.3.1. [**Information Storage and Retrieval**]: Content Analysis and Indexing – *indexing methods; thesauruses; linguistic processing*; H.3.3. [**Information Storage and Retrieval**]: Information Search and Retrieval – *query formulation; search process; relevance feedback*
General terms: Experimentation, Performance
Additional Key Words and Phrases: information retrieval, document analysis, global techniques, local techniques, local context analysis, feedback, co-occurrence

1 Introduction

A fundamental problem in information retrieval (IR) is word mismatch, which refers to the phenomenon that the users of IR systems often use different words to describe the concepts in their queries than the authors use to describe the same concepts in their documents. Word mismatch is a serious problem, as observed by Furnas, et al, in a more general context[14]. In their experiments, two people use the same term to describe an object less than 20% of the time. The problem is more severe for short casual queries than for long elaborate queries because as queries get longer, there is more chance of some important words co-occurring in the query and the relevant documents. Unfortunately, short queries

[*] This work is partially based on chapters of the PhD thesis of the first author. Some results also appeared in a SIGIR 96 paper by the authors.

are becoming increasingly common in retrieval applications, especially with the advent of the World Wide Web. Addressing the word mismatch problem has become an increasingly important research topic in IR.

In this paper, we will discuss techniques that address the word mismatch problem through automatic query expansion. Automatic query expansion techniques have a significant advantage over manual techniques such as relevance feedback [26] and manual thesauri because they require no effort on the part of the user. Existing techniques for automatic query expansion can be categorized as either *global* or *local*. Global techniques aim to discover corpus-wide word relationships based on co-occurrence analysis of a whole collection. One of the earliest global techniques is term clustering [30], which groups words into clusters based on their co-occurrences and uses the clusters for query expansion. Other well-known global techniques include Latent Semantic Indexing [12], similarity thesauri [23] and Phrasefinder [18]. Generally speaking, global techniques did not show consistent positive retrieval results until better strategies for term selection were introduced in recent years. Global techniques typically need the co-occurrence information for every pair of terms. This is a computationally demanding task for large collections.

Local techniques expand a query based on the information in the set of top ranked documents retrieved for the query [1; 11; 5]. The simplest local technique is local feedback [5], which assumes the top retrieved documents are relevant and uses standard relevance feedback procedures for query expansion. A similar and earlier technique was proposed in [11], where information from the top ranked documents is used to re-estimate the probabilities of query terms in the relevant set for a query. The terms chosen by local feedback for query expansion are typically the most frequent terms (excluding stop words) from the top ranked documents. Recent TREC results show that local feedback can significantly improve retrieval effectiveness [5; 6]. But local feedback is not a robust technique. It can seriously degrade retrieval performance if few of the ranked documents retrieved for the original query are relevant, because in this case the expansion terms are mostly from non-relevant documents.

In this paper, we propose a new query expansion technique, called *local context analysis*, which is a combination of global and local techniques. Like global techniques, local context analysis selects expansion features based on their co-occurrences with the query terms. Like local techniques, it selects expansion features from the top retrieved documents for a query. The expansion features are normally nouns and noun phrases and are called *concepts*. Local context analysis ranks the concepts by their co-occurrence with the query terms in the top ranked documents and uses the highest ranked concepts for query expansion. Experimental results show that this combination of local and global techniques results in more effective and more robust query expansion.

The remainder of the paper is organized as follows: Sections 2 and 3 review existing techniques and point out the problems. Section 4 explains the motivations behind the combination of global and local techniques and how it leads to local context analysis. Section 5 outlines the experimental methodology and describes the test collections. Sections 6 to 13 present the experimental results. Section 14 discusses optimization and efficiency issues. Section 15 discusses other applications of local context analysis in IR. Section 16 draws conclusions and points out future work.

2 Global Techniques

2.1 Term Clustering

Global techniques for query expansion are typically based on the so-called association hypothesis, which states that words related in a corpus tend to co-occur in the documents of that corpus [31]. One of the earliest global techniques is term clustering, which groups related terms into clusters based on their co-occurrences in a corpus. The most representative work on term clustering was conducted by Sparck Jones in the late 60's and early 70's [30; 29]. She investigated a wide range of algorithms to form clusters and different methods to use them. Her major conclusion was that well-constructed term clusters can improve retrieval performance, but it was not supported by some follow-up studies [19]. A serious problem with term clustering is that it cannot handle ambiguous terms. If a query term has several meanings, term clustering will add terms related to different meanings of the term and make the query even more ambiguous. In this case, it will lower retrieval effectiveness.

2.2 Dimensionality Reduction

Related to term clustering are the dimensionality reduction techniques. The most well-known technique of this type is Latent Semantic Indexing (LSI) [12; 15]. Other dimensionality reduction techniques were proposed in a number of studies [7; 27]. LSI decomposes a term into a vector in a low dimensional space. This is achieved using a technique called singular value decomposition (SVD). It is hoped that related terms which are orthogonal in the high dimensional space will have similar representations in the low dimensional space and as a result, retrieval based on the reduced representations will be more effective. Despite the potential claimed by its advocates, retrieval results using LSI so far have not shown to be conclusively better than those of standard vector space retrieval systems. As with term clustering, word ambiguity is also a problem with dimensionality reduction techniques. If a query term is ambiguous, terms related to different meanings of the term will have similar reduced representations. This is equivalent to adding unrelated terms to the query.

2.3 Phrasefinder

More recent global techniques address the word ambiguity problem by expanding a query as a whole. Examples are Phrasefinder [18] and Similarity Thesauri [23]. These techniques exploit the mutual disambiguation of the query terms by selecting expansion terms based on their co-occurrences with all query terms. Terms co-occurring with many query terms are preferred over terms co-occurring with few query terms.

We describe Phrasefinder in more detail since it represents one of the most successful global techniques and is similar to the new query expansion technique proposed in this paper. With Phrasefinder, a concept c (usually a noun phrase) is represented as a set of tuples $\{< t_1, a_1 >, < t_2, a_2 >, ...\}$, where t_i is a term co-occurring with c and a_i is the number of co-occurrences between c and t_i. The set of tuples is called the pseudo-document of concept c. Given a query Q, the pseudo-documents of all concepts are ranked against Q as if they are real documents. The highest ranked concepts are used for query expansion. As with any global technique, efficiency is a problem. The creation of the pseudo documents requires the co-occurrence data for all possible concept-term pairs. The retrieval effectiveness of Phrasefinder is mixed. It is one of best global techniques but judging from recent published results, it is not as effective as some local techniques [33].

3 Local Techniques

The idea of using the top ranked documents for query expansion can be traced at least to a 1977 paper by Attar and Fraenkel [1]. In that paper, term clustering was carried out by running traditional term clustering algorithms on the top ranked documents retrieved for a query. The term clusters were then used for query expansion. Attar and Fraenkel produced positive improvement in retrieval effectiveness, but the test collection they used is too small to draw any definite conclusion.

A more recent local technique is local feedback (also known as pseudo feedback). Local feedback is motivated by relevance feedback [26], a well-known IR technique to modify a query based on the relevance judgments of the retrieved documents. Relevance feedback typically adds common terms from the relevant documents to a query and re-weights the expanded query based on term frequencies in the relevant documents and in the non-relevant documents. Local feedback mimics relevance feedback by assuming the top ranked documents to be relevant. The added terms are therefore common terms from the top ranked documents. A similar and earlier technique proposed by Croft and Harper modifies the weights of the query terms but does not add new terms to a query [11]. Recent TREC results show that local feedback is a simple yet effective query expansion technique. But its performance is very erratic: It can seriously reduce the effectiveness of a few queries although the average performance over a large number of queries is improved. The queries hurt by local feedback are usually those which retrieve few relevant documents in the initial retrieval, because in this case the expansion terms are mostly from the non-relevant documents.

4 Local Context Analysis

Our objective is to solve the problems with existing query expansion techniques, particularly Phrasefinder and local feedback. These two techniques complement each other and their combination results in a new technique which is more effective than both. The technique is *local context analysis*. We will first explain from two perspectives why it makes sense to combine Phrasefinder and local feedback. Then we will describe the algorithm and the implementation of local context analysis in detail.

4.1 Local Information Helps Phrasefinder

Phrasefinder is inefficient because it needs to analyze a whole corpus. One way to make it more efficient is to analyze the top ranked documents instead of the whole corpus. Since the top ranked documents are rich in query terms, they shall give us a fairly good estimate of the co-occurrence patterns of the query terms with the concepts. In other words, co-occurrence information in the top ranked documents is a "cheap" and potentially reliable estimate of corpus-wide co-occurrence information. Although using co-occurrence information in top ranked documents requires an extra search for a query, it is still preferable to whole corpus analysis from an efficiency point of view.

We believe that using the top ranked documents not only makes Phrasefinder more efficient, but also makes it more effective. The goal of query expansion is to find representative words for describing the relevant documents. The best we can do without explicit relevance judgments is to select expansion features from a sample of documents which are likely to be relevant to the query. Naturally the sample should consist of the top ranked documents for a query. The higher the concentration of relevant documents in the sample, the more likely we will find good features. But the sample of documents used by Phrasefinder (in fact all global techniques) to expand a query has only a small proportion of relevant documents. The sample consists of all the documents that match at least one query term. For nearly any query, this sample is much larger than the total number of relevant documents. The chance of finding good expansion features from such a large set of documents most of which are non-relevant to a given query would be smaller than that of using the top ranked documents. The argument is supported by our experience in using Phrasefinder, as we know that a major source of spurious expansion concepts from Phrasefinder are the large number of documents that happen to match one or two terms of a long query.

4.2 Co-occurrence Analysis Helps Local Feedback

The most critical function of a local feedback algorithm is to separate terms in the top ranked relevant documents from those in top ranked non-relevant documents. The metric typically used for this purpose is the frequency of a term in the top ranked documents. In general, this metric fails if most of the top ranked documents are non-relevant. We will show that feature selection based on co-occurrence is a better strategy.

We first make an observation/hypothesis about the top ranked documents retrieved for a query. The observation is that the top ranked documents tend to form a number of clusters, each of which is about a certain topic. This is similar to the cluster hypothesis described by van Rijisbergen in [31]. The relevant documents usually are in one cluster. Besides the relevant cluster, there are some clusters of non-relevant documents. This phenomenon can be explained by the existence of *overlapping topics*. Two topics are overlapping if they share many terms but are about different information needs and correspond to different documents. Consider the example query "as a result of DNA testing, are more defendants being absolved or convicted of crimes?". One overlapping topic to this query might be "As the result of the polygraph test, are more defendants being convicted or absolved of crimes?". If a query has one or more overlapping topics, we expect that the top ranked documents will consist of several clusters, each of which can be viewed as the retrieval results for a different topic. Note that a relevant document does not necessarily match the query better than a document retrieved for an overlapping topic due to synonyms and term dependencies in the query. A relevant document for the above example query may match two query terms "DNA" and "crime" while a non-relevant one may match three query terms "test", "crime" and "defendant". The cluster phenomenon implies that the expansion terms chosen by local feedback would be mostly from the largest cluster. If the largest cluster is for an overlapping topic, local feedback fails. In other words, whether local feedback succeeds does not exactly depend on how many top ranked documents are relevant. Rather, it depends on whether the relevant cluster is the largest cluster of top ranked documents. However, the two conditions are highly correlated. The smaller the number of the top ranked documents that are relevant, the less likely that the relevant cluster is the largest.

The other observation is that a non-relevant cluster tends to miss one or two key query terms, since the documents in the cluster can be viewed as the retrieval results for an overlapping topic. For example, the documents about polygraph test and crime will probably miss the word "DNA". In contrast, the relevant cluster tends to contain all query terms. In other words, nearly every query term is used by at least some relevant documents, although it may not be used by all relevant documents. This is not surprising because the purpose of using a term in a query is that it helps describe the relevant documents. If a term is a common term in the relevant cluster, it will likely to co-occur with all query terms.

Based on the above discussion, we hypothesize that the common terms in the top ranked relevant documents tend to have the unique property of co-occurring with all query terms in the top ranked documents. In other words, we can apply the Phrasefinder-style technique on the top ranked documents and potentially achieve better query expansion than local feedback. That is essentially what local context analysis will do for query expansion. We should point out that we have assumed that there are a reasonable number of relevant documents in the top ranked set. This is the assumption behind local context analysis.

4.3 Implementation of Local Context Analysis

We now discuss the implementation issues of local context analysis. Since previous research suggests that nouns are more informative than other types of terms and are better features for query expansion [18], local context analysis normally uses nouns and noun phrases as expansion features. These nouns and noun phrases are called expansion concepts. For English text, concepts are recognized by a part of speech tagger, *Jtag* [34].

Ideally we would like each document to deal with a single topic, but in practice many long documents deal with several topics. To avoid the potential problem of using words from the unrelated parts of a long document for query expansion, local context analysis uses the top ranked passages. Passages are created by breaking documents into fixed-length text windows. The default passage size is 300 words.

In the following discussion, we assume that the query to be expanded is Q, the query terms in Q are $w_1, w_2 ... w_m$, and the collection being searched is C. The top ranked n passages for query Q is retrieved from collection C by the INQUERY retrieval system [3]. We denote the set of top n passages as $S = \{p_1, p_2, ... p_n\}$. Based on our previous discussion, we should use concepts that co-occur with all query terms for query expansion. In practice, however, a query term may not be used in any relevant document. Our approach is to prefer concepts co-occurring with more query terms over those co-occurring with fewer query terms for query expansion. Specifically, we will derive a function $f(c, Q)$ which measures how good a concept c is for expanding query Q based on c's co-occurrences with w_i's in S. All concepts are ranked by f and the best concepts are added to Q. Although function f is heuristically driven and is derived experimentally, it can be easily justified:

- Co-occurrence metric

  We hypothesized that good expansion concepts tend to co-occur with all query terms in the top ranked passages. But we must take random co-occurrences into account: a concept c could just co-occur with a query term w_i in the top ranked passages by chance. The higher the concept c's frequency in the whole collection, the more likely it is that it co-occurs with w_i by chance. The larger the number of co-occurrences, the less likely that c co-occurs with w_i by chance. Let N be the number of passages in C, N_c the number of passages that contain c, $co(c, w_i)$ the number of co-occurrences between c and w_i in S. We proposed the following metric to measure the degree of co-occurrence of c with w_i:

  $$\begin{array}{lcl} co\_degree(c, w_i) & = & log_{10}(co(c, w_i) + 1)idf(c)/log_{10}(n) \\ co(c, w_i) & = & \sum_{p\ in\ S} tf(c, p)\ \ tf(w_i, p) \\ idf(c) & = & min(1.0, log_{10}(N/N_c)/5.0) \end{array}$$

  where $tf(c, p)$ and $tf(w_i, p)$ are the frequencies of c and w_i in passage p respectively. The metric can be interpreted as the likelihood that concept c and query term w_i co-occur non-randomly in the top ranked documents. The metric takes into account the frequency of c in the whole collection ($idf(c)$) and the number of co-occurrences between c and w in the top ranked passages ($co(c, w_i)$). The logarithm function is used to dampen the raw numbers of occurrences and co-occurrences. The metric is also normalized over n, which is an upper-bound for the raw number of co-occurrences in most cases.

  We should point out that the above co-occurrence metric is different from well-known metrics such as EMIM (expected mutual information measure) [31; 9], cosine [25] and so forth. One reason we choose not to use the available metrics is that they are designed to measure corpus-wide co-occurrences and it is not clear how to adapt them to measure co-occurrences in the top ranked passages. The other reason is that we want to explicitly bias against high-frequency concepts but available metrics cannot do that.

  We should also point out that our definition of *idf* is somewhat different from the standard definition $idf(c) = log_{10}(N/N_c)$ used by other researchers. The problem with the latter definition is that mathematically it has no upper limit when N approaches infinity. Our formula sets an upper limit on $idf(c)$. Any concept which occurs in 1/100000 of the passages or less frequently will have *idf* 1.0.

- Combining the degrees of co-occurrence with all query terms

  As we just discussed, $co\_degree(c, w_i)$ can be interpreted as the likelihood that concept c co-occurs with query term w_i non-randomly in the top ranked passages. Now we need to estimate the likelihood that c co-occurs non-randomly with all w_i's in the top ranked passages. Assuming c's co-occurrences with different query terms are independent, the natural estimate is to multiply the n $co\_degree(c, w_i)$'s. A problem with such an estimate is that if one of the n numbers is 0, the product is 0 no matter what the other $n - 1$ numbers are. A more desirable function should produce a non-zero value based on the other $n-1$ numbers. For this purpose, we add a small constant δ to each degree of co-occurrence. The function for combining the n numbers is therefore

$$g(c,Q) = \prod_{w_i \text{ in } Q} (\delta + co\_degree(c, w_i))$$

The use of δ in g is a simple smoothing technique. Smoothing is widely used in various statistical models (including IR models) which deal with limited amount of data. For example, INQUERY uses a default belief (typically 0.4) to prevent zero values from its #AND operator when one operand is zero.

From another perspective, because of δ, g achieves a "relaxed" interpretation of the Boolean statement that good concepts co-occur with all query terms. To simplify, we assume Q has only two terms w_1 and w_2. We can rewrite $g(c,Q)$ as:

$$g(c,Q) = co\_degree(c,w_1)co\_degree(c,w_2) + \delta(co\_degree(c,w_1) + co\_degree(c,w_2)) + \delta^2$$

The first part of the formula, $co\_degree(c,w_1)co\_degree(c,w_2)$ emphasizes co-occurrence with all query terms. The second part, $\delta(co\_degree(c,w_1)+co\_degree(c,w_2))$, emphasizes co-occurrence with individual query terms. The third part, δ^2, has no effect on the ranking of the concepts. The relative weights of the first and second parts are controlled by the δ value. With a small δ, concepts co-occurring with all query terms are ranked higher. With a large δ, concepts having significant co-occurrences with individual query terms are ranked higher. In a sense, g is similar to the *and* operator in the p_norm model [13]. The purpose of δ in g is the same as that of p in the p_norm model.

- Differentiating rare and common query terms

  Obviously, not all query terms are equally important. While deciding the importance of a query term is a hard problem in general, it is well-known that rare terms are usually more important than frequent ones. This is the reason behind the $tf \times idf$ formula used by most IR systems. Taking into account the *idf* of the query terms, we get the following function,

$$f(c,Q) = \prod_{w_i \text{ in } Q} (\delta + co\_degree(c, w_i))^{idf(w_i)}$$

  This function f is used by local context analysis for ranking concepts. The $idf(w_i)$'s can be viewed as weights in the formula. It is easy to see this by taking log on both sides of the formula. Note that multiplying $idf(w_i)$'s and g does not work because given a query it only scales g by a constant factor.

In summary, local context analysis takes these steps to expand a query Q on a collection C: (1) Use INQUERY to retrieve the top n ranked passages from C. (2) Rank the concepts in the top ranked passages using the formula $f(c,Q)$. (3) Add the best k concepts to Q.

The parameter setting for local context analysis is experimentally determined. The default passage size is 300 words. The default number of top ranked passages used per query is 100. The default δ value is 0.1. The default number of concepts added to a query is 70. The concepts are added to a query according to the following formulas:

$$\begin{aligned}
Q_{new} &= \#\mathrm{WSUM}(1.0\ 1.0\ Q_{old}\ wt\ Q\prime) \\
Q\prime &= \#\mathrm{WSUM}(1.0\ wt_1\ c_1\ wt_2\ c_2\ ...\ wt_{70}\ c_{70}) \\
wt_i &= 1.0 - 0.9i/70
\end{aligned}$$

where c_i is the ith ranked concept. We call $Q\prime$ the auxiliary query. The default value for wt is 2.0. #WSUM is an INQUERY operator to combine evidence from different parts of a query. Specifically, it computes a weighted average of its operands. Other parameter settings are also used to observe the effect on retrieval performance.

Figure 1 shows an example query expanded by local context analysis.

```
#WSUM(1
      1 #WSUM (1 1 status 1 nuclear 1 proliferation 1 treaties
                 1 violations 1 monitoring)
      2 #WSUM (1
               1        #PHRASE(nuclear non proliferation treaty)
               0.987143 treaty
               0.974286 weapon
               0.961429 pakistan
               0.948571 missile
               0.935714 iraq
               0.922857 proliferation
               0.91     #PHRASE(non proliferation treaty)
               0.897143 #PHRASE(international atomic energy agency)
               0.884286 india
               0.871429 warhead
               0.858571 uranium
               0.845714 disarmament
               0.832857 china
               0.82     #PHRASE(chemical weapon)
               0.807143 spread
               ...
               ))
```

Figure 1: Query expansion by local context analysis for TREC topic 202 “Status of nuclear proliferation treaties, violations and monitoring”. #PHRASE is an INQUERY operator to construct phrases.

5 Experimental Methodology

Table 1 lists the test collections used in the experiments in this paper. These test collections have quite different characteristics. The TREC3 queries are very long, averaging 34.5 words per query. The TREC5 queries are much shorter, averaging only 7.1 words per query. The WEST documents are very long, more than 10 times longer than TREC5-SPANISH documents on average. The TREC3 queries have far more relevant documents than the WEST queries. The WEST collection is homogeneous in that its documents have similar types and similar content. Other collections are more heterogeneous and contain documents of different types, different content, different lengths and different sources. The collections are in three languages: English, Spanish and Chinese. It is well-known that many IR techniques are sensitive to factors such as query length [32], document length [28], collection size and so forth. The purpose of using a wide variety of collections is to ensure the generality of the conclusions we reach in this paper.

collection	query count	size (GB)	document count	words per query	words per document	rel docs per query
WEST	34	0.26	11,953	9.6	1967	28.9
TREC3	50	2.2	741,856	34.5	260	196
TREC4	49	2.0	567,529	7.5	299	133
TREC5	50	2.2	524,929	7.1	333	110
TREC5-SPANISH	25	0.34	172,952	8.2	156	100
TREC5-CHINESE	19	0.17	164,779	21	411	73.6

Table 1: Statistics about test collections. Stop words are not included. Each Chinese character is counted as a word.

We will compare the performance of local context analysis not only with the performance of the original (unexpanded) queries, but also with the performance of local feedback and that of Phrasefinder. The main evaluation metric is interpolated 11 point average precision. Statistical t-test [17] and query by query analysis are also employed. To decide whether the improvement by method A over method B is significant, the t-test calculates a p_value based on the performance data of A and B. The smaller the p_value, the more significant is the improvement. If the p_value is small enough (p_value < 0.05), we conclude that the improvement is statistically significant.

Experiments with local feedback and Phrasefinder are carried out using established parameter settings for the two techniques. The local feedback experiments are based on the procedure used by the Cornell group in the TREC conferences [6]. It represents one of the most successful techniques used in the TREC conferences. The most frequent 50 terms and 10 phrases (pairs of adjacent non-stop words) from the top ranked documents are added to a query. The terms and phrases in the expanded query are then re-weighted using the Rocchio weighting method [24] with $\alpha : \beta : \gamma = 1 : 1 : 0$. The Phrasefinder experiments are based on the method described in the UMass TREC3 report [4]: 30 concepts are added to a query and are weighted in proportion to their rank position. Concepts containing only terms in the original query are weighted more heavily than those containing terms not in the original query.

6 Experimental Results

We now present the experimental results of three query expansion techniques: Phrasefinder, local feedback and local context analysis. The experiments were carried out on TREC3, TREC4, TREC5 and WEST (Tables 2, 3, 4 and 5). In the experiments, 10 documents were used per query for local feedback and 100 passages per query for local context analysis. Other parameters took their default value. A Phrasefinder result for TREC5 is not available.

On TREC3, local context analysis is 23.3% better then the baseline, which is statistically significant (p_value=0.000005). Local context analysis is also better than Phrasefinder by 14% and local feedback by 2.4%. The improvement by local context analysis over Phrasefinder is statistically significant (p_value=0.0003), but the improvement over local feedback is not. Query expansion is generally regarded as a recall device, but the results show that precision at low recall points is improved as well. The reason for the improved precision is that some relevant documents which are ranked low by the original queries are pushed to the top of the ranked output because they contain many expansion concepts. The results show that query expansion potentially can be a precision device too.

On TREC4, local context analysis is significantly better than the baseline (23.5%, p_value=0.00000006). Local context analysis is also significantly better than Phrasefinder (19.6%, p_value=0.00001) and local feedback (11.5%, p_value=0.001).

On TREC5, local context analysis is 2.3% better than the baseline and 2.1% better than local feedback. The improvements are not statistically significant. We think that the TREC5 result is less impressive mainly because of the peculiarities of the TREC5 query set. A number of queries are ill-formed from the retrieval point of view in that they use terms which are poor or even negative indicators of relevance. Examples are queries "existence of human life 10,000 years ago" (which is highly ambiguous) and "Identify social programs for poor people in countries other than the U.S." ("U.S." is negative evidence of relevance). For such queries, local context analysis is likely to choose bad concepts for query expansion by requiring them to co-occur with the bad query terms. Furthermore, three of the TREC5 queries have only one or two relevant documents in the whole collection. Local context analysis assumes that there are a reasonable number of relevant documents. Since the assumption is violated, local context analysis fails to improve them.

On WEST, local context analysis is 0.8% better than the baseline, which is not statistically important. But it is significantly better than Phrasefinder (11.8%, p_value=0.00003) and local feedback (8.8%, p_value=0.002). The reason for the small improvement is due to the high quality of the original queries. A comparison of the WEST and TREC queries shows that the WEST queries are more precise descriptions of the information needs. This is also shown by the very good performance of the original queries. Word mismatch is a less serious problem and therefore query expansion is less useful. But even with such high quality queries, local context analysis manages to produce a small improvement. The improvement at low recall points is more noticeable. In contrast, Phrasefinder and local feedback seriously degrade the WEST queries. Since the original queries are very good, we conjecture that better retrieval is possible if we downweight the expansion concepts. This is

	Precision (% change) – 50 queries			
Recall	base	Phrasefinder	lf-10doc	lca-100p
0	82.2	79.4 (−3.3)	82.5 (+0.4)	87.0 (+5.9)
10	57.3	60.1 (+4.8)	64.9 (+13.3)	65.5 (+14.3)
20	46.2	50.4 (+9.1)	56.1 (+21.5)	57.2 (+23.8)
30	39.1	43.3 (+10.7)	48.3 (+23.5)	48.4 (+23.8)
40	32.7	36.9 (+12.8)	41.6 (+26.9)	42.7 (+30.4)
50	27.5	31.8 (+15.9)	36.8 (+34.1)	37.9 (+38.0)
60	22.6	26.1 (+15.1)	30.9 (+36.7)	31.5 (+39.3)
70	18.0	20.6 (+14.0)	25.2 (+40.0)	25.6 (+42.1)
80	13.3	15.8 (+18.6)	19.4 (+45.7)	19.4 (+45.7)
90	7.9	9.4 (+18.7)	11.5 (+44.3)	11.7 (+47.3)
100	0.5	0.8 (+60.9)	1.2 (+143.5)	1.4 (+177.0)
average	31.6	34.1 (+7.8)	38.0 (+20.5)	38.9 (+23.3)

Table 2: A comparison of baseline, Phrasefinder, local feedback and local context analysis on TREC3. 10 documents for local feedback (lf-10doc). 100 passages for local context analysis (lca-100p)

	Precision (% change) – 49 queries			
Recall	base	Phrasefinder	lf-10doc	lca-100p
0	71.0	68.6 (−3.3)	68.4 (−3.6)	73.2 (+3.2)
10	49.3	48.6 (−1.6)	52.8 (+7.0)	57.1 (+15.7)
20	40.4	40.0 (−1.0)	43.2 (+7.0)	46.8 (+16.0)
30	33.3	33.9 (+1.8)	36.0 (+8.0)	39.9 (+19.8)
40	27.3	28.0 (+2.5)	29.8 (+9.2)	35.3 (+29.1)
50	21.6	23.9 (+10.3)	24.5 (+13.2)	29.9 (+38.4)
60	14.8	18.8 (+27.1)	19.7 (+33.4)	23.6 (+59.8)
70	9.5	11.8 (+24.7)	14.8 (+56.9)	17.9 (+89.1)
80	6.2	8.1 (+31.0)	10.8 (+74.7)	11.8 (+91.0)
90	3.1	4.2 (+33.6)	6.4 (+104.6)	5.7 (+80.2)
100	0.4	0.6 (+24.0)	0.9 (+93.3)	0.8 (+88.2)
average	25.2	26.0 (+3.4)	27.9 (+11.0)	31.1 (+23.5)

Table 3: A comparison of baseline, Phrasefinder, local feedback and local context analysis on TREC4. 10 documents for local feedback (lf-10doc). 100 passages for local context analysis (lca-100p)

supported by retrieval results: When the weights of expansion concepts are reduced by 50%, local context analysis produces a 3.3% improvement over the baseline. When we reduce the β parameter of the Rocchio weighting formula by 50%, the result of local feedback is also improved, but it is still 3.3% worse than the baseline. In the remainder of the paper, we will always downweight the expansion features by 50% for local context analysis and local feedback on WEST.

As we mentioned before, a problem with local feedback is its inconsistence. It can

	Precision (% change) – 50 queries		
Recall	base	lf-10doc	lca-100p
0	64.1	48.5 (−24.4)	53.7 (−16.2)
10	37.5	36.8 (−1.9)	34.4 (−8.4)
20	29.1	31.0 (+6.5)	30.9 (+6.3)
30	24.1	26.5 (+9.9)	26.4 (+9.3)
40	21.3	22.6 (+6.2)	23.5 (+10.5)
50	17.9	19.3 (+7.8)	20.7 (+15.8)
60	12.6	15.6 (+24.2)	17.1 (+36.2)
70	10.1	12.9 (+27.6)	12.7 (+25.2)
80	7.2	9.4 (+31.5)	8.7 (+21.6)
90	4.8	6.6 (+36.3)	6.2 (+28.2)
100	2.7	2.7 (−2.3)	2.4 (−13.5)
average	21.0	21.1 (+0.2)	21.5 (+2.3)

Table 4: Retrieval performance on TREC5. 10 documents are used for local feedback (lf-10doc). 100 passages are used for local context analysis (lca-100p).

	Precision (% change) – 34 queries					
recall	base	Phrasefinder	lf-10doc	lf-10doc-dw	lca-100p	lca-100p-dw
0	88.0	83.9 (−4.7)	80.7 (−8.3)	81.9 (−7.0)	91.9 (+4.4)	92.1 (+4.7)
10	80.0	74.5 (−6.9)	76.0 (−5.0)	76.9 (−4.0)	85.7 (+7.1)	84.3 (+5.4)
20	77.5	67.2 (−13.3)	70.5 (−8.9)	71.4 (−7.8)	76.3 (−1.5)	78.5 (+1.3)
30	74.1	64.3 (−13.2)	66.8 (−9.8)	68.2 (−7.9)	71.1 (−4.0)	73.9 (−0.1)
40	62.9	54.6 (−13.2)	59.5 (−5.4)	60.8 (−3.3)	61.3 (−2.6)	61.8 (−1.7)
50	57.5	49.5 (−14.0)	54.2 (−5.8)	56.8 (−1.2)	55.2 (−3.9)	56.8 (−1.2)
60	49.7	44.6 (−10.1)	45.5 (−8.3)	50.1 (+0.8)	49.2 (−0.8)	50.7 (+2.2)
70	41.5	37.4 (−9.9)	38.7 (−6.8)	42.1 (+1.3)	41.1 (−1.1)	44.2 (+6.4)
80	32.7	30.0 (−8.2)	29.2 (−10.8)	33.1 (+1.1)	33.6 (+2.6)	36.4 (+11.2)
90	19.3	18.4 (−4.6)	18.1 (−6.1)	21.8 (+13.0)	21.5 (+11.7)	22.6 (+17.1)
100	8.6	8.7 (+0.3)	8.2 (−4.8)	9.3 (+7.8)	9.6 (+10.9)	10.0 (+15.3)
average	53.8	48.5 (−9.9)	49.8 (−7.5)	52.0 (−3.3)	54.2 (+0.8)	55.6 (+3.3)

Table 5: A comparison of baseline, Phrasefinder, local feedback and local context analysis on WEST. 10 documents are used for local feedback (lf-10doc). Rocchio β parameter is reduced by 50% in lf-10doc-dw. 100 passages are used for local context analysis (lca-100p). Expansion concepts are downweighted by 50% in lca-100p-dw.

improve some queries and serious hurt others. A query by query analysis on TREC4 shows that local context analysis is better than local feedback in this aspect. Although both techniques significantly improve retrieval effectiveness on TREC4, local context analysis improves more queries and hurts fewer than local feedback. Of 49 TREC4 queries, local feedback hurts 21 and improves 28, while local context analysis hurts 11 and improves 38. Of the queries hurt by local feedback, 5 queries have a more than 5% percent loss in average precision. In the worst case, the average precision of one query is reduced from 24.8% to

4.3%. Of those hurt by local context analysis, only one has a more than 5% percent loss in average precision. Local feedback also tends to hurt queries with poor performance. Of 9 queries with baseline average precision less than 5%, local feedback hurts 8 and improves 1. In contrast, local context analysis hurts 4 and improves 5.

Overall, experimental results show that local context analysis is a more effective and more consistent query expansion technique than local feedback and Phrasefinder. Of the three techniques, Phrasefinder is the least effective. The results show that while analysis of the top ranked documents/passages is more effective than analysis of a whole corpus, co-occurrence analysis traditionally used by global techniques can make local techniques more effective and more consistent.

7 Local Context Analysis vs Local Feedback

7.1 Varying the Number of Passages/Documents Used

A parameter in local context analysis and local feedback is how many top ranked passages/documents to use for a query. So far we have not found a satisfactory method to automatically determine the optimal number of passages/documents on a query by query basis. Until we find a solution, we hope that a technique does not rely too heavily on the particular value chosen for the parameter. In other words, a desirable technique should work well for a wide range of choices.

First we vary the number of top ranked passages used per query and check the impact on the performance of local context analysis. The results are shown in Table 6. For each collection and each choice of the number of passages, the table shows the average precision and the improvement over the baseline. The results show that local context analysis works for a wide range of choices except for TREC5 (as we discussed before, the TREC5 query set is very peculiar and should be considered an exception). For TREC3 and TREC4 in particular, any choice between 30 to 300 works pretty well.

We now check the impact of the number of top ranked documents used per query on the performance of local feedback. The results are shown in Table 7. Local feedback depends heavily on the number of documents used per query except for TREC3.

	Number of passages							
collection	10	20	30	50	100	200	300	500
TREC3	36.6	37.5	38.7	38.9	38.9	39.3	39.1	38.3
	+16	+18.9	+22.6	+23.2	23.3	+24.4	+23.7	+21.3
TREC4	29.5	29.9	30.2	30.4	31.1	31.0	30.7	29.9
	+17	+18.6	+19.8	+20.6	+23.5	+23.0	+21.8	+18.6
TREC5	23.0	23.0	22.5	21.1	21.5	21.1	20.8	20.9
	+9.2	+9.2	+6.8	+0.3	+2.3	+0.1	-1.0	-0.9
WEST	55.9	56.5	55.6	55.8	55.6	54.6	54.4	53.6
	+3.8	+5.0	+3.4	+3.7	+3.3	+1.6	+1.2	-0.4

Table 6: The impact of the number of passages used per query on the performance of local context analysis

	number of documents used					
collection	5	10	20	30	50	100
TREC3	36.6	38.0	37.6	37.7	37.7	36.6
	+16.0	+20.5	+19.1	+19.4	+19.3	+15.8
TREC4	28.7	27.9	26.9	27.2	26.7	26.1
	+14.0	+11.0	+6.8	+8.2	+6.2	+3.5
TREC5	21.1	21.1	19.3	19.4	19.4	17.8
	+0.5	+0.2	-8.2	-7.9	-7.6	-15.2
WEST	52.6	52.0	48.7	47.5	44.5	40.0
	-2.2	-3.3	-9.5	-11.6	-17.2	-25.7

Table 7: The impact of the number of documents used per query on the performance of local feedback

In general, the results show that local context analysis is less sensitive to the number of passages/documents used than local feedback. The difference between the two techniques in this aspect is more clearly shown in Figure 2. From this figure we can see another difference between the two techniques: their performances peak at quite different numbers of passages/documents. The performance of local feedback peaks when relatively few documents are used per query, while the performance of local context analysis peaks when significantly more passages are used. On TREC4, increasing the number of documents/passages from 10 to 100 hurts one technique but improves the other. The same is observed even if the top ranked passages were used for local feedback. The difference can be explained by the assumptions made by the two techniques. The assumption made by local feedback (e.g. the relevant cluster is the largest) is less likely to hold with more passages/documents. When more passages/documents are used, the percentage of non-relevant ones will increase. The chance of retrieving large clusters of non-relevant passages/documents will also increase. On the other hand, the assumption made by local context analysis (e.g. a reasonable number of top ranked passages are relevant) is more likely to hold with more passages. When the number of passages is increased, the number of relevant ones is increased too. That is why increasing the number of passages/documents hurts one technique but helps the other.

7.2 Dependence on the Quality of the Top Ranked Set

Although both local context analysis and local feedback assume that some top ranked documents/passages are relevant, experimental results on TREC4 show that local context analysis is less heavily dependent on the assumption than local feedback. For easy comparison, both techniques use the top ranked 100 passages for query expansion. We will discuss passage level local feedback in Section 7.4. We define a relevant passage as a passage from a relevant document. The average number of relevant passages in the top ranked set is 26 per query. In the discussion we are only interested in queries which are improved or degraded by a technique, with performance change 1% or more.

We first examine local context analysis. For queries improved by local context analysis, the average number of relevant passages in the top ranked set is 29.7 per query. For queries degraded, the number is 8.6. There is no clean separation of the two sets of queries based

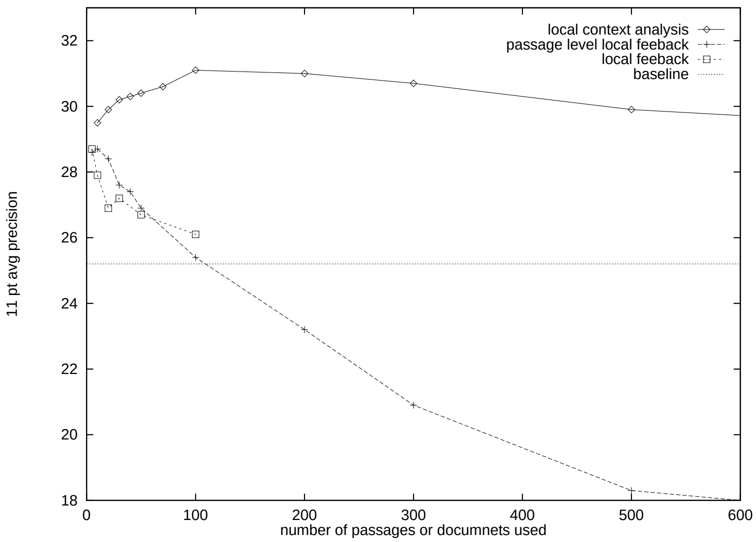

Figure 2: Performance curves of local context analysis and local feedback on TREC4.

on the number of relevant passages. For queries with 18 or more relevant passages, none of them is hurt by local context analysis. For queries with fewer than 18 relevant passages, some are hurt and some are improved by local context analysis.

We now examine local feedback. The improved queries have 32 relevant passages in the top ranked set per query on average, while the degraded queries have 20 relevant passages per query. The statistics show that local feedback generally requires more relevant passages in the top ranked set to improve a query. Unlike local context analysis, local feedback hurts a number of queries even if they have a substantial number of relevant passages in the top ranked set. An examination of the top ranked passages show that such queries usually contain a large cluster of non-relevant passages. This supports our hypothesis that local feedback fails if the relevant cluster is not the largest in the top ranked set. One such query is “What research is ongoing to reduce the effects of osteoporosis in existing patients as well as prevent the disease occurring in those unafflicted at this time?”. Many passages about treatment of heart disease are retrieved because they happen to use many of the query terms. Because local feedback selects expansion features based on frequency in the top ranked set, terms such as “cholesterol” and “coronary” are chosen for query expansion. In comparison, local context analysis avoids this problem because the passages about heart disease do not use the query term “osteoporosis” and therefore concepts in them will not co-occur with that the term.

The fact that local context analysis is still dependent on the relevance of some top ranked passages means that if we can predict with high accuracy whether the top ranked set contains enough relevant passages, we would be able to do a better job improving retrieval performance. Rather than expand all queries, we would only expand those which are likely to have enough top ranked relevant passages. One method we have tried is based on the number of matched query terms in the top ranked passages. The basic idea is that if the top ranked passages contain few query terms, they are unlikely to be relevant and we will not expand the query. Unfortunately, this simple method does not work. Data obtained on TREC4 show that the queries improved and the queries hurt by local context analysis have roughly the same average number of matched query terms (about 4.0) in a top ranked passage when top 10 passages are considered for each query. Whether more refined methods will work awaits further investigation.

7.3 Differences in Expansion Features

Another difference between local feedback and local context analysis is that the expansion features chosen by the two techniques are very different, even though both techniques select expansion features from the top ranked set. Due to the syntactic difference between the expansion features (noun phrases vs terms and term pairs), direct comparison is impossible. To get around the problem, we break composite features into terms and use terms as the units of comparison. When the best runs of the two techniques on TREC4 are considered, the average number of unique terms in the expansion features is 58 per query for local feedback and 78 for local context analysis. The number of overlapping terms is only 17.6 per query. Some queries expanded quite differently are improved by both methods. The small overlap is understandable because the expansion features chosen by local feedback and local context analysis have different properties. In general, those selected by local feedback have a high frequency in the top ranked set while those selected by local context analysis co-occur with all query terms in the top ranked set.

7.4 Other Differences

Recall that local context analysis is different from local feedback in three aspects. Local context analysis uses passages while local feedback uses whole documents. Local context analysis uses noun phrases while local feedback uses terms and pairs of terms. The strategy for feature selection in local context analysis is co-occurrence-based while it is frequency-based in local feedback. It is reasonable to doubt that the difference between local context analysis and local feedback is simply due to the first two factors. To dispel that doubt, we have done two more sets of experiments. Firstly we re-did the local feedback experiments on TREC3, TREC4 and WEST using passages. The results are shown in Table 8. Comparing Tables 7 and 8, we can see that while using passages improves local feedback noticeably on WEST and marginally on TREC4, it hurts on TREC3. Overall, using passages seems to improve performance somewhat but cannot bridge the performance difference between local context analysis and local feedback. Secondly, we did local context analysis using terms and pairs of terms similar to local feedback. The experiment was done on TREC4 and 100 passages were used per query. The results show that using noun phrases is only 0.2% better than using terms and pairs of terms (Table 9). In short, we have to conclude that the main factor accounting for the difference between local context analysis and local feedback is feature selection.

In summary, we have shown that local context analysis and local feedback are two quite different query expansion techniques. Local context analysis is in general more effective and more predictable than local feedback. The differences between the two are caused by the different feature selection strategies used.

	number of passages used					
collection	5	10	20	30	50	100
TREC3	34.7	35.5	36.4	37.0	36.6	35.8
	+9.8	+12.4	+15.2	+17.3	+15.8	+13.5
TREC4	28.6	28.7	28.4	27.6	26.9	25.4
	+13.6	+13.9	+12.7	+9.6	+6.8	+0.8
WEST	55.1	56.0	52.9	53.9	52.2	49.5
	+2.4	+4.1	-1.7	+0.1	-3.0	-8.1

Table 8: Performance of local feedback using top ranked passages

	Precision (% change) – 49 queries		
Recall	noun phrases	terms and pairs	
0	73.2	75.8	(+3.5)
10	57.1	56.9	(−0.2)
20	46.8	46.9	(+0.1)
30	39.9	39.3	(−1.4)
40	35.3	33.5	(−4.9)
50	29.9	28.2	(−5.9)
60	23.6	22.2	(−5.9)
70	17.9	18.0	(+0.5)
80	11.8	12.9	(+9.9)
90	5.7	6.9	(+21.1)
100	0.8	0.8	(−3.6)
average	31.1	31.0	(−0.2)

Table 9: Comparing noun phrases with terms and pairs of terms as local context analysis expansion features on TREC4. 100 passages are used.

8 Comparing Local Context Analysis with Phrasefinder

We have discussed the impact of the number of passages used on the performance of local context analysis. We now revisit this issue. If all the passages in a collection are used, local context analysis is essentially Phrasefinder. In other words, Phrasefinder is an extreme special case of local context analysis. The performance curve of local context analysis in Figure 2 predicts that this special case of local context analysis will be worse than using the top ranked passages. This is confirmed by the actual retrieval results of Phrasefinder. As discussed before, the reason for the inferior performance of Phrasefinder is the low ranked passages (documents). Since the overwhelming majority (approaching 100% on large collections) of them are not even remotely related to the topic of the query, using them is not only inefficient, but also hurts the chance of choosing the good concepts for query expansion. For example, let us consider one TREC4 query "Is there data available to suggest that capital punishment is a deterrent to crime?". Phrasefinder added some non-relevant concepts such as "Iraqi leader", "world order", "Iraqi army" and "shields" to the query because they co-occur with query terms "available", "suggest", "capital", "crime" and "deterrent" in the collection. It happens that many documents in the collection are about the desert storm and many of the terms in the query occur in some of these documents (though not simultaneously). Since Phrasefinder uses global co-occurrences for query expansion, it chooses the common concepts from these documents. This is not a problem for local context analysis because these passages are not in the top ranked set.

9 Parameter Variation

We now experiment with different parameter values and see how the performance of local context analysis is affected. The parameters we consider are the passage size, the δ value (in function $f(c, Q)$) and the number of concepts added to a query.

9.1 Passage Size

In Table 10, we list the retrieval performance of local context analysis on WEST using different passage sizes. We experimented with passage sizes 100, 200, 300, 400 and 500 words. For each passage size, we used the top 10, 20, 30, 40, 50, 100, 200 and 300 passages for query expansion. As we can see from the table, local context analysis produces improvements over the baseline for a wide range of passage sizes and numbers of passages used. In general, with a larger window size, optimal performance occurs with a smaller number of passages. Performance seems to be independent of the passage size.

	Number of passages used							
passage size	10	20	30	40	50	100	200	300
100	55.3	55.9	55.8	56.3	56.6	55.4	54.8	54.7
	+2.8	+3.8	+3.7	+4.7	+5.2	+3.0	+1.8	+1.6
200	55.3	56.7	57.0	57.3	57.0	56.1	55.1	54.6
	+2.8	+5.5	+5.9	+6.5	+6.0	+4.4	+2.5	+1.5
300	55.9	56.5	55.6	55.7	55.8	55.6	54.6	54.4
	+3.8	+5.0	+3.4	+3.6	+3.7	+3.3	+1.6	+1.2
400	56.5	56.7	55.7	55.5	55.8	55.9	54.5	53.7
	+4.9	+5.4	+3.6	+3.1	+3.8	+4.0	+1.3	-0.1
500	56.0	56.4	56.3	56.7	56.2	55.9	55.0	54.2
	+4.1	+4.8	+4.7	+5.5	+4.5	+4.0	+2.2	+0.7

Table 10: Effect of passage size and number of passages used on retrieval performance of local context analysis on WEST

Though the experiments on WEST show that the performance is independent of the passage size, we believe that neither too large a passage size nor too small a passage size is desirable. If the passage size is too small, there will be fewer matched words between a query and the passages. A passage matching only the non-content words in the query may be ranked high for the query. Consequently, the quality of the retrieved passages is affected. Since the WEST queries are short, this is not a serious problem. But this could be a serious problem for longer queries such as the TREC3 queries. If the passages are too long, the top ranked passages may contain extraneous text and the performance may be affected too. Experiments on other collections are needed to test the above hypothesis. To avoid the potential problems with too small and too large passage sizes, we recommend 300 words as a good choice for the passage size. This is consistent with findings reported by previous research on passage-based retrieval [8].

The best approach may be to segment long documents into passages so that each passage is about a topic. Techniques that automatically segment long documents or text streams by topics have been proposed in a number of studies [16; 22]. We plan to investigate the application of these techniques for local context analysis in future work.

9.2 δ Value

Table 11 shows the effect of the δ value on the performance of local context analysis on TREC4. We can see that the average precision is relatively insensitive to the δ value. But precision at individual recall points is affected. In general, a small δ value is good for precision and a large δ is good for recall. We have discussed that a small δ favors concepts co-occurring with all query terms while a large δ favors those co-occurring with individual query terms. The experimental results seem to imply that concepts co-occurring with all query terms are good for precision and concepts co-occurring with individual query terms are good for recall.

	Precision (% change) – 49 queries					
recall	0.001	0.01	0.05	0.1	0.2	0.3
0	77.8	75.7 (−2.7)	73.4 (−5.6)	73.2 (−5.9)	72.9 (−6.3)	73.1 (−6.0)
10	55.1	54.9 (−0.2)	57.0 (+3.6)	57.1 (+3.7)	56.8 (+3.2)	56.6 (+2.7)
20	46.7	47.4 (+1.6)	47.7 (+2.2)	46.8 (+0.3)	46.6 (−0.1)	46.5 (−0.5)
30	40.0	40.4 (+1.0)	40.0 (+0.1)	39.9 (−0.2)	39.6 (−0.9)	39.4 (−1.5)
40	34.8	35.4 (+1.6)	35.4 (+1.7)	35.3 (+1.4)	34.8 (+0.1)	34.6 (−0.7)
50	29.4	30.1 (+2.4)	30.2 (+2.6)	29.9 (+1.7)	30.1 (+2.4)	30.1 (+2.2)
60	22.5	23.1 (+2.6)	23.7 (+5.3)	23.6 (+5.2)	24.1 (+7.3)	24.1 (+7.1)
70	15.9	17.0 (+6.6)	17.7 (+11.2)	17.9 (+12.3)	17.6 (+10.7)	17.6 (+10.3)
80	11.0	11.6 (+6.0)	11.7 (+7.2)	11.8 (+7.3)	11.9 (+8.3)	11.9 (+8.2)
90	5.1	5.3 (+4.6)	5.6 (+11.3)	5.7 (+11.9)	5.9 (+15.4)	5.9 (+16.8)
100	0.8	0.9 (+11.2)	0.8 (+7.5)	0.8 (+7.4)	0.9 (+11.9)	0.9 (+11.6)
avg	30.8	31.1 (+0.8)	31.2 (+1.3)	31.1 (+0.9)	31.0 (+0.7)	30.9 (+0.4)

Table 11: Effect of δ values on performance of local context analysis on TREC4

9.3 Number of Concepts to Use

In the previous experiments, we added 70 concepts to each query. Adding so many concepts can significantly slow the retrieval process. We now examine how the performance of local context analysis is affected if we use fewer concepts for query expansion. We add 30 concepts with equal weights to each query. A property of the INQUERY #WSUM operator is that the fewer operands that are inside the operator, the larger the belief value it returns. To offset the larger belief value produced by the auxiliary query because of the decrease in the number of concepts used, we set the weight of the auxiliary query to 1.0 rather than the default 2.0. The retrieval performance is shown in Table 12. The performance using 30 concepts is close to that using 70 concepts. The performance at low recall points is better and the performance at high recall points is worse than using 70 concepts. This means that the concepts ranked below 30 are somewhat useful in retrieving more relevant documents but can bring non-relevant documents to the top of the ranked output.

	Precision (% change) – 49 queries		
Recall	baseline	lca-70-cpt	lca-30-cpt
0	71.0	73.2 (+3.2)	73.8 (+4.1)
10	49.3	57.1 (+15.7)	57.3 (+16.1)
20	40.4	46.8 (+16.0)	48.0 (+19.1)
30	33.3	39.9 (+19.8)	40.1 (+20.5)
40	27.3	35.3 (+29.1)	35.0 (+28.0)
50	21.6	29.9 (+38.4)	29.6 (+37.0)
60	14.8	23.6 (+59.8)	23.0 (+55.2)
70	9.5	17.9 (+89.1)	16.4 (+73.5)
80	6.2	11.8 (+91.0)	10.7 (+74.1)
90	3.1	5.7 (+80.2)	5.5 (+74.9)
100	0.4	0.8 (+88.2)	0.7 (+67.1)
average	25.2	31.1 (+23.5)	30.9 (+22.9)

Table 12: Using 70 concepts vs using 30 concepts on TREC4

10 Relevance Feedback

Retrieval results show that the feature selection strategy of local context analysis is superior to that of local feedback if we know nothing about the relevance of the top ranked documents. It is possible that this is still true even if we have perfect information about the relevance of the top ranked documents. In other words, can we use the feature selection strategy of local context analysis to improve the performance of relevant feedback? Experiments described below show that the answer is no.

We used 50 queries from the TREC5 and TREC6 topics to form a query set and used the Financial Times documents in TREC volume 4 as a training collection. For each query, 20 documents were retrieved from the training collection to form a training sample. Two versions of expanded queries were created. The first version was created by standard relevance feedback: The most frequent terms (excluding stop words) from the relevant documents in the training sample were used for query expansion. The second version used the feature selection strategy of local context analysis. That is, the expansion terms in the second version were chosen based on their co-occurrences with the query terms in the relevant passages in the training sample (a relevant passage is a passage from a relevant document). For a fair comparison, both versions used the same number of expansion terms (30 per query) and the same weighting method (Rocchio $\alpha : \beta : \gamma = 2 : 8 : 1$). Two test collections were used: one for queries from TREC5 and consisting of AP and WSJ documents in TREC volume 2, and the other for queries from TREC6 and consisting of FBIS and L.A. Times documents in TREC volume 5. Each query in the query set has more than 10 relevant documents in the training collection and in the test collection.

The retrieval results are shown in Table 13. While both versions are significantly better than the unexpanded queries, standard relevance feedback is noticeably (about 5%) better than local context analysis. The results show that automatic query expansion without relevance judgments and relevance feedback are quite different tasks and require different strategies for feature selection.

	Precision (% change) – 50 queries				
Recall	base	feedback by frequency		feedback by co-occurrence	
0	77.7	82.0	(+5.6)	82.9	(+6.8)
10	50.6	56.0	(+10.6)	56.9	(+12.2)
20	37.1	48.3	(+30.2)	48.4	(+30.4)
30	30.0	41.1	(+37.3)	37.9	(+26.7)
40	24.5	34.1	(+39.3)	31.0	(+26.9)
50	20.3	28.7	(+41.4)	24.8	(+22.2)
60	15.8	23.4	(+47.9)	19.8	(+25.6)
70	10.7	18.3	(+70.9)	15.4	(+44.0)
80	7.2	12.9	(+78.4)	10.6	(+47.1)
90	2.7	7.2	(+160.6)	6.4	(+133.4)
100	0.6	1.7	(+193.7)	1.9	(+225.7)
average	25.2	32.1	(+27.6)	30.6	(+21.3)

Table 13: Comparing term selection by frequency and term selection by co-occurrence for relevance feedback

11 Results on Chinese and Spanish Collections

We have shown that local context analysis works on English collections. We now show that it also works on collections in other languages. The experiments in this section were carried out using the Chinese and Spanish collections of the TREC5 conference. They were mostly carried out by fellow IR researchers at UMass for the TREC5 conference.[1] As a result, some parameter values are different from the ones used in the previous experiments.

The Chinese experiments were carried out on the TREC5-CHINESE collection. Query terms are Chinese words recognized by *Useg* [21], a Chinese segmenter based on the hidden Markov model. We define concepts as words in the documents recognized by the segmenter. Documents are broken into passages containing no more than 1500 Chinese characters. To expand a query, the top 30 concepts from the top 10 passages for the query are used. Concept i is assigned the weight

$$w_i = 1.0 - (i-1)/60$$

[1] Thanks to Lisa Ballesteros, who did all the Spanish experiments, and John Broglio and Hongmin Shu, who did the major part of the Chinese experiments.

The weight of the auxiliary query is set to 1.0. Table 14 shows the retrieval performance of local context analysis on the Chinese collection. The improvement over the baseline is 14% and is statistically significant (p_value=0.01).

The Spanish experiments were carried out on the TREC5-SPANISH collection. Concepts are noun phrases in the documents recognized by a part of speech tagger for Spanish. The passage size is 200 words. The top 31 concepts from the top 20 passages are added to each query. The top concept is given the weight 1.0 with all subsequent concepts downweighted by 1/100 for each position further down the rank. The weight of the auxiliary query is 1.0. Table 15 shows the retrieval performance of local context analysis on the Spanish collection. Local context analysis produces a 13% improvement over the unexpanded queries. The t-test indicates the improvement is statistically significant (p_value=0.005).

	Precision (% change) – 19 queries		
Recall	base	lca	
0	69.1	74.9	(+8.4)
10	56.8	60.7	(+6.8)
20	49.2	56.2	(+14.1)
30	43.1	48.5	(+12.4)
40	37.2	44.2	(+18.9)
50	33.2	36.3	(+9.3)
60	26.9	31.2	(+16.1)
70	20.1	26.3	(+31.2)
80	16.8	21.9	(+30.1)
90	12.5	14.5	(+16.2)
100	3.7	5.7	(+53.4)
average	33.5	38.2	(+14.0)

Table 14: Performance of local context analysis on TREC5-CHINESE.

	Precision (% change) – 25 queries		
Recall	base	lca	
0	85.5	79.6	(−7.0)
10	72.0	74.9	(+4.2)
20	55.2	66.8	(+21.1)
30	49.8	60.1	(+20.8)
40	45.4	51.2	(+12.9)
50	39.8	47.1	(+18.3)
60	33.5	40.5	(+20.8)
70	27.9	34.9	(+24.7)
80	22.0	28.2	(+28.1)
90	16.6	21.3	(+27.9)
100	1.8	3.5	(+86.9)
average	40.9	46.2	(+13.0)

Table 15: Performance of local context analysis on TREC5-SPANISH.

12 Cross Corpora Expansion

Local context analysis assumes that query words and their alternatives have some chance to co-occur in a collection. Put in another way, it assumes that vocabulary X and its alternative Y are used together in some documents D in a collection. When a user posts a query written in X, we search for it and find documents D. Then from documents D we find the alternative vocabulary Y and use it to expand the query. In fact, this is the assumption behind all query expansion techniques except for perhaps manual thesauri. But the assumption does not always hold. It sometimes happens that X and Y are not used together in any documents in a collection. For example, a reporter used the query "elderly black Americans" to search a collection of congressional bills and found no relevant documents for his query because politicians do not use "black Americans" to describe "African Americans" [10].

	Precision (% change) – 50 queries		
Recall	base	use-TREC5	use-newspaper
0	64.1	53.7 (−16.2)	60.2 (−6.1)
10	37.5	34.4 (−8.4)	41.3 (+10.0)
20	29.1	30.9 (+6.3)	34.7 (+19.2)
30	24.1	26.4 (+9.3)	28.6 (+18.4)
40	21.3	23.5 (+10.5)	24.7 (+16.4)
50	17.9	20.7 (+15.8)	21.8 (+21.8)
60	12.6	17.1 (+36.2)	18.5 (+47.3)
70	10.1	12.7 (+25.2)	15.3 (+50.9)
80	7.2	8.7 (+21.6)	9.9 (+37.9)
90	4.8	6.2 (+28.2)	6.8 (+42.4)
100	2.7	2.4 (−13.5)	2.7 (+0.0)
average	21.0	21.5 (+2.3)	24.1 (+14.3)

Table 16: Using a newspaper collection to expand TREC5 queries. 100 passages are used.

One method to address the above problem is to expand a query on a different collection which indeed uses alternative vocabularies in its documents. We have done an experiment to demonstrate this idea. The experiment was carried out on TREC5. But unlike in previous experiments, the TREC5 queries were expanded on a different collection. The collection consists all the newspaper articles in TREC volumes 1–5, totaling 3 GB, with sources Associated Press, Wall Street Journal, San Jose Mercury, Financial Times, Foreign Broadcast Information Service and L.A. Times. Since newspaper articles generally have a very wide audience, we conjecture that they would use different vocabularies and therefore be a good document source for query expansion. The conjecture is supported by the retrieval results in Table 16. Using the newspaper collection significantly improves the retrieval performance (14.3% over the baseline and 11.8% over using the native TREC5 collection for query expansion).

13 Very Short Queries

Although queries in some of the query sets used in previous sections are relatively short (see Table 1), queries in some applications are even shorter. For example, applications that provide searching across the World-Wide Web typically record average query lengths of two words [10]. In order to simulate retrieval in such applications, we did an experiment using a set of very short queries to search the TREC5 collections. The queries are the title fields of the TREC5 topics and mostly consists of a phrase, e.g., "gun control", "computer security" and so forth. The average query length after removal of stop words is only 3.2 words per query, which is close to the length of typical queries on the World Wide Web. In comparison, the average length of the TREC5 queries used in previous sections is 7.1 words per query. Since the queries are significantly shorter, word mismatch is a more serious problem. We conjecture that query expansion should produce more substantial improvement than using the longer queries.

The conjecture is supported by retrieval results in Table 17. The queries were expanded by local context analysis using the newspaper collection described in Section 12. Query expansion results in a substantial 24.9% improvement over the unexpanded queries. In comparison, query expansion results in smaller (14.3%) improvement for the long queries (Table 16). Without query expansion, the short queries are 19.4% worse than the long queries. This suggests that for the same information need, word mismatch is more serious and query expansion is more helpful for a short query than for a long query. We should note that the same is not necessarily true for different information needs, because the effectiveness of query expansion also depends on other factors. For example, the WEST queries are shorter than the TREC3 queries, but the WEST baseline is much better than the TREC3 baseline and query expansion is less helpful for the WEST queries than for the TREC3 queries. Further work is needed to find other factors affecting the effectiveness of query expansion.

	Precision (% change) – 50 queries		
Recall	title	title-lca	
0	53.2	57.5	(+8.1)
10	29.7	35.1	(+18.3)
20	23.7	29.3	(+23.4)
30	20.1	24.9	(+23.7)
40	17.5	21.9	(+25.3)
50	15.2	19.6	(+29.1)
60	10.8	16.5	(+53.1)
70	6.9	12.3	(+77.9)
80	4.9	8.4	(+71.0)
90	2.8	5.1	(+83.0)
100	1.8	2.5	(+40.5)
average	17.0	21.2	(+24.9)

Table 17: Using local context analysis to expand TREC5 title queries.

14 Efficiency and Optimization

In this section we discuss the computational costs of local context analysis. We should note that the main purpose of this paper is to demonstrate the usefulness of the technique and therefore efficiency is a secondary consideration in our implementation. Before we present the performance figures under the current implementation, it is important to know the issues and overheads when local context analysis is used in a production system.

In a production environment, local context analysis would be an integral part of the retrieval system rather than separate software. The only augmentation to the index structure is a dictionary which stores the frequencies of the concepts in a collection. At indexing time, we need to recognize the concepts as documents are parsed. Based on our current implementation, we estimate that concept recognition will increase the indexing time by at most 50%. At query time, the system retrieves the top ranked passages, parses them and collects the co-occurrences between concepts and query terms, and then ranks the concepts. Finally the system adds the best concepts to the query and performs a second retrieval. Most of the extra work at query time is likely to be the initial retrieval. If parsing the top ranked passages turns out to be the bottleneck, an option is to store the concepts in the index structure. That will slightly increase the storage overhead.

We now report the overheads under our prototype implementation. Experiments were carried out on a DEC alpha workstation. The TREC4 collection (2 GB of text) was used in the experiments. Passage size is 300 words. Our implementation requires a local context analysis database in order to carry out query expansion on a collection. The database for TREC4 takes about 0.67 GB, which is broken down as following:

- The concept dictionary, which stores the frequency of the concepts, 167 MB.
- The term dictionary, which stores the frequency of the terms, 43 MB.
- The concept file, which sequentially stores for each passage the concepts that occur in the passage and the numbers of occurrences, 251 MB.
- The term file, which is analogous to the concept file except it is for terms, 213 MB.

The time to build the local context analysis database for TREC4 is about 4 hours of wall clock time, most of which is spent on parsing and part of speech tagging. Currently we also need to index the passages in order for INQUERY to retrieve the top ranked passages for a query. The is simply a software artifact. With minor modification to the retrieval system, INQUERY can retrieve the top ranked passages without creating a separate index for passages.

Query expansion consists of two steps. In the first step, INQUERY retrieves the passage identifiers of the top ranked passages for a query. This step takes about 10 seconds per query. In the second step, concepts in the top ranked passages are ranked and the top ranked concepts are output for query expansion. This steps takes about 2 seconds per query when 100 passages are used. The total time to expand a query is about 12 seconds.

The memory usage under the current implementation is very high (200 MB) but can be easily cut to an acceptable level. The high memory usage is due to a large number of infrequent concepts in the concept dictionary. The TREC4 concept dictionary contains

4.9 million concepts, but most of them occur no more than a couple of times in the whole collection. Experimental results in Table 18 show that such concepts have limited, if any, impact on retrieval effectiveness. Filtering out those concepts occurring less than 5 times only affected retrieval performance by 0.7%. However, the size of the concept dictionary is reduced from 167 MB to 17 MB. Throwing out concepts occurring less than 10 times does not affect performance further and reduces the size of the concept dictionary to 10 MB.

	Precision (% change) – 49 queries				
Recall	no filter	frequency 5		frequency 10	
0	73.2	71.4	(−2.4)	71.4	(−2.4)
10	57.1	56.9	(−0.3)	56.9	(−0.3)
20	46.8	46.9	(+0.0)	46.9	(+0.0)
30	39.9	40.3	(+1.1)	40.3	(+1.1)
40	35.3	35.1	(−0.4)	35.1	(−0.5)
50	29.9	29.9	(−0.2)	29.9	(−0.2)
60	23.6	23.2	(−1.9)	23.2	(−1.9)
70	17.9	17.4	(−2.7)	17.4	(−2.7)
80	11.8	12.0	(+1.7)	12.0	(+1.7)
90	5.7	5.7	(−0.4)	5.7	(−0.4)
100	0.8	0.9	(+1.1)	0.9	(+1.1)
average	31.1	30.9	(−0.7)	30.9	(−0.7)

Table 18: Filtering low frequency concepts on the performance of local context analysis (TREC4)

15 Other Applications

Recently we and fellow IR researchers at UMass have applied local context analysis in other IR related problems and demonstrated promising results. One application is in distributed IR (Xu and Callan [35]). A critical problem for distributed IR is choosing the right collections to search for a query. This is usually done by comparing a query with the dictionary information of each collection (e.g, terms and their document frequency in the collection). However, this method does not work well for typical ad hoc queries because most of the query terms are not discriminatory enough for the purpose of separating good collections from bad ones. Local context analysis can find highly specific terms for a query and enhance the discriminatory power of the query.

The second application is in cross-language retrieval (Ballesteros and Croft [2]), where a query in one language must be translated in order to be used for retrieval in another language (e.g. English to Spanish). A major hindrance for effective cross-language retrieval is the poor translation caused by the ambiguity of the query terms. Local context analysis can provide very specific expansion concepts and as a result improve the quality of query translation and retrieval effectiveness.

The third application is in topic segmentation (Ponte and Croft [22]). The task of topic segmentation is to detect topic transitions in a text stream (e.g. a news feed) and break it into coherent documents. The commonly used technique is to compare two adjacent pieces of text (e.g sentences) to see whether they share any words. The assumption is that within topic sentences significantly share words and cross topic sentences do not. However, the assumption is often violated because of synonyms and word ambiguity. The solution is to treat the two pieces of text as two queries and expand them using local context analysis. Because the expansion concepts are related to the original texts, comparing the expanded texts results in more accurate detection of topic changes.

16 Conclusions and Future Work

In this paper we have proposed a new technique for automatic query expansion, called local context analysis, by combining the advantages of a global technique (Phrasefinder) and a local technique (local feedback). Experimental results on a number of collections show that the new technique is superior to both Phrasefinder and local feedback.

We will pursue the work in several directions. Firstly, the current function for concept selection, though works well, is mostly heuristically driven. We hope that a more theoretically driven function will further improve the retrieval performance. We are currently investigating several approaches including using language models for concept selection [20]. Secondly, we need to be more flexible in query expansion. Currently we expand every query, even though some queries are inherently ambiguous and the best strategy is no expansion at all. An ambiguous query typically retrieves several clusters of documents which match the query equally well. We hope to utilize this property to determine whether a query is ambiguous. For an ambiguous query, we can choose to not expand it or ask the user to refine it. Lastly, our method to assign weights to the expansion concepts is ad hoc and needs to be improved. We hope that a more theoretically driven function for concept selection will also lead to a better weighting method.

Acknowledgments

This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623. This material is also based on work supported in part by United States Patent and Trademark Office and Defense Advanced Research Projects Agency/ITO under ARPA order number D468, issued by ESC/AXS contract number F19628-95-C-0235. Any opinions, findings and conclusions or recommendations expressed in this material are the authors and do not necessarily reflect those of the sponsors.

References

[1] R. Attar and A. S. Fraenkel. Local feedback in full-text retrieval systems. *Journal of the Association for Computing Machinery*, 24(3):397–417, July 1977.

[2] Lisa Ballesteros. Phrasal translation and query expansion techniques for cross-language information retrieval. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 84–91, 1997.

[3] J. Broglio, J. P. Callan, and W.B. Croft. An overview of the INQUERY system as used for the TIPSTER project. In *Proceedings of the TIPSTER Workshop*, pages 47–67. Morgan Kaufmann, 1994.

[4] John Broglio, James P. Callan, W. Bruce Croft, and Daniel W. Nachbar. Document retrieval and routing using the INQUERY system. In D. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*, pages 22–29. NIST Special Publication 500-225, 1995.

[5] Chris Buckley, Gerald Salton, James Alan, and Amit Singhal. Automatic query expansion using SMART : TREC 3. In D. Harman, editor, *Proceedings of the TREC 3 Conference*, 1995.

[6] Chris Buckley, Amit Singhal, Mandar Mitra, and Gerard Salton. New retrieval approaches using SMART : TREC 4. In D. Harman, editor, *Proceedings of the TREC 4 Conference*, pages 25–48, 1996.

[7] William Caid, Susan Dumais, and Stephen Gallant. Learned vector-space models for document retrieval. *Information Processing and Management*, 31(3):419–429, 1995.

[8] J. P. Callan. Passage-level evidence in document retrieval. In *Proceedings of 17th ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 302–310, 1994.

[9] K. Church and P. Hanks. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th ACL Meeting*, pages 76–83, 1989.

[10] Bruce Croft, Robert Cook, and D. Wilder. Providing government information on the Internet: Experiences with THOMAS. In *Digital Libraries Conference DL'95*, pages 19–24, 1995.

[11] W. Bruce Croft and D. J. Harper. Using probabilistic models of document retrieval without relevance information. *Journal of Documentation*, 35:285–295, 1979.

[12] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[13] Edward A. Fox. *Extending the Boolean and Vector Space Models of Information Retrieval with P-Norm Queries and Multiple Concept Types.* PhD thesis, Cornell University, 1983.

[14] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, November 1987.

[15] George Furnas, Scott Deerwester, Susan Dumais, Thomas Landauer, Richard Harshman, Lynn Streeter, and Karen Lochbaum. Information retrieval using a singular value

decomposition model of latent semantic structure. In *Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 465–480, 1988.

[16] Marti Hearst. Mini-paragraph segmentation of expository discourse. In *Proceedings of 32nd Meeting of the Association for Computational Linguistics*, June 1994.

[17] David Hull. Using statistical testing in the evaluation of retrieval experiments. In *Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 329–338, 1993.

[18] Y. Jing and W.B. Croft. An association thesaurus for information retrieval. In *Proceedings of RIAO 94*, pages 146–160, 1994.

[19] Jack Minker, Gerald Wilson, and Barbara Zimmerman. An evaluation of query expansion by the addition of clustered terms for a document retrieval system. *Information Storage and Retrieval*, 8:329–348, 1972.

[20] Jay Ponte. Personal communications, 1998.

[21] Jay Ponte and Bruce Croft. USeg: A retargetable word segmentation procedure for information retrieval. In *Symposium on Document Analysis and Information Retrieval*, 1996.

[22] Jay Ponte and Bruce Croft. Text segmentation by topic. In *Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries*, pages 113–125, 1997.

[23] Yonggang Qiu and H. P. Frei. Concept based query expansion. In *Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval*, pages 160–169, 1993.

[24] J. J. Rocchio. *The SMART Retrieval System: Experiments in Automatic Document Processing*, chapter 14. Prentice-Hall, 1971.

[25] Gerald Salton. *Automatic Text Processing.* Addison Wesley, 1989.

[26] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41:288–297, 1990.

[27] Hinrich Schütze and Jan O. Pedersen. A cooccurrence-based thesaurus and two applications to information retrieval. In *Proceedings of RIAO 94*, pages 266–274, 1994.

[28] Amit Singal, Chris Buckley, and Mandar Mitra. Pivoted document length normalization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29, 1996.

[29] K. Sparck Jones. *Automatic Keyword Classification for Information Retrieval.* Butterworth, 1971.

[30] K. Sparck Jones and D.M. Jackson. The use of automatically-obtained keyword classifications for information retrieval. *Information Processing and Management*, 5:175–201, 1970.

[31] C. J. van Rijsbergen. *Information Retrieval.* Butterworths, second edition, 1979.

[32] Ross Wilkinson, Justin Zobel, and Ron Sacks-Davis. Similarity measures for short queries. In *Proceedings of the TREC 4 Conference*, 1996.

[33] Jinix Xu and Bruce Croft. Query expansion using local and global document analysis. In *Proceedings of the 19^{th} Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4–11, 1996.

[34] Jinxi Xu, John Broglio, and Bruce Croft. The design and implementation of a part of speech tagger for English. Technical Report IR52, CIIR, Computer and Information Science Department, University of Massachusetts, Amherst, MA 01003, 1994.

[35] Jinxi Xu and Jamie Callan. Effective retrieval with distributed collections. In *Proceedings of the 21^{th} Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998. To appear.