Improving Automatic Query Expansion

Mandar Mitra*
Cornell University
Ithaca, NY 14853.
mitra@cs.cornell.edu

Amit Singhal
AT&T Labs–Research
Florham Park, NJ 07932.
singhal@research.att.com

Chris Buckley
Sabir Research Inc.
Gaithersburg, MD 20878.
chrisb@sabir.com

Abstract Most casual users of IR systems type short queries. Recent research has shown that adding new words to these queries via *adhoc feedback* improves the retrieval effectiveness of such queries. We investigate ways to improve this query expansion process by refining the set of documents used in feedback. We start by using manually formulated Boolean filters along with proximity constraints. Our approach is similar to the one proposed by Hearst[12]. Next, we investigate a completely automatic method that makes use of term cooccurrence information to estimate word correlation. Experimental results show that refining the set of documents used in query expansion often prevents the query drift caused by blind expansion and yields substantial improvements in retrieval effectiveness, both in terms of average precision and precision in the top twenty documents. More importantly, the fully automatic approach developed in this study performs competitively with the best manual approach and requires little computational overhead.

1 Introduction

With the proliferation of the World Wide Web and the widespread use of Web search engines, the number of casual users of IR systems has grown. It is generally accepted that such users typically formulate very short queries and are not likely to take the trouble of constructing long and carefully stated queries. Such short queries lack words that, if provided by the user, can be very useful search terms. In order to achieve reasonable retrieval effectiveness on these short queries, we need special techniques; a straightforward keyword match may not always be adequate.

Automatic query expansion via *relevance feedback* is an effective technique commonly used to add useful words to a query [15, 18]. Unfortunately, casual users seldom provide a system with the relevance judgements needed in relevance feedback. In such situations, *adhoc* or *blind feedback* [6, 7, 14, 3] is commonly used to expand the user-query. This method takes the form of "pseudo" relevance feedback, where actual input from the user is not required. In this method, a small set of documents is retrieved using the original user-query; these documents are all *assumed* to be relevant without any intervention by the user. These assumed relevant documents are then used in a relevance feedback process to construct an expanded query, which is then run to retrieve the set of documents actually presented to the user.

This method has an obvious drawback: if a large fraction of the documents assumed relevant is actually non-relevant, then the words added to the query (drawn mostly from these documents) are likely to be unrelated to the topic and the quality of the documents retrieved using the expanded query is likely to be poor.

Consider query 203 from the TREC collection [9], for example: *What is the economic impact of recycling tires?* For this query, out of the first 20 documents retrieved by our system, only 4 are relevant. Most of the remaining documents discuss recycling of plastics, glass, etc. without referring to tires. When these 20 documents are used to expand the query via adhoc feedback, words like 'plastic', 'glass', etc. are added to the query, while words like 'rubber', 'car', and 'burn' (car tires are recycled into chips that can be burnt for energy) are not added to the query at all. The expanded query, therefore, begins to look more like a query about plastics and general recycling rather than tire recycling. In the final retrieved list, many more non-relevant documents dealing with plastics and general recycling appear at top ranks, and relevant documents discussing various ways to reuse old tires are ranked lower in the retrieved list. Thus, for this query, adhoc expansion results in a significant loss of performance.

On the other hand, if the initial retrieval quality is reasonably good, i.e. a good proportion of the initially retrieved documents are relevant, then the terms added to the query are mostly related to the search topic, and the expanded query matches more relevant documents and retrieves them at high ranks. Final results are therefore likely to improve.

Thus, adhoc feedback seems capable of both improving and hurting performance for different queries. In the experiments conducted for this study, we used 199 queries; the straightforward adhoc expansion method described above resulted in a drop in performance for about one-third of the queries. The performance improvements on the remaining queries are substantial however, and the net effect (averaged over all queries) is a significant increase in retrieval effectiveness (for more details, see Sections 4 and 5). Recent experiments by several groups participating in TREC also suggest that, averaged over large query sets (with 50 queries), adhoc expansion yields significant improvements in overall performance [10, 1, 4, 13]. Thus, it seems worthwhile to continue using adhoc expansion in retrieval for short queries, while exploring ways to prevent *query drift* — the alteration of the focus of a search topic caused by improper expansion (as described in the above example).

*This study was supported in part by the National Science Foundation under grant IRI-9624639.

Since the presence of a large proportion of non-relevant documents at top-ranks seems to be one of the main reasons for query drift, an obvious approach to preventing query drift would involve improving precision at top-ranks. One method for increasing precision is the following: (i) the initially retrieved documents are examined for additional, strong indications of relevance; (ii) a new score is assigned to each document based on the presence of these additional relevance indicators; (iii) the document set is reranked based on the new similarity. By promoting only those documents which contain strong relevance indicators, we are likely to eliminate non-relevant documents (which typically lack such indicators) from top-ranks, and precision should increase.

Incorporating this precision-enhancing step into the usual adhoc feedback process, we arrive at the following algorithm:

1. To use K (say 20) documents in the feedback process, retrieve a larger number T (say 50) of documents using the original user query.
2. For each retrieved document, compute a new similarity score Sim_{new} based on the occurrence of additional relevance indicators in the document.
3. Rerank the T retrieved documents based on Sim_{new}, breaking ties by the original similarities.
4. Select the top K documents in the new ranking and use them in the Rocchio relevance feedback process to expand the query.
5. Use the expanded query to retrieve the final list of documents returned to the user.

The reranking stage (steps 2 and 3) is expected to result in a higher proportion of relevant documents in the set of documents used for feedback. At the same time, we need to ensure that this process does not skew the feedback set towards relevant information of a particular kind. The expanded query should then be a high-quality, balanced representation of the search topic and the final retrieval results should improve.

We now need to devise an actual reranking algorithm by deciding what document features are to be used as additional relevance indicators and how Sim_{new} is to be calculated. In the rest of this study, we investigate various reranking algorithms for refining the set of feedback documents. In the next section, we describe how manually formulated Boolean filters can be used for this purpose. Section 3 presents some automatic approaches for achieving the same ends. Section 4 describes the experiments we conducted to test our methods and presents the results. Section 5 analyzes the query drift caused by the expansion methods used in this study. Finally, Section 6 concludes the paper.

2 Boolean Constraints

An examination of the top-ranked non-relevant documents for various queries shows that a commonly occurring cause of non-relevance among such documents is inadequate query coverage, i.e. the search topic consists of multiple aspects, only some of which are covered in these documents. For example, query 226 of the TREC collection asks: *Have large scale state allowed lotteries/gambling improved the state's financial conditions? Any reduction noted in property tax, state income tax, roads?* Several top-ranked non-relevant documents contain information about taxes and taxation, but not about the effect of state-run lotteries and casinos on taxes.

If we promote documents which address all the aspects of the query — these documents are more likely to be relevant — we will eliminate several non-relevant documents from top-ranks, thereby increasing precision.

In order to determine whether a document covers all aspects of a search topic, we can check whether the document satisfies certain Boolean constraints. In a recent study [12], Hearst proposed this approach and showed that it resulted in significant improvements in retrieval effectiveness. As shown in [12], the constraint for a given query may be expressed as a formula in conjunctive normal form as follows: each aspect of a query is represented by a set of words joined by the OR operator, and the different aspects are AND-ed together. For example, for query 226 (given above), one possible constraint is:

(lotteries OR gambling) AND (finance OR tax)

Documents that satisfy the constraint contain at least one word from each aspect of the query. In the reranking step of our algorithm, documents that pass the filter are ranked ahead of documents that don't, with ties broken by the original similarity scores. The highest ranked documents are then used for feedback.

Refinements

In addition to using Boolean constraints in the straightforward way described above, we also investigate the benefits of adding some simple refinements to the basic idea. In particular, we consider *proximity constraints* and *fuzzy* formulations of Boolean operators.

Proximity constraints. It is conceivable that certain long documents contain all the aspects of a query, but discuss the various aspects in scattered and unrelated contexts, and are therefore non-relevant. In order to eliminate such documents, we use proximity constraints in addition to Boolean filters, i.e., a document passes a filter if the multiple aspects not only appear in the document, but also occur close to each other, say, within a block of 100 words. This approach was also proposed by Hearst [12]. While Hearst focused on improving initial retrieval results, we extend her experiments by studying the effect of these improvements on query expansion. Recently, Xu and Croft [20] have also successfully used text passages in combination with adhoc feedback.

Fuzzy Boolean operators. In our experiments, we also investigate fuzzy interpretations of Boolean operators in addition to using the usual binary forms. In the fuzzy method, documents are given some credit for partially satisfying a constraint, i.e. when a document covers some aspects of a query but not all. More precisely, for a filter with M aspects such that the ith aspect is represented by n_i terms $t_{i1} \ldots t_{in_i}$

$$(t_{11} \text{ OR } \ldots \text{ OR } t_{1n_1}) \text{ AND } \ldots \text{ AND } (t_{M1} \text{ OR } \ldots \text{ OR } t_{Mn_M})$$

we use the following expression to compute the new score assigned to a document D:

$$Sim_{new}(D) = \sum_{i=1}^{M} \max_{j\in\{1\ldots n_i\}} idf(t_{ij}) \qquad (1)$$

where $idf(t_{ij})$ is the inverse document frequency of term t_{ij} if it occurs in D, and is 0 otherwise.

This formulation may be regarded as a simple special case of the *pnorm* method for extended Boolean operators [16, 17] and has the following benefits[1]:

- A document that contains several query aspects gets a higher score than a document that has fewer.
- If a document contains a specific term (as determined by its *idf* value) from a query aspect, it gets a higher score than a document that contains a more general representative of the same aspect.

Compared to the binary approach, this method makes a more fine-grained distinction between documents. We therefore expect this scheme to be more useful in the reranking step, especially for queries for which very many or very few documents satisfy the Boolean constraint.

3 Automatic Approaches

The Boolean filters described in the preceding section have to be constructed by humans. Unfortunately, lay users are not very competent at formulating appropriate Boolean filters for a natural language query. Also, the overhead involved in asking users to provide extra constraints in addition to their natural language queries may turn some users away from using a search system. We are therefore interested in completely automatic methods for refining the set of feedback documents, using only the initial query provided by the user.

We start with a crude first approximation to the manually constructed Boolean filters. In the absence of information provided by humans about what constitutes the various aspects of a query, we simply assume that each non-stop word in the query constitutes an individual aspect, and construct a filter automatically by AND-ing all the query words (excluding stop words) together. Using Equation 1 for this filter, we get the following expression for Sim_{new} for a document D:

$$Sim_{new}(D) = \sum_{t_i \in Q \wedge t_i \in D} idf(t_i) \qquad (2)$$

In other words, Sim_{new} is the sum of the idfs of query terms present in the document.

Using Term Correlations

It is easy to see that the method outlined above does not distinguish between multiple query-document matches on words related to the same aspect of the query and matches on words from different aspects of the query. Thus, in this approach, a match on a two-word phrase (or on two strongly related words) may be considered as useful as a match on two independent query concepts. For our purposes, however, a document that matches the query on multiple independent concepts is preferable. Therefore, we need to modify the above method by incorporating information about term relatedness into Equation 2.

To estimate the relatedness or independence of query words, we study their cooccurrence patterns in a large set S of documents (say 1000) initially retrieved for a query. If two words are correlated (or constitute a phrase-like structure), then they are expected to occur together in many of these documents. Given the presence of one of the words in a document, the chance of the other occurring within the same document is likely to be relatively high. On the other hand, if two words deal with independent concepts, the occurrences of the words should not be strongly correlated.

Using this idea, we modify the similarity computation in Equation 2 as follows. Let $df_S(t)$ be the number of documents in S that contain term t (this can be regarded as the "local" document frequency of t within a query zone). Then, given a query and a document, we consider the matching terms in increasing order of df_S. The first or "most rare" matching term contributes its full idf weight to Sim_{new}. The contribution of any subsequent match is deprecated depending on how strongly this match was predicted by a previous match — if a matching term is highly correlated to a previous match, then the contribution of the new match is proportionately down-weighted. More precisely, if $\{t_1, \ldots, t_m\}$ is the set of query terms present in document D (ordered by increasing df_S), then Sim_{new} is given by:

$$Sim_{new}(D) = idf(t_1) + \sum_{i=2}^{m} idf(t_i) \times \min_{j=1}^{i-1}(1 - P(t_i|t_j)) \qquad (3)$$

where $P(t_i|t_j)$ is estimated based on word occurrences in S and is given by

$$\frac{\textit{\# documents in } S \textit{ containing words } t_i \textit{ and } t_j}{\textit{\# documents in } S \textit{ containing word } t_j}$$

For example, consider TREC query 248: *What are some developments in electronic technology being applied to and resulting in advances for the blind?* The terms *electronic*, *technology* and *advance* are strongly related to each other. If the term *electronic* occurs in a document, the probability of *technology* occurring in the same document is high; given a match on *electronic*, a match on *technology* does not provide us with much additional information about a document. Accordingly, the contribution of *technology* to Sim_{new} is reduced in this case. On the other hand, terms *technology* and *blind* correspond to two independent aspects of the query and the occurrences of these two terms are relatively uncorrelated. Therefore, if a document contains these two terms, the contribution of *technology* is higher and it counts as an important new matching term since its occurrence is not well predicted by the other matching term (*blind*).

When no term correlations are used (Equation 2) in computing Sim_{new} for this query, a non-relevant document discussing ACM's plans to create an electronic archive of its literature is ranked much higher than a relevant document describing a new computer for the blind (owing to multiple matches in the non-relevant document on correlated words like *electronic*, *technology*, and *development*). If we incorporate term-relationship information by using Equation 3 instead, the relevant document gets a higher rank. The benefits of this approach are investigated more extensively in the next section.

4 Experiments and Results

In order to determine the usefulness of the techniques described above, we test them on a variety of tasks. We use the TREC collection in our experiments. Our methods are evaluated on the adhoc tasks for TRECs 3–6 [8, 9, 10, 11]. The query sets and document collections

[1] We conducted some preliminary experiments with the pnorm operators, but found the proposed method more useful because of its simplicity compared to the full power of pnorm operators.

Task	Queries	Documents
TREC 3	151 - 200	TREC disks 1, 2
TREC 4	202 - 250	TREC disks 2, 3
TREC 5	251 - 300	TREC disks 2, 4
TREC 6	301 - 350	TREC disks 4, 5

Table 1: Query and document sets for TREC adhoc tasks

Task	Measure	No Query Expansion	Query Expansion via adhoc fdbk.
TREC 3	Avg. P	0.2397	0.3335 (+39.1%)
	P@20	0.4850	0.5350 (+10.3%)
TREC 4	Avg. P	0.2297	0.3038 (+32.3%)
	P@20	0.4286	0.4755 (+10.9%)
TREC 5	Avg. P	0.1532	0.1944 (+21.2%)
	P@20	0.2830	0.3240 (+14.5%)
TREC 6	Avg. P	0.2040	0.2034 (- 0.3%)
	P@20	0.3400	0.3350 (- 1.5%)

Table 2: Improvements obtained by adhoc feedback

$w \rightarrow$ $T \downarrow$	50	100	200	full doc
TREC 3 (0.3335)				
50	0.3519	0.3549	0.3603	0.3715
	5.5%	6.4%	8.1%	11.4%
100	0.3646	0.3681	0.3807	0.3872
	9.3%	10.4%	14.2%	16.1%
200	0.3756	0.3869	0.3883	0.3932
	12.6%	16.0%	16.5%	17.9%
500	0.3867	0.3895	0.3931	**0.3948**
	16.0%	16.8%	17.9%	**18.4%**
TREC 4 (0.3038)				
50	0.3106	0.3117	0.3118	0.3138
	2.3%	2.6%	2.6%	3.3%
100	0.3148	0.3143	0.3173	0.3170
	3.6%	3.5%	4.5%	4.4%
200	0.3154	0.3144	0.3169	0.3181
	3.8%	3.5%	4.3%	4.7%
500	0.3117	0.3107	0.3140	**0.3190**
	2.6%	2.3%	3.4%	**5.0%**
TREC 5 (0.1944)				
50	0.1955	0.1965	0.1968	0.1983
	0.6%	1.1%	1.2%	2.0%
100	0.1968	0.1971	0.1960	0.1987
	1.2%	1.4%	0.8%	2.2%
200	0.1963	0.1958	0.1969	0.2011
	1.0%	0.7%	1.3%	3.5%
500	0.1980	0.1985	0.2023	**0.2031**
	1.9%	2.1%	4.1%	**4.5%**
TREC 6 (0.2034)				
50	0.2101	0.2104	0.2151	0.2155
	3.3%	3.5%	5.7%	5.9%
100	0.2174	0.2196	0.2220	0.2236
	6.9%	8.0%	9.1%	9.9%
200	0.2209	0.2248	0.2270	0.2329
	8.6%	10.5%	11.6%	14.5%
500	0.2270	0.2305	0.2343	**0.2392**
	11.6%	13.3%	15.2%	**17.6%**

Table 3: Feedback results using binary Boolean filters

used in these tasks are shown in Table 1. Since we are interested in studying short queries, we use only the "Description" field for queries 151–300. For the TREC-6 queries, we use the "Title" field in addition. For these queries, the use of the "Description" field alone is problematic. See [11] for a discussion of this issue.

Our experiments use the SMART IR system. We start with the standard Smart *Lnu.ltu* run [5, 4, 2]: documents and queries are indexed using single terms and statistical phrases; term-weights are computed using the *Lnu.ltu* weighting scheme proposed by Singhal et al. [19]; and 1000 documents are retrieved for each query using a simple vector inner-product similarity measure. The top 20 documents are assumed to be relevant, documents ranked 501-1000 are assumed non-relevant and these documents are used in the Rocchio feedback process [15, 18]. 25 words and 5 phrases are added to the original query; $\alpha = 8, \beta = 8, \gamma = 8$ are used as the Rocchio parameters. The new query is used to retrieve the final set of 1000 documents. This adhoc feedback scheme and the parameters used have been found to work well in previous experiments [4, 2] and is therefore adopted as our standard baseline query expansion scheme against which the other reranking-based expansion methods are compared. Table 2 shows that this method yields large improvements in non-interpolated average precision, as well as in precision at a 20 document cutoff, for three of the four tasks. For the TREC-6 task, this method does not significantly affect performance.

Next, we alter the above method by introducing a reranking step prior to the expansion step. The first T documents from the initial retrieval are reranked as described in Section 1 and the new set of 20 top-ranked documents are used as relevant documents in the feedback stage described above. The overall method remains the same otherwise. We try both the Boolean reranking method (described in Section 2) as well as the automatic one (Section 3). The details for each reranking technique and a comparison of the final results obtained using these methods are given below.

4.1 Boolean Constraints

The Boolean constraints for the 199 queries used in our experiments were constructed by one of the authors. The constraints contain single words from the original query only; no new terms are added. We use both binary and fuzzy Boolean operators with these filters. We also study the effect of varying T, the number of top-ranked documents reranked to select the final set of documents used in feedback. In addition, we experiment with proximity constraints — for binary operators, a document passes the filter only if the user-specified constraint is satisfied by a contiguous block of w words; for fuzzy operators, the new score of a document is the score of the highest scoring w-word window contained in it.

The performance of the final feedback run (measured using non-interpolated average precision) for various parameters are shown in Tables 3 and 4. The tables also show the average precision for the baseline expansion method (in parentheses) as well as %-age improvements over this baseline figure for various parameter settings. The figures for the best settings are highlighted.

The improvements obtained by using binary Boolean filters for refining the feedback set range from 4.5% on the TREC-5 task to a very substantial 18.4% for the TREC-3 task. When binary operators are replaced by fuzzy operators, the improvements range from about 7% on the TREC-4 task to as much as 24% on the TREC-6 task. From the tables, we can make the following additional observations:

Fuzzy Boolean Operators. The fuzzy form of the

$w \rightarrow$ $T \downarrow$	50	100	200	full doc
TREC 3 (0.3335)				
50	0.3711	0.3686	0.3656	0.3725
	11.3%	10.5%	9.6%	11.7%
100	0.3756	0.3765	0.3792	0.3826
	12.6%	12.9%	13.7%	14.7%
200	0.3895	0.3888	0.3891	0.3924
	16.8%	16.6%	16.7%	17.7%
500	0.3968	**0.3977**	0.3965	0.3955
	19.0%	**19.3%**	18.9%	18.6%
TREC 4 (0.3038)				
50	0.3162	0.3171	0.3170	0.3213
	4.1%	4.4%	4.3%	5.8%
100	0.3204	0.3226	0.3220	0.3239
	5.5%	6.2%	6.0%	6.6%
200	0.3217	0.3225	0.3234	**0.3253**
	5.9%	6.2%	6.5%	**7.1%**
500	0.3202	0.3183	0.3183	0.3203
	5.4%	4.8%	4.8%	5.4%
TREC 5 (0.1944)				
50	0.1987	0.2017	0.2020	0.2046
	2.2%	3.8%	3.9%	5.2%
100	0.2104	0.2133	0.2094	0.2113
	8.2%	9.7%	7.7%	8.7%
200	0.2068	0.2043	0.2054	0.2097
	6.4%	5.1%	5.7%	7.9%
500	0.2092	0.2118	0.2099	**0.2142**
	7.6%	8.9%	8.0%	**10.2%**
TREC 6 (0.2034)				
50	0.2298	0.2302	0.2316	0.2312
	13.0%	13.2%	13.9%	13.7%
100	0.2389	0.2446	0.2439	0.2439
	17.4%	20.2%	19.9%	19.9%
200	0.2469	0.2477	0.2458	0.2489
	21.4%	21.8%	20.8%	22.4%
500	**0.2517**	0.2496	0.2413	0.2461
	23.7%	22.7%	18.6%	21.0%

Table 4: Feedback results using fuzzy Boolean filters

Boolean operators works much better than the binary form. Comparing corresponding cells in Tables 3 and 4, we find that improvements obtained over the baseline expansion run using the fuzzy operators are usually higher.

This supports our intuition behind the use of fuzzy operators. Since the filters are not always "perfect", a document could fail to pass the filter and still be relevant. A document that narrowly fails to satisfy the specified constraint is more likely to be relevant than one that does not satisfy the constraint even partially. The binary method does not distinguish between these two types of documents, however. This is especially a problem when too few (or too many) documents pass the filter. In contrast, by assigning partial credit to documents that partially satisfy the constraints, the fuzzy method makes a more fine-grained distinction between documents. Therefore, the ranking produced by the fuzzy method is typically more useful.

Proximity Constraints. Breaking a document into w-word windows and using proximity constraints does not seem to be particularly helpful; using entire documents while computing Sim_{new} works well. Even when using the entire document is not optimal, it is quite close to the best window run.

As explained in Section 2, the proximity constraint is useful for eliminating documents in which query words appear in unrelated contexts. It turns out, however, that the top-ranked documents in our experiments are fairly homogeneous — if two key concepts cooccur in a document, they usually appear in the same context. Multi-topic documents (news articles containing brief items on various subjects, for example) are not all that common. Thus, proximity information does not prove to be very useful, and using full documents yields good results.

Number of reranked documents. Reranking a larger number of documents before selecting the feedback documents yields improvements in performance. For each task, reranking 200 to 500 documents is better than reranking just 50 or 100.

For a given query, as the number of documents reranked increases, the number of relevant documents present in the reranked set also increases. Since the manually formulated constraints are fairly accurate, our reranking method successfully promotes these relevant documents to top-ranks and eliminates highly ranked non-relevant documents. Thus, the concentration of useful documents in the set of feedback documents increases, resulting in improved performance. We do observe diminishing returns as we rerank more and more documents.

$w \rightarrow$ $T \downarrow$	50	100	200	full doc
TREC 3 (0.3335)				
50	0.3451	0.3468	**0.3475**	0.3389
	3.5%	4.0%	**4.2%**	1.6%
100	0.3355	0.3450	0.3383	0.3298
	0.6%	3.4%	1.4%	-1.1%
200	0.3402	0.3428	0.3382	0.3241
	2.0%	2.8%	1.4%	-2.8%
TREC 4 (0.3038)				
50	0.3118	**0.3134**	0.3124	0.3127
	2.6%	**3.2%**	2.8%	2.9%
100	0.3088	0.3124	0.3133	0.3123
	1.7%	2.9%	3.1%	2.8%
200	0.3015	0.2919	0.2883	0.2912
	-0.7%	-3.9%	-5.1%	-4.2%
TREC 5 (0.1944)				
50	0.2081	0.2060	0.2016	**0.2154**
	7.1%	6.0%	3.7%	**10.8%**
100	0.2061	0.1976	0.1987	0.2048
	6.0%	1.6%	2.2%	5.4%
200	0.2065	0.1941	0.2038	0.2010
	6.2%	-0.1%	4.9%	3.4%
TREC 6 (0.2034)				
50	0.2212	0.2256	0.2271	0.2222
	8.7%	10.9%	11.6%	9.3%
100	0.2352	**0.2446**	0.2382	0.2363
	15.7%	**20.3%**	17.1%	16.2%
200	0.2423	0.2425	0.2386	0.2362
	19.1%	19.2%	17.3%	16.1%

Table 5: Feedback results using automatic filters without term correlation information

4.2 Automatic Methods

As with the Boolean filters, in our experiments with the proposed automatic filtering schemes, we confine ourselves to using single terms only (i.e. phrase matches between the query and a document are not considered in Equations 2 and 3). Once again, proximity constraints are used in these experiments — a document is broken into overlapping blocks of w words and the score given to a document is the score of the best-matching w-word window. Different values of w are tried. We also vary T (the number of documents reranked). The final results for automatic filters which do not use any term correlation information (described by Equation 2) are shown in Table 5. Table 6 corresponds to Equation 3; this set of runs makes use of term correlation information.

$w \rightarrow$ $T \downarrow$	50	100	200	full doc
TREC 3 (0.3335)				
50	0.3587	0.3581	0.3560	0.3525
	7.6%	7.4%	6.8%	5.7%
100	0.3575	**0.3634**	0.3523	0.3305
	7.2%	**9.0%**	5.6%	-0.9%
200	0.3578	0.3520	0.3470	0.3166
	7.3%	5.6%	4.1%	-5.1%
TREC 4 (0.3038)				
50	**0.3208**	0.3195	0.3196	0.3168
	5.6%	5.2%	5.2%	4.3%
100	0.3109	0.3159	0.3084	0.3164
	2.3%	4.0%	1.5%	4.2%
200	0.2867	0.2867	0.2908	0.2937
	-5.6%	-5.6%	-4.3%	-3.3%
TREC 5 (0.1944)				
50	**0.2116**	0.2082	0.2036	0.2093
	8.9%	7.1%	4.8%	7.7%
100	0.2069	0.2036	0.1964	0.1990
	6.4%	4.7%	1.0%	2.4%
200	0.2067	0.1956	0.1947	0.2052
	6.3%	0.6%	0.2%	5.6%
TREC 6 (0.2034)				
50	0.2301	0.2346	0.2295	0.2272
	13.1%	15.3%	12.8%	11.7%
100	**0.2556**	0.2499	0.2474	0.2417
	25.6%	22.9%	21.7%	18.8%
200	0.2550	0.2503	0.2455	0.2342
	25.3%	23.1%	20.7%	15.2%

Table 6: Feedback results using automatic filters with term correlation information

The trends in the results for automatic filters are somewhat different from those for the manual ones.

Number of reranked documents. Reranking a smaller number of documents (50 to 100) produces better results. When T is 50 or 100, the reranked set contains a reasonable proportion of relevant documents, and most documents have a fairly strong match with the query. Under these circumstances, the automatic methods work well. When T is increased to 200 or 500, the manual algorithm usually successfully detects the small number of additional relevant documents while eliminating the non-relevant documents. In contrast, the automatic method is less accurate in distinguishing relevance from non-relevance: while the manual method uses human knowledge to determine term relationships, the automatic method uses heuristics (which are subject to error); further, the user uses only the core content-bearing terms from the query to construct the manual filter, whereas the automatic filters contain all non-stop query words, including words like 'impact' (query 203), 'developments' (query 248), etc. that are fairly general and may falsely promote non-relevant documents. Given these weaknesses, adding a large number of non-relevant documents to the reranked pool while adding only a few additional relevant documents increases the likelihood of error. Thus, with the automatic method, a more conservative approach of only reranking the top 50 or 100 documents is safer as it reduces susceptibility to error.

Proximity constraints. Using proximity constraints with small text windows ($w = 50$ or 100) is helpful. Once again, this shows the need to be conservative with the less precise automatic methods. Even when our automatic methods make mistakes, it is generally true that if multiple query words appear close to each other in a document, the chances of relevance of the document are higher. Therefore, using proximity information contains the damage that an imprecise automatic reranker could have caused.

Term correlation. The intuition behind using term correlation information seems to be correct. The refined automatic method almost always results in improved performance compared to the naive approach. This indicates that our term correlation based selection of different aspects of a query is working as desired.

4.3 Manual vs. Automatic Methods

To summarize, we compare the manual and automatic methods across various tasks in Table 7. This table shows the final (post expansion) average precision, and precision at a 20 document cutoff for each technique on each task. It also shows the corresponding improvements obtained over the baseline expansion method. Since it is not possible to fine tune parameters in an adhoc setting, we choose a set of parameters for each technique that results in reasonably good performance across different tasks. Specifically, we choose the following parameters:

Boolean: Fuzzy operators, top 200 documents reranked, full documents considered (no proximity used).

Naive automatic: Top 50 documents reranked, similarity based on best 100-word window in document.

Automatic: (with term correlation) Top 50 documents reranked, similarity based on best 50-word window.

As expected, the manual method produces substantial improvements for all tasks. What is encouraging is that it is possible to achieve reasonable improvements using a naive automatic approach that does not require any user intervention. By making use of additional statistical information about term correlation, we get further improvements, and in fact, for certain tasks, the performance of the automatic method is at par with the manual one. This result is especially encouraging in light of the fact that casual users are usually not willing to put in the extra effort involved in the manual methods, i.e. providing the system with a carefully chosen Boolean constraint. The fact that the improvements shown in Table 7 are over the baseline expansion method, which in itself is far better than no expansion (see Table 2), makes these improvements even more significant.

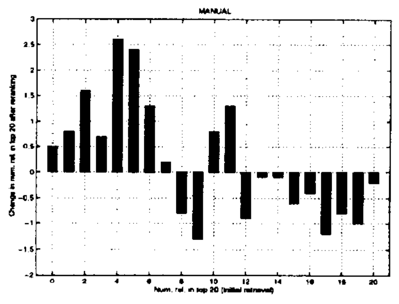
5 Query Drift

Recall that our reranking approach was intended to prevent query drift caused by blind expansion, mainly by improving the precision in the set of documents used for feedback. In this section, we examine the issue of query drift in greater detail.

We expect query drift to be most severe for queries that initially retrieve few relevant documents in the top-ranks. On the other hand, query drift should not occur for queries retrieving a good number of relevant documents at top-ranks. To study the relationship between

Task	Measure	Baseline	Fuzzy Boolean	Auto (no corr)	Auto with corr
TREC 3	Avg. P	0.3335	0.3924 (18%)	0.3468 (4%)	0.3587 (8%)
	P@20	0.5350	0.6280 (17%)	0.5620 (5%)	0.5710 (7%)
TREC 4	Avg. P	0.3038	0.3253 (7%)	0.3134 (3%)	0.3208 (6%)
	P@20	0.4755	0.5112 (8%)	0.4847 (2%)	0.5020 (6%)
TREC 5	Avg. P	0.1944	0.2097 (8%)	0.2060 (6%)	0.2116 (9%)
	P@20	0.3240	0.3250 (0%)	0.3160 (-3%)	0.3200 (-1%)
TREC 6	Avg. P	0.2034	0.2489 (22%)	0.2256 (11%)	0.2301 (13%)
	P@20	0.3350	0.3870 (16%)	0.3580 (7%)	0.3660 (9%)

Table 7: Comparison of feedback results (AvgP and P@20) for various reranking techniques



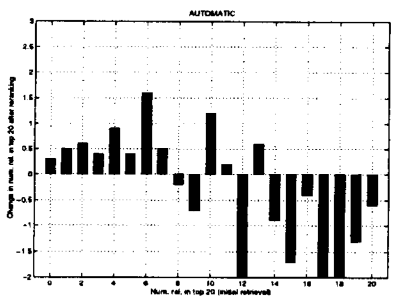


Figure 1: Effect of MANUAL and AUTOMATIC reranking on precision at 20 docs. (*no expansion*)

query drift and initial precision, we partition the 199 queries used in our experiments into bins corresponding to the number of relevant documents retrieved by the *initial* query within the top 20 ranks. Thus, the first bin contains the queries which retrieve no relevant documents in the top 20, and the last bin contains queries for which all top 20 documents are relevant.

We first check if our reranking algorithms are indeed improving precision in the top-ranks. This is done by computing the average number of relevant documents in the *refined* feedback set (i.e. the top 20 documents *after the reranking step*) for the set of queries in each bin. We plot histograms with the x-axis representing the query bins, and on the y-axis we plot the difference in the average number of relevant documents contained in the original and in the refined feedback set for queries in that bin. A bar above the x-axis indicates that the post-reranking P@20 is better than the initial P@20. A bar below the x-axis indicates that reranking hurts P@20.

Figure 1 shows the histograms corresponding to the manual (fuzzy Boolean) and automatic reranking (using term correlations) techniques. For the first several bins, the bars are above the x-axis, showing that the reranking step does indeed result in an increased concentration of relevance in the feedback set for these queries. Since these queries retrieve very few relevant documents in the initial 20, adding one or two new relevant documents can noticeably improve feedback effectiveness. For the rightmost bins, reranking results in a slight decrease in the number of relevant documents in the feedback set, but since the queries in these bins retrieve a large number of relevant documents to start with, losing one or two of them does not greatly affect feedback effectiveness.

Figure 2 shows how these changes affect query expansion. We generate the histograms shown in the figure as follows. For each bin, the average precision (averaged over all the queries in that bin) is computed for the initial retrieval and the final (post-feedback) run. The percentage difference between these two figures is plotted on the y-axis. A bar above the x-axis shows that there was an improvement in average precision due to expansion. A bar below the x-axis points to the contrary. The leftmost histogram corresponds to our baseline expansion scheme. The other histograms correspond to runs using manual and automatic reranking.

The baseline histogram confirms that query drift is indeed a problem, but perhaps not as severe as feared. For the queries in the first few bins, expansion generally hurts performance, as shown by the negative change in average precision[2]. When the number of relevant documents in the feedback set reaches 5 or 6, however, expansion turns out to be useful on average (though for some queries, expansion still results in poorer performance).

The other histograms show how both the manual and automatic reranking methods reduce query drift: in these graphs, there are no longer any bars below the x-axis. Thus, the improvement in precision due to reranking for the queries in the first few bins (Figure 1), seems to have had the desired effect of reducing the query drift caused by blind expansion. The queries in the rightmost bins continue to benefit from expansion due to the high proportion of relevant documents in the feedback set for these queries (despite the slight loss in the number of relevant documents in the feedback set due to reranking; see Figure 1).

More detailed information about query drift is contained in Table 8. This table shows the number of queries in each bin along with the number of queries for which expansion yields poorer retrieval (in terms of average precision). Numbers are shown for the baseline expansion run, as well as expansion runs that use reranking. As expected, the proportion of queries adversely affected by expansion is high in the initial bins and almost uniformly 0 for the last few bins. Thus, out of the 19 queries that initially retrieve only two relevant documents within the top

[2]The very first bin is ignored as an outlier. The queries in this bin perform very poorly and even a random improvement seems to be enormous.

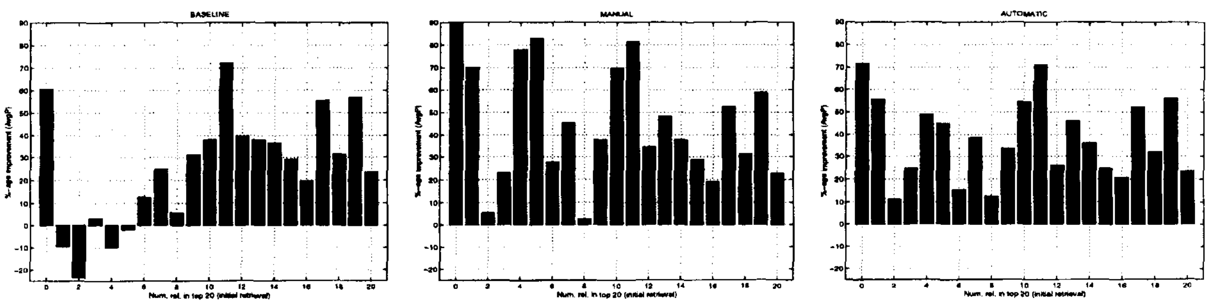


Figure 2: Query drift in terms of average precision — BASELINE, MANUAL, and AUTOMATIC runs

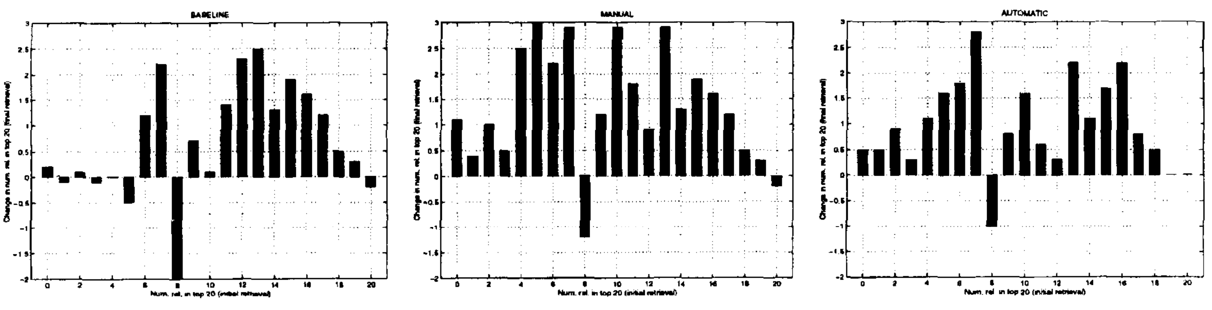


Figure 3: Query drift in terms of precision at 20 docs. — BASELINE, MANUAL, and AUTOMATIC runs

Bin	Num.	Num. queries hurt by expansion		
#	queries	Base	Manual	Automatic
0	10	5	3	4
1	17	7	6	5
2	19	12	6	9
3	14	6	4	4
4	15	9	2	4
5	14	8	4	4
6	12	3	3	4
7	14	5	5	4
8	5	2	2	2
9	6	1	2	1
10	8	2	1	1
11	9	1	2	2
12	7	0	0	3
13	10	0	0	0
14	7	2	2	1
15	7	0	0	0
16	9	1	1	2
17	4	0	0	0
18	4	0	0	0
19	3	0	0	0
20	5	0	0	0
Total	199	64	43	50

Table 8: Query drift caused by expansion

20, 12 deteriorate on straightforward expansion. When the manual reranking step is introduced, expansion hurts only 6 queries. The reranking step thus successfully prevents query drift on 6 queries. It is encouraging to note that some queries which initially retrieve only a few relevant documents benefit by expansion when the set of feedback documents is appropriately chosen by reranking. Also, when queries suffer in spite of a reranking step, the losses are generally smaller than the losses caused by blind expansion.

Comparison with LCA. Xu and Croft report a similar query-by-query analysis for LCA (local context analysis) [20], a technique for improving adhoc feedback, on the TREC-4 task. On this task, the LCA-based feedback method yields an 11-pt. average precision of 0.31 and hurts only 11 of the 49 queries. For the 9 "bad" queries (queries with initial average precision below 5%) in the set, LCA-based expansion hurts only 4. On the same task, our baseline expansion scheme (without reranking) performs as well as LCA — it yields an 11-pt. average precision of 0.32, hurts 12 out of 49 queries, and 3 out of 9 "bad" queries; the reranking based methods do rather better.

Most casual users are more interested in precision at top ranks and a measure like precision in top 20 documents is more meaningful for them. Figure 1 shows that reranking alone increases P@20 for queries that have poor initial precision but for queries that initially have a reasonable number of relevant documents in the top 20, P@20 is poorer post reranking. To show the effect of expansion on P@20, Figure 3 compares P@20 for the initial retrieval and all post-expansion runs—baseline, manual reranking, and automatic reranking.

As expected, for the queries in the first few bins, straightforward query expansion hurts P@20. But when a reranking step is introduced before expansion, almost all the bars are above the x-axis (except for bin 8 — for each method, P@20 is not significantly affected for four of the five queries in this bin, but a sharp drop in P@20 for one query causes the overall change to be negative). More interestingly, even though the reranking step results in a poorer P@20 for the last few bins (see Figure 1), P@20 for these bins improves on query expansion. This indicates that reranking alone is not enough for higher precision. Reranking should be followed by query expansion to obtain better results. Also, the improvements are larger and more consistent when a reranking step is used, once again demonstrating that the reranking step helps to reduce query drift. The competitive performance of our fully automatic method reinforces that automatic methods can be used without any user intervention to get retrieval effectiveness comparable to manual methods.

6 Conclusions

In this study, we explore ways to improve automatic query expansion via adhoc feedback. We focus on trying to prevent query drift, one of the major problems that adhoc feedback is susceptible to. We start by attempting to refine the set of feedback documents using manually formulated Boolean filters that specify the different independent aspects of a query. Next, we investigate a completely automatic approach to the problem. This approach makes use of term cooccurrence information to estimate word correlation and to identify independent concepts present in a given query.

We evaluate our techniques on the adhoc tasks for TRECs 3–6. Both the manual and the automatic methods consistently improve performance on the different tasks. While the manual approach increases retrieval effectiveness by 7 to 22% as compared to blind expansion, the automatic method also performs very creditably, effectively reducing query drift and resulting in significant improvements in retrieval effectiveness, ranging from 6 to 13%. Not only do results improve in terms of average precision, expansion using our automatic approach also yields a higher precision in the top twenty documents. What makes our method more attractive is the fact that it imposes only a small computational overhead on the usual adhoc feedback process. In future, we would like to develop better reranking methods that characterize the various aspects of a query more accurately.

References

[1] M.M. Beaulieu, M. Gatford, X. Huang, S.E. Robertson, S. Walker, and P. Williams. Okapi at TREC-5. In *Proc. of the Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication 500-238, 1997.

[2] C. Buckley, M. Mitra, J. Walz, and C. Cardie. Using Clustering and SuperConcepts within SMART: TREC6. In *Proc. of the Sixth Text REtrieval Conference (TREC-6)*, To appear.

[3] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic Query Expansion using SMART: TREC-3. In *Proc. of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication 500-225, 1995.

[4] C. Buckley, A. Singhal, and M. Mitra. Using Query Zoning and Correlation within SMART: TREC5. In *Proc. of the Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication 500-238, 1997.

[5] C. Buckley, A. Singhal, M. Mitra, and (G. Salton). New Retrieval Approaches using SMART: TREC-4. In *Proc. of the Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication 500-236, October 1996.

[6] E. Efthimiadis and P. Biron. UCLA-Okapi at TREC-2: Query Expansion Experiments. In *Proc. of the Second Text REtrieval Conference (TREC-2)*. NIST Special Publication 500-215, 1994.

[7] D. Evans and R. Lefferts. Design and Evaluation of the CLARIT-TREC-2 system. In *Proc. of the Second Text REtrieval Conference (TREC-2)*. NIST Special Publication 500-215, 1994.

[8] D.K. Harman. Overview of the Third Text REtrieval Conference (TREC-3) . In *Proc. of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication 500-226, 1995.

[9] D.K. Harman. Overview of the Fourth Text REtrieval Conference (TREC-4) . In *Proc. of the Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication 500-236, October 1996.

[10] D.K. Harman. Overview of the Fifth Text REtrieval Conference (TREC-5) . In *Proc. of the Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication 500-238, 1997.

[11] D.K. Harman. Overview of the Sixth Text REtrieval Conference (TREC-4) . In *Proc. of the Sixth Text REtrieval Conference (TREC-6)*. NIST Special Publication, To appear.

[12] M.A. Hearst. Improving Full-Text Precision on Short Queries using Simple Constraints. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*, April 1996.

[13] K.L. Kwok and L. Grunfeld. TREC-5 English and Chinese Retrieval Experiments using PIRCS. In *Proc. of the Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication 500-238, 1997.

[14] S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proc. of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication 500-225, 1995.

[15] J.J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System—Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall, Inc., 1971.

[16] G. Salton, C. Buckley, and E.A. Fox. Automatic query formulations in information retrieval. *Journal of the American Society for Information Science*, 34(4):262–280, July 1983.

[17] G. Salton, E.A. Fox, and H. Wu. Extended boolean information retrieval. *Communications of the ACM*, 26(12):1022–1036, December 1983.

[18] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.

[19] A. Singhal, C. Buckley, and M. Mitra. Pivoted Document Length Normalization. In *Proc. ACM SIGIR*, 1996.

[20] J. Xu and W.B. Croft. Query Expansion Using Local and Global Document Analysis. In *Proc. ACM SIGIR*, 1996.