Improving Web Search Through Collaborative Query Recommendation

Evelyn Balfe and Barry Smyth [1]

Abstract. Search engines are the primary means by which people locate information on the Web. Unfortunately most Web users are not information retrieval experts and there is a tendency for Web queries to be ambiguous and under-specified. Query expansion and recommendation techniques offer one way to solve the ambiguous query problem in Web search, by automatically identifying and adding new terms to a vague query in order to focus the search. In this paper we describe and evaluate a novel query recommendation technique based on reusing previous search histories. This is achieved by selecting, ranking, and then recommending previously successful queries to users. Its novelty stems from the way in which queries are scored and ranked using relevance and coverage factors in order to prioritise those queries that are most likely to be successful in the current search context. We demonstrate that these recommendations can lead to improved search performance based on live-user data.

1 INTRODUCTION

Prominent search engines continuously develop new techniques to aid Web searching. Unfortunately even the leading search engines frequently fail to satisfy a user's stated information needs, the reasons for this often lie with the searcher as much as the search engine. At their core, modern Web search engines rely heavily on information retrieval techniques that were originally designed with information retrieval experts in mind, information retrieval experts who are trained to produce precise queries. These techniques were never designed to cope with the vague queries that are commonplace on the Web. For example, researchers have highlighted how the average Web query consists of only 2 or 3 search terms [12]. Because of these difficulties, many of the more recent developments in Web search have attempted to improve the way that Web search engines respond to vague queries. This includes the development of new indexing and ranking techniques that take advantage of the structural properties of the Web, as well as relying on the content of Web documents (see for example, [3, 10]). As a more direct response to vague queries, researchers have looked at ways in which additional search context can be inferred and used to elaborate an initial user query (e.g., [11, 17]).

Another common strategy involves expanding or elaborating vague queries, or even recommending new queries, to provide users with more precise search terms. It is this approach that mainly interests us in this paper. In Section 4 we describe and evaluate a novel query recommendation technique that suggests queries that have proven to be successful during past search sessions. In addition we propose a novel ranking metric that prioritises queries based on their relevance and coverage characteristics. Finally, in Section 5 we show that, when evaluated on live user data, this ranking metric produces superior query rankings in comparison to alternative metrics. But first we outline related work in the area of query expansion and recommendation and summarise our work in the area of collaborative search (Section 3), which serves as a backdrop to our new query recommendation technique.

[1] Smart Media Institute, University College Dublin, Ireland. email: firstname.lastname@ucd.ie

2 RELATED WORK

A variety of methods have been developed to assist Web users with their searching tasks. In this section we focus on those methods that centre around query formation. These techniques can generally be classified into three areas: introducing context to a search, using relevance feedback techniques and taking advantage of the information contained in query logs.

2.1 Context in Search

Vague queries are often problematic because they lack context; the query 'jaguar' does not distinguish between an automobile, wildlife or the operating system, any of which might be relevant to the searcher. Context can be captured according to two basic approaches: either by explicitly establishing context up-front or by implicitly inferring context. Explicit context is relied upon by the Inquirus 2 system [9] in that users are expected to select category information prior to search (e.g., 'research paper', 'homepage' etc.) and this information is then used to focus the search.

Without a doubt, user's explicit judgements are the most reliable source of context. Unfortunately users are often reluctant to stop and provide explicit feedback. The alternative is to automatically infer context information. This can be achieved by monitoring the user's activity immediately prior to search. For example, the Watson system monitors user's Word Processing activity and uses extracted keywords to initiate search sessions [5]; see also Letizia [13].

2.2 Relevance Feedback

For many years researchers have suggested the use of relevance feedback as a source of query modification. The basic idea is to elicit feedback from the user to indicate the relevance (or non-relevance) of a particular retrieval result. This feedback, and information about the terms contained within the documents, are used to suggest additional query terms (see for example, [15, 16]). Unfortunately, casual users (and Web searchers especially) are rarely inclined to provide direct relevancy judgments. For this reason researchers have proposed the use of *ad hoc* or *blind* feedback as a form of *pseudo* relevance feedback, without the need for direct user input. For instance, one common approach (e.g., [4]) is to assume that the top k documents retrieved are all relevant and to proceed according to a standard relevance feedback technique. Obviously a potentially severe disadvantage with blind feedback approaches is that top-ranking documents may not be truly relevant. A number of coping strategies have been proposed and positively evaluated for preventing the expanded query drifting off topic (see for example, [14]).

2.3 Query-Log Analysis

Finally, it is worth highlighting a relatively new strand of related research, one that emphasises the value of historical search session information contained within query logs. This approach is particularly well suited to Web search applications given that the query logs of Web search engines are liable to contain vast quantities of valuable information. For example, [7] look for correlations between query terms and document terms that can be mined from a search engine's query log. The basic idea is that if a set of documents is often selected for the same queries then the terms in these documents must be strongly linked to the terms in the queries. These correlated document terms serve as candidate expansion terms (see also [19]).

3 A REVIEW OF COLLABORATIVE SEARCH

Collaborative search is motivated by two key ideas. First, specialised search engines attract communities of users with similar information needs. For example, a search field on an AI Web site is likely to attract queries with a computer-related theme, and queries such as '*cbr*' are more likely to relate to Case-Based Reasoning than to the Central Bank of Russia. This idea is also supported by the growing emphasis that is being placed on the importance of social networks on the Web with services like Friendster (www.friendster.com) and Google's Orkut (www.orkut.com) actively promoting and facilitating the development of online communities. Second, by monitoring user selections for a query it is possible to build a model of query-page relevance based on the probability that a given page, p_j, will be selected by a user when returned as a result for query, q_i. The collaborative search approach combines these ideas in the form of a meta-search engine that analyses the patterns of queries, results and user selections from a given search interface serving a specific community of users. I-SPY's key innovation involves the capture of search histories and their use in ranking metrics that reflect user behaviour. This approach has been implemented in the I-SPY search engine and has been fully described previously; see [18]. However, for completeness, in this section we will review the implementation of collaborative search.

The I-SPY collaborative search architecture is presented in Figure 1. Briefly, the user query, q, is adapted and then submitted to base-level search engines (S_1 - S_n) . Similarly, the result-set, R_i, returned by a particular S_i is adapted, combined and re-ranked by I-SPY to produce R'_i, just like a traditional meta-search engine.

A unique feature of I-SPY is its ability to personalize its search results for a particular community of users without relying on content-analysis techniques (e.g., [2, 12]). I-SPY achieves this by maintaining a *hit-matrix* that records past search histories (see Figure 1). Each element of the hit-matrix, H, contains a value, v_{ij} (that is, $H_{ij} = v_{ij}$), to indicate that v_{ij} users have found page, p_j, relevant for query, q_i. Each time a user selects a page, p_j, for a query, q_i, I-SPY updates the hit-matrix accordingly.

I-SPY exploits the hit-matrix as a *direct* source of relevancy information; after all, its entries reflect relevancy judgments by users with

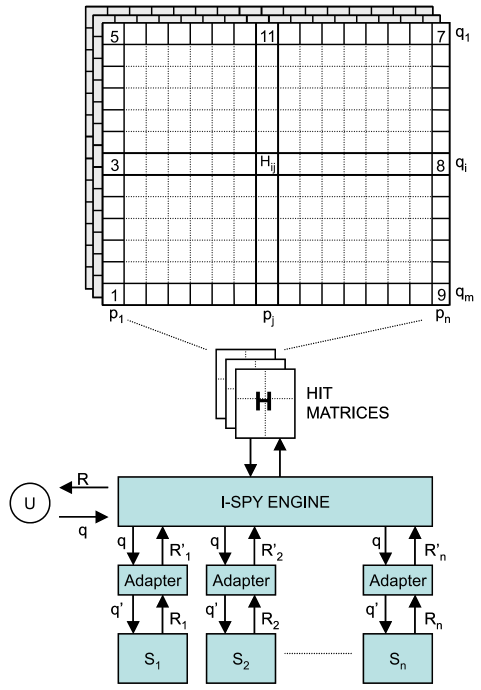

Figure 1. The I-SPY system architecture combines meta-search with a facility for storing the search histories of individual communities of searchers.

respect to query-page mappings. While, it is acknowledged that the relevance metric is only a measure of apparent relevance rather than an explicit judgement, it has been shown that implicit ratings, similar to our relevance metric, have a strong correlation with explicit interest in a result [6]. The *relevance* of a page, p_j, to a query, q_i, is estimated by the probability that p_j will be selected for q_i (see Equation 1). I-SPY then uses this relevancy information to rank the result pages for a given query. Pages that are represented in the hit-matrix for the current query are ranked ahead of all other pages retrieved from the base-level search engines and are listed in descending order of their relevance values. The remaining pages are ordered according to a standard meta-search engine ranking metric.

$$Relevance(p_j, q_i) = \frac{H_{ij}}{\sum_{\forall j} H_{ij}} \tag{1}$$

A key point to understand about I-SPY is that its relevancy metric is tuned to the preferences of a particular set of users - the community of I-SPY users using a particular version of I-SPY in a well-defined context - and the queries and pages that they tend to prefer. Deploy I-SPY on a wildlife Web site and its hit-matrix will be populated with query terms and selected pages that are relevant to wildlife fans. Over time pages relating to wildlife will tend to be prioritised because previously users have tended to select these wildlife sites. Other sites, relating to different topics, may still be returned but will be relegated to the bottom of the result-list. Studies to date have shown that this can result in significant performance benefits [8, 17].

4 COLLABORATIVE QUERY RECOMMENDATION

In this paper we are primarily interested in query recommendation as a further means of helping users to locate their target information.

Consider the following common search scenario: a user has entered a query and received a result-list. They select one result that they believe to be the most relevant but this result does not contain their required information; perhaps it is broadly relevant but is missing the specific target information that the user is looking for. At this point the user has two choices: either they move on to the next relevant result in the original result-list or they try a new query in the hope that it will produce a better result-list. As we have mentioned earlier query recommendation and elaboration techniques attempt to assist the searcher by suggesting alternative queries, or expansions of their current query, that they might try at this stage.

4.1 Generating Candidate Queries

Our collaborative query recommendation approach relies on the hit-matrix data collected during collaborative search to recommend new queries. However, it differs from other query recommendation techniques that rely on query logs and past search histories in a number of important respects. Firstly, the query recommendations are triggered and informed by the user selecting a result page; we call this the *trigger page*. Secondly, instead of looking for term overlaps between the current query and past queries, we focus on past queries that also led to this result being selected by searchers; these so-called *candidate queries* (see Equation 2) may or may not contain terms in common with the original query.

$$CandidateQueries(p_j) = \{q_i : Hij > 0\} \tag{2}$$

For example, let's consider a student with a term paper to complete on wildcats who searches using a version of I-SPY that is deployed on a wildlife Web site. The student submits the query 'jaguar competitors'; they are looking for information on animals that the jaguar competes with for survival. We have already seen how I-SPY could help this student by disambiguating vague queries, such as 'jaguar competitors', so that wildlife pages are prioritised above other pages about the car or Apple's operating system. Let us suppose that the student selects the result 'www.wildlife.com/jaguar', but that this page does not provide detailed information about the Jaguar's natural competitors. Let us further suppose that past queries for which this page was selected include 'jaguar', 'jaguar cats', 'habitat jaguar', and 'jaguar enemy'. All of these queries can be recommended on the basis that they too have led users to select the trigger page, 'www.wildlife.com/jaguar'. These are all candidate queries that could be recommended to the user.

The problem now is how to judge which of these queries is likely to be the best one for the user. Given the original 'jaguar competitors' query it appears that the last two candidates are more likely to yield a satisfactory result than the first two; the last query clearly refers to the natural competitors of the jaguar where as the first two are likely to lead to more general information about the jaguar. This leads us to the third notable feature of our query recommendation approach, candidate queries are ranked to prioritise those queries that are more likely to yield result lists that contain the desired target page.

4.2 Ranking Candidate Queries

In an earlier section we saw how collaborative search employs a usage-based ranking function to calculate the relevance of a page, p_j, for a query, q_i, (see Equation 1). In our query recommendation technique we use the same metric, but this time we use it to rank the candidate queries that have been recommended based on a given page selection. This will prioritise those queries that, in the past, have resulted in frequent selections of the current trigger page. Our intuition is that the user has selected this trigger page because it appears to be relevant to their needs and that, as such, other queries for which this page has a high relevance score are, all other things being equal, likely to serve as useful recommendations.

Of course all other things are not equal and, as a result, relevance on its own is unlikely to produce an effective ranking of candidate queries. For a start, while the user has selected the trigger because it appears to be relevant, the fact is, it does not completely fulfill their needs. At best, this page is actually only partially relevant to the user - remember our assumption is that they are not satisfied at this point - so queries that result in very high relevance scores for this page may not accurately capture our user's precise requirements either. For instance, the trigger page might be a very general page leading to candidate queries that are very broad in scope; the general page 'www.wildlife.com/jaguar' is likely to have high relevance scores for general queries such as 'jaguar' and 'jaguar cats' but these pages are not likely to be the ideal candidates for a user who is interested in more specialised pages about the natural competition faced by the jaguar in the wild.

Given that the user is not satisfied with the trigger page it is also very important to ensure that the result selections for the candidate queries are not dominated by the trigger page. If the trigger page has a very high relevance for some candidate query, q_c, then it is likely that very few other results have ever been selected for q_c. Obviously this reduces the chances that a satisfactory page will be one of the few other pages that have been selected for q_c. In the extreme, when the relevance of the trigger page for q_c is 1, then q_c has nothing to offer as a query recommendation because none of its other results have ever been considered to be relevant; they have never been selected.

Thus, we need to recommend queries that have many relevant results because this will clearly increase the likelihood that their result-list will contain the target information. This idea is captured by the coverage metric (see Equation 3). For a given query, q_i, the coverage computes the number of results selected for q_i in proportion to the total number of pages selected in Q, where Q is the set of all candidate queries for a page, p_j (see Equation 2). If a query has a high coverage then many of its results have been deemed relevant (selected) in the past. If it has a low coverage then its result list contains very few relevant results.

$$Coverage(q_i, Q) = \frac{|\{H_{ki} \forall k \colon H_{ki} \neq \emptyset\}|}{|\cup_{\forall ql \in Q}\, p_k \forall k \colon H_{kl} \neq \emptyset|} \tag{3}$$

Ideally we would like to recommend queries that have a high relevance (to the trigger page) and that offer a high degree of coverage. Thus our proposed ranking function combines the relevance and the coverage of a query to produce a single ranking score, and in the next section we will evaluate a number of possible combination methods.

5 EVALUATION

Ultimately the success of our query recommendation technique will rely on its ability to recommend queries that are likely to lead to successful results, and on its ability to prioritise these queries accordingly. We will test this ability using search data collected from 92 computer science students from the Department of Computer Science at University College Dublin. This current evaluation is part of an experiment that was designed to evaluate the benefits of I-SPY in the context of a fact-finding or question-answering exercise. To frame the search task we developed a set of 25 general knowledge

AI and computer science related questions, each requiring the student to find out a particular fact (time, place, person's name, system name etc.). The resulting search logs provided information about the queries submitted and the results selected; in total the 92 users generated 2673 queries and made 2751 page selections.

5.1 Methodology

We manually assessed all pages in terms of whether they did or did not contain the answer to a particular question. With the logs produced from the aforementioned evaluation, and using the hit-matrix data that I-SPY gathered during the experiment, each time a user selected a page that did not contain the answer they were looking for, we generated a set of candidate queries. Next, we scored and ranked each set of candidate queries using five separate ranking functions as follows:

1. *RelevanceOnly* - candidate queries are sorted in decreasing order of their relevance to the trigger page.
2. *CoverageOnly* - candidate queries are sorted in decreasing order of their coverage.
3. *Product* - candidates are sorted in decreasing order of their relevance value multiplied by their coverage value.
4. *ArithMean* - candidates are sorted in decreasing order of the arithmetic mean of their relevance and coverage values.
5. *HarmMean* - candidates are sorted in decreasing order of the harmonic mean of their relevance and coverage values.

The RelevanceOnly and CoverageOnly serve as benchmark metrics and are unlikely to serve as suitable ranking metrics because either factor on its own does not provide enough information to judge query quality. The Product and ArithMean provide more promising alternatives by combining relevance and coverage in standard ways, although they are biased towards the minimum and maximum factor for a given query. For instance, a query with a relevance of 0.8 and a coverage of 0.2, will obtain a ranking score of 0.16 according to the Product metric but a much higher score of 0.5 according to the ArithMean metric. In contrast, the HarmMean will penalise queries that differ greatly in their individual relevance and coverage score, so that in this example the HarmMean metric would produce a score of 0.32.

Finally, we estimated the success of each query as the proportion of times that it results in the selection of a page that is known to contain the correct answer. The point to remember here is that just because a page is returned for a query, does not mean it will actually be selected by the user. However, the success metric, is as close to an explicit judgement as possible, as we know whether or not a result contained the correct answer. The further down the result-list that a relevant result is positioned, the chances of it being selected fall away rapidly. High quality queries should include correct result pages at the top of their result lists, and these pages will have a better selection likelihood, and thus a better success estimate. Hence one of the central issues to be investigated is the correlation between each ranking metric and the success likelihoods. Ideally, there should be a strong correlation; high query ranking scores should be associated with high success likelihoods and vice versa.

5.2 Results

The results are presented for each of the 5 metrics in Figure 2 as plots of the average success observed for queries with various ranking scores. We have divided the scoring range into low (< 0.3), medium

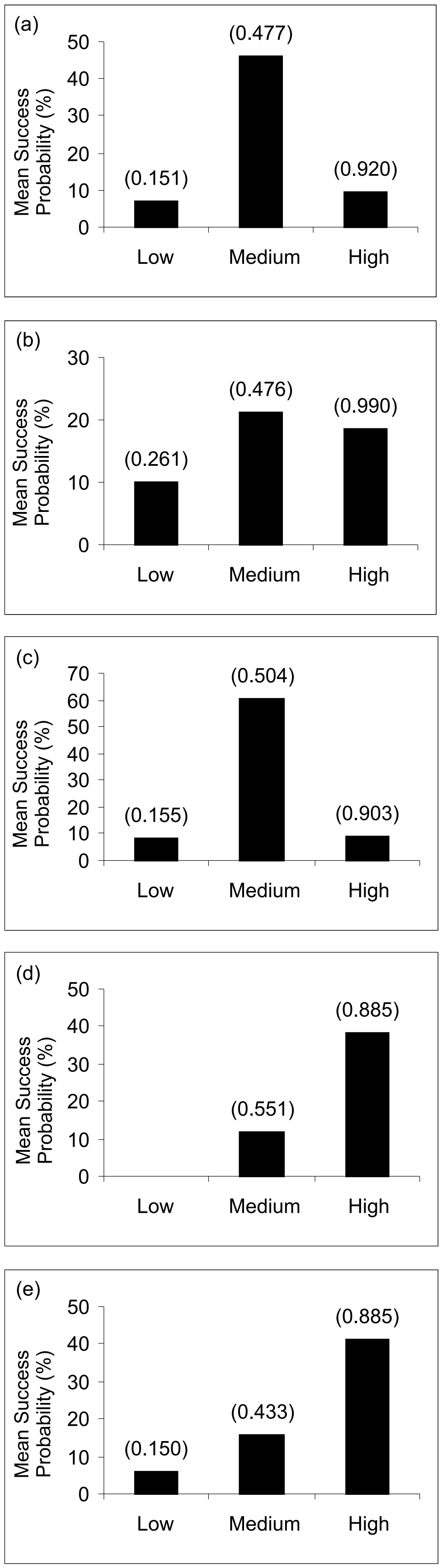

Figure 2. Mean Query Success results: (a) Relevance (b) Coverage (c) Product (d) Arithmetic Mean (e) Harmonic Mean

(0.3 to 0.7) and high (> 0.7) categories, these categories were chosen based on the dispersion of the data points. The mean scoring range for each category is also indicated above each bar. Overall the harmonic mean (HarmMean) and arithmetic mean (ArithMean) metrics perform best. Each shows a strong positive correlation between the ranking score and the probability that a query with this score will be successful. For example, the HarmMean metric results (Figure 2(e)) indicate that queries with a low ranking score (< 0.3) have only a 6% probability that their result lists will contain the correct page. For queries with a medium ranking score this success probability increases to 16%. And high ranking queries (> 0.7) have a 41% chance of retrieving the correct result page. A similar pattern is seen for the ArithMean metric although it is worth noting that none of the queries that received a low ranking for this metric turned out to be useful. In the case of the RelevanceOnly, CoverageOnly and Product metrics we find a markedly different pattern. In each case queries with a moderate ranking score are more likely to be successful than those with a high ranking score. This is clearly not a desirable result, as it would lead to the prioritisation of query recommendations that were less likely to be useful.

Table 1. Overall Correlation Values of Metrics and Success Values

Metric	Correlation
HarmMean	0.993
ArithMean	0.937
Product	-0.024
CoverageOnly	0.555
RelevanceOnly	-0.028

The actual correlations between the ranking scores and the success scores are presented in Table 1. The HarmMean produces the best correlation (0.993). The ArithMean correlation preformed slightly worse (0.937). The RelevanceOnly and Product both produced a marginally negative correlation (about 0.02), suggesting that these metrics are not suited for query recommendation.

6 CONCLUSIONS

The proposed query recommendation technique uses information about a user's result selections, and the selections of past users, in order to select, score, and recommend new queries. We argue the need for two important scoring factors, relevance and coverage, and show how they can be combined as ranking metrics. Our evaluation results confirm that, on its own, relevance does not adequately capture the circumstances under which a query is likely to prove to be useful - but that by combining relevance and coverage together, reliable query recommendation is possible. In particular, combining relevance and coverage using a harmonic mean metric is seen to produce the best ranking results because this metric is not biased by large individual coverage or relevance values. This work focuses on recommending queries for a particular result, similar work reverses this idea by recommending different results for similar queries [1]. Future work will build upon the current evaluation and we hope to avail of larger query logs as the basis for a more robust test of collaborative query recommendation. This search task was performed in a controlled environment, a larger scale experiment would allow us to confirm our belief that our algorithm would benefit users that perform related searches. In addition, we will consider additional factors that may also have a role to play during query recommendation.

ACKNOWLEDGEMENTS

The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged.

REFERENCES

[1] E. Balfe and B. Smyth, 'Case-Based Collaborative Web Search', in *Proceedings of the 7th European Conference on Case-Based Reasoning (ECCBR 2004)*. Springer, (2004).

[2] K. Bradley, R. Rafter, and B. Smyth, 'Case-based User Profiling for Content Personalization', in *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, eds., O. Stock P. Brusilovsky and C. Strapparava, pp. 62–72. Springer-Verlag, (2000).

[3] S. Brin and L. Page, 'The Anatomy of A Large-Scale Web Search Engine', in *Proceedings of the 7th International World-Wide Web Conference*, volume 30, pp. 107–117. Networks and ISDN Systems, (1998).

[4] C. Buckley, G. Salton, J. Allan, and A. Singhal, 'Automatic Query Expansion Using SMART: TREC 3', in *Proceedings of the 3rd Text Retrieval Conference (TREC-3). NIST Special Publication 500-225*, (1995).

[5] J. Budzik and K. Hammond, 'User Interactions with Everyday Applications as Context for Just-In-Time Information Access.', in *Proceedings of the 5th International Conference on Intelligent User Interfaces*, pp. 44–51. ACM Press, (2000).

[6] M. Claypool, P. Le, M. Waseda, and D. Brown, 'Implicit Interest Indicators', in *Proceedings of ACM Intelligent User Interfaces Conference (IUI)*, pp. 33–40. ACM, (2001).

[7] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma, 'Probabilistic Query Expansion Using Query Logs', in *Proceedings of the 11th International Conference on World Wide Web*, pp. 325–332, (2002).

[8] J. Freyne, B. Smyth, M. Coyle, E. Balfe, and P. Briggs, 'Further Experiments on Collaborative Ranking in Community-Based Web Search', *AI Review: An International Science and Engineering Journal*, (In Press).

[9] E. Glover, S. Lawrence, M. D. Gordon, W. P. Birmingham, and C. Lee Giles, 'Web Search - Your Way', *Communications of the ACM*, **44(12)**, 97–102, (2001).

[10] J. M. Kleinberg, 'Authoritative Sources in a Hyperlinked Environment', in *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 668–677. AAAI Press, (1998).

[11] S. Lawrence, 'Context in Web Search', *IEEE Data Engineering Bulletin*, **23(3)**, 25–32, (2000).

[12] S. Lawrence and C. Lee Giles, 'Context and Page Analysis for Improved Web Search', *IEEE Internet Computing 2*, **4**, 38–46, (1998).

[13] H. Lieberman, 'Letizia: An Agent That Assists Web Browsing', in *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI'95*, ed., C. Mellish, pp. 924–929. Morgan Kaufman Publishers, (1995). Montreal, Canada.

[14] M. Mitra, A. Singhal, and C. Buckley, 'Improving Automatic Query Expansion', in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 206–214. ACM Press, (1998).

[15] J.J. Rocchio, 'Relevance Feedback in Information Retrieval', in *The SMART Retrieval System - Experiments in Automatic Document Processing*, ed., Gerard Salton, pp. 313–323. Prentice Hall, (1971).

[16] G. Salton and C. Buckley, 'Improving Retrieval Performance by Relevance Feedback', *Journal of the American Society for Information Science*, **41**, 288–297, (1990).

[17] B. Smyth, E. Balfe, P. Briggs, M. Coyle, and J. Freyne, 'Collaborative Web Search', in *Proceedings of the 18th International Joint Conference on Artificial Intelligence, IJCAI-03*, pp. 1417–1419. Morgan Kaufmann, (2003). Acapulco, Mexico.

[18] B. Smyth, J. Freyne, M. Coyle, P. Briggs, and E. Balfe, 'I-SPY: Anonymous, Community-Based Personalization by Collaborative Web Search', in *Proceedings of the 23rd SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pp. 367–380. Springer, (2003). Cambridge, UK.

[19] J.-R. Wen, J.-Y., and H.-J. Zhang, 'Query Clustering Using User Logs', *ACM Trans. Inf. Syst.*, **20**(1), 59–81, (2002).