Beyond Personalization 2005

A Workshop on the Next Stage of Recommender Systems Research

San Diego, January 9, 2005

In conjunction with the 2005 International Conference on
Intelligent User Interfaces (IUI 2005)

Edited by:
Mark van Setten
Sean McNee
Joseph Konstan

http://www.grouplens.org

http://www.telin.nl

http://www.multimedian.nl

Table of Contents

About the Workshop

This workshop intends to bring recommender systems researchers and practitioners together in order to discuss the current state of recommender systems research, both on existing and emerging research topics, and to determine how research in this area should proceed. We are at a pivotal point in recommender systems research where researchers are both looking inward at what recommender systems are and looking outward at where recommender systems can be applied, and the implications of applying them out 'in the wild.' This creates a unique opportunity to both reassess the current state of research and directions research is taking in the near and long term.

Background and Motivation
In the early days of recommender systems research, most research focused on recommender algorithms, such as collaborative filtering and case-based reasoning. Since then, research has gone off into various directions. Some researchers continued working on the algorithmic aspects of recommenders, including a move to hybrid and group recommenders; others have been researching the application of recommenders in specific domains; yet others focused on user interface aspects of recommender systems.

This has led to the current state in which recommender systems are mature enough to be applied in various adaptive applications and websites. They have been deployed on several large e-commerce websites, such as Amazon.com; they are being integrated into corporate document warehouses; and they are still the center of focus for several research groups around the world. Moreover, these systems are appearing in products and services used by people around the world, such as personalized television programming and Internet-broadcast radio stations, movie recommenders, and even dating services.
This workshop aims to answer questions raised both by researchers and practitioners in order to improve both recommender quality and use. Issues discussed at the workshop will have an effect on these systems—and more importantly, the users of these systems—worldwide.

Topics and Goals
This workshop will focus on the following four main topics:

1. Understanding and trusting recommender systems.
   Do users understand and trust the recommendations they receive from recommender systems, what kinds of information do recommenders need to provide to users to build trust, and how difficult is it to regain trust in a recommender if it is lost?
2. User interfaces for recommender systems.
   What are good ways to present recommendations to users, how do you integrate recommenders into the displays of existing information systems, and how can interfaces encourage users to provide ratings in order to 'close the loop' for recommendations, that is, how can you get users to consume the items recommended and then tell the system how good the recommendations are?
3. The future of recommendation algorithms and metrics.
   How can we generate better individual and group recommendations, develop new metrics and evaluation criteria for recommendations, and achieve cross-domain recommendations?
4. Social consequences and opportunities of recommenders.
   How do individuals and groups of people respond to recommendations, how can recommendations be integrated with online and real world communities, and in what ways do recommendations affect social organizations?

Intended Audience
The workshop is intended for both established researchers and practioners in the domain of recommender systems as well as for new researchers and students with interesting ideas on recommender systems and their future. Participants do not have to come from a specific application domain, as long as their research or ideas are on one of the main topics of the workshop.

Website
All papers and the results of the workshop are also available online at:
http://www.grouplens.org/beyond2005

Organization

Workshop Chairs

Mark van Setten
Telematica Instituut
P.O. Box 589
7500 AN Enschede
The Netherlands
E-mail: Mark.vanSetten@telin.nl

Sean M. McNee
GroupLens Research
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, MN, 55455 USA
E-mail: mcnee@cs.umn.edu

Joseph A. Konstan
GroupLens Research
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, MN, 55455 USA
E-mail: konstan@cs.umn.edu

Program Committee

Liliana Ardissono - University of Torino (Italy)
Jon Herlocker - Oregon State University (USA)
Anton Nijholt - University of Twente (The Netherlands)
Barry Smyth - University College Dublin and Changing Worlds (Ireland)
Loren Terveen - University of Minnesota (USA)

Additional Reviewers

Betsy van Dijk – University of Twente (The Netherlands)
Harry van Vliet – Telematica Instituut (The Netherlands)

Crossing the Rubicon for An Intelligent Advisor

Răzvan Andonie
Computer Science Department
Central Washington University, Ellensburg, USA
andonie@cwu.edu

J. Edward Russo
Johnson Graduate School of Management
Cornell University, Ithaca, USA
jer9@cornell.edu

Rishi Dean
Sloan School of Management
Massachusetts Institute of Technology, USA
rdean@mit.edu

ABSTRACT

Recommender systems (RS) are being used by an increasing number of e-commerce sites to help consumers find products to purchase. We define here the features that may characterize an "intelligent" RS, based on behavioral science, data mining, and computational intelligence concepts. We present our conclusions from building the WiseUncle Inc. RS, named Rubicon, and give its general description. Rather than being an advisor for a particular application, Rubicon is a generic RS, a platform for generating application specific advisors.

Keywords

Recommender systems, electronic commerce, user interface, user modeling

INTRODUCTION

E-commerce sites use RS to guide potential consumers through the buying process by providing customized information and product recommendations. Based on the customers' individual needs, values, and preferences, the goal of a RS is to find the "best" possible product from a large set of complex options. We shall only mention some online recommender systems that have been used, or are being considered for use: [4, 5, 3, 10]. There are several well-known e-commerce businesses that use, or have used, RS technology in their web sites: Amazon, Travelocity, BMW, MovieFinder, and Dell among them.

Although commercial RS have been available for several years now, we are still at the beginning of using RS on a large scale. In reality, sellers provide an RS to help improve the (long-term) business relationship. This goal gives rise to several desiderata that can be difficult to achieve. The RS should be flexible, scalable, multifunctional, adaptive, and able to solve complex search and decision problems.

The RS interface with the customer should be based on the same consumer psychology knowledge and strategies used in marketing. Behind this "visible" task, a RS can bring valuable information to marketers, making them improve their offer and products (customer profiling, marketing segmentation). For instance, RS can help businesses decide to whom to send a customized offer or promotion.

RS use knowledge to guide consumers through the often overwhelming task of locating suitable products. This knowledge may come from experts (e.g. marketing, product domain) or it can be "mined" knowledge learned from the behavior of other consumers. These two types of knowledge can be used not only during the recommendation process, but also to adaptively improve the system itself.

When optimizing the recommendation, it is possible that the system has to search in a huge admissible solution space for the "best" product. Solving such an optimization during the course of an Internet interaction creates a difficult problem, one that requires devising fast heuristic solutions.

Another knotty problem is related to the formal definition of the "best" recommendation. The optimality of the recommended option generally requires several criteria. The question is how to quantify the importance of such different attributes like color, shape, speed, price, etc?

These considerations and others make us define an *intelligent advisor (IA)* as a RS having the following features:

a.) During each interaction with a customer, it extracts knowledge from the customer that is used to build and update the corresponding customer profile. When the interaction is concluded, the system makes a valid recommendation.

b.) The IA saves the extracted knowledge and the customer

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

profile can be further "mined" for marketing-relevant knowledge.

c.) The IA - customer interface is based on the psychology of the consumer and the purchase decision process. Therefore, behavioral science techniques should create the fundamentals of an IA, in particular a customer dialog that embraces what/how people think, rather than forcing consumers to feed an optimization algorithm. Thus the IA divorces users from some of the complexity of their decisions.

d.) The IA should be able to improve its functionality by continually learning from its interactions with consumers.

e.) An IA should be robust in the face of data that are uncertain, noisy, sparse, or missing.

f.) An IA should be scalable and able to work in real-time, to meet the requirements of an Internet application.

g.) An IA should know how to draw multiobjective comparisons among products.

h.) An IA should be largely domain-independent, such that with minimum modification effort, one should be able to customize the same platform for other applications (e.g., selling computers, cars, financial services).

How far are current RS from an ideal IA? Some of the existing RS already incorporate some of these requirements. For instance, TalkMine uses the behavior of its users to adapt the knowledge stored in information sources [8]. However, most probably, none of the commercial available RS fulfills all requirements.

The idea of defining an IA came after several years of building a commercial RS, called Rubicon. Rather than being an advisor for a particular application, Rubicon is a generic RS, a platform for generating application-specific advisors. In this paper we look at the difficulties we faced when building Rubicon and the main concepts we used.

DIFFICULTIES IN BUILDING A RS

There are two categories of problems we faced when building our RS. The first is related to the system design, the second to the customer behavior.

Design and Integration

Incorporating complex behavioral data. There are many types of information that can be collected and used: customer knowledge (goals, needs and profile of the user), domain knowledge (product information and business rules specific to a particular vertical application), traffic logs, and expert knowledge. Using expert knowledge alone we can recommend "good" products. Using only customer knowledge, we can recommend products that were sold successfully in the past. The first strategy is much better at dealing with new products, whereas the latter one reflects only the customer experience.

Scalability and real-time performance. Scalability in recommender systems includes both very large problem sizes and real-time latency requirements. For instance, a recommender system connected to a large web site must produce each recommendation within a few tens of milliseconds while serving thousands of consumers simultaneously and searching through potentially billions of possible product configurations.

Noisy, missing, uncertain, and sparse data. The value of a RS lies in the fact that most customers have not deeply considered many of the available products, the product features, or their personal needs. This means that we must often deal with extremely sparse data, such as that resulting from a customer responding "I don't know", "I don't care", or "I don't want to answer this question".

Connecting recommenders to marketers. RS should be connected to the vendor's product database and the marketer's reporting systems. Only products that are currently in stock should be recommended, or products that can be configured in a feasible manner, both from engineering and logical perspectives. The highly volatile nature of real-world products and information systems creates the necessity of adequate database maintenance in the IA.

Domain independence. From the software engineering point of view, building a domain-independent RS platform can be done by separating the generic part from the domain specific knowledge modules.

User Experience

The customer-recommender interface is usually based on a series of interactive questions presented to the customer by the RS, accompanied by multiple-choice options for the customer to input their answers. In this case, a difficult problem is what strategy to follow when selecting questions to present. An intelligent dialog should be personalized. Some randomness should be used when selecting the questions [7]. Too much randomness leads to excessive customer effort, but a small amount of randomness may help to extend the space over which the recommender understands the customer's interests and ensures that all questions are occasionally presented to customers. A reasonable strategy for selecting information from customers is to minimize customer effort while still being able to make accurate predictions [7]. However, this strategy is quite simplistic, and a behavioral science-based investigation is necessary here. What we should measure is not customer effort (measured in the duration of the dialog), but customer satisfaction. Satisfaction quantification results from longer-term statistics on usage and surveying customers.

During the conversation, an IA adopts a five-stage process, described by [2]: **i)** Opening; **ii)** Utilitarian Needs; **iii)** Hedonic Preferences; **iv)** Optional Features / Add-ons; and **v)** Endgame.

Stage **i** frames the buyer (e.g., knowledge of the product category and extent of product search to date) and the main product characteristics (e.g., a desktop PC versus a laptop).

Stages **ii** and **iii** encompass, respectively, the utilitarian and hedonic or emotional needs. The former include the functional uses of the product, such as an automobile's seating capacity or environmental friendliness. Stage **iii**'s hedonic needs, like the image of a car's body style and brand name, are often harder for a buyer to express. Needless to say, extracting such knowledge can be a substantial behavioral challenge in itself. Stage **iv** captures the remaining, minor product specifications, like an automobile's audio speakers or aspects of its interior. The final stage covers such external elements as a PC's warranty or the local availability of reliable repair service for an automobile. These five stages are sufficient to structure the process of a purchase decision for all complex products.

How can a trusted recommender validate itself to consumers through a web client? The following factors contribute to the success of such conversations in Internet-mediated dialogs [9].

The benefits of the conversation should exceed its costs. People use information only if it is perceived as adding benefits or as reducing costs. If (expected) costs exceed (expected) benefits at any point, there is a clear risk of the customer terminating the dialog.

Credibility and trust. The information and advice must be credible, and the source must be trustworthy. An Internet-delivered RS cannot provide the face-to-face cues of trustworthiness that a human can. However, although a RS may have no initial reputation for trust (based on past experience), such an image can be built over time by personal usage, word-of-mouth recommendations, or public endorsements (e.g., by consumer-oriented magazines' endorsement of the system's knowledge and disinterestedness). One alternative is to add a confidence metric, and this has the potential to improve user satisfaction and alter user behavior in the RS [6]. A second alternative is to make the RS adaptive. This would reduce the risk of manipulation: users can detect systems that manipulate prediction and, this has a negative impact on their trust [1].

Intelligence and customization. First, the advisor must know what kinds of information people can validly provide and how to successfully extract that information from buyers. Consumers can usually say what they need or want the product to do and can articulate such personal preferences as style and color. However, they may have difficulty specifying the product features that meet those needs. Second, based on whatever can be learned from the customer, the problem of identifying the optimal product must be solved. Thus, the advisor must first extract the customer's needs and then build an inferential bridge from those needs to the most suitable product.

Control. Customers should be able to request additional or explanatory information. Or as the conversation proceeds, the customer may learn something that requires returning to an earlier point in the dialog and changing a preference stated there. Buyers who feel impatient should be able to request a recommendation at any time, even before the advisor would normally feel comfortable providing one. Finally, the buyer might even like to suspend the conversation and return later. More control in any situation is empowering, and more so in situations where control is expected. Providing satisfactory conversational control is a special challenge to RS.

Feedback. Specific feedback might include (a) how much progress has been made toward identifying the best product, and (b) how much longer the conversation is expected to take. Whatever specific feedback options are provided, however, users do not want to receive feedback only after they have answered every question (as they must in many static surveys).

SYSTEM DESCRIPTION

Rubicon is a generic domain-independent advisor, recommending products from an existing set. Each product is configurable, meaning it is comprised of several components, which may each be described in turn by several attributes. Building a RS depends largely on the knowledge representation model, and we chose a computational intelligence framework. Our RS is a classifier that "learns" to make good recommendations. This classifier is an expert system, able to explicitly expose its acquired knowledge. The main characteristics of Rubicon are:

- The inferential process from the customer's needs to the best product is constructed in two stages, called *Bridges*, one from needs to product attributes, and the second from attributes to the products themselves.
- It can easily be customized for different applications since the interface to the application-specific knowledge domain is separated from the main system.
- The front-end dialog is dynamic and changes based on user responses. It elicits information from users to determine a location in a needs space which is then used to find optimal (sub-optimal) solutions in a products space.
- It accepts imprecise input from users.
- It provides a justification for all recommendations.
- Reversibility: The system can reverse the decision process from effect to cause. This allows forecasting the adoption of new products or services using real customer decision data.

The Rubicon system diagram (Fig. 1) shows the following main modules.

Conversation Engine (CE)

The CE is responsible for dialog management, presenting questions to the user and processing the resulting responses

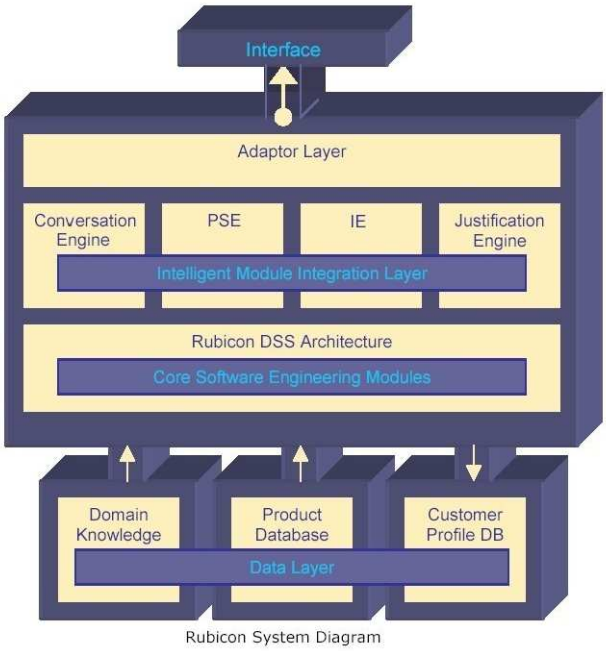


Figure 1: Rubicon High Level System Description

sent to it by the user. Questions and their associated responses are processed to accomplish the following two results: *i)* Propagate the knowledge gained from a response to the subsequent inference mechanisms and *ii)* Determine the next question to pose to the user.

In doing so, the dialog management occurs subject to the following constraints:

- Presents the appropriate questions for the system to confidently determine an intelligent, personalized recommendation.
- Presents questions conforming to users' expectation of a real dialog with respect to flow, organization, and coherence.
- Minimizes the number of questions presented.
- Scalable addition/ subtraction/ modification of questions.
- Allows users and administrators to reproduce particular dialogs.
- Uses proper constructs for further data mining.

From a computational standpoint a rule-based expert system is used to implement the CE's dialog management process. Questions and responses are linked by sets of predetermined rules, and a number of other intermediary constructs. In this way, the questions, responses, and rules can be specified, along with goals (i.e., knowledge to be gathered) independently of knowing the dialog flow in advance. The system at runtime determines, based on the behavioral and informational goals, which question to present next to the user. When all appropriate questions have been presented, the conversation is determined to be complete. However, the user may intervene at any time to ask for the system's current best recommendation based on the information provided thus far.

Inference Engine (IE)

The purpose of the IE is to map the user's profile of needs (the output of the CE) to the attributes necessary to comprise the appropriate recommendation. Given a set of responses resulting from the dialog, the IE can indicate a set of recommendations, ordered by the degree of their preference. These recommendations are not concrete (physical) product recommendations yet, but a mapping from the user needs space to the space of attributes, yielding generic descriptions of the product, like "RAM Amount" (e.g., standard, large, maximum) and "Network Card type". Collectively this inference is called the *First Bridge*.

The IE is taught by a human expert. However, it can learn incrementally as well: new teaching examples can be added without restarting the teaching process from the beginning. Conditional rules can be extracted to describe the behavior of the IE and justify recommendations, market research and performance improvement. The IE is stable under noisy inputs and user uncertainty. Such "noise" may be produced by "I don't know" answers, or by contradictory answers in the dialog.

To implement the IE, a fuzzy neural net architecture is used, trained to represent the expert knowledge of a particular product domain. For instance, in the case of a personal computer RS, experts develop training patterns to represent the varying needs profiles of customers along with their corresponding feature sets for a recommended PC. The inference process is fast, online.

Product Search Engine (PSE)

The PSE is the *Second Bridge*, a mapping from the space of attributes to the space of (physical) products. It is an optimization module interfacing with the retailer's product database to select the best, valid product configurations that match the criteria specified by the user, such as the minimum cost, the maximum likelihood of success, or a number of other simultaneous criteria. The inputs to the PSE are the levels of the attributes (the output of the IE), the configuration constraints (i.e., incompatibilities among components), and a user's criteria for optimization (e.g., a desired price point). These criteria for the algorithm can be set by the IE and CE and are, therefore, uniquely tailored to a given user. The PSE can sort though billions of options in real time, allowing searches to be completed online. The products with the highest degree of fit are passed to the Justification Engine for further processing.

The response to a question is subsequently used to provide

more information that adds to Rubicon's knowledge base during a user-experience. This, in turn, leads to a recalculation and optimal selection of the next most appropriate question. This question-response model continues until Rubicon is either asked, or has sufficient confidence, to make a recommendation.

The PSE navigates a vast search space, taking into account different optimization criteria. We used a genetic algorithm approach for this (NP-complete) optimization problem. In the initial phases, the PSE operates on abstractions of the real world, and then through an adaptor layer translates these abstractions into concrete items. This information is capable of being read and processed at runtime. The PSE remains independent of constant updating of the "real world" items. This adaptor level is implemented as an XML data bridge.

Justification Engine (JE)

Recommendations are run through the JE to provide a plain English explanation of why the system has provided a specific recommendation. This justification is delivered in the same vernacular as the dialog, personalized to the user, and is present to facilitate user understanding and adoption of the recommendation. The JE takes the set of If-Then rules from the IE and the set of recommended configurations from the PSE and develops a rationale for selecting each product. The value of the JE is that it creates confidence in the recommendation.

Rubicon is implemented using a complimentary modular software approach that encapsulates the individual computational blocks, as well as the necessary software architecture emphasizing a stable and reusable model that is compliant with the J2EE technology standard. The user interface is HTML 4.0 compliant, utilizing DHTML, and combining client-side scripting and styles. It is implemented using XML/XSLT, built for 4th generation web browsers, and rendered via the adapter layer using JSP/servlets.

PRELIMINARY TESTS

Although still under development, Rubicon was sufficiently developed to be submitted to usability testing by two major PC manufacturers. Each test involved about a dozen users and compared three RS. One was the manufacturer's current online RS, one was an attractive competitor, while the third was Rubicon. The results made available revealed that Rubicon was judged clearly superior in both tests. For instance, in one test, when asked which of the three RS the user would "be most likely to use again", nine of eleven respondents chose Rubicon.

Rubicon was tested online by a webhosting services provider. Of 2200 online users who began a conversation, 83% completed it to the point of receiving a recommendation (which was the only result made available to us). This was judged by the host company to be an extraordinary high completion (i.e., non-abandonment) rate.

CONCLUSIONS

Is Rubicon "intelligent"? According to our IA criteria from the Introduction, yes. Furthermore, we believe that these criteria may be a base for classifying RS. We have not presented here other modules of Rubicon, used for prediction, customer profiling, and marketing segmentation, since we have tried to focus on the core system. It was a challenging task to build Rubicon, especially because of its generic character. Making the system largely independent of a specific e-commerce application required greater complexity and abstraction. But do we really need a generic RS? From a user perspective this may be a non-issue. However, for the RS designer and software engineer this is a critical requirement. We should think not only in terms of how to use a RS, but also how to build it and how to adapt it fast for very different application areas.

REFERENCES

1. D. Cosley, S. Lam, I. Albert, J. Konstan, and J. Riedl. Is seeing believing? how recommender systems influence users' opinions. In *Proceedings of CHI 2003 Conference on Human Factors in Computing Systems*, pages 585–592, Fort Lauderdale, FL, 2003.

2. A. Horowitz and J. Russo. Modeling new car customer-salesperson interaction for a knowledge-based system. *Advances in Consumer Research*, 16:392–398, 1989.

3. S.-Y. Hwang, W.-C. Hsiung, and W.-S. Yang. A prototype www literature recommendation system for digital libraries. *Online Information Review*, 27:169–182, 2003.

4. W. Lin, S. Alvarez, and C. Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data Mining and Knowledge Discovery*, 6:83–105, 2002.

5. J. Liu and J. You. Smart shopper: An agent-based web-mining approach to Internet shopping. *IEEE Transactions on Fuzzy Systems*, 11:226–237, 2003.

6. S. McNee, S. Lam, C. Guetzlaff, J. Konstan, and J. Riedl. Confidence displays and training in recommender systems. In *Proceedings of INTERACT '03 IFIP TC13 International Conference on Human-Computer Interaction*, pages 176–183, 2003.

7. A. M. Rashid, I. Albert, D. Cosley, S. Lam, S. McNee, J. Konstan, and J. Riedl. Getting to know you: Learning new user preferences in recommender systems. In *Proceedings of the 2002 International Conference on Intelligent User Interfaces*, pages 127–134, San Francisco, 2002.

8. L. M. Rocha. Talkmine: a soft computing approach to adaptive knowledge recommendation. In V. Loia and S. Sessa, editors, *Soft Computing Agents: New Trends for Designing Autonomous Systems - Studies in*

Fuzziness and Soft Computing, pages 89–116. Physica-Verlag, Springer, 2001.

9. J. E. Russo. Aiding purchase decisions on the Internet. In *Proceedings of the Winter 2002 SSGRR (Scuola Superiore Guglielmo Reiss Romoli) International Conference on Advances in Infrastructure for Electronic Business, Education, Science, and Medicine on the Internet*, L'Aquila, Italy, 2002.
10. H.-W. Tung and V.-W. Soo. A personalized restaurant recommender agent for mobile e-service. In *Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)*, pages 259–262, 2004.

APPENDIX: THE USER EXPERIENCE WITH THE RUBICON COMPUTER ADVISOR

Instantiated as a computer advisor, Rubicon shows (Fig. 2) its first screen to a customer. Buyer control is offered, inter alia, by the "Why Ask" button, which pops up a justification for the current question. This feedback also exhibits respect for the buyer's right to know why their time and effort are being spent answering this particular question.

Figure 2: Getting acquainted

Ten questions into the conversation, just as it moves to Stage 2, the screen from Fig. 3 appears.

The Progress box on the right now shows a current recommendation in the form of the best type of computer for the particular buyer and the two most suitable alternative types. For each of the three types, a measure of fit to the buyer's needs is displayed. This measure should increase as more needs are elicited through further questioning and the recommended product is further customized to those needs. Below the recommendation is an announcement about the number of facts known, either directly or inferred. And below that is an indicator of how far through the conversation the system expects the buyer is. The measure is not time, but the number of questions already asked and the expected number to be asked before the conversation is finished.

After this screen the next question is "Do you have any problems with your current computer?" If the buyer answers

Figure 3: Extract the customer's needs

"Yes" and identifies "Too Slow" as the only problem, then the screen shown in Fig. 4 appears.

Figure 4: Continue extracting the needs

Note that the fit of the Hobbyist PC to the buyer's needs has moved from 40% to 53% and the number of known facts from 14 to 21. Both changes provide feedback and show the benefit of participating in the dialog, as it progresses toward the best recommendation. If the conversation reaches its natural conclusion (i.e., is not terminated prematurely by the buyer), three products are recommended, in order. This is another illustration of buyer control. People prefer to make the final choice themselves from several options, although they also want to know the advisor's ranking. A demo version of Rubicon can be found at www.wiseuncle.com.

Explaining Recommendations: Satisfaction vs. Promotion

Mustafa Bilgic
Computer Science Dept.
University of Maryland at College Park
College Park, MD 20742
mbilgic@cs.umd.edu

Raymond J. Mooney
Computer Sciences Dept.
University of Texas at Austin
Austin, TX 78712-0023
mooney@cs.utexas.edu

ABSTRACT

Recommender systems have become a popular technique for helping users select desirable books, movies, music and other items. Most research in the area has focused on developing and evaluating algorithms for efficiently producing accurate recommendations. However, the ability to effectively explain its recommendations to users is another important aspect of a recommender system. The only previous investigation of methods for explaining recommendations showed that certain styles of explanations were effective at convincing users to adopt recommendations (i.e. promotion) but failed to show that explanations actually helped users make more accurate decisions (i.e. satisfaction). We present two new methods for explaining recommendations of content-based and/or collaborative systems and experimentally show that they actually improve user's estimation of item quality.

Introduction

The use of personalized recommender systems to aid users' selection of reading material, music, and movies is becoming increasingly popular and wide-spread. Most of the research in recommender systems has focused on efficient and accurate algorithms for computing recommendations using methods such as collaborative filtering [4, 5], content-based classifier induction [9, 8], and hybrids of these two techniques [1, 7]. However, in order for users to benefit, they must trust the system's recommendations and accept them. A system's ability to explain its recommendations in a way that makes its reasoning more transparent can contribute significantly to users' acceptance of its suggestions. In the development of expert systems for medicine and other tasks, systems' ability to explain their reasoning has been found to be critical to users' acceptance of their decisions [12].

Several recommender systems provide explanations for their suggestions in the form of similar items the user has rated highly, like Amazon, or keywords describing the item that caused it to be recommended [8, 2]. However, Herlocker et al. [6] provide the only systematic study of explanation methods for recommenders. Their experimental results showed that certain styles of explanation for collaborative filtering increased the likelihood that the user would adopt the system's recommendations. However, they were unable to demonstrate that any style of explanation actually increased users' satisfaction with items that they eventually chose.

Arguably, the most important contribution of explanations is not to convince users to adopt recommendations (promotion), but to allow them to make more informed and accurate decisions about which recommendations to utilize (satisfaction). If users are convinced to accept recommendations that are subsequently found to be lacking, their confidence in the system will rapidly deteriorate. A good explanation is one which accurately illuminates the reasons behind a recommendation and allows users to correctly differentiate between sound proposals and inadequately justified selections.

This paper evaluates three different approaches to explaining recommendations according to how well they allow users to accurately predict their true opinion of an item. The results indicate that the *neighbor style* explanations recommended by [6] based on their promotion ability perform poorly, while the *keyword style* and *influence style* explanations that we introduce perform much better.

Methods for Recommender Systems

Recommender systems suggest information sources and products to users based on learning from examples of their likes and dislikes. A typical recommender system has three steps: *i.*) Users provide examples of their tastes. These can be explicit, like ratings of specific items, or implicit, like URLs simply visited by the user [10]; *ii.*) These examples are used to compute a *user profile*, a representation of the user's likes and dislikes; *iii.*) The system computes recommendations using these *user profiles*.

Two of the traditional approaches to building a user profile and computing recommendations are collaborative filtering (CF) and content-based (CB) recommendation. Hybrid systems that integrate these two different approaches have also been developed.

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

CF systems recommend items by matching a user's tastes to those of other users of the system. In the nearest-neighbor model [5], the *user profiles* are user-item ratings matrices. Recommendations are computed by first finding *neighbors*, similar users whose ratings correlate highly with those of the active user, and then predicting ratings for the items that the active user has not rated but the *neighbors* have rated using the *user profiles* and the correlation coefficients.

CB systems recommend items based on items' content rather than other users' ratings. The *user profiles* consist of concept descriptions produced by a machine-learning algorithm such as naive Bayes using a "bag of words" description of the items [8, 9]. Recommendations are computed based on predictions of these models which classify items as "good" or "bad" based on a feature-based description of their content.

Both CF and CB systems have strengths and weaknesses that come from exploiting very different sources of information. Consequently, a variety of different methods for integrating these two different approaches have recently been developed. Some of these hybrid methods use other users' ratings as additional features in a fundamentally content-based approach [1]. Others use content-based methods to create *filterbots* that produce additional data for "pseudo-users" that are combined with real users' data using CF methods [11]. Still others use content-based predictions to "fill out" the sparse user-item ratings matrix in order to allow CF techniques to produce more accurate recommendations [7]. A survey of hybrid recommenders can be found at [3].

Our Recommender System

We have previously developed a recommender system called LIBRA (Learning Intelligent Book Recommending Agent) [8]. The current version employs a hybrid approach we developed called *Content Boosted Collaborative Filtering* (CBCF) [7]. The complete system consists of three components. The first component is the Content Based Ranker that ranks books according to the degree of the match between their content and the *active user's* content-based profile. The second component is the Rating Translator that assigns ratings to the books based on their rankings. The third component is the Collaborative Filterer, which constructs final recommendations using an enhanced user-item ratings matrix.

LIBRA was originally developed as a purely content-based system [8] and has a database of approximately 40,000 books. Content descriptions are stored in a semi-structured representation with Author, Title, Description, Subject, Related Authors, and Related Titles. Each slot contains a bag of words, i.e. an unordered set of words and their frequencies. These data were collected in 1999 by crawling Amazon. Once the user rates a set of training books, the Content Based Ranker composes a *user profile* using a bag-of-words Naive Bayesian text classifier. The user profile consists of a table that has three columns: a slot column, a column for the token in that slot, and the strength column. The strength for a token t in a slot s is: $\frac{P(t|c_l,s)}{P(t|c_d,s)}$ where c_l is the category of *likes*, and c_d is the category of *dislikes*. A score for a test item is then computed by multiplying the strengths of each token t in slot s of the book. Lastly, the books are ranked based on their scores. This gives us the "Ranked Items" vector.

One of the main problems with CF methods is that the user-item ratings matrix upon which predictions are based is very sparse since any individual user rates only a small fraction of the available items. The basic idea of CBCF is to use content-based predictions to "fill out" the user-item ratings matrix. In [7], a 6-way CB classifier was used to predict integer ratings in the range 0–5. However, a 6-way classifier is less accurate than the 2-way (like vs. dislike) classifier originally used in LIBRA. Here, we use a Rating Translator as a bridge between the Content Based Ranker and the Collaborative Filterer.

The Rating Translator converts rankings into ratings by looking at the rating pattern of the user. However, since the rating pattern of a user usually tends to be skewed towards positive ratings, these data are first smoothed using a source of unskewed data: the rating pattern of several users who rated randomly selected items (Table 5 in [8]).

Once the user-item ratings matrix is filled-out using content-based predictions, we use a version of the CF method recommended in [5]. The system first computes correlations between the *active user* and other users of the system. The n users with the highest correlations are chosen as the *neighbors*. Predictions are computed using the *neighbors'* ratings for the test examples. Finally, the test items are sorted based on their predicted ratings and the top items are presented to the user as recommendations.

The Explanation Systems

A variety of recommender systems are now available. Some are developed for research purposes such as GroupLens [10], and some are in commercial use such as Amazon and NetFlix. Although a few of these provide some form of explanation for their recommendations, most are black boxes with respect to why they recommend a specific item [6]. Thus, the users' only way to assess the quality of a recommendation is to try the item, e.g. read the book or watch the movie. However, since users use recommender systems to reduce the time they spend exploring items, it is unlikely they will try an item without trusting that it is worth the effort. Herlocker et al. have shown that explanation systems increase the acceptance of collaborative filtering systems [6].

The effectiveness of an explanation system can be measured using two fundamentally different approaches: the *promotion* approach and the *satisfaction* approach. For the *promotion* approach, the best explanation is the one that is most successful at convincing the user to adopt an item. For the *satisfaction* approach, the best explanation is the one that lets the users assess the quality of the item the best.

Unfortunately, there is little existing research on explaining

recommender systems. The only detailed study is that of Herlocker et al. [6] in which twenty-one different styles of explanations where compared. The title of a recommended item was removed in order to prevent any bias it might cause, and the user was asked to rate a recommended item by just looking at its explanation. Herlocker et al. generally present explanation systems that produce the highest mean rating as the best. We believe that *satisfaction* is more important than *promotion*. If the users are satisfied with their selections in the end, they will develop trust in the system and continue to use it. Although in a second study in the same paper, Herlocker et al. did examine the effect of explanation on "filtering performance," they failed to find any consistent effect. Consequently, we explore how well an explanation system helps the user accurately estimate their actual opinion of an item.

We have used three explanation systems in our study: *keyword style explanation* (KSE), *neighbor style explanation* (NSE), and *influence style explanation* (ISE). Two factors played a role in choosing these three explanation styles. One factor is the type of information required, i.e. content and/or collaborative. We included KSE for systems that are partly or purely content-based, and NSE for systems that are partly or purely collaborative. ISE is not dependent on the recommendation method as described below. The second factor that affected our selection of these styles is that we wanted to test how KSE and ISE perform compared to NSE, which was the best performing explanation method (from the standpoint of promotion) in Herlocker et al. 's study.

Keyword Style Explanation (KSE)

Once a user is provided a recommendation, he is usually eager to learn "What is it about the item that speaks to my interests?" KSE is an approach to explaining content-based recommendations that was included in the original version of LIBRA. KSE analyzes the content of a recommended item and finds the strongest matches with the content in the user's profile. In LIBRA, the words are matched against the table of feature strengths in the user profile described above. For each token t occurring c times in slot s of the item's description, a strength of $c * strength(t)$ is assigned, where $strength(t)$ is retrieved from the user-profile table. Then, the tokens are sorted by strength and the first twenty entries are displayed to the user. An example is presented in Figure 1. This approach effectively presents the aspects of the item's content that were most responsible for the item being highly ranked by the system's underlying naive-Bayesian classifier.

If the user wonders where a particular keyword came from, he can click on the *explain* column, which will take him to yet another table that shows in which training examples that word occurred and how many times. Only positively rated training examples are included in the table. An example of such a table is presented in Figure 2. This approach effectively presents which user-rated training examples where responsible for this keyword having its high strength.

Slot	Word	Count	Strength	Explain
DESCRIPTION	HEART	2	94.14	Explain
DESCRIPTION	BEAUTIFUL	1	17.07	Explain
DESCRIPTION	MOTHER	3	11.55	Explain
DESCRIPTION	READ	14	10.63	Explain
DESCRIPTION	STORY	16	9.12	Explain

Figure 1: The Keyword Style Explanation

Title	Author	Rating	Count
Hunchback of Notre Dame	Victor Hugo, Walter J. Cobb,	10	11
Till We Have Faces : A Myth Retold	C. S. Lewis, Fritz Eichenberg,	10	10
The Picture of Dorian Gray	Oscar Wilde, Isobel Murray,	8	5

Figure 2: Explanation of Which Positively-Rated Books Have a Keyword

For more information on LIBRA's KSE method, see [8]. Billsus and Pazzani's news recommender provides similar explanations [2].

Neighbor Style Explanation (NSE)

If the recommender system has a collaborative component, then a user may wonder how other similar users rated a recommended item. NSE is designed to answer this question by compiling a chart that shows how the *active user's* CF *neighbors* rated the recommended item. To compute the chart, the *neighbors'* ratings for the recommended item are grouped into three broad categories: Bad (ratings 1 and 2), Neutral (rating 3), and Good (ratings 4 and 5). A bar chart is plotted and presented, as shown in Figure 3. NSE was tested

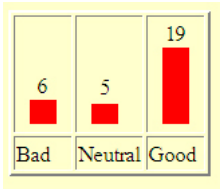


Figure 3: Explanation Showing Ratings of a User's Neighbors

along with twenty other explanation systems by Herlocker et al. [6] and performed the best from a *promotion* perspective. Grouping the rating into 3 coarse categories was found to be more effective than using a histogram with all 5 original ratings levels.

Influence Style Explanation (ISE)

ISE presents to the user a table of those training examples (which the user has already explicitly or implicitly rated) that had the most impact on the system's decision to recommend a given item. Amazon and NetFlix have a similar style of explanation, however it is unclear how they actually select the explanatory training items. LIBRA presents a table of

training books that had the most impact on its recommendation. Each row in the table has three entries: the book that the *active user* rated, the rating they gave the book, and the *influence* of that book on this recommendation. An example of such an explanation is shown in Figure 4.

BOOK	YOUR RATING Out of 5	INFLUENCE Out of 100
Of Mice and Men	4	54
1984	4	50
Till We Have Faces : A Myth Retold	5	50
Crime and Punishment	4	46
The Gambler	5	11

Figure 4: Influence Style Explanation

The ideal way to compute influences is to remove the book whose influence is being computed from the training set, recompute the recommendation score for each of the test items, and measure the resulting difference in the score of the recommended book. Therefore, unlike KSE or NSE, ISE is completely independent of the underlying recommendation algorithm. For purely collaborative or purely content based approaches, removing a training example and re-scoring the test examples can be done fairly efficiently. However, for the full CBCF algorithm currently used by LIBRA, this would require recomputing every single user's content-based user-profile and re-scoring every item for every user to update the "filled in" user-item matrix. Doing this to compute the influence of every training example is infeasible for a real-time explanation system.

To compute the influences efficiently, we compute two influences, the content influence and the collaborative influence, separately, rescale both and then average the two. The content influence of an item on the recommendation is computed by looking at the difference in the score of the recommendation computed by training the Bayesian classifier with and without the item. The collaborative influence is computed similarly: the correlation constants and predictions are computed with and without the item; the difference in the prediction for the recommended item is the collaborative influence. So that the users can easily interpret the results, we wanted the final influence to be in a fixed range [-100, 100]. Since the ranges for content influences and collaborative influences were different (content influence is a difference of log probability ratios and collaborative influence is a difference in predicted ratings), we re-scale them separately to a common range, [-100, 100], and then we compute the final influence by averaging the two. We sort the table using this final influence and present all positive influences to the user.

Experimental Methodology and Results

Methodology

To evaluate these three forms of explanation, we designed a user study in which people filled out an online survey. The ideal way to implement a survey to measure satisfaction is:

1. Get sample ratings from the user.
2. Compute a recommendation r.
3. For each explanation system e
3.1 Present r to the user with e's explanation.
3.2 Ask the user to rate r
4. Ask the user to try r and then rate it again.

If we accept that a good explanation lets the user accurately assess the quality of the item, the explanation system that minimizes the difference between the ratings provided in steps 3.2 and 4 is best. In step 1, we ask the *active user* to provide LIBRA with ratings for at least three items, ranging from 1 (dislikes) to 5 (likes), so that LIBRA can provide him a decent recommendation along with some meaningful explanations. We remove the title and author of the book in the step 3 because we do not want the user to be influenced by it. The ratings in step 3.2 are based solely on the information provided in the current explanation. To avoid biasing the user, we tell him that each explanation is for a different book (since the explanations present very different information, the user has no way of knowing they are actually for the same item.) Moreover, we randomize the order of the explanation systems used in each run to minimize the effect of seeing one explanation before another. Since running this experiment would be very time consuming should we asked the users to read the books recommended to them, we slightly modified step 4. Instead of reading the book, the *active user* is asked to read the Amazon pages describing the book and make a more informed rating based on all of this information.

We hypothesized that: 1. NSE will cause the users to overestimate the rating of an item. 2. KSE and ISE will allow users to accurately estimate ratings. 3. Ratings provided at step 3.2 and 4 should be positively correlated, with ISE and KSE correlating with the final rating better than NSE.

We believed that NSE would cause overestimation since the presented histograms are always highly skewed towards the top ratings since otherwise the book would not have been recommended. We believed that ISE and KSE would give better correlations since they do not suffer from this problem and they present additional information about this or similar books that we believed was more useful.

Results

Thirty-four subjects were recruited to fill out the online survey, most were students in various departments at the University of Texas at Austin. Since the system allowed the users to repeat the process with more than one recommendation, we were able to collect data on 53 recommendations. We use the following definitions in the rest of the paper. *Explanation-ratings* are the ratings given to an item by the users in step 3.2 by just looking at the explanation of the recommendation. *Actual-ratings* are the ratings that users give to an item in step 4 after reading detailed information about the book.

Since LIBRA tries to compute good recommendations, we expect both *explanation-ratings* and *actual-ratings* to be high. As can be seen from the Table 1, the mean ratings are pretty high, at least 3.75.

Table 1: Means and Std Deviations of Ratings

Type	μ	σ
Actual	3.75	1.02
ISE	3.75	1.07
KSE	3.75	0.98
NSE	4.49	0.64

We expect to have approximately normal distributions for the differences between the *explanation-ratings* and the *actual-ratings*. The histograms of the differences are displayed in Figure 5. The means of the differences can be seen in Table 2.

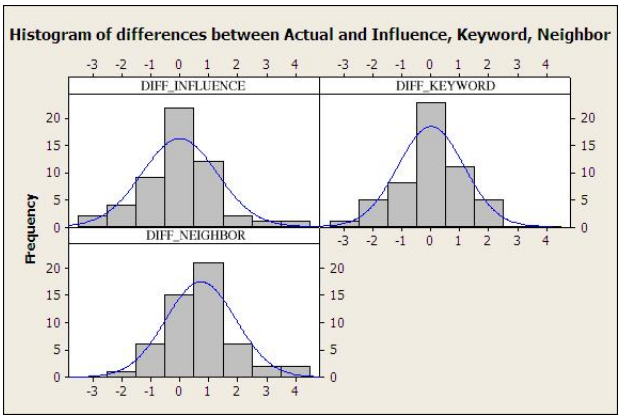


Figure 5: Histograms of Differences Between Explanation and Actual Ratings

Table 2: Means, Std Deviations, and Confidence Intervals of Differences

Type	μ	σ	**95% Conf. Int.**
ISE	0.00	1.30	(-0.36, 0.36)
KSE	0.00	1.14	(-0.32, 0.32)
NSE	0.74	1.21	(0.40, 1.07)

According to the *satisfaction* approach, the best explanation is the one that allows users to best approximate the *actual-rating*. That is, the distribution of *(explanation-ratings − actual-ratings)* for a good explanation should be centered around 0. Thus, the explanation whose μ_d (the mean of the difference between *explanation-rating* and *actual-rating*) is closest to 0 and that has the smallest standard deviation σ_d in Table 2 is a candidate for being the best explanation. KSE wins with $\mu_d = 0.00$ and $\sigma_d = 1.14$. When we look at the confidence intervals, we see that both KSE and ISE are very close. This table also shows that, with high probability, NSE causes the user to overestimate the *actual-rating* by 0.74 on average. Considering that the mean for *actual-ratings* is 3.75, and that the highest rating is 5.00, a 0.74 overestimate is a significant overestimation. This table supports both Hypotheses 1 and 2.

We have also run paired t-tests to find out whether these differences were likely to be due to chance only. The null hypothesis we used for all three types of explanations is $H_0(\mu_d = 0)$. Since we did not have prior estimates on whether KSE and ISE would cause the user to overestimate or underestimate should they estimate wrong, the alternative hypothesis for these explanation systems is $H_a(\mu_d \neq 0)$. However, since we postulated that the NSE would cause the user to overestimate the *actual-ratings*, the alternative hypothesis for NSE is $H_a(\mu_d > 0)$. The results in Table 3 clearly show that we can reject the null hypothesis for NSE, because the probability of having $\mu_d = 0$ is 0.00. (i.e. $P = 0.00$). So, we accept the alternative hypothesis for NSE. For ISE and KSE on the other hand, we cannot reject the null hypothesis, because $P = 1.00$. Thereby, the t-tests justify Hypothesis 1.

Table 3: t-tests

	Hypotheses	**P**
ISE	$H_0(\mu_d = 0), H_a(\mu_d \neq 0)$	1.00
KSE	$H_0(\mu_d = 0), H_a(\mu_d \neq 0)$	1.00
NSE	$H_0(\mu_d = 0), H_a(\mu_d > 0)$	0.00

One other thing that needs to be noted is that the means themselves might be misleading. Consider the following scenario. Assume that we have a new style of explanation called, the *fixed style explanation* (FSE), such that no matter what type of recommendation the user is given, FSE presents such an explanation that it makes the user think that the quality of the item is 3 out of 5. If the *actual-ratings* are equally distributed in the interval $[1, 5]$, then the mean difference between the *explanation-ratings* and the *actual-ratings* for FSE will be 0. However, this does not necessarily mean that FSE is a good explanation. *Explanation-ratings* for a good explanation style should have $\mu_d = 0$, a low σ_d, plus they should strongly correlate with the *actual-ratings*.

We have calculated the Pearson correlation between *actual-ratings* and *explanation-ratings* along with their respective probabilities of being non-zero due to chance for all explanation styles. Results are presented in Table 4. The most

Table 4: Correlations and P-Values

	Actual	
	r	**P**
ISE	0.23	0.10
KSE	0.34	0.01
NSE	-0.02	0.90

strongly correlating explanation is KSE at 0.34. The prob-

ability of getting this high of a correlation due to chance is only 0.01. ISE has a correlation of 0.23 and the probability of having this high of a correlation by chance of 0.1. Even though it does not meet the standard value of 0.05, it is close. The correlation constant for NSE is negative, however, the chance of having this small of a negative correlation is 90%. The correlation table supports our Hypothesis 3 fully for KSE and partially for ISE. NSE does not result in any correlation, indicating that it is ineffective at helping users evaluate the quality of a recommendation.

Future Work

Because of time issues, we had to ask the users to read the Amazon's pages instead of the books themselves. The experiment can be repeated in a domain where trying out the recommended item does not take much time, like a movie or music domain. Moreover, there are twenty other explanation styles described in Herlocker et al.'s paper [6]. The experiment could be repeated with these other explanation styles as well. Note that they found that NSE was the best explanation from a *promotion* perspective. Another style in that study could perform better from a *satisfaction* viewpoint.

Conclusions

The ability of recommender systems to effectively explain their recommendations is a potentially crucial aspect of their utility and usability. The goal of a good explanation should not be to "sell" the user on a recommendation, but rather, to enable the user to make a more accurate judgment of the true quality of an item. We have presented a user-study that evaluated three different approaches to explanation in terms of how accurately they allow users to predict a more in-depth evaluation of a recommendation. Our results demonstrate that the "neighborhood style" explanation for collaborative filtering systems previously found to be effective at promoting recommendations [6], actually causes users to overestimate the quality of an item. Such overestimation would lead to mistrust and could eventually cause users to stop using the system. Keyword-style explanations, which present content information about an item that caused it to be recommended, or influence-style explanations, which present ratings previously provided by the user that caused an item to be recommended, were found to be significantly more effective at enabling accurate assessments.

Acknowledgments

This research was partially supported by an undergraduate research assistant fellowship to the first author from the Dept. of Computer Sciences at the Univ. of Texas at Austin and by the National Science Foundation through grant IIS-0117308.

REFERENCES

1. C. Basu, H. Hirsh, and W. W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, pages 714–720, Madison, WI, July 1998.
2. D. Billsus and M. J. Pazzani. A personal news agent that talks, learns and explains. In *Proceedings of the Third Annual Conference on Autonomous Agents*, pages 268–275. ACM Press, 1999.
3. R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
4. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the Association for Computing Machinery*, 35(12):61–70, 1992.
5. J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237, Berkeley, CA, 2000. ACM Press.
6. J. Herlocker, J. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work*, pages 241–250, Philadelphia, PA, 2000. ACM Press.
7. P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, pages 187–192, Edmonton, Alberta, 2002.
8. R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, pages 195–204, San Antonio, TX, June 2000.
9. M. J. Pazzani, J. Muramatsu, and D. Billsus. Syskill & Webert: Identifying interesting web sites. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, pages 54–61, Portland, OR, Aug. 1996.
10. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, pages 175–186. ACM Press, 1994.
11. B. M. Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*, pages 345–354. ACM Press, 1998.
12. W. Swartout, C. Paris, and J. Moore. Explanations in knowledge systems: Design for explainable expert systems. *IEEE Expert: Intelligent Systems and Their Applications*, 6(3):58–64, 1991.

Identifying Attack Models for Secure Recommendation

Robin Burke, Bamshad Mobasher, Roman Zabicki, Runa Bhaumik
School of Computer Science, Telecommunications and Information Systems
DePaul University
Chicago, IL
{rburke, mobasher}@cs.depaul.edu, rfzabick@punkgrok.org, rbhaumik@cs.depaul.edu

ABSTRACT
Publicly-accessible adaptive systems such as recommender systems present a security problem. Attackers, who cannot be readily distinguished from ordinary users, may introduce biased data in an attempt to force the system to "adapt" in a manner advantageous to them. Recent research has begun to examine the vulnerabilities of different recommendation techniques. In this paper, we outline some of the major issues in building secure recommender systems, concentrating in particular on the modeling of attacks.

Keywords
Recommender systems, personalization, security, attack modeling.

INTRODUCTION
Recommendation systems are an increasingly important component of electronic commerce and other information access systems. Users have come to trust personalization and recommendation software to reduce the burden of navigating large information spaces and product catalogs. The preservation of this trust is important both for users and site owners, and is dependent upon the perception of recommender systems as objective, unbiased and accurate. However, because recommendation systems are dependent on external sources of information, such as user profiles, they are vulnerable to attack. If a system generates recommendations collaboratively, that is by user-to-user comparison, hostile users can generate bogus profiles for the purpose of biasing the system's recommendations for or against certain products.

Consider a recommender system that identifies books that users might like to read using a user-based collaborative algorithm. (See [7] for the classic formulation of user-based collaborative filtering.) Alice, having built up a profile from previous visits, returns to the system for new recommendations. Figure 1 shows Alice's profile along with that of seven genuine users. An attacker Eve has inserted profiles (Attack1-5) into the system, all of which give high ratings to her book labeled Item6. Without the attack profiles, the most similar user to Alice would be User3. The prediction associated with Item6 would be 0.0, essentially stating that Item6 is likely to be strongly disliked by the user. If the algorithm used the closest 3 users, the system would still be unlikely to recommend the item.

Eve's attack profiles closely match the profiles of existing users, so when these profiles are in the database, the Attack1 profile is the most similar one to Alice, and would yield a predicted rating of 1.0 for Item6, the opposite of what would have been predicted without the attack. Taking the most similar 3 users in this small database would not offer any defense: Attack1, Attack4 and User3 would be selected and Item6 would still be recommended. So, in this example, the attack is successful, and Alice will get Item6 as a recommendation, regardless of whether this is really the best suggestion for her. She may find the suggestion inappropriate, or worse, she may take the system's advice, buy the book, and then be disappointed by the delivered product.

This paper identifies key issues for the study of secure recommendation, focusing particularly on the identification of attack models. In doing so, we draw particularly on two recent studies. O'Mahony and his colleagues [12] conducted a pioneering study on the problem of the robustness of collaborative recommendation. The authors developed a formal framework for analyzing the vulnerability of kNN-based collaborative filtering, and conducted associated experiments comparing actual systems to their theoretical model. Lam and Riedl [9] have also recently published some empirical studies of attacks against collaborative algorithms.

SECURE RECOMMENDATION

As depicted in Figure 2 we identify six essential components that make up the study of secure recommendation. Attack models, the patterns of interaction that attackers may use to influence the system; algorithms, the methods by which predictions are made; profiling, the techniques by which user profiles are gathered and represented; data sources, the different types of data on which recommendation is based; detection, the process by which attacks against a system can be detected; and response, the actions that can be taken to remove bias injected by an attacker. Not part of the system per se, the topic of evaluation is also important for quantifying the vulnerabilities of systems and comparing different designs.

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

	Item1	Item2	Item3	Item4	Item5	Item6	Correlation w. Alice
Alice	1.0	0.2	0.6	0.6		?	
User1	0.2		0.8		0.8	0.2	-1
User2	0.2	0.0	0.6		0.2	0.2	0.33
User3	0.8	0.2	0.6	0.6		0.0	0.97
User4	0.6	0.5	0.5		0.5	0.2	0.87
User5		0.6		0.5	0.5	0.2	-1
User6	0.4	0.4		0.5	0.5	0.2	0.0
User7		1.0		0.2	1.0	0.0	-1
Attack1	1.0	0.2	0.6	0.6		1.0	1.0
Attack2	0.2		0.2		0.8	1.0	NaN
Attack3	0.2	0.0	0.6		0.2	1.0	0.33
Attack4	0.8	0.2	0.6	0.6		1.0	0.97
Attack5	0.4	0.4		0.5	0.5	1.0	0.65

Figure 1. A push attack favoring Item6.

Our primary goal in this paper is to provide a framework based on these six dimensions for further research in secure personalization. We briefly discuss each of the components introduced above, however, we focus primarily on identifying and characterizing different attack models.

Data sources

The research cited above has concentrated on recommenders that use collaborative data for making recommendations. Another class of recommender system uses algorithms based not on user ratings alone but also information about the items themselves. A content-based recommender, for example, predicts a particular user's preferences by learning to classify items as liked and disliked based on their features and the user's feedback; see [10] for example. A knowledge-based recommender reasons about the fit between a user's need and the features of available products [2].

These systems are vulnerable to a different form of attack, in which the data sources of content information are attacked. This phenomenon is well-known in the web search industry as "search engine spam"[1]: web page authors attach misleading keywords, metadata and other information to their pages in the hopes of being retrieved in response to popular queries.

While there is little direct defense against falsified content data, combining different sources of data in hybrid systems may offer some defense against attacks against any particular data source. This possibility is discussed further below.

Algorithms

The recent research cited above has concentrated on the characteristics of the kNN algorithm for recommendation as this is the most commonly-used technique. The most typical formulation of the idea is the user-to-user comparison seen in Figure 1.

Item-based collaborative filtering works slightly differently [13]. Instead of comparing users' pattern of ratings of items, the system compares the patterns of preference for each item across all users. [9] examines the application of the kNN technique to item-based collaborative filtering, finding some defensive advantages to this technique for certain attacks.

What the kNN techniques share is that they base their predictions on raw user profile data. A variety of model-based recommendation techniques are also well-studied in the recommender systems literature: Bayesian networks, association rules, decision trees, and latent semantic analysis are a few of the techniques that have been used.

A hybrid recommendation algorithm is one that uses multiple data sources of different types. Much of the success of the Google search engine[2] can be attributed to its use of an authority measure (effectively a collaboratively-derived weight) in addition to standard content-based metrics of similarity in query processing [1]. This hybrid technique means that a page that is misleadingly labeled is much less likely to be retrieved by Google than by a system that uses only content. Google is therefore an example of a hybrid approach to secure recommendation, defending against a biased content attack through the addition of collaborative information.

Hybrid recommendation, combining multiple recommenders of different types, is therefore a promising approach for securing recommender systems. The taxonomy of hybrid recommendation developed in [3] can be a useful guide to the landscape of possible hybrid combinations.

Profiling

The research described in [12] and [9] makes use of explicit ratings data: products are individually and explicitly rated by the user as liked and disliked. Some recommender systems use implicit ratings, ratings that are inferred from user behavior, rather than explicitly provided by the user. (See the research reviewed in [6].) Such data sources have different characteristics than the classic explicit rating scenario. In Web usage mining [4, 15], Web

[1]http://www.google.com/contact/spamreport.html

[2]http://www.google.com/

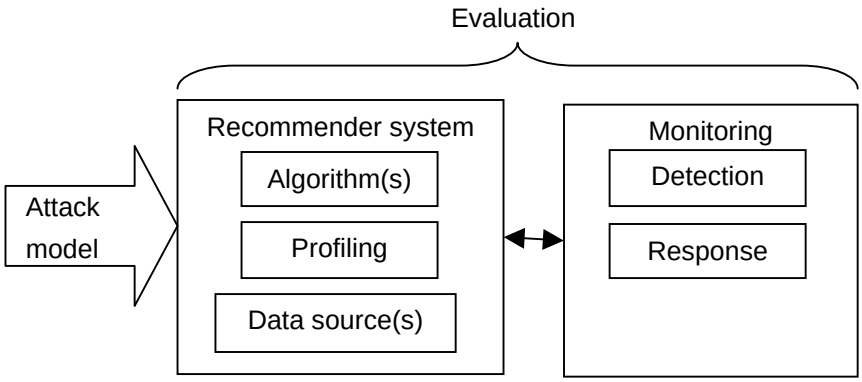


Figure 2. Components of secure recommendation.

server logs are examined for link traversal and dwell time and continuously-valued ratings derived from this analysis, and as a result, negative ratings are not available. We may infer that if page A got less attention than page B that page B is liked more, but not necessarily that page A was disliked. Web usage mining has been studied as the underlying framework for a whole class of Web personalization systems that take users' navigational activity into account to generate recommendations or create adaptive content [11].

Any attempt to secure such recommenders against attack must take into account their unique characteristics both in terms of the structure of the profiles and how profiles are collected. In a web personalization system, an attacker cannot simply call up a list of web pages and assign them ratings. A more sophisticated softbot attack must be mounted in which a software agent interacts with the web site in a predefined way. Such mechanisms are certainly not beyond the capabilities of attackers, and simulations of such attacks will be useful for understanding what distinctive traces they might leave.

Detection and response

Any on-line system may be subject to attack. For most information systems, the ability of an unknown user to change the behavior of the system itself would constitute an attack. For adaptive systems like recommender systems, the whole point is to allow public access and for users' choices to change system behavior. Detection therefore becomes a matter of determining if biased profiles are being injected into the system. Ideally, the most resistant algorithms would be used, the best sources of data and the most effective profiling techniques. However, while attention to these issues will harden the system against attack, they will not eliminate the threat.

Effective detection will depend on the ability to either (i) detect patterns of activity that are characteristic of an attack or (ii) to detect changes in system performance that suggest the presence of biased data. Research into the first option would draw from classic intrusion detection work, in which system behavior is monitored for activity that matches known attack patterns.[3] The second option is intriguing, but perhaps more difficult to realize. [9] demonstrates that an effective attack may not leave any obvious markers, such as a significant degradation of overall accuracy.

The response to an attack will depend on the method by which it is detected. If an attack can be detected by a behavioral measure or the examination of system logs, the profiles generated by the suspect activities can simply be eliminated. However, if bias is detected in a holistic manner, it may be impossible to discriminate the bogus profiles responsible for the bias from those created by genuine users. In this case, the system may need mechanisms to compensate for detected bias without editing the profile base.

Evaluation

There has been considerable research in the area of recommender systems evaluation. See [8] for a particularly comprehensive example. Some of these concepts can also be applied to the evaluation of the security of recommender systems, but in evaluating security, we are interested not in raw performance, but rather in the change in performance induced by an attack.

[12] introduced two evaluation measures: robustness and stability. Robustness measures the performance of the system before and after an attack to determine how the attack affects the system as a whole. Stability looks at the shift in system's ratings for the attacked item induced by the attack profiles.

As Lam and Riedl [9] point out, merely measuring the change in prediction strength may not be sufficient to determine the practical impact of an attack. If an attack changes the predicted rating for an item by some quantity, this may have little or no impact on a user if the item does not appear in the small set of items at the top of the

[3] (Denning, 1987) is a classic example, and there is a large body of continuing research in the area, with the regular RAID symposia serving as a major research forum (http://www.raid-symposium.org/).

recommendation list. Lam and Riedl measure instead how frequently a pushed item is moved into the top recommendation set. This is a more practical measure of the consequences of an attack for end users.

Stability allows us to evaluate how attacking the system benefits the attacker, but only if there is a single outcome of interest. In general, an attacker may have a more complex notion of utility, she may want some products recommended and others not, for example.

To capture a more multi-dimensional notion of the attacker's intent, we can turn to utility theory. The expected utility of the system may be defined as

$$E(u) = \sum_{k=1}^{n} w_k P(k)$$

where k ranges over all possible recommendation outcomes, w_k is the utility associated with that outcome for the attacker, and $P(k)$ is the probability of an outcome. After the system is attacked, we compute $E(u')$ for the biased system, and the denote this change as $\delta = E(u')-E(u)$, the expected utility gain for the attack, or the *payoff*.

Different models of attacker utility can then be employed to the task of evaluating payoff. Adopting a simple notion of utility, we can assign a value to each outcome in which the attacker's favorite item ends up in a user's recommendation list. In this case, δ would correspond to the change in frequency of top list appearance, roughly the measure proposed in [9].

However, some of the attacks described below argue for a more complex notion of utility. In the segmented attack, the attacker is trying to get the system to favor one item against others in a competitive environment. Simply getting the favored item into the top n recommendations might not be acceptable if the other items are also present. Such considerations are best addressed by a utility-based framework for evaluation.

ATTACK MODELING

There has been a number of attack types studied in previous work. This section enumerates these attacks and several additional ones.

Perfect Knowledge Attack

A perfect knowledge attack is one in which the attacker reproduces the precise details of the data distribution within the profile database.

In a perfect knowledge attack, the biased profiles injected by the attacker match exactly with the profiles already in the system except that they exhibit bias for or against some particular item. [12] formalized the perfect knowledge attack and identified its two variants: "push" in which the attacker attempts to have the target item recommended more often and "nuke" in which the aim is to prevent recommendation. We will adopt this terminology with respect to the aims of the attacker. Figure 1 is an example of a perfect knowledge push attack.

Their paper assumes that the bias with respect to class is the only perturbation introduced by the attacker. In other words, push profiles look just like real profiles in terms of their distribution of items and ratings: the only difference is that they give some item a positive rating much more frequently. This assumption lends analytical clarity to the approach, but it is probably not realistic that the attacker will be able to craft an attack so accurately.

Random attack

A random attack may be a push or nuke attack, that is, a particular item will be given high or low ratings, but other ratings in each profile are chosen randomly.

[9] used this attack style, crafting bogus profiles by using a normal distribution based on the overall mean and standard deviation of their dataset. Such profiles would have the same overall characteristics as the dataset, but not the same precise distribution of ratings across individual items.

Average attack

An average attack is a push or nuke attack in which the ratings in crafted profiles are distributed around the mean for each item.

With this attack, we assume that the attacker knows the average rating of each item and the profiles therefore match the distribution across items. Arguing that this level of data was likely to be accessible to an attacker (often exposed by the recommender itself), Lam and Riedl proposed the average attack as more realistic than the perfect knowledge attack, yet still sensitive to the data distribution. They showed that an attack using this level of knowledge was significantly more effective than the random attack.

Consistency attack

A consistency attack is one in which the consistency of the ratings for different items is manipulated rather than their absolute values.

[9] found that an item-based collaborative filtering algorithm was on the whole more robust than the user-based technique against the random and average attacks. This can be seen by looking at Figure 1 from an item-based point of view. An item-based recommender would try to predict the preference of Alice for Item6 by looking at other items that have a similar pattern of preference. Here we see that this particular attack is not successful. In the original, the system would have not had any items with a strong similarity with the entries for Item6. After the attack, the similarity with the pushed item is even worse. Most likely the system would not recommend Item6 in either case, but it would be more likely to recommend it before the attack than after.

What this indicates is that a successful attack against an item-based system would have to have a different character than an attack against a user-based one. The situations are not parallel, because the attacker can add user profiles in their entirety, but only can only augment the item profile.

	Item1	Item2	Item3	Item4	Item5	Item6
Alice	1.0	0.2	0.6	0.6		?
User1	0.2		0.8		0.8	0.2
User2	0.2	0.0	0.6		0.2	0.2
User3	0.8	0.2	0.6	0.6		0.0
User4	0.6	0.5	0.5		0.5	0.2
User5		0.6		0.5	0.5	0.2
User6	0.4	0.4		0.5	0.5	0.2
User7		1.0		0.2	1.0	0.0
Attack1	1.0	1.0	1.0	1.0	0.0	1.0
Attack2	1.0	0.0	1.0	1.0	1.0	1.0
Attack3	1.0	1.0	1.0	1.0	0.0	1.0
Attack4	1.0	0.0	1.0	1.0	1.0	1.0
Attack5	1.0	1.0	1.0	1.0	0.0	1.0
Cosine vs Item6 (pre)	-0.78	0.09	0.14	0.38	-0.70	
Cosine vs Item6 (post)	0.77	-0.09	0.92	0.11	-0.27	

Figure 3. A consistency attack favoring Item6.

To attack an item-based recommender successfully, the attacker must generate profiles for the preferred item that match the profiles of other items that the user likes. For example, Eve needs to make her title match well against items that Alice likes and poorly against others. In other words, Alice's favorite items should get ratings that match the pushed item and other titles should be distant. The specific algorithm used may determine what kind of attack would be successful. Against a median-adjusted algorithm like correlation or some forms of cosine similarity [13], the manipulation of values as in the average attack is not very effective because the bogus profiles essentially add the same thing to their respective items: relatively constant profile values. In order to attack this kind of algorithm the attacker must vary the ratings given to non-favorites across different profiles while using essentially fixed positive values for those favorites that we want to have match against the item to be pushed.

An example of consistency attack can be found in Figure 3. The pattern of ratings in the attacking profiles is the same for Item6 as for Item1, Item3 and Item4, those preferred by Alice. The items not preferred by Alice (Item2 and Item5) get profiles that violate the pattern. As the example shows, the item that Alice likes best Item1 moves from a very low similarity to Item6 to a very high similarity with the attack in place.

In this contrived example, the attacker knows exactly which items Alice likes and dislikes, but in practice, the kind of distributional data used in the average attack can be used to achieve the same effect by selecting items frequently rated high and low. We are currently conducting experiments to quantify the effectiveness of this novel form of attack.

Segmented attack

A segmented attack is an attack against a set of related items – equivalent to a market segment in an e-commerce domain. The aim is to push one item ahead of others considered to be its competitors.

With the segmented attack, the aim of the attacker is not as simple as in the uni-dimensional push or nuke models. In a segmented attack, the attacker's aim is to push a particular item, but also to simultaneously nuke other products within the same segment. The attacker will want to have an effect on recommendations given to precisely those users with an interest in the market segment in question.

For example, Eve may identify other books that share the same general topic as her book Item6 and craft profiles using them. Suppose that, in our example Items 3 and 4 are in the same genre as Item6. An examination of Figure 1 shows that the perfect knowledge push attack does not reduce the recommendation score of Item3 and Item4, and depending on how the score is computed, might even increase it.

Finding the optimal segmented attack is somewhat tricky. A naive approach of combining nuke and push in a single set of attack profiles will not be successful, since such profiles would not be similar to users interested in the genre – all of whom would presumably actually like some of the items that Eve is trying to nuke. Consider what would happen if the attack profiles in Figure 1 had very low scores for Items 3 and 4 – they would lose their similarity to Alice.

Launching a large number of separate nuke attacks, one against each item in the genre and a simultaneous push attack in favor of Item6 is a possible but computationally costly alternative: if Eve is pushing a title with many competitors such as a romance novel, she would have many competing titles to nuke. In addition, such an attack could only guarantee that the competing products would be recommended less often overall and the pushed products more often. It will not guarantee that the pushed item is pushed to Eve's target market.

A successful segmented attack will be one in which the push item is recommended to precisely those users that have many positive ratings in the genre, and in which the presence of similar ratings in the bogus profiles does not

	Item1	Item2	Item3	Item4	Item5	Item6	Correlation w. Alice
Alice	1.0	0.2	0.6	0.6		?	
User1	0.2		0.8		0.8	0.2	-1
User2	0.2	0.0	0.6		0.2	0.2	0.33
User3	0.8	0.2	0.6	0.6		0.0	0.97
User4	0.6	0.5	0.5		0.5	0.2	0.87
User5		0.6		0.5	0.5	0.2	-1
User6	0.4	0.4		0.5	0.5	0.2	0.0
User7		1.0		0.2	1.0	0.0	-1
Attack1	0.9	0.8			0.8	1.0	1.0
Attack2	0.2	0.8			0.8	1.0	-1
Attack3	0.8	0.2			0.8	1.0	1
Attack4	0.9	0.8			0.2	1.0	1
Attack5	0.8	0.2			0.2	1.0	0.65

Figure 4. Bandwagon attack favoring Item6; Items 1, 2 and 5 are the most-commonly

have the effect of increasing the likelihood of the competing products being recommended. Interestingly, it may easier to craft a segmented attack against an item-based algorithm than a user-based one. The goal in the item-based attack would be to add data in which the ratings for the pushed item follow the same pattern as the competing other genre items. We are still investigating the parameters of an optimal segmented attack.

Bandwagon attack

A bandwagon attack is one in which the aim is to associate the pushed item with a fixed set of popular items.

The segmented attack, which aims at particular users interested a particular corner of the item space, is in some ways the opposite of the bandwagon attack. In this attack, the attacker can take advantage of Zipf's law: a small number of items, best-seller books for example, get the lion's share of attention, and hence are more likely to appear in user profiles. By associating her book with current best-sellers, Eve can ensure that her bogus profiles have a good probability of matching any given user, since so many users will have these items on their profiles. The bandwagon attack shown in Figure 4 is even more successful that the perfect attack. Note that it is not necessary to rate the commonly-rated items highly: Alice, for example, does not like Item 2, but most will have some opinion. [12] looked at bandwagon attacks in which ratings for two well-liked items were the basis for push attacks, as in Figure 4. While such attacks could theoretically succeed, a more practical attack would need to target a larger set of popular items since fielded recommender systems will lower bounds on the degree of profile overlap that they will accept.

Probing attack

A probing attack is one in which the aim is to discover the algorithms and/or parameters of the recommender system itself.

As we have shown (and years of experience in computer security confirm), an attacker's knowledge of the system plays a large role in determining the success of an attack. The more Eve knows about the targeted recommender system, the more effective she can make her attack. This is clear with the consistency attack: Eve would only choose such an attack if she knew that the system was item-based.

On one hand, it is a well-established principle of computer security that the fewer secrets the security of a system depends on, the more secure it is [14]. In particular, the security of a recommender system should not depend on its algorithms being unknown – so-called "security through obscurity."

On the other hand, these technical details will not be public knowledge. It may be necessary for an attacker to acquire this knowledge through interaction with the system itself: hence, the probing attack. A probing attack will most likely consist of a series of small-scale attack / test combinations designed to yield enough information to support a later intervention that supports the attacker's real goal.

Given an infinite number of interactions, an attacker would in principle be able to learn everything there is to know about a recommender system. The relevant question for the probing attack is again one of utility: can the marginal benefit of each probe (in terms of increased certainty about algorithm and parameters) be minimized so that such attacks are rendered impractical?

CONCLUSIONS AND FUTURE WORK

Recent research has established the vulnerabilities of the most common collaborative recommendation algorithms. This paper has outlined some of the important issues for continuing research in secure recommender systems: attack models, algorithms, data sources, profiling techniques, detection and response, and evaluation. It is clear that this will be a fruitful area of research for some time.

A recommender system can be considered robust if it can maintain its recommendation quality in the face of attackers' attempts to bias it. The "robustness" property must be predicted on attack type: there is no algorithmic defense against a physical attack that unplugs a server, for example. Attack modeling is therefore a necessary first step to clarifying what we mean by a system's robustness:

against what attacks, on what scale, and with what system knowledge. This paper has outlined four attack models (consistency, segmented and bandwagon) against which our recommendation models have yet to be fully evaluated. Our future work will attempt both to expand this suite of attack models, to evaluate against them a range of different recommender system algorithms and designs, and to explore the problems of detection and response.

ACKNOWLEDGEMENTS

This research is supported in part by the National Science Foundation under the NSF CyberTrust Program, Award #IIS-0430303.

REFERENCES

1. Brin, S. and Page, L. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, *30*, *1–7* (1998), 107–117.
2. Burke, R. Knowledge-based Recommender Systems, in A. Kent (ed.), *Encyclopedia of Library and Information Systems*. Vol. 69, Supplement 32. Marcel Dekker, New York, 2000.
3. Burke, R.. (2002) "Hybrid recommender systems: Survey and experiments". *User Modeling and User-Adapted Interaction*, *12*, *4* (2002),331–370.
4. Cooley, R., Mobasher, B., and Srivastava J. (1999) "Data preparation for mining world wide web browsing patterns". *Journal of Knowledge and Information Systems*, *1*, *1* (1999).
5. Denning, D. E. An Intrusion-Detection Model. *IEEE Trans. on Software Engineering*, *13*, *2* (1987), 222-232.
6. Kelly, D. and Teevan, J. Implicit feedback for inferring user preference: A bibliography. *ACM SIGIR Forum*, *37*, *2* (2003).
7. Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. T. An Algorithmic Framework for Performing Collaborative Filtering. In *ACM SIGIR 1999*, 230-237.
8. Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems*, *22* , *1* (2004).
9. Lam, S. K. and Riedl, J. Shilling Recommender Systems for Fun and Profit. In *WWW 2004*, New York, May 2004.
10. Lang, K. Newsweeder: Learning to filter news. In *Proc. of the 12th International Conference on Machine Learning*, (Lake Tahoe, CA, 1995), 331-339.
11. Mobasher, B. Web usage mining and personalization, in Singh, M. P. (ed.), *Practical Handbook of Internet Computing*, Chapman Hall & CRC Press, 2004.
12. O'Mahony, M., Silvestre, G. and Hurley, N. Collaborative Recommendation: A Robustness Analysis. *ACM Transactions on Internet Technology*. In press. Accessed at http://www.cs.ucd.ie/staff/nick/-home/research/download/omahony-acmtit2004.pdf.
13. Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. Item-based collaborative filtering recommendation algorithms, in *Proc. of the 10th International WWW Conference* (Hong Kong, May 2001).
14. Schneier, B. Secrecy, Security and Obscurity. *Crypto-Gram*, May 15, 2002. Accessed at http://www.schneier.com/crypto-gram-0205.html.
15. Srivastava, J., Cooley, R., Deshpande, M., and Tan, P. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations*, *1*, *2* (2000), 12–23.

User-Specific Decision-Theoretic Accuracy Metrics for Collaborative Filtering

Giuseppe Carenini
Computer Science Dept. University of British Columbia
2366 Main Mall, Vancouver, B.C. Canada V6T 1Z4
carenini@cs.ubc.ca

ABSTRACT

Accuracy is a fundamental dimension for the effectiveness of recommender systems. Several accuracy metrics have been investigated in the literature. However, we argue, these metrics are not sufficiently user-specific. In previous work, we proposed accuracy metrics that take into account a user-specific pointwise decision threshold. In this paper, we present even more user-specific accuracy metrics that rely on the user utility function on the rating scale as well as on a user-specific sigmoid functional decision threshold.

Keywords

Collaborative Filtering, Evaluation Metrics, Decision Theory.

INTRODUCTION

A critical step in testing a recommender system is the choice of a set of evaluation metrics appropriate for the specific recommender task. However, it is only very recently that a coherent framework to support such a choice has been presented in the literature. In [7] Herlocker et al. critically discuss evaluation metrics that have been used in the past to test collaborative filtering (CF) recommender systems and propose several topics for future work.

Traditionally the most investigated and applied metrics have been measures of how accurately the system can predict the rating of items (*Accuracy Metrics*). [7] discusses similarity and differences among several popular accuracy metrics and shows how certain accuracy metrics are more appropriate for certain user tasks. For instance the ROC metric is more appropriate when the user wants to find good items and there is a clear binary characterization of items (as relevant/non-relevant). Furthermore, in [7], accuracy metrics are compared in an empirical testing which suggests that when applied on several different variations of the same CF algorithm all accuracy metrics appear to group in only three distinct classes.

Although accuracy is a fundamental dimension for recommendation effectiveness, it is not the only one. Other metrics explored in previous work and discussed in [7] include: (a) *Coverage* - the portion of the domain items for which the system can generate predictions; (b) *Learning Rate* - How the prediction ability of the system increases as more data is provided; (c) *Novelty and Serendipity* - Whether the recommended items are not known by the user and whether the user would probably not have discovered those items; (d) *Confidence* - How effectively the system is able to generate and express its confidence in the predicted ratings; (e) *User Evaluation* - How real users actually react to the system's recommendations when they interact with a recommender system in lab or field studies.

After a detailed survey of all current metrics for testing CF Herlocker et al. suggest the following guidelines for *applying* existing metrics and *developing* new metrics:

Researchers in CF should:

- choose the accuracy metrics that best match their assumed user task(s).
- consider in their choice the finding that accuracy metrics group in three equivalence classes.
- further develop and apply the other metrics besides accuracy (described above).
- develop comprehensive quality measures that effectively integrate accuracy metrics with the other metrics.

Although we recognize these as extremely valuable suggestions, in this paper we propose another goal for research on CF evaluation metrics and present some preliminary steps to achieve it.

We argue that more work is necessary to devise more adequate accuracy metrics. Remarkably, the only information about the user required by all the accuracy metrics currently

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

used to evaluate CF systems are the user's true ratings. We believe that more informative metrics should take into account more user specific information, especially if this information can be easily acquired.

In previous work [1] we have proposed a first step in this direction, namely that adequate accuracy metrics should take into account a user specific pointwise threshold on whether to accept or refuse a recommendation. In this paper we move a step further, we argue that accuracy metrics should be even more user specific. In particular, first, in determining how much the user is gaining from following a recommendation, they should take into account the user utility function in the rating scale. Secondly, the user specific threshold should not be required to be a point. We claim that a sigmoid function would be a more adequate and more user-specific representation for the threshold.

In the remainder of the paper, we first report on our previous work on decision-theoretic user-specific accuracy metrics based on a pointwise user specific threshold. Next, in light of results from a user study and practical considerations, we reconsider two key assumptions on which our metrics were based. After that, we present our novel metric that does not rely on the two assumptions. We then discuss possible problematic aspects of our proposal and describe future work to address them.

OUR PREVIOUS WORK ON DECISION-THEORETIC ACCURACY METRICS FOR CF

The Mean Absolute Error (MAE) is the most commonly used measure to evaluate the accuracy of CF algorithms. Let's assume the set P_a with cardinality n_a contains the ratings that the CF algorithm attempts to predict for current user a, then the MAE for that user is given as follows:

$$MAE_a = \frac{1}{n_a} \sum_{j \in P_a} |o_{a,j} - p_{a,j}|$$

where $o_{a,j}$ is user a's observed rating for item j and $p_{a,j}$ is user a's predicted rating for item j. The MAE reported in CF evaluations is the average MAE for all users in the test set. Notice that the lower the MAE, the more accurate the CF algorithm is.

In [1] we criticize MAE because it relies on viewing the recommendation process as a machine learning problem, in which recommendation quality is equated with the accuracy of the algorithm's prediction of how a user will rate a particular item. This perspective is missing a key aspect of the recommendation process. The user of a recommender system is engaged in deciding whether or not to experience an item (e.g., whether or not to watch a movie). So, the value of a recommendation critically depends on how the recommendation will impact the user decision making process and only indirectly on the accuracy of the recommendation. For illustration, consider a user who will watch only movies whose predicted rating $p_{a,j}$ is greater than 3.5. Now, consider the following two predictions for that user:

(i) $p_{a,j} = 0$; when $o_{a,j} = 2$; (Absolute Error = 2)

(ii) $p_{a,j} = 3$; when $o_{a,j} = 4$; (Absolute Error = 1)

The Absolute Error in *(i)* is greater than the one in *(ii)*. However, in terms of user decision quality, *(i)* leads to a good decision because it entails that the user will avoid watching a movie that she would not have liked, while *(ii)* leads to a poor decision, because it entails that the user will miss a movie that she would have enjoyed.

The key point of this example is that in measuring the quality of a recommendation we should take into account the criterion used by the user in deciding whether or not to accept the recommendation. When the recommendation is provided as a predicted rating, a simple plausible criterion is a user specific threshold in the range of possible ratings. The user will accept a recommendation when the prediction is greater than the threshold.

More formally, let θ_a be a user specific threshold (in the range of possible ratings) such that:
$p_{a,j} \geq \theta_a \Rightarrow select(a, j)$
where, as before, $p_{a,j}$ is user a's predicted rating for item j, and $select(a, j)$ means that user a will select item j (e.g., the user will watch the movie).

Then, the quality of a recommendation $p_{a,j}$, which we call User Gain (UG), can be defined as:

$$UG(p_{a,j}) = \begin{cases} o_{a,j} - \theta_a & \text{if } p_{a,j} \geq \theta_a \\ \\ \theta_a - o_{a,j} & \text{otherwise} \end{cases}$$

where, as previously defined, $o_{a,j}$ is user a's observed rating for item j.

The first condition covers the situation in which, since the prediction is greater than the threshold, the user will decide to experience the item and will enjoy it to the extent that its true rating is greater than the threshold. The second condition covers the situation in which the user will decide not to experience the item. In this case, the user will gain to the extent that the item's true rating is smaller than the threshold.

Similarly to MAE_a, we can also define the active user Mean User Gain (MUG_a) as:

$$MUG_a = \frac{1}{n_a} \sum_{j \in P_a} UG(p_{a,j})$$

and MUG as the average MUG_a for the test set

In [1] we also discuss how the revision of MAE leads to a revision of the ranked scoring (RS) metric, a less commonly used measure of recommendation quality which can be applied when the recommender presents a recommendation to the user as a list of items ranked by their predicted ratings. In [1], the revised user-specific version of RS is called RS_a^{UG}.

In this paper we take a second look at UG and notice that it also makes two rather unrealistic assumptions:

- In UG the gain/loss of a decision is measured as a difference between ratings (see UG def.) rather than as a difference between the user-specific utility (in decision-theoretic sense [2]) of those ratings. Doing so implies that the utility of different ratings is a linear function for all users (and consequently it is the same for all users).
- In UG the user decision threshold θ_a is a point on the rating scale. This excludes the possibility that for certain ratings a user may *sometimes* follow a recommendation with that rating and sometimes not.

To test the soundness of the first assumption we ran a user study. As for the second assumption we will criticize it on the basis of discussions with users.

USER STUDY TO TEST UTILITY ASSUMPTION

In our user study participants filled out a questionnaire eliciting their utility functions for movie ratings (in the decision-theoretic sense). The questionnaire was completed by 15 participants (students and faculty at UBC).

This was based on the classic probability-equivalent (PE) procedure (see [2]), in which the utility of a rating v is equal to the probability p that makes the participant indifferent between:

- the gamble: watch a 5 rated movie with probability p; watch a 0 rated movie with probability $(1-p)$.[1]
- and the certain outcome of watching a v rated movie.

The outcomes of the utility elicitation process are summarized in Figure 1. The figure shows a box-and-whisker plot of the utility functions for the 15 participants in the study. It is clear that the utility of ratings varies considerably among participants [2].

In conclusion, this user study indicates that at least in the movie domain the assumption that all users have the same linear utility function on the rating scale is incorrect.

POINTWISE vs. FUNCTIONAL DECISION THRESHOLD

We also argue that the second assumption of a pointwise threshold for the user decision is rather unrealistic. Although we do not have any evidence from formal experiments, in several discussions with users of movie recommenders it became clear that people follow deterministic strategies only for extreme ratings, but are more flexible for ratings in the middle of the scale. For instance, they would definitely (not) go to a movie rated in the interval (1-2) 4-5, but they may

[1] 5 and 0 are respectively the best and the worst possible ratings in the movie domain.

[2] The line in the middle of the box marks the median (the second quartile), the lower and upper extremes of the box mark the first and third quartile respectively. The two whiskers from each end of the box mark to the smallest and largest values (except for outliers which are marked by single points)

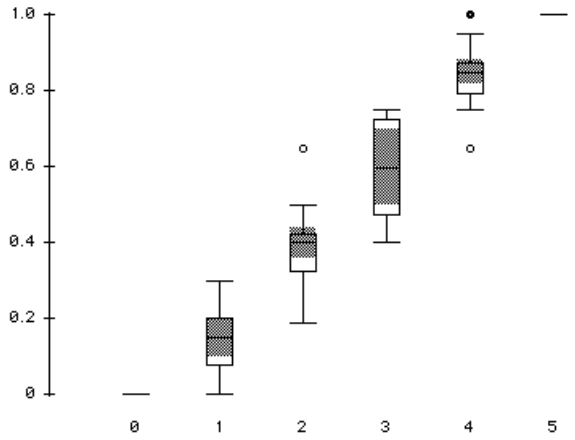


Figure 1: Box-whisker plot of the utility functions for the 15 participants in the study.

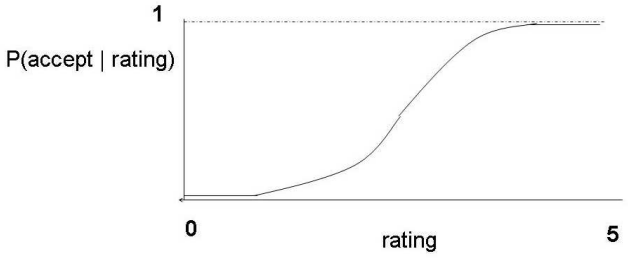


Figure 2: Sample sigmoid function for movie ratings: $1/1+e^{-2(x-2.5)}$.

or may not go to a movie rated in the interval 2-4 [3]. To represent this kind of decision strategy we propose to represent the user decision threshold as a sigmoid function: $1/1+e^{-ax}$. A smooth and continuous thresholding function frequently used in AI, Economics and other fields. An example is shown in Figure 2 for the $0,\ldots,5$ rating scale. This function may be interpreted as the probability of accepting a recommendation given a certain rating (these conditional probabilities do not have to sum up to 1).

NEW ACCURACY METRICS

In the previous two sections we have shown that two key assumptions underlying the specification of UG are rather unrealistic. In this section, we will define new accuracy metrics resulting from abandoning these assumptions.

[3] We recently realized that empirical evidence for this kind of decision strategy is presented in [3]. When people have to map their movie ratings from a 1-5 scale to a binary scale (i.e., accept vs. do not accept the recommendation), their mappings strongly point to a sigmoid decision threshold. See Figure 2 in [3].

Let's begin with the assumption about the utility function. Since our user study indicated that users' utility function on the rating scale vary widely, we propose to redefine UG (and consequently RS^{UG}) into a new metric that explicitly considers the user specific utility functions.

Formally, if u_a is the utility function for the active user on the rating scale and $\theta_a^u = u_a(\theta_a)$, a more refined UG could be defined as follows:

$$UG^u(p_{a,j}) = \begin{cases} u_a(o_{a,j}) - \theta_a^u & \text{if } u_a(p_{a,j}) \geq \theta_a^u \\ \theta_a^u - u_a(o_{a,j}) & \text{otherwise} \end{cases}$$

And correspondingly more refined MUG^u and $RS_a^{UG^u}$ could be defined, in which UG is substituted with UG^u.

As for abandoning the second assumption, we propose to revise UG^u so that the user decision criterion for accepting a recommendation is not based on a user-specific pointwise threshold, but on a user-specific sigmoid function sig_a which expresses the probability that a user will decide to experience an item with a given rating. Remember that UG considered two possibilities (i) when the prediction is greater then the threshold, the user will decide to experience the item, otherwise (ii) the user will decide not to experience the item. With a sigmoid threshold, we have that a user given a predicted rating $p_{a,j}$ will decide to experience the item with probability $sig(p_{a,j})$ and decide not to experience the item with probability $(1 - sig(p_{a,j}))$. Therefore, we have to take into account both possibilities simultaneously in what formally is an Expected User Gain (EUG):

$$\begin{aligned} EUG^{u-sig}(p_{a,j}) = \\ [sig(p_{a,j}) * (u_a(o_{a,j}) - \theta_a^u)] + \\ [(1 - sig(p_{a,j})) * (\theta_a^u - u_a(o_{a,j}))] \end{aligned}$$

And correspondingly more refined MEUG^{u-sig} and $RS_a^{EUG^{u-sig}}$ could be defined, in which UG^u is substituted with EUG^{u-sig}.

Notice that, in EUG^{u-sig}, the computation of the gain (the second factor in the products) still relies on a pointwise θ_a^u. This is reasonable because it conceptually corresponds to the utility for the user of the resources invested in experiencing an item (e.g., time and money in the movie domain).

OPEN ISSUES

Since our proposal is quite preliminary, there are several open issues that may lead to interesting discussions at the workshop.

Elicitation of utility functions

- Will users be willing to go through the utility elicitation process? Presumably, it will depend on the number of ratings considered in the given domain.

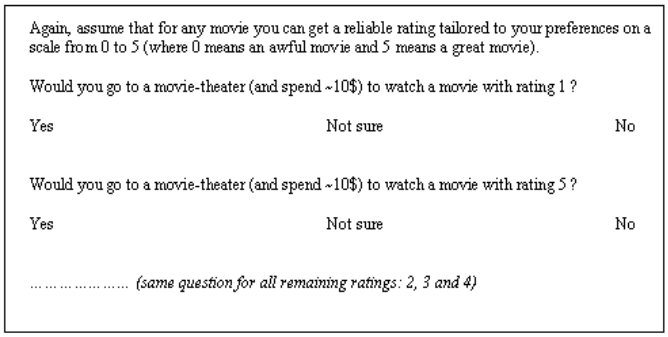


Figure 3: Portion of the questionnaire to assess θ_a and sig

- Is there any alternative elicitation procedure? If some stereotypical utility functions can be acquired (in a given domain) then the problem of utility elicitation would be simplified to mapping a given user into the appropriate stereotype [6].

- Can we safely use utility functions as components of evaluation metrics?
  It is well known in decision theory that differences between values of utility functions (that express risk-attitudes) are meaningless unless the utility function is also a measurable value function [4]. This may or may not be the case ([8] pag. 132). So before applying UG^{u-sig} in a domain, practitioners should make sure that the elicited utility functions are also measurable value functions.

How to elicit the functional decision threshold?

- The sigmoid function sig could be elicited by having the user fill out a questionnaire about her decision strategies in a given domain. Figure 3 shows the part of the questionnaire that we have designed for the movie domain. We have left the definition of the ratings as general as possible *"..where 0 means an awful movie and 5 means a great movie"* to preserve as much as possible the generality of the assessment. In interpreting the participant answers, the following schema could be applied. Let's call max^{no} the highest rating for which the participant answered "no" and min^{yes} the lowest rating for which the participant answered "yes". Then θ_a is assigned the midpoint of [max^{no}, min^{yes}] and sig can be specified so that its inflection point is in θ_a and $sig(max^{no}) = sig(min^{yes}) \approx 1$

- Alternatively, the functional decision threshold could be acquired from the user behavior history. If the user was providing the system with feedback on whether she has decided to experience recommended items the system could simply construct the functional decision threshold by computing for each rating the frequency by which the user accepted recommendations with that rating.

CONCLUSIONS AND FUTURE WORK

As CF recommender systems become more and more popular, there is a pressing need to develop effective evaluation metrics. Traditionally, accuracy metrics have been the most intensely investigated. In this paper, we reconsider popular accuracy metrics.

We noted that all accuracy metrics currently used do not sufficiently take into account the possibly highly user-specific decision process underlying the user interaction with a CF recommender system. To address this limitation we propose novel accuracy metrics based on the Expected User Gain measure. EUG is highly user-specific. It not only takes into account the user utility function on the rating scale in computing the user decision gain, but it also models the user decision criterion for accepting a recommendation as a sigmoid function expressing the probability that the user accepts a recommendation given a certain rating.

Once the open issues discussed in the previous section will be sufficiently clarified, we plan to apply our new metrics to compare existing CF algorithms. To do this, first we need to collect a new dataset in a domain that includes not only the user/rating matrix but also: (i) user specific θ_as; (ii) corresponding user specific sig functions (using a questionnaire); (iii) user specific measurable utility functions. On this dataset, it will be possible to evaluate CF algorithms with EUG^{u-sig}. We are considering movies as our first domain because it is by far the most investigated domain for CF.

However, we would like to test the ideas presented in this paper in at least another domain besides movies. We plan to consider the joke recommendation domain [5]. For this domain, we will go through the same steps aiming to identify similarities and differences with respect to the movie domain. In particular, we are interested in verifying: (i) whether the utility function also varies widely across users in the joke domain (ii) whether in this domain it is more effective to acquire the functional decision threshold by means of a questionnaire or from the user behavior history.

As a long-term goal we plan to perform an evaluation of our approach in live systems. We hope that at the workshop we will be able to establish some form of collaboration with researchers who are running such systems.

ACKNOWLEDGEMENTS

This research is being supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

1. G. Carenini and R. Sharma. Exploring more realistic evaluation measures for collaborative filtering. In *Proceedings of the 19th American National Conference of Artificial intelligence*, 2004.
2. R. T. Clemen. *Making Hard Decisions*. Belmont CA: Duxbury Press, 2nd edition edition, 1996.
3. D. Cosley, S. K. Lam, I. Albert, J. A. Konstan, and J. Riedl. Is seeing believing?: how recommender system interfaces affect users' opinions. In *Proceedings of the Conf. on Human Factors in Comp. Systems CHI03*, p. 585 – 592, 2003.
4. S. Dyer and R. K. Sarin. Measurable multiattribute value functions. *Operations Research*, 27:810–822, 1979.
5. K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.
6. V. Ha and P. Haddawy. Toward casebased preference elicitation: Similarity measures on preference structures. In *Proc. of UAI*, pages 193–201, 1998.
7. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1), 2004.
8. R. L. Keeney. *Value-Focused Thinking: A Path to Creative Decisionmaking*. Harvard University, 1996.

Off-Topic Recommendations

Elyon DeKoven
223 Avenue G, Redondo Beach, CA 90277
playful@dekoven.com

ABSTRACT
In order to make useful recommendations, the product and the person using the product need to focus on the same task. However, when working with a product, people may explore new product features or change their mind about what they want to do. Such behavior can confuse recommendation agents, unless people and their products communicate about the shift of focus.

This paper contributes to the design of recommendation agents that offer off-topic suggestions and consistently handle focus shifting. The paper describes key issues in the design of recommendations under focus shifting, and the three possible dialogue strategies for handling focus shifts. The results of an initial study on one of the strategies indicate trends in how people combine use of off-topic recommendations with focused action in the rest of the product interface. Study results also inform a model of focus shifting behavior, presenting a baseline for future research on the design of off-topic recommendations.

Keywords

Plan recognition, focus shift, dialogue design, recommendation agents

INTRODUCTION
Recommendation agents help people sift through options to come up with better alternatives. In order to make useful suggestions, an agent needs to have a good idea of what the user currently wants or is trying to do. However, at times people may explore unfamiliar product features or change their mind about what they want to do. Indeed, shifting focus can at times be the best thing to do.

People do not always tell their products when they shift focus. As (Lesh, Rich and Sidner, 2001) point out, "interruptions are part of natural collaborative behavior." An agent must be able to determine whether the user's next actions or selection should be interpreted within the same context as the previous one. Typically, at any point during product use there will be more than one reasonable interpretation of what the user is trying to do. Thus the product will rarely have a single 'best' guess.

When the agent is in doubt about the user's current intentions, guessing can lead to problems. For example, changing the temperature on a thermostat could be an attempt to save energy or increase personal comfort. In such cases, the thermostat could guess one or the other and offer advice, but if the guess is wrong the advice may have an adverse affect on the user's trust and use of the product. As Kuhme et al. (1993) say: "the provision of guidance has to be designed very carefully since wrong assumptions about the appropriateness of items can cause fatal problems. Obviously, misleading the user would be even worse than no guidance at all."

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

Guess, asking and waiting
Instead of risking a wrong guess, the system could ask the user to tell the system more explicitly about what the user is trying to do. Conversational recommendation agents bring the human in the loop to make better decisions. Recent work on such agents have examined *item* recommendations, such as comparing among restaurants (Thompson, Goker and Langley, 2004), and others have explored *action* recommendations, such as manipulating constraints to find available flights (Rich, Sidner and Lesh, 2001). The former relates more to the content of the domain, and the latter to the structure of the person-product communication. In both cases, the agent combines guessing with asking the user clarifying questions. However, such prompts can be an unwelcome intrusion, especially if the user is forced to respond to the prompt in order to continue.

	Guess	**Ask**	**Wait**
At best	Leads to efficient product usage and smooth task flow	Clarify what user wants to do, and how to do it	Avoid disturbing user
At worst	Leads to increase in user confusion and decrease in user trust	Annoy user if able to proceed without assistance	Withhold vital assistance when the user is lost

Table 1: Possible dialogue strategies and resulting user experience, in

In order to properly handle situations when a conversational agent is not completely sure about which way the user wants to go, agent design must address tradeoffs between *asking*, *guessing*, and *waiting* (see Table 1). In comparing asking versus guessing, Lesh et al. (2001) note that "The right balance depends, in part, on how often typical users unexpectedly shift their task focus and how often this intention is verbally communicated to the agent."

Towards designing the 'right balance', there is currently little empirical evidence concerning how often people shift focus. Lesh et al. (2001) offer a possible model as to how often people change focus with and without communicating is offered. They suggest that typical users' actions will be focused about 90% of the time, unexpected

focus shifts (within-task) about 5% of the time, and interruptions about 5% of the time. If correct, this model implies about **one out of every ten** actions people take with their products would not fit with what they were doing up to that point. In order to give good recommendations, agents must gracefully handle focus shifting.

Lesh et al. (2001) describe their discourse interpretation algorithm as being "optimized for users who seldom make unexpected focus shifts and, when they do, verbally communicate their intention roughly half the time." While this degree of focused behavior may be expected of 'typical' human collaborators, such an algorithm may not be optimal for first-time or occasional product use. Beginning users may not know how to communicate to the product about what it is they actually want to do. They may make frequent mistakes or randomly explore an interface. Even for people more experienced in using the product, during the course of solving open problems, such as finding a 'good enough' solution to saving energy while maintaining comfort, people might need to switch from one aspect of the problem to another, and from one strategy to another. Thus at times, the best recommendations the agent might have little or nothing to do with what the user has just done.

The rest of this paper examines the design of recommendation systems with respect to focus shifting and off-topic recommendations. Designing support for focus shifting requires three primary considerations: what the product should do when people go off topic, which off-topic items or actions to recommend, and design of the recommendation interface. This paper addresses all three of these issues in the context of action recommendation agents. First, three distinct conversational styles are identified and compared, and a recommendation interface, called the **S**om**e** Thi**n**gs To **Say** (SenSay) menu, is described. Then, results of a study are reviewed towards an empirical model of focus shifting in person-product collaboration.

FOCUS-SHIFTING AND CONVERSATIONAL STYLE

People may have different intentions when shifting focus with regards to whether or not they intend to return to the previous task. In human-human communication, different types of focus shifting are often indicated by linguistic and contextual cues (Grosz and Sidner, 1986). However, not all focus shifts are explicitly communicated.

For example, suppose two agents, say Maya and Reina, were doing something together, say playing marbles. Suddenly, Reina stands up and walks towards the kitchen. Maya might reasonably guess that Reina is hungry. Regardless, how would Maya know whether or not Reina plans on going back to the marbles, or whether she and Reina should clean them up? Choosing the right interpretation is necessary to ensure smooth communication.

Actually, choosing the right interpretation strategy to use depends on more than one focus shift; it is necessary to consider what happens over multiple sequential focus shifts. For example, suppose in the middle of eating, Reina runs off to play with marbles again. Should Maya think that playing with marbles is an interruption of eating as an interruption of playing marbles?

In order to maintain shared focus with the user, the agent can ask, guess or wait, as shown Table 1. The agent's choice hinges on how whether or not the user is done with the interrupted task. In this respect, waiting for more evidence would be the same as guessing that the user is indeed not done. The agent has thus only three options when the user tries to shift focus: ask the user, guess the user is done, or guess the user is not done. This leads to the following three interpretation strategies (see Table 2):

1) Presumptive: The first strategy in Table 2, *asking*, prevents the user from shifting focus until the user has stated whether or not they were done with the previous task. Such a strategy ensures the user is aware of the focus shift, and makes sure the system makes the right interpretation. The strategy is called 'presumptive' since the strategy implies the system knows what is in the best interests of the user, and hence the user will find it worthwhile to answer the system's questions. However, such questions could be confusing to the user if the user does not know whether or not to come back to the task. In the running example above, Reina may not be sure she wants to go back to playing marbles after she is done eating. This strategy also adds an extra turn at every focus shift for confirmation. As Constantine and Lockwood (1999, p. 257) say "Confirmations interrupt the progress of work and annoy users. Nearly all confirmations are unnecessary or ineffective." People might evaluate such extra effort as undesirable, and make them less likely to want to shift focus.

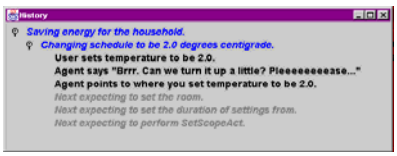
2) Nested Interruptions: The second strategy in Table 2, *guessing the user is not done*[1], creates many levels of

Strategy	Description	Agent	User	Analysis
Ask (Presumptive)	Prevent focus shift until user confirms done	Has user consent	May not be sure whether to shift	Adds extra turns and possibility of nested interruptions
Guess Not done (Nested interruptions)	Assume user will go back to previous task	No user consent	May not be aware of shift	Nested interruptions
Guess Done (Go with the flow)	Assume user is done with previous task	No user consent	May not be aware of shift	No interruptions

Table 2: Interpretation strategies when the user shifts focus.

nested interruptions. In the running marbles example, suppose in the middle of eating a cookie Reina comes back to the marbles. Adopting this strategy, Maya would think that Reina is not yet done eating the cookie, and thus that playing with marbles is an interruption of snacking, which itself was an interruption of playing with marbles. Such situations could complicate communication, and recovery from communication failures. While nested interruptions could also occur with the presumptive strategy if the user is not done, the guessing-not-done strategy does not have the advantage of the presumptive strategy of informing the user that a focus shift took place.

Additional interface and dialogue mechanisms could alleviate these concerns, such as Collagen's segmented History window (Rich and Sidner, 1998) or the History window in Adobe PhotoShop 6.0, but might overly complicate a simple interface such as a thermostat. A second problem with this strategy occurs when actions



The segmented history window from Collagen shows dialogue topics in a nested fashion.

History window in Adobe PhotoShop 6.0.

Figure 1: Graphical representations of task focus.

taken during interruptions block performance of the interrupted task. For example, if Reina accidentally kicks all the marbles out the door on her way to the kitchen, it would no longer be possible to go back to the game. In the general case, competently handling such situations in collaborative agent design requires what are known as full causal models, containing preconditions, such as 'there are marbles with which to play', and postconditions, such as 'the marbles are back in their bag', specifying every possible situation in which the task is still possible to perform. Creating complete causal models can be difficult and time-consuming. In fact, one of the strengths of the planning algorithm in Lesh et al. (2001) is its ability to plan with partial models.

3) Go with the Flow: The third strategy in the table, *guessing the user is done*, does not add extra turns as does asking, and prevents nested interruptions. However, with this strategy there is also no context left around when the interrupting task is complete. For example, when Reina comes back to playing marbles after eating, Maya would not remember where they were in the marble game they were playing, nor that the marbles were still out and need to be put away. Moreover, this strategy has the potential disadvantage of increasing peoples' experience of lostness. Since the agent does not interfere with the user's task switching, the user may not be aware a switch has occurred. The user might want to know 'how do I get back to where I was?' Since there is no context left, the agent cannot answer the question.

SUPPORTING TASK FOCUS

As mentioned above, designing off-topic recommendations requires an understanding of agent dialogue strategy, recommendation interface design and human focus shifting behavior. The previous section described the first of these. Turning now to the second issue, there is little known about how to design an interface to help people usefully switch focus without confusing the product.

Existing efforts to design adaptive recommendations have focused on focusing the user, or at least confirming the user's current intentions. For example the 'intent interface' in (Miller and Hannen, 1999) gives the user a view on what the system thinks the user intends and lets the user override the interpretation, and the studies in (Sidner and Forlines, 2002; Freudenthal and Mook, 2003) presented sets of suggested things to say contributing to the user's current task. None of these efforts explicitly support the user in switching to a new task. They do not address situations when people might find value in switching between tasks (or subtasks) as part of normal problem solving, such as balancing heating comfort against energy costs, or planning a vacation the whole family could enjoy and afford.

A recommendation interface must support situations when people unexpectedly shift focus, and to encourage people to communicate about their current intentions. The **Some** Thi**n**gs to **Say** (SenSay) menu addresses this challenge. An example is shown in Figure 3. As described in DeKoven (2004), SenSay contents are generated through generic rules based on the collaborative planning algorithms described in Lesh et al. (2001). The system updates a *focus stack* and *plan tree* by comparing actions to a task model. Simple rules walk the stack and tree to create an *agenda* of likely next steps.

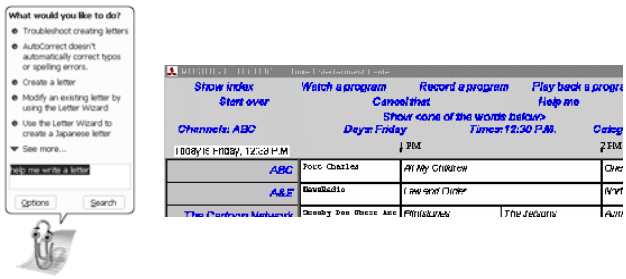


Figure 2: Two graphical interfaces for action recommendation. Left: Clippit agent from Microsoft Office XP; Right: part of a VCR interface described in (Sidner and Forlines, 2002).

For any reasonably complicated apparatus, such as a programmable thermostat, there could be many reasonable next steps the user could take. The decision concerning what to recommend rests with the agent. Typical recommendation systems hide the complexity and uncertainty from the user, and choose one best option from

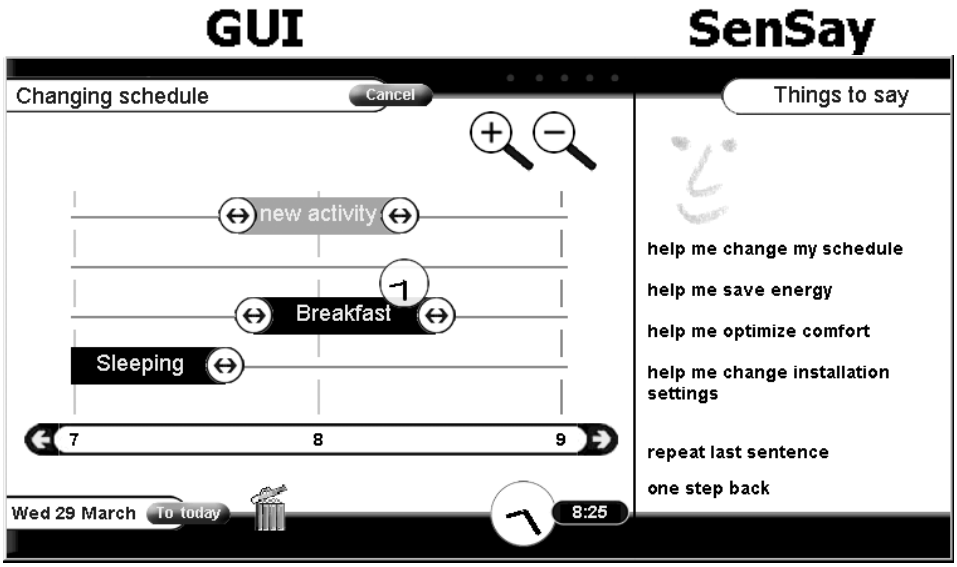


Figure 3: Example of a SenSay menu, in the case of a programmable thermostat, as described in Freudenthal and Mook (2003). The left-hand side of the interface is the basic graphical user interface (GUI), and the right-hand side contains the SenSay.

the agenda upon which to act next. In contrast, the SenSay presents the agenda as an ordered list of phrases the user can do or say to the system (through speech or directly pressing an item).

The SenSay can be viewed as a way to plug the user in to the system's reasoning process, giving the user means to explicitly communicate with the product about current intentions, in a way the system can understand. Since the SenSay allows the user to specify an intention, the system is not forced into making a single guess, nor does it have to ask as many clarifying questions. In effect, with the SenSay present the system can be satisfied with a certain degree of uncertainty. That is, the product can wait until it is more confident about what the user wants to do, while still being helpful.

A recommendation interface like the SenSay can make it easier for people to communicate their intentions to the thermostat. Furthermore, in order to support useful shifting of task focus, the SenSay can include suggestions about related tasks, even if not directly contributing to what the product thinks is the user's current task. By using the right rules for computing the agenda, the SenSay could present next steps for the current task, as well as ways to shift to completely different tasks. For example, while trying to increase home heating comfort by raising temperatures with a thermostat, the SenSay could include suggestions for saving energy by reducing temperatures.

It may be confusing or overwhelming to people if the SenSay were to include all possible focused and off-topic recommendations. It is a design decision as to which suggestions to include in a limited visual space like the SenSay. Finding the right balance requires knowing more about how people tend to shift focus when using a product, such as using a thermostat, to solve optimization problems, such as balancing comfort and costs.

TOWARDS A MODEL OF FOCUS SHIFTING

To reiterate, the goal of this paper is to discuss the design of recommendation systems to handle focus shifting, and offering off-topic suggestions. There is currently little empirical evidence related to how people might want or need to shift focus, and how they might communicate the focus shift with a product. There is even less known about how an interaction mechanism like the SenSay will affect the user's focus shifting and communication behavior.

As a starting point, Lesh et al. (2001) hypothesized that typically 90% of user actions are focused, 5% are focus shifting, and 5% are interruptions. However, there is little evidence upon which to validate this model. Quite likely, the model should be related to choices in agent design, as the different agent strategies described above might influence the results in different directions.

A case study was conducted in order to test this model (DeKoven, 2004). The full study incorporated tests of the SenSay as a multimodal (speech + touch) interface. Only those results relevant to modeling focus shifting and the utility of off-topic recommendations are reviewed here. Full results of the study, along with a more detailed analysis of SenSay item design and development, can be found in (DeKoven, 2004).

The study used an interface similar to that in Figure 3, translated into Dutch. The subjects were given a set of test tasks. Task order was random, except the last task. This task, called *Go Green*, was intended to force a situation in which subjects would need to balance comfort and costs to meet certain constraints.

In order to increase the likelihood of test participants switching focus, this experiment adopted the *Go with the Flow* strategy. One group of study participants used a SenSay containing focused and off-topic recommendations, and the other group used the same SenSay with only the

focused items, in a between subjects design. With this setup, it is possible to examine focus-shifting behavior, with and without the presence of off-topic recommendations, under the *Go with the Flow* strategy.

Across both conditions, subjects generally used the SenSay during the *Go Green* task more than during other tasks. As verified in post-test interviews, subjects were inclined to use the SenSay when they did not know what to do with the GUI or were looking for better ways to do something. The results indicate that recommendations in the SenSay can be utilized effectively in complex tasks like saving energy while staying comfortable, but can be distracting when the user can complete the task more quickly in the rest of the interface.

The test subjects did use the off-topic recommendations on the SenSay, though sometimes the items appeared to be confusing at times. On the other hand, several subjects in the focused-SenSay condition were not in the energy saving dialogues at the right time, and, unlike subjects in the focus-shift-SenSay condition, could only get there by starting over or otherwise manually exiting all the subtasks.

	Degree of focus	**St. Dev.**
Predicted (Lesh et al. 2001)	90%	
Observed: All tasks	96%	1.23
Observed: Go Green	89%	2.65

Table 3: Observed proportions of focussed behavior.

That is, the off-topic recommendations served as useful short cuts for flipping between tasks and strategies.

Table 3 summarizes the observed frequency of focused actions. Across all test tasks and conditions, approximately 96% (st. dev. = 1.23) of subjects' actions were focused (cases 1a and 1b), much higher than the prediction in (Lesh et al., 2001). Looking just at actions taken during the *Go Green* task, the last and most difficult of the test tasks, only about 89% (st. dev. = 2.65) of subjects' actions were focused. Thus the predictions in (Lesh et al., 2001) might be more accurate for more difficult user tasks, or more experienced users.

DISCUSSION

In order for an interactive agent to recommend useful next steps and options it needs to keep track of what the user is doing. This gets more difficult if the user shifts focus.

As discussed in this paper, people do shift between tasks and between strategies while working on constraint-solving problems, such as when using a thermostat to balance heating comfort against saving energy. Recommendation agents can help people navigate to better answers, but only if people and their products are focused on the same task and plan. This paper has discussed how to incorporate focus shifting into recommendation agent design.

The primary design question addressed in this paper is how to best balance guessing, asking and waiting when the agent is unsure of the user's current intention. This paper compared three possible response strategies an agent can adopt when the user shifts focus. In particular, the *Go with the Flow* strategy appeared the strongest of the three in terms of allowing for smooth task transitions. It was also the most likely strategy to engender a feeling of lostness. In the study reviewed here, subjects did not indicate strong feelings of lostness. That is, the agent being loose did not lead to the user feeling lost. Moreover, the study subjects used the off-topic SenSay items to quickly switch between alternate strategies. Thus we can say that *Go with the Flow* did in fact support the user in flexibly redirecting the dialogue, via the off-topic recommendations.

These study results need to be understood in terms of the study participants. Most of the participants were young and well educated (university students). Many of the older subjects used in pilot testing in particular had more difficulty completing test tasks. Given the results and designs of other similar studies (Freudenthal and Mook, 2003; Sidner and Forlines, 2002), more agent utterances or visual feedback confirming the focus shifts could have significantly helped subjects find their way with the SenSay. More studies are needed comparing all three strategies with respect to usefulness of off-topic recommendations.

Central to future research in this area is a baseline model of human focus shifting when using a product to solve constraint problems. The study reviewed here is the first to provide empirical evidence towards a predictive model of focus shifting such as that presented in Lesh et al. (2001). The model in that paper appears more correct for experienced usage and/or complicated tasks. More studies are needed to confirm this result. In particular, the same SenSay with a Presumptive or Nested Interruption agent might not lead to the same results. Longitudinal studies would be useful for tracking these relationships. Combining these results with user modeling (such as in Rickel et al., 2002) could lead to creating agents that adaptively alternate between the three strategies.

To what degree should an action recommendation interface support people in shifting focus? Clearly, the modalities and manners of expressing focus shifts impact the frequency of shifting focus. The SenSay discussed in this paper is only one such way to present off-topic recommendations. More design research is needed to better understand the impact of interface design, such as the SenSay, on use of action recommendations in general, and off-topic recommendations in particular.

While more studies are needed to verify the results discussed in this paper, there does appear to be support for the following propositions:

- People do shift focus when solving constraint problems. Moreover, they shift focus for different reasons as they get more experienced and as the tasks get more difficult.

- Graphical interfaces such as the SenSay help people in communicating with a product, at least initially (see Sidner and Forlines, 2003).
- Off-topic recommendations can be useful for problem solving as well as presenting unfamiliar product capabilities. However, they may be more useful for people more experienced with regular focused interface usage.
- Choosing strategies for action recommendation agents must be done in tandem with designing the rest of the interface.

As discussed in this paper, focus shifting is fine, and can even be good, as long as it is communicated. Interfaces for conversational recommendation agents need to motivate people to tell their products what they want to do. The main goal in the line of research leading up to this paper has been to design products to better help people tell the product what they want to do. This has been called the Help Me Help You principle (DeKoven, 2004).

Underlying this research has been the SharedPlans model of human-human collaboration (Grosz and Sidner, 1986), and the discourse interpretation algorithm based on SharedPlans defined in (Lesh, Rich and Sidner, 2001). Unlike most current recommendation agents that are based on fact databases (e.g. information about restaurants), collaborative agents use a task model to determine which actions to recommend. Future work should combine these two lines of work (as was attempted in Rickel et al., 2002) with the design research described in this paper, towards collaborative recommendation agents that can help people with both facts and actions.

ACKNOWLEDGEMENTS

This work was conducted at and supported by the Delft University of Technology. Thanks are due to David Keyson and Jans Aasman for their assistance in focusing the study, and to Marc de Hoogh and Tanja Veldhoen for assistance in preparing and running the study.

REFERENCES

Constantine, L. L., & Lockwood, L. A. D. (1999). *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. New York: ACM Press.

DeKoven, E. (2004), *Help Me Help You: Designing Support for Person-Product Collaboration*, Ph.D. Thesis, Delft University of Technology, Delft, The Netherlands.

Freudenthal, A., & Mook, H. J. (2003). "The Evaluation of an Innovative Intelligent Thermostat Interface: Universal Usability and Age Differences." *Cognition, Technology and Work, 5*, 55-66.

Grosz, B. J., & Sidner, C. L. (1986). "Attention, intentions, and the structure of discourse." *Computational Linguistics, 12*(3), 175-204.

Kuhme, T., Malinowski, U., & Foley, J. D. (1993). *Adaptive Prompting* (No. GIT-GVU-93-05). Atlanta, GA: Georgia Institute of Technology.

Lesh, N. B., Rich, C., & Sidner, C. L. (2001, July). *Collaborating with Focused and Unfocused Users Under Imperfect Communication.* International Conference on User Modelling, Sonthofen, Germany.

Miller, C. A., & Hannen, M. D. (1999). *User Acceptance of an Intelligent User Interface: A Rotocraft Pilot's Associate Example.* Intelligent User Interfaces (IUI'99), Redondo Beach, CA.

Rich, C., & Sidner, C. L. (1998). "COLLAGEN: A Collaboration Manager for Software Interface Agents." *User Modeling and User-Adapted Interaction, 8*(3/4), 315-350.

Rich, C., Sidner, C. L., & Lesh, N. B. (2001, Winter). "COLLAGEN: Applying Collaborative Discourse Theory to Human-Computer Interaction." *Artificial Intelligence Magazine, 22*, 15-25.

Sidner, C. L., & Forlines, C. (2002, September). *Subset Languages for Conversing with Collaborative Interface Agents.* International Conference on Spoken Language Processing (ICSLP'02), Denver, CO.

Thompson, C.A., Göker, M. H. & Langley, P. (2004). "A Personalized System for Conversational Recommendations." *Journal of Artificial Intelligence Research, 21*, 393-428.

Rickel, J., Lesh, N., Rich, C., Sidner, C.L. & Gertner, A. (2002, June). Collaborative Discourse Theory as a Foundation for Tutorial Dialogue. In *Proc. Sixth International Conference on Intelligent Tutoring Systems*, pp. 542-551, Springer.

Item-Triggered Recommendation for Identifying Potential Customers of Cold Sellers in Supermarkets

Han-Shen Huang[1] **Koung-Lung Lin**[1,2] **Jane Yung-jen Hsu**[2] **Chun-Nan Hsu**[1]
[1]Institute of Information Science, Academia Sinica Taipei, Taiwan 105
[2]Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan 106

ABSTRACT

Recommendation has achieved successful results in many applications. However, for supermarkets, since the transaction data is extremely skewed in the sense that a large portion of sales is concentrated in a small number of hot seller items, collaborative filtering recommenders usually recommend hot sellers while rarely recommend cold sellers. But recommenders are supposed to provide better campaigns for cold sellers to increase sales. In this paper, we propose an alternative "item-triggered" recommendation, which aims at returning a ranked list of potential customers for a given cold-seller item. This problem can be formulated as a problem of rare class learning. We present a Boosting-SVM algorithm to solve the rare class problem and apply our algorithm to a real-world supermarket database. Experimental results show that our algorithm can improve from a baseline approach by about twenty-five percent in terms of the area under the ROC curve (AUC) metric for cold sellers that as low as 0.7% of customers have ever purchased.

Keywords

Recommendation, Cold Seller, SVM, Boosting, ROC

INTRODUCTION

Recommender systems (RSs) have achieved successful results in many applications. We can see them work in our daily lives. A good RS can help increasing sales at on-line merchants as well as brick-and-mortar retailer stores. Examples include net news [15], on-line shopping [13], TV programs [19, 1], etc.. Previous work can be classified into three categories: content-based filtering (e.g., [3]), collaborative filtering (CF) (e.g., [15]) and hybrid solutions (e.g, [14]).

In this paper, we address the problem of applying CF recommender system in a brick-and-mortar supermarket. This problem is challenging for current RSs in that the transaction data is extremely skewed in the sense that a large portion of sales is concentrated in a small number of items. Figure 1 shows the sales distribution of items sold in a supermarket. In the figure, items are sorted and re-numbered by the order of their sales figures in a given period of time. The sales figure here is the amount of the items purchased by the customers. The plot shows that the sales distribution is skewed and concentrated on a very small portion of product items. This is typical to retailer stores and is an example of "*the 80-20 rule*" known in business management. A trivial recommender that always recommends hot sellers to any customer can achieve pretty accurate prediction of the customers' shopping preference. In fact, it is difficult to improve from the trivial recommender because it is difficult for a recommender to identify potential customers for the items in the tail of the curve. From the point of view of the supermarket which pays to deploy a RS, it is much more useful for a RS to recommend those *cold sellers* accurately than recommend hot sellers accurately.

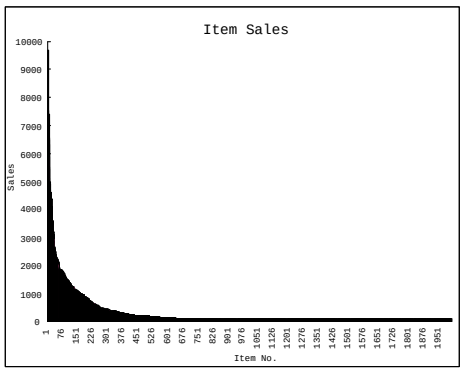


Figure 1: Sales distribution of items in Ta-Feng retailer store

In this paper, we propose *item-triggered* recommendation, an alternative view of recommendation. Previous RSs are *customer-triggered* in that they return a list of items as the recommendation for each customer. In contrast, item-triggered RS will return a potential customers list for each cold seller. A given proportion of the customers from the top of the list will then receive the recommendation to buy that cold seller item. If this can be done accurately, the RS

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

will help increasing sales of those cold sellers, which are in need of better campaigns. Previously, Sarwar et al. [16] proposed an item-based approach to recommendation. Item-based recommendation is not item-triggered because it is basically customer-triggered and still aims at returning a list of items for each customer.

The problem specification of item-triggered recommendation is, given a cold seller, estimating the probability that a customer will buy that item given a set of features of that customer. We can apply a binary classifier that returns a confidence score of its classification result (i.e., will buy or will not buy) to solve the problem. Previously, many classifier learning algorithms have been applied to recommendation (see e.g. [2], [23], [7]). Among many classifiers, we chose SVM because SVM can handle sparse data better than other classifiers [20]. Transaction databases in supermarkets are very sparse in that each customer only purchased a very small subset of the entire set of available items. Therefore, if we use transaction data as the features, we will need a classifier that can handle sparse data. A variation of SVM in LIBSVM [4] can output the probability of its classification results [21]. We will use that version of SVM from LIBSVM to solve our problem. Since we aim at identifying potential customers for cold sellers, the training data for the SVM will be very *imbalanced* — by definition of the cold seller, only a very small portion of customers have purchased the item. As a result, the ratio of positive data (customers who have purchased the item) and negative data (customers who did not purchase the item) will be very small. This problem is known as the *rare class* problem in machine learning. SVM alone cannot handle imbalanced training data. We propose a boosting algorithm to train an ensemble of SVMs to handle imbalanced data. The idea is to extract a subset of the training data such that the subset is less imbalanced and has more incorrectly classified data so that each SVM in the ensemble is trained using different combination of positive and negative data. In this way, the combination of the SVM ensemble can provide a finer classification boundary to separate positive and negative data.

We will use the AUC metric to measure the quality of the output list of potential customers by our new RS. See [8] for its use in RSs and [5] in machine learning. The AUC metric takes into account both false positives and false negatives and is suitable to measure the quality of a ranked list. Other metrics that focus on evaluating individual recommendation, such as accuracy or absolute deviation, are not appropriate here because the score will be high if all the customers are predicted as not going to buy cold sellers.

In our experiments, the SVM ensembles were compared with sorting customer lists by their shopping frequency. The latter serves as a baseline for item-triggered recommendation. The results show that the SVM ensemble outperforms the baseline one by increasing the AUC of the latter by 25%.

This paper is organized as follows. First, we describe the background motivation and the problem definition of item-triggered recommendation. Next, we present the Boosting-SVM algorithm and experimental results. At last, we discuss related work and conclusion.

ITEM-TRIGGERED RECOMMENDATION

In 2001, we had a chance of collaboration to develop a personalized shopping recommender with Ta-Feng, a large retailer store in Taiwan that sells a wide range of merchandise, from food and grocery to office supplies and furniture. After surveying a variety of technologies, we agreed a specification of the recommender. The specification requires that for each customer, the recommender should produce a ranked list of items in the order of the customer's preference, given his/her historical shopping records.

The transaction data set from Ta-Feng contains the transactions collected in a time span of four months, from November, 2000 to February, 2001[1]. Each record consists of four attributes: the shopping date, customer ID, product ID, and the amount of purchase. Shopping records with the same customer ID and the same shopping date are considered as a transaction. There are 119,578 transactions and 32,266 distinguishable customers in this data set. Ta-Feng adopts a common commodity classification standard that consists of a three-level product taxonomy. Products are classified into 201 product classes and 2012 sub-classes.

Figure 1 shows the sales distribution plot of the items sold at Ta-Feng according to the transaction data set described above. The plot shows that the sales figure is skewed and concentrated on a very small portion of product items, which is typical for supermarkets. The skewness of the data makes it easy to accurately recommend a hot seller item but difficult to identify potential customers for the items in the tail of the curve, that is, the cold sellers.

Our previous work shows that a probabilistic graphical model can be effective in handling skewed and sparse data [9]. By casting CF algorithms in a probabilistic framework, we derived HyPAM (Hybrid Poisson Aspect Modeling), a novel probabilistic graphical model for personalized shopping recommendation. Experimental results show that HyPAM outperforms GroupLens [15] and the IBM method [11] by generating much more accurate predictions of what items a customer will actually purchase in the unseen test data. HyPAM also outperforms the "default" method — the trivial recommender that always recommends the best sellers to any customer. However, when we compared the items recommended by HyPAM and the trivial recommender, we found that the difference is not that obvious. HyPAM can tailor to a customer's need by recommending some cold sellers but most of the times hot sellers are at the top of the recommendation. In fact, if we evaluate a RS's performance by comparing its recommendation with the un-

[1]The data set is available for download at the following URL: http://chunnan.iis.sinica.edu.tw/hypam/HyPAM.html.

seen data in the transaction data set, given the skewness of the data, a perfect RS must recommend cold sellers less often. This implies that our original problem formulation, the "customer-triggered" recommendation, and the evaluation metrics, will not lead to large sales increasing for cold sellers, but cold sellers provide a wide-open opportunity of large sales increasing.

This is why we propose an "item-triggered" recommendation approach. Rather than recommending a list of items to each customer, the item-triggered recommender outputs a customer list ordered by the probability that the customers are willing to buy a given item. An accurate predictor of customers' shopping preference may improving customers' shopping experience and indirectly increase the sales, but increasing sales of cold sellers can contribute directly and justify the investment of deploying a RS by the supermarkets. Item-triggered RSs can be complementary with customer-triggered ones by inserting cold sellers to the predicted shopping list of the potential customer. In this way, we can address not only the needs of the customers but also their unique needs that are shared with a very small number of other customers.

BOOSTING SVM

As we described in the introduction section, we can formulate the problem of item-triggered recommendation as a classifier learning problem, but we will need to face the "rare class" problem. This section presents our preliminary solution to item-triggered recommendation. Our goal is to develop a classifier for each cold seller item to accurately classify whether a customer will or will not buy the item with a probability. Ordered by the probability, the customers constitute a ranked list in the order of the likelihood that they will buy the item.

The training data of the classifier is the data of customers that we already know whether they bought or did not buy the given item. A customer who bought the item will be treated as a positive example and negative otherwise. Clearly, the training data will be very imbalanced because for cold sellers, there will be many negative examples and very few positive examples. In our experiment, the ratio of positive and negative examples is about as low as 2%. There are really "cold" sellers in our transaction data with an extremely low ratio. If the ratio for an item is lower than 0.7%, that is equivalent to having less than 100 customers in four months, then we will not consider the item here because it is too difficult to derive anything from the transaction database for this item and the item might not worth being recommended to customers at all.

We will evaluate the trained classifier with a test data set of customers. This test data set is disjoint with the training data set. Given a threshold of probability, we can divide the output customer list by the classifier into two sets: one set contains customers with the probability to buy the item higher than the threshold, and the other set contains customers with the probability lower than the threshold. Customers in the former set is predicted positive (i.e., will purchase the item) and the latter set is negative (i.e., will not buy the item). Then we can compare the positive and negative sets with the real class label in the test data set and calculate recall and precision. By adjusting the threshold, the classifier will yield different recall and precision and we can apply the AUC metric [5] to evaluate the quality of the predicted customer list. A classifier will have a high AUC score if those customers who actually buy the item are ranked at the top of the list. A perfect classifier will have AUC score equal to one while random guess will yield 0.5 AUC score. The baseline for our performance evaluation is shopping frequency. For any item, if a customer visits and shops at the supermarket more often, then the customer is predicted as more likely to buy the item. This baseline strategy can yield a better AUC score than random guess.

We have tried a standard SVM as the classifier, since SVM can handle sparse and high dimensional data better [20]. However, we found that the resulting recall is quite low due to the skewed transaction data. In many cases, the SVM simply predicts that nobody will buy the given item. This yields a very low error rate but clearly is not desirable. Instead, we chose LIBSVM 2.6 [4] because it includes a reliable SVM variant that can output classification probability [21]. With the probability, SVM can return a ranked list of potential customers. Also, by adjusting the threshold, at least we will have some customers predicted positive.

Though LIBSVM can reduce the impact of imbalanced data for SVM, its performance is barely better than shopping frequency. We altered the ratio of positive and negative examples in the training data and found that training data with a higher ratio tended to help improving the performance of LIBSVM. This led to the idea that we may apply *boosting* [17], a well-known technique in machine learning, to improve the performance by controlled re-sampling. The basic idea of boosting is to sample a subset of training data to train a "weak learner," in this case, SVM. Two parameters determine how sampling will be performed: the ratio of positive and negative examples and the classification results by the classifier trained in the previous iteration. The sampling iterates a constant number of times to yield as many SVM classifiers that constitute a classifier ensemble.

We now formally present our Boosting-SVM algorithm. Let $D = \{x_1, y_1, \ldots, x_N, y_N\}$ denote the training examples, where x_i is the feature vector of customer i and y_i indicates whether the customer bought the given item. $W_t(i)$ is the probability that (x_i, y_i) will be selected to train the t-th classifier $h_t(x)$. M is the number of examples that will be selected for $h_t(x)$ and T is the number of classifiers that will be trained in total. The Boosting-SVM algorithm is defined as follows:

To classify a customer, the trained classifiers will be combined linearly with $\alpha_1, \ldots, \alpha_T$, the weights of

Algorithm 1 Boosting-SVM

1: Initialize $D = \{x_1, y_1, \ldots, x_N, y_N\}$, T, M, W_0 and $t = 1$;
2: Let Z_t be a normalization constant
3: $W_t(i) = \frac{W_0}{Z_t}$ if $y_i = +1$, (positive examples)
4: $W_t(i) = \frac{1}{Z_t}$ if $y_i = -1$; (negative examples)
5: **while** $t \leq T$ **do**
6: Train $h_t(x)$ using D sampled according to W_t;
7: Calculate α_t for $h_t(x)$ based on D;
8: Calculate *err* for $h_t(x)$ based on D; (error rate)
9: $W_{t+1}(i) = \frac{W_t(i)}{Z_{t+1}} \exp(-(1 - \mathit{err}))$, if $h_t(x_i) = y_i$; (correctly classified cases)
10: $W_{t+1}(i) = \frac{W_t(i)}{Z_{t+1}} \exp(1 - \mathit{err})$, if $h_t(x_i) \neq y_i$; (incorrectly classified cases)
11: $t = t + 1$;
12: **end while**
13: Return $\{h_1, \alpha_1, \ldots, h_T, \alpha_T\}$;

$h_1(x), \ldots, h_T(x)$, respectively. That is, the probability that customer i will buy the item is estimated as:

$$P(y_i = +1) = \sum_t \alpha_t P(h_t(x_i) = +1). \quad (1)$$

There are many possible ways to calculate the weights α_t. We have tried 3 methods to calculate the weights and varied constant W_0 to adjust the initial sampling probability. This yields different variants of Algorithm 1 so that we can empirically determine their impact on the performance.

EXPERIMENTAL RESULTS

This section reports the experimental evaluation of our item-triggered recommendation approach. We randomly selected 15,000 customers from Ta-Feng data set to train the recommenders and a disjoint set of 1,000 customers as the test set. There are $F = 2,012$ product subclasses in the Ta-Feng transaction database. They constitute the items in our recommendation task. For a selected item, we converted the records of customer i in the transaction database into the form (x_i, y_i). The shopping records of the selected item were only used to assign y_i, and omitted when we generated x_i. Each x_i is a feature vector of length F and $x_i(j)$, the j-th feature, is one if the customer has bought item j within four months and zero otherwise.

For each item, we applied the following variants of SVM and the boosting algorithms to produce the ranked lists.

DEFAULT : This is the trivial algorithm that outputs a customer list sorted by the shopping frequency of each customer. Basically, if a customer comes more often, the prior probability that the customer will purchase a cold seller is higher. Note that this algorithm generates the same customer list for any item. The performance of DEFAULT is treated as the baseline of a qualified recommendation algorithm.

SVM15000 : SVM is trained by all the training data. This algorithm represents the strategy that we do not consider the rare class problem.

P+SVM : SVM is trained by all the positive examples and randomly selected negative examples, where the number of negative examples is twice as many as the positive ones. In other words, P+SVM will use only 2.1% to 8.1% of the training examples, but the distribution of positive and negative examples is different from the original training data set. This algorithm represents another strategy: we balance the positive/negative ratio and train a single classifier to rank customers.

U+SVM : This algorithm is the same as P+SVM except that the sample distribution is uniform. Therefore, the training examples of U+SVM are a subset of the original training data set with the same imbalanced distribution of positive and negative examples.

U+E, U+U : These two are the instantiations of the Boosting-SVM algorithm as defined in Algorithm 1 with uniform initial sampling probability (i.e., $W_0 = 1$ in Algorithm 1) for the whole training data set. In addition, the weight of a classifier is calculated differently. U+E calculates the weights using $1/2 \ln(\text{Accuracy}/\text{Error Rate})$, the Adaboost's formula [18], and U+U uses uniform weights for all classifiers.

U+ROC : This algorithm uses the same initialization of the sample distribution as U+E and U+U, but calculates the weights using the AUC scores of the classifiers against training data.

P+E, P+U, P+ROC : These three algorithms assign a high sampling probability for positive examples. More specifically, we set $W_0 = 100$ in Algorithm 1. Their weight calculation methods are the same as U+E, U+U and U+ROC, respectively.

We divided the cold seller items into four sets according to the number of customers who have purchased them in the training data: (A) 100–149, (B) 150–199, (C) 200–299 and (D) 300–399. Since there are a total of 15,000 customers, the corresponding percentages of buyers range from 0.7% to 2.7%. There are 120 items in the item set (A), 77 in (B), 88 in (C) and 59 in (D). Table 1 reports the experimental results of the algorithms for each of the item sets. The performance is measured by the average AUC scores. The results show that all algorithms outperform DEFAULT except U+SVM, which used less than 8.1% of the training examples. Since we fetched the positive and negative examples with the same probability, U+SVM is trained by a small and imbalanced data set with the same ratio of positive and negative examples. For SVM15000, we used the whole data set for training and obtained larger AUC scores. This shows that LIBSVM

Table 1: Average AUC scores of the nine approaches in the four unsought item classes. Best results in a data set are bolded.

Data set	DEFAULT	SVM15000	U+SVM	P+SVM	U+U	U+E	U+ROC	P+U	P+E	P+ROC
100–149	0.609	0.645	0.531	0.694	0.638	0.638	0.657	0.757	0.757	**0.758**
150–199	0.612	0.639	0.540	0.676	0.666	0.666	0.681	0.756	**0.757**	**0.757**
200–299	0.613	0.633	0.581	0.691	0.726	0.726	0.731	**0.763**	0.763	**0.763**
300–399	0.616	0.641	0.611	0.690	0.728	0.728	0.729	**0.753**	0.752	**0.753**

can outperform DEFAULT when the training data set is sufficiently large.

Then, we discuss the influence of imbalanced positive and negative examples. SVM15000 and P+SVM apply the same learning and predicting algorithms. The only difference is that P+SVM used only a small portion of the negative examples. The whole training data size for P+SVM is the same as U+SVM. From the experimental results, we can see that the AUC scores of P+SVM are larger than U+SVM and averagely about 0.06 more than those of SVM15000. The results show that using a small but more balanced data set is better than using a large but extremely imbalanced data set.

The experimental results also show that adopting ensembles of classifiers can enhance the performance of weak classifiers. For U+ROC, U+E, and U+U, the classifier learned in the first iteration is the same as the classifier learned by U+SVM. After several iterations, the weights of positive examples will be increased so that the subsequent data sets will become more balanced and therefore, U+ROC, U+E, and U+U can outperform U+SVM. P+ROC, P+E, and P+U perform better than U+ROC, U+E, and U+U because their initial training data set is more balanced. Consequently, the best results for all item sets are produced by P+ROC, P+E, and P+U, which improve from DEFAULT by 25% in terms of the AUC scores.

The ROC curve allows us to see how many buyers will be identified by different algorithms from their corresponding recall scores. Figure 2 shows the averaged ROC curves of DEFAULT and P+ROC for the items in the item set (A), the "coldest" among the item sets of the cold sellers. Each curve consists of 100 data points, representing 100 cutoff points to divide the customers ranked by the predicted probability that they will buy the items. Suppose that we are recommending an item to the customers at the top 10% of the ranked list. Since the recall values of DEFAULT and P+ROC are 20.9% and 42.7% at that point, respectively, we can expect that P+ROC will help us identifying twice as many potential buyers as those by DEFAULT.

With the ranked list of potential customers, marketing staffs can design a campaign strategy targeting a certain percentage of customers at the top of the list. The optimal percentage can be determined by maximizing a utility function that takes into account many factors such as resource available for the campaign, supply and stock status of the item, etc. The ranked list can also complement recommendation made by CF recommenders by recommending more cold sellers to further increase sales.

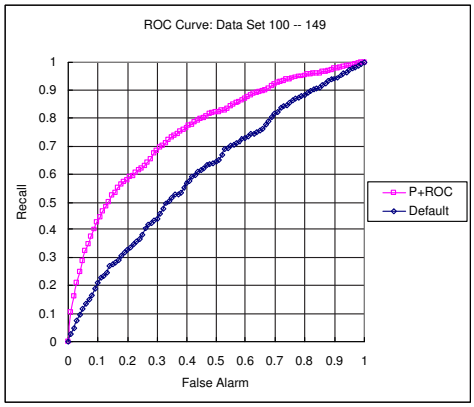


Figure 2: Comparison of the ROC curves of DEFAULT and P+ROC.

RELATED WORK

Support Vector Machines

In this paper, we combine SVM and boosting to train the RS. Originally proposed by Vapnik [20], SVM learns a hyperplane to separate positive and negative examples. The hyperplane is oriented from the maximal margin between positive and negative classes so that the risk of misclassification is minimized.

One of the approaches to the rare class problem for SVM is sample balancing, hierarchical SVM framework [22]. In this work, negative examples are divided uniformly and combined with all the positive ones to train a set of SVM classifiers. A top level SVM then takes their classification results as the input to produce the final classification result. Unlike their approach, we apply boosting and linear combination to combine the ensemble of SVM classifiers.

Boosting

Boosting [17] uses re-sampling techniques to learn a set of classifiers, and linearly combine them to predict the class of input data. The probability that an example is chosen to train a classifier is determined by whether its true class can be correctly predicted or not by the classifier learned in the previous iteration.

Many boosting methods have been proposed for general or specific purposes. One of the most well-known algorithm is AdaBoost [18], which minimizes the error rate of the whole

training data set without imposing any restriction to the training data. When the training data is imbalanced, AdaUBoost, a variant of AdaBoost, suggests that minority examples be initialized with higher weights and lower updating rate when they can be correctly classified [10, 12]. DataBoost-IM, another method to deal with the rare class problem, is to synthesize more positive examples using the previously learned classifiers [6]. Currently, we adopt the method similar to AdaUBoost to initialize weights of training examples.

CONCLUSION

We have presented our item-triggered recommendation approach that predicts customer lists for cold sellers. Each customer list contains customers sorted by their probability to purchase the corresponding item. We believe that item-triggered recommendation can complement CF-based customer-triggered recommendation by recommending cold sellers to further increase sales. For cold sellers, we will need to deal with the rare class problem to train the RS. Experimental results show that our approach, combination of SVM and boosting, seems promising for this problem. From the experimental results, we conclude that in terms of the AUC scores, (1) SVM outperforms shopping frequency; (2) using balanced positive and negative examples is better than using imbalanced ones for SVM; (3) SVM ensembles perform better than the variants with a single SVM.

Our future work including three issues: To further enhance the training algorithm for cold seller recommendation. To include other information such as demographical data to improve the recommendation. To identify the similar or relative items then re-use training models between them.

REFERENCES

1. L. Ardissono, C. Gena, P. Torasso, F. Bellifemine, A. Chiarotto, A. Difino, and B. Negro. Personalized recommendation of TV programs. In *AI*IA 2003: Advances in Artificial Intelligence*, pages 474–486, 2003.
2. D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *ICML'98*, pages 46–54, Jul 1998.
3. D. Billsus and M. J. Pazzani. User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10(2-3):147–180, 2000.
4. C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
5. P. Flach. The many faces of ROC analysis in machine learning, Tutorial in ICML2004., 2004.
6. H. Guo and H. L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *SIGKDD Explor. Newsl.*, 6(1):30–39, 2004.
7. E. F. Harrington. Online ranking/collaborative filtering using the perceptron algorithm. In *ICML2003*, pages 250–257, 2003.
8. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.
9. C.-N. Hsu, H.-H. Chung, and H.-S. Huang. Mining skewed and sparse transaction data from personalized shopping recommendation. *Machine Learning*, 57:35–59, 2004.
10. G. Karakoulas and J. Shawe-Taylor. Optimizing classifiers for imbalanced training sets. In *NIPS'98*, 1999.
11. R. D. Lawrence, G. S. Almasi, V. Kotlyar, M. S. Viveros, and S. Duri. Personalization of supermarket product recommendations. *Data Mining and Knowledge Discovery*, 5:11–32, 2001.
12. J. Leskovec and J. Shawe-Taylor. Linear programming boosting for uneven datasets. In *ICML2003*, pages 351–358, 2003.
13. G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.
14. M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, 13(5-6):393–408, 1999.
15. P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proc. of ACM Conf. on Computer Supported Cooperative Work*, pages 175–186, 1994.
16. B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proc. of the 10th Int. Conf. on World Wide Web*, pages 285–295. ACM Press, 2001.
17. R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.
18. R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.
19. B. Smyth and P. Cotter. Personalized electronic program guides for digital TV. *AI Magazine*, 22:54–61, 2001.
20. V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.
21. T.-F. Wu, C.-J. Lin, and R. C. Weng. Probability estimates for multi-class classification by pairwise coupling. *J. Mach. Learn. Res.*, 5:975–1005, 2004.
22. R. Yan, Y. Liu, R. Jin, and A. Hauptmann. On predicting rare classes with SVM ensembles in scene classification. In *IEEE ICASSP2003*, volume 3, pages 21–24, 2003.
23. T. Zhang and V. S. Iyengar. Recommender systems using linear classifiers. *J. Mach. Learn. Res.*, 2:313–334, 2002.

The Good, Bad and the Indifferent: Explorations in Recommender System Health

Benjamin J. Keller and Sun-mi Kim
Department of Computer Science
Eastern Michigan University, Ypsilanti, MI 48108
bkeller@emich.edu, skim8@emich.edu

N. Srinivas Vemuri and Naren Ramakrishnan
Department of Computer Science
Virginia Tech, Blacksburg, Virginia 24061
nvemuri@vt.edu, naren@cs.vt.edu

Saverio Perugini
Department of Computer Science
University of Dayton, Ohio 45469
saverio@udayton.edu

December 7, 2004

Introduction

Our work is based on the premise that analysis of the connections exploited by a recommender algorithm can provide insight into the algorithm that could be useful to predict its performance in a fielded system. We use the jumping connections model defined by Mirza *et al.* [6], which describes the recommendation process in terms of graphs. Here we discuss our work that has come out of trying to understand algorithm behavior in terms of these graphs. We start by describing a natural extension of the jumping connections model of Mirza *et al.*, and then discuss observations that have come from our studies, and the directions in which we are going.

Jumping Connections Revisited

Mirza *et al.* define a model that describes algorithms based on user-similarity, such as the nearest neighbor algorithms described by Herlocker *et al.* [2]. The ratings data correspond to a directed, weighted, bipartite graph called the *rating* graph in which vertices are users and items, and arcs are the ratings. Fig. 1 shows the subgraph of a rating graph involved in computing a nearest neighbor prediction of item a for user p. A *social network* is formed the ratings by using the users as vertices, and using a similarity measure (and possibly filtered by a threshold) to determine the edges. Mirza *et al.* use commonality of ratings to define a *hammock* measure of similarity where a threshold can be used to indicate the minimum number of ratings that must be common. The *recommender* graph is formed by adding the ratings back into the social network, and is the space in which predictions are computed. Fig. 2 shows the recommender graph for the neighborhood in Fig. 1.

Sarwar *et al.* [8] introduce *item-based* nearest neighbor algorithms, which in the graph model is just a dual construction. The item-based analogue to the social network is formed by using the dual similarity relationship between items, which forms an *artifact* network. The artifact network can be extended to a (item-based) recommender graph by adding the ratings as shown in Fig. 3.

Observations

Our work coming out of the experiments reported by Mirza *et al.* has dealt with analysis of the social and artificial networks, and trying to relate graph structure to algorithm performance. There are three key points to our work so far: (1) ignoring ratings is not useful in studying algorithms that employ them, (2) there is some significance of the graph structure to accuracy, but (3) what that influence is, is not yet clear.

Ratings change everything. The experiments described in Mirza *et al.* [6] showed that there is a place for studying recommendation based on commonality of

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

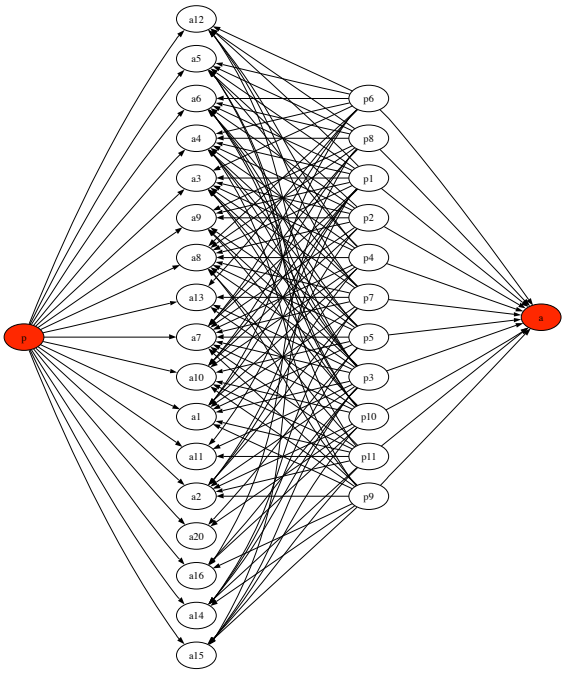


Figure 1: Subgraph of rating graph for prediction of item a for user p.

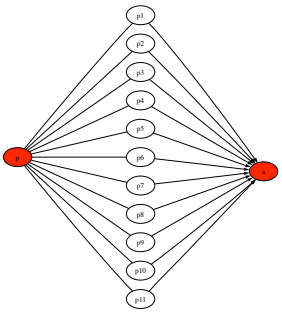


Figure 2: Subgraph of user-based recommender graph for prediction of item a for user p.

ratings. In the case of movies, we know that a few users tend to rate a lot of movies, so in a user-based algorithm, these users play an important role in forming the social network by making it possible for users with relatively few ratings to get recommendations (see the results on the minimum rating constraint in Mirza *et al.* [6]). Looking at commonality therefore allows us to understand how what people rate is important. However, if we look at the properties of the graphs induced by commonality and those induced by similarity measures based on the ratings, we see that the ratings change everything.

Just to illustrate the point, consider the plots of degree correlation for the social network based on commonality and the social network based on the Pearson r correlation shown in Fig. 4 and Fig. 5. Degree correlation measures the similarity of the degrees of adjacent vertices [7]. We should first caution that the x-axis in these plots are different, and so the comparison is slightly dangerous (which is part of the point). Both plots show the effects of filtering the edges of the graph by increasing a threshold (minimum items in common, and minimum correlation). The figure shows that the edges in the commonality-based social network are only between vertices of dissimilar degree, which suggests that users who rate many movies are serving as hubs for users who have not rated many movies. The plot for the Pearson similarity network, on the other-hand shows a phase-shift in the connectedness of the graph that indicates most connections between users of dissimilar degree have low correlations.

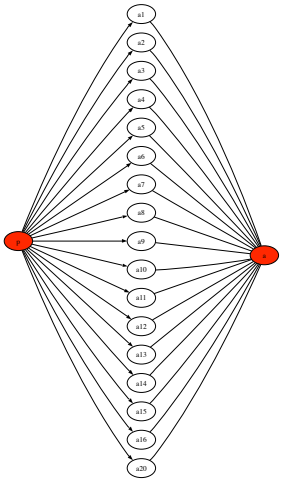


Figure 3: Subgraph of and item-based recommender graph for prediction of item a for user p.

Neighborhood structure does affect predictive accuracy. A question that had been posed to us in several settings was whether we could say something interesting about predictive accuracy through the graph structure. We first attempted a "jacknife" study using the 100,000 rating MovieLens data set, where for each user-rating pair we cut out a user's rating of an item and then predicted the rating. The results were inconclusive, so we took a different approach, which led to Srinivas Vemuri's thesis [9].

The approach in this case was to introduce a structural filter on the neighborhood and then measure the affect in terms of predictive accuracy. The filters applied were basically the requirement that two neighbors of the user are only kept if they are neighbors of each other — thus

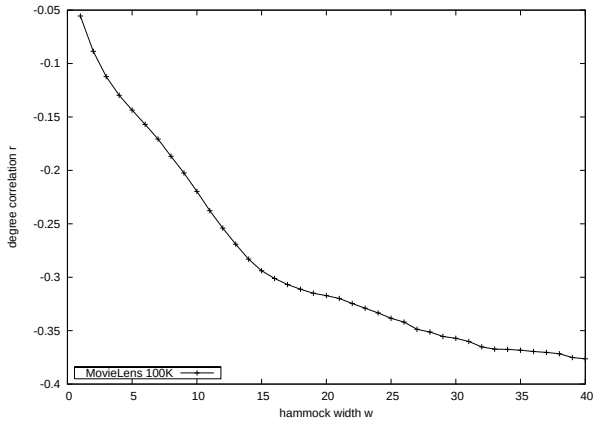


Figure 4: Plots of degree correlations for hammock social network (based on commonality) of MovieLens 100,000 rating data set.

forming a triangle. Vemuri was able to demonstrate an improvement in predictive accuracy (see Fig. 6), but at the price of loss of coverage. However, he also defined an approach that reweights the neighbors based on their involvement in triangles that produces similar results without the loss of coverage.

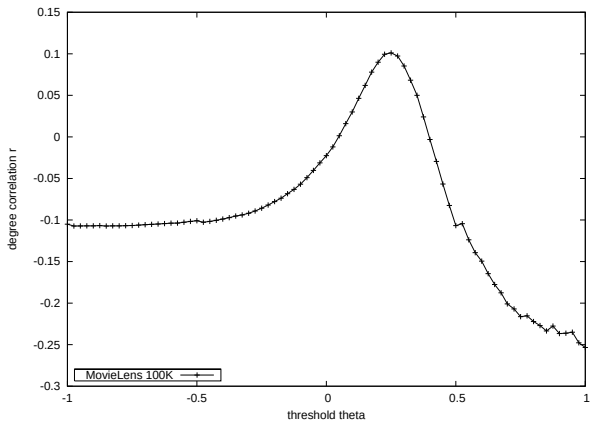


Figure 5: Plots of degree correlations for Pearson social network of MovieLens 100,000 rating data set.

Good neighborhoods don't always have good structure. The use of the triangular filter is based on the assumption that having better connected neighbors necessarily leads to better predictions, or, at least, eliminates the bad ones. However, further analysis of the results of applying the filters shows that the filters are somewhat indiscriminate, and make some predictions better, some worse, some impossible, and have no affect on the majority of predictions. Fig. 7 shows a typical configuration (although smaller than most) of a neighborhood affected by the triangle filters. In some cases, the loss

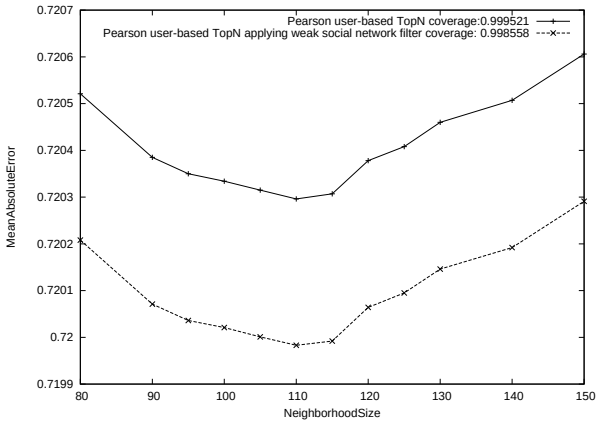
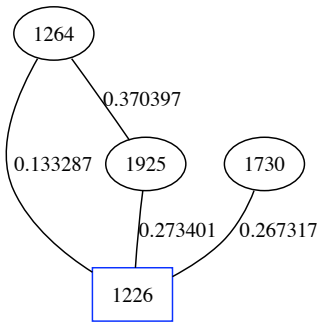


Figure 6: MAE versus neighborhood size for top-N user-based Pearson algorithm with and without triangles for 1 million rating MovieLens data set.

of the neighbor not involved in a triangle improves accuracy and in others makes it worse. The reason that the overall predictive accuracy improves is that the number of bad predictions lost exceeds the number of good predictions made worse or lost. However, the number of predictions changed is small (1% or less), unless a high threshold is used to define the triangles.



Prediction- 4.69065 Rating- 4

Figure 7: Typical neighborhood for a prediction affected by triangle filters.

Directions

The following describes work that is ongoing following the observations described above.

Recommendation Metrics

In considering the outcome of Vemuri's work on filtering neighborhoods, another question arose concerning whether the improvement in mean absolute error was significant. In particular, the issue of whether users would notice the minor improvement was raised. Of course, the problem with predictive accuracy as a metric of recommendation is that it has little to do with the way in which recommendations are typically presented. A user is presented a top-N list of items in decreasing order by prediction, and, in this setting, an error in prediction is only significant if it is noticeable by the user (either before or after the fact). Therefore, we are looking at recommendation metrics in terms of observability of errors by a user.

In looking at recommendation list metrics, we assume that we are measuring error over a test set of items that the user has rated (or ranked). Therefore, we can form a list of the user's ratings for these items, ordered by the prediction of the algorithm for each item. For instance, if the user rated five items 5, 5, 4, 3, 1, 1 and the algorithm predicted that they would be rated 3, 4, 4, 5, 2, 3, then we would consider the list 3, 5, 4, 5, 1, 1. A metric then measures the cost of sorting the list so that the ratings are ordered properly. The simplest metric is the count of the number of inversions, which is the cost of performing a bubblesort on the list. This is not a novel approach, since inversions are the basis of Kendall's tau, however, it matches the intuition behind comparing top-N lists. (Fagin *et al.* have shown Kendall's tau [4] is equivalent in a meaningful way to other reasonable choices by which top-N lists could be compared [1].) Vemuri [9] has also suggested counting the number of inversions between sorted runs of ratings as an alternative.

The problem of comparing comparing recommendation lists is much more complex than comparing the order, because the lists are presented in pages, and a user will also only view a prefix of the recommendation list [5, 3] Therefore, we might also consider factors such as the number of items presented per page, the total number of items (or pages) viewed by the user, and the user's tolerance for errors. In its simplest form, user tolerance can be modeled as an equivalence between rating values, since this would hide short swaps. However, a user's tolerance for longer swaps might be affected by whether they cross a page, or whether they include or exclude an item from the prefix of the list. It is not clear the extent to which these might be factors that are important to consider in the metric, and we are working on defining a user study that might help understand what a user might be able to observe (or care about).

Local Health

All of this work has led us in the direction of studying the "health" of a recommender system, and to begin with the local health of the system. We consider the *local* health of a recommender system as any property of the system that could affect the user's perception of the system, and *observability* by the user is a key property. Our goal is to define the user observable properties of recommender systems, and to characterize the underlying properties of the algorithm and data that lead to pathologies observable by the user. We concentrate on user observable properties of the recommendation list, including list accuracy, list stability, and variability of new items. This is the topic of Sun-mi Kim's research.

We have started by exploring the issue of what makes a good neighborhood from the standpoint of recommendation list error. To do this, we have identified users who have sufficient diversity in their use of ratings (an entropy value of 2 or more) and have either good or bad inversion rates, and have been studying their neighborhoods. We are starting with the obvious pathologies of the neighborhoods that lead to errors, and hope to find graph properties that we can use as measures of neighborhood quality.

For the other properties, we are following a similar approach to find alternative graph-based metrics that are descriptive. As an example, for novelty, we can define *bridge length* metrics based on the number of ratings that are required to add certain items to the recommendation list by bridging to a new neighbor who has rated the items. In some sense this is like *potential* coverage; coverage being a measure of how many of the total items can be recommended to the user [3].

Conclusion

Overall, our work is part of a larger agenda to be able to characterize the healthy properties of recommender systems. We believe that the graph models provide a useful framework for this study by focusing attention on the connections that are used in the computations. Both the work of Mirza [6] and Vemuri [9] already support this contention. (We should acknowledge that the term *recommender system health* came to us from Joe Konstan.)

1. R. Fagin, R. Kumar, and D. Sivakumar. Compar-

ing top k lists. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms in Artificial Intelligence*, pages 28–36, 2003.

2. J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237, 1999.
3. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM TOIS*, 22(1):5–53, 2004.
4. M. Kendall and J. D. Gibbons. *Rank Correlation Methods*. Edward Arnold, London, 1990.
5. S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th international conference on World Wide Web*, pages 393–402. ACM Press, 2004.
6. B. J. Mirza, B. J. Keller, and N. Ramakrishnan. Studying recommendation algorithms by graph analysis. *Journal of Intelligent Information Systems*, 20(2):131–160, March 2003.
7. M. Newman. Assortative Mixing in Networks. *Phys. Rev. Lett.*, Vol. 89(20):208701, 2002.
8. B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-Based Collaborative Filtering Recommendation Algorithms. In *WWW'10, Proceedings of the Tenth International World Wide Web Conference*, pages 285–295, 2001.
9. N. S. Vemuri. Linking accuracy of recommendation algorithms to objective measures. Master's thesis, Eastern Michigan University, 2004.

Impacts of Contextualized Communication of Privacy Practices and Personalization Benefits on Purchase Behavior and Perceived Quality of Recommendation

Alfred Kobsa
Donald Bren School of
Information and Computer Sciences
Irvine, CA 92697-3425
+1 949 824-3007
kobsa@uci.edu

Max Teltzrow
Institute of Information Systems
Humboldt University
Spandauer Str. 1, 10178 Berlin, Germany
+49 30 2093 5730
teltzrow@wiwi.hu-berlin.de

ABSTRACT*

Consumer surveys have consistently demonstrated that privacy statements on the web are ineffective in alleviating users' privacy concerns. We investigated a new user interface design approach in which the privacy practices of a website are explicated in a contextualized manner, and users' benefits from providing personal data clearly explained. To test the merits of this approach, we conducted a user experiment with two versions of a web store that allegedly provided personalized book recommendations: one with a traditional global disclosure and one that additionally provides contextualized explanations of privacy practices and personalization benefits. We found that subjects in the second condition were significantly more willing to share personal data with the website, rated the perceived benefit resulting from data disclosure significantly higher, and also made considerably more purchases. We discuss the implications of these results and point out open research questions.

Keywords

Privacy disclosure, personalization, user benefit, trust, recommendation, perceived quality, adoption, purchases

INTRODUCTION

Privacy plays a major role in the relationship between companies and Internet users. More than two third of the respondents in [3] indicated that knowing how their data will be used would be an important factor in their decision on whether or not to disclose personal data. It seems though that the communication of privacy practices on the Internet has so far not been very effective in alleviating consumer concerns: 64% of Internet users surveyed in [10] indicated having decided in the past not to use a website, or not to purchase something from a website, because they were not sure about how their personal information would be used.

The current predominant way for websites to communicate how they handle users' data is to post comprehensive privacy statements (also known as "privacy policies" or "privacy disclosures"). 76% of users find privacy policies very important [11], and 55% stated that a privacy policy makes them more comfortable disclosing personal information [13, 19]. However, privacy statements today are usually written in a form that gives the impression that they are not really supposed to be read. And this is indeed not the case: whereas 73% of the respondents in [1] indicate having viewed web privacy statements in the past (and 26% of them claim to always read them), web site operators report that users hardly pay any attention to them[1]. [2] criticizes that people are turned off by long, legalistic privacy notices whose complexity makes them wonder what the organization is hiding.

Relegating the communication of privacy policies to merely publishing comprehensive privacy disclosures also disregards the situational nature of privacy [18].[2] Users seem to make privacy decisions much more consistently in concrete situations than upfront. In fact, privacy preferences stated upfront and actual usage behavior often seem to differ significantly [4, 20].

Moreover, merely communicating a company's privacy policy is not sufficient. In situated interviews [5], users pointed out that "in order to trust an e-Commerce company, they must feel that the company is doing more than just protecting their data – it must also be providing

Workshop: Beyond Personalization 2005

IUI'05, January 9, 2005, San Diego, California, USA

http://www.cs.umn.edu/Research/GroupLens/beyond2005

* This paper is an excerpt of [15]. Results from a second experiment as well as a proposed explanatory model have been added.

[1] For example, [16] indicates that less than 0.5% of all users read privacy policies.

[2] This criticism also applies to P3P [8] that is intended to alleviate current problems with privacy statements.

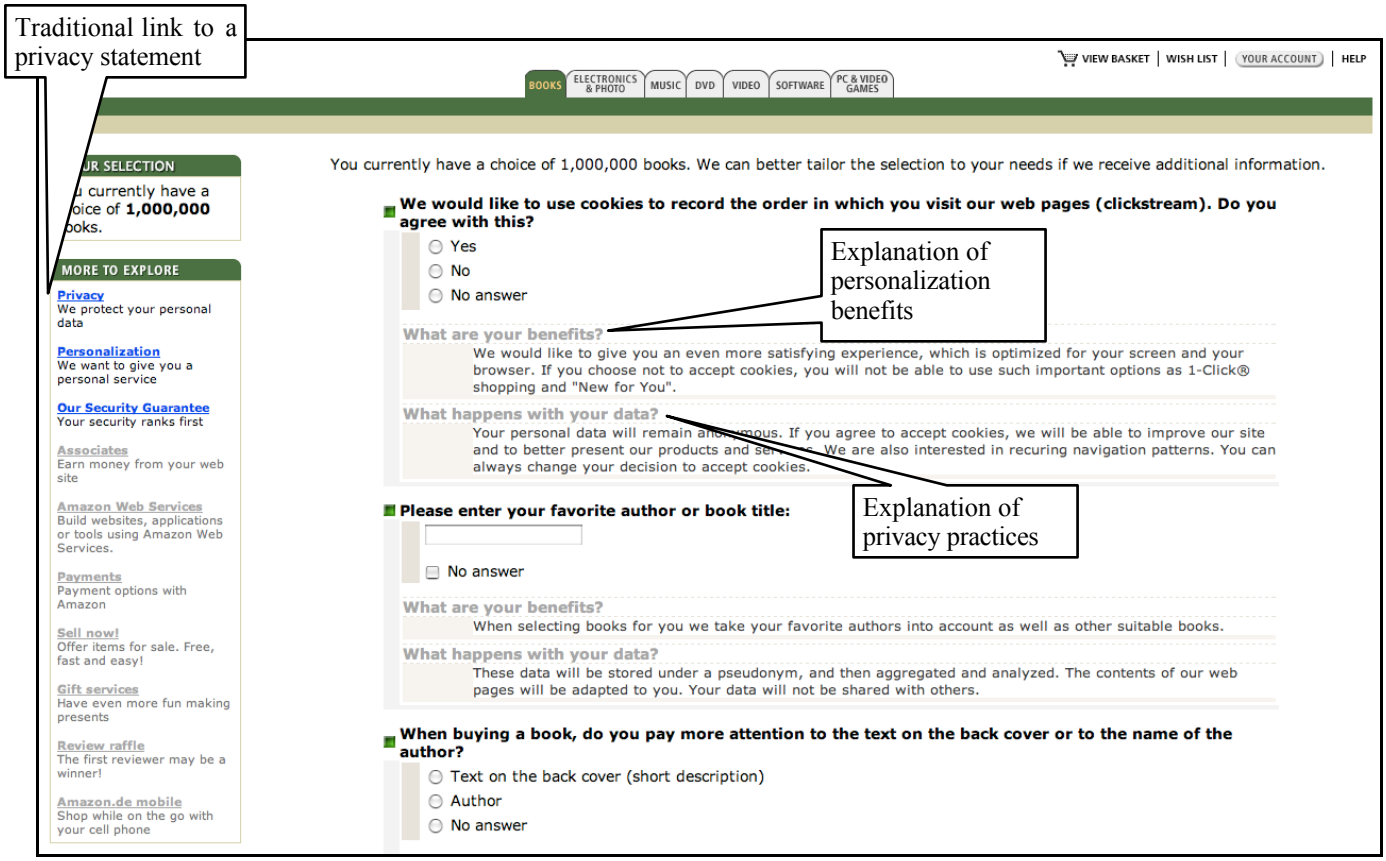


Figure 1: Global and contextual communication of privacy practices and personalization benefits

them with functionality and service that they value." The way in which personal data is used for the provision of these services must be clearly explained. Current web privacy statements hardly address the connection between personal data and user benefits.

A DESIGN PATTERN FOR WEBSITES THAT COLLECT PERSONAL DATA

To adequately address privacy concerns of users of personalized websites, we investigate user interface design patterns that communicate the privacy practices of a site both at a global and a local (contextualized) level. Similar to design patterns in object-oriented programming, interface design patterns constitute descriptions of best practices within a given design domain based on research and application experience [22]. They give designers guidelines for the efficient and effective design of user interfaces.

Global Communication of Privacy Practices and Personalization Benefits

Global communication of privacy practices currently takes place by posting privacy statements on a company's homepage or on all its web pages. Privacy policies are carefully crafted by legal council since they are legally binding and enforceable in many jurisdictions. Rather than completely replacing them by something new whose legal impact is currently unclear at best, our approach keeps current privacy statements in the "background" for legal reference and protection. However, we argue to enhance this kind of disclosure by additional information that explains privacy practices and user benefits, and their relation to the requested personal data, in the given local context.

Local Communication of Privacy Practices and Personalization Benefits

We expect that tailored in-context explanation of privacy practices and personalization benefits will address users' privacy concerns much better than global contextless disclosures. This approach breaks long privacy policies into smaller, more understandable pieces, refers concretely to the current context, and thereby allows users to make situated decisions regarding the disclosure of their personal data considering the explicated privacy practices and the explicated personalization benefits.

It seems safest to communicate privacy practices and personalization benefits at the level of each individual entry field for personal data. If a number of such fields form a visually separate sub-context on a page, compiled explanations may be given if the explanations for each individual field are not very different (due to legal differences, different sensitivity levels, privacy practices or personalization benefits). A page is the highest possible level at which compiled contextual explanations may be given (again, only if the field-level explanations are relatively similar). Visually separate sub-contexts on a page should be preferred though, due to the (cognitive) closure that they require.

An Example Website with Global and Contextual Communication of Privacy Practices and Personalization Benefits

Fig. 1 shows the application of the proposed interface design pattern to a web bookstore that gives personalized recommendations. The top three links in the left-hand frame lead to the global disclosures (to facilitate comprehension, we decided to split the usual contents of current privacy statements into three separate topics: privacy, personalization benefits, and security). The main frame contains input fields and checkboxes for entering personal data. Each of them is accompanied by an explanation of the site's privacy practices regarding the respective personal data (which focuses specifically on usage purposes), and the contribution to personalized recommendations that these data afford.

As in the theoretical model of [17], a user achieves an understanding of the privacy implications of the displayed situation both intuitively (taking the overall purpose of the site and page into account) and through adequate contextual notice. The traditional link to a privacy policy can still be accessed if so desired.

A COMPARATIVE EXPERIMENT

Materials

To evaluate the merits of our proposal, we developed a mock book recommendation and sales website whose interface was designed to suggest an experimental future version of a well-known online bookstore. Two variants of this system were created, one with contextual explanations of privacy practices and personalization benefits, and one without. Figure 1 shows an excerpt of the first variant, translated from German into English. The contextual explanations are given for each entry field (which is the safest of the strategies discussed above), under the headings "What are your benefits?" and "What happens with your data?" In the version without contextual explanations, these explanations are omitted.

In both conditions, the standard privacy policy of the web retailer is used. The three left-hand links labeled "Privacy", "Personalization" and "Our Security Guarantee" lead to the original company privacy statement (we split it into these three topics though and left out irrelevant text). In the condition with contextual explanations, the central policies that are relevant in the current situation are explained under "What happens with your data?" Such explanations state, for instance, that the respective piece of personal data will not be shared with third parties, or that some personal data will be stored under a pseudonym and then aggregated and analyzed. The explanation of the usage purpose is concise and kept in the spirit of P3P specifications [8].

A counter was visibly placed on each page that purported to represent the size of the currently available selection of books. Initially the counter is set to 1 million books. Data entries in web forms (both via checkboxes and radio buttons and through textual input) decrease the counter after each page by an amount that depends on the data entries made. The web forms ask a broad range of questions relating to users' interests. A few sensitive questions on users' political interests, religious interests and adherence, their literary sexual preferences, and their interest in certain medical subareas (including venereal diseases) are also present. All questions "make sense" in the context of filtering books in which users may be interested. For each question, users have the option of checking a "no answer" box or simply leaving the question unanswered. The personal information that is solicited in the web forms was chosen in such a way that it may be relevant for book recommendations and/or general customer and market analysis. Questions without any clear relation to the business goals of an online bookstore are not being asked. A total of 32 questions with 66 answer options are presented. Ten questions allow multiple answers, and seven questions have several answer fields with open text entries (each of which we counted as one answer option).

After nine pages of data entry (with a decreased book selection count after each page), users are encouraged to review their entries and then to retrieve books that purportedly match their interests. Fifty predetermined and invariant books are then displayed that were selected based on their low price and their presumable attractiveness for students (book topics include popular fiction, politics, tourism, and sex and health advisories). The prices of all books are visibly marked down by 70%, resulting in out-of-pocket expenses between €2 and €12 for a book purchase. For each book, users can retrieve a page with bibliographic data, editorial reviews, and ratings and reviews by readers.

Users are free to choose whether or not to buy one single book. Those who do are asked for their shipping and payment data (a choice of bank account withdrawal and credit card charge is offered). Those who do not buy may still register with their postal and email addresses, to receive personalized recommendations in the future as well as newsletters and other information.

Subjects and Procedures

58 subjects participated in the experiment. They were students of Humboldt University in Berlin, Germany, mostly in the areas of Business Administration and Economics. The data of 6 subjects were eventually not used, due to a computer failure or familiarity with the student experimenters. Participants were promised a € 6 coupon for a nearby popular coffee shop as a compensation for their participation, and the option to purchase a book with a 70% discount. Prospective participants were asked to bring their IDs and credit or bank cards to the experiment.

When subjects showed up for the experiment, they were reminded to check whether they had these credentials with them, but no data was registered at this time. Paraphernalia that are easily associated with the web book retailer, such as book cartons and logos, were casually displayed.

In the instructions part of the experiment, subjects were told that they would test an experimental new version of the online bookstore with an intelligent book recommendation engine inside. Users were advised that the more and the better data they provided, the better would be

the book selection. They were also told that their data would be given to the book retailer after the experiment. It was explicitly pointed out though that they were not required to answer any question. Subjects were asked to work with the prototype to find books that suited their interests, and to optionally pick and purchase one of them at a 70% discount. They were instructed that payments could be made by credit card or by withdrawal from their bank accounts.

A between-subjects design was used for the subsequent experiment, with the system version as the independent variable: one variant featured non-contextual explanations of privacy practices and personalization benefits only, and the other additionally contextualized explanations. Subjects were randomly assigned to one of the two conditions (we will abbreviate them by "-expl" and "+expl" in the following). They were separated by screens, to bar any communication between them. After searching for books and possibly buying one, subjects filled in two post-questionnaires, one online and one on paper. Finally, the data of those users who had bought a book or had registered with the system were compared with the credentials that subjects had brought with.

RESULTS

Data Sharing

We analyzed the data of 26 participants in the conditions "-expl" and "+expl". We first dichotomized their responses by counting whether a question received at least one answer or was not answered at all. Whereas on average 84% of the questions were answered in condition -expl, this rose to 91% in the second condition (see Table 1). A Chi-Square test on a contingency table with the total number of questions answered and not answered in each condition showed that the difference between conditions was statistically significant ($p<0.001$).

The two conditions also differed with respect to the number of answers given (see Table 1). In condition "-expl", subjects gave 56% of all possible responses on average (counting all options for multiple answers), while they gave 67% of all possible answers in condition "+expl". A Chi-Square contingency test showed again that the difference between the two conditions is highly significant ($p<0.001$). The relative difference between the number of answers provided in the two conditions is even higher than in the dichotomized case (19.6% vs. 8.3% increase).

	-expl	*+expl*	*diff*	*p*
Questions answered	84%	91%	+ 8%	<.001
Answers given	56%	67%	+20%	<.001
Book buyers	58%	77%	+33%	.07
"Data allowed store to select better books"	2.85	3.40	+19%	.035

Table 1: Effect on data sharing, purchases and perceived benefit

The results demonstrate that the contextual communication of privacy practices and personalization benefits has a significant positive effect on users' willingness to share personal data. The effect is even stronger when users can give multiple answers. We found no significant difference between questions that we regarded as more sensitive and less sensitive questions.

Purchases

Table 1 shows that the purchase rate in condition "+expl" is 33% higher than in condition "-expl" (note that all subjects saw the same set of 50 books in both conditions). A t-test for proportions indicates that this result approaches significance ($p<0.07$).

Perceived quality of recommendation

The paper questionnaire that was administered at the end of the study included several Likert questions on subjects' perception of the privacy practices of the website as well as its service quality. Possible answers ranged from "strongly agree" to "strongly disagree". One of these questions was "Did you feel that the particulars that you gave helped <bookseller> to chose interesting books for you?" Table 1 shows the average responses in the two conditions after encoding them on a one to five scale. The difference between the two conditions is highly significant (one-tailed t-test, $p<0.05$). Note again that all subjects were offered the same set of books.

DISCUSSION OF THE RESULTS AND OPEN RESEARCH QUESTIONS

Our experiment was designed so as to ensure that subjects had as much "skin in the game" as possible, and thereby to increase its ecological relevance. The incentive of a highly discounted book and the extremely large selection set that visibly decreased with every answer given was chosen to incite users to provide ample and truthful data about their interests. The perceptible presence of the web book retailer, the claim that all data would be made available to them, and the fact that names, addresses and payment data were verified (which ensured that users could not use escape strategies such as sending books to P.O. boxes or someone they know) meant that users really had to trust the privacy policy that the website promised when deciding to disclose their identities.

The results demonstrate that the contextualized communication of privacy practices and personalization benefits has a significant positive effect on users' data sharing behavior, and on their perception of the website's privacy practices as well as the perceived benefit resulting from data disclosure. The additional finding that this form of explanation also leads to more purchases approached significance. The adoption by web retailers of interface design patterns that contain such explanations therefore seems clearly advisable.

Our results support several of the assumptions underlying the model in Fig. 2, which is centrally based on the notion of trust. In condition "+expl", users' better understanding of the website's privacy practices and of the contribution of disclosed data to resulting personalization benefits is likely

to have increased users' trust and alleviated their privacy concerns. This in turn led to more data disclosure.

The decision to buy a book was a significant step in our experiment since at this point users revealed personally identifiable information (name, shipment and payment data) and risk that previously pseudonymous information may be linked to their identities. We already reported above that users indicate in surveys to refrain from shopping if the are uncertain about the possible fate of their data. It seems that the increased trust of users in condition "+expl" due to contextualized privacy disclosure may have contributed to more users opting to reveal their identities.

We have no direct explanation for the higher perceived benefits from data disclosure in condition "+expl". One can speculate about positive transfer effects from higher perceived privacy standards via higher trust.

Other characteristics of our experiment are also in agreement with the literature. [14] found in their study of consumer privacy concerns that "in the absence of straightforward explanations on the purposes of data collection, people were able to produce their own versions of the organization's motivation that were unlikely to be favorable. Clear and *readily available* explanations might alleviate some of the unfavorable speculation" [emphasis ours]. [9] postulate that consumers will "continue to disclose personal information as long as they perceive that they receive benefits that exceed the current or future risks of disclosure. Implied here is an expectation that organizations not only need to offer benefits that consumers find attractive, but they also need to be open and honest about their information practices so that consumers […] can make an informed choice about whether or not to disclose." The readily available explanations of *both* privacy practices and personalization benefits in our experiment meet the requirements spelled out in the above quotations, and the predicted effects could be indeed observed.

Having said this, we would however also like to point out that additional factors may also play a role in users' data disclosure behavior, which were kept constant in our experiment due to the specific choice of the web retailer, its privacy policy, and a specific instantiation of our proposed interface design pattern. We will discuss some of these factors in the following.

Reputation of a website. We chose a webstore that enjoys a relatively high reputation in Germany (we conducted surveys that confirmed this). It is well known that reputation increases users' willingness to share personal data with a website (see e.g. [6, 12, 21]). Our high response rates of 84% without and specifically 91% with contextual explanation suggest that we may have already experienced some ceiling effects. In a more recent version of the experiment we therefore changed the name and logo of the website to ones that had received a medium reputation rating in the prior survey. We found indeed similar effects of contextualized disclosures as at the website with high reputation, but with smaller numbers for data disclosure and purchases in both conditions. There was no interaction between reputation and form of disclosure.

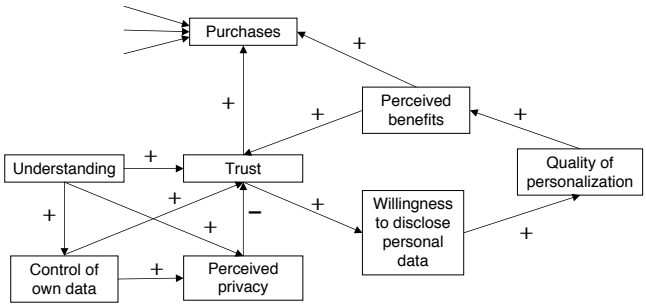


Figure 2: Suggested explanatory model

Stringency of a website's data handling practices. The privacy policy of the website that we mimicked is comparatively strict. Putting this policy upfront and explaining it in-context in a comprehensible manner is more likely to have a positive effect on customers than couching it in legalese and hiding it behind a link. Chances are that this may change if a site's privacy policy is not so customer-friendly.

Permanent visibility of contextual explanations. In our experiment, the contextual explanations were permanently visible. This uses up a considerable amount of screen real estate. Can the same effect be achieved in a less space-consuming manner, for instance with icons that symbolize the availability of such explanations? If so, how can the contextual explanations be presented so that users can easily access them and at the same time will not be distracted by them? Should this be done through regular page links, links to pop-up windows, or rollover windows that pop up when users brush over an icon?

References to the full privacy policy. As discussed above, privacy statements on the web currently constitute important and comprehensive legal documents. Contextual explanations will in most cases be incomplete since they need to be short and focused on the current situation, so as to ensure that users will read and understand them. For legal protection, it is advisable to include in every contextual explanation a proviso such as "This is only a summary explanation. See <link to privacy statement> for a full disclosure." Will users then be concerned that a website is hiding the juicy part of its privacy disclosure in the "small print", and therefore show less willingness to disclose their personal data?

Additional user experiments will be necessary to obtain answers or at least a clearer picture with regard to these questions.

ACKNOWLEDGMENTS

The work has been supported by the National Science Foundation (grant DST 0307504), Deutsche Forschungs-gemeinschaft (DFG grant no. GRK 316/2), and by Humboldt Foundation (TransCoop program). We would like to thank Christoph Graupner, Louis Posern and Thomas Molter for their help in conducting the user experiment described herein. The comments of the anonymous reviewers are also appreciated.

REFERENCES

1. A Survey of Consumer Privacy Attitudes and Behaviors, Harris Interactive, 2001.
2. Abrams, M., Making Notices Work For Real People. in *25th International Conference of Data Protection & Privacy Commissioners*, (Sydney, Australia, 2003).
3. Ackerman, M.S., Cranor, L.F. and Reagle, J., Privacy in E-commerce: Examining User Scenarios and Privacy Preferences. *First ACM Conference on Electronic Commerce* (Denver, CO, 1999), 1-8.
4. Berendt, B., Günther, O. and Spiekermann, S. Privacy in E-Commerce: Stated Preferences vs. Actual Behavior. *Communications of the ACM (to appear)*.
5. Brodie, C., Karat, C.-M. and Karat, J. How Personalization of an E-Commerce Website Affects Consumer Trust. In Karat, C.-M., Blom, J. and Karat, J. eds. *Designing Personalized User Experience for eCommerce*, Kluwer Academic Publishers, Dordrecht, Netherlands, 2004.
6. CG&I-R. Privacy Policies Critical to Online Consumer Trust, Columbus Group and Ipsos-Reid, 2001.
7. Chellappa, R.K. and Sin, R. Personalization versus Privacy: An Empirical Examination of the Online Consumer's Dilemma. *Information Technology and Management* , 6 (2-3), 2005.
8. Cranor, L., Langheinrich, M., Marchiori, M., Presler-Marshall, M. and Reagle, J. The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. W3C Recommendation 16 April 2002, http://www.w3.org/TR/P3P
9. Culnan, M.J. and Bies, R.J. Consumer Privacy: Balancing Economic and Justice Considerations. *Journal of Social Issues*, *59*. 323-353.
10. Culnan, M.J. and Milne, G.R., The Culnan-Milne Survey on Consumers & Online Privacy Notices: Summary of Responses. In: *Interagency Public Workshop Get Noticed: Effective Financial Privacy Notices*, (Washington, D.C., 2001). http://www.ftc.gov/bcp/workshops/glb/supporting/culnan-milne.pdf
11. Department for Trade and Industry. Informing Consumers about E-Commerce, Conducted by MORI, London: DTI, London, 2001. http://www.consumer.gov.uk/ccp/topics1/pdf1/ecomfull.pdf
12. Earp, J.B. and Baumer, D. Innovative Web Use to Learn About Consumer Behavior and Online Privacy. *Communications of the ACM Archive*, *46* (4). 81 - 83.
13. GartnerG2 Privacy and Security: The Hidden Growth Strategy.
14. Hine, C. and Eve, J. Privacy in the Marketplace. *The Information Society*, *14* (4). 253-262.
15. Kobsa, A. and Teltzrow, M. Contextualized Communication of Privacy Practices and Personalization Benefits: Impacts on Users' Data Sharing Behavior. In Martin, D. and Serjantov, A., eds. *Privacy Enhancing Technologies: Fourth International Workshop, PET 2004, Toronto, Canada*, Springer Verlag, Heidelberg, Germany, to appear. http://www.ics.uci.edu/~kobsa/papers/2004-PET-kobsa.pdf
16. Kohavi, R., Mining E-Commerce Data: the Good, the Bad, and the Ugly. in *Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (San Francisco, CA, 2001), 8-13.
17. Lederer, S., Dey, A. and Mankoff, J. A Conceptual Model and Metaphor of Everyday Privacy in Ubiquitous Computing, Intel Research, 2002. http://www.intel-research.net/Publications/Berkeley/120520020944_107.pdf
18. Palen, L. and Dourish, P., Unpacking "Privacy" for a Networked World. in *CHI-02*, (Fort Lauderdale, FL, 2002), 129-136.
19. Roy Morgan Research. Privacy and the Community, Prepared for the Office of the Federal Privacy Commissioner, Sydney, 2001. http://www.privacy.gov.au/publications/rcommunity.html
20. Spiekermann, S., Grossklags, J. and Berendt, B., E-privacy in 2nd Generation E-Commerce: Privacy Preferences versus Actual Behavior. in *EC'01: Third ACM Conference on Electronic Commerce*, (Tampa, FL, 2001), 38-47.
21. Teo, H.H., Wan, W. and Li, L., Volunteering Personal Information on the Internet: Effects of Reputation, Privacy Initiatives, and Reward on Online Consumer Behavior. in *Proc. of the 37th Hawaii International Conference on System Sciences* (Big Island, HI, 2004).
22. van Duyne, D.K., Landay, J.A. and Hong, J.I. *The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience*. Addison-Wesley, Boston, 2002.

InterestMap: Harvesting Social Network Profiles for Recommendations

Hugo Liu
MIT Media Laboratory
20 Ames St., Cambridge, MA, USA
hugo@media.mit.edu

Pattie Maes
MIT Media Laboratory
20 Ames St., Cambridge, MA, USA
pattie@media.mit.edu

ABSTRACT

While most recommender systems continue to gather detailed models of their "users" within their particular application domain, they are, for the most part, oblivious to the larger context of the lives of their users outside of the application. What are they passionate about as individuals, and how do they identify themselves culturally? As recommender systems become more central to people's lives, we must start modeling the person, rather than the user.

In this paper, we explore how we can build models of people outside of narrow application domains, by capturing the traces they leave on the Web, and inferring their everyday interests from this. In particular, for this work, we harvested 100,000 social network profiles, in which people describe themselves using a rich vocabulary of their passions and interests. By automatically analyzing patterns of correlation between various interests and cultural identities (*e.g.* "Raver," "Dog Lover," "Intellectual"), we built InterestMap, a network-style view of the space of interconnecting interests and identities. Through evaluation and discussion, we suggest that recommendations made in this network space are not only accurate, but also highly visually intelligible – each lone interest contextualized by the larger cultural milieu of the network in which it rests.

Keywords

User modeling, person modeling, recommender systems, item-item recommendation, social networks, collaborative filtering, cultural visualization.

1. INTRODUCTION

Recommender systems (*cf.* Resnick & Varian, 1997) have thus far enjoyed remarkable practical and commercial success. They have become a mainstay of e-commerce sites such as Amazon and Ebay for product recommendation; and recommenders have also been deployed to cater to subject domains such as books, music, tutoring, movies, research papers, and web pages. Most recommenders operate within a single application domain, and are powered by domain-specific data – either through explicitly given user profiles, or through implicitly gathered models of user behavior within the application framework. But why should recommenders be restricted to data gathered within the context of the application?

Enter the Web. Web-based communities are quite social and dynamic places – there are online chat forums, blogging and journaling sites, "rings" of personal web pages, and social network communities. In all of these communities, recommendations are happening "in the wild," all of the time. With natural language processing and a bit of finesse, we might hope to harvest information from these sources and use them to construct richer models of people, of communities and their cultures, and to power new kinds of recommender systems whose recommendations are sourced from online trends and *word-of-mouth.* The idea that recommendations could be sourced from traces of social activity follows from Terveen & Hill (2001), who refer to their approach as *social data mining*. They have looked at mining web page recommendations from Usenet messages, and through the structural analysis of web pages.

In this work, we turn to web-based social networks such as Friendster[1], Orkut[2], and MySpace[3] as a source of recommendations for a broad range of interests, *e.g.* books, music, television shows, movies, sports, foods, and more. On web-based social networks, people not only specify their friends and acquaintances, but they also maintain an explicit, self-crafted run-down of their interests and passions, inputted through free-form natural language. Having harvested 100,000 of these social network profiles, we apply natural language processing to ground interests into vast ontologies of books, music, movies, etc. We also mine out and map a category of special interests, called "passions," into the space of social and cultural identities (e.g. "Book Lover," "Raver," "Rock Musician"). By analyzing patterns of how these interests and identities co-occur, we automatically generated a network-style "map" of the affinities between different interests and identities, which we call an InterestMap. By spreading activation over the network (Collins & Loftus, 1975), InterestMap can be applied directly to make interest recommendations; due to InterestMap's unique network topology, we show that recommendations produced by this method incorporates factors of identity and taste. Outside of recommendation, we are also exploring other applications for InterestMap, such as marketing and matchmaking.

This paper is structured as follows. First, we discuss the source and nature of the corpus of social network profiles used to build

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

[1] http://www.friendster.com

[2] http://www.orkut.com

[3] http://www.myspace.com

InterestMap. Second, we outline the approach and implementation of InterestMap. Third, we describe how InterestMap may be used for recommendations, and we present an evaluation of the system's performance under this task. Fourth, we give further discussion for some lingering issues – the tradeoffs involved in using social network profiles to drive recommendations; and the implications of InterestMap's network-style representation for explainability and trust. We conclude with a greater vision for our work.

2. SOCIAL NETWORK PROFILES

The recent emergence and popularity of web-based social network software (*cf.* boyd, 2004; Donath & boyd, 2004) such as Friendster, Orkut, and MySpace can be seen as a tremendous new source of subject domain-independent user models, which might be more appropriately termed, *person models* to reflect their generality. To be sure, well over a million self-descriptive personal profiles are available across different web-based social networks.

While each social network's profile has an idiosyncratic representation, the common denominator across all the major web-based social networks we have examined is a *category-based representation* of a person's broad interests, with the most common categories being music, books, movies, television shows, sports, and foods. Within each interest category, users are generally unrestricted in their input, but typically enumerate lists of items, given as fragments of natural language. Even within a particular category, these items may refer to different things; for example, under "books," items may be an author's last name, a book's title, or some genre of books like "mystery novels," so there may be some inference necessary to map these natural language fragments into a normalized ontology of items. Figure 1 shows the structure and contents of a typical profile on the Orkut social network.

Figure 1. A screenshot of a typical profile taken from the Orkut social network. The interest categories shown here are typical to most web-based social network profile templates.

Note also in Figure 1, that there is a special category of interests called "passions." Among the social network profile templates we have examined, *all* of them have this special category, variously called "general interests," "hobbies & interests," or "passions." Furthermore, this special category always appears above the more specific interest categories, as it does in Figure 1, perhaps to encourage the thinking that these passions are more general to a person than other sorts of interests, and is more central to one's own self-concept and self-identification.

When mining social network profiles, we distinguish passions from other categories of more specific interests. With the hypothesis that passions speak directly to a person's social and cultural identity, we map the natural language items which appear under this category into an ontology of *identity descriptors.* For example, "dogs" maps into "Dog Lover," "reading" maps into "Book Lover", "deconstruction" maps into "Intellectual." Items in the other categories are mapped into their respective ontologies of *interest descriptors*.

In the following section, we describe how these profiles were harvested, normalized and correlated to build InterestMap.

3. THE INTERESTMAP APPROACH

The general approach we took to build InterestMap consists of four steps: 1) mine social network profiles; 2) extract out a normalized representation by mapping casually-stated keywords and phrases into a formal ontology of *interest descriptors* and *identity descriptors*; 3) augment the normalized profile with metadata to facilitate connection-making (*e.g.* "War and Peace" also causes "Leo Tolstoy," "Classical Literature," and other metadata to be included in the profile, at a discounted value of 0.5, for example); and 4) apply a machine learning technique to learn the semantic relatedness weights between every pair of *descriptors*. What results is a gigantic semantic network whose nodes are identity and interest descriptors, and whose numerically weighted edges represent strengths of semantic relatedness. Below, we give an implementation-level account of this process.

3.1 Building a Normalized Representation

Between January and July of 2004, we mined 100,000 personal profiles from two web-based social network sites, recording only the contents of the "passions" category and common interest categories, as only these are relevant to InterestMap. We chose two social networks rather than one, to attempt to compensate for the demographic and usability biases of each. One social network has its membership primarily in the United States, while the other has a fairly international membership. One cost to mining multiple social networks is that there is bound to be some overlap in their memberships (by our estimates, this is about 15%), so these twice-profiled members may have disproportionately greater influence on the produced InterestMap.

To normalize the representation of each profile, we implemented 2,000 lines of natural language processing code in Python. First, for each informally-stated list of interests, the particular style of delimitation had to be heuristically recognized. Common delimiters were commas, semicolons, character sequences (*e.g.* " \…/"), new lines, commas in conjunction with the word "and," and so on. A very small percentage of these "lists" of interests were not lists at all, so these were discarded.

The newly segmented lists contained casually-stated keyphrases referring to a variety of things. They refer variously to authorship like a book author, a musical artist, or a movie director; to genre like "romance novels," "hip-hop," "comedies," "French cuisine"; to titles like a book's name, an album or song, a television show, the name of a sport, a type of food; or to any combination thereof, *e.g.* "Lynch's Twin Peaks," or "Romance like Danielle Steele."

To further complicate matters, sometimes only part of an author's name or a title is given, *e.g.* "Bach," "James," "Miles," "LOTR," "The Matrix trilogy." Then of course, the items appearing under "passions," can be quite literally anything.

For a useful InterestMap, it is not necessary to be able to recognize every item, although the greater the recognition capability, the more useful will be the resulting InterestMap. To recognize the maximal number and variety of items, we created a vast formal ontology of 21,000 interest descriptors and 1,000 identity descriptors compiled from various comprehensive ontologies on the web for music, sports, movies, television shows, and cuisines, including The Open Directory Project[4], the Internet Movie Database[5], TV Tome[6], Wikipedia[7], All Music Guide[8], and AllRecipes[9]. The ontology of 1,000 identity descriptors required the most intensive effort to assemble together, as we wanted them to reflect the types of passions talked about in our corpus of profiles; this ontology was taken mostly from The Open Directory Project's hierarchy of subcultures and hobbies, and finished off with some hand editing. To facilitate the classification of a "passions" item into the appropriate identity descriptor, each identity descriptor is annotated with a bag of keywords which were also mined out, so for example, the "Book Lover" identity descriptor is associated with, *inter alia,* "books," "reading," "novels," and "literature." To assist in the normalization of interest descriptors, we gathered aliases for each interest descriptor, and statistics on the popularity of certain items (most readily available in The Open Directory Project) which could be used for disambiguation (*e.g.* "Bach" → "JS Bach" or → "CPE Bach"?).

Using this crafted ontology of 21,000 interest descriptors and 1,000 identity descriptors, the heuristic normalization process successfully recognized 68% of all tokens across the 100,000 personal profiles, committing 8% false positives across a random checked sample of 1,000 mappings. We suggest that this is a good result considering the difficulties of working with free text input, and enormous space of potential interests and passions. Once a profile has been normalized into the vocabulary of descriptors, they are expanded using metadata assembled along with the formal ontology. For example, a book implies its author, and a band implies its musical genre. Descriptors generated through metadata-association are included in the profile, but at a discount of 0.5 (read: they only count half as much). The purpose of doing this is to increase the chances that the learning algorithm will discover latent semantic connections.

3.2 Learning the Map of Interests and Identities

From these normalized profiles, we wish to learn the overall strength of the semantic relatedness of every pair of descriptors, across all profiles, and use this data to build InterestMap's network graph. Our choice to focus on the similarities between descriptors rather than user profiles reflects an item-based recommendation approach such as that taken by Sarwar *et al.* (2001).

Technique-wise, the idea of analyzing a corpus of profiles to discover a stable network topology for the interrelatedness of interests is similar to how *latent semantic analysis* (Landauer, Foltz & Laham, 1998) is used to discover the interrelationships between words in the document classification problem. For our task domain though, we chose to apply an information-theoretic machine learning technique called *pointwise mutual information* (Church *et al.*, 1991), or PMI, over the corpus of normalized profiles. For any two descriptors f_1 and f_2, their PMI is given in equation (1).

$$PMI(f_1, f_2) = \log_2\left(\frac{\Pr(f_1 f_2)}{\Pr(f_1)\Pr(f_2)}\right) \quad \textbf{(1)}$$

Looking at each normalized profile, the learning program judges each possible pair of descriptors in the profile as having a correlation, and updates that pair's PMI.

What results is a 22,000 x 22,000 matrix of PMIs. After filtering out descriptors which have a completely zeroed column of PMIs, and applying thresholds for minimum connection strength, we arrive at a 12,000 x 12,000 matrix (of the 12,000 descriptors, 600 are identity descriptors), and this is the complete form of the InterestMap. Of course, this is too dense to be visualized as a semantic network, but we have built less dense semantic networks from the complete form of the InterestMap by applying higher thresholds for minimum connection strength. Figure 2 is a visualization of a simplified InterestMap.

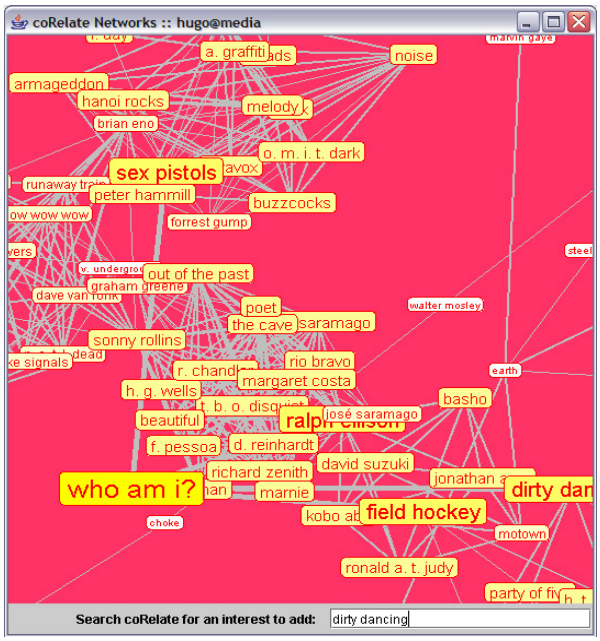


Figure 2. A screenshot of an interactive visualization program, running over a simplified version of InterestMap (weak edges are discarded, and edge strengths are omitted). The "who am i?" node is an indexical node around which a person is "constructed." As interests are attached to the indexical, correlated interests and identity descriptors are pulled into the visual neighborhood.

[4] http://www.dmoz.org

[5] http://www.imdb.com

[6] http://tvtome.com

[7] http://www.wikipedia.org

[8] http://www.allmusic.com

[9] http://allrecipes.com

3.3 Network Topology

Far from being uniform, the resultant InterestMap has a particular topology, characterized by two confluence features: *identity hubs,* and *taste cliques.*

Identity hubs are *identity descriptor nodes* which behave as "hubs" in the network, being more strongly related to more nodes than the typical *interest descriptor node.* They exist because the ontology of identity descriptors is smaller and less sparse than the ontology of interest descriptors; each identity descriptor occurs in the corpus on the average of 18 times more frequently than the typical interest descriptor. In InterestMap, identity hubs serve an *indexical* function. They give organization to the forest of interests, allow interests to cluster around identities. What kinds of interests do "Dog Lovers" have? This type of information is represented explicitly by identity hubs.

Another confluence feature is a *taste clique*. Visible in Figure 2, for example, we can see that "Sonny Rollins," is straddling two cliques with strong internal cohesion. While the identity descriptors are easy to articulate and can be expected to be given in the special interests category of the profile, tastes are often a fuzzy matter of aesthetics and may be harder to articulate using words. For example, a person in a Western European taste-echelon may fancy the band "Stereolab" and the philosopher "Jacques Derrida," yet there may be no convenient keyword articulation to express this. However, when the InterestMap is learned, cliques of interests seemingly governed by nothing other than taste clearly emerge on the network. One clique for example, seems to demonstrate a Latin aesthetic: "Manu Chao," "Jorge Luis Borges," "Tapas," "Soccer," "Bebel Gilberto," "Samba Music." Because the cohesion of a clique is strong, *taste cliques* tend to behave much like a singular identity hub, in its impact on network flow.

In the following section, we discuss how InterestMap may be used for recommendations, and evaluate the impact that identity hubs and taste cliques have on the recommendation process.

4. USING INTERESTMAP FOR RECOMMENDATIONS

InterestMap can be applied in a simple manner to accomplish several tasks, such as identity classification, interest recommendation, and interest-based matchmaking. A unique feature of InterestMap recommendations over straight interest-item to interest-item recommendations is the way in which identity and tastes are allowed to exert influence over the recommendation process. The tail end of this section describes an evaluation which demonstrates that identity and taste factors can improve performance in an interest recommendation task.

4.1 Finding Recommendations by Spreading Activation

Given a seed profile which represents a new user, the profile is normalized into the ontology of interest descriptors and identity descriptors, as described in Section 3.1. The normalized profile is then mapped onto the nodes of the InterestMap, leading to a certain activation pattern of the network.

With InterestMap, we view interest recommendation as a *semantic context* problem. By spreading activation (Collins & Loftus, 1975) outward from these seed nodes, a surrounding neighborhood of nodes which are connected strongly to the seed nodes emerges. As the distance away from the seed nodes increases (in the number of hops away), activation potential decays according to some discount (we having been using a discount of 0.75). The semantic neighborhood defined by the top *N* most related interest descriptor nodes corresponds with the top *N* interest recommendations produced by the InterestMap recommender.

Another straightforward application of InterestMap is identity classification. A subset of the semantic neighborhood of nodes resulting from spreading activation will be identity descriptor nodes, so the most proximal and strongly activated of these can be thought of as recognized identities. Identity classification with InterestMap can be useful in marketing applications because it allows a distributed interest-based representation of a person to be summarized into a more concise demographic or psychographic grouping.

Finally, we are experimenting with InterestMap for interest-based matchmaking, which may be useful for making social introduction. To calculate the affinity between two people, two seed profiles lead to two sets of network activations, and the strength of the contextual overlap between these two activations can be used as a coarse measure of how much two people have in common.

4.2 Evaluation

We evaluated the performance of spreading activation over InterestMap in the interest recommendation task. In this evaluation, we introduced three controls to assess two particular features: 1) the impact that identity hubs and taste cliques have on the quality of recommendations; and 2) the effect of using spreading activation rather than a simple tally of PMI scores. In the first control, identity descriptor nodes are simply removed from the network, and spreading activation proceeds as usual. In the second control, identity descriptor nodes are removed, and *n*-cliques[10] where *n>3* are weakened[11]. The third control does not do any spreading activation, but rather, computes a simple tally of the PMI scores generated by each seed profile descriptor for each of the 11,000 or so interest descriptors. We believe that this successfully emulates the mechanism of a typical non-spreading activation item-item recommender because it works as a pure information-theoretic measure.

We performed five-fold cross validation to determine the accuracy of InterestMap in recommending interests, versus each of the three control systems. The corpus of 100,000 normalized and metadata-expanded profiles was randomly divided into five segments. One-by-one, each segment was held out as a test corpus and the other four used to either train an InterestMap using PMI.

Within each normalized profile in the test corpus, a random half of the descriptors were used as the "situation set" and the remaining half as the "target set." Each of the four test systems uses the situation set to compute a *complete recommendation*— a rank-ordered list of all interest descriptors; to test the success of this recommendation, we calculate, for each interest descriptor in the target set, its percentile ranking within the complete recommendation list. The overall accuracy of recommendation is the arithme-

[10] a qualifying clique edge is defined here as an edge whose strength is in the 80th percentile, or greater, of all edges

[11] by discounting a random 50% subset of the clique's edges by a Gaussian factor (0.5 *mu,* 0.2 *sigma*).

tic mean of the percentile scores generated for each interest descriptor of the target set.

We opted to score the accuracy of a recommendation on a sliding scale, rather than requiring that descriptors of the target set be guessed exactly within *n* tries because the size of the target set is so small with respect to the space of possible guesses that accuracies will be too low and standard errors too high for a good performance assessment. For the InterestMap test system and control test systems #1 (Identity OFF) and #2 (Identity OFF and Taste WEAKENED), the spreading activation discount was set to 0.75). The results of five-fold cross validation are reported in Figure 3.

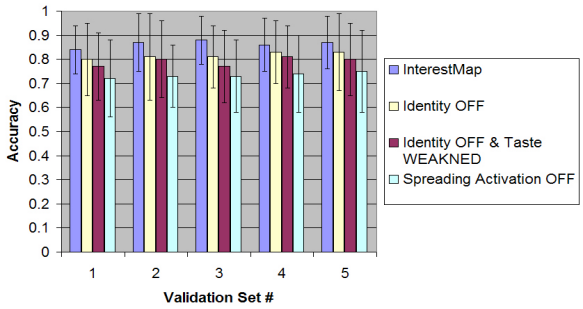


Figure 3. Results of five-fold cross-validation of InterestMap and three control systems on a graded interest recommendation task.

The results demonstrate that on average, the original InterestMap recommended with an accuracy of 0.86. In control #1, removing identity descriptors from the network not only reduced the accuracy to 0.81, but also increased the standard error by 38%. In control #2, removing identity descriptors and weakening cliques further deteriorated accuracy slightly, though insignificantly, to 0.79. When spreading activation was turned off, neither identity hubs nor taste cliques could have had any effect, and we believe that is reflected in the lower accuracy of 73%. However, we point out that since control #3's standard error has not worsened, its lower accuracy should be due to overall weaker performance across all cases rather than being brought down by exceptionally weak performance in a small number of cases.

We believe that the results demonstrate the advantage of spreading activation over simple one-step PMI tallies, and the improvements to recommendation yielded by identity and taste influences. Because activation flows more easily and frequently through identity hubs and taste cliques than through the typical interest descriptor node, the organizational properties of identity and taste yield proportionally greater influence on the recommendation process; this of course, is only possible when spreading activation is employed.

5. DISCUSSION

In this section, we discuss some of the advantages and disadvantages of using social network profiles to drive recommendations, and implications of InterestMap's network-style view of a space for trust and explainability.

5.1 Tradeoffs in Using Social Network Profiles to Drive Recommendations

The harvesting of social network profiles for recommendations involves several important tradeoffs to be considered.

Fixed Ontology versus Open-ended Input. While domain-specific behavior-based recommenders model user behavior over a predetermined ontology of items (*e.g.* a purchase history over an e-commerce site's ontology of products; a rating history over an application's ontology of movies), items specified in a social network profile are open-ended. Granted that in the normalization of a profile, items will have to be eventually normalized into an ontology, there still remains the psychological priming effects of a user working over the artifacts of a fixed ontology as he/she is composing ratings. For example, in a movie domain, a user may choose to rate a movie because of the way the movies are browsed or organized, and may find movies to rate which the user has long forgotten and is surprised to see in the ontology. In filling out the movies category in a social network profile, there is no explicit display of a movie ontology to influence user input, and a user could definitely not input movies which he/she has long since forgotten.

This generates the following tradeoff: Recommenders based on domain-specific behaviors will be able to recommend a greater variety of items than open-ended input based recommenders, including the more obscure or not entirely memorable items, because the application's explicit display of those items will remind a user to rate them. On the other hand, open-ended input may tend to recommend items which are more memorable, more significant, or possessing greater communicative value. This is especially true for social network profiles, where users have an explicit intention to communicate who they are through each of the interests descriptors they specify. We suggest that high-communicative value adds a measure of fail-softness to recommendations. For example it might be easier to rationalize or forgive the erroneous recommendation of a more prominent item like "L.v. Beethoven's Symphony No. 5" to a non-classical-music-lover than an equally erroneous recommendation of a more obscure or arbitrary feature like "Max Bruch's Op. 28."

Socially Costly Recommendation. The social cost paid by a user in producing a "rating" can greatly affect the quality and nature of recommendations. To begin with, in some domain-specific behavior-based recommender systems, the profile of user behavior is gathered implicitly and this profile is kept completely private. Here there is no cost paid by a user in producing a "rating." In a second case, some domain-specific recommender systems make their users' ratings publicly viewable. The introduction of the publicity dimension is likely to make a user more conscious about the audience for the ratings, and more careful about the ratings that he/she produces; thus, the user is paying some social cost, and as a result, we might expect the ratings to be less arbitrary than otherwise. Employing this same logic for monetary cost, we might expect Amazon recommendations based on purchases to be less arbitrary that recommendations based on products viewed.

Thirdly, in the case of social network profiles, the greatest cost is paid by a user in listing an item in his/her profile. Not only is the profile public, but it is viewed by exactly the people whose opinions the user is likely to care about most – his/her social circle; Donath & boyd (2004) report that a person's presentation of self in profiles is in fact a strategic communication and social signalling game. Items chosen for display are not just any subset of possessed interests, but rather, are non-arbitrary items meant to be representative of the self; furthermore, users may consciously intend to socially communicate those items to their social circle.

The social cost dimension to recommendation produces another interesting tradeoff. The higher the social cost paid by the user in producing a rating, the more deliberate the ratings. So we can anticipate recommendations made using this data to be consistent in their social apropos. On the other hand, social stigma will tend to suppress the rating and recommendation of malapropos items, for instance, perhaps the cost of listing the publicly-derided but oft privately-appreciated "Britney Spears" in one's profile is prohibitively high. Of course, these social pressures also manifest in real-life social recommendations, and the thought of recommending "Britney Spears" to someone you are not very comfortable with may be just as dissuasive.

5.2 Impact of Network-Style Views on Explainability and Trust

That a user trusts the recommendations served to him by a recommender system is important if the recommender is to be useful and adopted. Among the different facilitators of trust, Wheeless & Grotz (1977) identify transparency as a prominent desirable property. When a human or system agent discloses its assumptions and reasoning process, the recipient of the recommendation is likely to feel less apprehensive toward the agent and recommendation. Also in the spirit of transparency, Herlocker *et al.* (2000) report experimental evidence to suggest that recommenders which provide explanations of its workings experience a greater user acceptance rate than otherwise.

Unlike opaque statistical mechanisms like collaborative filtering (Shardanand & Maes, 1995), InterestMap's mechanism for recommendation can be communicated visually as a large network of interests and identities. The cliques and idiosyncratic topology of this *fabric of interests* visually represents the common tendencies of a large group of people. For example, in Figure 2, it is plain to see that "Sonny Rollins" and "Brian Eno" are each straddling two different cliques of different musical genres. The rationale for each recommendation, visually represented as the spreading of flow across the network, is easily intelligible. Thus it may be easier for a user to visually contextualize the reasons for an erroneous recommendation, *e.g.* "I guess my off-handed taste for Metallica situated me in a group of metal heads who like all this other stuff I hate."

The ability to interact with the InterestMap network space may also afford the system an opportunity to *learn* more intelligently from user feedback about erroneous recommendations. Rather than a user simply stating that she did not like a particular recommendation, she can black out or deprecate particular clusters of the network which she has diagnosed as the cause of the bad recommendation, *e.g.* "I'll black out all these taste cliques of heavy metal and this identity hub of "Metal Heads" so the system will not make *that* mistake again." Although we have not yet implemented such a capability in InterestMap, we hope to do so shortly.

6. CONCLUSION

As recommender systems play ever-larger roles in people's lives, providing serendipitous suggestions of things to do and people to meet, recommendation technology will have to be based on something other than domain-specific knowledge, which is facing a semantic interoperability crisis. To some degree, we will have to abandon *user modeling* in favor of *person modeling*, and *cultural modeling.* We hope that the work presented in this paper begins to illustrate a path in this direction. By harvesting the traces of how people behave *in the wild* on the Web and on their computers, we can build a more general model of their person. By looking at interests within the context of emergent cultural patterns, we find new bases for recommendation, driven by cultural identities, and modes of taste. And the best part of this new paradigm for recommendation is that it would be more intelligible and transparent to people, for we, as persons, are already well-equipped to understand interests in the context of cultural milieus.

7. ACKNOWLEDGMENTS

The authors would like to thank the blind reviewers for their helpful feedback, and Judith Donath for inspiring our social perspective on this work.

8. REFERENCES

[1] danah boyd: 2004, Friendster and publicly articulated social networks. *Conference on Human Factors and Computing Systems (CHI 2004)*. ACM Press.

[2] K.W. Church, W. Gale, P. Hanks, and D. Hindle: 1991, Using statistics in lexical analysis. In *Uri Zernik (ed.), Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon,* pp. 115-164. New Jersey: Lawrence Erlbaum, 1991

[3] A.M. Collins, and E.F. Loftus: 1975, A spreading-activation theory of semantic processing. *Psychological Review, 82*, pp. 407-428

[4] Judith Donath and danah boyd: 2004, Public displays of connection, *BT Technology Journal 22(4):*pp. 71-82. Kluwer

[5] J. Herlocker, J. Konstan and J. Riedl: 2000, Explaining Collaborative Filtering Recommendations. *Conference on Computer Supported Cooperative Work*, pp. 241-250

[6] T.K. Landauer, P.W. Foltz, D. Laham: 1998, An introduction to Latent Semantic Analysis . *Discourse Processes, 25,* 259-284.

[7] P. Resnick and H.R. Varian: 1997, Recommender Systems. *Communications of the ACM, Vol. 40(3):*pp. 56-58.

[8] B.M. Sarwar *et al.*: 2001, Item-Based Collaborative Filtering Recommendation Algorithms. *10th Int'l World Wide Web Conference*, ACM Press, pp. 285-295.

[9] U. Shardanand and P. Maes: 1995, Social information filtering: Algorithms for automating `word of mouth'. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pp. 210-217.

[10] L.G. Terveen and W.C. Hill: 2001, Beyond Recommender Systems: Helping People Help Each Other, in *Caroll, J. (ed.), HCI In The New Millennium*. Addison-Wesley

[11] L. Wheeless and J. Grotz: 1977, The Measurement of Trust and Its Relationship to Self-disclosure. *Communication Research 3(3)*, pp. 250-257.

What Affects Printing Options? - Toward Personalization & Recommendation System for Printing Devices

Masashi Nakatomi, Soichiro Iga,
Makoto Shinnishi, Tetsuro Nagatsuka, and Atsuo Shimada
RICOH Co., Ltd. Office System R&D Center
16-1 Shinei-cho, Tsuzuki-ku,
Yokohama, 224-0035 JAPAN
+81 45 590 1039
nakatomi@rdc.ricoh.co.jp

ABSTRACT
We introduce our ongoing research toward context-aware based personalization and recommendation system for a multi-functional printing device. The initial purpose of this research is to provide personalized user interface (UI) by inferring printing options such as number of sheets, simplex/duplex printing, or media size from context information. Context information includes time, kind of document, user's tasks, location, and so on. The problem here is how we could make this inference reliable from the limited context information that the device can capture. For the first step toward solving this problem, we should understand which context information really affects printing options that the user is going to select.

In this paper, we report the result of the analysis that we carried out to specify such context information. We found out that the file-related context would affect printing options. We then discuss how we could realize personalized UI for printing devices from our result. We also propose an early prototype system to test our approach.

Keywords
Context-aware computing, Interaction technique, Shared printing devices

INTRODUCTION
A printing device is no more a simple printer. We can use multiple functions such as faxing, scanning, e-mailing, and so on through a printer. The more it obtains new functions, the more the user has to take steps to reach his or her goal. The users have many options even for a simple printing such as duplex printing or stapling, and for those who change default printing options, it may be troublesome to take several actions to change options. (For example, four mouse-clicks are necessary to select duplex printing.) General users are therefore thought to select default options. So the problems are at complexity of functions and user's unawareness of functions.

To cope with increasingly complex devices, more intelligent and interactive UI is highly expected. To introduce such UI to shared devices like printers, there should be two approaches: personalization and recommendation. Personalization is an adaptive technique to infer the preferences of individuals [1]. Recommendation is a technique based on supporting an individual by other users [2][3], which is widely accepted in e-commerce [4]. When we introduce such suggestive UI to the shared devices, we assume that there are 4 types of users:

1) The users who do not use functions because they are not necessary.

2) The users who are not aware of functions and would not use them.

3) The users who have an idea of functions, but would not use them because the users think they are difficult to operate.

4) The users who use functions when they are necessary.

The first and fourth type users are not likely to use intelligent and interactive UI. It is the second and the third type users who need a support from UI. To support them, recommendation of the functions would be useful for the second type while suggestion based on personalization would be useful for the third type. Though printing devices is shared devices, it is a timeshared public resource like ATM machines where the user owns all resources, not like whiteboards where multiple people use resource simultaneously [5]. On the other hand the users have their clear goal even for a simple printing, so the UI should be adaptive and personalized for their task. Furthermore the inference of the preference of individuals could be applied to recommendation systems by making a cluster of users with the same goal such as the same tasks, the same working group, etc. and sharing preferences among the

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

users of the cluster. Therefore, we mainly focus on realizing personalized UI for printers where the user's goal is inferred with consideration of the user's task, i.e. contexts.

Recent context-aware computing approach provides possible key to realizing personalization system [6][7]. Highly user-oriented personalized service could be provided through the notion of context-aware computing, which is to capture, represent, and process context information to interact with the computing system. On the other hand, effective and efficient interaction technique without inferring user's task should also be important for a realization of the practical personalization. So our goal will be achieved through two threads of research: an accurate inference about the user's goal and design of interaction technique suitable for personalized UI.

In order to make an accurate inference of the user's goal, it is necessary to specify contexts that affect user's selecting printing options. For the first step, we performed a quantitative analysis on printer-usage logs for approximately 6 months. The log includes 10 attributes like the number of printing sheets. In this paper, we investigated correlation among these attributes based on Cramer's V value as a correlation coefficient. We found out that the file type and the number of pages of the original documents have correlation with printing options in different ways for different printing options.

Then we consider future potential of personalization where user's goal is inferred based on context information and recommended. Here we define context information as any information related to the users, the devices, and the tasks (e.g. the document the user is printing). We also introduce our preliminary design of personalized UI for printers where default options are made by inference based on file extensions.

Finally, we conclude that the analysis result plays an important role for the achievement of the personalized UI with the perspective that specification of more external contexts is needed.

TARGET DEVICES

The purpose of this work is to set up a personalized UI for multifunctional printing devices. What we call printing devices do not refer only to devices which just print out electronic documents, but also to devices which have multiple functions other than printing, e.g. scanning, photocopying, data storage, networking, etc.

However the users would be able to get further benefit from the combination of these functions, there are two main problems. One is that it gets more complex to select particular function on UI as the number of functions increases, and the other is that it gets more difficult to take control of a large number of functions, even if the functions are beneficial.

The number of functions would gradually increase furthermore, and it will be more important to solve these problems, which we think it has a potential to solve with personalization and recommendation.

ANALYSIS METHOD

We conducted an experiment to investigate the degree of correlation between context information. First we collected printing logs by software that obtain information from printer driver and send it to the database server every time client machines make a printing job. The overviews of the printing logs are as follows:

- 1/September/2003 to 9/March/2004 (for approximately 6 months)
- 77 persons from the department of sales
- 77719 logs in total
- Each log includes 10 attributes: user's ID, time of the print out, the title or name of electronic document (URL in the case of printing webpage), the number of the pages of the original document, the number of copies, the number of output sheets, the media size, duplex or simplex, the number of pages per sheet, color mode.

After collecting all these logs, we made an analysis as follows.

For all possible pairs of the attributes included in the logs above except for user's ID, we calculated the Cramer's V value based on the cross tabulations of frequencies we constructed for each pair of attributes. Cramer's V is defined in a cross tabulation as square root of $\chi^2/n \cdot m$, where n is a sample size and m is the smaller of (row size - 1) or (column size - 1). It is considered a better choice since it ranges from 0 (no correlation) to 1 (perfect correlation) regardless of table size or sample size.

In the following, we express the 9 attributes except for user's ID as "DUPL" to "TIME" as explained in the table 1. Note that we did not make an analysis of correlations between DUPL-NOS, PPS-NOS, NPD-NOS, NC-NOS, and among NPD, EXT, and TIME, since NOS is inherently dependent on DUPL, PPS, NPD, and NC, and NPD, EXT, and TIME are not printing options.

Based on the obtained Cramer's V value, we specified pairs of the attributes that have correlation with each other. We also carried out the same calculation for each user separately since it is necessary to find the difference of tendency among the users in order to achieve our purpose of personalization. For further investigation, we calculated the value for each user for each month in order to exclude the influence of variation with time of the user's tendency.

Table 1. 9 attributes used for the analysis of correlation.

	Attribute (examples of options)
DUPL	duplex or simplex
PPS	pages per sheet (2in1 printing)
COL	color mode (full-Color, monochrome)
NPD	the number of pages of the original document
NC	the number of copies
NOS	the number of output sheets
MSIZE	media size (A4, B4)
EXT	file extension (doc, pdf)
TIME	time expressed by hour number of printing time

RESULT

Analysis of tendency for whole logs

Table 2 shows the result for the whole users from the whole logs. In the table, the value between PPS and NPD is the highest and the value between PPS and EXT is the second highest. (0.454 and 0.381 respectively.) The details for these two relationships are expressed in table 3 as cross tabulations. From table 3(a) we can find the tendency that the users prefer 2in1 printings as the number of pages of original documents increases. However we could not conclude from table 3(b) that file extensions have correlation with 2in1 printing options. So we identified the correlation only between PPS and NPD based on table 2.

Analysis of the correlation unique to particular users

However it is ideal to confirm the cross tabulations of frequencies as we did above, it is not realistic to check them out for all 77 users. So we analyzed the distribution of Cramer's V for each relationship.

Table 4 expresses the result when calculated separately for each user. The values represent the top-10th and the top-20th percentile point in the distribution of Cramer's V for each relationship. The two tables show that the value between DUPL-COL, PPS-NPD, DUPL-NPD, COL-NPD, COL-NOS, DUPL-EXT, PPS-EXT, COL-EXT, NPD-EXT, MSIZE-EXT, and COL-TIME are higher than the others. It indicates correlations could exist at least for particular users. To find correlations at least for particular users, we construct histograms of relative frequencies as shown in Figure 1. If the correlation exists only for particular users, there are two types of users: the users whose Cramer's V is higher and the users whose Cramer's V are lower. So we confirmed the correlation by verifying that there are two types of users.

The histograms of relative frequencies in Figure 1 represent distribution of Cramer's V value for each pair of the attributes. In figure 1(a) we can see the peak at higher value (0.7-0.8) of Cramer's V. It indicates general tendency of strong correlation, which agrees with the result of table 2.

Table 2. Cramer's V between 9 attributes for whole user: DUPL~TIME stands for duplex or simplex, pages per sheet, color mode, pages of the original document, copies, output sheets, media size, file extension, time respectively.

	DUPL							
PPS	0.099	PPS						
COL	0.050	0.091	COL					
NPD	0.141	0.454	0.195	NPD				
NC	0.038	0.046	0.076	0.175	NC			
NOS	0.248	0.278	0.195	0.679	0.583	NOS		
MSIZE	0.106	0.041	0.138	0.296	0.289	0.294	MSIZE	
EXT	0.174	0.381	0.265	0.117	0.035	0.075	0.204	EXT
TIME	0.059	0.052	0.068	0.060	0.025	0.041	0.044	0.063

Table 3. The cross tabulation of frequencies (a) for the number of pages per sheet (rows) and the number of pages of the original document (columns) and (b) for major 6 file extensions (rows) and the number of pages per sheet (columns).

(a)

	1-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80	81-90	91-100	101-
2in1	2553	788	319	153	118	75	20	19	10	9	63
normal	71557	1236	368	141	97	48	23	16	14	19	63

(b)

	xls	html	pdf	doc	ppt	jpg
2in1	347	207	400	282	1968	1
normal	27607	3603	1961	6265	5020	70

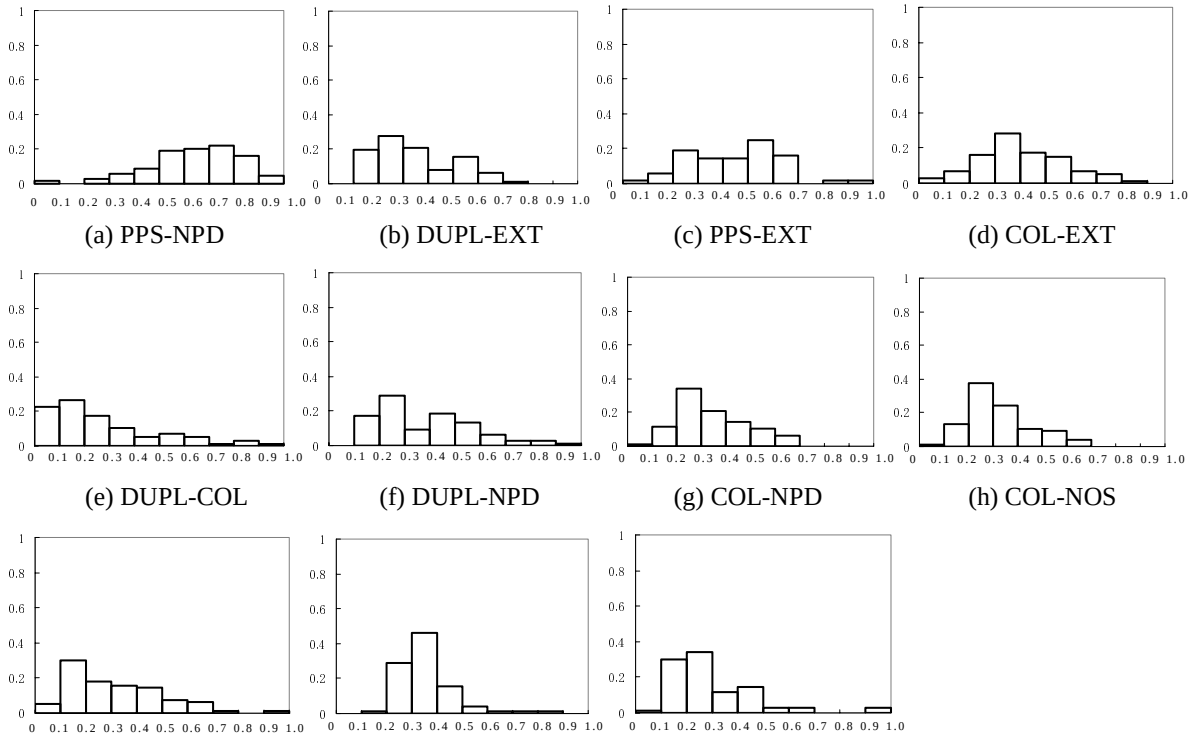
Table 4. (a) The top-10th and (b) the top-20th percentile point of the Cramer's V when calculated separately for each user.

(a)

	DUPL							
PPS	0.454	PPS						
COL	0.604	0.292	COL					
NPD	0.642	0.818	0.577	NPD				
NC	0.214	0.234	0.229	0.381	NC			
NOS	0.617	0.685	0.518	0.920	0.808	NOS		
MSIZE	0.302	0.164	0.303	0.403	0.180	0.372	MSIZE	
EXT	0.595	0.635	0.645	0.482	0.272	0.413	0.591	EXT
TIME	0.493	0.338	0.473	0.410	0.215	0.372	0.468	0.443

(b)

	DUPL							
PPS	0.290	PPS						
COL	0.477	0.211	COL					
NPD	0.543	0.777	0.457	NPD				
NC	0.128	0.118	0.147	0.296	NC			
NOS	0.497	0.619	0.413	0.904	0.736	NOS		
MSIZE	0.178	0.111	0.230	0.255	0.117	0.239	MSIZE	
EXT	0.534	0.583	0.578	0.413	0.202	0.347	0.464	EXT
TIME	0.358	0.293	0.410	0.333	0.172	0.280	0.308	0.353



(a) PPS-NPD (b) DUPL-EXT (c) PPS-EXT (d) COL-EXT

(e) DUPL-COL (f) DUPL-NPD (g) COL-NPD (h) COL-NOS

(i) MSIZE-EXT (j) NPD-EXT (k) COL-TIME

Figure1. Histograms of relative frequencies of the Cramer's V

The histograms in figure 1(b) (c) (f) show two peaks at a lower value (0.2-0.3) and at a higher value (0.5-0.6 for DUPL-EXT and PPS-EXT, 0.4-0.5 for DUPL-NPD). They indicate the correlations for these two relationships exist for some users while the correlations do not exist for some.

Figure 1(d) shows broad unimodal distribution and the peak is not at a higher value (0.3-0.4). But we can recognize a shoulder at 0.5-0.6 indicating correlation between COL-EXT for particular users.

We could not recognize strong correlation between DUPL-COL, COL-NPD, COL-NOS, MSIZE-EXT, NPD-EXT, and COL-TIME, from histogram of figure 1 (e) (g) (h) (i) (j) (k), since the peaks of the histograms are low (0.1-0.4) and show unimodal distributions.

Table 5 and table 6 are the example of the tendency difference among the users. In table 5(a), we can see that the user is likely to select 2in1 printings for pdf or ppt documents than the other kinds of documents, while another user prefer 1-page-per-sheet printings for all kinds of documents as shown in table 5(b). We can also see in table 6(a) that the user prefer duplex printings for doc and pdf documents, while another user prefer duplex printings for all kinds of documents as shown in table 6(b).

Table 5. The cross tabulation of frequencies for file extensions (rows) and the number of pages per sheet (columns) for 2 users.

(a)

	xls	**doc**	**html**	**pdf**	**ppt**
2in1	0	4	0	21	130
normal	640	202	105	40	28

(b)

	xls	**doc**	**html**	**pdf**	**ppt**
2in1	2	4	0	3	1
normal	966	175	134	47	58

Table 6. The cross tabulation of frequencies for file extensions (rows) and duplex/simplex printing (columns) for 2 users.

(a)

	xls	**doc**	**html**	**pdf**	**ppt**
duplex	41	44	3	40	28
simplex	137	3	7	20	31

(b)

	xls	**doc**	**html**	**pdf**	**ppt**
duplex	918	178	133	50	54
simplex	50	1	1	0	5

Variation of the correlation with time

With the Cramer's V obtained separately for each month, we made histograms of relative frequencies and evaluated the correlation for each pair based on the histograms as we did in the former section. The result shows that there were correlation between COL-NPD, PPS-EXT, and MSIZE-EXT. Between COL and NPD, the histograms show bimodal distributions which have the second peak at more than 0.5 for 4 months and unimodal but broad distribution with the peak positioned at 0.3-0.4 for the other 2 months. Between PPS and EXT, the histograms show bimodal distributions that have the second peak at more than 0.5 for whole 6 month. Between MSIZE and EXT, the histograms show bimodal distributions with the peak positioned at more than 0.5 for 4 months and at 0.3-0.4 for the other 2 months. For the other pairs, we could recognize the histograms that indicate correlation for less than 2 months. Note that we did not make an analysis for correlations between PPS-NPD, DUPL-EXT, PPS-EXT, DUPL-NPD, and COL-EXT since we already confirmed the correlations for these pairs.

We also confirmed the variation of the value for these correlations seeing that particular users have wide range of the Cramer's V value, 0.288 to 0.691 for user X as expressed in table 7 for example. Table 8 is the cross tabulation of frequencies of the user X for September and December. We can see the A3 sheets are preferred only for "xls" documents in September while A4 sheets are preferred for all kinds of documents in December.

Table 7. The variation of Cramer's V between the attribute MSIZE and EXT for 3 users.

	Sep.	Oct.	Nov.	Dec.	Jan.	Feb.
user X	0.691	0.573	0.387	0.288	0.397	0.380
user Y	0.571	0.641	0.355	0.344	0.642	0.587
user Z	0.244	0.454	0.249	0.118	0.222	0.274

Table 8. The cross tabulation of frequencies between media size and 5 major file extensions for user X in the table 7.

(a)September

	xls	**ppt**	**pdf**	**doc**	**html**
A4	78	49	3	27	2
A3	181	1	1	0	1

(b)December

	xls	**ppt**	**pdf**	**doc**	**html**
A4	10	22	13	11	3
A3	1	0	0	0	0

DISCUSSION

From the statistical analysis above, we found that the correlation between PPS and NPD is strong for most users as a general tendency. It means it is beneficial for most users to suggest 2in1 printings when the number of pages of the original document is high. It may become important to identify the threshold of the number of the pages to recommend 2in1 printings as a personalizing variable.

Then we investigated the tendency under the hypothesis that the tendency is different among users, and we found the tendency is distinctly different among users for the correlation between DUPL-EXT, PPS-EXT, COL-EXT, and DUPL-NPD. It will be necessary to identify the degree of the influence for each user as personalizing variables.

From the result when we investigated the correlation separately for each month, correlations were found between COL-NPD, PPS-EXT, and MSIZE-EXT for particular users each month. We confirmed these correlations vary with time for particular users. However it is not enough to estimate the effect of variation with time based on the variation with month and further research is necessary.

It is interesting that file-related contexts such as the file extensions and the number of pages of the original documents have correlations with printing-options. It is also interesting that the correlation among printing options is not observed. For example, there is no correlation between duplex printings and the number of copies. Note that the result here holds only for the specific department (sales). Research would be needed to clarify the same result holds for other departments.

The result of the analysis that file-related contexts affect printing options can be utilized to apply the context-aware computing to the printing devices. Additionally we think it have the potential to use other contexts which can identify file type because we think the file extension itself is not the only factor of file type. Inference of the user's preferences based on the predictive statistical model [8][9] would be also essential to the personalized UI. But it is thought to be necessary to specify more external contexts that affect the decision of printing options since the file type is not enough for complete inferences for the whole users.

Our goal is to accomplish a personalized UI for printing devices. Toward this purpose, we made a preliminary design of UI where default options are inferred from the users' past printing options based on the file extensions. We show it in Figure 2. Following two points are considered in this UI:

- Inferred option and selected option are visibly distinguished.
- Correcting the inferred option can be easily achieved.

In this UI, the user's preferences are inferred and showed as default printing options (Figure 2(a)). If the inferences

agree with the user's intention, the user just has to push a "Start" button. If the inference does not agree with the user's intention, the user can easily change the inferred options by pushing buttons below the inferred options. After changing the default options, the color of icons changes only for the changed options (Figure 2(b)).

Igarashi *et al* proposed a suggestive interface for drawing tool that shows thumbnails of future potentials [10]. For the printing devices where there is a time interval between each operation, it would cause cognitive distress if the inference appears closely to user's operation partition as applied in the drawing tool of Igarashi *et al*. Weld *et al* described that saliency is essential in adaptive systems and is increased by "partition dynamics": to segregate dynamic and static area [11]. In this UI, inferred printing options (dynamic part) and buttons for choice (static part) are segregated and cognitive distress would decrease.

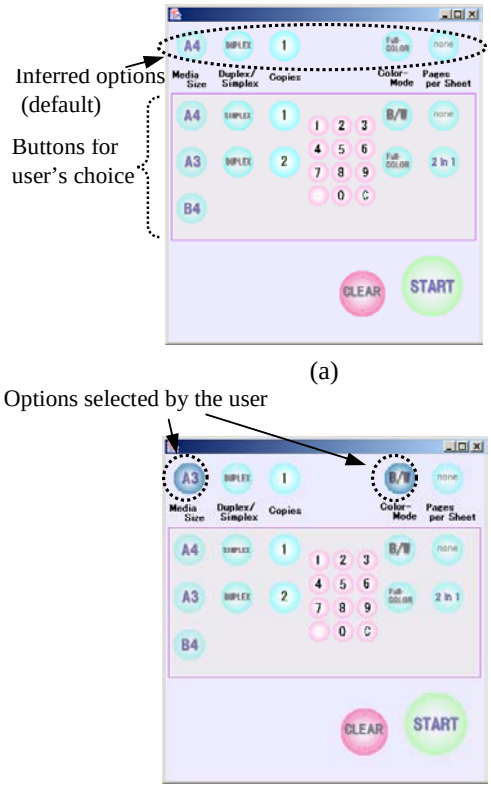


Figure 2. The design of personalized UI for the printing devices where inferred printing options are recommend as default options. (a) represents the default values and (b) represents the UI where options selected by the user are visually distinguished. Note that the user can interactively override the system's recommendation.

CONCLUSION

In this paper, we introduced our analysis of printing logs toward personalized UI where the user's preferences are inferred and suggested based on the contexts. The statistical analysis shows file-related contexts affect printing options and personalization would be achieved in different ways among the type of attributes. We made a preliminary system to infer the printing option using the file extensions, which includes a UI where the users could reach their goal without cognitive distress even when the inference is not correct. Future work is to achieve technique for capturing broader range of potential contexts that would affect printing options.

REFERENCES

1. Trevor, J., Hilbert, D.M., and Schilit, B.N. Issues in Personalizing Shared Ubiquitous Devices, *In Proc. of UbiComp 2002* (2002), 56-72.
2. Grasso, A., and Meunier, J.L. Who Can Claim Complete Abstinence from Peeking at Print Jobs? *In Proc. of CSCW '02* (2002), 296-305.
3. Resnick, P., and Varian, H.R. Recommender systems, *Communications of the ACM* 40, (1997), 56-58.
4. Schafer, J.B., Konstan, J.A., and Riedl, J. E-commerce recommendation applications. *Data Mining and Knowledge Discovery* 5(1/2), (2001), 115-153.
5. O'Hara, K., Perry, M., Churchill, E., and Russell, D. *Public and Situated Displays*, Kluwer Academic Publishers (2003), pp. xx-xxi.
6. Schilit, B.N., Adams, N., and Want, R. Context-Aware Computing-Applications, *In Proc. of IEEE Workshop on Mobile Computing Systems and Applications* (1994), 85-90.
7. Dey, A.K., Salber, D., and Abowd, G.D. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications, *Human Computer Interaction* 16, (2001), 97-166.
8. Zukerman, I., and Albrecht, D.W. Predictive Statistical Models for User Modeling, *User Modeling and User-Adapted Interaction* 11, (2001), 5-18.
9. Hart, P., and Graham, J. Query-free information retrieval, *IEEE Expert* 12, (1997), 32-37.
10. Igarashi, T., Matsuoka, S., Kawachiya, S., and Tanaka, H. Interactive Beautification: A Technique for Rapid Geometric Design, *In Proc. of UIST '97* (1997), 105-114.
11. Weld, D.S., Anderson, C., Domingos, P., Etzioni, O., Gajos, K., Lau, T., and Wolfman, S. Automatically personalizing user interfaces. *In Proc. of IJCAI-03* (2003), 1613-1619.

P2P-based PVR Recommendation using Friends, Taste Buddies and Superpeers

Johan Pouwelse, Michiel van Slobbe, Jun Wang, Marcel J.T. Reinders, Henk Sips
Faculty of Electrical Engineering, Mathematics and Computer Science,
Delft University of Technology,
Delft, The Netherlands
j.a.pouwelse@ewi.tudelft.nl

ABSTRACT

In this paper we present a novel distributed recommendation method based on exchanging similar playlists among *taste buddies*, which takes into account the limited availability of peers, lack of trust in P2P networks, and dynamic identities of peers, etc. Our preliminary experiment shows that only exchanging a small portion of similar playlists from taste buddies could lead to an efficient way to calculate recommendations within a context of P2P networks.

Keywords

Collaborative filtering, recommendation, peer-to-peer, superpeers, taste buddies

INTRODUCTION

The world of Television is finally meeting the PC world. The TV-capture card is becoming a standard PC accessory. Products from television-oriented companies now have to directly compete with products from PC-oriented companies.

Personal Video Recorder

The arrival of the Personal Video Recorder (PVR) is changing the way people watch TV. A PVR enables people to record TV programs on a hard disk. This sounds similar to a common VCR, but the selection of which program to record is much easier. A PVR uses an Electronic Program Guide (EPG) to show the user which programs are broadcasted by a TV station at a certain time. Due to the information in this EPG a PVR can be instructed to, for example, record the program *Boston Public*:

- in this timeslot at this TV station
- any time it is shown on this TV station
- every episode broadcasted on any station

Video compression techniques and ever growing storage capacity ensure that ten of hours of television can be recorded on a PVR. Studies have shown that PVR users shift from watching live TV towards watching the programs on their hard disk. After a few days or weeks of usage, PVR users become less aware of which TV station they are watching and fast-forward through commercials.

The Tivo was the first Consumer electronics (CE) based PVR in the marketplace. Hardware MPEG encoding was used to compress the video stream before storage on the hard disk. Recently, PVR-like functionality was no longer bound to CE hardware with the arrival of PC software such as MythTV [1] on Linux and the Microsoft Media Center Edition of Windows XP [2].

Recommendation

A recommender system can be defined as: a system which "has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options"[24].

A PVR recommendation system observes users TV watching habits either *explicitly* or *implicitly* in order to calculate a recommendation [2, 3, 12, 19]. A recommender system can build an *explicit* profile of a user. Such a profile is filled explicitly by the users ratings. For example, preferred channels, favorite genres, hated actors, etc. An *implicit* profile is based on passive observations and contains the TV watching habits with channel surfing actions. We believe that asking the user to rate programmes is annoying and should be avoided when possible, we therefore use implicit profiles.

A TV recommendation system can either work *stand-alone*, by using only content descriptions (content based) [2, 12], or by *collaborating* and exploiting the profiles of other users

[1] http://www.mythtv.org/
[2] http://www.microsoft.com/windowsxp/mediacenter/default.mspx

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

(collaborative filtering based) [5, 6, 14, 17, 18, 27], or by combining both of them (hybrid) [3, 19, 24]. In this paper we focus on collaborate filtering based recommendation due to its simplicity and high quality.

Recently, a few early attempts towards decentralized collaborative filtering have been made [6, 13, 22]. Canny [6, 7] proposed a distributed EM (Expectation Maximization) update scheme. However, the update is partially distributed since a ”totaler” (server) is still required to connect with all the peers. In [13], a DHTs based technique was proposed to store the user rating data. Those solutions aimed to collect rating data to apply the centralized collaborative filtering and they hold independently of any semantic structure of the networks. This inevitably increases the volume of traffic within the networks. In [30], we introduced fully distributed item-based collaborative filtering algorithm. The similarity between content (items) are derived from the profiles of the different users and stored in a distributed and incremental way as *item-based buddy tables*. By using the item-buddy tables, items are organized in the form of relevance links. Recommendation can then be done according to the similarities stored in the item-buddy tables.

Different with the above approaches, we take into account the limited availability of peers, lack of trust, and dynamic identities of peers in P2P networks. For each target user, our method selects a set of users profiles by measuring the similarity to that user. The *top-N* similar users are then identified in a fully distributed manner and their profiles are employed to make recommendations.

Peer-to-Peer

Peer-to-Peer (P2P) technology is best known for its P2P file sharing programs such as Napster, Kazaa, and Bittorrent. One definition of P2P is “a class of applications that takes advantage of resources – storage, cycles, content, human presence – available at the edges of the Internet” [26].

A general property of P2P technology is its often disruptive nature. This property arises from the fact that P2P technology eliminates central controlling points in both a technical and business sense.

In this paper we use P2P-based sharing of the detailed TV watching habits. The P2P technology must distribute this information and propagate updates. Several papers focus on a method called “epidemic distribution” to disseminate information [9, 10]. Epidemic distribution is based on forwarding information to a set of randomly selected peers. New information from a single source peer is then quickly spread out to numerous peers. This information dies out when nodes have already received this information and no longer forward it to others. This is an example of *undirected* spreading of information or gossiping without a central server.

When peers share a common interest, such as the same TV programs, it is possible to form “virtual communities” around such common interests. When common interests are identified it is possible to spread updates of information using less bandwidth, thus in a more efficient manner.

PROBLEM DESCRIPTION

The aim of this paper is to *produce a scalable, near-zero cost solution for television content recommendation on PVR-style devices using P2P technology*. The scalability aspect is important because few recommendation algorithms scale to millions of users and millions of (archived) television programs to recommend [17]. The aim of creating a near-zero cost solution is motivated by the desire to make a PVR which is not dependant on a central recommendation server. In this paper we will show that P2P technology can replace such a central server which is difficult to scale, contains a wealth of privacy sensitive information, and may require a subscription fee. Recommendations should be calculated in a distributed fashion by the PVRs themselves. The remaining cost for generating a recommendation are then only the modest cost for communication between PVRs and PVR processor usage cost.

Given this aim, we can derive two problem domains. First the collaborative filtering algorithms and second the distribution of TV watching habits using P2P technology. Numerous publications focus on improvements of collaborative filtering algorithms. In this paper we do *not* try to improve existing collaborative filtering algorithms, but to solve the problems which emerge when the most effective algorithms are applied in a P2P setting. The problems we solved are not unique to our recommendation context, but pose a problem in the general P2P domain. We solved the following four P2P-related problem in the context of recommendations:

- Short peer uptime
- Dynamic IP addresses
- Lack of trust
- Selfish behavior

The uptime of a networked PVR is short because many people will deactivate their PVR after use to spare electricity. Each peer in the PVR network may also fail at any time due to, for example, termination of the Internet connection. This creates a severe challenge as the distributed recommender needs to have a high availability, but the underlying PVRs have *limited availability*.

The usage on the Internet of fixed IP numbers is becoming less common. The usage of dynamic IP numbers (DHCP), firewalls, and NAT boxes results in what we call the *dynamic identity problem*, it becomes impossible to directly contact or to guess the IP number of a peer which comes on-line. When all peers in a network use fixed IP numbers it is trivial to contact a previously known peer.

The intermitted on-line/off-line behavior of peers greatly amplifies the severity of the dynamic identity problem. This

amplified problem arises, for example, when two PVR devices with dynamic identities want to exchange information for a P2P recommendation algorithm on a daily basis. When these two PVRs are only simultaneously on-line for a part of the day, it is impossible for them to rendezvous again without a third peer. When the two peers are both behind a firewall, a deadlock even arises because both need to take the initiative to puncture their firewall [21]. We call this amplified problem the *rendezvous deadlock problem*.

Another issue in P2P networks is the lack of trust. The experience with P2P file sharing systems shows that maintaining system integrity while using donated resources from untrusted peers is problematic [23]. We define *integrity* as the ability to ensure that a system operates without unauthorized or unintended modification of any of its components. Calculating recommendations is vulnerable to integrity attacks. Dedicated fans could attack the integrity of the system by, for instance, spreading bogus TV watching habits on a certain TV program. The bogus information would then raise the number of people which have this program recommended. Integrity of P2P systems in general has received little attention from academia. A unique study found that for popular songs on the Kazaa P2P file sharing, up to 70 % of the different versions are polluted or simply fake [16].

Data at several levels in the system can be attacked, namely system information, meta-data level, and the raw data (content) itself. If a P2P system needs to have any integrity the following rule must be followed: *All data obtained from peers in a P2P network is by definition untrusted and must be labeled with its origin and (implicit) trust level*. It is a significant challenge to calculate recommendations and take into account trust levels.

The last problem is the selfish behavior of people. The whole concept behind P2P technology is to pool resources together and use them to deliver services. The resource economy is by definition balanced: resources are not created out of thin air. For instance, in the case of bandwidth, every download MByte has an uploaded MByte. The first challenge in a P2P network is preventing that people do not donate any resources, and are thus *freeriding* on others. In one of the first studies (August 2000) related to freeriding [1], over 35,000 Gnutella peers where followed for one day. Nearly 70 % of the peers did not contribute *any* bandwidth. We define *fairness* in a P2P system as the level to which each user gets back as many resources from the system as they donated.

To calculate distributed recommendations, each peer needs to share its TV watching habits and send regular updates of newly watched programs. Without an incentive to share this privacy, peers will freeride due to their selfish nature. The challenge is thus to create an incentive to share TV watching habits.

Year	Month	Daily TV Minutes
2002	November	190.1
2002	December	204.5
2003	Januari	216.3
2003	February	202.0
2003	March	202.1
2003	April	187.0
2003	May	173.2

Table 1: Average amount of daily television watching in The Netherlands (Source: [28]).

SOLUTION

Our solution to the above four problems is based on three ideas. First, we identify the most reliable peers in the network of PVRs and turn them into superpeers with increased responsibilities. Preferably, these nodes have a fixed IP address. This solves the reliability issues and creates stable node to solve the dynamic identities and rendezvous deadlock problem. Second, each PVR identifies the peers with similar TV taste to efficiently exchange TV watching habits (and updates). This is the method for distributing the collaborative filtering. We present simulation results showing the recall rate of our method for distribution. Third, the end-user must enter in the PVR which other PVR users are his friends to build a network of trust. By using this information we can calculate who is a friend, friend-of-a-friend, etc. and use this to information improve our second idea.

Superpeers

We first explore the problem of limited uptime in more depth before we discuss the superpeer idea. Table 1 shows the average number of minutes people where watching television in The Netherlands [28]. From these numbers we can derive a first estimate for PVR uptime, if we assume that a PVR is only online when the attached television is in use. These number indicate that the average peer uptime is roughly between 170 and 215 minutes, depending on the season.

Instead of merely the average uptime, it is also important to know the distribution of uptime between PVR nodes. A good source of information on the great diversity in peer uptime is from the field of P2P file sharing. We assume that the PVR uptime will show some similarity to peer uptime in P2P file sharing networks and therefore discuss it at length.

Several measurement studies of P2P file sharing networks have addressed the issues of peer availability [4, 8, 11, 25]. Most of the availability studies only span a few days [4] or weeks [8], making it difficult to draw conclusions on long-term peer behavior.

In P2P file sharing networks, a very small fraction of the peers has a high uptime (measured in weeks) because they are never turned off and have a reliable Internet connection. We believe that for networked PVR systems, a small percentage of users will leave their PVR on continuously.

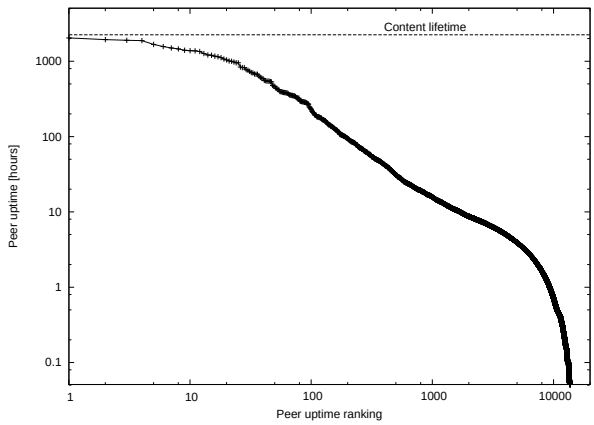


Figure 1: The uptime distribution of 53,833 peers on the Bittorrent P2P file sharing network (Source: [23]).

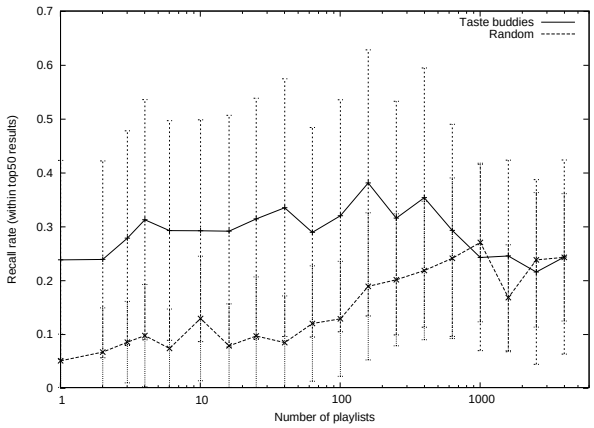


Figure 2: Comparison of the performance for random peer contacts and taste buddy contact only.

In a previous study the authors measured peer availability for over three months in the Bittorrent P2P file sharing system [23].

On December 10, 2003 the popular PC game "Beyond Good and Evil" from Ubisoft was injected into BitTorrent using the web site Suprnova.org and on March 11, 2004 it died. We measured the uptime of 53,883 peers which downloaded this content.

Figure 1 shows the results of our uptime measurements. Here we plot the peer uptime in hours after they have finished downloading. The horizontal axis shows the individual peers, sorted by uptime. The time scale for the uptime ranges from 3 minutes to nearly 7 months. The longest uptime is 83.5 days. Note that this log-log plot shows an almost straight line between peer 10 and peer 5,000. The sharp drop after 5,000 indicates that the majority of users disconnect from the system within a few hours after the download has been finished. This sharp drop has important implications because the actual download time of this game spans several days. Figure 1 shows that peers with a high availability are rare. Only 9,219 out of 53,883 peers (17 %) have an uptime longer than one hour after they finished downloading. For 10 hours this number has decreased to only 1,649 peers (3.1 %), and for 100 hours to a mere 183 peers (0.34 %).

From the Bittorrent measurements we conclude that a very small group of peers is significantly more reliable than the average peer. Some Bittorrent users leave their computer running for days and we assume a small percentage of PVR users will also leave their device on for days or perhaps even permanently.

We exploit this knowledge on the skewed distribution of uptime in the superpeer idea. A supernode is "a peer which has a heightened function, accumulating information from numerous other peers", according to the definition used in the MGM studios versus Grokster case [29]. The Kazaa file sharing system uses superpeers to implement a fast file search algorithm and NAT traversal [15]. We use this concept to distributed the recommendation.

Our solution works as follows. Each PVR keeps a large list of other peers it has seen on the network. The average uptime of each peer is used to sort this list from most reliable peer to least reliable peer. PVRs can exchange such lists and can thus discover the IP addresses of new PVRs. Note that the bandwidth requirements from exchanging such lists are modest because IP number and timestamps only require a few bytes. Each peer can request any other peer to store its current IP number. Reliable peers receive many of such requests and automatically become a reliable source of information to overcome the dynamic identity problem and the rendezvous deadlock problem. When peers come online they register their current IP number at multiple superpeers, allowing others to find them again.

Without any load balancing measures this superpeer algorithm would quickly overwhelm the most reliable peers in the network with numerous requests. We will use the friends concept to implement load balancing. Peers prefer to store their current IP number on reliable peers which belong to a friend, friend-of-a-friend, etc.

The benefit of our superpeer method is its simplicity and low overhead. It does not offer any guarantees that the new IP number of a peers becomes known to all, but in our architecture this is also not required.

Taste buddies

This section explains our solution to calculate accurate recommendations with information exchanges with just a few peers. We present simulation results which show that regular exchanges of TV habits with 100 peers is sufficient to achieve a good recommendation recall rate.

The central concept in our taste buddy idea is the exchange of TV habbits which are stored in playlists. A playlist is a complete list of programs which the user has seen in the past or explicitly programmed for recording in the future. Thus, a sufficiently large number of playlists will also contain in-

formation about content which is not even broadcasted.

A simple method to exchange playlists is to contact a random peer and swap playlists. The implementation of this *random peer contact* method requires only a procedure to obtain IP addresses of peers. Due to the lack of trust in a P2P network, a minimal amount of contact with other peers is desired. Especially for the exchange of playlists because less contact lower the risk of integrity attacks.

Instead of contacting peers at random, we use a cosine similarity function [5] to identify peers with similar playlists. This ensures that we obtain playlists from our *taste buddies* only. We then apply the Amazon collaborative filtering algorithm [17] on these collected similar playlists to calculate the recommendations.

We used the MovieLens data-set [20] to evaluate the performance. We divided the MovieLens database into a training part and a testing part. We use 20 % of a users best-rated movies as the testing set. *Recall* rate is employed to measure the performance. It measures the proportion of the ground truth (GT) of the liked-items that are really recommended, which is shown as follows:

$$Recall = \frac{|liked\ items(GT) \cap recommended\ items|}{|liked\ items(GT)|} \quad (1)$$

where $|S|$ indicates the size of the set S.

The recall is calculated from the return of the Top-50 highest-ranking recommended items. We compare the recall of our taste-buddy approach to the random peer contact approach, shown in Figure 2. We varied the number of playlists which are collected to calculate a recommendation. Each data point shows the average for over 100 different runs of the recommendation. From the left to right side, playlists are exchanged from one single user to all the users, up to the size of the data-set (6,040 users).

From the figure, we observed that:

- In general, our taste-buddy approach outperforms the random peer contact approach. The performances of the two approaches converge when the playlists are fully collected.
- The recall rate of our taste-buddy approach increases with the number of the collected similar playlists increases. It reaches the peak when collecting about 130 similar playlists. Then the performance decreases as the number of the collected playlists increases. Contrarily, the recall rate of the random peer contact keeps increasing as the number of the collected playlists increases.

From these results we can conclude that only a small portion of taste buddies (similar playlists) are needed to calculate accurate recommendations. In a network of millions of PVRs, the similarity function can be used to quickly identify the peers with similar taste. We expect that this network will quickly cluster taste buddies together. The superpeer concept needs to ensure that each peer comes into contact with many peers while the similarity function will ensure the clustering.

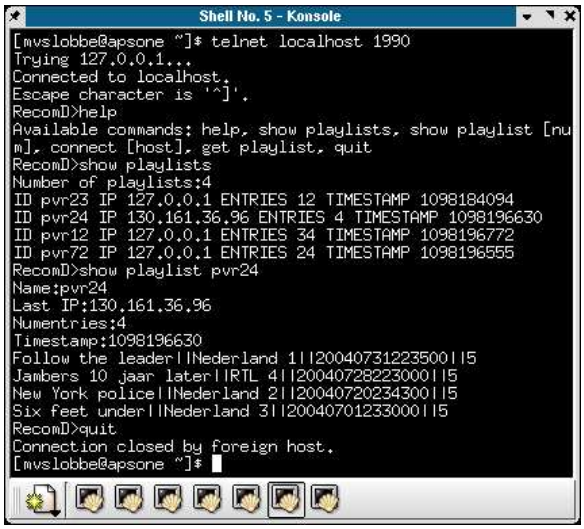


Figure 3: Debugging view of the MythTV implementation.

Friends

To solve the trust issue in a P2P network, we ask the user to identify his/her real-world friends. By using social software similar to `Orkut.com` we can identify the social relation and distance to the users of two PVR nodes. This means that the PVR knows who your friends, friends-of-friends, etc. are on the Internet and exploit this information to ensure system integrity.

The research area of *social-aware* PVRs is still poorly understood. The idea is to set hard thresholds on the maximum social distance that a supernode user or taste buddie can be. The simple hard threshold is easy to implement and already provides a solid method to guard integrity. The concept of friends also reduces the selfish behavior, you do not freeride on your social friends.

IMPLEMENTATION

We are currently implementing all our ideas in the MythTV Open Source PVR (`MythTV.org`). Figure 3 depicts the inner working of this implementations. We created a daemon which is able to show and exchange playlists. This daemon has a command line interface with basic display and exchange commands.

Figure 4 shows the output of the Amazon recommendation algorithm which is now fully integrated within MythTV. We are currently looking into distributing our MythTV recommender to a large number of people to get feedback and to make some improvements.

Discussion & Conclusion

We have identified four problems when recommendations are distributed in a P2P network which are limited uptime, dynamic identities, lack of trust, and selfish behavior.

Figure 4: MythTV implementation screenshot.

By using our ideas of superpeers, taste buddies, and friends we are able to address all four problems.

We have implemented a distributed recommender using P2P technology and are currently testing its limits.

REFERENCES

1. E. Adar and B. A. Huberman. Free riding on gnutella. Technical report, Xerox PARC, August 2000.
2. L. Ardissono, A. Kobsa, and M. Maybury, editors. *Personalized Digital Television: Targeting Programs to Individual Viewers*. Kluwer Academic Publishers, 2004.
3. C. Basu, H. Hirsh, and W. W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *AAAI/IAAI*, pages 714–720, 1998.
4. R. Bhagwan, S. Savage, and G. M. Voelker. Understanding availability. In *International Workshop on Peer to Peer Systems*, Berkeley, CA, USA, February 2003.
5. J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of UAI*, 1998.
6. J. Canny. Collaborative filtering with privacy via factor analysis. In *Proc. of ACM ICIR*, 1999.
7. J. Canny. Collaborative filtering with privacy. 2002.
8. J. Chu, K. Labonte, and B. Levine. Availability and locality measurements of peer-to-peer file systems. In *ITCom: Scalability and Traffic Control in IP Networks*, Boston, MA, USA, July 2002.
9. P. T. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulie. From epidemics to distributed computing. *IEEE Computer*, 2004.
10. A. J. Ganesh, A.-M. Kermarrec, and L. Massouli. Peer-to-peer membership management for gossip-based protocols. *IEEE Trans. Comput.*, 52(2):139–149, 2003.
11. K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *19-th ACM Symposium on Operating Systems Principles*, Bolton Landing, NY, USA, October 2003.
12. S. Gutta, K. Kurapati, K. P. Lee, J. Martino, J. Milanski, J. D. Schaffer, and J. Zimmerman. Tv content recommender system. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 1121–1122. AAAI Press / The MIT Press, 2000.
13. P. Han, B. Xie, F. Yang, and R. Sheng. A scalable p2p recommender system based on distributed collaborative filtering. *Expert systems with applications*, 2004.
14. T. Hofmann and J. Puzicha. Latent class models for collaborative filtering. In *Proc. of IJCAI*, 1999.
15. J. Liang, R. Kumar, and K. Ross. The kazaa overlay: A measurement study. September 2004.
16. J. Liang, R. Kumar, Y. Xi, and K. Ross. Pollution in p2p file sharing systems. In *IEEE Infocom*, Miami, FL, USA, March 2005.
17. G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.
18. B. Marlin and R. S. Zemel. The multiple multiplicative factor model for collaborative filtering. In *Proc. of ICML*, 2004.
19. P. Melville, R. Mooney, and R. Nagarajan. Content-boosted collaborative filtering. In *ACM SIGIR Workshop on Recommender Systems*, Sept. 2001.
20. *MovieLens dataset*, as of 2003. `http://www.grouplens.org/data/`.
21. T. Oh-ishi, K. Sakai, T. Iwata, and A. Kurokawa. The deployment of cache servers in p2p networks for improved performance in content-delivery. In *Third International Conference on Peer-to-Peer Computing (P2P'03)*, Linkoping, Sweden, September 2003.
22. T. Oka, H. Morikawa, and T. Aoayama. Vineyard : A collaborative filtering service platform in distributed environment. In *Proc. of the IEEE/IPSJ Symposium on Applications and the Internet Workshops*, 2004.
23. J. Pouwelse, P. Garbacki, D. Epema, and H. Sips. A measurement study of the bittorrent peer-to-peer file-sharing system. Technical Report PDS-2004-007, Delft University of Technology, Apr. 2004.
24. R.Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 2002.
25. S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Multimedia Computing and Networking (MMCN'02)*, San Jose, CA, USA, January 2002.
26. C. Shirky. What is p2p... and what isn't, 2000. `http://www.openp2p.com/pub/a/p2p/2000/11/24/shirky1-whatisp2p.html`.
27. L. Si and R. Jin. Flexible mixture model for collaborative filtering. In *Proc. of ICML*, 2003.
28. Stichting KijkOnderzoek. Resultaten van het onderzoek, 2004. `http://www.kijkerspanel.nl/resultaten.php`.
29. S. V. W. United States District Court, Central District of California. Mgm studios versus grokster; opinion granting def. motions for partial summary judgment, 2003. `http://www.eff.org/IP/P2P/MGM_v_Grokster/`.
30. J. Wang, M. J. T. Reinders, R. L. Lagendijk, and J. Pouwelse. Self-organizing distributed collaborative filtering. Submitted to WWW05, Oct 2004.

DynamicLens:
A Dynamic User-Interface for a Meta-Recommendation System

J. Ben Schafer
Department of Computer Science
University of Northern Iowa
Cedar Falls, IA 50614-0507 USA
+1 319 273 2187
schafer@cs.uni.edu

ABSTRACT
Recommendation systems help users find items of interest. Meta-recommendation systems provide users with personalized control over the combination of recommendation data from multiple information sources. In the process, they provide users with more helpful recommendations by allowing users to indicate how important each parameter is in their decision process, and how data should be weighted during recommendation generation. Most current meta-recommendation systems require the submission of large, form-based queries prior to the receipt of recommendations. Such systems make it difficult for a user to conclude what effect a given requirement has on the overall recommendations. This paper considers the construction of an interface to allow dynamic queries for a meta-recommender. It is believed that the addition of a dynamic query interface will provide users with more meaningful meta-recommendations by allowing them to explore these causes and effects.

Keywords
Meta-recommendation systems recommendation system, collaborative filtering, dynamic query interface.

INTRODUCTION
On a daily basis we are faced with information overload as we choose from an overwhelming number of options. To keep abreast of the latest developments in our career field, we can choose from a whole host of journal articles, conference proceedings, textbooks, and web sites. During our personal time we must choose which television show to watch, which movie to see, which CD to play, or which book to read. The number of options from which to choose in each of these categories is often more than we can possibly process.

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

Recommender Systems have emerged as powerful tools for helping users reduce information overload. Such systems employ a variety of techniques to help users identify items of interest [5]. For example, a recommender system in the domain of movies might suggest that a user go see *Ladder 49* because she requested films classified as "drama" (query fit using information retrieval), because she has previously liked films starring John Travolta (personalization using information filtering), or because people like her have indicated it was a movie they enjoyed (personalization using collaborative filtering). Regardless of technique, these systems attempt to help users identify the items that best fit their needs, their tastes, or even both.

This paper discusses a class of recommendation interface known as *meta-recommendation systems.* These systems present recommendations fused from "recommendation data" from multiple information sources. Meta-recommendations systems encourage users to provide both ephemeral and persistent information requirements. The systems use this data to produce recommendations that blend query-fit with long-term personalization. For example, MetaLens – a real- time meta-recommender in the domain of movies – might recommend Ladder 49 based not only on a combination of the reasons previously mentioned, but also based on the fact that the user indicated she requires a movie showing in her local theater that starts after 9:00 PM. Furthermore, these systems provide a high level of user control over the combination of recommendation data, providing users with more unified and meaningful recommendations. This paper also presents early work in the development of a dynamic query interface for a meta-recommendation system.

RELATED WORK
The earliest "recommender systems" were information filtering and retrieval systems designed to fight information overload in textual domains. Recommender systems that incorporate information retrieval methods are frequently used to satisfy ephemeral information needs from relatively static databases. Conversely, recommender systems that

incorporate information filtering (IF) methods are frequently used to identify items that match relatively stable and specific information needs in domains with a rapid turnover or frequent additions. Although information retrieval and information filtering are considered fundamentally different tasks [1], they are based on similar techniques. This paper will consider both under the singular term "information filtering." Systems which use IF techniques include RE:Agent [2], Ripper [5], NewT [9], and Amalthaea [10].

Collaborative filtering (CF) is an attempt to facilitate the process of "word of mouth." Users provide the system with evaluations of items that may be used to make "recommendations" to other users. The simplest of CF systems provides generalized recommendations by aggregating the evaluations of the community at large. More advanced systems personalize the process by forming an individualized neighborhood for each user consisting of a subset of users whose opinions are highly correlated with those of the original user. Recommender systems based on collaborative filtering include MovieLens [6], Tapestry [7], GroupLens [12], Ringo [17], and PHOAKS [19].

Hybrid Recommender Systems

As researchers have studied different recommender system technologies, many have suggested that no single technology works for all situations. Thus, hybrid systems have been built in an attempt to use the strengths of one technology to offset the weaknesses of another. Burke [3] discusses several different hybridization methods, but points out that most hybrid systems involve the combination of collaborative filtering with either a content-based (IF) or data mining technique.

Tango [4] recommends articles in the domain of an online newspaper. It does so by creating separate recommendations from CF and IF algorithms and merging these using a separate combination filter. The combination filter employed by Tango uses per-user, per-article weights. The calculation of these weights takes into account the degree of confidence each filter has in a particular document's recommendation, as well as error analysis for each filter's past performance for the user in question.

Torres et al. [20] present the results of several experiments involving TechLens. Similar to Tango, TechLens combines both a collaborative filter and a content-based filter to recommend research papers. In both offline and online studies they consider five different algorithms for combining the recommendations from these filters, including sequential algorithms. These techniques take the recommendations from one filter as a seed to the second filter. They conclude that different algorithms should be used for recommending different kinds of papers, although they discovered that sequential algorithms tend to produce poor results under most circumstances.

The SmartPad supermarket product recommender system [8] suggests new or previously unpurchased products to shoppers creating shopping lists on a personal digital assistant (PDA). The SmartPad system considers a consumer's purchases across a store's product taxonomy. Recommendations of product subclasses are based upon a combination of class and subclass associations drawn from information filtering and co-purchase rules drawn from data mining. Product rankings within a product subclass are based upon the products' sales rankings within the user's consumer cluster, a less personalized variation of collaborative filtering.

Nakamura and Abe [11] describe a system for the automatic recording of programs using a digital video recorder. They implement a set of "specialist" algorithms that use probabilistic estimation to produce recommendations that are both content-based (based on information about previously recorded shows from the electronic program guide) and collaborative (based on the viewing patterns of similar users).

META-RECOMMENDERS

Consider the following scenario. Mary's 8-year-old nephew is visiting for the weekend, and she would like to take him to the movies. Mary has several criteria for the movie that she will select. She would like a comedy or family movie rated no "higher" than PG-13. She would prefer that the movie contain no sex, violence or offensive language, last less than two hours and, if possible, show at a theater in her neighborhood. Finally, she would like to select a movie that she herself might enjoy.

Traditionally, Mary might decide which movie to see by checking the theater listings in the newspaper and asking friends for recommendations. More recently, her quest might include the use of the Internet to access online theater listings and search databases of movie reviews. Additionally, she might be able to obtain personalized, CF-based recommendations from a web site such as MovieLens. Producing her final selection, however, requires a significant amount of manual intervention; Mary must visit each source to gather the data and then decide how to apply this data in making her final decision.

The hybrid systems mentioned in the previous section are a significant step toward solving problems like Mary's. A hybrid movie recommendation system would provide Mary with lists of movies blended from her long-standing collaborative filtering and content-interest profiles. It is likely, however, that such a system would not offer her the ability to provide information that might improve the recommendations produced by the combination algorithm. For example, if given access to the combination algorithm, Mary could indicate that predictions should be biased less towards the British art films she frequently likes and more toward the family movies appropriate for her nephew, or that the movie should be relatively free of offensive language and last less than two hours.

Prior work [15,16] has defined a new form of hybrid system with the level of user control needed to allow for

the meaningful blending of recommendations from multiple techniques and sources. These systems, known as meta-recommenders, provide users with personalized control over the generation of a single recommendation list formed from a combination of rich data using multiple information sources and recommendation techniques. Based on the lessons we learned from existing hybrid systems, we built the MetaLens Recommendation Framework (MLRF), a general architecture for the construction of meta-recommenders. Using this framework, we implemented MetaLens, a meta-recommender for the domain of movies. Much like Mary, who makes her final choice by examining several movie data sources, MetaLens uses IF and CF technologies to generate recommendation scores from several Internet film sites.

The user interface for MetaLens centers on two screens. On the preferences screen, users indicate their ephemeral requirements for their movie search. They do this by providing information concerning nineteen features of movies and theaters including genre, MPAA rating, critical reviews, and distance to the theater (Figure 1). For each feature the user may indicate the specific factors he considers important (e.g., "I want to see a film from the 'comedy' or 'family' genre"), a weight that indicates how important it is that the recommended movie matches these factors (e.g., "It is very important that the movie I see be one of the genres I selected") and a "Display Info?" selection which indicates that data related to the specific feature should be included with the recommendations. As an example, Figure 1 might represent a portion of Mary's requirements for the movie that she views with her nephew.

The meta-recommendation algorithm is based on the extended Boolean information retrieval algorithm proposed by Salton et al [13] as a way to rank partial matches in Boolean queries in the domain of document retrieval. This algorithm is an ideal initial choice for meta-recommenders. In essence, Mary submits a query that says "I want a movie that is a comedy or family movie rated no "higher" than PG-13, containing no sex, violence or bad language, lasting less than two hours and, showing at a theater in my neighborhood." A traditional Boolean query of these requirements will return only movies matching ALL of these features. Most users, however, will settle for a movie matching a majority of these features.

MetaLens judges overall query fit based on recommendation scores from these multiple data sources. No attempt is made to resolve potential information conflicts. Instead, each piece of data is converted as-is**,** and the item match scores combined to calculate a query-fit score for each triple. These recommendations are sorted to contain only the highest-rated triple for each movie – each movie is recommended once in conjunction with the theater and show time that best fits the user's requirements – and the final recommendations displayed. Thus, Figure 2 might represent the MetaLens recommendations concerning which movie Mary should take her nephew to see.

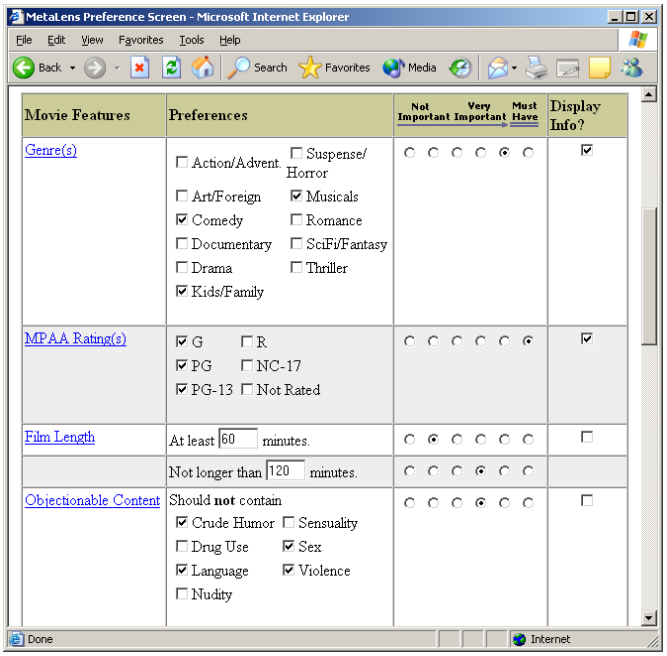


Figure 1: MetaLens Preferences Screen. Users provide information regarding which items among nineteen features are important, the degree to which each recommendations must match the features selected, and what data should be included with final recommendations.

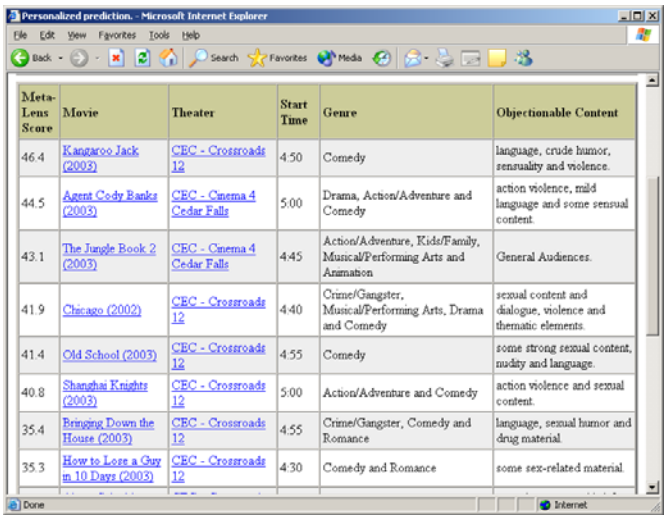


Figure 2: MetaLens Recommendation Screen. Users are provided with a list of {movie, theater, showtime} triples ranked according to how well each triple matches the query provided in the preferences screen.

DYNAMIC META-RECOMMENDERS

DynamicLens

One of the advantages of meta-recommenders is that they involve such a rich assortment of recommendation data. Prior research has concluded that users accept meta-recommendation systems such as MetaLens and that they believe such systems provide them with more meaningful recommendations when compared to "traditional" systems [16]. While the current interface is similar to other comparable recommender systems, user interface design experts like Shneiderman [18] would argue that the current interface does not allow users to interact properly with the data. In order for a user of MetaLens to "tweak" a query, the user must return to the preferences screen, modify the requirements or weights for each feature, and resubmit the query to the system. While careful trial and error may indicate the effect different requirements have on the final recommendations, this process can be time consuming and difficult.

This leads to the question how would the recommendation process change as modifications are made to the various interfaces with which users interact. In particular, what would be the affect of "dynamic query" interfaces on the way users interact with a meta-recommender.

In order to begin consideration of answering such a question a new interface was built for MetaLens that employs Shneiderman's concept of a dynamic query interface. DynamicLens uses the same underlying algorithms and data as MetaLens. However, it merges the preferences interface and the recommendations interface within a single interface. Users select which items are of interest in their query, and to which extent each recommendation should match these items through a preferences panel located on the left side of the interface (Figure 3). Recommendations based on the current query are displayed in the right-hand panel of the interface. Individual changes to the preferences panel generate an automatic and immediate update to the recommendations panel.

There are several potential advantages of a system such as DynamicLens. First, it allows users immediate feedback on the affects of each query requirement. For example, a user adding "violence" to his list of objectionable content can immediately observe which movies drop lower in the recommendations list. Similarly, a user raising the importance factor from "very important" to "Must match" can observe how many movies will be eliminated from contention for not matching her currently selected items. In both cases, the user has the feedback necessary to understand how such requirements affect the final outcome from the recommendation engine. The user can choose to maintain these requirements, select to "soften" the requirement, or choose to eliminate the requirement completely in order to generate recommendations he or she feels will best suit the current needs.

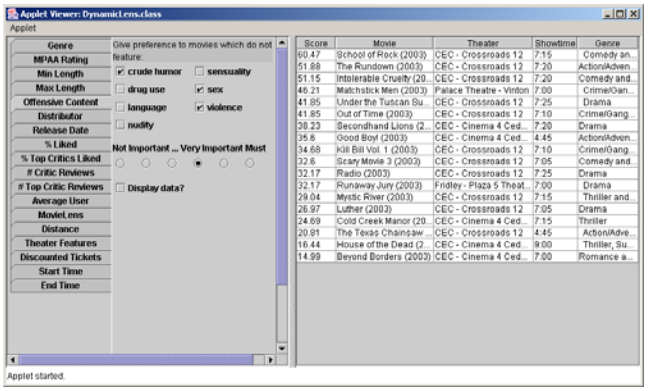


Figure 3: DynamicLens merges the preferences interface with the recommendation interface to provide users with a direct manipulation interface. Users are provided with immediate feedback regarding the effect caused by individual changes to their preferences.

Looking at this from a slightly different angle, a user is better able to interact with a direct query interface in an attempt to discover why a given set of ranking recommendations were made. For example, a user may provide his recommendations and wonder why a given film is recommended so low in the rankings. A dynamic query interface allows him to modify the current recommendation query in an attempt to see which requirement(s) pushed the item so low in the list. In doing so, the user has the ability to gather information that can help him decide that the film really is inappropriate given the current set of requirements.

But is it worth it?

While users were very accepting of MetaLens, it has yet to be shown that they will be similarly accepting of DynamicLens. One advantage of a multi-screen interface like MetaLens is that the interface fits within a standard metaphor of "build query/analyze results." Furthermore, very few instructions are needed and those that are fit naturally within the appropriate interface.

The potential pitfalls to a dynamic interface like DynamicLens include the decrease in real estate for "natural" instructions, and the increased complexity that comes with adapting what was formerly multiple screens into a single screen. While it is natural in MetaLens to state "tell us what you are looking for" at the top of a long query form, it is less natural to do so when the various elements of the query are divided among multiple tabbed panels. Furthermore, these tabbed panels provide an increase in complexity which may cause a decrease in usefulness – e.g. if a user can't figure out how to modify the preferences to match his needs, the tool rapidly becomes one with no real use. While the general concept of tabbed option panels has become increasingly common,

anecdotal experience has suggested that many users continue to be confused by their use. A review of the research literature failed to yield any usability studies discussing the long-term effectiveness and usability of such an interface. Clearly, DynamicLens will be affected by the overall acceptance of tab-based interfaces.

Acceptance of a new interface is not solely based on a user's perception of usability, however. It is also based on a user's belief in the overall appropriateness of the interface. Several potential benefits of this new interface were proposed in the previous section. However, it remains unclear if users will actually notice that the interface allows them to discern the impact of each recommendation attribute and, thus, detect the benefits of the interface. Worse yet, users may notice, but find the knowledge unimportant. In either case, the complexity introduced into the interface in an effort to provide dynamic queries becomes inappropriate.

It remains to be seen whether controlled user studies will indicate that users consider DynamicLens either usable or appropriate.

CONCLUSIONS

This paper has discussed meta-recommenders – a relatively new way to help users find recommendations that are understandable, usable, and helpful. Furthermore, this paper has considered ways in which the interfaces for such recommenders might be improved through the addition of dynamic queries. All told, it is believed that there is great potential for such interfaces to change the way in which users gather information for decision-making

ACKNOWLEDGMENTS

The author would like to thank the members of the GroupLens Research group at the University of Minnesota who assisted with the early versions of this research.

REFERENCES

1. Belkin, N.J. and Croft, W.B. (1992). Information Filtering and Information Retrieval: Two Sides of the Same Coin? CACM 35(12) pp.29-38.
2. Boone, G. (1998). Concept Features in RE:Agent, an Intelligent Email Agent. Proceedings of Autonomous Agents 98. pp.141-148.
3. Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction, v. 12, pp 331-370.
4. Claypool, M., Gokhale, A., and Miranda, T. (1999). Combining Content-Based and Collaborative Filters in an Online Newspaper. ACM SIGIR Workshop on Recommender Systems.
5. Cohen, W.W. (1996). Learning Rules that classify E-mail. Proceedings of the AAAI Spring Symposium on Machine Learning on Information Access.
6. Dahlen, B.J., Konstan, J.A., Herlocker, J.L., Good, N., Borchers, A., Riedl, J. (1998). Jump-starting movielens: User benefits of starting a collaborative filtering system with "dead data". University of Minnesota TR 98-017.
7. Goldberg, D., Nichols, D., Oki, B.M., and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. CACM 35(12) pp.31-70.
8. Lawrence, R.D., Almasi, G.S., Kotlyar, V., Viveros, M.S., and Duri, S.S. (2001). Personalization of Supermarket Product Recommendations. Data Mining and Knowledge Discovery 5(1/2) pp 11-32.
9. Maes, P. (1994). Agents that Reduce Work and Information Overload. CACM 37(7) pp.31-40.
10. Moukas, A. and Zacharia, G. (1997). Evolving a Multi-agent Information Filtering solution in Amalthaea. Proceedings of Autonomous Agents 97 pp.394-403.
11. Nakamura, A. and Abe, N. (2000). Automatic Recording Agent for Digital Video Server. Proceedings of MM-00 pp.57-66.
12. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. CSCW 94, pp. 175-186.
13. Salton, G., Fox, E., and Wu, H. (1983). Extended Boolean Information Retrieval. CACM 26(11) pp.1022-1036.
14. Schafer, J.B., Konstan, J.A., and Riedl, J. (2001). E-Commerce Recommendation Applications. Data Mining and Knowledge Discovery 5(1/2) pp.115-153.
15. Schafer, J.B., Konstan, J.A., and Reidl, J. (2002). Meta-recommendation Systems: User-controlled Integration of Diverse Recommendations. Proceedings of CIKM-02 pp. 196-204.
16. Schafer, J.B., Konstan, J.A., and Reidl, J. (2004). The View through MetaLens: Usage Patterns for a Meta-Recommendation System. IEE Proceedings Software. Anticipated publication December 2004.
17. Shardanand, U. and Maes, P. (1995). Social Information Filtering: Algorithms for Automating Word of Mouth. Proceedings of CHI-95 pp.210-217.
18. Shneiderman, B. and Plaisant, C. (2005). Designing the User Interface: Strategies for Effective Human-Computer Interaction, Fourth Edition. Addison Wesley Longman, Inc.
19. Terveen, L., Hill, W., Amento, B., McDonald, D., and Creter, J. (1997). PHOAKS: A System for Sharing Recommendations. CACM 40(3) pp.59-62.
20. Torres, R., McNee, S., Abel, M., Konstan, J.A., and Riedl, R. (2004). Enhancing Digital Libraries with TechLens. Proceedings of JCDL'04. pp 228-236.

Modeling a Dialogue Strategy for Personalized Movie Recommendations

Pontus Wärnestål
Department of Computer Science
Linköping University
ponjo@ida.liu.se

ABSTRACT

This paper addresses conversational interaction in user-adaptive recommender systems. By collecting and analyzing a movie recommendation dialogue corpus, two initiative types that need to be accommodated in a conversational recommender dialogue system are identified. The initiative types are modeled in a dialogue strategy suitable for implementation. The approach is exemplified by the MADFILM movie recommender dialogue system.

INTRODUCTION

A crucial issue for recommender system performance is the way the system acquires preferences from the user in order to generate a correct and sufficient preference model [3]. In general, this issue can be addressed by two approaches: The first is to improve or combine recommendation techniques and algorithms in order to predict items to a specific user. This approach is focused on the internal workings of the recommender system, and has resulted in significant advances [3]. The second approach is focused on exploring interaction models and techniques for conveying preferences to the system, and is thus more interaction-oriented. Of the two, the former approach has received the most attention, while the latter has been neglected to a large extent [5].

Advances in natural language processing have provided techniques for building speech interfaces and *spoken dialogue systems* [11]. The most common type of such systems is information-providing dialogue systems, and this research body has provided a number of frameworks and methods for building such systems. Natural language interaction and dialogue may have a positive impact on *recommender system* interaction and performance, since it allows a wide range of users to express their preferences in new ways, e.g. attribute tweaking and the candidate/critique model [4, 10]; and in various environments, e.g. using speech for personalized recommendations while driving [13].

Consider the example dialogue between a user and a spoken dialogue recommender system in Figure 1. This dialogue

U1: Hi! I would like a movie recommendation please
S1: Okay. What kind of movie do you feel like?
U2: I feel like watching a thriller
S2: Can you tell me one of your favorite thriller movies?
U3: Uhm, I'm not sure
S3: Okay. Can you tell me one of your favorite actors or actresses?
U4: I think *Bruce Willis* is good
S4: I see. Have you seen *Pulp Fiction*?
U5: No
S5: *Pulp Fiction* is a thriller starring *Bruce Willis*

Figure 1: Example recommendation dialogue about movies between the MADFILM system (S) and a user (U). Movie titles and actor names are in *italics*.

gives an idea of how recommendation dialogues in MADFILM are carried out. The system guides the user by asking for relevant preferences, but remains flexible depending on the user's responses (as in utterances U3 and S3 in Figure 1). Recommendations are presented and explained in a conversational manner (S5), based on the gathered user preferences (U2 and U4).

This paper aims to make a contribution on natural language dialogue in user-adaptive recommender systems. Specifically, two issues are addressed: (1) how recommendation dialogues between humans are manifested, and (2) how recommendation dialogue strategies can be modeled in a spoken recommender dialogue system. The contribution put forward is a dialogue control strategy suitable for implementation. The approach has been implemented in the MADFILM system, which is described in detail in [8].

HUMAN-HUMAN RECOMMENDATION DIALOGUES

In order to get empirical data covering aspects of recommendation dialogues, a corpus of human-human dialogues was collected. This section describes the study and its implications on dialogue strategy design.

Participants

Forty-eight participants (24 female and 24 male) were recruited. The participants did not know each other's movie

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

preferences. Each dialogue session required two subjects, one acting in the role of a movie *recommender*, and the other in the role of a *client* wanting movie recommendations. In order to avoid repetition of recommendation strategies in the dialogues, each session had a new recommender. All sessions were kept as varied as possible in terms of gender and roles, including male-female, male-male and female-female dialogues, in both recommender and client roles in order to vary the dialogues as much as possible.

Apparatus

The study was set in a home environment designed for usability studies. Apparatus for the study included a laptop connected to an extensive movie information database [1] for the recommender to use, and a movie recommendation protocol for keeping track of movie recommendations. The dialogue sessions were recorded on Mini Disc.

Procedure

The recommender got a 15-minute tutorial on how to use the movie information database prior to the session. She also received a scenario, which provided her with the task of recommending in total ten movies. Five of these should have been seen by the client and verified as "good" recommendations (in order to indicate that recommendations were fit for the current client), and five should be previously unseen. When the recommendation protocol was completed, the session was terminated. The client was also presented with a scenario, which explained the client role. A total of 24 dialogues were recorded with two participants in each session, resulting in 7.5 hours of recorded material. Transcription of the dialogues resulted in 2684 utterances with a mean of 112 utterances per dialogue.

Results

When analyzing the material a number of dialogue phenomena were identified and quantified. The results include a categorization of critical issues to consider when modeling recommendation dialogues. In particular, two initiative types were identified and is the topic of this section. The material has been analyzed from other aspects as well, e.g. multimodal interaction, object manipulation, and global focus shifts. These aspects of the study are covered in [7].

Information Requests

The first of the two initiative types—*information requests*—is concerned with information about movies found in the database. Information requests are client-driven and occur when clients ask about properties (e.g. director or actor information) for a specific movie. Requests may be posed as a stand-alone initiative introduced by the client, or as a subdialogue within a recommendation initiative. Figure 2 shows an example of this.

[1] The Internet Movie Database (http://us.imdb.com).

R1: have you seen *The Bone Collector*?
C1: who is acting in it?
R2: these guys [displays a list of actors]
C2: yeah / I liked that one

Figure 2: Dialog excerpt where the client issues an information request about a movie (C1) in order to respond to the recommender's question (R1).

Preference Requests and Recommendations

The second initiative type—*preference requests*—is concerned with information about clients' movie preferences. They are recommender-driven, where the client's responses aid the recommender to assess a client "preference profile" in order to make qualified movie suggestions.

At an early stage in the dialogues a "recommendation base" is established. This is typically one of the first explicit attributes that the client puts forward, and on which a series of recommendations are based. It is common to return to the recommendation base throughout the dialogue, such as when wrapping up a sub-series of recommendations. On top of the original recommendation base several modifying attributes can be put for a few turns. The recommendation base can thus be modified; but it can also be changed completely. Changing the recommendation base requires some sort of explicit utterance from either of the dialogue partners. Four principal types of recommendation base changes or verifications are found in the corpus:

1. Changing the recommendation base when the client is done with the current one.
2. Changing or relaxing the recommendation base when there are no more options (i.e. all movies matching the current recommendation base have been considered).
3. Providing a new recommendation base suggestion when the client is uncertain or out of ideas.
4. Ensuring or verifying the attributes and importance of attributes in the current recommendation base depending on client feedback on recommendations (i.e. if the agreed recommendation base seems to generate several "bad" recommendations in the client's eyes).

An important implication for a recommender dialogue system design is thus to (a) allow users to change the recommendation base explicitly, as well as (b) let the system suggest recommendation base when the user does not suggest one herself, or when the current recommendation base has been exhausted.

Integration of Information and Preference Requests

As exemplified in Figure 2, information requests are often posed by clients in order to respond to recommenders' preference requests. Another important coupling between the two request types is that information request responses can

drive the recommendation dialogue forward, since the presented information triggers the user to provide new preference data or issue new information queries in the dialogue. Figure 3 shows an example of this. R1 is a preference re-

R1: I see / please name another good movie
C1: uhm / who's starring in *Ransom*
R2: here are all the actors in *Ransom* [shows the actor list of *Ransom*]
C2: so what other movies has *Mel Gibson* done?
R3: all of these [points at Gibson's filmography list]
C3: right / oh yeah / *Braveheart* is one of my absolute favorites
R4: oh then I think you'd like *Gladiator*

Figure 3: Dialogue excerpt showing how client-initiated information requests move the dialogue forward.

quest issued by the recommender. The sub-dialogue initiated by the client in C1 results in another list of movies from which the client can pick favorites. C3 can thus be viewed as a response to the overall goal introduced in R1. However, without the possibility to interleave information requests, the client would not have arrived at the list in R3, and the preference given in C3. The dialogue continuing from R4 is a direct result of the content of C3, which in turn is a result of the information requests in C1 and C2[2]. Handling information and preference requests and the integration of the two in this manner is thus an important issue for modeling the dialogue in a system.

HUMAN-MACHINE RECOMMENDATION DIALOGUES

In order to operationalize the dialogue in an implementation, we need to analyze the collected corpus in greater detail. The basis for this analysis is through the process of *dialogue distilling*, which is a method for analyzing dialogue corpora with a particular aim for dialogue system development [9]. When distilling, we aim at systematically re-writing the original human-human dialogue corpus into a plausible human-machine dialogue by applying a set of guidelines. In the movie recommendation corpus, we appoint the recommender participant to function as "the system", and the client as "the user". In the following, this is how they are referred to.

Recommendation Dialogue Control

With the completion of the distillation, we have an empirical ground to start developing the language resources (e.g. lexicons and grammars) necessary for a system implementation. We also consider the two initiative types from the previous section for the dialogue management components. The simplest form of dialogue management is finite-state machines, which guide the user through a series of pre-defined steps. This form of dialogue is completely system-driven in terms of initiative [1]. One way to visualize a distilled dialogue is to draw a finite-state network consisting of system utterances represented as nodes, and user utterances represented as arches. Each separate network has a unique linear sequence, since there is exactly one user utterance following a given system utterance in each of the dialogues.

As outlined above, recommendation dialogues in the corpus consists of a combination of (a) system-driven preference requests, and (b) user-driven information requests. The mixed-initiative character of the dialogue can roughly be said to correspond to a seamless integration of these initiative types. Based on this assumption, we turn to the network graphs of the distilled corpus and try to merge the system-driven initiative into one general recommendation dialogue network graph modeled as a finite-state machine covering the distilled corpus. This is done by comparing system utterances with focus on *preference requests*, and then incrementally adding arches corresponding to the various user utterances that occur as responses to the nodes. The resulting network is—not surprisingly—larger and more complex than the individual dialogue networks. The view of the complete dialogue graph in Figure 4 corresponds to system-driven preference requests.

Dialogue Node Functionality

This section describes the nodes in Figure 4 and their function in the dialogue.

Initiating the Recommendation Dialogue

The START node simply generates a welcome message. A user response indicating that she wants a movie recommendation leads to the RECBASE node. In Figure 4, there is only one arrow arching out from the START node, leading to the RECBASE node. This is a simplification, since users have the possibility to directly set the recommendation base from this node, thus skipping the RECBASE node. In RECBASE, we establish a "recommendation base", which is the principal attribute set that future recommendations will be based on. There are several possible responses to the RECBASE depending on what attribute the user prefers. Most users want to base their recommendations on genre (e.g. a drama, comedy, or action movie), whereas some users aim for movies starring their favorite actor (e.g. "I would like a movie starring *Cary Grant* please").

Getting Attribute Values

GETVALGENRE is responsible for trying to assess what genre(s) the user is interested in. The GETVALACTOR node functions in a similar way, asking the user for names of their favorite actors or actresses. The information retrieved by these two GETVAL nodes is integrated in the recommendation base.

Acquiring Title Ratings

A central issue when utilizing recommender engines is to acquire title ratings from the user [12]. The more titles that are included in the user preference model, the better recom-

[2] *Mel Gibson*, who is mentioned in C2, occurs on the *Ransom* actor list in R2. *Braveheart* occurs in the filmography list of *Mel Gibson* in R3.

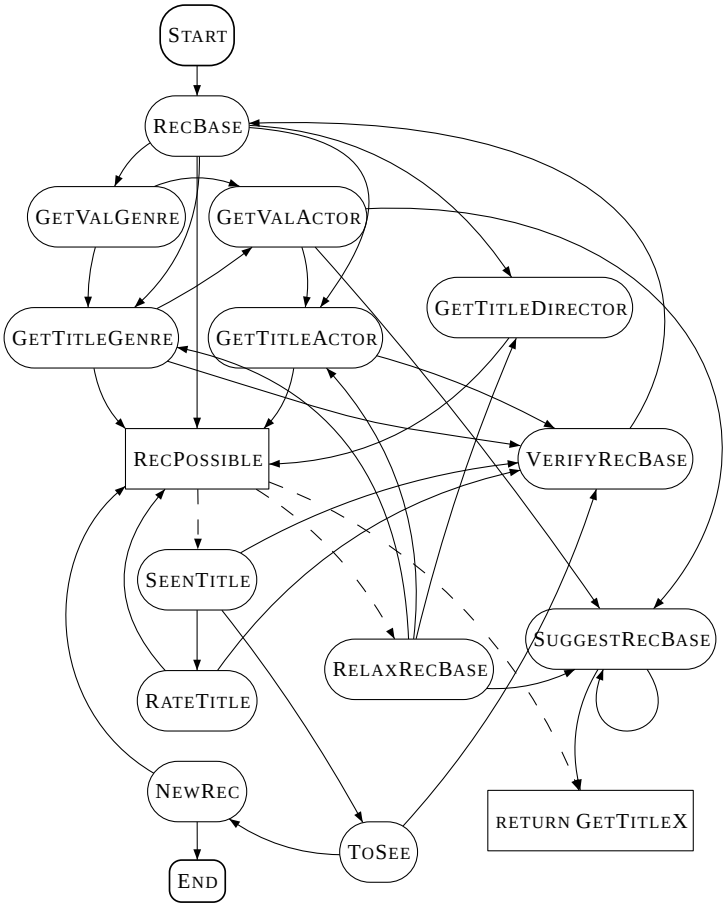


Figure 4: A recommendation dialogue network graph covering dialogue flows of the 24 distilled dialogues from the movie recommendation dialogue corpus. Some arches have been omitted for clarity. Rounded nodes correspond to a system utterance, whereas square nodes correspond to internal system actions not visible to the user. Solid arches correspond to interpreted user utterances.

mendations the engine can provide. Furthermore, the system needs some way of keeping track of which movies the user has seen, so the system does not recommend them again.

Thus, we have three GETTITLE nodes, each based on one of the attributes *genre*, *actor*, and *director*. The typical GETTITLE node usage is when the user has provided an attribute value (such as the name of an actor). The system then provides the user with a list of titles matching the given attribute values and asks her to identify movies that she likes. Note that this list is a non-personalized list and not a recommendation set. The GETTITLE nodes typically occur *before* any requests have been passed to the recommendation engine. Interleaved information requests can influence how the lists turn out (such as the excerpt in Figure 3). Thus, there is no hard connection between the GETTITLE node and the current recommendation base, since the titles in the list at any given moment do not need to reflect the recommendation base. This serves two purposes. First, we do not decrease the user's freedom of posing information requests, and indeed utilize these in the recommendation task. Second, it is good for the user preference profile to be as diverse as possible and not only include ratings for movies matching the current recommendation base.

RATETITLE comes into function *after* a recommendation has been proposed. Its function is to extract the rating of an already seen recommended movie, so that we constructively can utilize an otherwise "useless" recommendation, while maintaining a conversational tone in the interaction.

Performing Recommendations

SEENTITLE is one of the central nodes in the usage situation, since this is where the system presents a movie suggestion to the user. The corresponding system utterance for this node is "Have you seen this movie?" along with the title of the highest ranked recommendation. All nodes that have arches leading to SEENTITLE need to pass a check[3], since there are cases where it is not possible to traverse to SEENTITLE (i.e. perform a recommendation). This depends on the chosen recommendation engine. The SEENTITLE node is thus called only if the recommendation engine is able to deliver a suggestion. Otherwise, there is a need to continue to get ratings from the user (by returning to an appropriate GETTITLE node), or to change the current recommendation base.

Handling Changes

As pointed out above, the user may change the recommendation base. A change in the recommendation base can also arise from the system's part (e.g. to relax the constraints posed by the current recommendation base). The excerpt in Figure 5 shows an example of how the system suggests to change the recommendation base. In terms of network

S1: Have you seen *The Fifth Element*
U1: yeah / awesome
S2: It seems like we have covered all movies. Is there any other kind of movie you would like to watch?
U2: uhm / are there any movies directed by *Oliver Stone*?

Figure 5: Dialogue excerpt showing how MADFILM suggests a relaxation of the recommendation base when the matching titles have been exhausted.

traversing, S1 is an instantiation of the SEENTITLE node. The response in U1 is a positive rating of the recommended title, causing the system to return to the RECPOSSIBLE node to perform another suggestion based on the current recommendation base. Now, since all movies based on the current recommendation base have been considered, we traverse to the RELAXRECBASE node (S2). From this node there are several options, depending on the user's response. Since

[3]This check is represented as the square RECPOSSIBLE node in Figure 4.

the user provides a new recommendation base (recommendations should henceforth be based on the director in U2) the system moves to the GETTITLEDIRECTOR node according to Figure 4.

Managing Recommendation Dialogue

In case the suggested title in a SEENTITLE node is indeed unseen by the user, we have a potential recommendation. The system now needs to explain, or motivate, the recommendation objectively following the theory of building trust [2], and according to the findings in the dialogue corpus. This is done in the TOSEE node, which (a) generates an explanation by relating to the matching attributes in the current recommendation base, and (b) provides the user with the option of putting the recommended movie on the recommendation protocol. In case the user declines, the system needs to verify the current recommendation base, since this response is interpreted as negative feedback to the recommendation. On the other hand, if the user responds positively, we have a successful recommendation. The system can then add the recommended movie to the protocol and move on.

After a successful recommendation has been made the system asks if the user wants a new recommendation in the NEWREC node. A wide range of responses may follow this question. A simple "no" indicates that the session is terminated (moving the END node), whereas a simple "yes" is equally easy to handle, since we simply test if we can go to the SEENTITLE node to perform a new recommendation (after passing the RECPOSSIBLE check). However, the user may also change the recommendation base if she decides to continue the dialogue. It is easy to assume that this is because the users want variation in a set of recommendations in a session and desires e.g. one action movie, one drama comedy starring their favorite actor, and one animated movie. Examples responses to the question "Would you like a new recommendation?" include:

- "yes / something like *Gladiator* please"
- "a drama starring *Mel Gibson* would be nice"
- "do you have any animated movies?"
- "sure / give me movies directed by *Ridley Scott*"

In the case of a changed recommendation base, we traverse to the appropriate GETTITLE node (depending on which attribute(s) has been changed), in order to get a complete picture of any modifying attributes to the new recommendation base before moving on to a new SEENTITLE node.

Influencing Transitions

Several nodes in Figure 4 have multiple arches branching to different nodes. Three ways of influencing the network node transition are identified in the corpus: (a) user utterances, (b) the user preference model, and (c) database content and current recommendation base.

User Utterances

The default way to resolve transitions is to take the content of the user's response into account. This is done by having nodes check the interpreted utterance and decide which node to traverse to next. The content of the user utterance is thus the most important way to influence dialogue node transitions. However, while this is the default and most common transition influence, there are cases where the content of a user utterance may yield two (or more) equally valid system responses. We then need to consider other parameters.

User Preference Model

One alternative parameter is the user's movie preferences. This reflects that the recommender needs to know a number of preferences (ideally covering both positive and negative preferences about the bulk of all available attributes) before a qualified recommendation can be issued. It seems sound to assume that the recommender utilizes previously known preferences about movies, actors, and genres to dictate his or her next utterance.

In recommender system terms, this relates to the *density* and *size* of the user preference model [12]. Concretely, a standard collaborative filtering (CF) system is not able to calculate any prediction scores unless the user preference model has reached a certain density and size. Other types of recommendation engines have other constraints. Interfacing with the back-end system is the purpose of the RECPOSSIBLE node, and may be customized depending on the chosen engine. Thus, the size and content of the user preference model serves as an input to the dialogue nodes' transition decisions. In Figure 4, this is shown as the dashed arch from the RECPOSSIBLE node to the GETTITLE node.

Database Content and Exhausted Recommendation Base

The third transition influence is when the recommender realizes that the user's preferences takes the form of too demanding *constraints*. The recommender then asks the user to relax these constraints. This happens both when an information query from the user is too narrow, or when all movies matching the current recommendation base have been considered.

When there are no matching movies, the system must have ways to proceed if the user does not take initiative and starts introducing new preferences or search constraints. An exhausted recommendation base can thus be the reason for traversing to a RELAXRECBASE node instead of a new SEENTITLE node (see Figures 4 and 5).

IMPLEMENTATION

Hitherto we have focused on system-driven preference requests and recommendations. However, as noted above, a system implementation will also have to accommodate user-driven information requests. Fortunately, there is a fairly large body of research addressing exactly this issue. One such initiative is the phase-based PGP design pattern[4] that allows for information-providing dialogue system construction [6].

[4] PGP is hosted at the NLPFARM open source initiative (http://nlpfarm.sourceforge.net).

The dialogue strategy presented in this paper has been implemented in the MADFILM movie recommender system by adopting the PGP pattern and integrating the finite-state recommendation dialogue network with the information-providing capabilities [8]. Each node in the graph thus holds the same basic phase-based information-providing machinery, so that users can issue information requests at any time in the underlying system-driven dialogue, in line with the empirical findings in the corpus. Figures 1 and 5 exemplifies dialogue interaction in MADFILM.

MADFILM'S back-end part consists of a CF server[5] and a movie information database holding information on actors, genres, directors, and plot information. The database is used both to accommodate information requests, as well as providing attributes for the recommendation base. The recommendation engine is thus a hybrid engine [3], since it utilizes both the CF server as well as the domain-dependent database.

CONCLUSION

Acquiring user preferences in recommender systems is a non-trivial problem. Due to its flexible nature, natural language dialogue interaction is one promising approach. The work reported on here provides an empirically based model for implementing recommendation dialogue initiative. The recommendation initiative is modeled as a finite-state graph of nodes representing the system-driven preference dialogue. The arches are affected by three ways to influence transitions, namely (a) the meaning of the utterance itself, (b) the accumulated user preference knowledge, and (c) database content. User-driven information requests are modeled by re-using existing techniques from standard information-providing dialogue systems. Combining the two initiative types adds flexibility to the interaction and also functions a natural way to drive the dialogue forward, and should be utilized for unobtrusive preference acquisition. The proposed dialogue strategy has been implemented in the MADFILM recommender dialogue system. Future work includes evaluating MADFILM, as well as exploring the approach proposed here to a more general recommendation dialogue initiative strategy.

ACKNOWLEDGMENTS

This research has been supported by the Swedish National Graduate School of Language Technology (GSLT), Santa Anna IT Research, and VINNOVA.

REFERENCES

1. J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. S. Magazine. Towards conversational human-computer interaction. *AI Magazine*, 22(4):27–38, 2001.
2. A. Buczak, J. Zimmerman, and K. Kurapati. Personalization: Improving Ease-of-Use, Trust and Accuracy of a TV Show Recommender. In *Proceedings of the 2nd Workshop on Personalization in Future TV*, Malaga, Spain, 2002.
3. R. D. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12:331–370, 2002.
4. R. D. Burke, K. J. Hammond, and B. C. Young. The FindMe Approach to Assisted Browsing. *IEEE Expert*, 12(4):32–40, 1997.
5. G. Carenini, J. Smith, and D. Poole. Towards more conversational and collaborative recommender systems. In *Proceedings of the International Conference of Intelligent User Interfaces*, pages 12–18, Miami, Florida, USA, 2003.
6. L. Degerstedt and P. Johansson. Evolutionary Development of Phase-Based Dialogue Systems. In *Proceedings of the 8th Scandinavian Conference on Artificial Intelligence*, pages 59–67, Bergen, Norway, 2003.
7. P. Johansson. MadFilm: A Multimodal Approach to Handle Search and Organization in a Movie Recommender System. In *Proceedings of the 1st Nordic Symposium on Multimodal Communication*, pages 53–65, Copenhagen, Denmark, 2003.
8. P. Johansson. Design and development of recommender dialogue systems. Licentiate Thesis 1079, Linköping Studies in Science and Technology, Linköping University, April 2004.
9. A. Jönsson and N. Dahlbäck. Distilling dialogues - a method using natural dialogue corpora for dialogue system development. In *Proceedings of the 6th Applied Natural Language Processing Conference*, 2000.
10. G. Linden, S. Hanks, and N. Lesh. Interactive assessment of user preference models: The Automated Travel Assistant. In *Proceedings of the 6th conference on User Modeling*, pages 67–78, 1997.
11. M. F. McTear. Spoken dialogue technology: enabling the conversational user interface. *ACM Computing Surveys*, 34(1):90–169, Mar. 2002.
12. A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl. Getting to know you: learning new user preferences in recommender systems. In Y. Gil and D. B. Leake, editors, *Proceedings of the 2002 International Conference on Intelligent User Interfaces (IUI-02)*, pages 127–134, New York, 2002. ACM Press.
13. C. Thompson, M. Göker, and P. Langley. A personalized system for conversational recommendations. *Journal of Artificial Intelligence Research*, 21:393–428, 2004.

[5]The user rating matrix is provided by the GroupLens research group (http://www.grouplens.org).

Behavior-based Recommender Systems for Web Content

Tingshao Zhu, Russ Greiner
Dept. of Computing Science
University of Alberta
Canada T6G 2E8

Gerald Häubl
School of Business
University of Alberta
Canada T6G 2R6

Bob Price, Kevin Jewell
Dept. of Computing Science
University of Alberta
Canada T6G 2E8

ABSTRACT

In this paper, we demonstrate the implementation of an effective complete-web recommender system (i.e., *WebICLite*) using browsing behavior models to predict relevant Web pages. Behavior-based models use fine-grained information about the actions a user takes while browsing the web and the exact sequence of pages they follow to proactively provide responsive session-specific site-independent recommendations. The current paper also briefly presents browsing behavior-based models, and summarizes initial results from a large-scale field trial. The study suggests that the positive laboratory results for the original model transfer to real users browsing arbitrary web pages for day-to-day tasks.

Keywords

Browsing Behavior Model, Machine Learning, Information Search, Web Content Recommendations

1 Introduction

While the World Wide Web contains a vast quantity of information, it is often time consuming and difficult for web users to find the information they are seeking on the Web. Typically users will employ a search engine to find information. In order to benefit from these search engines, however, users must have intuitions about what keywords they should use to effectively discriminate the information they are seeking from the information they don't want from among the billions of Web pages that search engines typically index.

A number of researchers have proposed web-recommender systems that attempt to learn a user's information needs from observations of their past web-browsing behaviors. These recommenders use advanced information retrieval techniques to locate web resources that satisfy the user's needs. In this way, the user receives the information they need without having to reason about the best query to retrieve the information.

Zukerman [16] distinguishes two main classes of web-recommenders: *Content-based* systems use samples of past user behavior to learn what types of content appeal to a user. The system then recommends pages with similar content. *Collaborative filtering* systems uses samples of past user behavior to learn how the current user is similar to other users. The system then recommends pages to the user that have also been selected by similar users. Both Content-based and collaborative filtering systems have been well studied in the literature and their strengths and weaknesses are well understood. Content-based systems need a large sample of past user selections to establish user interests whereas collaborative filtering systems only work well when there is a sufficient pool of similar users. Both types of systems suffer from objective measures of validity as we cannot know if the user's past choices or the choices made by other similar users were really satisfactory and both types of systems tends to be site specific due to their need for information about the user's past browsing behavior.

In previous work [14], we have pointed out two opportunities for extending current web-recommender systems. First, we observed that a user's needs can change dramatically as the user plays different roles in life and works on various tasks and subtasks. A sensible recommender systems should recognize the differences between current interests and long term interests and makes its recommendations based on the user's current needs. We think of the searches for each distinct information need as occurring in distinct "sessions" and we call this concept *session specific recommendation*.

Second, we observed that recommenders can also help users by bringing relevant material to their attention even though they may not have thought to ask for it. We call this *proactive recommendation*.

We then proposed that we can use passive observations of the user's fine-grained web-browsing actions and the specific sequences of web-pages they were applied to to learn more about user's interests than is possible with static analysis of a bag of web pages visited by the user and with less input from the user than systems that require the user to label specific pages with their judgments. Since our analysis is based on the user's current dynamic actions it can be made session

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

specific. Since we can obtain this information unobtrusively, we can gather user's interests and make proactive recommendations. We say that we use the user's browsing behaviors to make proactive session-specific recommendations.

Our focus on the extraction of the user's information needs instead of the indexing of material has an additional benefit: namely, the approach is compatible with many existing methods for indexing material within a content-based framework. In particular, our system can turn inferences about user information needs into queries for standard search engines. This potentially allows us to recommend any web-resource indexed by major search engines to our users.

In this paper we report on a newly developed Web recommender system — *WebICLite*. Like most recommendation systems, *WebICLite* watches a user as s/he navigates through a sequence of pages, and suggests pages that (it predicts) will provide the relevant information. *WebICLite* differs from most other web recommendation system in several respects. First, while many recommendation systems are server-side and hence specific to a single web site [10, 1, 13], our client-side system is not so specific, and so can point users to pages *anywhere on the Web*. Secondly, *WebICLite* can predict the user's information need *dynamically*, based on the current context — that is, the current session. (This is based on patterns found over the "browsing properties" of the words appearing in the session; see Section 3.) The third difference deals with the goal of the recommendation system: our goal is to recommend only useful pages; i.e., pages that are relevant to the user's task. These "Information Content" pages (aka IC-pages) are just the pages the user needs to solve his/her current task, and not the overhead pages required to reach them. This differs from systems that instead attempt to lead the user to pages that similar users have visited earlier (independent of whether those "familiar" pages in fact contain the answer to the user's current quest). Finally, *WebICLite* is "passive", in that it can recommend pages relevant to the user's *current* information need without requiring the user to do any additional work — e.g., the user does not need to answer intrusive questions, etc.

Section 2 discusses related work. Section 3 then describe a simple procedure for training our models, and the results of a user study (i.e., LILAC) that demonstrates the ability of our model to predict pages useful to the current user, from anywhere on the Web. Section 4 describes an implementation of our ideas in the form of a stand-alone web browser, *WebICLite*, that runs on the user's computer, and provides on-line recommendations, to pages anywhere on the Web, not just on the user's current Web site. Finally, Section 5 concludes with a summary of our key contributions and insights.

2 Related Work

Many groups have built various types of systems that make recommendations to users. One can get a sense of the breadth of work in this area from the table below which summarizes a number of common approaches and representatives systems within these approaches:

- COB: Co-occurrence Based — e.g., Association Rule [2], Sequential Pattern [3], etc.
- CF: Collaborative Filtering [12]
- CB: Content-Based [5, 8, 4]
- HBM: Heuristic-Based Model [11, 9, 6]
- IC-Models: IC-based Models; see Section 3.

We find it useful to compare these systems on a number of parameters:

- All systems require a model of the user's interests, but some learn the model and some do not.
- Some systems require a training phase in which users distinguish content they desire from content they do not.
- Systems vary in the extent to which they can use information learned from specific users (individual) and groups of users (Group or Population).
- Systems vary according to how they validate the recommendations they make. Some use indirect information contained in correlations whereas others use explicit direct judgments of content.
- Some systems take the sequence of pages into account, and some do not.

A table comparing our representative systems on these dimensions appears in Table 1. Due to space restrictions, we have had to abbreviate our discussion of recommender systems, but we invite the reader to consult the references to follow up on the details of these approaches.

3 Session Specifc Information Needs Model

Like other researchers, we have chosen to conceptualize web browsing as a search for content satisfying a specific, well-defined "information need". Like many systems, we observe choices made by users while browsing. In our model, however, we are interested in the user's session specific information need and we use the user's individual fine-grained browsing actions and the exact sequence of pages visited by the user to find out what it is. First, we give some general background on our browsing-behavior based approach to recommendation and then we briefly describe several specific algorithms.

3.1 Browsing Behavior Models

Consider the example suggested by Figure 1. Imagine the user, needing information about marine animals, sees a page with links on "Dolphins" and "Whales". Clicking on the "Dolphins" link takes the user to a page about the NFL Football team. As this page does not contain the information that this user is seeking (at least, not at this time), the user "backs up". As the word "Football" appears on the previous page but not on the current page, this "backing-up" behavior suggests that "Football" might not be relevant to his/her search

Dimensions	COB	CF	CB	HBM	Our Models
Specific Site/Domain	Yes	Yes	Yes	No	No
Model Acquisition	Learning	Learning	Learning	Hand-coded	Learning
Annotation Required (training)	No	No	Yes	No	Yes
Annotation Required (performance)	No	No	No	No	No
Reference Set	Population	Group	Individual	None	Individual/Group/Population
Recommendation Validity	Indirect	N/A	Direct	Direct	Direct
Using Sequential Information	No	No	No	Yes	Yes

Table 1: Techniques for Recommender System

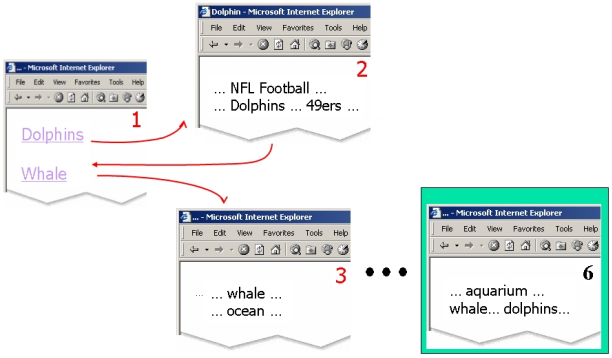


Figure 1: Browsing Behavior of Words, within Session

task. The user then tries the "Whale" pointer, which links to a page whose title includes "whale", and which includes whales, oceans, and other marine terms in its content. The user then follows a link on that page, anchored with "whale" to another with similar content, and so forth, until terminating on a page about a local aquarium. This behavior suggests that words such as "Whale" and "Ocean" are relevant, but "football" is not.

The above observation suggest that the user's current information need can be identified from the pages the user visits and the actions that the user applies to the pages. The content of the page is communicated by the roles that words play on the page. We make the simplifying assumption that we can represent the information need of the session by a set of significant words from the session. We further assume that the significance of words can be judged independently for each word from the roles played by instances of the word on pages throughout the sequence (e.g., appearing as plain text, highlighted in a title, appearing in followed hyperlink, etc.). To capture the reaction of the Web user, we recorded a number of browsing features from the page sequence. Our papers [14, 15] provide a complete list of these "browsing features".

To obtain the user model for Web recommendation, we first collected a set of annotated web logs (where the user has indicated which pages are IC-pages), from which our learning algorithm learned to characterize the IC-page associated with the pages in any partial subsession [14, 15].

3.2 Specific Models

Within the framework described above, we have implemented a number of specific algorithms for identifying information-need-revealing patterns. These patterns can be used to form queries to a search engine which does the actual retrieval of recommended pages. All of the algorithms require data labeled by human subjects.

IC-word: is our original algorithm. It requires subjects to explicitly identify pages with useful content. In this algorithm, we attempt to predict if a word that occurs in pages during the user's browsing sequence will appear on the information content page identified by the user. This prediction is done for each word independently and is based on only the browsing features of words in the sequence; that is, their pattern of occurrences within pages in the sequence and the actions applied to the pages they appear on. Any word with the same features will get the same score.

IC-Relevant: is a new algorithm developed for the current study. It requires subjects to explicitly indicate words that were relevant to their need. In this algorithm, we attempt to predict if a word that occurs in the pages of the user's browsing sequence will be in a set of words the user explicitly marks as relevant keywords.

IC-Query: is our most sophisticated algorithm and was also newly developed for this study. It is based on the observation that many of the words occurring on IC-pages are general (e.g., "the", "page", etc.) and therefore not particularly relevant to the page content. In particular, few of these words would help locate this page, in a search engine. We also observe that the words used in a search query are not independent. The specific combination of words and the order they appear are significant. The goal of the IC-Query algorithm is to find the 4-word search query that would most likely return the IC-pages identified by the user. Empirical investigations has revealed that 4-word queries are quite effective. The precise details of training the IC-Querymodel are complex and will be the subject of a future paper.

3.3 Experimental Design of the LILAC Study

Earlier laboratory studies revealed significant potential for behavior based methods [14, 15] and were an important motivation for the current work. The current study, a large field experiment code-named "LILAC" (Learn from the Internet:

How does your page compare to the recommended page?

- Fully answered my question
- Relevant, but does not answer my question fully
- Interesting, but not so relevant
- Remotely related, but still in left field
- Not related at all

Figure 2: Evaluation dialog options

Log, Annotation, Content), was intended to gather training data for creating future recommendation models and to evaluate the quality of recommendation models on a wide sample of users working on realistic, unconstrained tasks, seeking information from arbitrary sites on the Web.

LILAC was scheduled to last 5 weeks and involved 104 participants who installed a modified version of the internet-explorer web-browser, *WebIC*, on their home or business computers. Users were encouraged to disable tracking during personal or confidential browsing and were given the option of declining to submit web logs. User's were paid an honorarium for both their time and the number of sessions they generated.

The experimental design had 4 cases based on the model used to provide recommendations to the user. The four models were Followed Hyperlink Word (FHW) [1], a sensible recommendation strategy taken from the literature, our original behavior based model IC-word and our two newer methods IC-Relevant and IC-Query described above. Each of these models produced a set of words which we then sent to the Google^{TM} search engine. Our recommendation consisted of the first results page returned by Google.

The joint goals of obtaining training data and evaluating models lead to a slightly complex experimental protocol: Subjects were asked to browse normally, but explicitly mark information content pages by pressing a button on the browser tool bar. This step provides us with a complete sample of the user's behavior: a browsing sequence and the resulting information content page. This data can be used for training future models.

At this point, the recommender generates a web-page recommendation using a randomly chosen recommendation model. The subject is then asked to compare the usefulness of the page they marked as having satisfying information content to the page generated by the recommender model (See Figure 2). This second step allows us to assess the ability of the chosen recommender model relative to the user's own standard of quality and to assess the value added to the user's existing information search efforts.

In the case where subjects could not find a page that addresses his/her information need, subjects were instructed

[1] FHW is based on a simple rule: rank words according to how often they appear in the anchor text of followed hyperlinks. This is essentially the core of the model used in "Inferring User Need by Information Scent" (IUNIS) model [7].

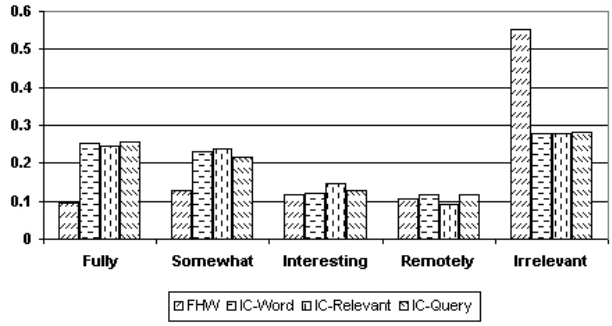


Figure 3: Overall Results of the LILAC Study

to click on the "Suggest" button. *WebIC* presents a recommended page for review. Subjects were then asked to provide an absolute subjective rating of the usefulness of the suggested page with respect to his/her current information needs. Users were also asked to rate subsets of words appearing in the sessions according to how relevant they felt they were to their current needs.

Data obtained from earlier weeks in the study were used to train improved behavior models for the current week of the study.

The 104 subjects visited 93,443 web pages, marked 2977 pages as IC-pages and asked for recommendations by clicking the "Suggest" button 2531 times over the course of the 5-week LILAC study. Summary statistics for the comparative ranking used for IC-pages appear in Figure 3. The bars show the relative percentage of each of the evaluation responses for each model. The best models would be expected to have more "Fully" answered ratings and fewer "Irrelevant" ratings.

As suggested by this figure, and confirmed by statistical tests (shown in "http://www.web-ic.com/lilac/results.html"), each of the different IC-models perform better than the baseline model (FHW). This result supports our basic assumption that we are able to provide useful recommendations by integrating the user's browsing behaviors into the prediction. Further analysis of the results will appear in a future paper.

4 *WebICLite*— An Effective Complete-Web Recommender System

The *WebIC* system that we used in the LILAC study has evolved into the *WebICLite* recommendation system, whose interface appears in Figure 4. *WebICLite* is also a client-side, *Internet Explorer*-based multi-tab web browser, that observes the user's browsing behavior, extracts the browsing properties of the words encountered, and then uses those browsing properties to predict the user's current information need, which it then uses to suggest (hopefully) useful pages from anywhere on the Web, without any explicit input from the user. It first gathers browsing properties for essentially all of the words that appear in any of the observed pages in the current session, then uses a model of user browsing patterns, obtained from previously annotated web logs, to gen-

Figure 4: *WebICLite* — An Effective Complete-Web Recommender System

erates an appropriate query to a search engine (here Google), which produces a candidate page to present to the user.

4.1 Hybrid Recommender models

Data from the LILAC study suggests that people tend to surf the Web by following some general browsing session patterns. One example of a search pattern looks like:

Query a search engine (Q)
Obtain a search results page, P
Open one URL from P
Return to P
Obtain another URL from P
Return to P
...

This pattern suggests that this query to this search engine is not producing the relevant page.

We found that some models work better for certain browsing patterns, as determined by general characteristics of the current session. For example, our evidence shows that IC-Relevant works better than any other models for the above pattern. *WebICLite* therefore includes a set of rules to choose the model that works best for the current browsing session.

4.2 Ongoing Evaluation

In the current implementation, the user can click "Suggest" to ask *WebICLite* to propose a Web page, anytime s/he needs assistance.

In order to collect the relevance feedback, which our system can use to improve its performance, *WebICLite* will then ask the user to evaluate the suggested page, using the interface shown in Figure 5. Here, the user is asked to "Tell us what you feel about the suggested page", to indicate whether the information provided on the page suggested by *WebICLite* was relevant for his/her search task, just as the user did in LILAC. Note that this is optional, if the user could provide such feedback, we can train his/her personalized models.

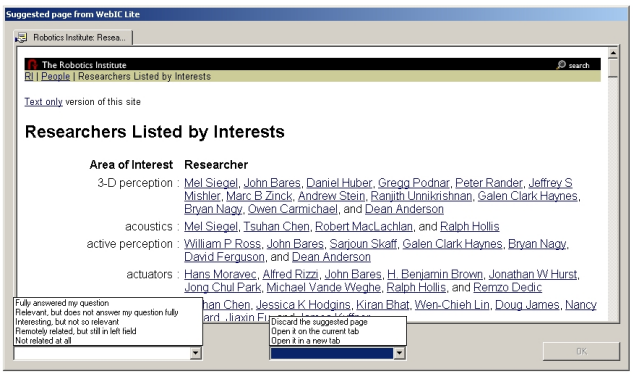


Figure 5: The Evaluation Interface of *WebICLite*

WebICLite also provides the options that allow the user to keep surfing from the suggested page P, by asking the user if s/he wants to *Discard P*, *Open P on the current tab*, or *Open P in a new tab*.

4.3 Learning from Evaluation

In order to train our models in LILAC, the study participants must actively label IC-pages while browsing the Web; of course, this is inconvenient for the user, and unrealistic in a production version of the product. To solve this data collection problem, we propose to passively train a model based on previous evaluation results. Recall that every time a user requests a recommendation, we generate a search query using one of the models to return a page to the user, which s/he is then asked to evaluate. If we assume that the search engine (e.g., Google) remains relatively consistent (i.e., in terms of the quality of pages returned) over time, we can infer the evaluation of the search query from the actual evaluation of the recommended page. Thus we can label each query as one of the evaluation outcomes. We can then attempt to learn a "Fully"-page classifier by considering (as positive examples) only the queries that are evaluated as "Fully", and the rest as negative.

In the LILAC study, the IC-Query models were trained directly based on the pages marked IC-pages. In the last week, we changed the experimental protocol to train a model based on all queries that resulted in a "Fully" evaluation in the previous weeks. Figure 6 presents the results of these two models, trained by the pages marked IC-page, vs the retrospective one.

As suggested here, both approaches produce similar performance. This result is significant as it will allow us to continuously refine the model without requiring user input to label IC-pages while browsing the Internet. Importantly, this alternate training method will make the use of *WebICLite* more realistic in real world situations.

5 Conclusion

In this paper we introduced two new behavior-based models. While we are still in the process of analyzing our results, we

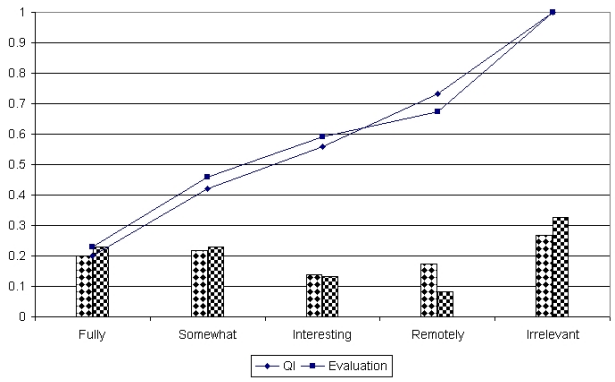


Figure 6: Alternate Training on Evaluation

have some evidence that these new models outperform both the control model (FHW) and our existing model. Our current study has shown that the positive potential of behavior-based recommendation models seen in our laboratory studies can be transferred to real users browsing arbitrary web pages during day-to-day tasks. In the couse of this study, we have collected a large amount of high quality data and expect to train significantly better models in the near future. While still preliminary, we believe our results support the conclusion that behavior-based models have a unique ability to provide responsive session-specific recommendations independent of any particular site and that these models have a promising range of useful future extensions.

Acknowledgement

The authors gratefully acknowledges the generous support from Canada's Natural Science and Engineering Research Council, the Alberta Ingenuity Centre for Machine Learning (http://www.aicml.ca), and the Social Sciences and Humanities Research Council of Canada Initiative on the New Economy Research Alliances Program (SSHRC 538-2002-1013).

REFERENCES

1. F. Abbattista, M. Degemmis, N. Fanizzi, O. Licchelli, P. Lops, G. Semeraro, and F. Zambetta. Learning user profiles for content-based filtering in e-commerce. In *Proceedings AI*AI Workshop su Apprendimento Automatico: Metodi e Applicazioni*, Sienna, Italy, 2002.
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th Int'l Conference on Very Large Databases (VLDB'94)*, Santiago, Chile, Sep 1994.
3. R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. of the Int'l Conference on Data Engineering (ICDE)*, Taipei, Taiwan, Mar 1995.
4. C. Anderson and E. Horvitz. Web montage: A dynamic personalized start page. In *Proceedings of the 11th World Wide Web Conference (WWW 2002)*, Hawaii, USA, 2002.
5. D. Billsus and M. Pazzani. A hybrid user model for news story classification. In *Proceedings of the Seventh International Conference on User Modeling (UM '99)*, Banff, Canada, 1999.
6. J. Budzik and K. Hammond. Watson: Anticipating and contextualizing information needs. In *Proceedings of 62nd Annual Meeting of the American Society for Information Science*, Medford, NJ, 1999.
7. E. Chi, P. Pirolli, K. Chen, and J. Pitkow. Using information scent to model user information needs and actions on the web. In *ACM CHI 2001 Conference on Human Factors in Computing Systems*, pages 490–497, Seattle WA, 2001.
8. A. Jennings and H. Higuchi. A user model neural network for a personal news service. *User Modeling and User-Adapted Interaction*, 3(1):1–25, 1993.
9. H. Lieberman. Letizia: An agent that assists web browsing. In *International Joint Conference on Artificial Intelligence*, Montreal, Canada, Aug 1995.
10. D. S. W. Ngu and X. Wu. Sitehelper:a localized agent that helps incremental exploration of the world wide web. In *6th International World Wide Web Conference*, pages 691–700, Santa Clara, CA, 1997.
11. P. Pirolli and W. Fu. Snif-act: A model of information foraging on the world wide web. In *Ninth International Conference on User Modeling*, Johnstown, PA, 2003.
12. P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina, 1994. ACM.
13. A. Stefani and C. Strapparava. Personalizing access to web sites: The siteif project. In *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98*, Pittsburgh, USA, June 1999.
14. T. Zhu, R. Greiner, and G. Häubl. An effective complete-web recommender system. In *The Twelfth International World Wide Web Conference(WWW2003)*, Budapest, HUNGARY, May 2003.
15. T. Zhu, R. Greiner, and G. Häubl. Learning a model of a web user's interests. In *The 9th International Conference on User Modeling(UM2003)*, Johnstown, USA, June 2003.
16. I. Zukerman and D. Albrecht. Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction*, 11(1-2):5–18, 2001.

Who do trust? Combining Recommender Systems and Social Networking for Better Advice

Philip Bonhard
Department of Computer Science
University College London
Gower Street, London WC1E 6BT
+44 20 7679 3642
p.bonhard@cs.ucl.ac.uk

ABSTRACT
Faced with overwhelming choice people seek advice from their peers or other trusted sources. Collaborative filter recommender systems aim to emulate this process by filtering all the options according to the user tastes expressed through prior item evaluations. Until now the recommender systems literature predominantly focused on improving the algorithms for making suitable predictions for unrated items, while usability research mainly concentrated on interface issues with existing applications. This approach ignored the social elements of decision-making and advice seeking. The research here aims to consider a broader range of factors that motivate people in their decision making in order to improve recommender systems. Qualitative research conducted to date has shown that the relationship between recommender and recommendee has a significant impact on decision-making. Thus it is proposed that the impact of social elements on the quality of recommendations needs to be considered for the design of effective recommender systems, which can be addressed through the integration of social networking.

Keywords
Recommender Systems, Social Networking, Decision Making, Advice Seeking, Human Centered Design

INTRODUCTION
When faced with overwhelming choice and lacking specific domain knowledge, many people seek advice from peers and other trusted sources. Recommender systems emulate this process by drawing on user preferences and filtering the set of possible options to a more manageable subset.

Collaborative filter algorithms have emerged as one of the dominant strategies for computing recommendations mainly because they are not item domain bound. User preferences are expressed as item ratings and recommendations are based on matching users with similar ratings, assuming that high correlation in ratings among users is an indicator of taste overlap.

Two major technical problems with this approach are 1) the *cold start problem* and 2) the *sparsity problem*. The former refers to the fact that the system cannot compute any recommendations for a new user because it has no information about his preferences. The latter is about the fact that the number of people who have rated particular items in the database might be relatively small compared to the total number of items. This means that there might not be significant similarity among users leading to possibly lower quality recommendations as they are based on little data.

HCI Recommender Systems Research to Date
To date, Human Computer Interaction (HCI) examination of recommender systems has mainly focused on interface issues with existing applications [2, 4, 7]. One of the key issues addressed here is how the user's mental model of the system does not match the system model. Thus the aim is to manipulate the interface in such a way so as to make the system functionality transparent. This aims to generate trust in the system rather than applying knowledge about human decision making processes to the system design and thus supporting existing advice seeking strategies. Thus a revised HCI perspective should take a step back and consider human advice seeking and decision making strategies and incorporate them into recommender system design.

Social Embedding of Recommendations
Recommendations are not made in rational isolation, which means that they are not evaluated merely by their information value. Rather they are delivered within an informal community of users and a social context [5]. The *social embedding* of a recommendation is crucial to understanding the decision making process of an individual; it is determined by factors such as experience, background, knowledge level, beliefs and personal preferences [3].

Sinha and Swearingen [6] found that when comparing recommendations from friends with collaborative filter recommender systems (movies or books), in terms of quality and usefulness, friends' recommendations are

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

preferred. Friends are seen as more qualified to make good and useful recommendations compared to recommender systems mainly because they are assumed to know more about the recommendee.

The psychology literature has examined advice seeking and developed theories for factual, objective domains, whereas advice seeking in subjective domains of taste, which the majority of recommender systems are designed for, have not yet sufficiently been explored [8].

Therefore there is a need for closer examination of how people actually seek advice, consider recommendations and make decisions in taste domains (like books, CDs, restaurants etc.) in real life and how that can be applied to the recommender systems design.

Research Conducted & In Progress

The research here has two fundamental elements. The first is concerned with gaining a better understanding of how people seek advice and consider recommendations, whereas the second aims to apply those findings in a recommender system test bed permitting in vivo experiments and user studies. The overall goal is to investigate the effects of the quality of recommendations, the affective elements in decision-making and how these can be incorporated into useful recommender systems. Further the social elements in user matching and recommendation generation through collaborative filtering suggest interesting research potential for the integration of recommender system and social networking functionality.

In the first phase, a series of 12 one-on-one interviews and five focus groups (32 participants) were conducted with the aim of examining people's strategies for advice and recommendations seeking and decision making. The preliminary results indicate that the following concepts are crucial:

- The relationship between advice seeker and giver is a key indicator for taste overlap and mutual knowledge
- Decision makers differentiate between objective (factual & specification driven) and subjective domains (taste) and apply different advice seeking strategies for each
- Past experience with the recommender impacts on trust
- Aggregation of user opinions used as a popularity indicator
- Ulterior motives of a recommender are perceived to have a negative effect the quality of the advice given
- Decision making transfer & sharing of responsibility as a motivator for seeking advice

For the second phase of the research a testbed is currently being developed that will integrate a recommender system for restaurants with social networking functionalities thus taking a user connection centric approach. Similar to the GroupLens' movielens project [1] this testbed seeks to recruit a broad base of heterogeneous users. The goal is to study the effect of different information and social elements on decision making and system usage in the short, medium and long run.

Questions for Recommender System Research

Thus there are various interesting questions that beg further consideration and exploration by the recommender systems research community:

In how far can recommender systems and social networking applications (like Friendster or Orkut) benefit from each other in order to increase consistent user participation and contribution?

Can the introduction of social networking aspects such as existing networks of friends (or recommenders) increase trust in recommender systems?

Can user matching via collaborative filtering encourage communication among like-minded users?

What kind of metrics can collaborative filtering algorithms contribute not only to the computation of recommendations, but also effective user matching?

Is it possible to alleviate the cold start problem through explicit specification of one's closest neighbors?

Considering the above questions among others should lead to a greater understanding of its target users and thus contribute to more effective recommender system design.

References

[1] http://movielens.umn.edu

[2] Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (2000). Explaining collaborative filtering recommendations. In *Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work* (pp. 241-250)

[3] Lueg, C. (1997). Social Filtering and Social Reality. In *Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering*

[4] McNee, S. M., Lam, K. S., Guetzlaff, C., Konstan, J. A., & Riedl, J. (2003). Confidence Displays and Training in Recommender Systems. In *Proceedings of INTERACT '03 IFIP TC13 International Conference on Human-Computer Interaction* (pp. 176-183)

[5] Perugini, S., Goncalves, M. A., & Fox, E. A. (2004). A connection centric survey of recommender system research. *Journal of Intelligent Information Systems* 23[1].

[6] Sinha, R. & Swearingen, K. (2001). Comparing Recommendations made by Online Systems and Friends. In *Proceedings of the DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries*

[7] Swearingen, K. & Sinha, R. (2002). Interaction design for recommender systems. In *Interactive Systems (DIS2002), London, June 25--28 2002*

[8] Yaniv, I. (2004). The Benefit of Additional Opinions. *American Psychological Society, 13*, 75-78.

Recommender Systems Research at Yahoo! Research Labs

Dennis Decoste*, David Gleich, Tejaswi Kasturi*, Sathiya Keerthi*, Omid Madani*, Seung-Taek Park*, David M. Pennock*, Corey Porter*, Sumit Sanghai**, Farial Shahnaz**, Leonid Zhukov***

*Yahoo! Research Labs,
{decosted|tejaswi|keerthis|madani|parkst|pennockd|porterc|zhukovl}@yahoo-inc.com
**Work conducted at Y! RL, dgleich@stanford.edu, sanghai@cs.washington.edu, shahnaz@cs.utk.edu

ABSTRACT

We describe some of the ongoing projects at Yahoo! Research Labs that involve recommender systems. We discuss recommender systems related problems and solutions relevant to Yahoo!'s business.

INTRODUCTION

A number of projects at Yahoo! Research Labs[1] involve collaborative filtering, recommendation, and personalization. A number of business units within Yahoo! either currently use recommendation technology with success, or plan to implement recommendations and personalization in the future. This paper briefly describes some of the relevant projects ongoing at Yahoo! Research Labs.

MAD6

MAD6 (**M**ovies **A**ctors **D**irectors; **6** degrees of separation) is a prototype movie search engine with five major design goals: (1) leveraging relational data implicit in the graph of movies and people (2) extensive metadata indexing, (3) leveraging user ratings and activity logs for personalization and recommendation (4) pseudo natural language query support ("shortcuts on steroids"), and (5) intelligent search ranking based on a combination of popularity, relational inference, and personalization.

We have a working prototype that supports extensive indexing beyond title/name matching (so, for example, queries like "arnold action" and "neo trinity" return meaningful results), intelligent ranking based on popularity and ratings, browsing by movie graph relations, related movie information by overlapping casts and by overlapping user interest, within-genre recommendations and global recommendations. We are currently experimenting with a number of machine learning algorithms to support recommendations, including cold-start recommendations (85% of movies do not have any explicit ratings). We plan to implement other personalizations like allowing a user to search their past activity, to examine plot words, genres, actors, etc. that he/she tends to visit, and to view a "prototypical" movie based on the user's browsing behavior. Finally, we plan to develop a pseudo natural language query interface to MAD6.

[1] `http://research.yahoo.com`

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

GROUP RECOMMENDATIONS

Group collaborative filtering is the use of individual ratings of various items to form a consensus recommendation for an entire group of people. This can be useful when choosing a form of entertainment for a set of people, such as a movie for a group of friends, or a restaurant for a family. Group recommendations can come through various algorithms, including collapsing a group into a single fake user and using traditional collaborative filtering methods, using a "least misery" method wherein the worst-off person in the group still has acceptable results, or using various voting methods to "elect" an appropriate best choice. We are beginning by tackling the problem of recommending music to a group of friends, building the functionality on top of the open source **Co**llaborative **F**iltering **E**ngine (CoFE) developed at Oregon State University.[2] As in any such work with a subjective output, the difficulty is in finding the best method with which to evaluate the results without requiring a large number of users over a long time period.

ALGORITHMS

Dimensionality reduction

We have employed singular value decomposition (SVD) / latent semantic indexing (LSI) to provide recommended keywords for Overture advertisers to bid on, based on keywords they and others are currently bidding on. We have also employed SVD/LSI for more standard recommendation problems in the movies and music domains.

[2] `http://eecs.oregonstate.edu/iis/CoFE/`

Learning relative/ordinal rankings

Many machine learning methods for recommenders focus on learning numeric ratings. Yet, users are often more comfortable/confident articulating their preferences as relatives (e.g. "like X more than Y") than absolutes (e.g. "like X at level 5 and Y at level 4"). The traditional problem with learning pairwise ranking functions is that this can involve training times that scale quadratically with the number of votes. To overcome this problem, we have been developing fast new methods that, under certain sets of practical situations (such as linear models) scale only log-linearly.

Content vs. collaborative filtering

In many instances we have extensive metadata about items, and sometimes demographic data about users. We are exploring a variety of machine learning algorithms to mimic collaborative similarities using content data, which can then be applied to sparse regions of the data.

Active learning and the cold-start problem

One key problem is what to do with new users (often a large fraction of users) and new items. We are exploring active learning techniques to determine which questions to ask users. For example, in the movie domain, we can ask for ratings on movies, actors, directors, genres, awards, etc. We can ask for numeric ratings or comparisons. We can ask more generic personality questions ("do you cry at movies?", "which among this set of abstract images[3] appeals to you?"). Choosing the right question involves determining informativeness, ability to answer, and willingness to take the time to answer (user burden). Determining the best questions is ultimately an exploration/exploitation tradeoff, since we won't know a question's informativeness until we receive sufficient answers on which to base inferences.

Music recommendation via audio similarity

With the recent explosion of availability of MP3 and other music data sources, significant recent research activity has focussed on new methods for summarizing music audio data and defining suitable similarity measures to supplement traditional metadata and ratings data. Often the somewhat hard-to-predefine audio nature of a song (i.e. combination of beat style, existence of unique guitar riffs, overall audio "feel", etc.) has more impact on whether someone likes the song than on what genre or artist is associated with it. If asked, most everyone seems to think that their own musical tastes are "eclectic"—hard to characterize as some simple cluster in artist/genre space and not necessarily particularly similar any other user. Exploiting such audio content presents huge new machine learning challenges, due to the significantly increased raw dimensionality (i.e. millions of bits of raw audio data per song) of the content data, and determining similarity metrics that correspond as closely as possible to human perception.

[3] `http://www.cs.ucr.edu/~chua/`

INTERFACES

We have built a prototype "World of Music" searchable map, which is a low-dimensional projection of music artists proximity for visual display of artist-artist similarity. We have also built a Java applet that implements a simple spring-force-based layout for exploring the space of recommendations and related items in the movie and music domains. We plan to use MAD6 as a platform for testing various search, browsing, personalization, and recommendation interfaces in the movie domain. Some of these demos may become available in the future via the lab website, `http://research.yahoo.com`.

EVALUATION METRICS

We plan to explore the utility of several offline and online metrics, with the goal of determining which offline metrics best predict important online metrics. As a company with a large user base, it is possible to try beta algorithms on small percentages of traffic and still obtain meaningful statistics. Moreover, as a largely advertiser-funded media company, Yahoo! can mainly focus on satisfying users to encourage retention, without need to consider inventory for example.

YAHOO! BUSINESS APPLICATIONS

A number of Yahoo! properties and business units use recommendation technology, or are planning or considering using recommendation technology, including Launch Music on Yahoo!, MusicMatch, Yahoo! Movies, Yahoo! TV, Yahoo! Personals, Yahoo! Local, Yahoo! Autos, Yahoo! Search, targeted banner advertising, Overture sponsored search advertising, and Overture contextual advertising. A company-wide effort to offer packaged recommendation technology to any interested Yahoo! property is underway.

ACADEMIC COLLABORATION

We try to maintain close ties to the academic community by staying current on the latest research, publishing our own research results, attending and sponsoring relevant workshops and conferences, hiring graduate student interns, hosting faculty sabbaticals, and hosting *spot workshops* on site. For example, in August 2004, we hosted a spot workshop on recommender systems featuring both external academic speakers and internal Yahoo! speakers from various business units.[4]

We have had success using and building on Oregon State's CoFE software. We have been able to share data on a case by case basis with academic collaborators, and would like to expand our data sharing activities.

ACKNOWLEDGMENTS

We thank Todd Beaupre, Donna Boyer, Nilesh Gohel, and Steve Souders.

[4] `http://research.yahoo.com/~pennockd/spot/rs/`

A Multi-agent Smart User Model for Cross-domain Recommender Systems

Gustavo González, Beatriz López, Josep Lluís de la Rosa
Institut d'Informàtica i Aplicacions. Agents Research Lab. Universitat de Girona
Campus Montilivi, Edifici P-4. E-17071, Girona, Spain.
{gustavog, blopez, peplluis}@eia.udg.es

ABSTRACT
This paper describes our approach to the next generation of open, distributed and heterogeneous recommender systems using *Smart User Models (SUM)*. Our work focuses on integrating multiple agent-based services based on a unique representation of the user in what is called a *Multi-agent Smart User Model*. Intelligent agents are used in order to obtain a single user model instead of having several versions of the same user spread throughout various services. A methodology has been developed using incremental aggregation of information, which favors non-intrusive behavior of the user model in order to determine objective, subjective and emotional user features.

Keywords
User modeling, cross-domain recommender systems, incremental learning, Smart User Models.

INTRODUCTION
The development of smart adaptive systems [1] is a cornerstone for personalizing services for the next generation of open, distributed and heterogeneous recommender systems. Agent Technology has contributed to the integration of services [11], but this integration has mainly been performed from a service point of view and is not usually centered on the user. Over the last years, our research group has been working with distributed services on the Internet using Agent Technology [9]. Currently we are dealing with challenges concerning: 1) the development of a unique, reusable and adaptive user model regarding objective, subjective and emotional user features ; 2) mapping user preferences from specific applications in several domains to this unique user model.

The next generation of recommender systems will have a moderately portable user model, which will interact with services in several open, distributed and heterogeneous environments to communicate user preferences in several domains. This requires the definition of the *Smart User Model (SUM)* and the corresponding infrastructure to integrate the user information across several services.

This paper is organized as follows: First, we define the *SUM* components. Second, we describe the mechanism for incremental aggregation of information in the *SUM*. Third, we explain the multi-agent framework in which it operates. The paper concludes with some contributions and plans for further research.

SMART USER MODEL
We have carried out work based on creating an adaptive user model [8] that should be able to pick up any type of objective, subjective or emotional user features (explicit or implicit). For this purpose, [4] defines the following *SUM* as the collection of attribute-value pairs that characterize the user. Where the collection of attribute-value pairs represents objective *(O)*, subjective *(S)* and emotional *(E)* features of the user. These sorts of features form three components in the user model: U^O, U^S and U^E. To summarize:

$$SUM_i = \langle U^O, U^S, U^E \rangle$$

This definition is useful in order to develop the mechanism for incremental aggregation of user information.

MAPPING USER FEATURES IN SEVERAL DOMAINS
From the *SUM* definition, we propose a methodology that can be applied to both learn user features from user information stored in recommender systems and deliver the user features to other recommender systems. In order to use the *SUM* in several application domains, we first define the user model *(UM)* in a given existing application domain $_i$ as follows:

$$UM_i = \{AD_i, AI_i, AU_i\}$$

Where *AD* is the set of domain characteristics, *AI* is the set of user interests and *AU* is the set of socio-demographic features of the user *i* required by the application. Then, we establish a relationship between the *SUM* and the UM_i by means of a weighted graph, *G (SUM, UM$_i$)*. This graph connects *SUM* user features with particular user features

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

required in the application domain UM_i. In particular, *SUM* emotional features modify the weights used on the graph according to the emotional state of the user (For more details see [4]). The methodology is based on the combination of machine learning methods: inductive methods (generalization) and deductive methods (specialization). For details on *SUM* management see ([5]). Therefore, instead of making the user fill out the *UM* of each application, we shift information from and to *UM*s of different domains according to the graphs that are defined by each application.

MULTI-AGENT FRAMEWORK

We exploit the synergy between the flexibility of multi-agent systems and the learning capabilities of smart adaptive systems in order to develop a *Multi-agent Smart User Model*. Our approach to user modeling includes the *interoperability* and *coordination* [3] of several autonomous agents with an incremental learning process based on Support Vector Machines [2]. Our framework of *Multi-agent Smart User Model* is able to provide information about the user when a new application in the environment requires it (reactivity); it is able to search for new applications in which the user could be interested (pro-activity); and it can interact with other user models to obtain recommendations in a collaborative way [7]. It is based on two groups of agents: The *Web Service Agents* group *(WSA)* and the *Ubiquitous Agents* group *(UA)*. The WSA provides autonomy capabilities regarding automatically finding services in a specific domain. The UA provides initialization, identification, interoperability, control, coordination, management and storage of the user preferences allowing flexible and autonomous human-agent interaction. The UA integrates a new generic and portable user model that works in accordance with [10] and our *SUM* definition. Coordination between the WSA and UA is established mainly by two mechanisms. First, the WSA requests personalized information from the UA in order to deal with the recommender systems in the environment. Second, the UA receives information from the WSA regarding the success or failure of the application interaction. This relevance feedback is used by the UA to learn about the user's interests, so the corresponding *SUM* and the graph *G (SUM, UMi)* of the application is updated.

CONCLUSIONS AND FUTURE WORK

The next generation of open environments will use *Smart User Models*, which include, among other attributes, the emotional factor [6] of the human being who they represent. The implementation of the *Multi-agent Smart User Model* makes transferring knowledge feasible (i.e. user preferences) from one domain, in which the user has already been profiled, to another, with which the user has never interacted before. The methodology developed can be used to learn user features from user information stored in recommender systems, and deliver the user features to other recommender systems. We are currently testing our hypothesis on the use of kernel-based methods [12] in order to construct automatic mapping of user features into the high-dimensional feature space of several domains. We think that in the near future our model will provide a rich workbench to test learning methods (acquisition and information shifts of user features) in open environments.

ACKNOWLEDGMENTS

This research project has been supported by the Spanish project DPI2001-2094-C03-01 of the Science and Technology Ministry (MCYT).

REFERENCES

1. C. Angulo, and A. Català, Online Learning with kernels for Smart Adaptive Systems: a review, In Proc. European Network for Intelligent Technologies , Oulu, Finland. 2003.

2. C. Angulo, X. Parra, and A. Català. K-SVCR. A Multiclass Support Vector Machine. *Neurocomputing. Special issue: Support Vector Machines*, 55(1/2):57–77, 2003.

3. FIPA. http://www.fipa.org/specs/fipa00001/

4. G. González, B. López, and J. de la Rosa. Managing Emotions in Smart User Models for Recommender Systems. In *Proc. of ICEIS'04*. pp. 187–194, Porto, Portugal, April 2004.

5. G. González, B. López, and J. de la Rosa. Smart User Models for Tourism: A Holistic Approach for Personalized Tourism Services. *ITT Information Technology Tourism Journal*, 6(4):273–286, March 2004.

6. G. González, B. López, and J. L. de la Rosa. The Emotional Factor: An Innovative Approach to User Modelling for Recommender Systems. *Workshop on Recommendation and Personalization in e-Commerce.*, pp. 90–99, Málaga, (Spain), May 2002.

7. N. Good, J. Schafer, J. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of AAAI*, volume 35, pp. 439–446. AAAI Press, 1999.

8. A. Kobsa. Generic User Modelling Systems. *User Modelling and User-Adapted Interaction (UMUAI)*, 11:49–63, 2001.

9. M. Montaner, and et.al. *IRES: On the Integration of Restaurant Services.* AgentCities Agent Technology Competition: Special Prize, Barcelona (Spain), 2003. Available at: http://arlab.udg.es/GenialChef.pdf.

10. P3P. http://www.w3.org/TR/2002/REC-P3P-20020416/

11. S. Willmott. Deploying Intelligent Systems on a Global Scale. *IEEE Intelligent Systems*, 19(5):71–73, Sept-Oct 2004.

12. T. Zhang and V. Iyengar. Recommender Systems Using Linear Classifiers. *Journal of Machine Learning Research*, 2:313–334, March 2002.

Personalized Product Recommendations and Consumer Purchase Decisions

Gerald Häubl
School of Business
University of Alberta
(780) 492-6886
Gerald.haeubl@ualberta.ca

Kyle B. Murray
Richard Ivey School of Business
University of Western Ontario
(519) 661-4210
kmurray@ivey.uwo.ca

ABSTRACT
We discuss some of the key findings from an ongoing research stream that focuses on the effects of recommendation agents ("RAs") – electronic decision aids that generate personalized product recommendations – on buyer behavior in online shopping environments. Consumers who rely on agent assistance can drastically reduce their search effort while, at the same time, improving the quality of their purchase decisions. However, when consumers rely on such an agent, they also become susceptible to being influenced by it. Consequently, RAs are "double agents" that act on behalf of the buyer and the seller to both improve and influence buyer decision making. We provide a brief overview of the major findings from this stream of research, and we discuss the implications of this work for building better recommender systems.

Categories and Subject Descriptors
H.1.2 [**Models and Principles**]: User/Machine Systems – *human information processing, human factors, software psychology.*

H.4.2 [**Information Systems Applications**]: Types of Systems – d*ecision support.*

General Terms
Algorithms, Management, Performance, Design, Economics, Experimentation, Human Factors, Theory.

Keywords
Recommendation agents, recommender systems, human-computer interaction, consumer psychology, economics, search, human decision making, influence, consumer behavior, e-commerce.

1. INTRODUCTION
Electronic shopping environments on the Internet are capable of providing consumers with a volume of relevant information that poses a severe challenge to the capacity limits of human information processing [1] – whether those limits are in memory, attention, motivation, or elsewhere. Software tools that assist consumers in filtering and organizing information into more digestible amounts and formats represent a response to this challenge. Many of these tools are altruistic in the sense that they have no vested interest in what the user does with that information; i.e., the tool is an end unto itself. Often these tools exist simply to assist buyers who are trying to make purchase decisions in information-intensive marketplaces.

However, not all such decision aids are necessarily altruistic and, in fact, many of these tools are designed to not only assist buyers, but to also influence the very choices they make. Such tools are *double agents* – in addition to helping the buyer make a better choice, the agent works for the seller to influence the buyer's purchase decision.

Our research has focused on one important type of electronic decision aids for consumers: recommendation agents ("RAs") that provide consumers with personalized product recommendations. We conceptualize an RA as a decision aid that (1) attempts to understand a shopper's subjective preference in terms of multiple product attributes based on an initial preference-elicitation phase, and (2) provides recommendations in the form of a sorted list of products based on its understanding of the consumer's preference [2].

2. MAJOR FINDINGS
The major findings from this stream of research can be summarized as follows. First, there are several potential benefits to a consumer of using RAs, including reduced search costs and improved decision quality. In particular, it has been demonstrated that the use of RAs tends to enable consumers to make better purchase decisions than they would otherwise, and to do so with less effort than would be required when shopping without agent assistance [2].

Second, consumer preferences are susceptible to being influenced by an RA in a systematic manner. Although, a number of different psychological mechanisms may underlie the influence that a recommender system has on shoppers [3], our research has focused on three: (1) associative feature-based priming, (2) effects due to the format of information presentation, and (3) consumer inferences about attribute importance based on conversational logic. These mechanisms range in the level of consciousness at which they operate – from the unaware to conscious information processing.

Third, some of the effects of using an RA persist beyond situations where the agent is available, while others are only evident when the agent is present. The impact of an RA that is selective in which attributes it considers during preference

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

elicitation on consumer preference can persist into future decision making environments in which the agent is no longer present. On the other hand, our evidence indicates that the benefits of using an RA (i.e., reduced search effort and improved decision quality) do not persist beyond situations in which consumers actually use the agent.

3. BUILDING BETTER AGENTS

We argue that electronic RAs can play a dual role by both assisting and influencing consumer decision making. We believe that this perspective leads to a deeper, and more comprehensive, understanding of the interaction between electronic agents and consumers, which in turn can lead to the design of better and more effective RAs. Anecdotal evidence suggests that consumers in the bricks-and-mortar world are willing to accept some degree of influence or bias from human sales agents in exchange for the benefits that come with the latter's advice. There is no reason to believe that consumers are not equally willing to accept the dual role played by an electronic RA if they are able to benefit from their interaction with such a tool.

The consumer-centric double-agent perspective outlined here provides a framework for the design of RAs that focuses on delivering value to consumers by accelerating their decision processes, while at the same time improving the quality of the choices that they make. However, this perspective also takes into account the competitive advantage gained by a seller who delivers the benefits of agent-assisted shopping and improves its own position in the marketplace through its electronic agent's influence on consumer preferences.

It is important to note that, although we have discussed only one type of influence in this article, other types of computer-human influence have also been demonstrated [3]. For example, subtle contextual cues can prime (i.e., predispose) consumers towards some alternatives and away from others [4], an RA's selectivity in terms of the product attributes that it takes into account can affect consumers' choice processes [5], and the degree of match between agent advice and consumers' pre-existing attitudes can determine the extent to which consumers rely on an agent's recommendations [6]. It has also been demonstrated that lowering search costs for quality information can in some cases reduce consumer price sensitivity [7], and in others increase consumer price sensitivity [8].

These types of influence are relatively subtle and do not necessarily cause the consumer to make a poor purchase decision. On the contrary, each of these types of influence is compatible with improved consumer decision making. The electronic agent's influence can simply sway the buyer towards one good product over another. The conclusion for both buyers and sellers is clear: well-designed electronic RAs can, and should, play a more prominent role in improving the overall value of online shopping.

4. CONCLUSION

The stream of research reviewed here is based on numerous controlled experiments (conducted both in the lab and via the Internet) aimed at examining the impact of recommendation systems on shopping behavior, as well as the psychological mechanisms that underlie these effects. The following list of references includes work cited in this brief review, as well as other relevant work (including working papers) by the authors.

5. REFERENCES

5.1 Cited Work

[1] Alba, J., Lynch, J.G., Weitz, B., Janiszewski, C., Lutz, R., Sawyer, A., and Wood, S. (1997). Interactive Home Shopping: Consumer, Retailer, and Manufacturer Incentives to Participate in Electronic Marketplaces. *Journal of Marketing*, *61* (July), 38-53.

[2] Häubl, G. and Trifts V. (2000). Consumer Decision Making in Online Shopping Environments: The Effects of Interactive Decision Aids. *Marketing Science*, 19, 1, 4-21.

[3] Fogg, B.J. (2002). *Persuasive Technology: Using Computers to Change What We Think and Do*. San Francisco: Morgan Kaufmann Publishers.

[4] Mandel, N. and Johnson, E.J. (2002). When Web Pages Influence Choice: Effects of Visual Primes on Experts and Novices. *Journal of Consumer Research*, 29, 2, 235-245.

[5] Häubl, G. and Murray, K.B. (2003). Preference Construction and Persistence in Digital Marketplaces: The Role of Electronic Recommendation Agents. *Journal of Consumer Psychology*, 13, 1&2, 75-91.

[6] Gershoff, A.D., Mukherjee, A., and Mukhopadhyay, A. (2003). Consumer Acceptance of Online Agent Advice: Extremity and Positivity Effects. *Journal of Consumer Psychology*, 13, 1&2, 161-170.

[7] Lynch, J.G. and Ariely, D. (2000). Wine Online: Search Cost and Competition on Price, Quality, and Distribution. *Marketing Science*, 19, 1, 83-103.

[8] Diehl, K., Kornish, L.J., Lynch, J.G. (2003). Smart Agents: When Lower Search Costs for Quality Information Increase Price Sensitivity. *Journal of Consumer Research*, 30, 1, 56-71.

5.2 Other Relevant Work

[9] Häubl, G. and Murray, K.B. (2004). The Double Agent: Potential Benefits and Pitfalls of an Electronic Agent's Recommendations. working paper. University of Alberta.

[10] Murray, K.B. and Häubl, G. (2004). Processes of Preference Construction in Agent-Assisted Online Shopping. In: Haugtvedt, C., Machleit, K. and Yalch, R. (Eds.), *Online Consumer Psychology: Understanding and Influencing Behavior in the Virtual World*, Mahwah: Erlbaum.

[11] Häubl, G., Dellaert, B.G.C., Murray, K.B., and Trifts, V. (2004). Buyer Behavior in Personalized Shopping Environments: Insights from the Institute for Online Consumer Studies. In: Karat, C.-M., Karat, J., and Blom, J. (Eds.), *Designing Personalized User Experiences for E-Commerce*. New York: Kluwer.

[12] Häubl, G., Murray, K.B., and Trifts, V. (2003). Personalized Product Presentation: The Influence of Electronic Recommendation Agents on Consumer Choice. In: Pal, N. and Rangaswamy, A. (Eds.), *The Power of One: Gaining Business Value from Personalization Technologies*. Victoria: Trafford.

[13] Häubl, G. and Dellaert, B.G.C. (2004). Consumer Product Search With Personalized Recommendations. working paper. University of Alberta.

[14] Godek, J. and Murray, K.B. (2004). The Effects of Rational and Experiential Processing on Preferences for Product Choice Modes. working paper. University of Oregon.

Toward a Personal Recommender System

Bradley N. Miller
Computer Science Dept.
Luther College
Decorah, IA 52101
bmiller@luther.edu

ABSTRACT

This position paper is an outgrowth from work reported in a *Transactions on Information Systems* article entitled PocketLens: Toward a Personal Recommender System, July 2004, in which we propose and compare several architectures for a decentralized recommender system built on top of peer-to-peer infrastructure. In this paper we review the need for personal recommender systems and propose the deployment of personal recommender systems using existing RSS and weblog technologies as the underlying communication infrastructure.

Keywords

Recommender Systems, Collaborative Filtering

INTRODUCTION

As I look at the applications I use on my personal computer each day, more and more of them are becoming net-enabled. My browser shows me web pages, but I am also able to synchronize my bookmarks between the browser on my home machine and my browser at school. My recipe program allows me to publish a personal recipe collection to the web, and is savvy enough to import recipes found on websites published by other people using the same software. My RSS news reader keeps me up to date on hundreds of topics I am interested in reading about. My iTunes music player allows me to sample, purchase, and share music with other members of my family. Version tracker software alerts me to new releases of my favorite applications.

Each of these applications would be enhanced by recommender features. Just as Usenet news used to overwhelm me with new articles each day, my RSS client contains more new headlines than I could ever hope to read. I would love to discover new artists and albums to purchase through iTunes. I would be very pleased to learn about new and interesting recipes that I should make, and I would like to find out about additional software that I could use.

In this paper I propose that the next wave of recommender systems should be integrated with many of the client programs we use for work and play each day. Further, I propose that the engine that drives the recommendations can and should be located on the user's desktop, close to the applications it serves. In [3] we argue that there are two primary issues, *trust* and *control*. In this paper I will summarize our arguments in favor of personal recommender systems, and then briefly present an architecture that would allow for recommenders to run on every desktop.

When a website collects personal information, the user must trust the site to protect the information appropriately. Foner identifies five ways that this trust might be misplaced [2]: **Deception by the recipient**, a site could set out to deliberately trick users into revealing personal information. **Mission creep** refers to a situation in which an organization begins with a clearly defined use for personal data, but expands the original purpose over time. **Accidental disclosure** happens when a website accidentally makes private information available. **Disclosure by malicious intent** happens when personal information that you entrust to an organization is stolen. Finally, information is sometimes released because of **subpoenas**. Even though an organization may take great care to protect personal information, they are obliged under the law to disclose that information when they are subpoenaed.

Most websites try to alleviate their customer's fears by posting a privacy policy on their site that describes how the merchant will use information provided by the customer. However, over time, merchants may rethink their policies, or abandon them altogether for a variety of reasons. This was the case with defunct Toysmart.com. When Toysmart ceased operations in May 2000, the company was forced to sell its assets, including its customer list and customer profiles. In addition, some of the largest websites, like `Yahoo.com` have altered their privacy policy to allow them to sell their customers' email addresses in order to add additional sources of revenue.

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

The key issue highlighted in these scenarios is that once merchants have control of users' personal information, the users can no longer control the uses to which that information is put. These are exactly the kind of control issues that the users in Ackerman, Cranor, and Reagle's privacy study were concerned about [1]. User control of personal information is a key trust issue for recommenders.

A second key trust issue is the reliability of the recommendations. The credibility of Amazon.com's recommendation services was called into question in 2002 when it admitted to using 'faux recommendations' in order to drive business to its new clothing store partners. The 'faux recommendations' were presented right next to the traditional recommendations for other products, but were not based on a customer's past shopping behavior or preferences. How can users know whether to trust the recommendations that are made?

In summary there are two main concerns with centralized recommenders. First, a centralized recommender might share personal data in inappropriate ways. Second, a recommender owned by a commerce site might make recommendations that serve the site, not the user. Both concerns can be met by a *personal recommender*. The personal recommender holds the user's personal data locally, and only shares data the user explicitly identifies as sharable. The personal recommender is software owned by the user and running on the user's local machine. It answers to noone except the user. My long-term vision is to develop such a personal recommender.

A PERSONAL RECOMMENDER ARCHITECTURE

PocketLens is an algorithm that is designed to support personal recommenders. In [3] we introduce PocketLens and five reference architectures that use the algorithm. The PocketLens algorithm is a variant of the accurate and efficient item–item algorithm [4], with three key features introduced that would allow the algorithm to operate on the user's local machine. First, we have modified the algorithm to construct the model incrementally. Second, the PocketLens model is small, since it is only for one user, which allows us to use the algorithm on desktop computers and palmtops. Third, to preserve privacy, the PocketLens algorithm has the property that none of ratings are saved in conjunction with anything that would identify the user they came from.

The PocketLens algorithm divides the recommendation process into two parts. In part one PocketLens searches the network to find ratings to use in building a similarity model. Ratings may come from randomly selected users, or may come from a defined set of users that the algorithm has learned are 'good' over time. Much of the discussion in [3] is devoted to exploring five peer-to-peer architectures that support the sharing of ratings and discovery of 'good' users. In part two the similarity model is used to make recommendations to the user.

Although the PocketLens algorithm has been implemented and tested in the lab, it has not yet been deployed in an actual peer-to-peer environment. Part of the reason for the delay in deployment is because up until recently there has been no widely available infrastructure to use in publishing a set of ratings. The popularity of RSS (http://blogs.law.harvard.edu/tech/rss) syndication technologies has led me to believe that RSS would provide just such an infrastructure. RSS is an XML standard that allows a user to publish 'channels' of information. Each channel can have many items, and each item is represented by a set of attributes including a globally unique identifier (guid), a title, a description, and a URL.

With RSS as the backbone of a ratings exchange system, users could rate new items and publish the ratings just as a user would add an entry to their weblog. To help new users find sources of ratings, individual users may also choose to publish their list of links to other RSS feeds of ratings.

Although the process of publishing ratings may sound complex, there are already many tools available that would enable users to publish their ratings like a weblog. Some of these tools include local weblog servers like Blosxom and WordPress or web based services, with local APIs like Radio Userland and MoveableType. The fact that these technologies are already in place should help speed user adoption because there is no need to introduce yet another set of technologies into the user's system.

There are many exciting applications being developed today. Some of them are already taking their first steps toward adding recommendation capabilities using a rule based approach. For example iTunes provides 'smart playlists' that allow you specify a set of criteria including the artist, genre, or play count to create a custom list of songs to play. iTunes and others also allow users to rate songs on a scale from 1 – 5 stars. NetNewsWire and others provide 'smart list' capability to do keyword matching on new articles. Personalized recommenders could take all of these applications to new levels, while at the same time opening up recommender systems research in exciting new content areas.

REFERENCES

1. M. S. Ackerman, L. F. Cranor, and J. Reagle. Privacy in e-commerce: Examining user scenarios and privacy preferences. In *ACM Conference on Electronic Commerce*, pages 1–8, 1999.
2. L. Foner. *Political Artifacts and Personal Privacy: The Yenta Multi-Agent Distributed Matchmaking System*. PhD thesis, Massachusetts Institute of Technology, 1999.
3. B. N. Miller, J. A. Konstan, and J. Riedl. Pocketlens: Toward a personal recommender system. *ACM Trans. Inf. Syst.*, 22(3):437–476, 2004.
4. B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International World Wide Web Conference (WWW10)*, Hong Kong, May 2001.

Beyond Idiot Savants: Recommendations and Common Sense

Michael J. Pazzani
Department of Informatics
School of Information and Computer Science
University of California, Irvine
Irvine CA 92612
pazzani@ics.uci.edu

ABSTRACT
The current generation of recommendation systems exhibits little if any common sense. While adept at finding patterns in purchase data, such systems are plateauing well below the goal of having intelligence agents be analogous to human concierges.

Keywords
Recommendation systems, personalization, intelligence.

INTRODUCTION
In an early paper on intelligent agents, Etzioni and Weld [1] argued that intelligent agents should be like concierges. However, today's deployed recommendation systems fall short of this goal and are more like idiot savants in that they rely on one or a small number of exceedingly narrow algorithms.

THE PROBLEM
Amazon.com recently recommended that I purchase an 8-inch lid because I had purchased a 10-inch skillet as illustrated in Figure 1. A system that recommends an 8-inch lid for a 10-inch skillet is missing the common sense of sales clerk on his or her first day on job. The problem is not restricted to e-commerce sites, for example the San Diego Union Tribune wireless news site has deployed a content-based recommendation system [2] that recently recommended that I read this story: "Lowe will start Game 7 for Red Sox" because I had recently read a similar story: "Lowe resurrects career with Game 7 gem." Although the two stories have many words in common (i.e. have very similar term vectors), a system that that would recommend an earlier story because one had read the later story is clearly not exhibiting common sense.

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

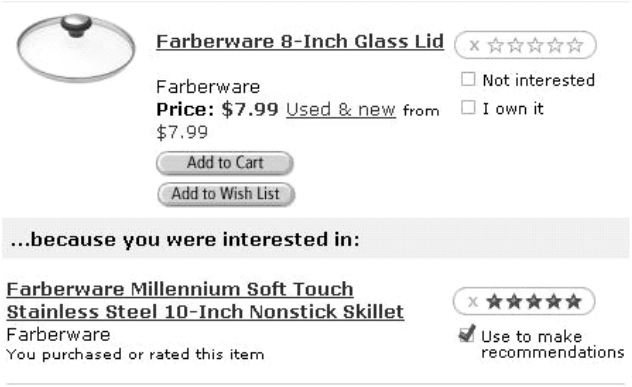


Figure 1. Recommending a 8-inch lid for a 10-inch skillet.

Of course, one could construct a scenario in which these recommendations do make sense. For example, if one had a complete set of Faberware and damaged a skillet and a lid while moving, it would make perfect sense to suggest purchasing two items of different sizes. However, today's recommendation systems don't even have the common sense to construct such scenarios.

HYBRID ALGORITHMS AREN'T THE SOLUTION
Many have recognized the shortcomings of individual recommendation algorithms (e.g. [3-5]) and have advocated hybrid algorithms combining two or more algorithms in the hopes that the strengths of one complement the weaknesses of the other. However, none of the hybrid systems have attempted to encode common sense knowledge, i.e., today's recommendation systems don't know what they are doing [6].

The problem is that the recommendation systems do not have the common sense to recognizing the user's goals and relating those goals to the recommendation nor can they explain how the recommendations would help satisfy the user's goals. A good concierge would have these abilities.

Other data mining algorithms share the failure of today's recommendation system. For example, through detailed analysis of sales data a system used by a department store might learn that ski jackets sell better in Colorado than

Florida. However, a system that lacks the common sense to explain why this pattern holds would be of little use when the first store opens in Utah or Hawaii.

GOALS AND COMMON SENSE

Today's recommendation systems, although useful do not even approximate the utility of a concierge. To achieve the next level of intelligence, recommendations systems will need at least three capabilities missing from most of today's deployed systems:

- Understanding the user's goals, whether stated by the user or inferred from the user's behavior.
- Representing common sense knowledge that indicates how various actions, such as purchasing items or obtaining information, relate to these goals.
- Integrating knowledge and data from disparate sources. Today's systems are almost always deployed within a single web site while the web provides a vast network of independently developed information resources.

An important benefit of embodying recommendation systems with explicit representation of goals is that recommendations agents would be able to interoperate across sites (cf. [7]). A study of Internet usage revealed that users visit on average 10 sites per session (see http://www.it-analysis.com/article.php?articleid=1660) to achieve their goals. An agent that understood the content of such sites would prove valuable in making recommendations by synthesizing information from multiple sources. To give one I example, I recently had business in Australia and decided to take a vacation as well. After exploring options, I decided to plan the vacation after my business rather than before because that would allow me to observe a meteor shower on a moonless night from an island with little light pollution. While I had to search manually to construct this plan, I'd expect an intelligent agent that understood my preferences by analyzing my previous vacations to emulate this decision.

Some initial work along the lines advocated here has shown promise. Kim's work on InfoQuilt [8] has demonstrated that ontologies and semantic web allow for personalization across different information sources. Babaian [9] has shown how a declarative representation of preconditions and effects of a system's actions enhances a personalization system. Lieberman et al. [10] have demonstrated the utility of incorporating common-sense reasoning into a variety of applications ranging from organizing digital photos to personalizing the selection of music.

CONCLUSION

Today's recommendation systems operate for the most part by detection correlations betweens the activity of different users or among the features that describe objects a user likes. However, without explicit representation of user's goals and an explicit representation of world knowledge, such systems lack the ability of a concierge who can explain why there may be correlations in the data and generalize these explanations to new situations.

REFERENCES

1. Etzioni, O &. Weld, D. (1995). *Intelligent Agents on the Internet: Fact, Fiction, and Forecast.* IEEE Expert 10(3): 44-49.
2. Billsus, D., Brunk, C., Evans, C., Gladish, B. & Pazzani, M. (2002). *Adaptive Interfaces for Ubiquitous Web Access*, CACM p 34-38.
3. Burke, R. (2002). *Hybrid Recommender Systems: Survey and Experiments*. User Modeling and User-Adapted Interaction v.12 n.4, p.331-370, November 2002
4. Melville, P. Mooney, R. and Nagarajan, R. (2001). *Content Boosted Collaborative Filtering*. Proceedings of the SIGIR-2001 Workshop on Recommender Systems, New Orleans, LA, September 2001
5. Pazzani, M. (1999). *A Framework for Collaborative, Content-Based and Demographic Filtering*. Artificial Intelligence Review. 13(5-6): 393-408
6. Brachman, Ronald J. (2002). *Systems that Know What They're Doing*, IEEE Intelligent Systems (November/December): 67-71.
7. Kuno, H. & Sahai, A. (2003). *My Agent Wants to Talk to Your Service: Personalizing Web Services through Agents*. In B. Burg, J. Dale, T. Finin, H. Nakashima, L. Padgham, C. Sierra, and S. Willmott, editors, Agentcities: Challenges in Open Agent Environments, pages 25-31. Springer-Verlag, 2003.
8. Kim, W. (2001). *Knowledge-Based Personalization*, M.S. Thesis, Department of Computer Science, University of Georgia
9. Babaian, T. (2003). *Knowledge-Based Personalization*, in Information Management: Support Systems and Multimedia Technology, Idea Group Inc, Hershey, PA
10. Lieberman, H., Liu, H., Singh, P., Barry, B. (in press). *Beating some common sense into interactive applications*. AI Magazine.

Towards More Personalized Navigation in Mobile Three-dimensional Virtual Environments

Teija Vainio
Hypermedia Laboratory, University of Tampere,
Kanslerinrinne 1, 33014 University of Tampere, Finland
teija.vainio@uta.fi

ABSTRACT
Adaptive information systems and adaptive user interfaces are important issues for design when human computer interaction is being studied in large information systems. Three-dimensional virtual environments can be defined as visually emphasized large information spaces. Designing a user interface that adapts automatically according to user's behavior or an interface that adopts according to user's own choices can be seen as an assistance tool for navigation in three-dimensional environments. This paper presents an approach in which adaptive support to three-dimensional virtual environments is provided. This approach is grounded on three basic concepts related to the locating of a user, to the adapting of three-dimensional environment automatically, and the adopting environment by user. In conclusion, some preliminary guidelines for designers to support a user's navigation in three-dimensional virtual environments are presented.

ACM Classification Keywords
H.5.2 User Interfaces: Graphical user interfaces (GUI), and H.5.1 Multimedia Information Systems: Artificial, augmented, and virtual realities

INTRODUCTION
Three-dimensional (3D) virtual environments have been used for several years to visualize information. On the other hand, the problems of using large information systems have been one major cause for the development of adaptive information systems and user interfaces. It is not possible to offer "all things to all" in large information systems, and a diversity of users should be taken into account In applications utilizing 3D virtual environments, navigation is one of the key usability challenges from the users' point of view. Navigation in 3D environments has been studied widely, e.g., [5], but as it is argued here, research of adaptive 3D environments that aim to support users to navigate better is not as abundant. Some projects, primarily related guides for tourists [3] already exist, but challenges of designing adaptive 3D environments for mobile users are not widely known. In addition, in mobile applications one focus is turning to more personalized and more context-aware applications than, for example, in adaptive information systems for wired applications.

Navigation is a process of moving through an environment There are different kinds of navigation; it can e.g., be goal-directive or explorative [5]. Way finding is an essential part of navigation. The tasks of way finding can be categorized as naïve search, primed search, and exploration. Purposeful movement during navigation improves with increased spatial knowledge of the environment. Spatial knowledge can be described as three-level information: landmark knowledge, procedural knowledge and survey knowledge. [5] It is stated that most of the users are not willing to spend too much effort in navigating in the environment [3]. An adaptive user interface can be defined as "a software artifact that improves its ability to interact with a user by constructing a user model based on partial experience with that user" [7]. Adaptive user interfaces can be focused, e.g., to a task of information or content-based filtering, a task of recommendation, a task of social or collaborative filtering, and a task of optimizing. [8] Designing adaptivity and adaptability is based on a theoretical model about the user's behavior. A user model describes what is known about the user and the user's interests. [7]. Furthermore, the user's interests can be classified, e.g., the users' short-term interests (such as current tasks), long-term interests that are stable (such as work or a learning path) and to hybrid interests, which include both of these. [8]

TOWARDS MORE PERSONALIZED NAVIGATION
One solution to develop adaptability and support navigation in 3D environments is to use personal agents, e.g., intelligent virtual agents.[6] In this paper the main focus is on finding navigation assistance from the 3D environment itself, not from additional tools. In previous studies it has been stated that with agents like numens, which are bound to a user, and *genius loci*, which are related to interaction areas or locus, a user's interaction can be observed and adapt the environment to his/her needs. [2] Approaches of this kind are directed mainly for wired users. In case when the 3D environment is displayed in a small screen for mobile users, first of all, finding a space enough to display a recognizable agent is not easy, and furthermore, the com-

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

munication traffic needed between a user, a genius loci, and a numen [see 2] can overload the network connections and cause extra delays in using of the system. Recent studies have also suggested that it would be useful if 3D systems were able to adjust the elements of a user interface based on tracking of the position of users. In addition, target sizing, target positioning and movement angles are the key issues for navigation. [6]

This study aims to continue in this direction by defining additional factors that are properties of 3D environments themselves. In this paper, two basic assumptions are made. The first one is that the location of a user is known and log data of his/her movements is saved. The other assumption is that the task of navigation is way finding. It is also argued here that the distance of the actual user's location and the location of the objective is being searched for is the most essential factor when adaptability is to be implemented in 3D environments. Utilizing the location of a user implicates context-awareness mobile applications. The 3D environment should somehow visually change while the user is approaching the object he/she is searching for. This can be done in several ways, e.g. by creating symbols or changing the brightness of the objects in the 3D environment. It is argued here that the system should automatically adapt if the navigation and way finding are more goal-oriented, and therefore the user is able to concentrate on finding something rather than to navigate in the system. The user's speed, position, and orientation seem to be some key issues in mobile navigation systems [1]. Furthermore, the task (e.g. way finding), navigation style (goal-oriented or explorative navigation), manipulation of UI (scalability and zooming), the levels of detail of the 3D model, and user specific issues are seen as categories of the adaptive 3D environments. If way finding is goal-oriented, the relevance scale of objects searched for should be visualized to a user, for example, by changing the brightness of the objects in the 3D environment. Furthermore, if navigation is explorative, a user should be able to adopt a system himself/herself more actively, for example, by downloading his/her own preferences, such as his/her *personal navigation style* or his/her *personal landmark setting* in the system. A user should also be able to use an environment as a client side application, for example, by marking his/her own routes in an environment. It is assumed here that no other data except the user's actual location, orientation and navigation path is gathered actively by the system. The task of navigation in this case is way finding. The guidelines are presented according to two navigation styles: goal-oriented and explorative. The former is more specifically defined when way finding is more like a formal learning situation, and the latter is more an informal learning situation. Based on this distinction, in the goal-oriented way finding the system is more active by adapting the environment, and in the explorative way finding the user has a more active role.

Goal oriented navigation (the system is adapting)

- **Moving speed**: The speed of the user's movement slows down automatically as the user is approaching the target.
- **Manipulation of UI**: The 3D environment automatically changes visually, e.g., by changing the brightness as the user is approaching the target.

Explorative navigation (the user is adopting the system)

- **Moving speed**: The speed of the user's movement should be free.
- **Manipulation of UI**: The user is able to make his/her *personal preferences*, e.g., a navigation style or personal symbols for landmarks, directly to the 3D environment. The system visualizes the routes and places where the user has already been automatically or by the user's remarks.

REFERENCES

1. Baus, J., Krüger, A., Wahlster, W.,: A Resource-Adaptive Mobile Navigation System. In: Proceedings of the 7th international conference on Intelligent user interfaces, San Francisco, USA, ACM Press, 15-22, (2002)
2. Celetano, A., Nodari, M., Pittarello, F., Adative Intrecation in Web3D Virtual Worlds. In Proceedings of the ninth international conference on 3D Web technology, ACM Press, 41-50, (2004)
3. Cheverts, K.,: Mitchell, K., Davies, N., The Role of adaptive hypermedia in a context-aware tourist guide. Communication of the ACM. Vol. 45, no. 5, 47-51, (2002).
4. Chittaro L., Scagnetto I., Is Semitransparency Useful for Navigating Virtual Environments? Proceedings of VRST-2001: 8th ACM Symposium on Virtual Reality Software & Technology, ACM Press, 159-166, (2002).
5. Darken, R. and Sibert, J. Navigating Large Virtual Spaces. International Journal of Human-Computer Interaction, January-March 1996, 8 (1), 49-72, (1996).
6. Grossman, T. Pointing at Trivariate Targets in 3D Environments. In Proceedings of CHI 2004. ACM 2004, 447-454.
7. Langley, P, (1999). User modeling in adaptive interfaces. Proceedings of the Seventh International Conference on User Modeling. Banff, Alberta: Springer, 357-370
8. Macskassy, S. A., Danyluk, A. A., Hirsh, H. (2000). Information valets for intelligent information access. Papers from the 2000 AAAI Spring Symposium on Adaptive User Interfaces. Menlo Park, CA: AAAI Press. At: http://www-csli.stanford.edu/cll/schedule.htm

Issues of Applying Collaborative Filtering Recommendations in Information Retrieval

Xiangmin Zhang
School of Communication, Information and Library Studies
Rutgers University
4 Huntington Street
New Brunswick, NJ 08901 USA
+1 732 932 7500 x 8229
xzhang@scils.rutgers.edu

My interest in the "Beyond Personalization 2005" workshop is the application of collaborative filtering recommendations in web information retrieval systems.

Collaborative filtering recommender systems are normally found in the domains of movies, music, merchant products, restaurants, and USENET newsgroup messages. One common feature of these domains is that the item space is relatively small, compared to the number of documents available on the web. While the collaborative filtering methods may work well for small item spaces, it is difficult to be applied in the web information retrieval. Web information retrieval is a domain with huge amount of items/documents. In addition, the value of each item to the user's interest is up to a particular person who needs it. Although there are some recommender systems for web search, most of these systems are based on content-based approaches, not the collaborative filtering approach [3, 4].

On the other hand, information retrieval is a natural domain for the use of collaborative filtering recommendations. As part of the scientific research process, collaboration in information seeking is a common practice. People often seek recommendations from colleagues or friends for the needed information [2, 5]. As the number of digital documents increases rapidly on the internet, the demand for this collaboration becomes more and more urgent in order to help people find relevant information. Collaborative filtering recommendations should find a good fit in this field.

However, different from other domains, use of the collaborative filtering for information retrieval tasks needs to meet several challenges. In addition to the rating sparsity and ramp-up problems typical in collaborative filtering systems, as pointed out by many researchers [e.g., 1], we also need to address the issues of 1) What to recommend (relevant documents or the knowledge to find and identify the relevant documents) and 2) what kind of users would like to have recommended relevant documents.

We have recently worked on a project to explore the effectiveness of collaborative filtering recommendations for web search tasks. We built a pilot user interface system that can display previous users' relevant search results, as well as the associated search queries. The system can use one of the publicly available Internet search engines as the information retrieval system and allow experimental participants to search on a set of pre-defined search topics. While the project is still on-going, part of our preliminary results show that users prefer more other people's queries to their relevant judgments or relevant items. This implies that when applying collaborative filtering recommendations for web search tasks, the priority of what to be recommended might be given to the search knowledge of finding relevant documents, rather than the documents themselves. The reason may be that the relevance judgment (rating) is so subjective that people trust only their own judgment. This is particularly true with the trained information searchers. A detailed description of this research will be presented elsewhere.

In summary, web information retrieval provides both an opportunity and challenges for applying collaborative filtering recommendations. The issues discussed in this statement may be related to all topics concerned by this workshop. I hope at the workshop I can learn other people's experiences and thoughts, and hope my research can contribute to the filed as a whole.

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

REFERENCES

1. Burke, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction 12,* 4 (2002), 331-370.

2. Karamuftuoglu, M. Collaborative IR: Toward a social informatics view of IR interaction. *Journal of the American Society for Information Science, 49* (1998), 1070-1080.
3. Konstan,J.A., Miller,B.N., Maltz, D., Herlocker,J.L., Gordon,L.R., and Riedl, J. GroupLens: Applying collaborative filtering to USENET news. *Communications of the ACM, 40*, 3 (March 1997), 77-87.
4. Montaner, M., Lopez, B. and Josep Lluis De La Rosa. A taxonomy of recommender agents on the Internet. *Artificial Intelligence Review, 19*, 4 (2003), 285-330.
5. Zhang, X. Collaborative relevance judgment: A group consensus method for evaluating user search performance", *Journal of the American Society for Information Science and Technology*, 53 (2002), 220-235.