INCORPORATING DIFFERENT SEARCH MODELS INTO ONE DOCUMENT RETRIEVAL SYSTEM

W. Bruce Croft
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

Abstract

Many effective search strategies derived from different models are available for document retrieval systems. However, it does not appear that there is a single most effective strategy. Instead, different strategies perform optimally under different conditions. This paper outlines the design of an adaptive document retrieval system that chooses the best search strategy for a particular situation and user. In order to be able to support a variety of search strategies, a general network representation of the documents and terms in the database is proposed. This network representation leads to efficient methods of generating and using document and term classifications.

One of the most desirable features of an adaptive system would be the ability to learn from experience. A method of incorporating this learning ability into the system is described. The adaptive control strategy for choosing search strategies enables the system to base its actions on a number of factors, including a model of the current user.

Finally, some ideas for a flexible interface for casual users are suggested. Part of this interface is the heuristic search, which is used when searches based on formal models have failed. The heuristic search provides a browsing capability for the user.

1. Introduction

The theory and design of document retrieval systems have undergone significant changes in the past few years. A number of techniques have been developed for searching text documents which, based on theoretical and empirical evidence, can be expected to improve system effectiveness relative to simple heuristic techniques. These techniques or search strategies, which are predominantly based on probabilistic models of document retrieval, include relevance weighting [1,2,3,4], relevance weighting modified to use term dependencies [5,6], relevance weighting modified to use within-document frequencies [7,8] and document cluster searching [9,10]. The availability of these techniques does not, however, solve some fundamental questions regarding the design of document retrieval systems. These questions are

Which search strategy, if any, is the best in all cirumstances?

If there is no single best strategy, under what circumstances do the various search strategies achieve their best performance?

What happens if no relevant documents are found by the search strategy used by the system?

How can systems by made more responsive to individual users?

Experimental results indicate that no search strategy is superior to the others in all situations. Even when one strategy produces better average results than another, it is still possible that the latter strategy is preferable in some cases. For example, Croft and Harper [3] describe an experiment which shows that a simple cluster search retrieved relevant documents for many of the queries where relevance weighting had failed, even though over the entire sample of queries the cluster search had a much lower recall*. It would appear, then, that in order to achieve the best performance in a variety of situations involving different users, different queries and different documents, the system should be able to choose from a variety of strategies.

*Recall and Precision are used to measure system effectiveness. Recall is the proportion of the relevant documents for a given query that are retrieved. Precision is the proportion of retrieved documents that are relevant.

The answer to the second question raised above is that it would be extremely difficult to parameterize the situations under which various strategies achieve their best performance. To identify the importance of each of the factors involved would require an extremely large number of experiments. It is more sensible to allow the system to try a promising strategy and, if this fails to retrieve relevant documents, to try another strategy. In other words, the system will find the strategy appropriate to a given situation. This use of alternate strategies also provides an answer to the third question.

In this paper, a system which can modify its actions based on factors such as the history of the search and the needs of particular users is called an adaptive system. The concept of a system modifying its actions based on a model of the current user is very important in artificial intelligence research [11,12] and should lead to more effective document retrieval systems. The process of relevance feedback [13], in which the user identifies some relevant documents and the system modifies the original query based on this additional information in order to retrieve other relevant documents, is a simple type of adaptive system. Using the relevance judgements, the system modifies its model of a relevant document (the query). The type of adaptive system discussed here goes further in that it is able to modify the search strategy used as well as (or instead of) modifying the original query.

Three main issues arise when the implementation of an adaptive system is considered. The first is the representation of the documents in the system's database. Current systems use representations in the form of file organizations designed to operate efficiently for a particular search strategy. For example, systems that use Boolean queries store information in inverted files and systems which use document clusters usually have some form of tree structure representing a hierarchy of clusters [13]. A system capable of using many types of search strategy requires a more general form of representation.

The issue of central importance to an adaptive system is the control strategy. The control strategy is used to select the search strategy for the system. A control strategy could be designed on a heuristic basis, but ideally it should be able to learn which search strategy is best for a given situation. Rather than designing a large series of experiments to identify the important parameters for the search, this knowledge could then be gained by experience with real users.

The third issue, which is related to the last question about document retrieval system design, is the user interface. A friendly user interface is important for any information system but it is particularly crucial for an adaptive system. An adaptive system depends on the user's reactions to determine its next choice of search strategy. It must, therefore, be able to ascertain such things as the user's satisfaction with the current retrieved documents and the type of documents needed without too much effort on the part of the user. The interface should also provide the capability of browsing through the database when search strategies based on formal models are not successful.

In summary then, the research described in this paper is directed towards the development of an adaptive document retrieval system that chooses search strategies using a control strategy guided by information from the user. The adaptive system, therefore, should be responsive to individual users and, because other search strategies are available if the original strategy fails to retrieve relevant documents, it should be more effective than a conventional system. It also provides a framework in which new search strategies can be incorporated without redesigning the whole system. This is possible because the data representation and user interface are flexible enough for a new strategy to be added simply as another alternative in a "pool" of strategies.

The following section discusses the research issues involved with the development of the adaptive document retrieval system. These issues are discussed under the headings of representation, control strategy and user interface. The search strategies mentioned earlier are explained in more detail.

2. Research Issues

The major issues involved with the design of the adaptive document retrieval system are representation, control strategy and user interface. Before they are discussed in more detail, the search strategies that will be used by the system will be described. It is assumed that the documents in the system have been indexed by some procedure. This means that they have been assigned a set of index terms, possibly with weights indicating relative importance, which describe their contents. Indexing can be done manually, but the emphasis in most experimental systems is on automatic indexing for reasons of cost and availability. Sparck Jones reviews the techniques used in automatic indexing [14].

The main search strategy to be used is relevance weighting [1]. With this strategy, documents are ranked according to their probability of relevance to the query. This probability is estimated for each document by first calculating a weight (the relevance weight) for each query term based on its frequencey of occurrence in the relevant and non-relevant sets of documents. These weights are summed for each query term present in the documents and then the documents are ranked according to the total scores. In mathematical terms, the relevance weight r_i of the ith term for a given query is

$$\log p_i(1-q_i)/(1-p_i)q_i \qquad (1)$$

where p_i is the probability that term i occurs in the relevant set and q_i is the probability that term i occurs in the non-relevant set. The total score for a document j is

$$\sum_i r_i x_{ij} \qquad (2)$$

where x_{ij} is the value of the ith term in the jth document and it is assumed to be binary (i.e. it either is assigned (1) or not assigned (0) to the document). The summation is over the terms in the query. Croft [15] discusses the implementation of relevance weighting using inverted files.

This technique is based on the assumption that terms occur independently of one another. This assumption can be overcome to some extent by a modification of relevance weighting that uses terms related to the query terms to expand the query [6]. In order to identify related terms, some type of term classification is needed. This classification groups terms based on statistical co-occurrence in the documents and the assumption is that these groups contain semantically related terms. The generation of the classification can be expensive and other, cheaper, methods of query expansion such as adding terms from identified relevant documents seem to be as effective [4]. However, as is the case with most of the strategies, term classifications do appear to improve performance in some cases where other strategies do not work.

Another modification to relevance weighting is possible if the indexing process includes frequency weights for the terms in the documents. These weights (within-document frequency weights) indicate the relative importance of the terms to particular documents and there is evidence that they can be used to improve performance in some cases [7].

An important distinction for relevance weighting is searching before and after relevance feedback. Before relevance feedback, the only information about the relevant set of documents comes from the query and some assumptions must be made when using formulas (1) and (2). Croft and Harper [3] discuss these assumptions and Harper [16] shows that if the user can provide some information about the relative importance of the query terms, this can lead to a large improvement in performance. Therefore relevance weighting used before feedback will require an even higher degree of user-system interaction than in the feedback process itself.

The next strategy that will be included is document cluster searching. Document clusters, like term classifications, are generated by using a similarity measure to group documents with similar contents. Although the clustering process can be expensive, there is considerable evidence that cluster searching can achieve good performance in many cases where relevance weighting may fail [3,10]. It should, therefore, be an important alternative strategy. Clusters are searched by comparing the query to representatives of the clusters. The clusters whose representatives are most similar to the query are retrieved.

Another strategy that should be included due to its popularity in current systems in the Boolean search. The user specifies the query as a Boolean combination of terms and only the documents that satisfy this combination are retrieved. This strategy is useful for restricting document sets to those having specific characteristics.

The final strategy is one that is particularly appropriate for an adaptive document retrieval system. This is the heuristic search strategy, which is simply a collection of algorithms to be used when the search strategies based on formal models have failed. The heuristic search will generally require a high degree of user-system interaction. It will be discussed more in the next section.

To summarize, the search strategies that are considered as alternatives for the adaptive document retrieval system are

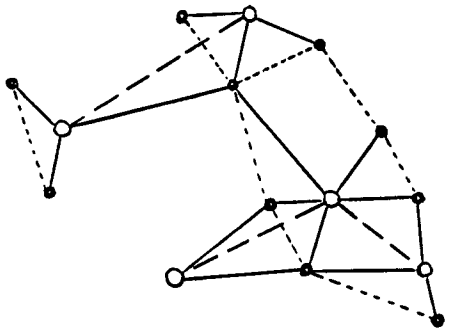
- Relevance Weighting
  - -prior to feedback
  - -after feedback
  - -using related terms to expand query
  - -using term frequency weights
- Cluster Search
- Boolean Search
- Heuristic Search

Although these are not the only strategies that have been suggested in the literature, they do represent the major types of search and each one has been shown to be effective under different conditions. The system is designed to be flexible enough to make the addition of other search strategies a simple process.

2.1 Representation

The representation of the documents and terms in the system must be general enough to be able to implement any of the searches outlined above. The representation proposed is a network of documents and terms (Figure 1). The nodes in the network represent terms or documents and the links represent the connections between them. There is a weight associated with each link that indicates the strength of the relationship. A weight on a document-document link indicates how similar the documents are, a weight on a term-term link indicates how closely related the terms are and a weight on a document-term link indicates how important that term is in describing the document.

FIGURE 1: A NETWORK REPRESENTATION



0 document nodes ● term nodes

____ document-term links -----term-term links

—— —— document-document links

Network representations have been proposed before [13(pg.228),17], but this is the first representation that would include weights on all types of links. The network is obviously a powerful and flexible representation because it incorporates both an inverted and a serial file of the documents. This means that any strategy that can be implemented with an inverted file (relevance weights, Boolean searching) or that uses a serial file (cluster searching) can be implemented using this network. There are, however, a number of problems concerning the use of network. The first consists of implementation issues such as how the network is to be stored in the computer, which weights are to be stored and how the network will be used by the different strategies. The data structure representing the network will contain some extra pieces of information such as the weights on the links. These weights should be very similar to the raw data for maximum flexibility. This means, for example, that the weight on a term-document link should be the actual frequency count rather than some probability estimated from that count. Storing the frequency count adds flexibility because the probability can be calculated in several different ways. The same reasoning applies to the similarity weights on document-document and term-term links. Other data, such as the number of terms describing a document and the number of documents indexed by a particular term, will be stored in the data structure because they are accessed frequently.

Even if a very efficient storage structure for the network is available, it is obviously impractical to generate all possible term-term and document-document links in the network because generating pairwise similarities is an order n^2 problem for n objects and, more importantly, many of the document pairs and term pairs are not significant. In other words, many documents (terms) are only weakly related to each other. The standard way of representing the relationships is by using the term-term and document-document links to define classifications or clusters. In previous experiments, the term classification has been represented by a maximum spanning tree [5]. The spanning tree is a network of terms where each term is connected to at least one other term, no closed loops occur, the whole tree is connected and, if the link weights represent similarity values between the connected terms, the sum of the link weights in the tree is the maximum possible. The maximum spanning tree can obviously be represented in our network but the generation of the tree is still an order n^2 process. Similarly, the most effective clustering procedure for documents generates a single-link hierarchy, which can also be represented as a maximum spanning tree [18]. Although the generation of the tree can be speeded up to some degree by using the inverted file to discover documents (terms) with nothing in common [16,19], the maintenance of these classifications as new documents are added to the system is a serious problem.

A partial solution to this link generation and maintenance problem lies in some recent experimental results by Croft [10], Harper[16] and Sparck Jones[4]. These results show that, when using both term and document classifications, it is only the strongest similarities that are useful for retrieval. This means that instead of generating maximum spanning trees for terms and documents only the strongest similarities need be recorded. Finding "nearest neighbors" is a well-studied problem and it can generally be done much faster than generating a maximum spanning tree. An efficient algorithm using the inverted file in the network to identify documents with terms in common with each other can be developed. If new documents are added to the network under this scheme, all that needs to be done is to add a few links rather than changing the entire spanning tree (which can be a major effort).

The simplification of document-document links also simplifies the algorithm for doing cluster searching in the network. Cluster searching requires queries to be compared to cluster representatives. These representatives are not and, indeed, for maximum flexibility and storage savings, should not be stored in the network as separate entities. Rather, the following process will take place. The documents that contain query terms are located using the network. The document-document links from these documents are followed to construct the cluster representatives. These cluster representatives can then be compared to the query. This process is efficient because the clusters formed in this way are very small (approximately 2-4 documents). Croft has shown that these small clusters are the most effective for retrieval [10].

In the context of a network representation, we can now consider in more detail the question of the heuristic search. This search will be somewhat similar in character to the searches described by Oddy [17]. It will be invoked when all other search strategies have failed. The heuristic search will make use of the links in the network directly to allow the user to browse through the database. For example, the term-term links can be used to show the user terms that are related to the ones originally specified. The user may then find more appropriate terms. Similarly, documents connected to terms the user is interested in can be displayed or, if any relevant documents have been found, documents connected to the relevant documents can be displayed. Once relevant documents or more interesting terms have been found, the system should revert back to the formal strategies since these should be, in general more effective.

2.2 Control Strategy

The control strategy, as mentioned, chooses the appropriate search strategy for a given situation. The ultimate goal of this research is a control strategy that learns which strategies are the most effective. A possible method of implementing a control strategy capable of learning is discussed later in this section. It is worthwhile, however, to consider the design of a very simple control strategy. This strategy can be used to test the effectiveness of the adaptive system and the heuristic ideas incorporated in it will be useful in a more sophisticated strategy. One possible control strategy could be

a. User formulates query.
b. Use relevance weights for query terms and related terms to rank documents.
c. Present a few (say 5) of the top-ranked documents to the user.
d. Are relevant documents found?
   If yes, adjust weights, go to step b.
   If no and less than 10 documents from ranking shown, then show next 5 documents from ranking.
   Otherwise go to next step.
e. Try cluster search.
f. If relevant documents found, go to b
   otherwise try heuristic search.
g. If relevant documents or interesting terms found, go to b.

At any stage of this process, the user can terminate the search. The Boolean search would be used when the user specifies it or as an alternative in the heuristic search. This control strategy may not be the best possible and certainly more work can be done on its design, especially in making it more responsive to the characteristics of individual users. One feature of this type that could be added is an adjustment to allow for users who are specifically interested in high precision results (a few highly relevant documents) or high recall results (a large number of documents containing most of the relevant documents). The control strategy could be designed to take advantage of the bias in certain searches by knowing that, for example, cluster searching is precision-oriented and relevance feedback tends to increase recall.

The most important topic in this section is the development of a control strategy capable of learning. This work is based on the associative search network (ASN) of Barto et al [20]. Figure 2 shows a simple version of the ASN (not to be confused with the network of terms and documents).

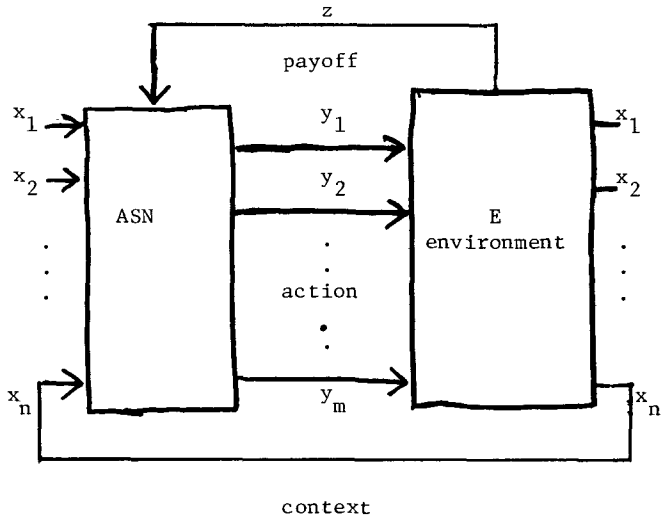


Figure 2 : The Associative Search Network

The ASN takes as input the context x_1, $x_2 \ldots x_n$ which are features of the environment (E) under study. On the basis of the context, the ASN chooses an action $(y_1, y_2 \ldots y_m)$ designed to maximize the payoff function z which also comes from the environment. The ASN is made up of a number of adaptive elements each of which determine a component of the system's actions. The output of an adaptive element is a function of a weighted combination of the context variables (x_i). The output function can be changed by changing the weights.

In terms of our document retrieval system, the environment is the retrieval system including the user, the actions are the various search strategies and the payoff function is some measure of user satisfaction, such as the number of relevant documents found at this stage. The context could be a wide variety of features such as the history of the search, the type of query and the type of user. The ASN has the capability of learning which action (search strategy) is the best to take given a particular context. A control strategy such as that discussed earlier in this section can be put into the ASN as a "first guess" that can be adjusted according to experience.

The more complicated the context, the more difficult it will be to evaluate the performance of the ASN. If a large amount of the context depends on a particular user, then the system is using a model of the user in a very similar way to an AI system.

2.3 User Interface

The previous two sections have shown that it will be important to design an interface that does not overburden the user but at the same time performs all the necessary tasks. The main use of the interface occurs in steps a and f of the control strategy described in the last section. Apart from these two steps, the system will be presenting retrieved documents to the user, possibly with the user determining the amount of output, and then obtaining relevance judgements. Obtaining relevance judgements is relatively easy as the user is simply required to specify relevance or non-relevance for the displayed documents. Some care must be taken to make sure that the user understands the difference between a relevant document and a useful document [16]. If a document appears that the user already knows, this will not be a useful document but it is relevant and this information can be used by the system to locate other useful and relevant documents.

Step f of the control strategy involves the heuristic search. The type of interaction that will take place here is described in section 2.1. Step a is the initial formulation of the query. Unless the user specifies a Boolean query, the main input in this step is the natural language statement of interest. The system would extract the index terms from this natural language query and display them to the user. It would then encourage the user to specify some order of importance for these terms. The user may even want to browse the database to find more satisfactory terms. Other information may be derived from the dialogue at this time that will

affect the model of the user used by the control strategy. For example, the system may ask whether the user is interested in recall or precision-oriented results. The interface must therefore be able to handle natural language queries, Boolean queries and conversational dialogue with the user at various stages of the search process. This will not be a complete natural language interface because the syntactic and semantic analysis required would lead to large storage and computational overheads compared to the simpler interface and it would take a major research effort for its design and the evaluation of its effectiveness. However, the interface designed should, like natural language, allow casual users easy access to the system.

3. Conclusion

The major advantages of the adaptive document retrieval system outlined in this paper are

a. The ability to efficiently use a number of different search strategies.
b. A representation that can be used for unconventional search strategies.
c. The ability to learn which strategy is the most effective for a given situation.
d. An emphasis on a flexible, friendly interface.

Combined together, these advantages imply that such a system will be more effective and much easier to use than conventional systems.

REFERENCES

1. Robertson, S.E.; Sparck Jones, K. "Relevance weighting of search terms." Journal of the American Society for Information Science, 27: 129-146; 1976.

2. Yu, C.T.; Salton, G. "Precision weighting - an effective automatic indexing method." Journal of the ACM, 23: 76-85; 1976.

3. Croft, W.B.; Harper, D.J. "Using probabilistic models of document retrieval without relevance information." Journal of Documentation, 35: 285-295; 1979.

4. Sparck Jones, K; Webster, C.A. "Research on relevance weighting 1976-1979." British Library Research and Development Report 5553, Computer Laboratory, University of Cambridge (1980).

5. Van Rijsbergen, C.J. "A theoretical basis for the use of co-occurrence data in information retrieval." Journal of Documentation, 33: 106-119; 1977.

6. Harper, D.J.; Van Rijsbergen, C.J. "An evaluation of feedback in document retrieval using co-occurrence data." Journal of Documentation, 34: 189-216; 1978.

7. Robertson, S.E.; Van Rijsbergen, C.J.; Porter, M.F. "Probabilistic models of indexing and searching." Proceedings of the ACM-BCS Symposium on Research and Development in Information Retrieval. Cambridge, England; 1980.

8. Croft, W.B. "Document representation in probabilistic models of information retrieval." Journal of the American Society for Information Science, (to appear).

9. Yu, C.T.; Luk, W.S. "Analysis of effectiveness of retrieval in clustered files." Journal of the ACM, 24: 607-622; 1977.

10. Croft, W.B. "A model of cluster searching based on classification." Information Systems, 5: 189-195; 1980.

11. Rich, E. "Building and exploiting user models." Technical Report CMU-CS-79-119, Carnegie-Mellon University.

12. Mark. W. "Consul's user model." Consul note 1, USC/Information Sciences Institute, 1980.

13. Salton, G. Automatic Information Organization and Retrieval, McGraw-Hill, New York, 1968.

14. Sparck Jones, K. "Automatic Indexing." Journal of Documentation, 30: 393-432; 1974.

15. Croft, W.B. "On the implementation of some models of document retrieval." Proceedings of the 2nd International ACM SIGIR Conference, SIGIR Forum, 15: 71-77; 1979.

16. Harper, D.J. "Relevance feedback in document retrieval systems: An evaluation of probabilistic strategies." Ph.D. Thesis, University of Cambridge, England (1980)

17. Oddy, R.N. "Information retrieval through man-machine dialogue." Journal of Documentation, 33: 1-14; 1977.

18. Van Rijsbergen, C.J. Information Retrieval. 2nd Edition. Butterworths, London (1979).

19. Croft, W.B. "Clustering large files of documents using the single-link method.", Journal of the American Society for Information Science, 28; 341-344; 1977.

20. Barto, A.G.; Sutton, R.S.; Brouwer, P.S. "Associative search network: A reinforcement learning associative memory." Biological Cybernetics, (In press).