Don't Look Stupid: Avoiding Pitfalls when Recommending Research Papers

Sean M. McNee, Nishikant Kapoor, Joseph A. Konstan
GroupLens Research
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, Minnesota 55455 USA
{mcnee, nkapoor, konstan}@cs.umn.edu

ABSTRACT
If recommenders are to help people be more productive, they need to support a wide variety of real-world information seeking tasks, such as those found when seeking research papers in a digital library. There are many potential pitfalls, including not knowing what tasks to support, generating recommendations for the wrong task, or even failing to generate any meaningful recommendations whatsoever. We posit that different recommender algorithms are better suited to certain information seeking tasks. In this work, we perform a detailed user study with over 130 users to understand these differences between recommender algorithms through an online survey of paper recommendations from the ACM Digital Library. We found that pitfalls are hard to avoid. Two of our algorithms generated 'atypical' recommendations—recommendations that were unrelated to their input baskets. Users reacted accordingly, providing strong negative results for these algorithms. Results from our 'typical' algorithms show some qualitative differences, but since users were exposed to two algorithms, the results may be biased. We present a wide variety of results, teasing out differences between algorithms. Finally, we succinctly summarize our most striking results as "Don't Look Stupid" in front of users.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *information filtering, relevance feedback, retrieval models*

General Terms
Algorithms, Experimentation, Human Factors

Keywords
Personalization, Recommender Systems, Human-Recommender Interaction, Collaborative Filtering, Content-based Filtering, Information Seeking, Digital Libraries

CSCW'06, November 4–8, 2006, Banff, Alberta, Canada.

1. INTRODUCTION
Recommender systems are supposed to help users navigate through complex information spaces by suggesting which items a user should avoid and which items a user should consume. They have proven to be successful in many domains, including Usenet netnews [15], movies [10], music [28], and jokes [7], among others. Even more, recommenders have transitioned from a research curiosity into products and services used everyday, including Amazon.com, Yahoo! Music, TiVo, and even Apple's iTunes Music Store.

Yet, with this growing usage, there is a feeling that recommenders are not living up to their initial promise. Recommenders have mostly been applied to lower-density information spaces—spaces where users are not required to make an intensive effort to understand and process recommended information (i.e. such as movies, music, and jokes) [13]. Moreover, recommenders have supported a limited number of tasks (i.e. a movie recommender can only help find a movie to watch). Can recommenders help people be productive, or only help people make e-commerce purchasing decisions? Herlocker et al. stated it best when they said, "There is an emerging understanding that good recommendation accuracy alone does not give users of recommender systems an effective and satisfying experience. Recommender systems must provide not just accuracy, but also *usefulness*." [9] (Emphasis in original)

But what is usefulness? We believe a useful recommendation is one that meets a user's current, specific need. It is not a binary measure, but rather a concept for determining how people use a recommender, what they use one for, and why they are using one. Current systems, such as e-commerce websites, have predefined a user's need into their business agendas—they decide if a system is useful for a user! Users have their own opinions about the recommendations they receive, and we believe if recommenders should make personalized recommendations, they should listen to users' personalized opinions.

There are many recommender pitfalls. These include not building user confidence (trust failure), not generating any recommendations (knowledge failure), generating incorrect recommendations (personalization failure), and generating recommendations to meet the wrong need (context failure), among others. Avoiding these pitfalls is difficult yet critical for the continued growth and acceptance of recommenders as knowledge management tools.

This concern is of even greater importance as recommenders move into denser information spaces—spaces where users face

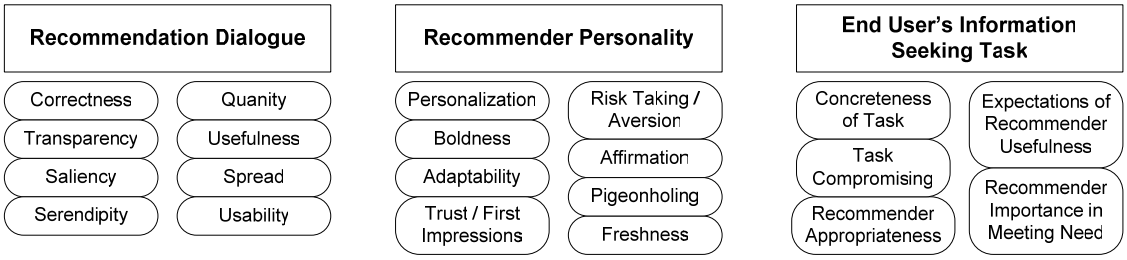

Figure 1-1: Aspects of Human-Recommender Interaction. The Aspects are divided into three 'Pillars'

serious challenges and the cost of failure is high. One such space is that of researchers seeking peer-reviewed, published work from a digital library. In previous work, we generated recommendations for computer science research papers [21]. In that paper and in follow-up work, we found that not only could recommenders generate high quality recommendations in this domain, but also users felt that various recommender algorithms generated qualitatively different, *yet meaningful*, recommendation lists [21, 31]. Even though this work was in a denser information space of a digital library, it only supported one task: find more citations for a given paper.

To make recommenders more useful, we believe they must support multiple information-seeking tasks. To test this, we build on previous results. We hypothesize that the differences between algorithms are quantifiable and predictable, and these differences are meaningful and valuable for end users. That is, some algorithms are better suited for particular user information-seeking tasks. We believe that matching users, and their specific tasks, to the appropriate algorithm will increase use satisfaction, efficiency, and the usefulness of the recommender system.

We will use Human-Recommender Interaction theory as our grounding to test these hypotheses. Human-Recommender Interaction (HRI) is a framework and process model for understanding recommender algorithms, users, and information seeking tasks [18-20]. It takes a user-centric view of the recommendation process, shifting the focus of attention from the system and associated algorithms to the (possibly repeated) interactions users have with such systems. By describing recommendation lists using descriptive keywords (called Aspects), HRI provides a language to articulate the kinds of items that would best meet a user's information seeking task. For example, an experienced user may want a high level of Boldness and Saliency from a recommender, where as a novice may want more Transparent and Affirming recommendations. See Figure 1-1 for an overview of HRI Aspects.

In this paper, we report on a user study of over 130 users on research paper recommendations we generated from the ACM Digital Library using several different recommender algorithms. We asked users about the suitability of these algorithms for different information seeking tasks, as well as for their opinions on the recommendation lists across multiple dimensions, dimensions based on HRI.

2. DIGITAL LIBRARIES, INFORMATION SEEKING TASKS, AND RECOMMENDERS

A main tenet of HRI is that recommenders need to tailor recommendation lists not just to a user, but to a user's information seeking task. To understand this, we will review information seeking theory, present example tasks in a DL environment, reflect on how recommenders change digital libraries, and finally explain how to apply HRI in this domain.

Information seeking theory provides us with a framework to understand user information needs and context. Models of information seeking behavior, including Taylor's Four Stages of Information Need, Wilson's Mechanisms and Motivations model, Dervin's theory of Sense Making, and Kuhlthau's Information Search Process [5, 16], reveal the ways in which emotion, uncertainty, and compromise affect the quality and nature of a user's information search and its results. More recently, Information Foraging theory suggests how users 'hunt' for information by analyzing the current cues, or scents, in their environment [24].

These theories ground our analysis of what forms of information seeking behavior appear in a digital library environment: users come to a DL looking for research information, but their confidence, emotional state, and comfort/familiarity with the system affect their information seeking behavior as much as their intellectual desire to solve their information need. One method for gathering this information in traditional libraries was the 'reference interview' where the librarian and user had a continuing discussion about the user's information seeking task [30]. As librarians have been replaced by search boxes, users may have faster access to raw information, but without the support and help they may need. By leveraging the opinions of other DL users as well as content authors, recommenders can bring that 'human touch' into digital libraries.

In particular, HRI suggests that users and recommenders have a similar interaction with the end goal of generating meaningful recommendations. In order to apply HRI to digital libraries, such as the ACM Digital Library, we need an understanding of possible user tasks in this domain. We will review a few examples here.

Find references to fit a document. This is our baseline task. Given a document and list of references, what additional references might be appropriate to consider and review? Documents of interest include paper drafts, theses, grant proposals, and book chapters, among others.

Maintain awareness in a research field. Dedicated researchers continually need a stream of information on the events in their research area. It is not always clear which new items are interesting, or what older items have become relevant for various reasons. This task includes maintaining a prioritized list of papers to read, pushing a "paper of the week", or triggering an alert whenever a new paper of interest is added to the collection.

Find people to send paper preprints and e-prints. Junior academics, for example, often send copies of their papers to senior colleagues for advice and support. Building these connections help research communities grow by linking new and established researchers together, but often new researchers do not know who might be interested in their work.

Inform collection management at a research library. With declining government support for universities and increasing journal subscription costs, nearly all public university libraries in the United States (and many private and corporate libraries across the world) must make careful decisions about which subscriptions to start, renew, or discontinue. Gathering the needed information is an elaborate and expertise-intensive process, including journal usage analysis (both in-print and electronically), journal importance in the field, journal availability through other libraries, journal relevance to faculty and students at the university, and the cost of the subscription. Moreover, finding information on journals for which the library does not have a subscription can be very difficult.

These tasks are only suggestive of what users might want from a digital library. Since each task has specific needs, we believe incorporating a *tuned* recommender into a DL can help better meet these tasks. The HRI Process Model suggests how we can tune recommenders to meet user tasks; it is done through the language of the HRI Aspects. Aspects are selected that best describe the kinds of results users expect to see from the recommender for this task [20]. The HRI Aspects can then be matched to recommender algorithms to select the most appropriate algorithm for the given task from a family of algorithms with known recommendation properties.

For example, the initial task, "Find references to fit a document", implies the user is trying to 'grow' a list. Thus, important Aspects could include Saliency (the emotional reaction a user has to a recommendation), Spread (the diversity of items from across the DL), Adaptability (how a recommender changes as a user changes), and Risk (recommending items based on confidence). Through these Aspects we can map this task to the appropriate algorithms, as algorithms are mapped to aspects via a variety of metrics [18]. One of the strengths of HRI is that it only claims the mappings exist. While it provides suggestions for creating the mappings automatically, we believe user input is critical to get the mappings correct. In this paper, we add users into this feedback loop by asking them to evaluate these mappings independently.

3. RECOMMENDING PAPERS

We focused on four recommender algorithms to cover in this domain: three collaborative algorithms and one content-based. We selected User-Based Collaborative Filtering (CF), a Naïve Bayesian Classifier, a version of Probabilistic Latent Semantic Indexing (PLSI), and a textual TF/IDF-based algorithm. We chose these because they represent a spread of recommendation approaches and are well known in the research community. Table 3-1 has a summary of the four algorithms.

To generate paper recommendations, a content-based algorithm would mine the text of each paper, and correlate an input stream of words to mined papers. The collaborative algorithms, on the other hand, ignore paper text. Further, instead of relying on user opinion to generate recommendations, previous work has mined the citations between papers to populate the ratings matrix for CF recommendations [21, 31]. In this model, each paper "votes" for the citations on its reference list. Thus, papers are "users" and citations are "items"—papers receive citation recommendations. When an author puts a citation in a paper she writes, it is an implicit vote for that citation. All of the collaborative algorithms use this data to train models and generate recommendations. While Byes and PLSI could be trained on many different features, we are only considering them as collaborative algorithms and training them with citation data.

In this collaborative model, a paper is a collection of citations; the content is ignored. Moreover, any collection of citations can be sent to the recommender algorithm for recommendations (a 'pseudo-paper', if you will). For example, we could send one paper's worth of citations or one author's entire bibliography of citations to the recommender. The interaction with the recommender is the same in both cases, but the meaning behind the interaction could be quite different.

Table 3-1: Summary of Recommender Algorithm Properties

	Run-time Speed	Pre-Process	Expected Rec. Type
User-User CF	Slow	None	High Serendipity
Naïve Bayes	Fast	Slow	High Ratability
PLSI	Very Fast	Very Slow	"Local" Serendipity
TF/IDF	Very Slow	Fast	High Similarity

3.1 User-based Collaborative Filtering

User-based Collaborative filtering (CF) has been widely used in recommender systems for over 12 years. It relies on opinions to generate recommendations: it assumes people similar to you also have similar opinions on items as you do, thus their opinions are recommendations to you. The most well known algorithm is User-based collaborative filtering (a.k.a. User-User CF, or Resnick's algorithm), a *k*-nearest neighbor recommendation algorithm [10, 26]. To generate recommendations for a target user, a neighborhood of *k* similar users is calculated, usually using Pearson correlation, and the highest weighted-average opinions of the neighborhood are returned as a recommendation list.

User-based collaborative filtering has many positives and negatives. It is a well-known and studied algorithm, and it has typically been among the most accurate predictors of user ratings. Because of this, it can be considered the "gold standard" of recommender algorithms. Conceptually, it is easy to understand and easy to implement. It has a fast and efficient startup, but can be slow during run-time, especially over sparse datasets. Choice of neighborhood sizes and similarity metrics can affect coverage and recommendation quality. It is felt in the community that User-based CF can generate 'serendipitous recommendations' because of the mixing of users across the dataset, but the structure of the dataset greatly affects the algorithm—it is possible that for some users, a User-based CF algorithm will never generate high quality recommendations.

There are many other variations of collaborative filtering, including item-based CF [27], content-boosted CF [8, 22], CF augmented with various machine learning optimizations [2, 4, 12], as well as hybrid approaches [3]. In this paper, we chose User-

based CF for its simplicity, high quality recommendations, and familiarity in the recommender systems community.

Based on results of previous work, we expect User-based CF to perform very well generating research paper recommendations. Since we are guaranteed each item in a digital library rates other items (all papers cite other papers!), we expect good level of 'cross-pollination' for serendipitous recommendations as well as high coverage. While the size of ACM Digital Library is not overwhelmingly large, issues of scale and sparsity are relevant if User-based CF is applied to larger databases, such as the PubMed digital library from NIH, containing over 16 million citations [23].

3.2 Naïve Bayes Classifier

If we assume independence of co-citation events in a research paper, then a Naïve Bayes Classifier [1] can be used to generate citation recommendations. All co-citations pairs are positive training examples. Posterior probabilities are the likelihood an item is co-cited with items (citations) in the target paper class. Items that are strongly classified as belonging to the target paper class are returned as recommendations (usually a top-*n* list).

A Bayes Classifier also has many pros and cons. It is a well-known and established algorithm in the machine learning literature. The independence assumption between co-citation pairs is a large one to make, but even in domains where this assumption fails, this classifier still performs quite well [6]. The classifier requires a model to generate recommendations, and this model must be re-built when new data is entered—a potentially slow process. Generating recommendations, however, is quick.

Because at heart, a Naïve Bayesian Classifier is a classification algorithm, we have specific expectations for it. Classifiers calculate 'most likely events'; they determine what classes items belong to. From a user's point of view in a recommender, a classifier returns the next most likely item the user will rate. This 'ratability' property of classifiers may not match a user's expectations in a recommender. As such, we believe that the recommendations would be more "mainstream", and not as serendipitous as User-based CF. Finally, full coverage could make a difference in low-data situations.

3.3 Probabilistic Latent Semantic Indexing

Probabilistic Latent Semantic Indexing (PLSI) is a relatively new algorithm to be applied to recommender systems [11]. PLSI is a probabilistic dimensional reduction algorithm with a strong mathematical foundation. In PLSI, the ratings space (citation space) is modeled using a set of independent latent classes. A user (i.e. paper) can have probabilistic memberships in multiple classes. PLSI uses a variant of the EM algorithm to optimize the class conditional parameters through a variational probability distribution for each rating (citing) instance based on previous model parameters. Items (citations) with the highest probabilities relative to the latent classes are recommended

Similar in spirit to the Bayes classifier, PLSI has similar benefits and drawbacks. It is mathematically rigorous, using latent classes to find probabilistic relationships between items. Model creation takes a very long time, requiring exceptional computing resources, but runtime is very efficient. It also will have 100% coverage. The latent classes are the key feature of this algorithm, thus performance and quality are highly dependent on the number of latent classes used.

This is a relatively new algorithm in this domain. We believe the latent aspect of this algorithm makes it more like User-based CF: generating interesting and serendipitous recommendations. In some ways, PLSI is like a 'soft' clustering algorithm, creating clusters around the latent classes. By performing local maximizations, the EM nature of the algorithm could reinforce more "popular" connections between items at the expense of "further reaching" (and potentially more interesting) connections, thus recommendations could be interesting locally, finding unexpected papers closely related to the input, but not interesting for finding a broad set of items from across the dataset.

3.4 Content-based Filtering with TF/IDF

Content-based filtering for papers uses the full text to generate recommendations. One of the most popular and well-used content filtering algorithms is Porter-stemmed Term Frequency/Inverse Document Frequency (TF/IDF) [25]. By using the frequency of stemmed words in a document compared to the entire corpus, this algorithm recommends items based on how well they match on important keywords.

TF/IDF is well known in the information retrieval community, and is considered as the standard vector-space model (a.k.a. "bag of words") approach to information processing. Since all papers contain content, they can be immediately processed, and thus recommended. This compares to the collaborative methods where the strength of the citations can unduly influence recommendations towards older, more cited works. Yet, "bag of words" content analysis is limited by semantic differences (e.g. "car" is not equated to "automobile"). It also may return many irrelevant results, recommending papers who mention the search terms only in passing. These issues are not as important in scientific literature as papers in one area have a generally agreed-upon vocabulary. Finally, TF/IDF does have an associated cost in pre-processing time and storage space, like other model-based algorithms.

Prior results show that when Content-based Filtering works, it performs very well at generating highly similar results. But more often than not, it fails to generate relevant results. This cherry-picking behavior limits it usefulness, especially when searching for novel or obscure work. On the other hand, this algorithm may excel at more conservative user tasks, especially those that start with a fair amount of information (e.g. "I know many of the conference papers in this research area, but I don't know about journal papers or workshops"). In general, we expect it to generate a narrow, yet predictable kind of recommendation list.

4. THE USER STUDY

Previous simulation experiments suggest recommender algorithms behave differently from each other on a variety of metrics [18]. In this study, we tackle the following research questions:

- How will real users describe the recommendations generated by different recommender algorithms?
- How will users rate the ability of recommender algorithms to meet the needs of specific user tasks?

4.1 Experimental Design

In the experiment, we generated two recommendation lists of five items each, called 'List A' and 'List B'. These recommendation lists were based on a 'basket' of papers that the user provided to us. Users were randomly assigned two algorithms; the display of

the lists was counterbalanced to cancel any possible order effects. Algorithm recommendation lists were pre-screened to make sure there was not strong overlap between result lists. At most, lists had 2 out of 5 recommendations in common. Since order is important in the returned results, if there was overlap, the list that has the shared result 'higher up on the list' retained the item. In the rare event of a tie, List A always won.

Our dataset is based on a snapshot of the ACM Digital Library, containing over 24,000 papers. For each paper, we have text of the paper's abstract; each paper cites at least two other papers in the dataset and is cited by at least two other papers in the dataset.

Users were recruited from several different pools. A subject either had to be an author of papers appearing in the dataset or be "very familiar" with the literature in a particular research area. Users were mined from the DL itself: a random selection of users who authored more than five papers in the dataset were sent an email invitation to participate. We also sent invitations to several computer science mailing lists asking for participants.

As a summary, the following algorithms were used in the online experiment:

1. User-based collaborative filtering at 50 neighbors, we used the SUGGEST Recommender as our implementation [14]
2. Porter-stemmed TF/IDF content-based filtering over paper titles, authors, keywords, and abstracts; we used the Bow Toolkit as our implementation [17]
3. Naïve Bayes Classifier trained using co-citation between papers as its feature; we used a custom implementation
4. Probabilistic Latent Semantic Indexing with 1000 classes; we used a unary co-occurrence latent semantic model and tempered our EM algorithm to avoid overfitting

4.2 Experiment Walkthrough

After consenting, subjects were asked as to their status as a researcher/academic (novice, expert, or outside the field of computer science) and as to their familiarity with the ACM Digital Library. Next, subjects were asked to create their 'basket', a list of papers to be sent to the recommender engines. This list is seeded by the name of an author who has published in the ACM Digital Library. There were options for selecting 'yourself' as an author or selecting 'someone else'. Figure 4-1 shows the author selection page.

After confirming the author selection, the subject was presented with a list of papers by that author in our dataset. The subject was allowed to prune any papers he did not want in the basket. If the subject stated he was an author himself, he saw a listing of papers he had written as well as a listing of papers he had cited from our dataset. If the subject selected another author, he was only presented with the listing of papers that author had published in our dataset. This decision had great implications on our results, as we will discuss later.

After pruning was finished, we presented the user with a selection of two possible information-seeking tasks, see Table 4-1. After selecting a task, the subject received two recommendation lists. See Figure 4-2 and Figure 4-3 for an overview of the paper selection and recommendation interfaces. There were two pages of questions associated with the recommendation lists. On the first page, we asked comparative questions: user opinion about the kinds of papers appearing on the lists. On the second page, we asked task-related questions: rating the suitability of each list to meet the subject's chosen information seeking task.

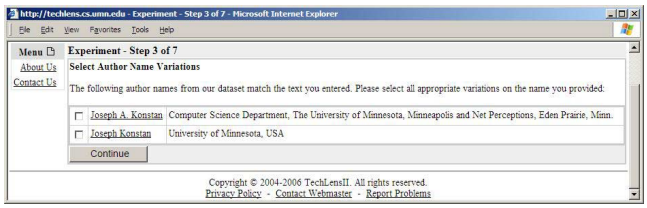

Figure 4-1: The Author Selection Page

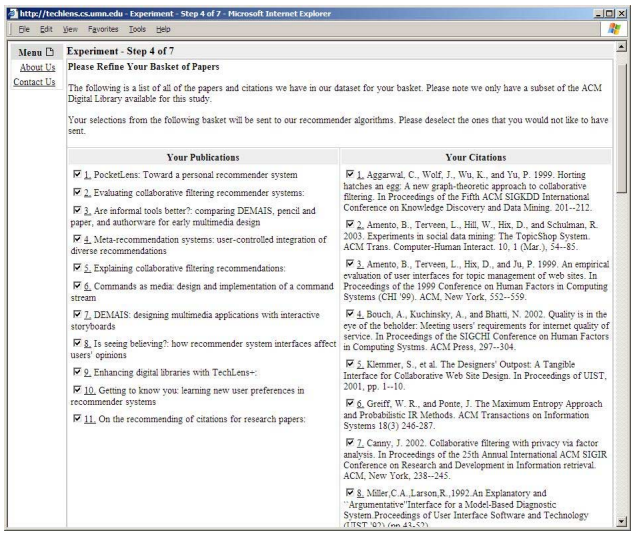

Figure 4-2: The Paper and Citation Selection Page

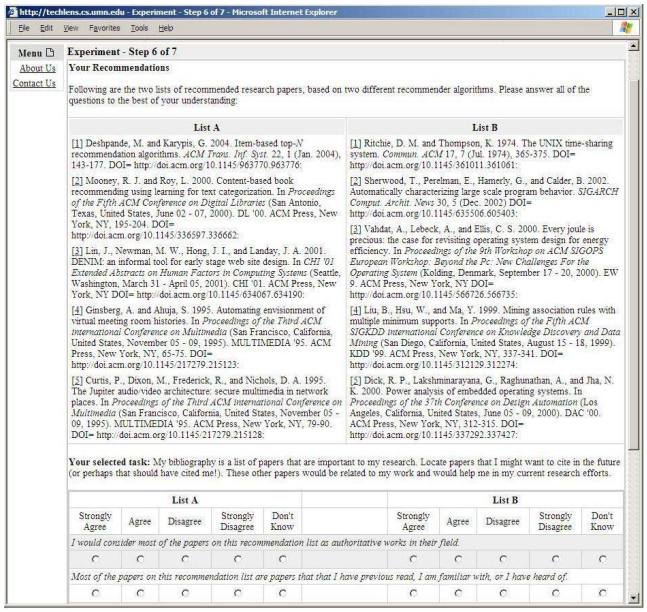

Figure 4-3: The Recommendation List Screen

Table 4-1: Available Information Seeking Tasks

Group	Available Tasks
Author, with my own work	My bibliography is a list of papers that are important to my research. Locate papers that I might want to cite in the future (or perhaps that should have cited me!). These other papers would be related to my work and would help me in my current research efforts.
	Given my bibliography as a description of what I find interesting in research, locate papers I would find interesting to read. These papers would not necessarily be work in my area, and could suggest potential collaborators or new research questions to explore in the future.
Someone else's publications	You would like to build a list of papers to describe a research area, and you started with this input basket. You are concerned about covering the area and want to make sure you no not miss any important papers. Which papers would next appear on this list?
	You are looking to expand your research interests into new or related areas, hoping to find interesting papers to read. Your input basket represents some papers in your current research area and interests. What papers might expand your knowledge, find new collaborators, or suggest new research areas to explore in the future?

Table 4-2: Survey Questions for Both Pages

Page 1: Comparative Questions	
1-1	I would consider most of the papers on this recommendation list as **authoritative** works in their field.
1-2	Most of the papers on this recommendation list are papers that that I have previous read, I am **familiar** with, or I have heard of.
1-3	This recommendation list is closely tuned-tailored-**personalized** to my given input basket, and is not a generic list of papers in this research area.
1-4	This list feels like a **good recommendation** list. I like it. It resonates with me.
1-5	This recommendation list contains papers that I was **not expecting** to see, but are good recommendations considering my input basket.
1-6	This list contains a good **spread** of papers from this research area and is not overly specialized, given the input basket.
Page 2: Task-related Questions	
2-1	In your opinion, which recommendation list generated a **better set** of recommendations for you and your task?
2-2	How **satisfied** are you with the recommendations in the list you preferred?
2-3	Pretend you were going to perform the **alternate** task, the one you did not choose (see above). In your opinion, which recommendation list generated a better set of recommendations for this other task?
2-4	We have the ability to send you a text file containing the standard ACM citation format for these recommendation lists. Would you like to **keep a copy** of these recommendation lists?
2-5	If so, which **format** would you prefer?
Note: For referencing, questions may be referred by number or by the **bolded text** in the question. No words were bold in the online survey.	

Table 4-3: Number of Users per Experimental Condition

Algorithm Pair	Users
User-User CF and TF/IDF	24
User-User CF and Naïve Bayesian	28
User-User CF and PLSI	25
TF/IDF and Naïve Bayesian	24
TF/IDF and PLSI	18
Naïve Bayesian and PLSI	23

A summary of all questions is in Table 4-2. On the first page, there were five possible responses to each question: 'Strongly Agree', 'Agree', 'Disagree', 'Strongly Disagree', and 'Not Sure'. Question 2-1, asking which list was better for the chosen task, had two options: 'List A' and 'List B'. We forced users to choose between the lists. In Question 2-2, we ask for the user's satisfaction with that selection, and responses ranged from 'Very Satisfied' to 'Very Dissatisfied', with a 'Not Sure' option. Question 2-3 was a hypothetical question, asking users to ponder the better list for the alternate task, and it had 'List A', 'List B', and 'Not Sure' as its possible responses. Note that for the hypothetical question we allowed a 'Not Sure' response. In Question 2-4, we offered users an option to receive a copy of the citations we generated for them, asking which citations they would prefer: 'List A Only', 'List B Only', 'Both Lists', or 'Neither List'. Finally, in question 2-5, we asked them which format they would prefer: 'the ACM DL BibTex Format' or the 'Standard ACM Citation Format'. While at first glance, Questions 2-4 and 2-5 appear to be a thank-you service for users participating in our study, we wanted to study the difference between stating they are satisfied with a recommendation list, and choosing to receive a copy of the list for future use—the difference between action and words. We believe action is a much stronger statement of satisfaction.

4.3 Results

138 subjects completed the survey over the 3-week experimental period. There were 18 students, 117 professors/researchers, and 7 non-computer scientists. All but six subjects were familiar with the ACM Digital Library, 104 used it on a regular basis. Each subject was assigned to two algorithms, see Table 4-3. We will first present overall survey results followed by algorithm-pair interactions and conclude with results by user selected task.

4.3.1 Survey Results: Comparative Questions

Figure 4-4 and Figure 4-5 summarize the results for the six questions on the first page. Answers have been binned into three categories: 'Agree', 'Disagree', and 'Not Sure'. As the figures show, there is a dramatic difference in user opinion about the four algorithms. Users tended to like ('agree' with) User CF and TF/IDF, where as users tended to dislike ('disagree' with) Bayes and PLSI. Results between these pairs are exceptionally significant ($p << 0.01$). We have found a pitfall; we discuss these unusual results in a separate in-depth analysis.

Focusing on User CF and TF/IDF, differences are statistically significant for Question 1-2 ($p < 0.01$) and almost so for Question 1-1 ($p = 0.11$). That is, User CF is more authoritative and generates more familiar results than TF/IDF. The trends on the other four questions also suggest that User CF generates recommendations that are more 'interesting'.

4.3.2 Survey Results: Task-Related Questions

There were two kinds of information-seeking tasks: "Find closely related papers", and "find more distant paper relationships". Users were required to select one of these tasks before receiving recommendations. Figure 4-6 shows how users judged the suitability of each of the four algorithms to their chosen task. Please note: users were forced to answer Question 2-1; there was no 'Not Sure' option. Continuing the above trend, users chose User CF and TF/IDF over Bayes and PLSI at about a 3:1 ratio ($p < 0.01$). Question 2-2 asked users about how satisfied they were with the algorithm they chose. Users were not pleased with Bayes

or PLSI (all results for Bayes were 'Disagree'!), and were happy ('agree') with User CF and TF/IDF just over half of the time.

Question 2-3 asked which algorithm would be best for the task the user did not select. When asked about this alternate task, 56% chose the same algorithm, 16% chose the other algorithm, and 28% were not sure. This trend carried across all four algorithms, except for Bayes, where 'Not Sure' was selected 50% of the time.

The final questions asked users if they wanted to keep a copy of the recommendations. While offered as a service, it provides an insight into user satisfaction. 32% of all users elected to 'keep a copy'. 67% of satisfied users chose to while only one user who was 'dissatisfied' chose to. Finally, users chose to keep recommendations generated from User CF and TF/IDF over Bayes and PLSI again at a 3:1 ratio ($p < 0.01$).

4.3.3 Results across Algorithm Pairs

Users answered questions in the context of a pair of algorithms. The comparative questions showed minor levels of interaction. When either User CF or TF/IDF was paired with Bayes or PLSI, users tended to score algorithms towards the extremes, most notably for questions 1-2 ("familiarity"), 1-3 ("personalized"), and 1-4 ("good recommendation list"). There were no discernable effects when User CF was paired with TF/IDF or when Bayes was paired with PLSI.

When answering task-related questions, User CF and TF/IDF dominated over Bayes and PLSI. When shown together, User CF was selected 90% of the time over Bayes and 95% over PLSI. TF/IDF was selected 88% and 94%, respectively. Compared against each other, User CF was selected more frequently (60%) than TF/IDF. Bayes and PLSI were preferred an equal number of times when placed together. More interestingly, when paired against Bayes and PLSI, User CF received higher praise, earning all of its 'Very Satisfied' scores in these cases. TF/IDF saw no such increase.

When asked for algorithm preferences for the alternate task, users rarely switched algorithms. Of the switches between algorithms, users switched to PLSI 9% of the time, and switched to Bayes 20% of the time. Users who first chose PLSI or Bayes always switched to either User CF or TF/IDF. Between User CF and TF/IDF, users were twice as likely to switch from User CF to TF/IDF as vice-versa.

4.3.4 Results by User-selected Task

There was a 60%-40% split between subjects selecting the "closely related papers" task or the "distant relationships" task. Responses showed some significant differences across tasks for different algorithms. For example, User-CF was more authoritative for the "close" task ($p < 0.05$), whereas TFIDF was more familiar for the "distant" task ($p < 0.005$). Further, PLSI was more personalized for "narrow" ($p < 0.01$), and both User-CF and PLSI were more unexpected for "distant" ($p < 0.05$ and $p < 0.075$, respectively).

Task selection was a 2x2 grid, where subjects also chose "Self" vs. "Someone else". 85% of subjects selected "Self". The above results are also significant for "Self"-only subjects. Trends in the "Someone else"-only subjects also support the results above.

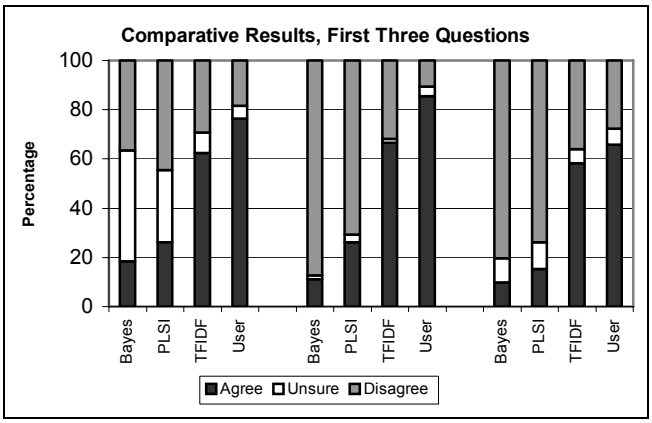

Figure 4-4: Displayed from Left to Right, Results for the Comparative Questions on 'Authoritative Works', 'Familiarity', and 'Personalization'

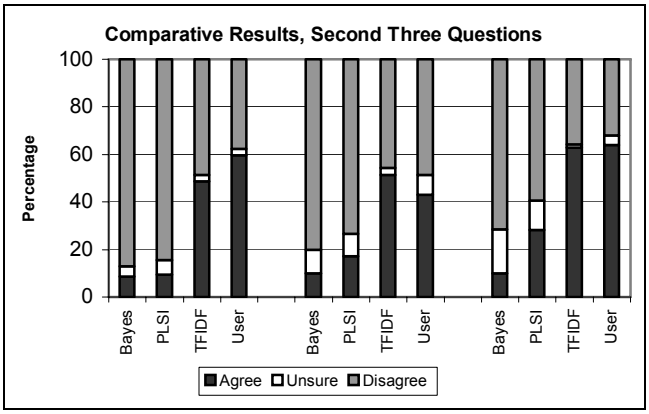

Figure 4-5: Displayed from Left to Right, Results for the Comparative Questions 'Is a Good Recommendation', 'Not Expecting', and 'Good Spread'

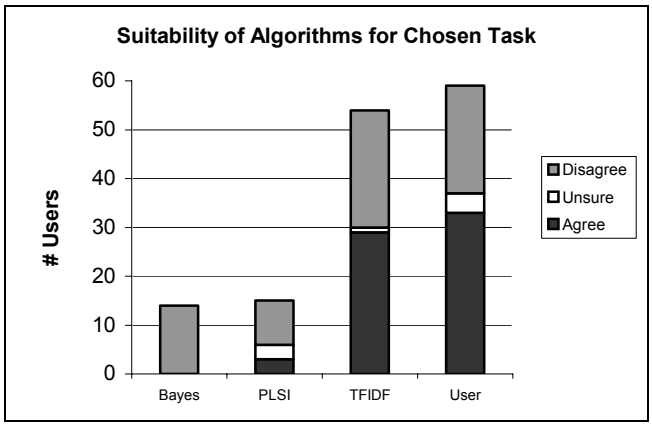

Figure 4-6: Results for Questions 2-1 and 2-2, User Opinion on Suitability of Algorithms for the Chosen User Task

Table 4-4: Overlap for Top 10 Recommendation Lists by Basket Size. Computed as the average of the intersection / union for all possible pairs in basket range. The lower the score, the fewer shared items. 0 means no overlap

	< 5	5 - 15	15 - 30	30+
Bayes	0.22	0.36	0.20	0.04
PLSI	0.01	0.01	0.002	0.004
TF/IDF	0.002	0	0	0.03
User	0	0	0.001	0.04

4.4 Analysis and Discussion

Before discussing the results of this study, we first perform a closer analysis of the atypical results we found and review our results for any potential order bias.

4.4.1 Analysis of Atypical Results

The Bayes and PLSI algorithms did not perform as expected. Prior work suggests that a Naïve Bayesian Classifier should be similar to User-based CF in this domain [21], and PLSI has shown to be of high quality in other domains [11]. What happened?

A careful review of our logs reveals that Bayes was generating similar recommendation lists for all users. Looking at top-10 recommendations lists, on average 20% of the recommended papers were identical for all users. Instead of returning personalized recommendations, Bayes returned the most highly co-cited items in the dataset. Changing the input basket size was did not have an effect: see Table 4-4. Trend data suggests as overlap increased, user satisfaction decreased across all questions, most notably for 'personalization' and 'good spread', but also for task usage as well.

PLSI was doing something equally as odd. While it returned personalized recommendations, a random sampling from logs revealed seemingly nonsensical recommendations. For example, an input basket composed of CHI papers received recommendations for operating system queuing analysis. We were unable to determine any patterns in the PLSI behavior.

In this work, the input basket size varied greatly (average of 26 items, stddev 32.6), with many baskets containing five or fewer items. Much as search engines need to return relevant results with little input [29], recommenders in this domain need to adapt to a variety of user models. User CF and TF/IDF we able to do so, Bayes and PLSI were not.

Yet, it is not accurate to say the algorithms 'failed'—some users did receive high quality recommendations. Rather, Bayes and PLSI were *inconsistent*. While Bayes averaged an overlap score of 0.20, the standard deviation was 0.26. We believe the quality of the recommendations generated by these two algorithms depended on how well-connected items in the input basket were to the rest of the dataset, especially for Bayes. For example, in our tests, adding one well-cited item to a questionable Bayesian basket radically improved the results. Adding such items to PLSI was not always as helpful. In PLSI, we believe many of the latent classes were 'junk classes', thus to improve recommendations, items from good classes need to be in the input basket.

4.4.2 Order Bias

The experiment was cross-balanced, so each algorithm appeared an equal number of times as List A and List B. With this balance, we were able to detect an order bias, where people preferred List A (on the left) to List B (on the right). Specifically, equal numbers of people selected List A and List B as their preferred list (Question 2-1). But those who chose List B were less satisfied with their selection (Question 2-2) ($p < 0.10$).

4.4.3 Comparative and User Task Analysis

The atypical results of the two recommender algorithms have skewed the results of the survey, making a detailed comparative analysis difficult. For User CF and TF/IDF, users recognized the recommendations and deemed them authoritative in their research area, but the lists did not contain unexpected results, and users were not sure if the recommendations contained came from a wide spread of the dataset. The differences between User CF and all other algorithms for authority and familiarity were significant.

These results are different from previously published work where User CF generated more unexpected recommendations than TF/IDF. We do reinforce the result that User CF generates authoritative paper recommendations. Further, previous work suggested that TF/IDF had a higher level of user satisfaction, whereas here, both algorithms received positive scores. Of course, the interaction effects may have influenced user responses, especially for User CF.

TF/IDF showed to be equally as useful for all four given user tasks with around a 54% satisfaction rating for all tasks. User CF showed a higher satisfaction rating (60%) for the 'find closely related papers' task. These results also reinforce previous findings, but only in context of the possible interaction effects.

Finally, we showed user opinion of algorithms varied by task. That is, depending on the current task, users perceived differences among algorithms across multiple dimensions. This is the first experimental evidence in support of HRI, suggesting that the qualities of algorithms important to users vary according to task.

4.4.4 Dataset Limitations

While our dataset was of high quality, it contained several limitations. The two striking ones are the scope of the data and range of the data. Only items for which the ACM holds copyright were contained in this dataset, many relevant papers were not included. Due to the nature of the DL, the bulk of items were published in the last 15 years. Finally, items had to cite other items and be cited by other items in the dataset. This limitation excluded much of the newest published work. We received several emails from users complaining about these limitations, including requests to add their work to our dataset, statements that they have changed research fields, and concerns for only receiving recommendations for older papers.

Further, these limitations may have affected PLSI's and the Naïve Bayesian Classifier's performance. While all papers were connected to each other, many were only weakly so. Both of these algorithms use strong statistical models as a basis for their recommendations. When calculated over a dataset such as this one, the meaningful statistics could be 'washed out' by the other calculations. In Bayes, a paper with a strong prior probability could influence posterior probabilities, or perhaps the naïve assumption was one we should not make in this domain. In PLSI, the local maximization calculations could reinforce stronger global calculations in place of the weaker, yet meaningful local connections. Finally, is worth asking the question if measuring co-citations is the correct mapping for recommenders in this domain. A complete analysis and discussion, however, is outside the scope of this paper.

5. IMPLICATIONS AND FUTURE WORK

As previously mentioned, Bayes and PLSI perform well as recommenders in offline simulation experiments. Specifically, both have scored well on accuracy metrics in using a leave-*n*-out methodology [1, 11]. As we have argued elsewhere (i.e. [19, 20]) such offline measures may not translate into useful measures for end users.

In particular, Bayes recommended users a combination of personalized and non-personalized recommendations. In many cases, the personalized recommendations were good suggestions. In fact, in offline leave-*n*-out scenarios, it did not matter that the recommendation lists contained a mixture of highly personalized and non-personalized items, as long as the withheld item appeared on the list, the algorithm scored well on the metric [20]. Users, however, were not satisfied with these recommendation lists. These results suggest that the research community's dependence on offline experiments have created a disconnect between algorithms that score well on accuracy metrics and algorithms that users will find useful.

This argument is even more subtle. In our testing of HRI, we performed an offline analysis of recommendation lists using a series of non-accuracy-based metrics. If the Naïve Bayes Classifier were generating a mixture of personalized and non-personalized results, a personalization metric should have revealed this problem. In our results, Bayes generated very personalized responses [20]. The difference between then and now is the input baskets. The baskets then were based on the citations from a single paper; usually such lists contain a mixture of well and loosely connected papers, lists for which Bayes could generate personalized recommendations. The input baskets here were different—they were based on papers written by a single author. This reveals not just the importance of the dataset but also the importance of the input basket when analyzing algorithms. The disconnect is even larger than we thought.

In previous work, we argued that showing one good recommendation in a list of five was enough to satisfy users. It is not that simple: showing one horrible recommendation in five is enough for users to lose confidence in the recommender. We call this the **Don't Look Stupid** principle: only show recommendation lists to users when you have some confidence in their usefulness. While this principle applies most dramatically when talking about Bayes and PLSI in our results, we believe it is just as important when dealing with users' information seeking tasks. A recommendation list is bad when it is not useful to the user, independent of why it is bad.

To understand this principle in context, we can use HRI. This experiment was one-time online user survey. These users had no previous experience with the recommender algorithms, and they were given an information seeking task. Because of this, many HRI Aspects are not relevant to our discussion, but a few become very important, such as: Correctness, Saliency, Trust, and Expectations of Usefulness. By being asked to be in an experiment, users had a heightened awareness of the recommendation algorithms; they expected the algorithms to be useful and they expected them to generate correct results. Indeed, we received several emails from users worried about the poor recommendations they received from either Bayes or PLSI. We had no time to build trust with our users, nor did the users gain a sense of the algorithms' personality. When users received nonsensical results, we believe they had a strong emotional reaction. The results went against their expectations of being an experiment to receive personalized recommendations. Thus, the users provided the strong negative feedback.

There many threads of possible future work. First, we need a deeper understanding of Bayes and PLSI in this domain. How much of the difficulties experienced in this work are related to properties of the algorithms, properties of the dataset and input baskets, or implications of how these algorithms were applied in this domain? Second, while this study provides evidence to the tenet of HRI that specific recommender algorithms are better suited to certain information needs, more work needs to be done. Yet it does raise one interested question from our HRI analysis, could a recommender be 'stupid' in front a user with whom the recommender has already built a relationship? This work must be done with real users; offline analysis is not enough. Finally, the performance of Bayes and PLSI in this domain suggest that dataset properties and input basket selection greatly influence recommendation lists, implying the need for a study comparing multiple datasets across multiple algorithms.

6. CONCLUSIONS

Recommending research papers in a digital library environment can help researchers become more productive. Human-Recommender Interaction argues that recommenders need to be approached from a user-centric perspective in order to remain relevant, useful, and effective in both this and other domains. HRI suggests tailoring a recommender to a user's information need is a way to this end. To test these ideas, we ran a study of 138 users using four recommender algorithms over the ACM Digital Library. Instead of validating our research questions, we ran into a large pitfall and discovered a more telling result: Don't Look Stupid. Recommenders that generate nonsensical results were not liked by users, even when the nonsensical recommendations were intermixed with meaningful results. These results suggest that it is critically important to select the correct recommender algorithm for the domain and users' information seeking tasks. Further, the evaluation must be done with real users, as current accuracy metrics cannot detect these problems.

7. ACKNOWLEDGMENTS

We would like to thank the ACM for providing us with a snapshot of the ACM Digital Library. Thanks to GroupLens Research and the University of Minnesota Libraries for their help and support, especially Shilad Sen for helping with PLSI. This research is funded by a grant from the University of Minnesota Libraries and by the National Science Foundation, grants DGE 95-54517, IIS 96-13960, IIS 97-34442, IIS 99-78717, and IIS 01-02229.

8. REFERENCES

[1] J.S. Breese, D. Heckerman and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering", *in Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence,* pp. 43-52, 1998.

[2] J. Browning and D.J. Miller, "A Maximum Entropy Approach for Collaborative Filtering", *J.VLSI Signal Process.Syst.,* vol. 37(2-3), pp. 199-209, 2004.

[3] R. Burke, "Hybrid Recommender Systems: Survey and Experiments", *User Modeling and User-Adapted Interaction,* vol. 12(4), pp. 331-370, 2002.

[4] J. Canny, "Collaborative Filtering with Privacy Via Factor Analysis", *in Proc. of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,* pp. 238-245, 2002.

[5] D.O. Case, *Looking for Information: A Survey of Research on Information Seeking, Needs, and Behavior,* San Diego: Academic Press, 2002, pp. 350.

[6] P. Domingos and M. Pazzani, "Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier", *in Proc. of the 13th International Conference on Machine Learning (ICML 96),* pp. 105-112, 1996.

[7] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A Constant Time Collaborative Filtering Algorithm", *Inf.Retr.,* vol. 4(2), pp. 133-151, 2001.

[8] N. Good, J.B. Schafer, J.A. Konstan, A. Borchers, B. Sarwar, J. Herlocker and J. Riedl, "Combining Collaborative Filtering with Personal Agents for Better Recommendations", *in Proc. of the Sixteenth National Conference on Artificial Intelligence,* pp. 439-446, 1999.

[9] J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl, "Evaluating Collaborative Filtering Recommender Systems", *ACM Trans.Inf.Syst.,* vol. 22(1), pp. 5-53, 2004.

[10] J.L. Herlocker, J.A. Konstan, A. Borchers and J. Riedl, "An Algorithmic Framework for Performing Collaborative Filtering", *in Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,* pp. 230-237, 1999.

[11] T. Hofmann, "Latent Semantic Models for Collaborative Filtering", *ACM Trans.Inf.Syst.,* vol. 22(1), pp. 89-115, 2004.

[12] Z. Huang, H. Chen, and D. Zeng, "Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering", *ACM Trans.Inf.Syst.,* vol. 22(1), pp. 116-142, 2004.

[13] I. Im and A. Hars, "Finding Information just for You: Knowledge Reuse using Collaborative Filtering Systems", *in Proc. of the Twenty-Second International Conference on Information Systems,* pp. 349-360, 2001.

[14] G. Karypis, "SUGGEST Top-N Recommendation Engine", http://www-users.cs.umn.edu/~karypis/suggest/, 2000.

[15] J.A. Konstan, B.N. Miller, D. Maltz, J.L. Herlocker, L.R. Gordon, and J. Riedl, "GroupLens: Applying Collaborative Filtering to Usenet News", *Commun ACM,* vol. 40(3), pp. 77-87, 1997.

[16] C.C. Kuhlthau, *Seeking Meaning: A Process Approach to Library and Information Services,* Westport, CT: Libraries Unlimited, 2004, pp. 247.

[17] A.K. McCallum, "Bow: A Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering", http://www-2.cs.cmu.edu/mccallum/bow/, 1996.

[18] S.M. McNee. *Meeting User Information Needs in Recommender Systems.* Ph.D. Dissertation, University of Minnesota. 2006.

[19] S.M. McNee, J. Riedl and J.A. Konstan, "Being Accurate is Not enough: How Accuracy Metrics have Hurt Recommender Systems", *in Ext. Abs. of the 2006 ACM Conference on Human Factors in Computing Systems,* pp. 997-1001, 2006.

[20] S.M. McNee, J. Riedl and J.A. Konstan, "Making Recommendations Better: An Analytic Model for Human-Recommender Interaction", *in Ext. Abs. of the 2006 ACM Conference on Human Factors in Computing Systems,* pp. 1003-1008, 2006.

[21] S.M. McNee, I. Albert, D. Cosley, P. Gopalkrishnan, S.K. Lam, A.M. Rashid, J.A. Konstan and J. Riedl, "On the Recommending of Citations for Research Papers", *in Proc. of the 2002 ACM Conference on Computer Supported Cooperative Work,* pp. 116-125, 2002.

[22] P. Melville, R.J. Mooney and R. Nagarajan, "Content-Boosted Collaborative Filtering for Improved Recommendations", *in Proc. of the Eighteenth National Conference on Artificial Intelligence,* pp. 187-192, 2002.

[23] National Institutes of Health (NIH), "Entrez PubMed", http://www.ncbi.nlm.nih.gov/entrez/, 2006.

[24] P. Pirolli, "Computational Models of Information Scent-Following in a very Large Browsable Text Collection", *in CHI '97: Proc. of the SIGCHI Conference on Human Factors in Computing Systems,* pp. 3-10, 1997.

[25] M.F. Porter, "An algorithm for suffix stripping," *in Readings in Information Retrieval,* K.S. Jones and P. Willett eds., San Francisco, CA: Morgan Kaufmann Publishers Inc, 1997, ch. 6, pp. 313-316.

[26] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews", *in Proc. of the 1994 ACM Conf. on Computer Supported Cooperative Work,* pp. 175-186, 1994.

[27] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms", *in Proc. of the Tenth International Conference on the World Wide Web,* pp. 285-295, 2001.

[28] U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating "Word of Mouth"", *in Proc. of the SIGCHI Conference on Human Factors in Computing Systems,* pp. 210-217, 1995.

[29] C. Silverstein, H. Marais, M. Henzinger, and M. Moricz, "Analysis of a very Large Web Search Engine Query Log", *SIGIR Forum,* vol. 33(1), pp. 6-12, 1999.

[30] R.S. Taylor, "Question-Negotiation and Information Seeking in Libraries", *College and Research Libraries,* vol. 29pp. 178-194, May, 1968.

[31] R. Torres, S.M. McNee, M. Abel, J.A. Konstan and J. Riedl, "Enhancing Digital Libraries with TechLens+", *in Proc. of the 2004 Joint ACM/IEEE Conference on Digital Libraries,* pp. 228- 236, 2004.