Relevance Feedback and Inference Networks

David Haines & Bruce Croft

Computer Science Technical Report 93-31
April 1993

Relevance Feedback and Inference Networks *

David Haines and W. Bruce Croft

Department of Computer Science
University of Massachusetts
Amherst MA 01003
internet: {haines,croft}@cs.umass.edu

Abstract

Relevance feedback, which modifies queries using judgements of the relevance of a few, highly-ranked documents, has historically been an important method for increasing the performance of information retrieval systems. In this paper, we extend the inference network model introduced by Turtle and Croft to include relevance feedback techniques. The difference between relevance feedback on text abstracts and full text collections is studied. Preliminary results for relevance feedback on the structured queries supported by the inference net model are also reported.

1 Introduction

Relevance feedback methods in information retrieval attempt to improve performance for a particular query by modifying the query, based on the user's reaction to the initial retrieved documents. Specifically, the user's judgements of the relevance or non-relevance of some of the documents retrieved are used to add new terms to the query and to reweight query terms. For example, if all the documents, that the user judges as relevant, contain a particular term, then that term may be a good one to add to the original query [16]. Perhaps the relative importance of that term should also be increased.

Given the apparent effectiveness of relevance feedback techniques [15, 5], it is important that any proposed model of information retrieval includes these techniques. The inference net model recently described by Turtle and Croft [19, 20] has been shown to be an effective and general basis for an information retrieval system. One of the purposes of this paper is to show how feedback techniques can be used with this model. This includes both simple techniques, as described by Salton and Buckley [15], and techniques that exploit the ability of the inference net model to represent structure in the query.

The other major topic addressed in this paper is the effect of full text collections on relevance feedback techniques. Virtually all of the previous relevance feedback experiments have been done using collections of document abstracts. Full text collections are becoming increasingly important, and there is the possibility that the increased amount of text in the identified relevant documents will make the selection and weighting of terms more difficult.

1.1 Prior Work

Work on relevance feedback methods in information retrieval has a long history [16, 4]. Rocchio [13] describes an elegant approach to relevance feedback in the vector space model. He shows how the optimal vector space query can be derived using vector addition and subtraction if the sets of relevant and non-relevant documents are known. Of course the optimal query cannot be derived, as the full sets of relevant and non-relevant documents are not available. Relevance feedback judgments can, however, provide an approximation to these sets that, empirically, does improve performance. New terms are added to query by adding terms found in the relevant documents. The importance of query terms is adjusted by adding and subtracting corresponding weights found in relevant and non-relevant documents.

In a recent paper, Salton and Buckley [15] report the results of a number of relevance feedback techniques on a variety of document collections. In all but one collection, Salton and Buckley found average increases in precision, averaged over five collections, ranging from 60% to 90%.[1] For two collections they found increases of 170%. Little difference in performance was found between adding a subset of the terms or adding all the terms from relevant documents.

Harman [5] reported relevance feedback results using a simple probabilistic model and a single document collection. She looked at the effect of reweighting terms, various methods for selecting terms to expand a query, and the number of terms added to the query. For a variety of term selection

*This research was supported by a contract with West Publishing Company, and by the NSF Center for Intelligent Information Retrieval at the University of Massachusetts. Thanks also to Michael Gordon and the Cognitive Science and Machine Intelligence Laboratory at the University of Michigan.

[1] The evaluation of relevance feedback techniques is discussed in section 3.3.

methods, she found increases in precision of between 65% to 110%. The method for selecting terms, and the number of terms added, had a noticeable effect on performance. When adding terms, the performance of the most effective selection techniques peaked after adding approximately 20 to 40 terms.

1.2 Hypotheses Tested

In this paper, we extend the previous research on relevance feedback by testing the following six hypotheses:

1. Relevance feedback is effective in the inference network framework. This involves showing that basic techniques can be successfully implemented in a system based on the inference net model.
2. Changing the relative importance of terms is effective for relevance feedback. This hypothesis, which has been thoroughly examined with collections of document abstracts, should be tested in a full text environment.
3. Adding new terms to a query is effective for relevance feedback. This is another "known" result that needs testing in a full text environment.
4. Different methods for reweighting and selecting terms will have different effectiveness. Combinations of different techniques are also studied.
5. The relative contribution of original query terms and added terms affects relevance feedback performance. This was shown by Salton and Buckley using collections of abstracts.
6. Relevance feedback will be effective on queries that contain structured operators such as phrases and proximity.

In the next section, we discuss how relevance feedback can be included in the inference net framework. Section 3 describes the experiments which are carried out using two test collections. Section 4 reports the results, and Section 5 summarizes them.

2 Relevance Feedback and Inference Networks

2.1 The Inference Net Model

Turtle and Croft [18, 19] introduced the inference network model of reasoning under uncertainty [7] to information retrieval. Like simpler probabilistic retrieval models, this is a probability-based method that follows the probability ranking principle [11]. However, rather than ranking documents by their calculated probability that the document is relevant, (given the selected document and query), it ranks them based on the probability that a document satisfies the user's *information need*. This differs from other probability-based methods, in that the information need may be based on complex interactions between various sources of evidence and different representations of the user's need. The user's query is a primary contributor to structuring that representation, but other information, such as thesaurus information, can also be added.

The details of the inference net model have been discussed in previous papers and here we confine ourselves to a brief overview in order to present the approach to relevance feedback.

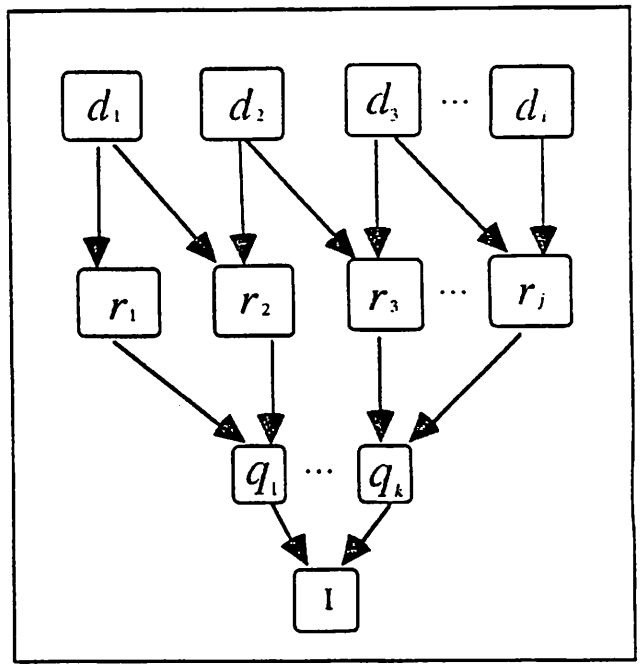

Figure 1: *A generic inference network query.*

The inference network used for the simple model of information retrieval is shown in Figure 1. There are four kinds of nodes in this network. The d_i nodes represent particular documents and correspond to the event of observing that document. The r_j nodes are *concept representation nodes*. These correspond to the concepts that describe the contents of a document. The q_k nodes are *query nodes*. They correspond to the concepts used to represent the *information need* of the user. The single leaf node I corresponds to the (unknown) information need.

The d_i nodes are roots of the network. To evaluate a particular document, the single corresponding d_i node is instantiated and the resulting probabilities are propagated through the network to derive a probability associated with the I node. To generate a ranking for all documents in the collection, this occurs for each of the d_i nodes in the network. Each d_i node is instantiated only once and no other d_i nodes are active at the same time. In other words each document is evaluated seperately, not as part of a set of more than one relevant document.

The probabilities associated with child nodes are based on the probabilities of their parents and on a "link matrix" [19] that specifies how to combine the evidence from the parents. The "link matrices" between the d_i nodes and the r_j nodes represent the evidence for the proposition that this concept occurs in this document. The link matrices between the r_j nodes and the q_k nodes specify how the representation concepts are combined to form the final probability. The d_i and r_j nodes are static for a given collection and are constructed independently of any particular query. The q_k and I portions of the network are constructed individually for each query.

Figure 2 shows a simple network for a query that requests

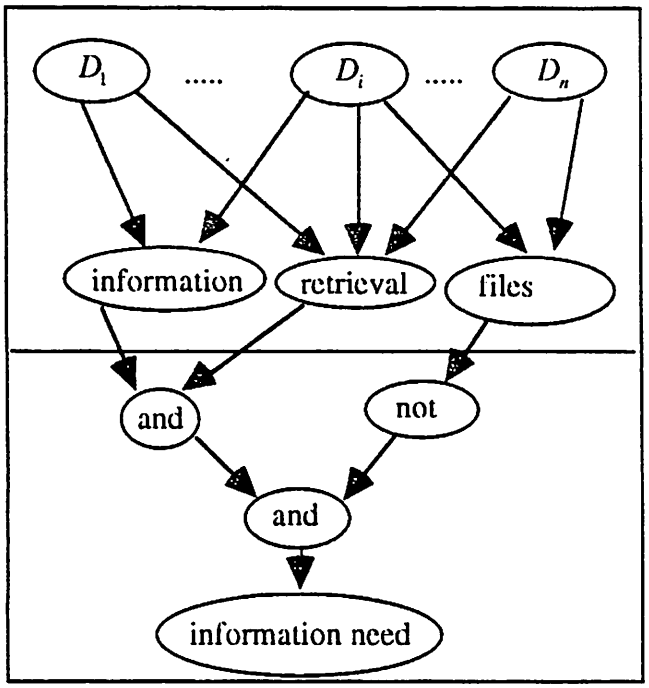

Figure 2: *An inference network for the query* (and (and information retrieval) (not files)).

documents concerning information retrieval but *not* concerning files. The portion above the horizontal dividing line can be constructed before any query is asked. The section below is the expression of the user's information need and has been created specifically for this request.

2.2 Relevance Feedback

In the vector space model, the basic relevance feedback strategy [13] for producing a new query, given an old query and relevance judgements, is as follows:

$$Q_{new} = Q_{old} + \beta \sum_{rel} \frac{D_i}{|D_i|} - \gamma \sum_{nonrel} \frac{D_i}{|D_i|} \qquad (1)$$

where the summation is taken over the known relevant and nonrelevant documents, and the D_i represent document vectors. In the vector space model, then, relevance feedback involves changing weights associated with query terms and adding new terms to the original query.

In the probabilistic model described by Robertson and Sparck-Jones [12] and Van Rijsbergen [21], relevance feedback is described in different terms. In this model, documents are ranked using a (generally) linear discriminant function in which each term corresponds to a representation concept in the collection. Typically, only the representation concepts found in the query have non-zero values and the coefficients of these terms are estimated using some model-specific function. A representative function is

$$g(d) = \sum_i \left(\log \frac{p_i}{1 - p_i} + \log \frac{1 - q_i}{q_i} \right) \qquad (2)$$

where p_i is the probability that term i occurs in a relevant document and q_i is the probability that term i occurs in a non-relevant document. The second term in the summation is typically estimated using each term's *idf*, and the first term is based on the characteristics of the set of known relevant documents. This term is initially estimated using a fixed value (e.g., $p_i = 0.5$) or a value based on the frequency of the term in the query.

In relevance feedback, we are given a sample of documents that have been judged relevant, and we re-estimate our linear discriminant function based on this sample. This involves computing a new set of p_i values for equation 2, based on the relevant sample and adding the top n relevant document terms (according to some measure) to the original query terms. Addition of new terms from the relevant documents was not done in early experiments with the probabilistic model, although it has been investigated subsequently. The probabilistic model does, in fact, indicate that the addition of these terms, up to a point, should improve performance.

Although the inference net model is a probabilistic model, there are differences from earlier models [18]. In particular, because this model does not use Bayesian inversion, there are no probabilities that correspond directly to p_i and q_i. This means that feedback in the inference net model is not the same as in the model described by Robertson and Sparck-Jones. There are two basic ways in which feedback can be incorporated in an inference network model: adding evidence or altering the dependencies represented in the network. The two approaches are fundamentally different. Adding evidence always leaves the probability distribution represented in the network unchanged, but alters beliefs in the network to be consistent with that distribution. Altering the dependencies, either by changing the topology of the network or by altering the link matrices, changes the underlying probability distribution which, in turn, alters belief. The use of evidence is appropriate when we know that the distribution is "correct" (if, for example, the topology is known and the link matrices have been learned from a reliable sample). Evidential feedback is appropriate in the document network which is largely determined by the characteristics of the collection. Frisse and Cousins [3] use this approach to implement feedback in a hierarchy of index terms associated with a hypertext medical handbook.

Altering dependencies is appropriate when the initial network is known to be an approximation to the correct distribution and we obtain better information about the nature of the true distribution. This is the approach we use to change the query network in response to user relevance judgements.

In the network model, queries are represented by links between query nodes and the information need node, and query term weights are represented using the "weighted sum" form of the link matrix at the information need node [18]. This operator takes as input the probabilities from the parent nodes and a vector of weights that describes how much each parent should contribute to the final probability. The computation of this operator is easily interpreted — it computes the belief in this node as the weighted average of the beliefs in the parents. The weights used can be any weights that indicate the relative importance of different parents.

The basic relevance feedback strategy of adding terms to

the query and recalculating weights is implemented in the inference net by adding links between the information need node and the query concepts to be added, and re-estimating the link matrix weights based on the sample of relevant documents rather than on the query text. The fact that we are explicitly modifying the query, means that the inference net model can accurately simulate the vector space approach to feedback. However, since this is a probabilistic model, it should be possible to say what probabilities are being re-estimated during feedback instead of talking about changing weights. In the link matrix for the weighted sum operator, which is used for relevance feedback, the weight associated with a query term is used to estimate the probability that an information need is satisfied given that a document is represented by that term. Simple relevance feedback in the inference net model, then, involves re-estimation of that probability instead of the p_i probability in earlier models.

In summary relevance feedback using the inference network model adds new terms as parents of a query node using a weighted sum link matrix, and re-estimates the relative weights of the parents' contributions to that weighted sum.

2.3 Feedback with Structured Queries

A number of models have been proposed for using relevance feedback with Boolean retrieval systems [14, 17, 10, 9]. While some of these models have been shown to significantly improve performance when compared to conventional Boolean retrieval, they are not attractive in the context of the network model. These models generally adapt probabilistic relevance feedback techniques to estimate weights for terms in very restricted Boolean query forms (e.g., disjunctive normal form with no negation and *and* terms containing at most three representation concepts). Since these models do not make use of any linguistic or domain knowledge, it is unlikely that they will afford performance gains that cannot be achieved with normal probabilistic relevance feedback.

The development of an effective relevance feedback mechanism for Boolean and structured queries [2] is a potentially important area for further research. Encoding feedback information in a structured query could improve performance more than in a simple query since it is possible to encode information in the structured query that can not be represented in a simple vector of terms.

A number of techniques for feedback with structured queries are possible. In this paper, we report preliminary experiments that focus on queries that incorporate phrase structure identified in the queries.

3 Experiments

Experiments were carried out with two different document collections. The CACM collection is a standard collection of 3,204 documents with text from the title and (sometimes) abstract. It contains less than 2 megabytes of text. We used two query sets. The first was a 50 query subset of the standard natural language queries that we have used in many previous experiments. The second set, which was used for testing feedback with structured queries, augmented these 50 queries by adding manually selected phrases [2].

The WEST collection consists of 11,953 full text legal documents. It contains approximately 250 megabytes of text. The documents contain an average of approximately 3,250 words and 530 unique terms. We used a set of 34 natural language queries provided with the WEST collection as the standard query set. The queries contain an average of 9.4 unique terms. Relevance judgements were obtained by expert judgement of the highly-ranked documents. The structured queries for the WEST collection were created by recognizing phrases from a legal dictionary.

The following sections discuss the two groups of relevance feedback experiments. The major group is made up of the the experiments that use the word-based queries. The second group is made up of the preliminary experiments with the structured queries.

3.1 Word-Based Queries

In order to test the hypotheses mentioned in section 1.2. We examined the following independent variables:

1. The collection.
2. Methods for selecting new terms.
3. Methods for reweighting terms.
4. Relative weighting of query terms and new terms.
5. Number of terms added.

The number of variables led to a very large number of experiments. Summaries of the results are presented in this paper.

3.1.1 Term Selection Methods

A variety of methods were used for selecting terms to add to the query. Each of the methods described below is designed to provide a numeric value for each term. The terms were then sorted by this number and the top n of them were added to the query. If tied scores required adding more than n terms to the query, the tied terms were not added.

1. EMIM: This is the *expected mutual information* between term occurrence in a document and the judgment of the document as relevant [6],[21, pg 123].

   The formula for the calculation is

   $$\sum_{\substack{i \in \{t_k, \overline{t_k}\} \\ j \in \{R, \overline{R}\}}} P_{i,j} \log \left(\frac{P_{i,j}}{P_i \cdot P_j} \right).$$

   $P_{i,j}$ is the probability that a judged document has both characteristics i and j. P_l is the probability that a judged document has the value of characteristic l collapsing over the value of the other variable.

   We let t_k stand for the event that term k occurs in a particular document and $\overline{t_k}$ stand for the event that the term does not occur in a particular document. Similarly, let R and $\overline{R}$ stand for the event that a particular document is, or is, not relevant. In this application, the calculation looks at each combination of the term occurring (or not occurring) in a relevant (or non-relevant) document. It computes a measure of the predictiveness of the term occurrence for relevance. This measure is equivalent to the *information gain* measure used in the ID3 learning algorithm [8] to select nodes in a decision tree. However, the classification algorithm that is used is quite different.

2. PMIM: This computes the EMIM score for just the state where the term is present and the document is relevant. Specifically, it computes:

$$P_{t_k,R} \log \left(\frac{P_{t_k,R}}{P_{t_k} \cdot P_R} \right)$$

3. P_4: This is the probability that the document will be correctly classified as relevant (or not) depending on the occurrence (or absence) of the term t_k. it is computed as:

$$\left(P_{t_k,R}\right) \left(P_{\overline{t_k},\overline{R}}\right) \left(1 - P_{\overline{t_k},R}\right) \left(1 - P_{t_k,\overline{R}}\right)$$

4. idf: Idf is defined as $\log \left(\frac{\text{collection size}}{\text{\# documents containing } t_k} \right)$. If we define P_{t_k} to be the probability that a randomly selected document contains the term t_k, then idf is also $-\log P_{t_k}$. Taking the antilog of the log factor gives us a function monotonic with idf: $(-1)(P_{t_k})$. The -1 factor serves to invert the role of minimization and maximization. Selecting the terms with high values from this measure will select terms will low probabilities of appearing in a randomly-selected document. Idf selection favors terms that are unlikely to appear in a document by chance.

5. rdfidf: This is product of a term's idf and the number of relevant and judged documents in which it appears. Since idf is monotonic with $(-1)(P_{t_k})$ multiplying this by df and taking the antilog of $df \cdot \log P_{t_k}$ gives us $(-1)\left(P_{t_k}{}^{df}\right)$. Again the -1 serves to interchange the the roles of maximization and minimization.

   Since P_{t_k} is the probability that one could select a single document and have the term t_k appear in it $P_{t_k}{}^{df}$ is the probability that one could select df documents from the collection randomly and have this term appear in all of them. In other words, selecting the terms with the highest rdfidf score is the same as selecting the terms with the lowest probability of appearing by chance in the set of relevant documents.

6. rtf: This is simply the frequency of the term in the documents judged relevant by the user.

7. rtfidf: This is the frequency of the term in the documents judged relevant by the user multiplied by the idf of the term. The idf factor provides a correction for the possibility that a term will occur in a document by chance.

3.1.2 Term Weighting Methods

The last two selection methods also make plausible term weighting methods and are the two methods explored. They are plausible as we expect terms that are related to relevance to occur often in relevant documents.

3.1.3 Relative Weighting of Query Terms and Added Terms

It is possible that terms from different sources should not be given the same weight. Salton and Buckley [15] found that during relevance feedback a 75%-25% weighting split between terms from relevant documents and terms from non-relevant documents was better than equal weighting. This does not precisely correspond with our experimental setup as we do not consider non-relevant documents. However, it suggests that relative weighting may be an important variable. Therefore, we examined a number of different weighting distributions from equal weighting of original query terms and added terms to weighting that was 90% original terms and 10% added terms. In vector space retrieval this variation would be accomplished by setting γ to 0 and varying β in equation 1.

3.1.4 Number of Terms Added

If term selection is important, then the number of terms added to a query must also be important. Harman [5] found that adding between 20 and 40 terms led to peak performance. Adding either fewer or more terms decreased performance. We therefore examined adding various numbers of terms to the query. For CACM, we examined adding between 0 and 150 new terms. Because of the greater cost of running WEST experiments, we considered only adding 0 to 100 terms.

3.2 Structured Queries

Additional experiments were run to determine the effectiveness of relevance feedback for structured queries. For both the WEST and CACM collections, the following three experiments were tried:

1. Relevance feedback on queries containing *phrase* operators [2].
2. Relevance feedback with all *phrase* operators replaced by proximity operators. With proximity operators, the belief in phrase concepts is based entirely on the presence of the words in proximity to each other.
3. Relevance feedback where the modified query constructed by relevance feedback deleted all the structured operators in the original query but kept the original terms.

In these experiments, no new structured terms were added by relevance feedback — the structured terms served to increase the baseline performance of the queries. These terms were reweighted through relevance feedback to see if reweighting of structured terms was effective. For all of these experiments additional *single* terms were added to the query as in the single word experiments.

These were preliminary experiments designed to assess the utility of current relevance feedback methods on structured queries. They were run using a set of parameters found to be effective during the prior experiments.

3.3 Evaluation of Relevance Feedback Experiments

The evaluation of relevance feedback performance must be somewhat different than that used for a typical information retrieval experiment. Normally documents are ranked and presented to the user only once. With relevance feedback, the user has seen and judged some of the documents before the relevance feedback query is evaluated. When assessing the effectiveness of relevance feedback, the ranks of these judged documents are not relevant and may even be misleading. They must be factored out of the evaluation of the effectiveness of relevance feedback. We do this by using the *residual collection* method [1]. Judged documents

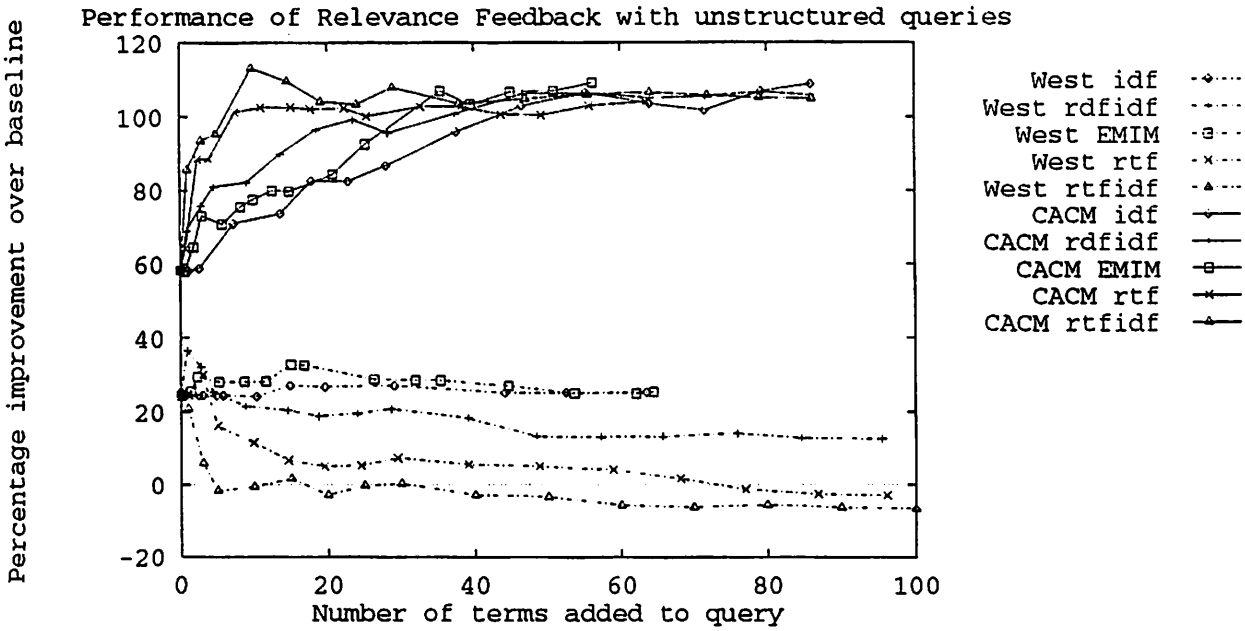

Figure 3: *Typical performance of relevance feedback for the WEST and CACM collections using rtfidf term reweighting and relative weighting factors of 65% for original query terms and 35% for new terms. Each line represents a particular collection, term selection combination.*

are removed from the document rankings produced by both the original query and the relevance feedback query. Recall-precision values are then used to summarize performance. These recall-precision values measure performance for only those documents the user hasn't seen. The percentage increase (or decrease) can then be used to determine if the user would have been better off continuing to use the ranking given by the original query or the one produced by the relevance feedback query. The disadvantage of this approach is that the methods it compares must be based on identical baselines.

While the residual collection method is commonly used to report relevance feedback results, any direct comparison with other published results must be made carefully, as there are inevitable differences in collections and in retrieval and indexing algorithms. Performance for each experiment is measured by the average precision at 3 recall points: 0.25, 0.50, 0.75. Effectiveness of relevance feedback was measured by the percentage change from the baseline condition of continuing to use the original query. Any query which had no remaining unjudged relevant documents after the initial user judgement was dropped from the analysis. User judgements were simulated by assuming the documents that appeared in both the supplied list of relevant documents and the top ranked 15 documents for this query were judged relevant. Other documents appearing in the top 15 documents were assumed to be non-relevant.

When comparing results across collections, these percentage increase figures were inappropriate as there are differences in the basic level of performance for the two collections. To compensate, the *percentage increase* values reported for individual collections were normalized to be a *percentage of the collection maximum increase* when reporting combined data. The experiments can then be compared across collections on the basis of their performance relative to the best method for their own collection.

4 Results

4.1 Results for Simple Queries

Effect of number of terms added

Typical results for reweighting terms and adding new terms to a query are seen in Figure 3. This clearly shows the effectiveness of both reweighting the original query terms (adding 0 terms) and of adding additional terms. For CACM, the effect is very clear. Adding terms increases performance for all selection methods. For WEST, the increase depends on the selection method. However, the best methods are further improved by adding terms. The hypotheses that the relevance feedback is effective, and that reweighting and term addition are both effective, are supported.

Effects of other variables

Because of the number of experiments, examining each experiment separately is infeasible. We summarize the results of adding terms by characterizing the performance for each condition by the maximum increase for any number of terms added. Tables 1 and 2 show the maximum percent increase over the baseline results for each collection individually. Table 3 presents the percentage of maximum results collapsed

across collections for the rtf and rtfidf weighting schemes. The main conclusions are:

1. Term reweighting is effective for both abstract and full text collections, although the percentage increases are much smaller with the full text WEST collection.
2. Adding additional terms is generally also effective. For the CACM collection, the number of terms added does not seem to be critical. In the WEST collection, adding too many additional terms can actually decrease performance.
3. The effect of the term weighting method depends on the collection used. However, rtfidf weighting produces good results on both collections.
4. Performance is somewhat improved by weighting the added terms less than the query terms.

With regard to the hypotheses put forward in section 1.2, we see that the first three are supported. Relevance feedback with the inference net is effective. It produces performance increases under many combinations of variables, and these increases can be as large as 118% on the CACM collection. Both the term reweighting and the term addition methods contribute to this increase. Given that the CACM baseline is higher than that used in the Salton and Buckley experiments, this is very good performance.

As suggested by the fourth hypothesis, the choice of selection method has an effect. However, the overall pattern of differences is not consistent for the two collections and no simple interpretation can be made.

The status of the fifth hypothesis is less than clear. The extreme relative weighting of 90% for query terms and 10% for added terms was poor, but the rest of the conditions were fairly similar in performance. The effect of this variable varies by collection. Relative weighting has a larger effect on the CACM collection than on the WEST collection.

The other overall observation we make is that while relevance feedback is effective, it appears to be much less effective on the full text collection than it is with the collection of abstracts.

Based on the results in Table 3, a good overall feedback method is rdfidf term selection with rtfidf term weighting and a 65%/35% split between the query term and added term weights. This combination provides 90.4% of the peak performance for each collection, with little variation between the collections. This is not the maximum increase over all conditions, but it provides nearly the performance of the peak combination (91.3% for the PMIM term selection method with rtfidf term weighting and 75%/25% relative contributions). In addition, the rdfidf selection method does not require user judgements of non-relevance. The user is only required to accurately assess relevant documents rather than judge all the documents in the top n documents presented.

4.2 Structured Queries

The results for structured queries are presented graphically in Figure 4. It is important to note that the structured queries for each collection have two different starting baselines. The resulting percentage increase figures must therefore be compared with caution. Of the three kinds of experiments run for each collection, two of the experiments start with same initial queries that contain *phrase* operators. They therefore have the same baseline. For the other experiment, the phrase operators have been changed to proximity operators in the initial queries. However, the differences in the baselines between the phrase-based and proximity-based queries are not large. For the CACM collection, the baseline retrieval precision is 35.3 for the proximity-based queries and 38.5 for the phrase-based queries. The baseline for phrase-based queries show an improvement of 9.1%. For the WEST collection, the corresponding precision values are 55.9 and 56.9. Phrase queries are 1.7% better for WEST.

For CACM, without adding terms, we see an increase of about 5% over the baseline except when using proximity operators. There we see an increase of 80%. After adding additional terms, the proximity condition eventually rises up to a 20% gain. The other conditions rise to about a 50% gain. For the phrase-based queries there is little difference in performance between the queries that continue to contain phrases and those where the phrase operator is deleted.

For WEST, the increases are smaller. Without adding terms, the increases in performance are small (at most 5.5%). When adding additional terms phrase-based queries produce a 11% increase in performance. When deleting the initial structure the performance of simple reweighting of terms is little changed over the baseline. The queries containing proximity operators have increases of approximately 20%.

If we take a 10% improvement in performance as a criterion, then in five of the six conditions relevance feedback does produce improvements. The improvements are dramatic for the CACM collection. The improvements are much smaller for the large, full text, WEST collection.

Discussion of Structured Queries

Adding terms does increase the performance of structured queries. Simply reweighting the terms in the structured query usually does not seem to increase performance a great deal. One possible explanation for this decreased performance, relative to the word-based queries, is that using appropriate phrases increases the baseline performance a great deal and this may may it difficult to further increase performance. However, the continuing increase in performance, when adding terms, makes this unlikely. A second explanation may be that the appropriate methods for selecting terms or reweighting parts of structured terms are not the same as those for single terms. The current implementation of the *phrase* operator measures belief in a phrase concept by the presence of single words as well as words in proximity. Therefore, our current method of simply counting words in proximity to calculate a new weight for these phrases is probably not appropriate. For the proximity-based queries, the counting-based reweighting methods are more appropriate, and are more successful.

Support for the sixth hypothesis is in section 1.2 is clear. Relevance feedback can effectively be done on structured queries. However, the process is less effective than for word-based queries.

5 General Discussion

As discussed in section 4.1 and section 4.2, four of the six hypotheses given at the start of the paper are supported. We

selection method	relative query term and added term weighting								
	equal	60-40	65-35	70-30	75-25	80-20	85-15	90-10	average
	rtf weighting								
rdfidf	82.9	88.5	90.8	86.0	83.1	82.3	79.4	70.9	*83.0*
EMIM	84.9	91.8	90.5	91.5	88.1	82.3	77.9	68.8	*84.5*
idf	94.4	95.0	98.7	94.1	88.5	81.8	75.9	67.6	*87.0*
PMIM	84.6	89.2	93.8	96.0	90.0	83.8	78.4	67.8	*85.5*
P_4	78.1	86.3	84.1	84.6	83.7	81.4	76.5	71.4	*80.8*
rtf	76.3	82.0	83.3	83.1	85.0	79.9	75.6	67.6	*79.1*
rtfidf	84.1	91.2	90.0	86.2	84.7	82.0	77.3	69.8	*83.2*
average	*83.6*	*89.1*	*90.2*	*88.8*	*86.2*	*81.9*	*77.3*	*69.1*	*83.3*
	rtfidf weighting								
rdfidf	97.3	102.9	106.7	109.1	110.1	108.0	108.8	92.5	*104.4*
EMIM	106.7	110.7	109.1	113.3	116.6	112.5	109.2	91.5	*108.7*
idf	102.0	107.9	110.0	110.3	110.6	106.9	105.6	90.6	*105.5*
PMIM	100.3	106.7	109.6	110.5	117.9	118.2	108.9	91.3	*107.9*
P_4	96.7	99.3	105.7	107.8	109.0	106.8	107.5	90.3	*102.9*
rtf	94.9	101.0	104.5	106.1	107.7	106.9	107.7	90.6	*102.4*
rtfidf	100.8	106.2	113.0	109.0	109.1	108.3	107.6	92.2	*105.8*
average	*99.8*	*105.0*	*108.4*	*109.4*	*111.6*	*109.7*	*107.9*	*91.3*	*105.4*

Table 1: *Average increase in performance for the relevance feedback modified query over the continuation of the original query for the CACM collection using two weighting methods and seven term selection methods.*

selection method	relative query term and added term weighting								
	equal	60-40	65-35	70-30	75-25	80-20	85-15	90-10	average
	rtf weighting								
rdfidf	40.1	38.2	37.1	37.0	36.6	36.1	36.2	34.9	*37.0*
EMIM	39.8	37.4	36.8	38.2	40.4	40.0	36.4	34.7	*38.0*
idf	32.4	32.2	32.2	31.2	31.1	31.1	31.1	27.1	*31.1*
PMIM	39.9	38.7	38.2	38.0	38.2	39.8	36.0	35.5	*38.0*
P_4	40.2	38.6	37.8	39.1	40.0	39.1	36.0	35.6	*38.3*
rtf	29.4	30.4	31.7	30.6	30.5	30.5	30.1	30.9	*30.5*
rtfidf	27.0	27.0	27.0	27.0	27.0	27.5	29.4	31.8	*28.0*
average	*35.5*	*34.6*	*34.4*	*34.4*	*34.8*	*34.9*	*33.6*	*32.9*	*34.4*
	rtfidf weighting								
rdfidf	36.2	36.6	36.6	35.2	35.7	34.4	35.1	36.2	*35.8*
EMIM	31.1	32.4	32.8	32.7	33.1	32.9	32.4	31.6	*32.4*
idf	27.2	27.6	27.1	26.4	25.2	25.1	24.8	24.8	*26.0*
PMIM	29.6	30.8	31.7	32.4	33.5	32.9	32.9	33.3	*32.1*
P_4	33.0	34.0	32.9	33.6	33.8	33.6	35.3	32.4	*33.6*
rtf	32.1	31.4	30.0	30.5	28.8	27.3	26.6	27.6	*29.3*
rtfidf	24.5	24.5	24.5	24.5	24.5	24.5	24.5	27.4	*24.9*
average	*30.5*	*31.0*	*30.8*	*30.8*	*30.7*	*30.1*	*30.2*	*30.5*	*30.6*

Table 2: *Average increase in performance for the relevance feedback modified query over the continuation of the original query for the WEST collection using two weighting methods and seven term selection methods.*

selection method	relative query term and added term weighting								
	equal	60-40	65-35	70-30	75-25	80-20	85-15	90-10	average
	rtf weighting								
rdfidf	84.7	84.7	84.3	82.2	80.4	79.5	78.4	73.2	*80.9*
EMIM	85.2	85.1	83.8	86.0	87.3	84.3	78.0	72.0	*82.7*
idf	80.0	80.0	81.6	78.4	75.9	73.1	70.6	62.1	*75.2*
PMIM	85.2	85.6	87.0	87.6	85.3	84.7	77.7	72.6	*83.2*
P_4	82.8	84.3	82.4	84.2	84.9	82.8	76.9	74.3	*81.6*
rtf	68.7	72.3	74.5	73.0	73.7	71.5	69.2	66.8	*71.2*
rtfidf	69.0	72.0	71.5	69.9	69.2	68.7	69.1	68.9	*69.8*
average	*79.4*	*80.6*	*80.7*	*80.2*	*79.6*	*77.8*	*74.3*	*70.0*	*77.8*
	rtfidf weighting								
rdfidf	86.0	88.8	90.4	89.7	90.8	88.3	89.5	83.9	*88.4*
EMIM	83.6	86.9	86.7	88.4	90.3	88.3	86.3	77.8	*86.0*
idf	76.8	79.8	80.1	79.3	78.0	76.3	75.4	69.0	*76.8*
PMIM	79.1	83.3	85.6	86.8	91.3	90.7	86.8	79.8	*85.4*
P_4	81.7	84.1	85.4	87.2	87.9	86.8	89.2	78.3	*85.1*
rtf	79.9	81.6	81.3	82.6	81.2	79.0	78.5	72.5	*79.6*
rtfidf	73.0	75.2	78.1	76.4	76.5	76.1	75.8	72.9	*75.5*
average	*80.0*	*82.8*	*84.0*	*84.4*	*85.1*	*83.6*	*83.1*	*76.3*	*82.4*

Table 3: *Average performance of relevance feedback across collections using rtf and rtfidf term reweighting. To make results for different collections comparable the results for each collection were transformed from percentage improvement over the baseline to percentage of the maximum performance obtained over each collection for any experiment.*

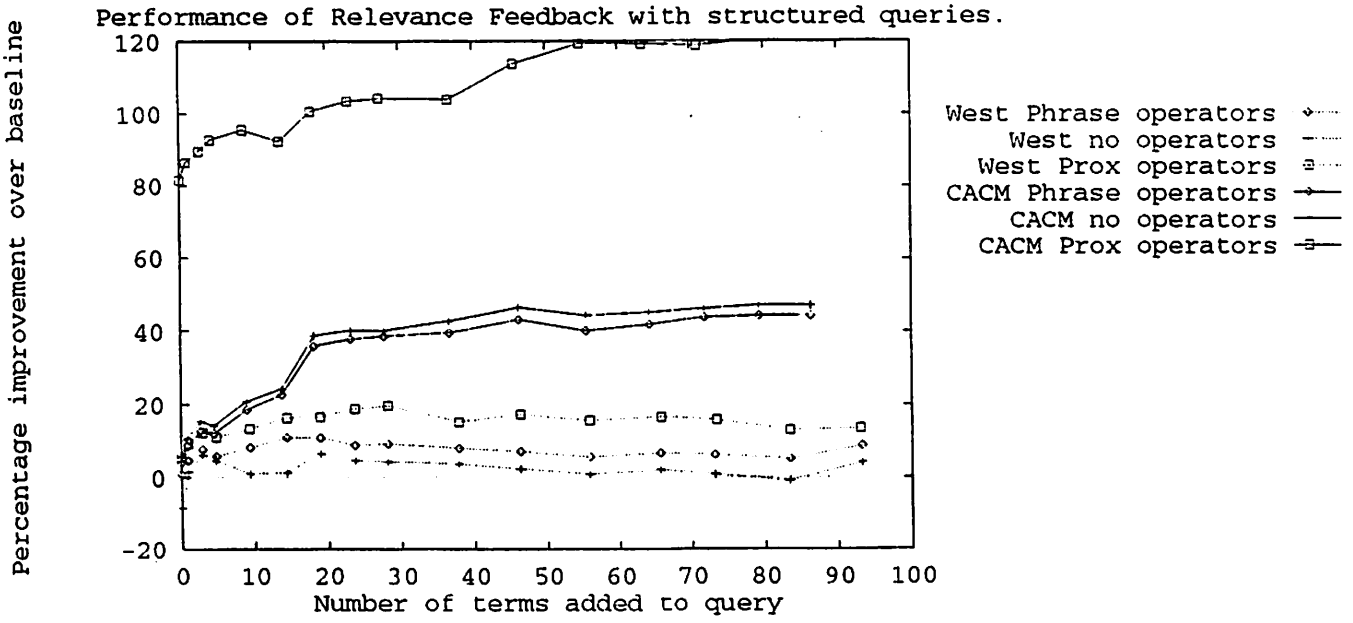

Figure 4: *Performance of relevance feedback on structured queries for the WEST and CACM collections using rdfidf term selection and rtfidf term reweighting.*

find that the general hypothesis, that relevance feedback in inference networks is effective, is strongly supported. Under many different combinations, we get significant increases in performance and, for the CACM collection, these performance increases seem to be comparable to those observed using the vector space model. The increases are robust over many combinations of conditions. For the CACM collection, the average peak increase over all the conditions was approximately 90%. For the WEST collection, the average peak increase was over 30%.

For structured queries, the results are less compelling. While still effective, the increases in performance are generally much smaller. As mentioned previously, the techniques used for single term queries may not be the best methods for structured queries and more experiments are needed to explore this further.

One striking pattern in the data is that there are large differences in performance between the two collections used here. It is far easier to get large performance increases with the CACM collection than with the WEST collection. An interesting question is whether this may be due to the superiority of relevance feedback methods when there is less text or whether initial performance in a full text collection might be superior to that for a selected text collection. That would leave less room for relevance feedback to improve performance. Our current data does not address this question, but it is an extremely important issue that needs to be understood. It may be that new techniques must be developed for effective feedback performance on full text collections.

For neither the unstructured or structured queries do we believe that we have found optimal selection and weighting methods. The ones chosen were based on previous research. Further research may well find others that would be better.

References

[1] Y. K. Chang, C. Cirillo, and J. Razon. *Evaluation of Feedback Retrieval using Modified Freezing, Residual Collection, and Test and Control Groups*, chapter 17, pages 355–370. Prentice-Hall Inc., 1971. in The SMART Retrieval System: Experiments in Automatic Document Processing.

[2] W. Bruce Croft, Howard R. Turtle, and David D. Lewis. The use of phrases and structured queries in information retrieval. In *Proceedings of the 14th Annual International Conference on Research and Development in Information Retrieval*, pages 32–45, Chicago, 1991. SIGIR.

[3] Mark E. Frisse and S. B. Cousins. Information retrieval from hypertext: Update on the dynamic medical handbook project. In *Proceedings of Hypertext 89*, pages 199–212. ACM Press, 1989.

[4] Donna Harman. Relevance feedback and other query modification techniques. In W. B. Frakes and R. S. Baeza-Yates, editors, *Information Retrieval: Data Structures and Algorithms*, chapter 11, pages 241–263. Prentice Hall, 1992.

[5] Donna Harman. Relevance feedback revisited. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Developement in Information Retrieval*, pages 1–10. SIGIR, 1992.

[6] D. J. Harper. *Relevance Feedback in Document Retrieval Systems: An Evaluation of Probablistic Strategies*. PhD thesis, Cambridge University, 1980.

[7] Judea Pearl. *Probablistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc, 1988.

[8] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[9] Tadeusz Radecki. Incorporation of relevance feedback into boolean retrieval systems. In *Proceedings of the 6th Conference on Research and Development in Information Retrieval*, pages 133–150. SIGIR, 1983.

[10] Tadeusz Radecki. Probabilistic methods for ranking output documents in conventional boolean retrieval systems. *Information Processing and Management*, 24(3):281–302, 1988.

[11] S. E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, 1977.

[12] S. E. Robertson and Karen Sparck-Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.

[13] J.J. Rocchio. *Relevance Feedback in Information Retrieval*, chapter 14, pages 313–323. Prentice-Hall Inc., 1971. in The SMART Retrieval System: Experiments in Automatic Document Processing.

[14] Gerald Salton. *Automatic Text Processing: The Transformation, analysis and Retrieval of Information by Computer*. Addision-Wesley, Reading MA, 1989.

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

[16] Gerald Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[17] Gerald Salton, E. Voorhees, and Edward A. Fox. A comparison of two methods for boolean query relevance feedback. *Information Processing and Management*, 20(5/6):637–651, 1984.

[18] Howard R. Turtle. *Inference Networks for Document Retrieval*. PhD thesis, University of Massachusetts, October 1990.

[19] Howard R. Turtle and W. Bruce Croft. Inference networks for document retrieval. In *Proceedings of the 13th International Conference on Research and Development in Information Retrieval*, pages 1–24, Brussels, Belgium, September 1990. SIGIR.

[20] Howard R. Turtle and W. Bruce Croft. A comparison of text retrieval models. *Computer Journal*, 35(3):279–290, June 1992.

[21] C. J. van Rijsbergen. *Information Retrieval, Second Edition*. Butterworths, 1979.