Collaborative Querying Through A Hybrid Query Clustering Approach

Lin Fu, Dion Hoe-Lian Goh, Schubert Shou-Boon Foo, Jin-Cheon Na
Division of Information Studies
School of Communication and Information
Nanyang Technological University
Singapore 637718

Abstract. Harnessing previously issued queries to facilitate collaborative querying is an approach that can help users in digital libraries and other information systems better meet their information needs. Here, the kernel step is to identify and cluster similar queries by mining the query logs. However because of the short lengths of queries, it is relatively difficult to cluster queries effectively using only the terms since they cannot convey enough information. This paper introduces a hybrid method to cluster queries by utilizing both the query terms and the results returned to queries. Experiments show that this method outperforms existing query clustering techniques.

1 Introduction

People have now come to depend more on the Web or digital libraries (DLs) to search for information. However the amount of information and its growth is a double-edged sword because of the problem of information overload, exacerbated by the fact that not all content on the Web or DLs is relevant nor of acceptable quality to information seekers. Information overload has thus led to a situation where users are swamped with too much information, resulting in difficulty sifting through the material in search of relevant content.

The problem of information overload has been addressed from different perspectives. The study of information seeking behavior has revealed that interaction and collaboration with other people is an important part in the process of information seeking and use [7][8][17]. Given this idea, collaborative search aims to support collaboration among people when they search information on the Web or in DLs [5]. Work in collaborative search falls into several major categories including collaborative browsing, collaborative filtering and collaborative querying [14]. In particular, collaborative querying seeks to help users express their information needs properly in the form of a question to information professionals, or formulate an accurate query to a search engine by sharing expert knowledge or other users' search experiences with each other [14]. Query mining is one of the common techniques used to support collaborative querying. It allows users to make use of other users' search experiences or domain knowledge by analyzing the information stored in query logs (query analysis), grouping (query clustering) and extracting useful related information on a given query. The extracted information can then be used as recommendation items (used in query recommending systems) or sources for automatic query expansion. An example is given below.

Consider a user A that is interested in novel human computer interfaces for information exploration, and she wants to look for research papers related to this field. Due to her limited domain knowledge, she enters "Human Computer Interface" as the query to her preferred search engine and gets lists of results. However nothing in the top 50 results contains the desired information and she does not know how to modify her query. At the same time, another user B may know that the paper "Reading of electronic documents: the usability of linear, fisheye and overview+detail interface" is a helpful paper that reviews novel interfaces and knows that good search results can be obtained by using "overview detail interface" as the query. Note that B's search history is usually stored in the query logs. Different search engines have query logs in different formats although most contain similar information such as a session ID, address of user, submitted query, etc. Thus, by mining the query logs, clustering similar queries and then recommending them to users, there is an opportunity for the first user to take advantage of previous queries that someone else had entered and use the appropriate ones to meet her information need.

From this example, we can see that the query clustering is one of crucial steps in query mining and the challenge here is to identify the similarities between different queries stored in the query logs. The classical method in information retrieval area suggests a similarity calculation between queries according to query terms (content-based approach) [13]. However the precision of this approach will likely be low due to the short length of queries and the lack of the contextual information in which queries are used [20]. An alternative approach is to use the results (e.g. result URLs in Web search engines) to queries as the criteria to identify similar queries (results-based approach) [5][10]. However one result might cover different topics, and thus queries with different semantic meanings might lead to the same result.

In this paper, we explore query similarity measures of query clustering by using a hybrid content-based and results-based approach. The basic principles of this approach are: (1) if different queries share a number of identical keywords in query terms, then these queries are similar; and (2) if different queries share some number of identical query results, then these queries are similar. As discussed, queries are often too short to convey enough information to deduce their semantic meaning. Thus by using the results to queries, our proposed combined method overcomes the lack of contextual information in which the queries are used. However the results-based approach will be affected greatly by the fact that queries representing different information needs might lead to the same results since one result can contain information in several topics. Hence, in order to compensate for this drawback, query terms are considered as complementary information for similarity calculations. Although traditional content-based methods are not suitable for query clustering by themselves, query terms do provide some useful information for query clustering as demonstrated in [20]. Consequently, we assume that a combination of both methods will help us detect similar queries more effectively and thus generate better query clusters. Our experimental results indicate that this hybrid approach generates better clustering results than using content-based or results-based methods separately.

This work will benefit information retrieval systems and DLs in better meeting the information needs of users through collaborative querying. Specifically, the hybrid approach proposed in this paper can be used to improve the performance of the algorithms adopted by query recommending systems to identify high-quality query clusters given a submitted query. Note that cluster quality encompasses coverage, precision and recall, and this will be discussed in detail in Section 4.

The remainder of this paper is organized as follows. In Section 2, we review the literature related to this work. Next, we describe our approach to clustering queries and report experimental results that assess the effectiveness of this approach. Finally, we discuss the implications of our findings for collaborative querying systems and outline areas for further improvement.

2 Related Work

2.1 Collaborative Search

Collaborative search can be divided into three types according to the ways that users search for information: collaborative browsing, collaborative querying and collaborative filtering [14]. Collaborative browsing can be seen as an extension of Web browsing. Traditional Web browsing is characterized by distributed, isolated users with low interactions between them while collaborative browsing is performed by groups of users who have a mutual consciousness of the group presence and interact with each other during the browsing process [6]. In other words, collaborative browsing aims to offer document access to a group of users where they can communicate through synchronous communication tools [12]. Examples of collaborative browsing applications include “Let’s Browse” [6], a system for co-located collaborative browsing using user interests, and “WebEx” [19], a meeting system that allows distributed users to browse a Web pages.

Collaborative filtering is a technique for recommending items to a user based on similarities between the past behavior of the user and that of likeminded people [1]. It assumes that human preferences are correlated and thus if a group of likeminded users prefer an item, then the present user may also prefer it. Collaborative filtering is a beneficial tool in that it harnesses the community for knowledge sharing and is able to select high quality and relevant items from a large information stream [4]. Examples of collaborative filtering applications include Tapestry [4], a system that can filter information according to other users’ annotations; GroupLens [12], a recommender system using user ratings of documents read; and PHOAKS [18], a system that recommends items by using newsgroup messages.

Collaborative querying on the other hand, assists users in formulating queries to meet their information needs by utilizing other people’s expert knowledge or search experience. There are generally two approaches used. Online live reference services are one such approach, and it refers to a network of expertise, intermediation and resources placed at the disposal of someone seeking answers in an online environment [9]. An example is the Interactive Reference Service at the University of California at Irvine, which offers a video reference service that links librarians at the reference desk at the University’s Science Library and students working one-half mile away in a College of Medicine computer lab [16].

Although online live reference services attempt to build a virtual environment to facilitate communication and collaboration, the typical usage scenario involves many users depending only on several "smart librarians". This approach inherently has the limitation of overloading especially if too many users ask questions at the same time. In such cases, users may experience poor service such as long waiting times or answers that are inadequate.

An alternative approach is to mine the query logs of search engines and use these queries as resources for meeting a user's information needs. Historical query logs provide a wealth of information about past search experiences. This method thus tries to detect a user's "interests" through his/her submitted queries and locate similar queries (the query clusters) based on the similarities of the queries in the query logs [5]. The system can then either recommend the similar queries to users (query recommending systems) [5] or use them as expansion term candidates to the original query to augment the quality of the search results (query automatic expansion systems) [10]. Such an approach overcomes the limitation of human involvement and network overloading inherent in online live reference service. Further, the required steps can be performed automatically. Here, calculating the similarity between different queries and clustering them automatically are crucial steps. A clustering algorithm could provide a list of suggestions by offering, in response to a query *q*, the other member of the cluster containing *q*. There are some commercial search engines that give users the opportunity to rephrase their queries by suggesting alternate queries. These include Lycos and Google.

2.2 Query Clustering Approaches

Traditional information retrieval research suggests an approach to query clustering by comparing query term vectors (content-based approach). Various similarity functions are available including cosine-similarity, Jaccard-similarity, and Dice-similarity [14]. Using these functions have provided good results in document clustering due to the large number of terms contained in documents. However, the content-based method might not be appropriate for query clustering since most queries submitted to search engines are quite short [20]. A recent study on a billion-entry set of queries to AltaVista has shown that more than 85% queries contain less than three terms and the average length of queries is 2.35 [15]. Thus query terms can neither convey much information nor help to detect the semantics behind them since the same term might represent different semantic meanings, while on the other hand, different terms might refer to the same semantic meaning [10].

Another approach to clustering queries is to utilize a user's selections on the search result listings as the similarity measure [20]. This method analyzes the query session logs which contain the query terms and the corresponding documents users clicked on. It assumes that two queries are similar if they lead to the selection of a similar document. Users' feedback is employed as the contextual information to queries and has been demonstrated to be quite useful clustering. However the drawback is that it may be unreliable if users select too many irrelevant documents [20].

Raghavan and Sever [10] determine similarity between queries by calculating the overlap in documents returned by the queries. This is done by converting documents into term frequency vectors. However this method is time consuming to perform and is not suitable for online search systems [3]. Glance [5] thus uses the overlap of result URLs as the similarity measure instead of the document content, which is quite similar to our approach. However, Glance does not take the query terms themselves into consideration, which can provide useful information for query clustering [20].

3 Calculating Query Similarity

As discussed, our approach is a hybrid method based on the analysis of query terms and query results. There are two principles behind our approach of determining query similarity. That is, two queries are similar when (1) they contain one or more terms in common; or (2) they have results that contain one or more items in common.

3.1 Content-based Similarity Approach

We borrow concepts from information retrieval [13] and define a set of queries as $Q=\{Q_1, Q_2 \ldots Q_i, Q_j \ldots. Q_n\}$. A single query Q_j is converted to a term and weight vector shown in (1), where q_i is an index term of Q_j and w_{iQj} represents the weight of the i^{th} term in query Q_j. In order to compute the term weight, we define the term frequency, tf_{iQj}, as the number of occurrences of term i in query Q_j and the query frequency, qf_i, as the number of queries in a collection of n queries that contains the term i. Next, the inverted query frequency, iqf_i, is expressed as (3), in which n represents the total number of queries in the query collection. We then compute w_{iQj} based on (2):

$$Q_j = \{< q_1, w_{1Q_j} >;< q_2, w_{2Q_j} >;\ldots\ldots\ldots.< q_i, w_{iQ_j} >\} \tag{1}$$

$$w_{iQj} = tf_{iQj} * iqf_i \tag{2}$$

$$iqf_i = \log(\frac{n}{qf_i}) \tag{3}$$

Given D, we define C_{ij} as (4) which represents the common term vector of queries Q_i and Q_j. Here, q refers to the terms that belong to both Q_i and Q_j.

$$C_{ij} = \{q : q \in Q_i \cap Q_j) \tag{4}$$

Given these concepts, we now can provide one definition of query similarity:

Definition I: A query Q_i is similar to query Q_j if $N(C_{ij})>0$, where the $N(C_{ij})$ is the number of common terms in both queries.

A basic similarity measure based on query terms can be computed as follows:

$$Sim\_basic(Q_i, Q_j) = \frac{N(C_{ij})}{Max(N(Q_i), N(Q_j))} \tag{5}$$

where $N(Q_i)$ is the number of the keywords in a query Q_i.

Taking the term weights into consideration, we can use any one of the standard similarity measures [13]. Here, we only present the cosine-similarity measure since it is most frequently used in information retrieval:

$$Sim\_cosine(Q_i, Q_j) = \frac{\sum_{i=1}^{k} cw_{iQ_i} \times cw_{iQj}}{\sqrt{\sum_{i=1}^{k} cw_{iQ_i}^2} * \sqrt{\sum_{i=1}^{k} cw_{iQ_j}^2}} \quad (6)$$

where cw_{iQi} refers to the weight of i^{th} common term of C_{ij} in query Qi.

As discussed, the effectiveness of using content-based similarity approaches is questionable due to the short lengths most queries. For example the term "light" can be used in four different ways (noun, verb, adjective and adverb). In such cases, content-based query clustering cannot distinguish the semantic differences behind the terms due to the lack of contextual information and thus cannot provide reasonable cluster results. Thus an alternative approach based on query results is considered.

3.2 Result URLs-based Similarity Approach

The results returned by search engines usually contain a variety of information such as the title, the abstract, the category, etc. This information can be used to compare the similarity between queries. In our work, we consider the query results' unique identifiers (e.g. URLs) in determining the similarity between queries.

Let $U(Q_j)$ be represented as set of query result URLs to query Q_j:

$$U(Q_j) = \{u_i, u_2, \ldots\ldots\ldots\ldots u_i\} \quad (7)$$

where u_i represents the i^{th} result URL for query Q_j. We then define R_{ij} as (8) which represents the common query results URL vector between Q_i and Q_j. Here u refers to the URLs that belong to both $U(Q_i)$ and $U(Q_j)$.

$$R_{ij} = \{u : u \in U(Q_i) \cap U(Q_j)\} \quad (8)$$

Next, the similarity definition based on query result URLs can be stated as:

Definition II: A query Q_i is similar to query Q_j if $N(R_{ij})>0$, where the $N(R_{ij})$ is the number of common result URLs in both queries.

The similarity measure can then be expressed as (9)

$$Sim\_result(Q_i, Q_j) = \frac{N(R_{ij})}{Max(N(U(Q_i), N(U(Q_j))} \quad (9)$$

where the $N(U(Q_i))$ is the number of result URLs in $U(Q_i)$. Note that this is only one possible formula of calculating similarity using result URLs. Other measures for determining the similarity can be used. For example, overlaps of result titles or overlaps of the domain names in the result URLs.

3.3 Hybrid Approach

The content-based query clustering method can mine the relationship between different queries with the same or similar keywords. Nevertheless, a single query term without much contextual information can represent different information needs. The results-based method can find the relationship between different queries using the query results returned by a search engine. This method uses more contextual information for a given query that can be used for clustering. However, the same document in the search results listings might contain several topics, and thus queries with different semantic meanings might lead to the same search results.

On the other hand, while content-based methods are not suitable for query clustering by themselves, query terms have been shown to have the ability to provide useful information for clustering [20]. Therefore, we believe that the content-based approach can augment the results-based approaches and compensate for the ambiguity inherent in the latter. Hence, unlike [5], we assume that a combination of both methods will provide more effective clustering results than using each of them individually. Based on this assumption we define a hybrid similarity measure as (10):

$$Sim\_com(Q_i, Q_j) = \alpha * Sim\_result(Q_i, Q_j) + \beta * Sim\_cosine(Q_i, Q_j) \qquad (10)$$

where α and β are parameters assigned to each similarity measure, with α + β=1.

3.4 Determining Query Clusters

Given a set of queries Q={Q_1, Q_2….. Qn} and a similarity measure between queries, we next construct query clusters. Two queries are in one cluster whenever their similarity is above a certain threshold. We construct a query cluster G for each query in the query set using the definition in (11). Here Sim(Q_i, Q_j) refers to the similarity between Qi and Q_j which can be computed by using various similarity functions discussed previously.

$$G(Q_i) = \{Q : Sim(Q_i, Q_j) \geq threshold\} \qquad (11)$$

where 1 < j < n; n is the total query number.

4 Query Clustering Experiments

This section provides empirical evidence to demonstrate the viability of the hybrid query clustering method proposed in this paper. The metrics used to determine quality are coverage, precision, recall.

4.1 Data Source and Data Preprocessing

We collected six-month user logs (around two millions queries) from the Nanyang Technological University (Singapore) digital library. The query logs were in text format and were preprocessed (see Table 1 for examples) as follows:

1. The submitted queries were extracted, ignoring other information in the logs.
2. Queries that contained misspellings were removed since such queries will not lead to any results.
3. Due of the large number of queries, 10000 were randomly selected for our experiments.

4. Options embedded in the queries were removed. Examples include options for searching by author and title only.

5. Stop words were removed since they do not convey useful information.

Table 1. Query Samples

cards game	fabrication of CMOS	mobile phone works
communications between people	handbook chemical engineering	NiTi matrix composites
desalination plant costs	intelligence and gene	packaging machinery

Within the 10000 queries selected, 23% of the queries contained one term, 36% contained two terms, and 18% of the queries contained three terms. Approximately 77% of the queries contained no more than three terms. The average length of the queries was 2.73 terms. This observation is similar to previous studies on query characterization in search engines [15].

4.2 Method

We calculated the similarity between queries using the following similarity measures:

- Content-based similarity (sim_cosine) – function (6)
- Results-based similarity (sim_result) – function (9)
- Hybrid similarity (sim_com) – function (10)

Computation for sim_cosine was straightforward using function (6). For sim_result, we posted each query to a reference search engine (Google) and retrieved the corresponding result URLs. By design, search engines rank highly relevant results higher, and therefore, we only considered the top 10 result URLs returned to each query. The result URLs are then be used to compute the similarity between queries according to (9). Note that this approach is similar with [5].

For the hybrid approach (10), the issue was to determine the values for the parameters α and β. Three pairs of values were used, each representing varying levels of contribution a particular method (content-based or results-based) had in determining query clusters. These were: sim_com1 (α =0.25 β=0.75), sim_com2 (α=0.5 β=0.5), and sim_com3 (α=0.75 β=0.25).

In all measures, clustering thresholds (11) were simply set to 0.5. After obtaining the clusters based on the different similarity measures, we first observed the average cluster size and the range of the cluster sizes. This information sheds light on the ability of the different measures to provide recommended queries on a given query. In other words, they can reflect the variety of the queries to a user.

Next, coverage, precision and recall were calculated. Coverage is the ability of the different similarity measures to find similar queries for a given query. It is the percentage of queries for which the similarity function is able to provide a cluster. This value will indicate the probability that the user can obtain recommended queries for his/her issued query.

Precision and recall are used to assess the accuracy of the query clusters generated by different similarity functions. Here, the standard definitions are used:

(1) Precision: the ratio of the number of similar queries to the total number of queries in a cluster.

(2) Recall: the ratio of the number of similar queries to the total number of all similar queries for the query set (those in the current cluster and others).

For precision, we randomly selected 100 clusters and checked each query in the cluster manually. Since the actual information needs represented by the queries are not known, the similarity between queries within a cluster was judged by a human evaluator by taking into account the query terms as well as result URLs. The average precision was then computed for the 100 selected clusters.

Recall posed a problem as it was difficult to calculate directly because no standard clusters were available in the query set. Therefore, an alternative measure to reflect recall was used. Recall was defined to be the ratio of the number of correctly clustered queries within the 100 selected clusters to the maximum number of the correctly clustered queries across the test collection [20]. In our work, the maximum number of the clustered queries was 577, which was obtained by sim_com1.

4.3 Results and Discussion

The three similarity functions led to different cluster characteristics as summarized in Table 2. The average cluster sizes of sim_cosine, sim_result, sim_com1, sim_com2 and sim_com3 were 7.86, 4.01, 6.62, 4.10 and 3.96 respectively. The maximum query cluster sizes are 59, 23, 31, 26 and 23 respectively. This indicates that for a query cluster, the content-based approach (sim_cosine) can find a larger number of queries for a given query than the other approaches. Stated differently, the content-based approach can provide a greater variety of queries to a user given his/her submitted query. Here, note that the hybrid approach sim_com1, ranks second in terms of both the average cluster size and the range of cluster sizes.

Table 2. Cluster Characteristics

Similarity Measure	**Sim_ cosine**	**Sim_ result**	**Sim_com1** (α =0.25 β=0.75)	**Sim_com2** (α =0.5 β=0.5)	**Sim_com3** (α =0.75 β=0.25)
Average Cluster Size	7.86	4.01	6.62	4.10	3.96
Range of Cluster Sizes	2~59	2~23	2~31	2~26	2~23
Coverage	77.63%	43.93%	65.69%	48.19%	43.97%
Precision	66.72%	92.71%	87.13%	96.97%	99.00%

Recall	90.92%	64.45%	100%	68.93%	67.97%

Further, the results show that sim_cosine ranks highest in coverage (77.63%), demonstrating that the content-based approach has a better ability to find similar queries from a given query. In other words, users have a higher likelihood to obtain a recommendation to a given query than using other approaches. Note again that the hybrid approach, sim-com1, ranks second (65.69%) while the results-based approach, sim_result, ranks last (43.93%).

Table 2 also indicates that the hybrid approach is better able to cluster similar queries correctly than the other approaches. In terms of precision, sim-com3 (99.00%) performs best, indicating that almost all of the queries in the cluster were considered similar. Close behind are sim_com2 (96.97%), sim_result (92.71%) and sim_com1 (87.13%), while sim_cosine's precision is the worst (66.72%). The good precision performance of the hybrid approach can be attributed to the boost given by the search results URLs since sim_result's precision is much better when compared with sim_cosine. For recall, sim_com1 has the best performance at 100%, indicating that all similar queries were contained in query clusters. This time, sim_cosine (90.92%), sim_com2 (68.93%), sim_com3 (67.97%) rank second to fourth respectively, while sim_result ranks last (64.45%). Note that although the recall used in this experiment is not the same with the traditional definition used in information retrieval research, it does provide useful information to indicate the accuracy of clusters generated by the different similarity functions [20]. That is, the modified recall measure reflects the ability to uncover clusters of similar queries generated by different similarity functions on the sample set queries used in the experiments.

Among the hybrid approaches, the results indicate that sim_com1 is the most promising approach to use because when compared with sim_com2 and sim_com3, its precision and recall performance are only slightly lower than the latter two (87.13% versus 96.97% and 99% respectively) while its coverage performs considerably better than the others (65.69% versus 48.19% and 43.97% respectively). Conversely, although sim_com2 and sim_com3 perform well in terms of precision and recall, they suffer from poor coverage, meaning that if these approaches were used in query recommending systems, users will not be able to get query recommendations in most cases.

Thus taking all three metrics into consideration (coverage, precision and recall), we conclude that sim_com1 provides the best mix of results among the hybrid approaches. Further, when comparing sim_com1 with sim_result and sim_cosine, sim_com1 exhibits the best recall (100%) while ranking second for coverage and precision (65.69% and 87.13% respectively). Hence while sim_cosine and sim_result perform badly either because of the low accuracy of the query clusters (66.72% precision for sim_cosine) or the poor ability to find similar queries from a given query (43.93% coverage for sim_result), sim_com1 provides a balanced set of performance results that suggest its ability to provide query clusters of good overall quality.

Table 3 shows example clusters for the submitted query "computer network" based on sim_com1, with the threshold at 0.5. This example illustrates the usefulness of recommending related queries in that users are given opportunities to explore semantically related areas not explicitly covered in the original query. In the example, queries such as "wireless LAN" and "internet" are suggested as alternate queries that might help meet the current information need.

Table 3. Sample Clusters

computer network, computer networking,
network programming, networks, wireless LAN, internet

In summary, our experiments show that users will have a higher chance of obtaining a recommendation using sim_cosine, although the accuracy of the recommended queries will be poor. The sim_result approach improves average precision significantly but suffers from poor coverage and recall. Finally, the hybrid approach, in particular, sim_com1 with α =0.25 and β=0.75, outperforms the content-based or results-based approaches since it can provide a balanced set of results in terms of coverage, precision and recall. In other words, the query clusters sim_com1 exhibit good overall quality. Thus, users can benefit from query recommending systems adopting sim_com1 in most cases since they will have a higher chance of obtaining high quality recommendations of related queries.

5 Conclusions and Future Work

In this paper, we compare different query similarity measures. Our experiments show that by using a hybrid content-based and results-based approach, considering both query terms and query result URLs, more balanced query clusters can be generated than using either of them alone.

Our work can contribute to research in collaborative querying systems that mine query logs to harness the domain knowledge and search experiences of other information seekers found in them. The experiment results reported here can be used to develop new systems or further refine existing systems that determine and cluster similar queries in query logs, and augment the information seeking process by recommending related queries to users.

In addition to the initial experiments performed in this research, experiments involving threshold values are also planned. Since the threshold values affect query clustering results, these experiments will determine extent of such effects. Further, alternative approaches to identifying the similarity between queries will also be attempted. For example, the result URLs can be replaced by the domain names of the URLs to improve the coverage of the results-based query clustering approach. In addition, word relationships like hypernyms can be used to replace query terms before computing the similarity between queries. Finally, experiments using other clustering algorithms such as DBSCAN [2] might also be conducted to assess clustering quality.

References

[1] Chun, I. G., & Hong, I. S. (2001). The implementation of knowledge-based recommender system for electronic commerce using Java expert system library. *Proceedings of IEEE International Symposium on Industrial Electronics*, 1766-1770.

[2] Ester, M., Kriegel, H., Sander, J., Xu, X., (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of second International Conference on Knowledge Discovery and Data Mining*, 226-231.

[3] Fitzpatrick, L. & Dent, M. (1997). Automatic feedback using past queries: Social searching? *Proceedings of SIGIR'97*, 306-313.

[4] Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of ACM, 35*(12), 61-70.

[5] Glance, N. S. (2001). Community search assistant. *Proceedings of Sixth ACM International Conference on Intelligent User Interfaces*, 91-96.

[6] Lieberman, H. (1995). An agent for web browsing. *Proceedings of International Joint conference on Artificial Intelligence*, 924-929.

[7] Lokman, I. M., & Stephanie, W. H. (2001) Information–seeking behavior and use of social science faculty studying stateless nations: A case study. *Journal of library and Information Science Research, 23*(1), 5-25.

[8] Marchionini, G. N. (1995). *Information seeking in electronic environments.* Cambridge, England: Cambridge University Press.

[9] Pomerantz, J., & Lankes, R.D. (2002). Integrating expertise into the NSDL: Putting a human face on the digital library. *Proceedings of the Second Joint Conference on Digital Libraries*, 405.

[10] Raghavan, V. V., & Sever, H. (1995). On the reuse of past optimal queries. *Proceedings of the Eighteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*, 344-350.

[11] Resnick, P., Iacovou, N., Mitesh, S., Bergstron, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of Netnews. *Proceedings of the 1994 ACM Conference on CSCW*, 175-186.

[12] Revera, G.D.J.H., Courtiat, J., & Villemur, T. (2001). A design framework for collaborative browsing. *Proceedings of Tenth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 362-374.

[13] Salton, G. & Mcgill, M.J. (1983). *Introduction to Modern Information retrieval.* McGraw-Hill New York, NY.

[14] Setten, M.V & Hadidiy, F.M. Collaborative Search and Retrieval: Finding Information Together. Available at: https://doc.telin.nl/dscgi/ds.py/Get/File-8269/GigaCE-Collaborative_Search_and_Retrieval__Finding_Information_Together.pdf

[15] Silverstein, C., Henzinger, M., Marais, H., & Moricz, M. (1998) Analysis of a very large Altavista query log. *DEC SRC Technical Note 1998-14.*

[16] Sloan, B. (1997, December 16). *Service perspectives for the digital library remote reference services.* Available at: http://www.lis.uiuc.edu/~b-sloan/e-ref.html

[17] Taylor, R. (1968). Question-negotiation and information seeking in libraries. *College and Research Libraries, 29*(3), 178-194.

[18] Teveen, L., Hill, W., Amento, B., David, M., & Creter, J. (1997). PHOAKS: A system for sharing recommendations. *Communications of the ACM, 40*(3), 59-62.

[19] WebEx home page. http://www.webex.com

[20] Wen, J.R., Nie, J.Y., & Zhang, H.J. (2002) Query clustering using user logs. *ACM Transactions on Information Systems, 20*(1), 59-81.