2009

# Search Personalization: Knowledge-Based Recommendation in Digital Libraries

Todd Will
*New Jersey Institute of Technology*, todd.wil@njit.edu

Anand Srinivasan
*New Jersey Institute of Technology*, as459@njit.edu

Il Im
*Yonsei University*, il.im@yonsei.ac.kr

Yi-Fang Brook Wu
*New Jersey Institute of Technology*, yi-fang.wu@njit.edu

# Search Personalization: Knowledge-Based Recommendation in Digital Libraries

**Todd Will**
New Jersey Institute of Technology
todd.will@njit.edu

**Anand Srinivasan**
New Jersey Institute of Technology
as459@njit.edu

**Il Im**
Yonsei University
il.im@yonsei.ac.kr

**Yi-Fang Brook Wu**
New Jersey Institute of Technology
yi-fang.wu@njit.edu

## ABSTRACT

Recommendation engines have made great strides in understanding and implementing search personalization techniques to provide interesting and relevant documents to users. The latest research effort advances a new type of recommendation technique, Knowledge Based (KB) engines, that strive to understand the context of the user's current information need and then filter information accordingly. The KB engine proposed in this paper requires less effort from the user in representing the search task and is the first of its kind implemented in a digital library setting. The KB engine performance was compared with Content Based (CB) and Collaborative Filtering (CF) recommendation techniques and the text search engine Lucene by asking sixty subjects to perform two different tasks to find relevant documents in a database of 212,000 documents from 22 National Science Digital Library (NSDL) collections. Our KB engine design outperforms CB, CF, and text search techniques in nearly all areas of evaluation.

### Keywords

Personalized search, recommendation engines, text search, knowledge based, content based, collaborative filtering

## INTRODUCTION

Digital library users are overwhelmed by the amount of information that is available and therefore require an effective search engine to guide them to the best documents available for their current search task. However, as search engines mostly consider keywords that the user enters, they do not consider the current information seeking context nor do they utilize other useful information provided by users such as a relevant document selected by the user. Despite the National Science Digital Library (NSDL) mission of providing scientifically rigorous resources to teachers and educators, these users preferred Google over the NSDL to find pedagogical resources due to three main factors: the NSDL did not appear to rank search results according to relevance, no advanced search interface is offered, and documents cannot be filtered by audience level (McCown, Bollen, and Nelson, 2005). While some of these shortcomings have been remedied, educators and other users of the NSDL would greatly benefit from the addition of personalized recommendation engines to adapt to their changing information needs.

Two common types of recommendation engines, content based (CB) and collaborative filtering (CF), have been proven useful in providing appropriate documents to users but do not consider the context of the individual's current information search. This study has developed a new personalized Knowledge-Based (KB) engine to overcome the limitations imposed by the CB and CF engines by considering the context of the current information need. The performance of the KB engine was compared to CB, CF, and text search techniques by asking sixty subjects at a technical university to undertake two different tasks and studying the performance of the four different systems according to both objective and subjective measures. The results show that the KB engine outperformed the CB, CF, and text search techniques in nearly all areas of evaluation due to its personalizable nature.

The following sections discuss previous work done in CB, CF, and KB recommendation engines and then describe the specific algorithms used to generate the recommendations in the study. The design and results of the experiment will then be provided, followed by the study limitations, and future work to be conducted.

## LITERATURE REVIEW

In this section, prior studies in collaborative filtering and content and knowledge based recommendation engines are discussed. Each type of technique has its own identified advantages and shortcomings.

### Collaborative Filtering Engines

CF engines use the behavior and evaluations of users to recommend resources to other users. A browsed or highly rated document by a high number of users should be recommended before a document that is rated poorly. A profile containing click histories and sometimes explicit ratings is compiled as each user browses documents. This history and ratings are then compared with the history of other users to find documents to recommend that are previously unknown to the current user. One version of CF engines requires users to explicitly assign ratings for every document that is browsed (Herlocker, Konstan, Borchers, and Riedl, 1999) whereas the simpler clickstream collaborative filtering (CCF) model considers only click histories (Breese, Heckerman, and Kadie, 1998). This recommendation engine considers all forms of digital media as only the identifier of the artifact and the browsing history of users is required. In the digital library domain, the CF engine effectively recommended research papers to its users (McNee, Albert, Cosley, Gopalkrishnan, Lam, Rashid, Konstan, and Riedl, 2002).

Despite the CF demonstrating promise at recommending relevant documents to users, several shortcomings have been identified. The scalability issue cannot be easily overcome; as the browsing history of users continually grows, the CF draws upon this history to increase accuracy of predictions but the efficiency of the system decreases (Sawar, Konstan, Borchers, Herlocker, Miller, and Riedl, 1998). Another CF algorithm problem is the cold start problem; all documents must be viewed or rated by at least one user in order to be recommended to other users. CF engines cannot overcome the sparseness problem where certain specialized documents are viewed by few users, leading to a decreased accuracy of recommendations as few or no selections exists for the document. Three common algorithms for CF recommendation employing CCF models are the Markov model, association rule, and clustering, each with its own advantages and drawbacks (Ahmad and Ahmad, 1999; Mobasher, 2004; Zhang and Im, 2002).

### Content Based Engines

While CF techniques have effectively provided highly relevant recommendations to users, it suffers from several problems that cannot be easily overcome. CB algorithms take a different approach by considering only textual information of documents in the collection. This technique identifies pairs of documents in the collection that have similar content and can then recommend documents that are similar to the one the user is currently viewing. Those resources in specialized domains will have an equal chance of recommendation to the user as more popular resources, overcoming the sparseness problem associated with CF engines.

Digital libraries consist entirely of digitized artifacts that can be exploited by the CB engine to generate recommendations. Most digital library document contain large amounts of textual information, as in the case of books and articles, while those resources that do not contain a large number of words, such as audio files and animations, are associated with a title and in most cases a description that can be used by the CB engine even if the media itself cannot. Most systems using the CB technique approach recommendation as a “classification problem with supervised learning” (Basu, Hirsh, and Cohen, 1998; Mooney and Roy 1999). Another approach involves the creation of statistical models, such as a regression model, based on a training data set to find relevant documents not already viewed by the user (Vucetic and Obradovic, 2000). As long as the resource contains at least some textual information, the CB engine can provide relevant documents by considering all documents in the collection as candidate recommendations.

### Knowledge Based Engines

KB engines take a completely different approach to document recommendation by considering the context of the individual user information need. This technique combats the problems associated with CB and CF approaches to tailor recommendations to the current session instead of using a click history or document content (Callan, Smeaton, Beaulieu, and Brusilovsky, 2003). KB engines consider the individual user’s background, search context, task information, and preferences by mapping individual needs to document properties through a detailed profile (Chung and Hong, 2001). This profile ranges from documents similar to those required by the user (Herlocker and Konstan, 2001) to the creation of an ontology by the user to group candidate recommendations into categories familiar to the user (Chaffee and Gauch, 2000). An earlier KB engine for digital libraries applied the browsing history and document type preferences, such as text or flash animations, to recommend documents (Tsai, Chiu, Lee, and Wang, 2006).

The KB engine guides the user through the information search from beginning to end. At the beginning, the user will typically search for general documents in the domain of interest. As the user reads more resources about the topic, he or she will become more familiar with the domain and adjust the profile to request more specific documents. In this way, the KB engine guides users to the best documents in the collection by allowing the relevance criteria to adapt to the changing stages of the information search (Vakkari and Hakkala, 2003), an adaptation the CB and CF techniques cannot accomplish.

## GENERAL RECOMMENDATION ENGINE (GRE) SYSTEM ARCHITECTURE

This section describes the design and implementation of the specific CB, CF, and KB algorithms used in this study. The interface consists of the popular, open source text search system Lucene (http://lucene.apache.org) to provide an initial set of indexed documents as well as the combined list of documents provided by the General Recommendation Engine (GRE) manager, consisting of CB, CF, and KB recommendations, on the right side.

### Content Based (CB) Engine Algorithm

The CB engine models each document as a Bag of Words, with feature words extracted to create a vector of words for each document. The cosine similarity measure calculates the cosine angle between every pair of document vectors in the document space. This similarity score and each pair of document ids is preprocessed and stored in the database for later use by the CB engine to generate recommendations. When a new document is encountered by the CB engine, the URL of the document is used to download the full text, extract content-bearing words, and quickly calculate the degree of similarity between the new document and other documents in the collection. After a certain number of new documents are added to the database, the similarity score between each new document and all other documents in the collection is calculated and stored. The CB engine then returns $n$ ($n$ is a predefined parameter) documents with the highest similarity scores to the current document that the user is viewing.

### Collaborative Filtering (CF) Algorithm

The CF engine uses the Markov model to generate recommendations due to its high prediction accuracy. The click history, up to the previous five clicks in the current session, of the user will be compared with the click histories of all other users to identify users with fully or partially matching clickstreams to be weight-averaged by the numbers in Table 1; these weights were optimized on our previous empirical studies. For example, if the current user's click history is A-B-C-D-E and there is a perfectly matching clickstream in the database that is T-T-T-T-T (the first row in Table 1), then the weight of 750 will be used for that clickstream. If there is a clickstream of Z-B-C-D-E, then the weight for F-T-T-T-T (700) will be used.

| First | Second | Third | Fourth | Fifth | Weight |
|---|---|---|---|---|---|
| T | T | T | T | T | 750 |
| F | T | T | T | T | 700 |
| T | F | T | T | T | 650 |
| T | T | F | T | T | 600 |
| T | T | T | F | T | 550 |
| T | T | T | T | F | 500 |
| … | … | … | … | … | … |

**Table 1. Weighted Average Matching Clickstream (Partial)**

As the CF engine requires at least one matching click in order to generate recommendations, the minimum number of matching clicks was set to one; using a higher threshold would result in few or no CF recommendations. When the user is not logged into the system, only the most recent click is used to generate recommendations. Among the next document in the past clickstreams, the $n$ documents with the highest weight averaged score would be provided to GRE manager as recommendations.

### Knowledge Based (KB) Algorithm

As most documents in our collection were crawled from NSDL pathway collections, the KB engine takes advantage of metadata provided by the collection managers by mapping the user's task description to document properties to further filter recommendations. First, the relative importance of each metadata element when determining whether to click on a relevant document must be rank ordered to determine the elements that should be completed in the profile. A survey was conducted at a technical university in New Jersey consisting of sixty subjects from predominantly math and science disciplines. For those users of the NSDL not represented in a university setting, the important metadata elements were selected based on a review of the literature (Masullo and Mack,1996; Hanson and Carlson, 2005). These elements, summarized in Table 2, are then provided in the user profile as initial options to be completed when performing a new information seeking task while all other elements were offered as advanced options that could be completed as the user becomes more familiar with the domain.

| Importance | Metadata Element |
|---|---|
| 1 | Description (Abstract) |
| 2 | Subject |
| 3 | Coverage |
| 4 | Audience |
| 5 | Relation |

**Table 2. Important Metadata Elements**

Since the NSDL has little control over metadata elements completed by individual collection managers, some important metadata elements were sparsely populated. Where possible, the missing elements were extracted from the full text of the document, but still many properties were not available. The automatic classification method Support Vector Machine (SVM) was chosen to populate the missing data in audience level and subject category fields as it was found to be most effective among classification algorithms (Kotsiantis, 2007); SVM classified the audience level with 80% accuracy and subject category with 59% accuracy.

The user can create many different profiles with one being set as default. The general search terms or task description is first entered with the elements in Table 2 offered as options to further describe the information seeking task. After a profile is saved, the KB engine initially queries Lucene with the task description to retrieve a set of candidate recommendations. The properties of these recommendations is compared with the user-specified metadata values in the profile to provide additional weight to the candidate recommendations. If a document metadata value matches that of the profile, a value of 1 is used; otherwise a value of 0 is used. If the document property is not provided in the returned document and its value is completed in the profile, a value of 0.5 is used. For some profile elements with continuous values, as in the case of publication date and audience level, a range of weights is used. The relevance score provided by Lucene is the degree of fit between the task description and the document; its value will range between 0 and 1. A final degree of fit score must be calculated between the document and the profile; the importance of each metadata element and the task description is found by dividing 1 by the number of elements completed in the profile plus one for the task description. For example, a user may elect to complete task keywords (“recommendation engines”), audience level (“college and above”), and subject category (“technology”); the relative importance of each element is 1 divided by the number of options entered in the profile, in this case three completed entries, so the relative importance of each completed profile element would be 0.33333. The most relevant document in the candidate recommendations returned by Lucene could have a relevance score of 0.9, an audience level of masters student (1 as it matches the level selected), and a subject category of earth science (0 as it does not match the selected category). The final score is calculated by multiplying the score by the importance of the metadata element and then adding them to find the final score for the document. In this case, the final score of the document is 0.63333. The top *n* items are returned to GRE manager by KB engine for display to the user.

### GRE Experimental System

The subjects will carry out two different information searching tasks in the experimental site and evaluate the performance of these recommendation techniques. As these three different techniques are designed differently and two require at least one click in order to generate recommendations, no recommendations are provided until Lucene provides an initial set of documents by considering a set of keywords entered by the user and the user selects a document in the search results. GRE manager requests candidate recommendations from the CB, CF, and KB engines by providing the current document URL to the CB and CF engines and the current user id to the CF and KB engines; GRE manager collates the recommendations and displays them to the user in an easily readable format.

The final ranking of documents provided to the user takes into account the number of engines that recommended the document and location in the candidate recommendation list. A document returned by all three engines is considered more relevant than one recommended by only two engines and finally those recommended by one engine. Within each group, the sum of the locations in the different recommendation engine lists (first being most relevant) would be calculated and the lowest value would be ranked higher. As the KB engine returned particularly low scores, this method of ranking the candidate recommendations is most fair to each engine and does not favor one engine over the others.

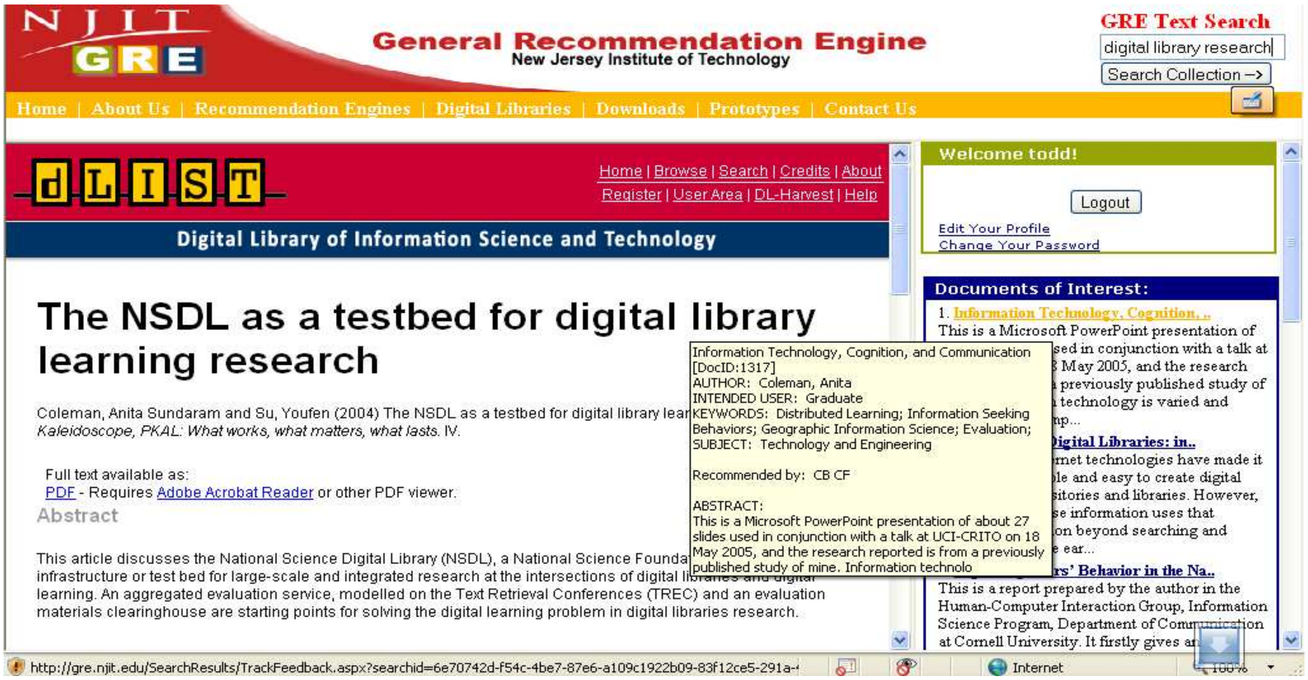


**Figure 1. Experimental System Screen Capture**

Before recommendations are provided, users first create a profile for the KB engine and then enter the keywords in the GRE Text Search box on the upper right to receive an initial list of documents. After the search results are returned and displayed on the left side of the screen, the user selects a potential relevant document, in this case "The NSDL as a testbed for digital library learning research," and then the document loads on the left side of the screen. The document identifier is then provided to GRE manager to query the CB, CF, and KB engines for recommendations for display in the Documents of Interest box; on mousing over the title in this box, additional information is provided about the document such as the author, abstract, intended user, and the recommendation engines that provided the document (CB and CF in this case). As the user browses around the site, the recommendation list changes depending on the currently viewed document.

## DESIGN OF THE USER STUDY

A user study was conducted to evaluate the performance of each individual engine and the text search. The subjects performed two different information seeking tasks, one to find five documents describing web spiders and a second to find five documents about fractals. After an initial pilot study containing eight subjects found CB, CF, and KB recommendation techniques were working in the anticipated way and experimental procedures were adequate, a full scale user study of sixty subjects drawn from the student body at a technical university was conducted. Subjects were rotated through the different tasks, so two subjects would carry out the web spiders task first and the next two would complete the fractals task with a total of thirty subjects completing each task. The subjects completed a consent form and questionnaire about their experience with search engines, then were shown a video tutorial of the experimental site, and finally completed a questionnaire about their experience.

## EXPERIMENTAL RESULTS

The personalized KB engine performance can now be compared to the performance of a text search engine and the CB and CF recommendation techniques. The best performing technique is not necessarily the one that provided the most documents but rather the one that provides the highest percentage of relevant documents to the user. Table 3 summarizes total browsed documents by all users and the total relevant documents selected as relevant by all sixty subjects. The performance of text search, CB, and CF techniques, defined as the ratio of relevant to total browsed documents by each engine, was compared with the performance of KB; significance levels are also provided.

| Task | Measure | Text Search | KB Vs Text Search Significance | CB | KB Vs CB Significance | CF | KB Vs CF Significance | KB |
|---|---|---|---|---|---|---|---|---|
| Web Spider | Total browsed documents | 48 | | 79 | | 88 | | 110 |
| | Selected relevant documents | 33 | | 55 | | 68 | | 90 |
| | Percentage relevant | 69% | 0.0917 | 70% | 0.0336 | 77% | 0.626 | 82% |
| Fractals | Total browsed documents | 48 | | 62 | | 72 | | 55 |
| | Selected relevant documents | 39 | | 47 | | 65 | | 50 |
| | Percentage relevant | 81% | 0.463 | 76% | 0.541 | 90% | 0.280 | 91% |
| Overall | Percentage relevant | 75% | 0.395 | 72% | 0.068 | 83% | 0.543 | 85% |

**Table 3. Relevant Documents Versus Total Documents**

The first row in each task is the number of documents browsed by all users, while the second row is the number of browsed documents that are relevant as judged by users. The KB engine performed best at personalizing the recommendation to the individual user's information need, with 82% of browsed documents being relevant in the web spider task and 91% relevant in the fractals task. While the users only found 55 relevant documents in the KB engine versus 72 relevant documents in the CF engine, the KB engine still was able to recommend a much higher percentage of accurate documents to the user. The CF engine also performed well with the CB and text search performing lowest by providing the lowest ratio of relevant documents to recommended documents. Even though the KB was not found to perform significantly better than CB, CF or text search, the KB performed comparable to other proven techniques and would have performed much better had six subjects identified the proper additional metadata selections in the web spider task profile and one subject identified the correct audience level for the fractals task due to unfamiliarity with the task description. As the CF and KB engines are personalized to the user's current search session, these were expected to outperform the CB and text search techniques.

The above analysis demonstrates that the KB outperformed the CB, CF, and Lucene text search engine, but this analysis is based on the individual's relevance decision. To determine the extent to which each technique was able to guide users to make correct relevance judgments, a list of all relevant and a second list of partially relevant documents in the collection were compiled by the authors and then compared to the judgments by subjects. The percentages are calculated by dividing the relevant or partially relevant documents as identified by the authors by the number of relevant documents selected by the subjects for each engine.

| Task | Measure | Text Search | CB | CF | KB |
|---|---|---|---|---|---|
| Fractals | Correct Relevance Judgments | 67% | 66% | 86% | 96% |
| | Document is relevant but not all metadata matches | 30% | 23% | 11% | 2% |
| | Incorrect relevance judgments | 3% | 11% | 3% | 2% |
| | Total | 100% | 100% | 100% | 100% |
| Web Spider | Correct Relevance Judgments | 91% | 93% | 97% | 99% |
| | Document is relevant but not all metadata matches | 0% | 0% | 0% | 0% |
| | Incorrect relevance judgments | 9% | 7% | 3% | 1% |
| | Total | 100% | 100% | 100% | 100% |

**Table 4. Accuracy Judgments by Users by Task**

Table 4 shows that all three engines and the text search performed much better in the web spider task versus the fractals task, as documents that describe web spiders were all college level and above in the technology category, while fractals documents spanned all audience levels and subject categories. Text search was expected to perform lowest as it does not consider the context of the user information need nor other documents that users have found relevant while KB should perform best. In both tasks, the KB engine was able to guide users to over 95% relevant documents to the user task while the text search with no personalization was able to only perform at 67% in one task and 91% in the second.

The KB engine outperformed the CB, CF, and text search in all areas of evaluation by not only supplying a high number of relevant documents but also guiding users to the most relevant documents in the collection. The CF engine was able to provide more relevant documents later in the study as more recent users were able to use the click history from earlier users to find documents. The personalization available by employing the KB technique is the clear winner to guide users to the best possible documents for the task and would have performed even better had all users completed the task profile correctly. The KB engine is effective and warrants further development to improve its performance.

## DISCUSSION AND CONCLUSION

All subjects in this experiment carried out only two tasks, not allowing them to develop a click history relating to their individual research interests. While requiring approximately three minutes on average to create the KB profile for a task that was new to users, this initial time cost was mitigated as subjects found more relevant documents faster than if the KB recommendations were excluded. Some subjects had difficulty completing the KB task profile correctly and judging the relevance of documents as they were not familiar with the information retrieval domain. However, as the document collection contains a few specialized domains, these two specific tasks were chosen as enough documents must be available to fulfill the experimental tasks and the analysis requires a small set of user tasks. The experimental site was seen by all subjects for the first time during this study and some subjects were not able to make full use of the provided recommendations due to their experience with other search engine interfaces such as Google. If subjects were given an extended period of time to use the system or even to customize the interface to appear similar to other familiar systems, the usage of the KB engine is expected to increase.

Unlike common digital libraries that contain domain specific collections, GRE provides "cross-collection" recommendations to users, allowing users of our system to explore other collections and documents that may not be available within a single collection but may be extremely relevant to the information search. In this way, users of a specific digital library may find resources in other collections that will be useful and, in addition, may even discover new collections that would be extremely helpful in future information seeking tasks.

Our design of the KB engine was found to consistently outperform the more common CB, CF, and text search techniques due to its much higher degree of personalization than that offered by other text search and recommendation engines. Subjects were able to use the KB engine to easily find documents relevant to their information-seeking task. A future study will require subjects to use the experimental system for a longer duration to determine the changing performance of these engines in a more dynamic environment with a much wider range of tasks. Another study could include users to fully customize the search system to meet their individual information seeking needs and styles rather than forcing all users to complete tasks by using the same system.

## ACKNOWLEDGEMENTS

Partial support for this research was provided by the United Parcel Service Foundation; the National Science Foundation under grants DUE-0226075, DUE-0434581 and DUE-0434998, and the Institute for Museum and Library Services under grant LG-02-04-0002-04.

## REFERENCES

1. Ahmad, M. and Ahmad, W. (1999) "Collecting User Access Patterns for Building User Profiles and Collaborative Filtering," *ACM IUI 99*, Redondo Beach CA USA.
2. Basu, C., Hirsh, H. and Cohen, W. (1998) "Recommendation as classification: Using social and content-based information in recommendation," *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, WI, July 1998, pages 714-720.
3. Breese, J, Heckerman, D, and Kadie, C. (1998) "Empirical analysis of predictive algorithms for collaborative filtering" *Proceedings of the Fourteenth Conference on Uncertainty in Information Intelligence*, Madison, WI.
4. Callan, J., Smeaton, M., Beaulieu, P., and Brusilovsky, M, (2003) "Personalisation and Recommender Systems in Digital Libraries." *Joint NSF-EU DELOS Working Group Report*.

5. Chaffee, J and Gauch, S. (2000). "Personal Ontologies for Web Navigation." *Proceedings of the Ninth International Conference on Information and Knowledge Management*.
6. Chung, I. and Hong, I. (2001). "The implementation of knowledge-based recommender system for electronic commerce using Java expert system library" *IEEE International Symposium on Industrial Electronics*, 12 – 16, Vol 3
7. Hanson, K. and Carlson, B. (2005). "Effective Access: Teachers use of digital resources in STEM teaching Research Report." Available at: http://www2.edc.org/GDI/effectiveaccess.htm
8. Herlocker, J., Konstan, J., Borchers, A. and Riedl, J. (1999) "An Algorithmic Framework for Performing Collaborative Filtering." *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval 1999*. pp. 230-237
9. Herlocker, J. and Konstan J. (2001). "Content-Independent, Task-Focused Recommendation," *IEEE Internet Computing*, 5, 6
10. Kotsiantis, S (2007). "Supervised Machine Learning: A Review of Classification Techniques." In *Informatica 31*, p249–268
11. Masullo, M. and Mack, R. (1996) "Roles for Digital Libraries in K-12 Education." *D-Lib magazine*. September 1996.
12. McCown, F., Bollen, J., and Nelson, M. (2005) "Evaluation of the NSDL and Google for Obtaining Pedagogical Resources." *Proceedings of the European Conference on Digital Libraries*.
13. McNee, S., Albert, I., Cosley, D., Gopalkrishnan, P., Lam, S., Rashid, A., Konstan, J., and Riedl, J. (2002). "On the Recommending of Citations for Research Papers." *Proceedings of the ACM 2002 Conference on Computer Supported Cooperative Work (CSCW 2002)*, New Orleans, LA, 2002, pp. 116-125.
14. Mobasher, B. (2004) "Web Usage Mining and Personalization," *Practical Handbook of Internet Computing*, Munindar P. Singh (ed.), CRC Press, 2005.
15. Mooney, R. J. and Roy, L. (1999) "Content-based book recommending using learning for text categorization", *SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation*.
16. Sarwar, B., Konstan, J., Borchers, A., Herlocker, J. Miller, B. and Riedl, J. (1998) "Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System." *Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work*, Seattle, WA, pp. 345-354.
17. Tsai, K., Chiu, T., Lee, M., and Wang, T. (2006) "A Learning Objects Recommendation Model based on the Preference and Ontological Approaches." *Proceedings of the Sixth IEEE international Conference on Advanced Learning Technologies (July 05 - 07, 2006). ICALT*. IEEE Computer Society, Washington, DC, 36-40.
18. Vucetic, S. and Obradovic, Z. (2000) "A regression-based approach for scaling-up personalized recommender systems in e-commerce." *ACM WebKDD 2000 Web Mining for E-Commerce Workshop*.
19. Zhang, L. and Im, I. (2002) "Recommender Systems: A Framework and Research Issues." *Americas Conference on Information Systems (AMCIS)*, Dallas TX.