An Approach for Implicitly Detecting Information Needs

Ryen W. White
Information Retrieval Group
Department of Computing Science
University of Glasgow
Glasgow, UK. G12 8QQ.
ryen@dcs.gla.ac.uk

Joemon M. Jose
Information Retrieval Group
Department of Computing Science
University of Glasgow
Glasgow, UK. G12 8QQ.
jj@dcs.gla.ac.uk

Ian Ruthven
Department of Communication and Information Sciences
University of Strathclyde
Glasgow, UK. G1 1XH.
ir@cis.strath.ac.uk

ABSTRACT
Searchers can have problems devising queries that accurately express their, often dynamic, information needs. In this paper we describe an adaptive approach that uses unobtrusive monitoring of interaction to help alleviate such problems and support searchers in their seeking. The approach we propose implicitly selects terms to better represent information needs, gathers evidence on potential changes in these needs, and uses this evidence to tailor the result presentation accordingly. A user evaluation of an interface implementing our approach, presented in [7], shows it can select terms that approximate current information needs and provide evidence to track changes in these needs.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: - search process, relevance feedback

General Terms
Theory, Design, Human Factors

Keywords
Information need detection, query expansion, implicit feedback

1. INTRODUCTION
In light of the Internet's seemingly inexorable growth, the value of search systems that help searchers find useful information is becoming increasingly apparent. Searchers are usually required to express their information need via a set of query terms submitted to the search system. The transformation of this need, implicit in the mind of the searcher, into a search expression, or query, is known as *query formulation*. However, queries are only an approximate, or 'compromised' information need [5] and may fall short of the description necessary to infer relevant documents. Consequently, search systems need to offer robust, reliable methods of query modification.

Relevance feedback (RF) [4] is the main post-query method for automatically improving a system's representation of a searcher's information need. However, the technique relies on explicit relevance assessments provided by the searcher: indications of which documents contain relevant information.

CIKM'03, November 3–8, 2003, New Orleans, Louisiana, USA.

This is a demanding and time-consuming task that places an increased cognitive burden on those involved [2]. As a result, searchers may be unwilling to provide such feedback.

In this paper we present an approach for approximating the current information needs of searchers through the implicit (unobtrusive) monitoring of their interaction. Based on this interaction we rank terms on how well they represent current information needs. Over time, we apply statistical methods to successive lists of query expansion terms and use the resultant evidence to predict the degree of change (or development) in a searcher's information need. As we show, different degrees of perceived change result in different system responses.

2. OVERVIEW OF APPROACH
The main aim of our approach is to develop a means of better representing searcher needs whilst minimising the burden of explicitly reformulating queries or directly providing relevance information. RF systems typically adopt a binary notion of relevance: either a document is relevant, or it is not. If a document is only partially relevant the approach requires the searcher to choose between these two extremes, i.e. there is no middle ground. In such circumstances it can be difficult for searchers to make decisions on what documents to assess as relevant.

The methods we propose do not impose such burdens or present such difficulties to the searcher. Through implicitly monitoring interaction at the results interface, searchers are no longer *required* to assess the relevance of a number of documents, or indeed consider entire documents for such relevance. Our approach makes inferences based on interaction and selects terms that approximate searcher needs.

Traditional RF systems require the searcher to instruct the system to perform RF, i.e. perform query modification and produce a new ranked list of documents. However, this is only one way of using relevance information and may not always be appropriate. Information needs are dynamic and can develop in a dramatic or gradual manner [1]. For gradual changes, the generation of a new result set is perhaps too severe, and revisions that reflect the *degree* of development may be more suitable.

Our approach uses the evidence it gathers to track potential *changes* in information need and tailor the results presentation to suit the *degree* of change in need. Large changes in perceived information needs result in new searches but smaller changes result in less radical operations, such as re-ranking the list of retrieved documents or re-ordering representations of the documents presented. In the next section we describe how granular document representations joined with a path metaphor

are used to select potentially utile terms that are then weighted for query expansion.

3. SEARCHER INTERACTION

We use a content-rich interface, presented in [7], to encourage searcher interaction and provide evidence for our implicit algorithms.

For each document there are six different representations that a searcher can view. These include the title and full-text of the document as created by the author, a query-biased summary of the document [6] and the summary's component sentences. A list of sentences from the top thirty documents retrieved, scored in relation to the query, called the *top-ranking sentences* (TRS), includes a maximum of four sentences from each document. Each sentence in the top-ranking sentence list is regarded as a representation of its source document. Finally, for each sentence in the summary the system can present that sentence in the context it occurs in the document (i.e. with the preceding and following sentence from the full-text of the document).

TRS → Title → Summary → Summary Sentence → Sentence in Context → Document

Figure 1. Representations in a path

The six representations combine to form an ordered *relevance path*. For example, passing the mouse over the title will show the summary, passing the mouse over a sentence in the summary will show the sentence in context, etc. Certain aspects of the path order are dictated by the interface; the search must request a summary before being able to see a sentence in context for example. However, a searcher can view titles and access full-texts as in standard Web search interfaces; they are not required to use any of the novel interface options.

Through their interaction, searchers have control over which representations they view. That is, a searcher does not need to complete a path for a document, the searcher can stop a path when the information contained in the representation is not of interest. The default display, shown when the result page loads, is the list of document titles and the list of top-ranking sentences.

The distance travelled along the path by the searcher can provide information on the relevance of terms used in the path representations. In the next section we describe how we use the representations to calculate the relevance weight of these terms.

4. BINARY VOTING MODEL

The representations viewed by a user are used to select query expansion terms. To identify potentially useful expansion terms we propose a *binary voting model* in which each representation 'votes' for the terms it contains. When a term is present in a viewed representation it receives a 'vote', when it is not present it receives no vote. All terms are candidates in the voting process, and these votes accumulate across all viewed representations. The winning terms being those with the most votes, and hence best describe the information viewed by the searcher. Our assumption here is that useful terms will be those contained in many of the representations that the user chooses to view.

However, different types of representation vary in length, and hence can be regarded as being more or less indicative of the content of the document. For example, a top-ranking sentence is less indicative than a query-biased document summary (typically composed of four sentences) as it contains less information about the content of the document. To counter this we *weight* the contribution of a representation's vote based on the indicative worth of the representations, e.g. we consider the contribution that viewing a top ranking sentence makes to the system's understanding of which terms are relevant to be less than a summary.

The weights used in the system used in our experiments are 0.1 for title, 0.2 for TRS, 0.3 for Summary, 0.2 for Summary Sentence and 0.2 for Sentence in Context. For example all terms in a viewed summary will receive a weight of 0.3, all terms in a viewed summary sentence will receive a weight 0.2, etc. The weights were defined for experimental purposes but were based on beliefs about the indicative worth of each representation and ensure that the total score for a term is between 0 and 1 (inclusive).

The decision to use binary (term presence/absence in a representation) rather than term frequency (*tf*) information was taken for reasons of simplicity and computational expense. We tested the effectiveness of other methods of term weighting such as *tf*, *tf.idf*, *tf* normalised by representation length, none of which performed better than binary voting, and in the case of *tf.idf*, performed worse.

The model is a simple solution to a potentially complex problem. The terms that eventually 'win' the vote are those that are taken to best describe the information viewed by the searcher (i.e. those terms that are present most often across all representations) and can, therefore, be used to approximate searcher interests.

As was shown in Figure 1, multiple representations can form a relevance path for each document. Some of the representations for each document are fixed in content, i.e. the title and full-text of the document, whereas other representations, such as the summary, are dependent on the query and are hence variable in content. Therefore, for each document, there may be many *potential* relevance paths. We use the distance travelled along the path and the particular representations used in the path to calculate a list of expansion terms for query modification.

In the voting model, each document is represented by a vector of length n, where n is the total number of unique non-stopword terms (including query terms) in the top 30[1] Web documents. We refer to the list holding these terms as the *vocabulary.*

We aim to construct a document × term matrix, $(d+1)\times n$, where d is the number of documents for which the searcher has travelled at least part of the path (Figure 2). Each row in the matrix is all n terms in the vocabulary [i.e. $(t_{k1},t_{k2},\ldots,t_{kn})$ where k is the row number], and each term has a weight. An additional row is included for the query.

$$\begin{array}{c} \\ q_0 \\ D_1 \\ D_2 \\ \\ D_d \end{array} \begin{array}{c} \begin{array}{cccc} t_1 & t_2 & & t_n \end{array} \\ \begin{bmatrix} t_{01} & t_{02} & \ldots & t_{0n} \\ t_{11} & t_{12} & \ldots & t_{1n} \\ t_{21} & t_{22} & \ldots & t_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ t_{d1} & t_{d2} & \ldots & t_{dn} \end{bmatrix} \end{array}$$

Figure 2. Document × Term matrix

Terms in the query vector are initially assigned a weight of 1 if they are included in the query and 0 if not. This vector is then normalised to give each term a value in the range [0,1] and ensure

[1] Only 30 retrieved documents are used for analysis to ensure the system responds in a timely manner.

the sum of all values is 1. This ensures that the query terms are not weighted too highly in the document × term matrix.

We treat each document representation as a source of terms, and the act of viewing a representation as an implicit indication of relevance. When a searcher visits the first representation of a document we add a new row to the document × term matrix. This row is a vector of length n, where n is the size of the vocabulary and all entries are initially set to 0. If a term occurs in a representation, no matter how many times, it is assigned a weight, w_t, which is based on the representation that contains the term.

This weight for each term is *added* to the appropriate term/document entry in the matrix. Weighting terms is therefore a *cumulative* process; the weights calculated for a term in one representation are added to the weights calculated for the preceding steps in the relevance path. The total score for a term in a document is computed by:

$$w_{t,D} = \sum_{r=1}^{p} \left(w_{t,r} \right)$$

where p is the number of steps taken, D is the document, t is the term, r is the representation and $w_{t,r}$ is the weight for representation r.

If the searcher visits one representation of a document and then goes onto the next representation in the path of that document, *at any time – not necessarily immediately*, we add the term scores to the row in the matrix occupied by that document. The scoring is cumulative, if a document already has a row in the matrix it does not get a new one.

If the searcher is at an individual step in the path and views another representation of the same type, i.e. views one sentence in a summary and then views a second sentence, we only use the most recent instance of the representation, i.e. the second sentence for updating. This ensures that the terms chosen relate to the searcher's *current* information need. The current version of our system does not consider more detailed interaction.

The matrix resulting from this process reflects the weights based on the final path viewed by the searcher. Each document may have a different length path and will contain different terms. Hence, we have a summary of which terms are important in each relevance path. This information is used for query modification, as will be described in the next section.

5. INFORMATION NEED CHANGE

To provide an appropriate level of support to the searcher, our approach uses a history of recent interaction and predicts changes (or developments) in the information need. This history provides insight into the recent interests of the searcher, and by comparing this with previous histories we track possible changes in the information need. Selecting the most appropriate form of support depends on the extent to which the need is seen to change. The smaller the change, the less radical the support offered. Tailoring the support in this way allowed the interface to work in concert with the searcher.

In the matrix created by the binary voting model, only the query terms and terms in representations viewed by the searcher will have a score greater than zero. These terms are potentially useful for query expansion. One novel aspect of our system is how we use these terms; the system detects the change in which terms are suggested by the system for query expansion and, based on the degree of change the system decides how the new query should be used. In this section we describe how we detect a change in suggested expansion terms and the resultant action.

For every ten paths we compute a new query. This allows the system to gather sufficient evidence of relevance from searcher interaction. It is possible for a relevance path to contain only one representation. Therefore, for the searcher to follow ten paths they need only view a representation from ten unique documents.

To compute the new query we calculate the average score for each term across all documents (i.e. down each column in the matrix). This gives us an average score for each term in the vocabulary. The terms are then ranked by their average score. A high average score implies the term has appeared in many viewed representations across the documents viewed. The top six ranked terms are used to form the new query. It is possible that this may not contain the searcher's original query terms. However, focusing these representations on the query increases the likelihood that the original terms will be present in the new, expanded query. It is only in situations where the information need changes dramatically that some of these original terms may be replaced. In situations where searchers submit initial queries containing more than six query terms, the expansion is limited to a maximum of ten terms. For example, if the initial query contained seven terms, then only the top three expansion terms would be added to the query. This proved adequate for our purposes. The degree of change between successive term lists (groups of ten paths) provides evidence to track the degree of change in a searcher's information need.

For each set of thirty retrieved documents the vocabulary is static, so we can gauge the level of change in the information need by comparing the change in the term ordering from the previous term list (q_m) to the new term list (q_{m+1}). As the vocabulary is static, the *terms* in the list will not change, only their order. So, by comparing q_m against q_{m+1} based on some operator ⊙ we can compute the degree of change between the lists and hence the information need. This can be shown formally as:

$$\Delta\psi = (q_m) \odot (q_{m+1})$$

where ψ is the system's view of the searcher's information need and ⊙ computes the difference between two lists of unique terms taken from all paths.

In our system we use the *Spearman rank-order correlation coefficient* for ⊙ in this computation. This coefficient tests for the degree of similarity between two lists of rankings. The test is non-parametric, so rankings, not the actual term scores, are used. We have two lists of terms representing q_m and q_{m+1} respectively. The first list is ordered by average term score; the second list contains the terms in the same order but updates the rankings (i.e. we assign new rankings, but we *do not sort*). An example is given below where n is the rank order of a term in relation to other terms.

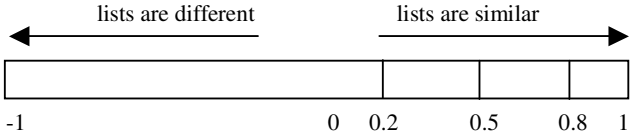
	q_m		q_{m+1}
t_1	1	t_1	3
t_{12}	2	t_{12}	2
t_3	3	t_3	6
t_{41}	4	t_{41}	1
	…		…
t_n	n	t_n	n'

The Spearman rank-order correlation coefficient returns values between –1 and 1, where 1 is *perfect positive correlation* (the lists are exactly the same), –1 is *perfect negative correlation* (the lists are the complete opposite i.e. 1,2,3,4,5,6,7,8 vs. 8,7,6,5,4,3,2,1) and any value in-between is reflective of their relation to these extreme values. A correlation of 0 implies *zero* (or no) *correlation* between the two lists. We handle ties in the standard statistical way, by summing the rank of all tied elements and dividing this sum by the number of elements, effectively taking the average rank for each group of ties.

All terms in the original vocabulary are ranked and present in both lists. However, there is a high level of redundancy in each list as the lower ranking terms that never appear in a viewed representation experience only slight changes in their ranking between iterations. To counter this problem we use only the top 100 terms. These are the most liable to change and therefore most likely to reflect any change in the information need.

We compare the lists every time we compute a new query (i.e. every ten paths). To compute the correlation coefficient both lists must contain the same terms and the same number of terms. Therefore, in practice we need to use the first 100 terms plus α. The α is the number of terms that have left or joined the top 100 terms between q_m and q_{m+1}. For terms *joining* the top 100, we sort them based on their original (q_m) ranks and assign them ranks (in q_m) in the range [101,101 + α]. We use the same procedure for terms that are *leaving* the top 100, except these terms are ranked based on their new (q_{m+1}) ranks.

We then have the coefficient in the range [–1,1], where a result closer to –1 means the term lists are dissimilar with respect to their rank ordering. As the coefficient gets closer to 1, the similarity between the two query lists increases. Based on the coefficient value returned we decide how to use the new list of terms. We implement three strategies:



i. Re-searching. If the coefficient value indicates that the two term lists are substantially different with respect to rank ordering, we take this to reflect a large change in ψ (the system's view of the information need). In this case we re-search the Web to retrieve a new set of documents. As will be explained in section 5, the searcher must request to view the results of the new search; the new result set is automatically generated in the background but the searcher must request to see it. Coefficient values of less than 0.2 are taken to indicate a large change in the term lists.

ii. Reordering documents. A result in the range [0.2,0.5] indicates a weak correlation between the two lists and consequently a less substantial change in ψ. Here we use the new query (i.e. the six top ranked terms) to reorder the retrieved documents using best match *tf.idf* scoring.

iii. Reordering TRS. Coefficients in the range [0.5,0.8] indicate a strong correlation between the two term lists and hence a small change in the system's view of the information need. In this case we use the new query to re-order the TRS list based on the term-occurrence of each of these expansion terms. The sentences are the most granular elements presented to the searcher and are therefore most suited to reflect minor changes in ψ.

Strategies ii. and iii. provide an updated view of the retrieved documents based on the current ψ. For differences between 0.8 and 1, the need is assumed to have not changed sufficiently to warrant action. All numerical bounds are experimental, chosen during pilot testing of the interface. The implementation of this approach for detecting information change was for investigative purposes only and is not a definitive solution.

6. DISCUSSION

We have presented an adaptive approach that uses implicit monitoring to detect the current state of a Web searcher's information need. To approximate current needs we do not use traditional, potentially unreliable [3], implicit sources of searcher preference (e.g. document reading time, scrolling), but interaction with granular document representations and paths that join them. The techniques introduced have the potential to alleviate some of the problems inherent in explicit relevance feedback whilst preserving many of the benefits that underlie the approach. The initial query is still expanded to become more attuned to a searcher's need based on an iterative process of feedback. However, there are three key differences; searchers do not have to explicitly assess and mark documents relevant; these documents are not the finest level of granularity and the way the expanded query is used depends on the extent to which the information need has changed (i.e. we do not simply re-search). Devising systems that adapt to the information needs of those who use them is an important step in developing systems to help struggling searchers find the information they seek.

ACKNOWLEDGMENTS

The work reported is funded by the UK Engineering and Physical Sciences Research Council grant number GR/R74642/01.

REFERENCES

[1] Campbell, I. 'Interactive Evaluation of the Ostensive Model, using a new Test-Collection of Images with Multiple Relevance Assessments'. Journal of Information Retrieval. **2**. 1. pp 89-114. 1999.

[2] Goecks, J. and Shavlik, J. 'Learning users' interests by unobtrusively monitoring their normal behavior'. Proceedings of the International Conference on Intelligent User Interfaces. pp. 129-132. 2000.

[3] Kelly, D. and Belkin, N.J. 'Reading Time, Scrolling and Interaction: Exploring Sources of User Preferences for Relevance Feedback During Interactive Information Retrieval'. Proceedings of the 24th Annual ACM SIGIR. pp. 408-409. 2001.

[4] Salton, G. and Buckley, C. 'Improving retrieval performance by relevance feedback'. Journal of the American Society for Information Science. **41**. 4. pp. 288-297. 1990.

[5] Taylor, R.S. 'Question-negotiation and information seeking in libraries'. College and Research Libraries. **29**. pp. 178-194. 1968.

[6] White, R.W., Jose, J.M. and Ruthven, I. 'A task-oriented study on the influencing effects of query-biased summarization in web searching'. Information Processing and Management. **39**. 5. pp. 707-733. 2003.

[7] White, R.W., Jose, J.M. and Ruthven, I. 'Adapting to Evolving Needs: Evaluating a Behaviour-Based Search Interface'. Proceedings of the 17th Annual Conference on Human Computer Interaction. in press. 2003.