Use of IR_Framework in the Teaching of Information Retrieval

Patrick Braekevelt and Steve Wade, *C^eDAR - Centre for Database Access Research, The School of Computing and Mathematics, The University of Huddersfield, Queensgate Huddersfield, West Yorkshire, HD1 3DH. (p.braekevelt@hud.ac.uk)*

This paper discusses recent work on the development of an application based on IR_Framework, an extensible class library for information retrieval currently under development at the University of Huddersfield. IR_Framework includes a range of data structures and algorithms which can be used in the development of applications in which it is necessary to store, automatically index and retrieve objects on the basis of their text content. The paper discusses the way in which IR_Framework has been used in the development of a reference management system called Papers and how this application has been used in the teaching of a course in Information Storage and Retrieval. Papers demonstrates a range of IR techniques and supports a hypertext based help system containing detailed information about these techniques.

INTRODUCTION

The work described here builds on our earlier work on the development of IR_Framework: a class library for information retrieval.[5,6] This work was in turn influenced by the work of Harper and Walker[2] IR_Framework has been designed to support the development of both best match and Boolean models of retrieval based on either inverted files or signature files and to provide the developer with facilities for implementing stopword and stemming facilities.

More specifically the class library has been developed to support the general application developer in the following areas:

- Automatic creation of indexes based on inverted files or signature files. This includes the provision of facilities for eliminating stop words and stemming terms prior to indexing and retrieval.

Multiple stopword lists should be supported to cater for different subject areas.

- Creation and maintenance of multiple document collections. Each collection may include different document types.

- Development of both probabilistic (i.e. "best match") and boolean models of retrieval. This should be supported in a manner which is largely independent of the type of indexing used. The best match search options will return a set of document identifiers, ranked on the basis of their similarity to the query. No such ranking will be possible with boolean queries which will simply return a set of document identifiers which satisfy the query.

- Provision of relevance feedback techniques that will allow queries to be re-weighted and/or expanded with new query terms to assist the user to improve an initial query.

The design of the framework and the way it can be used in the development of IR applications has been described elsewhere [5,7].

PAPERS

Papers is an interactive text retrieval system which has been developed using IR_Framework to support the user-friendly creation, maintenance and searching of "personalised" document collections such as files of references downloaded from an online system.

Papers supports Boolean and Best Match search options based on a full inverted file and a number of relevance feedback options. The application has been developed to allow users to “try out” these techniques as part of a course in Information Storage and Retrieval. Accordingly, a detailed hypertext-based help system has been provided to explain the algorithms and data structures at work behind the scenes.

WEIGHTING SCHEMES.
Papers demonstrates four predefined weighting schemes for use in Best Match Searching.

The simplest weighting scheme is where each term has equal weight and the total weight for a document is determined by the number of terms in common with the query. i.e. a document that has 5 terms in common with a query will be assigned a score of 5.

The Inverse Document Frequency can be used as determined by the following formula[4]:

$$IDF_i = 1 + \log_2\left(\frac{N}{n_i}\right)$$

where IDF is the Inverse Document Frequency Weight
n_i = number of documents term t(i) occurs in
N = number of documents in the collection
IDF_i = collection frequency weight for term t(i)

Note the addition of 1 to prevent zero values, so all query terms get at least a weight of 1. The log would be zero when n equals N.

Term frequency
If a term occurs a large number of times in a document, then that term is likely to be descriptive of the documents content. Thus while a term's collection frequency remains the same for any document , the document frequency of that term changes for each document.

Term frequency needs to be related to the document length because a term which occurs the same number of times in a short document and a long document is likely to be more important for the former. The

length of a document is therefore defined as the total number of term occurrences in that document [1].

$$similarity_j = \sum_{i=1}^{Q} \frac{\log_2(freq_{ij} + 1) \times IDF_i}{\log_2 length_j}$$

Therefore there are some predefined weighting schemes implemented that take account of these factors so that students can experiment with these. The students can experiment with relevance feedback, query expansion and see the difference that a weighting scheme makes on finding relevant documents.

RELEVANCE FEEDBACK

After a query has been executed and results have been obtained, the user can then look at each document and mark it as relevant if need be. After a few documents have been marked as relevant the system then uses these relevant documents to obtain a list of words. These words are ordered according to a simplified version of the following formula [3].

$$porter = \frac{r}{R} - \frac{n}{N}$$

r: number of relevant documents the term occurs in
R: total number of relevant documents for this query
n: collection frequency for the term
N: total documents in the collection

The algorithm sorts terms on the number of relevant documents they occur in. To distinguish between terms occuring in an equal number of relevant documents, we can use the 'specificity' of the term.
We can gain a measure of the specificity of the term by knowing how many documents it appears in over the whole collection. (n/N)

Since our feedback cycle refers to only one query, the same value of R is applied to all the extracted terms. We have therefore simplified the formulae by ignoring R .

The words are then displayed to the user who can select terms from that list and start a complete new search , thereby possibly obtaining relevant documents that were not retrieved with the initial version of the query.

QUERY EXPANSION

Words that co-occur frequently in documents are likely to be related to each other. Thus, a useful way to expand the query term is to find all the indexing terms that tend to co-occur with it.

A facility has been provided where a term can be expanded . Each document that the term is found in , is broken up into its constituent words. For each word the total number of documents is calculated in which both terms occur. With these data it is then possible to get a measure of cohesion between the two terms given their respective collection frequencies . Each term that fulfills certain criteria is then added to the expanded term list which is ordered in measure of coocurrence.

The formula[1] used is

$$Cohesion(A,B) = \frac{freq(A,B)}{\sqrt{freq(A) \times freq(B)}}$$

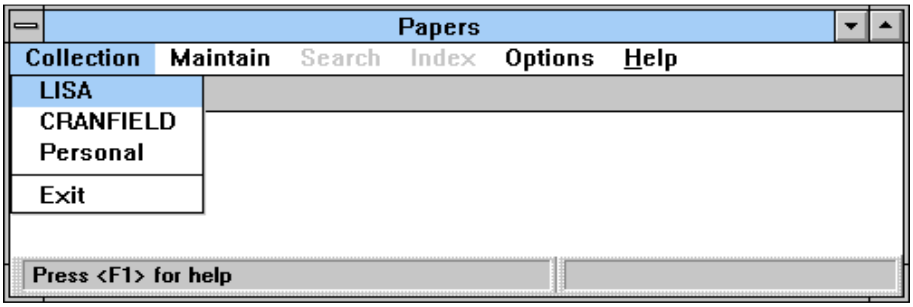
freq(A,B) represents the cooccurrence frequency of term A and B
freq(A) represents the number of postings associated with A
freq(B) represents the number of postings associated with B

If term A always coocurred with B then we can see that the maximum value for the cohesion would be 1 , and when there was never any coocurrence the minimum value would be 0. It is possible to set a threshold to limit the number of terms that would be included in the list of expanded terms.

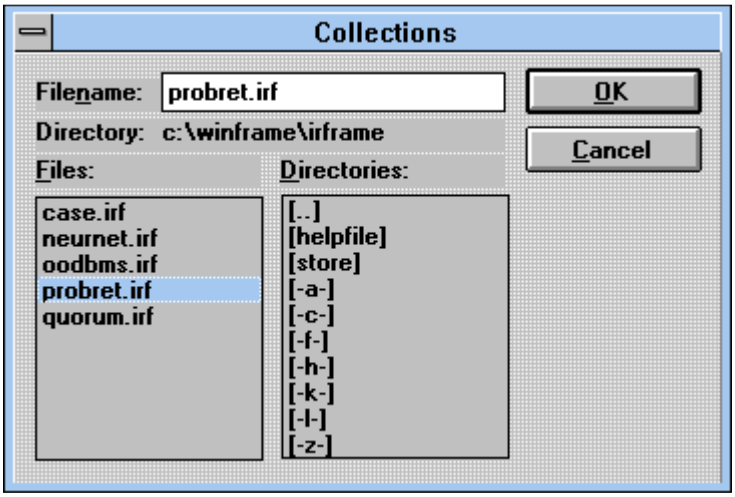
A SAMPLE SESSION WITH PAPERS

This section includes screen dumps for a typical search session with Papers.

Upon entering the system, the user is presented with a series of options in a menu across the top of the screen. If the user selects "Collection" they are presented with the following:

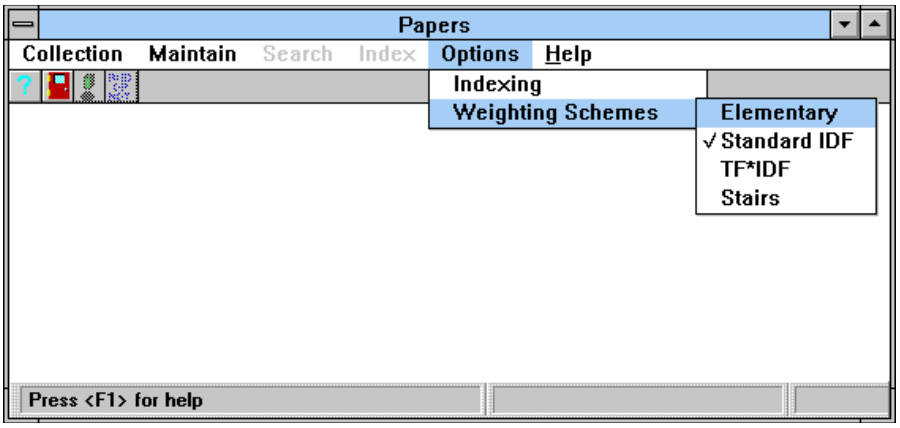


The LISA and CRANFIELD collections are test collections.
Selecting the "Personal" option gives the following screen:

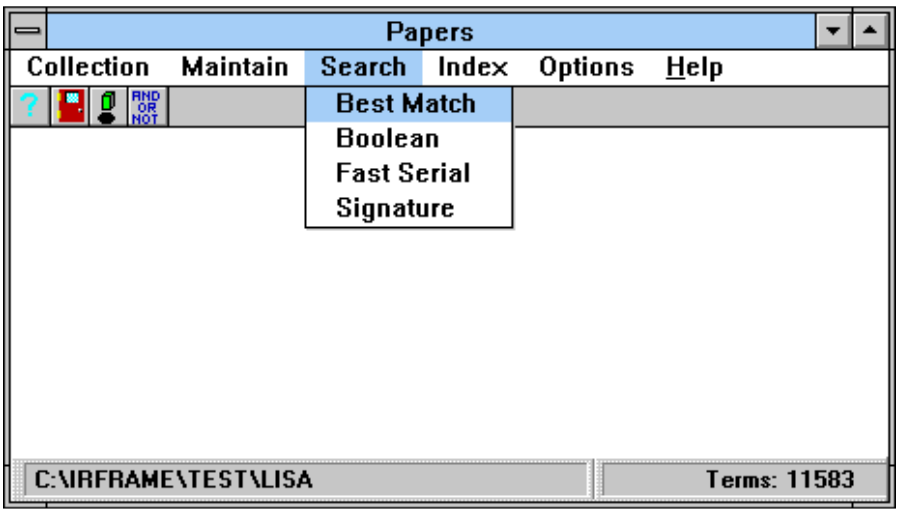


These five collections have been created by the user and contain references from downloaded file(s) obtained from CD-ROM or online searches. The searches will have been quite general in character intended to retrieve several hundred references on a particular topic.

The following screens are taken from a session in which the LISA collection was chosen as the search database.

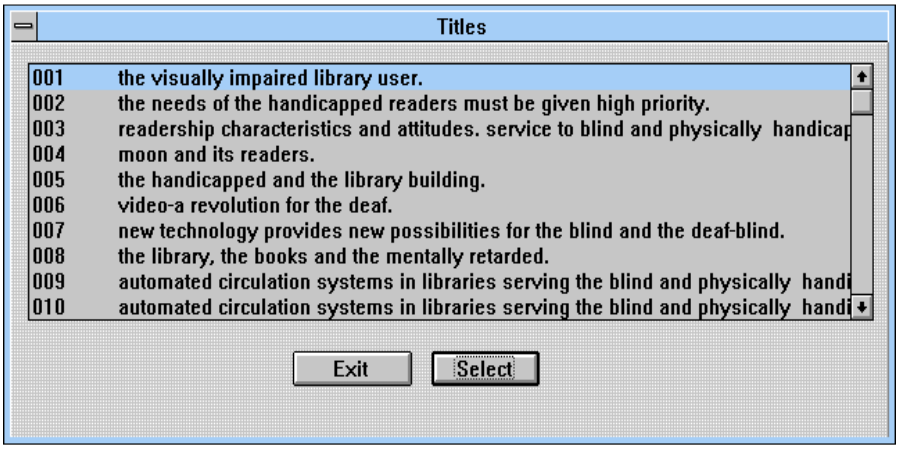


The user can select the weighting scheme to be used

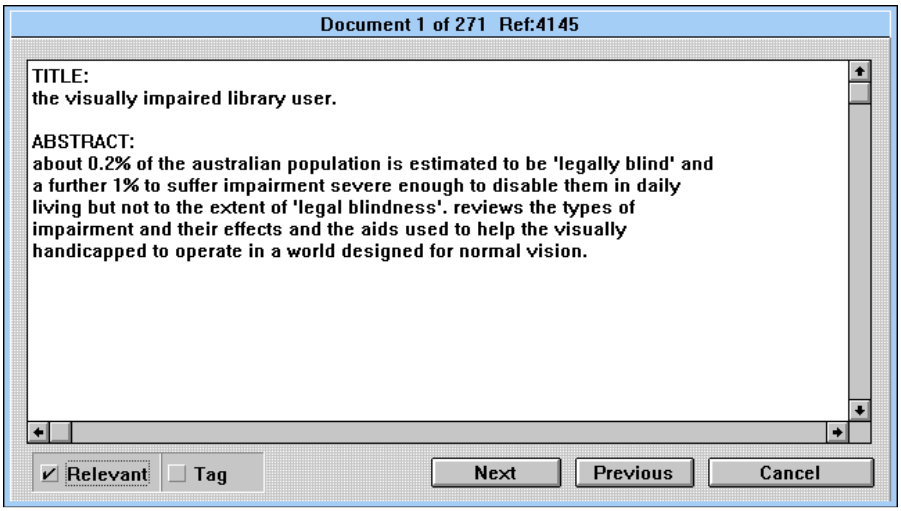


Initially the best match search option is selected and a natural language query entered:

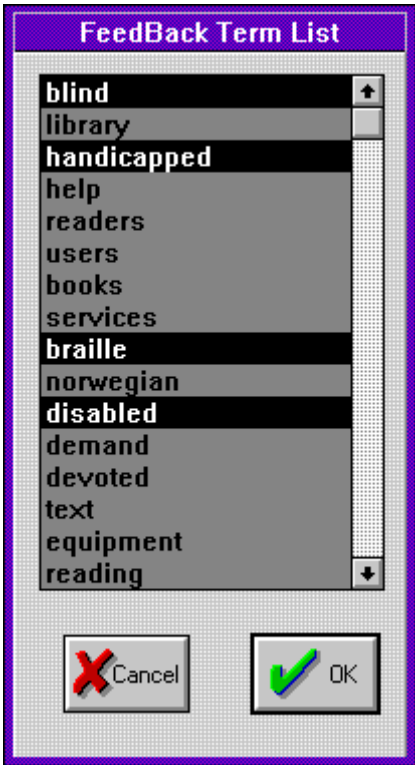
The user selects “Titles” indicating that they wish to see the titles of the documents in ranked order. The default option would display each document title and abstract in turn:



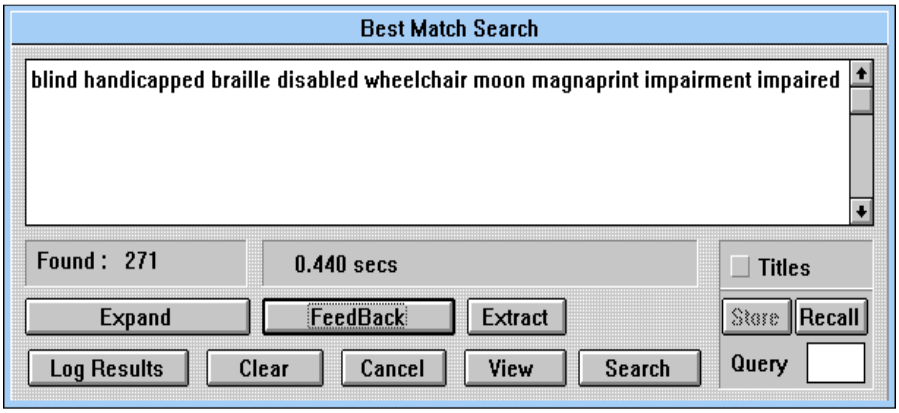
The user can select any of these titles and “double click” on it to see the abstract. The user can then use the “Relevant” option to indicate that this reference is of particular interest:



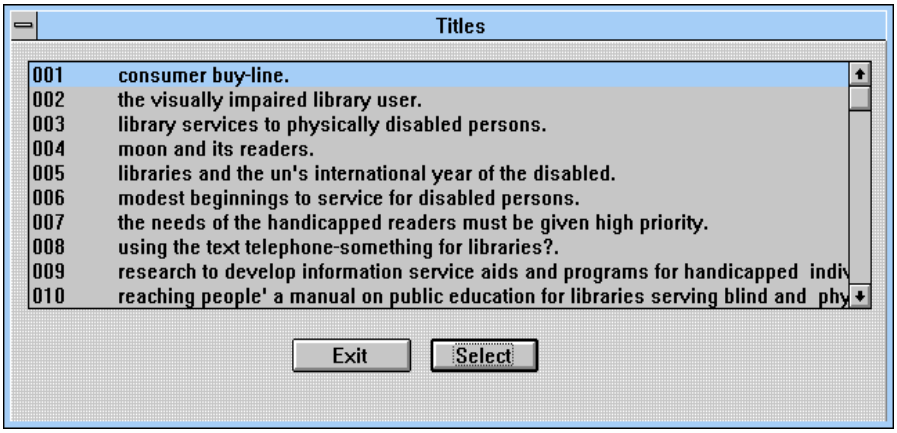
Having identified a number of documents as relevant the user returns to the main screen and selects “Feedback” the system then retrieves and ranks all terms occuring in relevant references. These are displayed to the user, who can select any number of them to include in a revised version of the query:



After scrolling through this list and selecting a number of new terms, the user’s modified query looks like this:



A search with this modified query yields the following list:

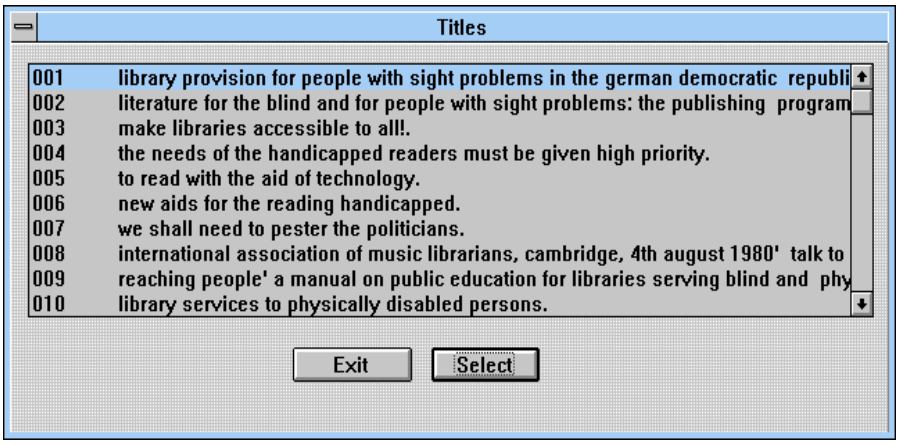


An alternative way to expand the query is to highlight a term from the query and choose the “Expand” option:

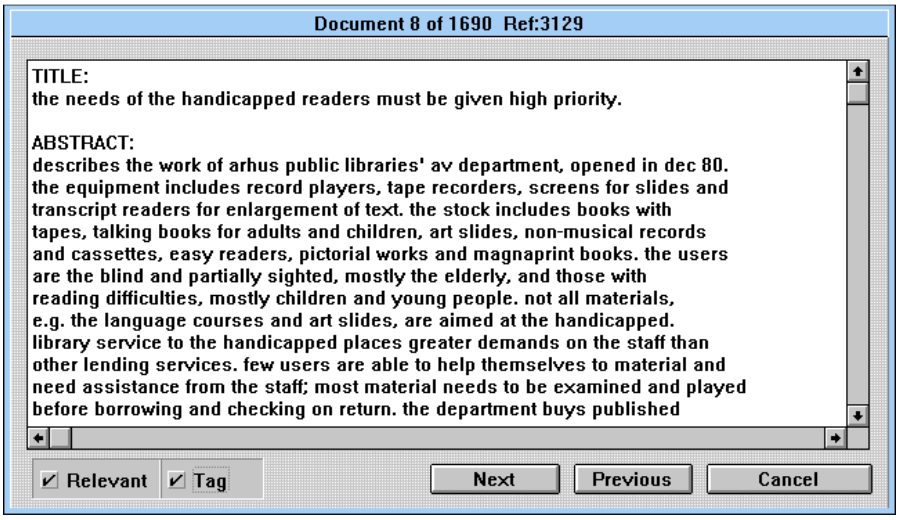
The system will then identify those terms that co-occur in the index with the selected term.

Again terms from this list can be selected and added to the query:

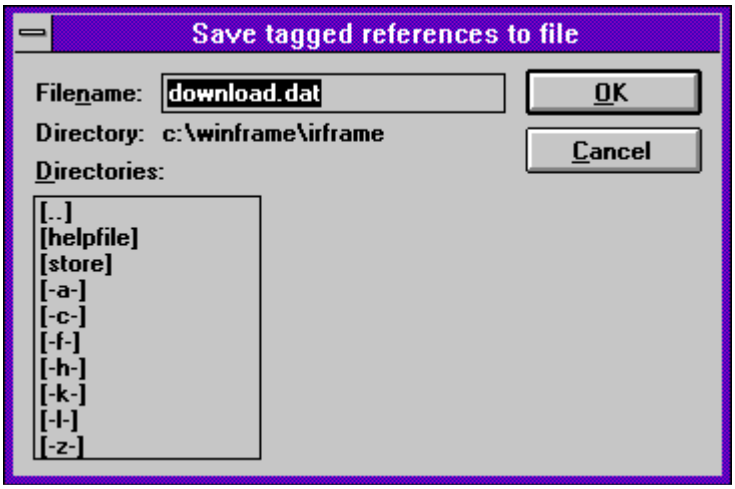
The titles retrieved with this version of the query are as follows:



In addition to identifying documents as relevant, users can “Tag” documents by selecting the Tag option.



All documents that have been tagged can be saved to a file as follows:



CONCLUSIONS

In this paper we have discussed Papers, an IR application developed using IR_Framework. Currently this application is being used in the teaching of ISR. The Papers application demonstrates a number of techniques which have been proven to work well by IR researchers but which are not available in many commercial packages. The software allows students to compare rankings produced with different weighting schemes with the output they can obtain using boolean search strategies.

Students can use the software to search a number of existing document collections (including the LISA and Cranfield test collections) or to index and search their own collections which may have been created from the output of an online or CDROM search.

REFERENCES

1. FRAKES, W.B. AND BAEZA-YATES, R. (1992). *Information Retrieval: Data Structures and Algorithms.* Prentice Hall. *Chapter 14: Ranking Algortihms Harman, D.*

2. HARPER, D.J. and WALKER A.D.M (1992) .*ECLAIR: An Extensible Class Library for Information Retrieval. The Computer Journal vol 34 No 3 pp 256-266.*

3. PORTER, M. and GALPIN, V. (1988) *Relevance feedback in a public access catalogue for a research library: Muscat at the Scott Polar Research Institute. Program vol 22 pp 1-20*

4 SPARCK JONES, K. 1972 *A statistical Iterpretation of Term Specificity and Its Application in Retrieval. J.Documentation,28(1),11-20*

5. WADE S., BRAEKEVELT, P. AND SUTTON D. (1994) *IR_Framework: A class library for information retrieval. Internal Research Report. University of Huddersfield, Dept. of Computing and Mathematics.*

6. WADE S.J. *AND* WILLETT P. (1988) *INSTRUCT: A teaching package for experimental methods in information retrieval. Part 3. Browsing, clustering and query expansion. Program 22 pp. 44-61.*

7. WADE S.J. AND BRAEKEVELT P. (1995) *IR_Framework:an objecte oriented framework for developing information retrieval systems. Program 29 pp 15-29*