Experiments with Collaborative Index Enhancement

Erik Selberg   Oren Etzioni

Department of Computer Science and Engineering

University of Washington

Seattle, WA 98195

{selberg, etzioni}@cs.washington.edu

http://www.cs.washington.edu/homes/{selberg, etzioni}

http://huskysearch.cs.washington.edu

Abstract

In this paper we present a method for improving the quality of a searchable index: Collaborative Index Enhancement, or CIE. CIE improves the index quality by feeding the results of searches over an index back into that index, so that the current user can harness the effort and insight of previous users that searched for the same or similar information. We present four prototype enhanced indices implemented as part of the HuskySearch system, a World Wide Web (WWW) search service available publicly. We describe the implementation and tradeoffs of each system, and present experiments based on user studies and log analyses detailing their effectiveness.

1 Introduction

Although WWW information retrieval (IR) systems have proven to be useful to a worldwide audience, there are still several issues that need to be addressed. Current "universal" WWW IR systems that attempt to index every available WWW page, such as Excite[9] or AltaVista[5], use spiders to retrieve pages for their index[20]. Unfortunately, network resources restrict the rate spiders can retrieve pages. Thus universal WWW IR systems only have a subset of avail-able pages, and need to make tradeoffs between the number of pages in their index and how up to date those pages are. Meta-search services that collect results from multiple services, such as MetaCrawler[16], are able to help alleviate these problems. However there are still limits as to how much improvement meta-search can provide.

Another approach being explored is to use forms of collaboration to identify and disseminate useful information on the WWW. These systems generally work by having a user with some information need match his or her user profile to other users' profiles. Once matched, the system then extrapolates relevant information from the experiences or preferences of those users. Recent collaboration systems, such as the Do-I-Care system[1] or Ringo[28], showcase the usefulness of this approach. However, these systems are typically geared towards a small group or a narrow domain.

This paper explores a combined approach called Collaborative Index Enhancement, or CIE. CIE is defined as taking information passively collected from a user during a session with a searchable index and using that information to augment that searchable index in some manner. We present HuskySearch, an implementation of a WWW IR system that uses CIE. We showcase two general enhancement techniques, one that uses the results of a search as a document

in its own right, and another that uses search results as augmenting information to enhance relevancy ranking criteria. We also compare using all available documents for enhancement versus using a subset of user-selected documents. Finally, we present both log analyses and a pilot user study of HuskySearch to evaluate the benefits of using CIE.

As part of our experimental validation of CIE, we observed that users only made one query to find the answer to any one particular question over 50% of the time. We present the available data we collected on this phenomenon, and our conjectures as to why this may be.

The remainder of this paper is organized as follows: we describe the design and implementation of our CIE prototypes in Section 2. We present experiments measuring the improvements in Section 3. Future and related work are discussed in Sections 4 and 5, and Section 6 concludes.

2 CIE Implementation

The CIE prototype is implemented as part of Husky-Search, a WWW metasearch service similar to MetaCrawler[27] and SavvySearch[6]. HuskySearch takes a user query and forwards it to a number of WWW search services in parallel, reformatting the query as appropriate. It then collates the results into a single ranked relevancy list, with an option to switch to either a sort by URL listing or a clustered listing[33].

2.1 HuskySearch Overview

One of the primary constraints we have placed on HuskySearch is that its interface needs to be simple and kept similar to other search services, so that the learning curve is minimal for the average WWW user. The interface is comprised of a query entry box, a menu that depicts the query logic, a multiple selection box to enable or disable various indices, and three buttons that each execute the search using different timeout values: "5 second search," "30 second search," or "5 minute search." The longer the system has to search, the more documents it will be able to

retrieve.

It has been shown that unstructured queries tend to be more effective for average users[32], thus we use a rudimentary search syntax of space-separated keywords, using quotes or parentheses to denote adjacency. The syntax also uses a menu option that defines the logic behind word spacing, e.g. users can choose to have "foo bar" mean "foo AND bar," "foo OR bar," "foo ADJ bar", or "Mr. or Ms. Foo Bar." This syntax is designed for novice users that do not necessarily know or understand even Boolean searching.

HuskySearch uses an algorithm we call *Normalize-Distribute-Sum* (NDS) to collate documents. The relevance scores from each service are normalized to [0..1000]. The scores are then redistributed via the following formula:

$$s_i' = \frac{N - h_i + 1}{N} * s_i$$

where N is the number of documents returned by the service, h_i is the rank of document i, ranging from $[1..N]$ with 1 being the top rank, and s_i being the original relevance score of i given by the reference source. The redistribution is done to prevent services that return multiple "perfectly" relevant results — e.g. a lot of results that all score 1000 — from being listed before the results from other services. We then sum the redistributed scores from duplicate entries, and then re-normalize the scores to [0..1000] for end-user presentation, keeping with a style similar to other search sites. One important, and intended, consequence of this algorithm is that references returned by two or more sites tend to be ranked higher than references returned by only one.

If we believed that each ranking algorithm was accurate or we knew that all services used the same ranking algorithm, the distribution step would not be needed. In fact, other fusion algorithms, such as those used to merge TREC results[4], would likely perform much better than the NDS algorithm were this the case. We do not yet have either a formal or experimental justification for this algorithm, but it does work well in practice. Some comparison with other WWW fusion algorithms[13, 15] as well as ones used in the TREC environment is definitely warranted.
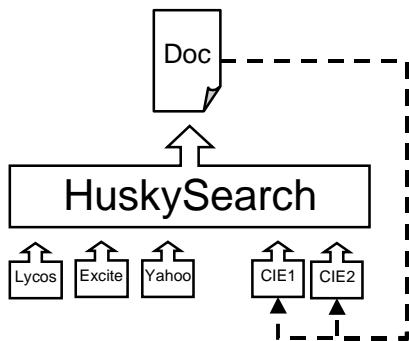
Figure 1: CIE Architecture. Each index feeds a list of documents into HuskySearch, which collates these documents into a single results document containing a list of references. This document is then fed back into the CIE databases.

2.2 CIE integration

A CIE prototype is incorporated into HuskySearch by creating a separate searchable index that contains the enhancements derived from previous queries. This new index is then added to HuskySearch. When a user makes a query, HuskySearch searches the CIE index in parallel with the rest, and the CIE results are then merged in with the other documents from traditional indices. Additional CIE systems can be integrated by adding a new index for each new CIE system. This is depicted graphically in Figure 1. By using this architecture, we are able to implement CIE without the modification or control of the original indices used for searching.

Four CIE indices are described in this paper:

ColList Each document in ColList (*Col*laborative *List*s) is the *results document* from a particular query; e.g. it is the HTML page containing URLs, titles, and snippets to other pages.

ColListSelect Each document in ColListSelect (*Col*laborative *List*s *Select*ed) is a results document where at least one of the results was clicked on.

ColRes ColRes (*Col*laborative *Res*ults) contains all the documents referred to by results documents

of previous queries; e.g. for all queries, the referenced documents are downloaded and inserted into this index.

ColResSelect ColResSelect (*Col*laborative *Res*ults *Select*ed) contains all the documents referred to by results document of previous queries that were followed; e.g. each document in ColResSelect was viewed by some user in the course of a search.

The ColList and ColListSelect indices are used to implement a form of convenient query expansion[8]. The snippets contained within a results document tend to highlight the relevant key terms for the referenced document. By matching a query to those key terms, users can be directed to a previous user's query with different results that might be more appropriate to the question at hand. Spink showed that the majority of terms users selected for refinement came from document title and descriptor fields[29]; we conjecture that users will also be able to make efficient use of terms other users used for their queries.

One important distinction between this form of query expansion and other methods is that there is no mode shift in the interface — users are never required to enter a "query refinement" stage, such as selecting relevant documents for feedback or by entering in new terms manually. Because each former results document is available in a ranked list with other potentially relevant documents, users are able to treat previous results documents as just another document with potentially relevant links. This integrated listing also provides the user with a metric as to how relevant a particular query refinement is compared with other actual documents, so that the user does not need to read through a preset number before determining that refinement is necessary.

The ColRes and ColResSelect indices are used in two fashions. The first is to give potentially relevant documents a second evaluation on the given query using a different retrieval method, a technique that has been shown to be productive using traditional IR benchmarks[2]. It is possible that after a million or more queries that ColRes and ColResSelect could become additional global WWW search indices like the other services HuskySearch queries, but the intention is to keep ColRes and ColResSelect small with docu-

3

ments that are likely to be relevant. The second aim of ColRes and ColResSelect is to improve the ranking of relevant documents. If both the original WWW service and the ColRes or ColResSelect index return the reference, then its ranking will be higher than if just one returned it. Thus, ColRes and ColResSelect simulate using popularity to help determine rank.

Each index has properties that affect how the Husky-Search fusion algorithm interacts with it. The ColListSelect collection is a strict subset of the ColList collection, and thus every reference returned from the ColListSelect collection will also be returned by the ColList collection. Thus, ColListSelect results will almost always be ranked above ColList results. The same holds for ColResSelect and ColRes. In addition, since ColRes and ColResSelect are drawn from the general WWW results returned, it is likely that the WWW search service that originally returned the result will also return it for the current query; thus ColResSelect results will tend to be ranked very highly.

One interesting phenomenon we observed while building this system is that the ColList index keeps an implicit query history for each user. Thus, if you are looking for a site you previously found via Husky-Search and have trouble remembering the query used, it's possible to search for the previous query explicitly.

The CIE indices used by HuskySearch use the Verity Search '97 search engine v2.0, running on a DEC AlphaStation under DEC UNIX 3.2. HuskySearch also uses Alta Vista[5], Excite[9], HotBot[17], Northern Light[21], PlanetSearch[23], WebCrawler[22], and Yahoo![10], as well as two local intranet search services, The Daily[31], the local student newspaper, and UWSearch, an index specific to HuskySearch using the Verity engine.

3 Experimental Validation

In this section, we present two sets of experiments. The first is based upon analysis of log entries from public use of HuskySearch. The second is based upon a pilot user study. We have designed our system for finding relevant documents on the WWW. Since there is not yet an available standard test corpus of WWW documents, we chose to conduct experiments using the WWW rather than existing static collections of non-WWW documents, such as the TREC collections[14].

3.1 Log Analysis

It was demonstrated with MetaCrawler that meta-search services are excellent tools to evaluate how well each underlying search service compares on a number of different metrics[26]. In a similar vein to the MetaCrawler analysis, we evaluate the four CIE indices in comparison to one another and the other WWW search services. The metric we use is to equate relevance with following a reference; this provides us with an approximation to true relevance. We thus approximate precision by dividing the number of followed references by the number of returned references. Because we do not have any data on the number of available relevant documents on the Web, we are not able to calculate recall. Instead, we use *precision at 20 documents*, which is a reasonable metric for a system designed for the average user[18].

While our approximation of precision at 20 documents is not a perfect metric, we observe from the logs that users do not frequently click on many results. For this study, there are 4310 queries across 6 days that returned a total of 752,219 results. 6,967 of these were followed, for an average rate of 1.61 followed results per query. Since this is a non-trivial number of queries and we receive less than one email a week regarding a user being unable to find any relevant documents for a particular query, we assume that most of the references they click on are relevant and lead them to the information they require.

Figure 2 shows the precision at 20 documents for the six day span from Jan. 17 through Jan. 22, 1998, which encompasses 4310 unique queries. As can be expected with 1.61 followed results per query, precision is extremely poor across the board. In comparison to the global WWW indices, ColRes performs exceptionally poorly, whereas ColResSelect performs a little bit better than most of the contemporary search services, with Yahoo! being the glaring exception. However, it is important to note that Yahoo! is the one service that does not use automatic spider
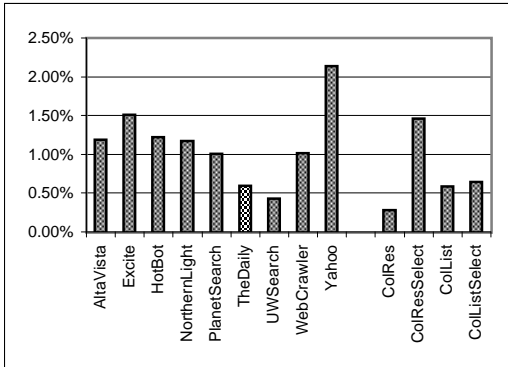
Figure 2: Precision at 20 documents of WWW search services and CIE systems, for a 6-day span covering 4310 queries.
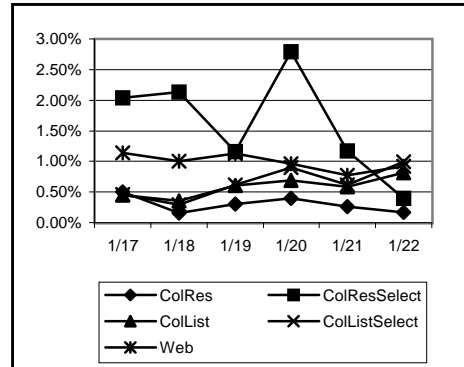

Figure 3: Daily precision at 20 of WWW search services and CIE systems. WWW services are averaged together for clarity.

selection to create the index. Rather each document in Yahoo!'s index has a human generated summary and is placed in appropriate categories. ColList and ColListSelect do not perform as well as we had hoped, although we conjecture that this may be due to the relative immaturity of both indexes and will improve over time.

We then calculate the precision at 20 documents on a day by day basis to determine if the CIE systems are improving with time. Figure 3 highlights our findings. The ColRes index is a chronic poor performer. For the six days listed, ColResSelect was about double the performance of the WWW average on 3 days, yet only slightly above and in one case below average on the other 3 days. This suggests that the ColResSelect is likely to be have a higher precision in general than the WWW average, but the high variation for the sample shown does indicate that more evaluation is necessary.

Both the ColList and ColListSelect indices start out extremely poorly, but after 3 days approach the WWW average, and in the case of ColListSelect, surpass it. This suggests that our conjecture regarding the early immaturity of these two indices may be valid, and further study should illuminate that point.

In summary, ColResSelect appears to be a useful index comparable with existing services, and ColRes is probably not worth much more evaluation. While the ColList and ColListSelect indices didn't perform

as well as expected, results suggest that using result documents as index documents is useful in some cases. Furthermore, both ColListSelect and ColResSelect indices had much better precision than ColList and ColRes respectively, which suggests that user selection can increase the performance of a CIE technique.

3.2 User Study

We conducted a user study to confirm that the results returned by the CIE indices are in fact useful to a group of searchers trying to answer five different questions. For this study, we are interested in answering the following questions:

- *Are users able to answer more questions correctly with CIE?*

- *Are users better able to determine they have the correct answer with CIE?*

- *Are users able to answer questions faster with CIE?*

The study was conducted by distributing instructions and forms to volunteers, who then followed the instructions on their own time. There were two distinct sets of instructions: one for a control group that did not use the CIE indices, and a CIE group which had

5

AltaVista	50.00%	Lycos	19.23%
Excite	26.92%	WebCrawler	23.08%
HotBot	19.23%	Yahoo	73.08%
InfoSeek	15.38%	Other	19.23%

Table 1: Percentage of users familiar with popular Internet search services. "Other" encompasses all services mentioned only once or twice.

	Kevin	Utah	Tree	Can	MS
Ctrl	100 (0)	87 (2)	80 (3)	40 (6)	7 (14)
CIE	100 (0)	80 (2)	90 (1)	60 (4)	30 (7)

Table 2: Percentage of users accurately answering each question. Numbers in parentheses indicate are number of users that were unable to answer the question.

full access to the four CIE indices. We received 25 responses, 15 from the control group and 10 from the CIE group.

The users for this study were volunteers from a second year graduate Library Science class, nearly all of whom have at least some experience with searching the Internet. A few did have prior experience with HuskySearch and MetaCrawler. This is detailed in Table 1.

The study consisted of the following five questions:

Kevin *What are the three most recent roles Kevin Spacey has played?*

Utah *Which Utah ski resort has the highest elevation, and what is it?*

Tree *Find a picture of a Fraser fir. Please give the URL.*

Can *How many members are there in the Canadian parliament?*

MS *What was Microsoft's IPO price?*

We attempted to order these questions from easiest to answer to hardest based on our own experience in finding the answers. For each question, users were asked to write down the answer, starting and stopping time accurate to the minute, their belief in the correctness of the answer, and the search terms for each query they made to HuskySearch. Users described their belief in correctness by circling one of "Very sure," "Pretty sure," "Not sure," or "Didn't finish."

3.2.1 Accuracy

To determine whether the CIE users were better able to answer questions correctly, we compared the percentage of accurate responses in control and CIE groups. These percentages are shown in Table 2. The two easier questions, Kevin and Utah, were answered well enough by the base system that the CIE didn't have much of an impact either way. The accuracy rate of Group 2 begins to surpass the control group starting with the Tree question. Of note is the MS query. Because "Microsoft" and "IPO" are such common terms, users were unable to make much headway using the obvious keywords. However, a few users commented in the CIE group that they were able to take advantage of previous queries that directed them to the proper site. In particular, a user commented that the inclusion of the term "1986" from a previous query was instrumental in finding the data. Thus, from this data, it appears that CIE does aid users when the questions are not easily answered from the obvious keyword search.

This example highlights an important aspect of using results documents from previous queries as indexed documents. In a small group searching for a similar data, members of the group are able to collaborate by using the queries of others, even if their own query is unique. Table 3 shows the total number of queries, the percent of unique queries, and the average number of terms per query. Even with users trying to answer the same question using between 2 and 4 terms, the percentage of unique queries is staggering, at over 80% for each question. This underscores the benefit for this kind of collaboration in guiding users to queries that provide useful information.

6

	Kevin	Utah	Tree	Can	MS
Terms	2.95	3.94	2.91	2.76	3.04
Total	38	34	31	34	54
% Uniq	81.6	82.4	80.7	85.3	92.6

Table 3: Average number of terms per query, total number of queries, and percentage of total queries that are unique. Query modifiers, such as searching for "All these words" or "the Person" are not counted as separate terms, but are used to determine uniqueness.

3.2.2 Confidence

Previous user studies have noticed discrepancies between the recall users thought they achieved versus the recall they had actually achieved[7, 30]. We wanted to see whether users had a realistic picture of the correctness of the answer they found, as well as whether or not the CIE system improved or degraded that judgment. For each question, we asked the user to enter their confidence in the answer as one of "Very sure," "Pretty sure," "Not sure," or "Didn't finish." For those that were able to answer the question, we converted their confidence to c_i having possible values of 0, 0.5, or 1 (corresponding to "Not sure" through "Very sure") and compared their confidence in their answer to the actual correctness of it, a_i, which was either 0 or 1, via the formula:

$$1 - |a_i - c_i|$$

These results are summarized in Figure 4.

The data shown does indicate that for the first three questions, both control and CIE users are able to accurately judge their results to the same degree. However, the Can and MS questions suggest again that for the harder questions, CIE users are both more sure of themselves and are more likely to have a realistic belief in their results.

3.2.3 Speed

In addition to accuracy and confidence, we also measured if users were able to answer the questions faster with the CIE system. Users were asked to time themselves, accurate to the nearest minute, and they were
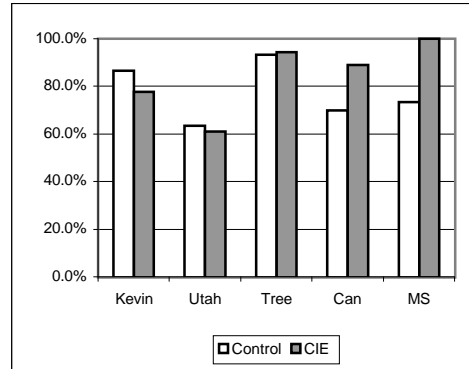

Figure 4: Accuracy of Result Judgment. This figure shows what percent of the time users correctly judge whether their answer is accurate or not.
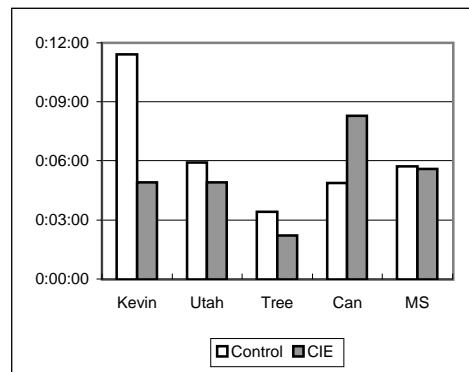

Figure 5: Time to answer each question.

asked to write down comments and their searches as they went along, so these numbers should be thought of as approximations. Our findings are summarized in Figure 5.

Two questions stand out: Kevin and Can. The rest suggest a trend that CIE users are able to answer faster, but the differences aren't statistically significant. The Kevin question shows that the CIE users are able to find the answer much faster than the control group. This is unexpected, as we feel that this is the easiest question and the CIE won't contribute that much to its success. However, we observe from the user comments that many in the CIE group found

a Yahoo! Filmography page[1] in the top few selections; this was caused in part by the ColRes and ColResSelect groups boosting that page's ranking.

The Can question, regarding the number of members in Canada's parliament, was also unexpected. Exploring the user's answers also revealed some confusion as to the "proper" answer to the question. Several users responded with the correct answer: 104 in the Senate, 301 in the House of Commons. However, many responded with 104 in the Senate and 295 in the House, which was the correct number for the previous year's parliament. After a brief bit of searching, it became clear that several official Canadian government pages had not been updated with the new figure. From the user comments, it appears that several users in the CIE group apparently spent extra time trying to determine which number was the current and correct number. In this case, the extra information returned by the CIE system was actually a hindrance, as they often contained wrong or out-of-date information.

The Can question illustrates a potential downside to the CIE system. If in fact users find data that is inaccurate, the enhancement method may make it more likely for future users to also find the inaccurate data. Thus, care must be taken so that inaccurate or irrelevant index enhancements can be detected and removed.

3.2.4 Multiple Queries

One trend stood out in the user responses that was unexpected. On the user form, there were spaces listed for three queries, and users were instructed to write down information about further queries on the back. Table 4 lists the percentage of time users entered a secondary and tertiary queries. Only the final query, which went unanswered by the students, had a significant amount of secondary and tertiary queries. This gives the indication that users make only a single query over 50% of the time and try to find the information they require by visiting subsequent hyperlinks in the returned results. Indeed, many comments from the users indicated that they spent a fair

---

[1] http://www.yahoo.com/Entertainment/
Movies_and_Films/Actors_and_Actresses/Spacey__Kevin/

	Secondary	Tertiary
Kevin	34.6	11.5
Utah	23.1	7.7
Tree	15.4	3.8
Can	23.1	7.7
MS	65.4	42.3

Table 4: Percentage of users making secondary and tertiary queries. There were only 4 queries in the entire study that were beyond the third.

amount of time "digging" through links. This phenomenon deserves more investigation; if it holds that users tend not to make secondary queries, then it may call into question the usefulness of any form of query refinement on the WWW.

4 Future Work

The CIE system is still in early development, and there is a great deal of future work that can be done. The four indices presented in this paper are simple in nature, and more sophisticated CIE systems should be explored and evaluated. One idea that we will be pursuing is to expand the ColRes and ColResSelect to include links from those pages to some finite depth, such as all pages 3 links away from a page a user clicked on.

An interesting scaling issue arises from using results documents as indexable documents. If a search service gets a million hits a day, then a million new results documents will be added into the index. Questions arise as to which of these results documents are still useful, and which are likely to have been superseded by other documents? Another more pragmatic issue is that no index can grow forever; at large scale, some of the results documents will need to be removed to conserve space. Thus, some metric to determine the quality of these documents needs to be developed.

Another question regarding scaling is whether or not user search patterns are such that there are a significant number of users that are searching for similar information. Certainly there are flurries of topical

interest, such as queries about the contenders before a championship game or queries pertaining to recent news. Thus, while CIE is likely to be useful for groups as large as corporations or universities, it is unclear if the collaborative benefits will scale to a worldwide audience and how large of an effect it will have.

The question of portability of this system is still open. It is unclear if this system would be of benefit in a TREC-like environment. Effort should be made to explore using non-WWW search systems as the basis for CIE.

Finally, more exhaustive testing is also in order. The results presented in this paper are promising, but the systems do need to be evaluated further. Of particular note is the observation that users who search the WWW do not often make more than one query; this observation needs to be further evaluated. If it is true, then it indicates that search enhancements should focus on what can be done in the initial query versus making improvements to refinement features.

We are currently in the process of creating a test corpus of questions, user-created keyword queries, documents, and user-defined relevance judgments based on our user study in order to compare future systems and other systems against the one presented in this paper. We will release this corpus publicly and make it initially available via the HuskySearch web site.

5   Related Work

Brauen reported his experiments on using user interaction to modify a searchable index[3]. He used the SMART[25] system as his testbed, and enhanced his index by directly modifying the document vectors in the index. He evaluated enhancing documents that appear relevant as well as documents that appeared irrelevant, in order to better separate the relevant documents in subsequent queries. He too found that this type of enhancement leads to higher precision and recall. The CIE architecture we have presented generalizes Brauen's system, and his index enhancement algorithms would still be appropriate for an index that used document vectors in a way similar to SMART.

Recently, Raghavan[24] described an elegant method

of locating stored optimized queries by comparing results from a current query to the results from the optimized query. The ColList and ColListSelect indices are in part inspired by this work. Unlike Raghavan, we do not attempt to retrieve only optimized queries, and we take advantage of having document summaries available in the results list.

Fitzpatrick and Dent explored using prior queries to expand a query automatically, and demonstrated performance improvements on TREC benchmarks[12]. Fitzpatrick and Dent's approach is complementary to ours: they modify the query to better match relevant documents, whereas we modify the document representation to better match the query. An obvious next step would be to combine both approaches.

There has also been substantial work on collaborative filtering to help with resource discovery. GroupLens[19] uses collaborative filtering to recommend USENET news articles based on whether or not other users with a similar profile read a particular article. The FireFly system, derived from Ringo, demonstrates that collaborative filtering could be used in the context of selecting music based on comparing two people's music profiles and suggesting the differences in similar profiles[11].

Our work differs from these in that users do not need to enter an explicit profile prior to using the system. However, each query and the results obtained can be thought of as one of many user profiles, and the CIE works by trying to match previous user profiles to a new one comprised solely of the query. It is an open question whether our methods of matching profiles are portable to traditional collaborative filtering methods.

6   Conclusions

We present Collaborative Index Enhancement, an architecture for enhancing a searchable index based on the experience of previous users. The key idea behind CIE is to take the results documents from a query and feed it back into the source index or indices in some manner. We demonstrate a prototype system based on the HuskySearch search service that implements CIE using auxiliary search indices. This

implementation allows us to use and experiment on several different CIE methods at once, without the need to modify or even control the original WWW indices we use.

In order to validate that CIE was a useful addition to HuskySearch, we conducted a series of experiments based on log analyses and a user study. Results from our log analyses indicate that the results documents are useful as index documents, and that documents selected in some manner by the user tend to be more beneficial in future queries. Our user study indicates that CIE was beneficial, and suggests that CIE aided users in answering harder questions as well as in speeding their discovery for easier questions. The results also suggest that CIE users were better able to correctly judge the quality of their answer for harder questions. Taken all together, these results suggest that CIE is a useful addition to HuskySearch.

The interface for our CIE system has two additional benefits: effortless collaboration and one step query refinement. By using results documents as index documents, users searching for the same or similar information as previous users are able to see previous queries that may be more useful, and can either modify their own query or can click directly on the link to refine their query to the previous one.

During the user study, we observed that users tend to make only one query. We hypothesize that this is because users prefer browsing results and following hyperlinks within those results rather than executing further queries.

Selecting documents for an index is now almost entirely automatic for large-scale systems. As large-scale systems continue to grow, it becomes more difficult to separate relevant documents from the irrelevant ones. CIE is one method that can help alleviate this problem by letting users augment the system so that it is better able to deliver what users require. This paper presents the beginning exploration into CIE, a general method that has much promise in helping users find relevant information quickly.

7 Acknowledgments

References

[1] Mark S. Ackerman, Brian Starr, and Michael Pazzani. The Do-I-Care Agent: Effective Social Discovery and Filtering on the Web. In *RIAO '97: Computer-Assisted Information Searching on the Internet*, pages 17–31, Montreal, CA, June 1997.

[2] N. J. Belkin, P. Kantor, C. Cool, and R. Quatrain. Combining Evidence for Information Retrieval. In Donna Harman, editor, *TREC-2, Proc. of the Second Text REtrieval Conference*, 1993.

[3] T. L. Brauen. Document Vector Modifications in the SMART Retrieval System. In *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, Englewood Cliffs, NJ:, 1971.

[4] James Callan, Zhihong Lu, and Bruce Croft. Searching Distributed Collections with Inference Networks. In *Proc. of the 1995 SIGIR Conference*, Seattle, WA, July 1995.

[5] Digital Equipment Corporation. Alta Vista Home Page.
URL: http://www.altavista.digital.com.

[6] Daniel Dreilinger. Integrating Heterogeneous WWW Search Engines.
URL: ftp://132.239.54.5/savvy-/report.ps.gz, May 1995.

[7] Efthimis Efthimiadis. *Interactive Query Expansion and Relevance Feedback for Document Retrieval Systems*. PhD thesis, City University, London, 1992.

[8] Efthimis N. Efthimiadis. Query Expansion. In Martha E. Williams, editor, *ARIST*, volume 31, chapter 4. Information Today, Inc., Medford, NJ, 1996.

[9] Excite, Inc. Excite Home Page.
URL: http://www.excite.com.

[10] David Filo and Jerry Yang. Yahoo Home Page.
URL: http://www.yahoo.com.

[11] FireFly Networks Inc. The FireFly Home Page.
URL: http://www.firefly.net.

[12] Larry Fitzpatrick and Mei Dent. Automatic Feedback Using Past Queries: Social Searching? In *Proc. of the 1997 SIGIR Conference*, pages 306–312, Philadelphia, PA, July 1997.

[13] Susan Gauch, Guijun Wang, and Mario Gomez. ProFusion: Intelligent Fusion from Multiple, Distributed Search Engines. *Journal of Universal Computing*, 2(9), Sept 1996.
URL: http://www.ittc.ukans.edu/~sgauch-/papers/JUCS96.ps.

[14] Donna K. Harman and Ellen M. Voorhees, editors. *Information Technology: The Fifth Text REtrieval Conference*, Gathersburg, MD, nov 1996. Dept. of Commerce, National Institute of Standards and Technology.
URL: http://trec.nist.gov.

[15] Adele Howe and Daniel Dreilinger. SavvySearch: A Meta-Search Engine that Learns Which Search Engines to Query. *AI Magazine*, 18(2), summer 1997.
URL: http://daniel.www.media.mit.edu/people/daniel/papers/ss-aimag.ps.gz.

[16] Go2Net Inc. MetaCrawler Home Page.
URL: http://www.metacrawler.com.

[17] Inktomi, Inc. HotBot Home Page.
URL: http://www.hotbot.com.

[18] Karen Sparck Jones. Reflections on TREC. *Information Processing & Management*, 31(3), 1995.

[19] J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM*, 40(3):77–87, 1997.
URL: http://www.acm.org/pubs/citations-/journals/cacm/1997-40-3/p77-konstan/.

[20] Michael L. Mauldin and John R. R. Leavitt. Web Agent Related Research at the Center for Machine Translation. In *Proceedings of SIGNIDR V*, McLean, Virginia, August 1994.

[21] Northern Light Technology LLC . Northern Light Home Page.
URL: http://www.northernlight.com/.

[22] Brian Pinkerton. WebCrawler Home Page.
URL: http://webcrawler.com.

[23] PlanetSearch Network Inc. PlanetSearch Home Page.
URL: http://www.planetsearch.com.

[24] Vijay V. Raghavan and Hayri Sever. On the Reuse of Past Optimal Queries. In *Proc. of the 1995 SIGIR Conference*, pages 344–350, Seattle, WA, July 1995.

[25] Gerard Salton, editor. *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice Hall, Englewood Cliffs, NJ, 1971.

[26] Erik Selberg and Oren Etzioni. Multi-Service Search and Comparison Using the MetaCrawler. In *Proc. 4th World Wide Web Conference*, pages 195–208, Boston, MA USA, December 1995.
URL: http://huskysearch.cs.washington.edu-/papers/www4/html/Overview.html.

[27] Erik Selberg and Oren Etzioni. The Meta-Crawler Architecture for Resource Aggregation on the Web. *IEEE Expert*, 12(1):8–14, January 1997.

[28] U. Shardanand and Pattie Maes. Social Information Filtering: Algorithms for Automating 'Word of Mouth'. In *Proceedings of the CHI-95 Conference*, Denver, CO, may 1995.

[29] Amanda Spink. Term Relevance Feedback and Query Expansion: Relation to Design. In *Proc. of the 1994 SIGIR Conference*, pages 81–90, Dublin, Ireland, July 1994.

[30] Louise T. Su. The relevance of recall and precision in user evaluation. *J. American Society of Information Science*, 45(3):207–217, 1994.

[31] The Daily of the University of Washington. The Online Daily of the University of Washington.
URL: http://www.thedaily.washington.edu.

[32] Howard Turtle. Natural Language vs. Boolean Query Evaluation: A Comparison of Retrieval Performance. In *Proc. of the 1994 SIGIR Conference*, pages 212–220, Dublin, Ireland, July 1994.

[33] O. Zamir, O. Etzioni, O. Madani, and R. Karp. Fast and Intuitive Clusturing of Web Documents. In *Proc. 3rd Int. Conf. Knowledge Discovery and Data Mining*, 1997.